Using full parallel Boltzmann Machines for Optimization

Johannes Faassen
Institute for
Applied Mathematics and Computer Science
RWTH Aachen

January 5, 1997

Abstract

It is well known that synchronously full parallel Boltzmann machines do not converge to the minima of the quadratic energy but to a so called pseudo energy. So far the pseudo energy was treated as a defect disturbing the convergence of these Boltzmann machines. Various simulations suggest that convergence can be ensured if one reduces the degree of parallelism. Another approach that we follow in this paper is to map the problem onto the pseudo energy i.e. we directly exploit the dynamics of the full parallel Boltzmann machine. First we prove that minimizing the pseudo energy is NP-hard and derive a simple mapping scheme that allows us to reduce an arbitrary quadratic 0-1-problem to the minimization of the pseudo energy. After that we present some simulation results showing the potential of this approach and compare it to the results obtained by Aarts, Korst, and Zwietering [1, 12].

1 Introduction

A Boltzmann machine $\mathcal{B}$ consists of n neurons which are connected via symmetric weighted links. Each neuron i can be in the state -1 or 1 such that a configuration of $\mathcal{B}$ is determined by the spin vector $x \in \mathcal{S} = \{-1, 1\}^n$ which has in general the energy

$$E_x = -\frac{1}{2}x^T C x + b^T x.$$

C is the weighted adjacency matrix of the underlying graph i.e. C is a real symmetric matrix with zero diagonal. Bruck [3] has shown that minimizing the above energy E over $\mathcal{S}$ can be reduced to minimizing the pure quadratic term

$$\mathcal{H}_x = -\frac{1}{2}x^T W x \tag{1}$$

which is equivalent to the general *min cut* problem in undirected graphs and is therefore NP-hard. So for ease of exposition we restrict our further considerations to the quadratic energy (1) without the linear bias term. But all results can straightforwardly be extended to the case of the general quadratic form.
The dynamics of the *sequential Boltzmann machine* are defined by the *simulated annealing* procedure where Glauber's criterium [6]

$$P(spinflip) = \frac{1}{1+\exp\left(\frac{\Delta\mathcal{H}}{T}\right)}$$

is used for the probabilistic *spin flip* and at most one neuron gets the possibility to flip at one time. This leads into the following actions at each trial:

1. one neuron is selected at random
2. the selected neuron i flips with probability $A_x(i,T) = \frac{1}{1+\exp\left(\frac{\Delta\mathcal{H}_x(i)}{T}\right)}$
3. any state transition is communicated through the network so that all neurons have up to date information about their neighbors

$\Delta\mathcal{H}_x(i)$ denotes the difference in energy caused by the flip of neuron i in configuration x and is given by

$$\Delta\mathcal{H}_x(i) = 2x_i h_x(i) \tag{2}$$

where

$$h_x(i) = \sum_{j=1}^{n} w_{ij}x_j$$

The sequential Boltzmann machine converges asymptotically to a Gibbs or Boltzmann distribution over the configuration space $\mathcal{S}$

$$\begin{aligned} q_x(T) &= \frac{1}{Z(T)}\exp\left(-\frac{\mathcal{H}_x}{T}\right) \\ Z(T) &= \sum_{x\in\mathcal{S}}\exp\left(-\frac{\mathcal{H}_x}{T}\right) \end{aligned}$$

and as T goes to zero we obtain a uniform distribution over the global minima of $\mathcal{H}$. This means that the sequential Boltzmann machine asymptotically minimizes the energy $\mathcal{H}$ and can therefore be used to solve NP-hard optimization problems [1].
One natural approach to speed up the computation of the Boltzmann machine is to parallelize it's dynamics. This was first done by AARTS and KORST leading into an interesting conjecture. They introduced the *synchronously parallel Boltzmann machine* which will be presented in the following.
The *synchronously parallel Boltzmann machine* is a time discrete model where the state transitions are scheduled in successive trials, each trial consisting of the following steps:

1. each neuron is *activated* with probability α
2. all *active* neurons try to flip independently with probability $A_x(i,T) = \frac{1}{1+\exp\left(\frac{\Delta\mathcal{H}_x(i)}{T}\right)}$
3. all state transitions are communicated through the network so that all neurons have up to date information of their neighbors

The parameter α is a measure for the degree of parallelism in the Boltzmann machine. As α converges to 0 the probability of simultaneous spin flips tends to 0 and the "parallel" Boltzmann machine behaves exactly as a sequential one. If $\alpha = 1$ all neurons are active in each trial and the Boltzmann machine acts in a "full parallel" mode. So far convergence is only proven for the sequential and the full parallel case but up to now the full parallel Boltzmann machine could not be used for optimization problems because it does not converge to the minima of the energy $\mathcal{H}$ any more. Instead it converges asymptotically to a uniform distribution over the minima of the *pseudo energy*

$$\overline{\mathcal{H}}_x = -\sum_{i=1}^{n} |h_x(i)| = -\sum_{i=1}^{n} \left| \sum_{j=1}^{n} w_{ij} x_j \right| \tag{3}$$

which are obviously different to the minima of the energy (1). To use the pseudo energy for problem solving BERTONI et al. tried to decide for which Boltzmann machines the energy and pseudo energy have identical minima [2]. They proved that it is coNP-complete to decide whether the global minima are identical or not, but they could not give any method allowing to exploit this for problem solving in any way.

In this paper we will show that it is also NP-hard to find the minima of the pseudo energy and that an arbitrary quadratic problem can be easily mapped on to the pseudo energy so that the full parallel Boltzmann machine can be used to solve NP-hard optimization problems too. Before we prove that, we present the conjecture of AARTS and ZWIETERING which gives an interesting interpretation of the results.

In [12] AARTS and ZWIETERING proved that the full parallel Boltzmann machine minimizes the pseudo energy $\overline{\mathcal{H}}$. Because the minima of the pseudo energy are different to the minima of the original energy they tried to reduce parallelism such that convergence to the global minima of the energy $\mathcal{H}$ is preserved.

> *Conjecture (Aarts, Zwietering 89)*: Let $\mathcal{B}$ be a synchronously parallel Boltzmann machine with activation level α. Then $\mathcal{B}$ converges asymptotically to the minima of the *extended energy* function:
>
> $$\widehat{\mathcal{H}} = (1-\alpha)\mathcal{H} + \alpha\overline{\mathcal{H}} \tag{4}$$
>
> and there exists a critical α_c such that for all $\alpha < \alpha_c$ the minima of $\widehat{\mathcal{H}}$ are the same as the minima of $\mathcal{H}$.

The original pseudo energy found by AARTS and KORST differs from the energy presented above because they have done their calculations with a $(0,1)$ Boltzmann machine including the bias term. In the next section we prove that the synchronously full parallel $(-1,1)$ Boltzmann Machine minimizes asymptotically the pseudo energy (1) which is the same PERETTO has found in the related Little model [8, 9]. The proof we show here is nearly the same one can find in [1]. But because of a signum error in their exposition and the central role for all the following results we show it again for this special case.

2 Pseudo energy

The fixed temperature behavior of the Boltzmann machine can be modeled in a finite homogeneous Markov chain over the configuration space $\mathcal{S}$. Let $X(i)$ be a stochastic variable denoting the outcome of the i^{th} trial, then the transition probability at the i^{th} trial for each pair x,y of outcomes is defined as

$$P_{xy} = Prob\{X(i) = y | X(i-1) = x\}.$$

The Markov chain corresponding to the Boltzmann machine is irreducible and aperiodic so that it is sufficient to verify the detailed balance equation (5) to prove convergence to a given distribution q.

$$\forall x, y : q_x P_{xy} = q_y P_{yx} \tag{5}$$

(q_x is the probability to be in state x.)
To do so we first have to evaluate the probabilities P_{xy}.
Let for each pair of configurations $x, y \in \mathcal{S}$, $\mathcal{U}_{xy}$ be the set of neurons that have to change their states in order to transform x into y, i.e.

$$\mathcal{U}_{xy} = \{i \mid x_i \neq y_i\}$$

The probability of neuron i to flip at temperature T is

$$A_x(i,T) = \frac{1}{1 + \exp\left(\frac{\Delta\mathcal{H}_x(i)}{T}\right)}$$

where $\Delta\mathcal{H}_x(i)$ is the energy difference due to the flip of neuron i in configuration x.

$$\begin{aligned} \Delta\mathcal{H}_x(i) &= \mathcal{H}_{x(i)} - \mathcal{H}_x \\ &= -\frac{1}{2}\left(x^T(i)Wx(i) - x^TWx\right) \\ &= 2x_i h_x(i) \end{aligned}$$

So we can rewrite the transition probabilities

$$
\begin{aligned}
A_x(i,T) &= \left(1+\exp\left(\frac{2x_i h_x(i)}{T}\right)\right)^{-1} \\
\overline{A_x(i,T)} &= \left(1+\exp\left(\frac{-2x_i h_x(i)}{T}\right)\right)^{-1}
\end{aligned}
$$

and get

$$
\begin{aligned}
P_{xy} &= \prod_{k\in\mathcal{U}_{xy}} A_x(k,T) \prod_{k\notin\mathcal{U}_{xy}} \overline{A_x(k,T)} \\
&= \prod_{k\in\mathcal{U}_{xy}} \left(1+\exp\left(\frac{2x_k h_x(k)}{T}\right)\right)^{-1} \prod_{k\notin\mathcal{U}_{xy}} \left(1+\exp\left(\frac{-2x_k h_x(k)}{T}\right)\right)^{-1} \\
&= \prod_{k} \left(1+\exp\left(\frac{-2y_k h_x(k)}{T}\right)\right)^{-1}
\end{aligned}
$$

The last equality follows easily by the fact that

1. $k \in \mathcal{U}_{xy}$ implies $x_k = -y_k$
2. $k \notin \mathcal{U}_{xy}$ implies $x_k = y_k$

Now we can calculate the stationary distribution via the detailed balance equation.

$$
\begin{aligned}
\frac{P_{xy}}{P_{yx}} &= \frac{\prod_k \left[1+\exp\left(\frac{-2y_k h_x(k)}{T}\right)\right]^{-1}}{\prod_k \left[1+\exp\left(\frac{-2x_k h_y(k)}{T}\right)\right]^{-1}} \\
&= \frac{\prod_k \left[\exp\left(\frac{x_k h_y(k)}{T}\right)+\exp\left(\frac{-x_k h_y(k)}{T}\right)\right] \prod_k \left[\exp\left(\frac{-x_k h_y(k)}{T}\right)\right]}{\prod_k \left[\exp\left(\frac{y_k h_x(k)}{T}\right)+\exp\left(\frac{-y_k h_x(k)}{T}\right)\right] \prod_k \left[\exp\left(\frac{-y_k h_x(k)}{T}\right)\right]} \\
&= \frac{\prod_k 2\cosh\left(\frac{x_k h_y(k)}{T}\right) \prod_k \left[\exp\left(\frac{-x_k h_y(k)}{T}\right)\right]}{\prod_k 2\cosh\left(\frac{y_k h_x(k)}{T}\right) \prod_k \left[\exp\left(\frac{-y_k h_x(k)}{T}\right)\right]} \\
&= \frac{\prod_k 2\cosh\left(\frac{x_k h_y(k)}{T}\right)}{\prod_k 2\cosh\left(\frac{y_k h_x(k)}{T}\right)} \exp\left(\frac{1}{T}\sum_k \{-x_k h_y(k) + y_k h_x(k)\}\right) \\
&= \frac{\prod_k 2\cosh\left(\frac{h_y(k)}{T}\right)}{\prod_k 2\cosh\left(\frac{h_x(k)}{T}\right)} \\
&= \frac{q_y}{q_x}
\end{aligned}
$$

From this follows that $q_x \sim \prod_k 2\cosh\left(\frac{h_x(k)}{T}\right)$ and rewriting this expression in the form of the Boltzmann distribution gives

$$
\begin{aligned}
q_x(T) &= \frac{1}{Z_0'(T)} \prod_k 2\cosh\left(\frac{h_x(k)}{T}\right) \\
&= \frac{1}{Z_0'(T)} exp\left[\sum_k ln\left(2\cosh\left(\frac{h_x(k)}{T}\right)\right)\right] \\
&= \frac{1}{Z_0(T)} exp\left[\sum_k ln\left(\cosh\left(\frac{h_x(k)}{T}\right)\right)\right]
\end{aligned}
$$

By using $\ln\cosh(x) \approx |x|$ for large x it follows directly that the parallel Boltzmann machine $\mathcal{B}$ converges in the limit $T \downarrow 0$ to a uniform distribution over the minima of the pseudo energy

$$
\overline{\mathcal{H}}_x = -\sum_{i=1}^{n} |h_x(i)| = -\sum_{i=1}^{n}\left|\sum_{j=1}^{n} w_{ij}x_j\right| \tag{6}
$$

3 Bilinear programming and cuts in bipartite graphs

In this section we will show that minimizing $\overline{\mathcal{H}}$ is equivalent to bilinear programming and that any quadratic problem can be reduced to a bilinear one. This proves that minimizing $\overline{\mathcal{H}}$ is NP-hard and therefore can be used in principle to solve NP-complete optimization problems. Moreover we give an easy mapping scheme allowing us to utilize the full parallel Boltzmann machine for optimization tasks.

The main problem in handling the pseudo energy $\overline{\mathcal{H}}$ is due to the absolute value terms. So the first thing to do is to eliminate them.

$$
\begin{array}{llll}
 & \overline{\mathcal{H}}_x & \longrightarrow \min(x)! & x \in \{-1,1\}^n \\
\Leftrightarrow & \sum_{i=1}^n \left|\sum_{j=1}^n w_{ij}x_j\right| & \longrightarrow \max(x)! & x \in \{-1,1\}^n \\
\Leftrightarrow & \sum_{i=1}^n s_i\left(\sum_{j=1}^n w_{ij}x_j\right) & \longrightarrow \max(x)! & s_i = sig\left(\sum_{j=1}^n w_{ij}x_j\right)\ x \in \{-1,1\}^n \\
\Leftrightarrow & \sum_{i=1}^n s_i\left(\sum_{j=1}^n w_{ij}x_j\right) & \longrightarrow \max(x,s)! & s, x \in \{-1,1\}^n \\
\Leftrightarrow & s^T W x & \longrightarrow \max(x,s)! & s, x \in \{-1,1\}^n
\end{array}
$$

This approach leads to the structural understanding of the pseudo energy. Instead of minimizing the quadratic function $\mathcal{H}_x = -x^T W x$ the parallel Boltzmann machine minimizes asymptotically the symmetric bilinear term $\overline{\mathcal{H}}_{x,s} = -s^T W x$. In general it is easy to reduce any quadratic problem to a bilinear one and vice versa, i.e. in theory quadratic and bilinear programming are equivalent. Every symmetric bilinear problem can be rewritten in the quadratic form

$$\overline{\mathcal{H}}_{x,s} = \binom{x}{s}^T \begin{pmatrix} 0 & W \\ W & 0 \end{pmatrix} \binom{x}{s} \qquad\qquad x, s \in \{-1, 1\}^n$$

and any quadratic problem can be mapped on a bilinear one with the same solution using the penalty function

$$p(x, s) = \sum_{i=0}^{n} c_i s_i x_i \qquad\qquad c_i > \sum_{j=0}^{n} |w_{ij}|$$

which forces that $x = s$, i.e. the function $\mathcal{H}'_{x,s} = -s^T W x - p(x, s)$ has the same minima as $\mathcal{H}_x = -x^T W x$. In terms of the matrix W this would mean that we add c_i to the diagonal elements w_{ii}. But the problem is that we do not have any diagonal elements in this special context because there is no self feedback in a $(-1, 1)$ Boltzmann machine. That is why we have to find another way to reduce the quadratic problem to a the bilinear one. One elegant way for this is to interpret the quadratic programming problem as a *min cut* problem (MC) in weighted undirected graphs. The general quadratic problem

$$x^T A x \longrightarrow \max(x)! \qquad\qquad x \in \{-1, 1\}^n \tag{7}$$

is equivalent [3] to

> *Given*: undirected weighted graph $\mathcal{G} = (V, E)$ with weighted adjacency matrix A
> *Problem*: find a bipartition V_1, V_2 of V with minimal weight, i.e. minimize $\sum_{i \in V_1, j \in V_2} a_{ij}$.

Every spin vector x corresponds in a natural way to a bipartition i.e. all components with the same value belong to the same partition. For the special case of the bilinear programming problem the adjacency matrix looks like

$$A = \begin{pmatrix} 0 & W \\ W & 0 \end{pmatrix} \qquad\qquad \text{with } W = W^T;\ w_{ii} = 0 \tag{8}$$

and we get the following symmetric bipartite graph (see fig.1). It is the same graph BRUCK [3] has used to prove a general convergence theorem for parallel Hopfield networks which are the deterministic counterpart of the Boltzmann machines. Although the bipartite graph is highly symmetric the minimal cuts are not. This reflects the observation that the global minima of the pseudo energy differ from the minima of the quadratic energy causing the convergence problems.

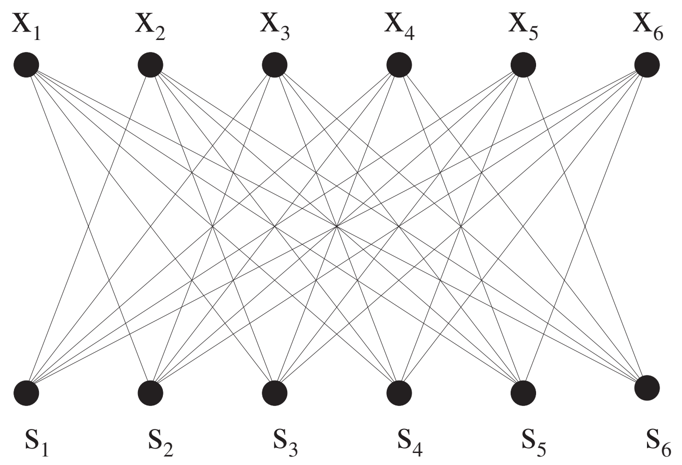

Figure 1: Bipartite graph corresponding to the pseudo energy ($n = 6$). $weight(x_i, s_j) = weight(x_j, s_i)$ and $weight(x_i, s_i) = 0$

As described above we could easily avoid this problem if there would be the edges $\{x_i, s_i\}$ because sufficiently strong links force the vertices x_i and s_i into the same partition and the solution of the quadratic problem would be the same as the solution of the bilinear problem. But due to the missing self feedback this attempt fails and any other modification of the weights could lead to completely different solutions. The usage of a model with self feedback does not solve this problem either because this would result into a different pseudo energy with the same structural problem. Therefore we add a gadget to every pair of vertices preserving the solutions of the quadratic problem but enforcing the solutions to be symmetric i.e. $x = s$ (see fig.2).
Due to the symmetry restrictions we can only choose the weight of 3 edges. It

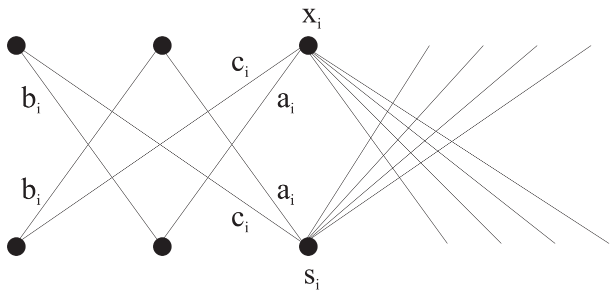

Figure 2: Gadget that enforces $x_i = s_i$

is obvious that sufficiently large weights will ensure that x_i and s_i belong to the same partition. But for some reason which we discuss in the next section it might be better to choose the weights as small as possible. After some reordering of the vertices the gadget becomes planar (see fig.3) and we can see that there must be

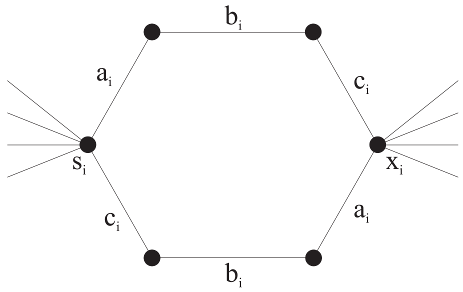

Figure 3: The reorganized gadget

at least two edges in the cut if the vertices x_i and s_i do not belong to the same partition and it is also clear that there are no structural differences between the possible simple cuts. Therefore we choose $K_i := a_i = b_i = c_i$ and still have to decide how large this weight must be. In order to avoid a cut in this cycle the highest possible gain by the outgoing edges must be smaller than the penalty of a cut in the cycle. Obviously it is sufficient to choose

$$K_i > \frac{\sum_{j=1}^{n} |w_{ij}|}{2} . \tag{9}$$

After adding these gadgets, we can be sure that $s = x$ i.e. the solution of the bilinear problem is the same as the solution of the quadratic one. But we still have to show that the gadgets preserve the solution of the original problem and how this construction affects the Boltzmann machine's graph structure.

The bipartite graph was obtained by splitting the graph that represents the Boltzmann machine

$$W \longleftrightarrow \begin{pmatrix} 0 & W \\ W & 0 \end{pmatrix}$$

where W is a symmetric matrix with zero diagonal, the underlying weighted adjacency matrix. Every bipartite graph with the above adjacency matrix structure has a corresponding original graph. Because the gadget preserves the graph structure (it is also symmetric and has no direct links) we have a corresponding gadget in the original graph which is shown in figure 4. From this we can see that the gadget has no effect on the minimal cut because all vertices of the gadget belong to the same partition resulting just in an additive term. If we add this construction to every neuron in the original Boltzmann machine we can guarantee for all i that x_i belongs to the same partition as s_i, i.e $x = s$ and so the solutions of the bilinear and the quadratic problem are identical. By this we have proved that it is NP-hard to find the global minima of the pseudo energy $\mathcal{H}$ and that every sequential Boltzmann machine can be easily transformed into a synchronously full

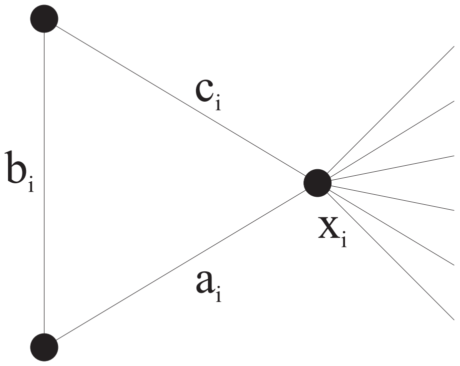

Figure 4: Gadget in the original graph ($a_i = b_i = c_i = K_i$)

parallel Boltzmann machine that converges asymptotically to the same minima. This allows us to use the synchronously full parallel Boltzmann machine in the same way for optimization tasks as the sequential Boltzmann machine. We close this section with a summarizing theorem.

Theorem 1 *Let $\mathcal{B}$ be a sequential Boltzmann machine with connection matrix W and $\mathcal{B}'$ the corresponding synchronously full parallel Boltzmann machine that converges asymptotical to the global minima of the pseudo energy $\overline{\mathcal{H}}$. Then we have:*

i) minimizing $\overline{\mathcal{H}_x}$, $x \in \{-1,1\}^n$ is equivalent to the bilinear programming problem:
minimize $\widetilde{\mathcal{H}}_{s,x} = -s^T W x$ subject to $x, s \in \{-1,1\}^n$.

ii) finding the global minima of $\widetilde{\mathcal{H}}$ is NP-hard.

iii) every sequential Boltzmann Machine $\mathcal{B}$ with n neurons can be transformed into a synchronously full parallel Boltzmann machine $\widetilde{\mathcal{B}}$ with $3n$ neurons that converges asymptotically to the same minima as $\mathcal{B}$.

In the next section we give some interpretations of the above theorem and discuss the relations to other results. We also present simulation results which give a first impression about the achievable speedup and point out some problems with this approach.

4 Interpretation and Simulation Results

In the previous section we have proved that the gadgets preserve convergence of the Boltzmann machine even if it runs in a full parallel mode. But to do so we have only considered the limit distributions of the corresponding Markov chains

giving us no idea how the gadgets really affect the dynamics of the parallel Boltzmann machine.
The problem in the convergence of parallel Boltzmann machines is due to the fact that neighboring neurons can flip simultaneously resulting in higher energy states although both flips separately would have lead into a lower energy. (This is what we call in the following as an erroneous flip.) From various simulations it was well known (but still not proved) that this effect does not take into account if these erroneous spin flips do not happen to often, i.e. if we reduce the noise effect of these spin flips under a certain level then the probabilistic nature of simulated annealing will tolerate their effect and convergence is preserved. This is also the interpretation of AARTS' and ZWIETERING's conjecture. They tried to reduce the noise level caused by the erroneous spin flips by reducing the degree of parallelism. To do so they activate not all but only a fraction α of all neurons in each time step what obviously reduces the probability of collisions. If their conjecture is true then there exists a critical α_c such that for all $\alpha < \alpha_c$ the Boltzmann machine converges correctly.
The gadgets we add to the Boltzmann machine do almost the same. Due to the large weights K_i the two additional neurons damp the activations of the adjacent neuron because it is unlikely that a single neuron in the gadget flips against the others and at least two neurons in a gadget have to flip before the whole gadget stabilizes in the opposite state. Therefore the gadget reduces the effective probability that a neuron will flip at a time what is equivalent to reducing the degree of parallelism. But there are some differences to the former approach.
The first one is that we do not regulate the degree of parallelism globally. Every neuron is as active as its connectivity allows and the level of activation depends only on the outgoing connections.
The second is that in our approach the damping also depends on the temperature, i.e. the temperature decreases as the damping increases. This behavior contradicts the heuristics developed so far. It was argued [4, 7] that in the low temperature regime it is more unlikely that two adjacent neurons flip at the same time than at higher temperatures. Because of that the dynamics of the annealing process would be primary disturbed in the high temperature regime. Our results motivate that this argumentation is not as clear as it seems to be since it does not keep into account that errors made at low temperatures disturb convergence more than the errors made at higher temperatures. If we interpret erroneous spin flips just as errors in the cost evaluation and approximate these errors by a Gaussian distribution with zero mean and variance σ^2 then we know from [5, 10] that the variance of the errors has to go to zero faster than T^4 to preserve convergence, more precisely

$$\lim_{T \to 0} \sigma^2 = o(T^4) \tag{10}$$

has to hold. From this it could be reasonable to increase the damping while lowering the temperature if the natural damping caused by the Glauber dynamics is not enough to hold criterium (10). But despite the hard convergence result for our approach we can not strictly answer that question neither. Up to this point we have only talked about asymptotic convergence. But the only reason for parallelization was to speedup the computation and so the most important question is what speedups we can achieve by this approach. In the following we show some simulation results that give a first impression about the potential gain of that method.

All the computations were made on a sequential computer simulating the time discrete model of the Boltzmann machine. Any communication times are neglected and so the number of time steps corresponds directly to the computation time. For both algorithms, the sequential and the parallel one, we used the same conceptual simple cooling schedule with

$$T_0 = max_{i \in \{1,\ldots,n\}} \sum_{j=1}^{n} |w_{ij}|$$

and

$$T_{k+1} = \alpha T_k.$$

At each temperature T_k the sequential Boltzmann machine tries n spin flips (n is the dimension of the sequential Boltzmann machine) while the parallel version just tries one parallel step. The cooling process is stopped when at 50 consecutive temperatures no state transition was accepted. For the sequential algorithm the cooling parameter α is set to 0.95 which gives good results for the problems considered here. The cooling parameters in the parallel algorithms are chosen such that the final results are comparable to the results obtained by the sequential algorithm. More precisely we adapt α such that the average deviation to the sequential results is less than 5%.

The first problem class we consider here is the *max cut* problem in weighted graphs which is equivalent to the general *min cut* problem that we have described above. Because the graph structure of the Boltzmann machine is the same as the structure of the problem graph this problem is suitable to examine the influence of data dependencies in the Boltzmann machine. The problem instances are randomly generated graphs of certain size. The edges are chosen independently with a probability p such that all vertices have the same expected degree. The weights of the edges are randomly chosen from $\{1, ..., 10\}$.

Figure 5 shows the speedups and the cooling parameters for graphs of size $n = 200$ depending on the connectivity which describes the average degree of the vertices. To reduce statistical errors we have averaged over 10 instances and 5 runs for each instance and every degree. We can see that the speedup is falling rapidly with increasing connectivity. Already at an average degree of 12 the speedup has shrinked to 1 and also at an average degree of 2 we just get a speedup of 60 instead of the ideal 200. This slowdown is due to the fact that we have to increase the cooling parameter in the parallel Boltzmann machine to preserve the quality of the final costs. The reason for this is that the damping of a gadget depends on its edge's weights and the more edges are incident to a vertex the higher the weights in the gadget will be. But the criterium (9) for the edges

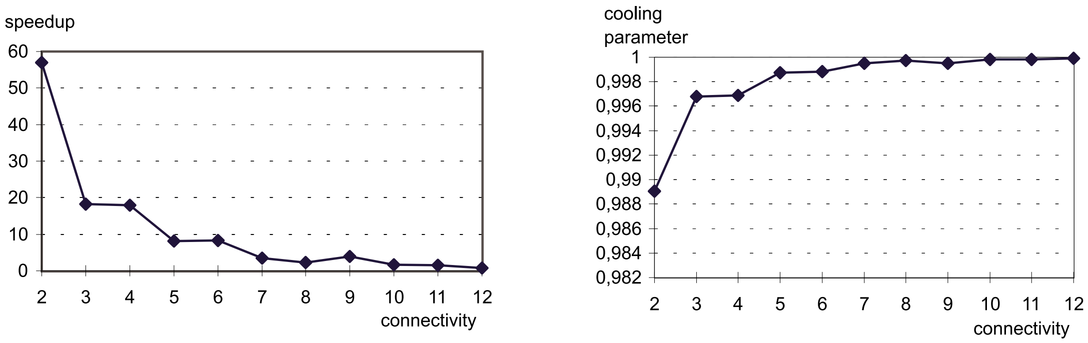

Figure 5: Speedup for the *max cut* problem of size $n = 200$ varying with the average connectivity of the graphs and the cooling parameter for the parallel annealing algorithm

in the gadget came from a very straight and local argumentation. It may be possible that we can do much better and avoid this paralyzing damping by a different choice of the weights. To check out the potential of this idea we try to determine the minimal factor by what the weights in the gadget can be reduced without losing convergence. We calculate this factor depending on the size and the average degree of the vertices for the *max cut* problem. Because we can not prove convergence we accept a certain class of Boltzmann machines with given size, factor, and fixed expected degree as convergent if 10 instances converge in 5 runs. Figure 6 shows the results depending on the size and the connectivity of the graphs. We can see that the factor only depends little on the size but once more massively on the connectivity of the graphs. With increasing size the factor can be reduced slightly in all instances but we think that this is just a finite size effect. More serious is the effect of the connectivity. For graphs with an average degree of 5 we can reduce the weights in the gadget by a factor of 0.48, at an average degree of 100 only by factor of 0.8. Once again we have a problem with the connectivity i.e. the data dependencies in the Boltzmann machine.
Figure 7 shows in the same way as figure 5 the speedup that can be achieved if we reduce the weights in the Boltzmann machine as determined. By reducing the weights we could massively increase the speedup (from 8 to 67 at degree 5 and from 1.5 to 15.9 at degree 10) but again the speedup is falling rapidly while we increase the expected degree and at a connectivity of 20 the speedup vanishes. From figure 8 we can see how the speedup scales with the instance size for the *max cut* problem with reduced gadget weights. In all cases we get a linear speedup with the specific connectivity dependent efficiency. So far the simulation results suggest that reasonable speedups can only be achieved if we

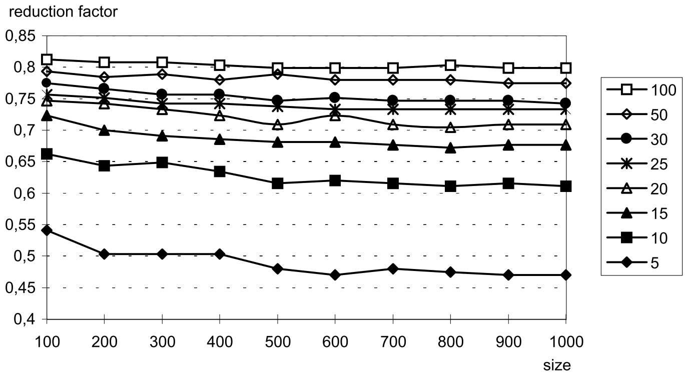

Figure 6: Factors by which the gadget can be lighten depending on the size of the graph and its connectivity

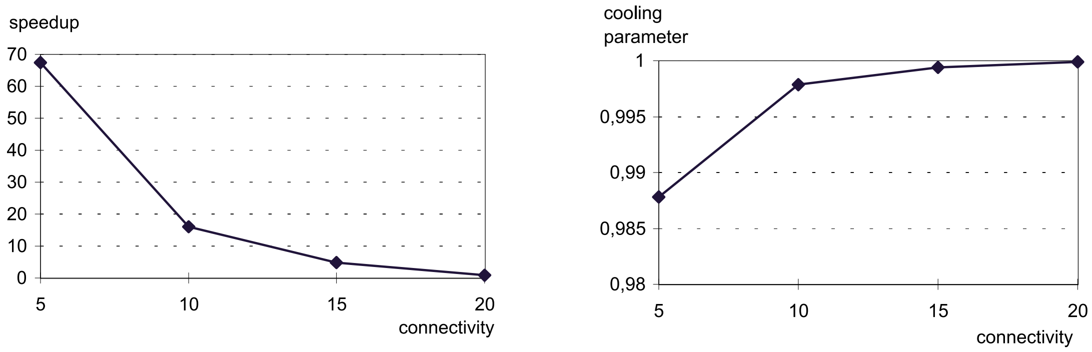

Figure 7: Speedup and cooling parameter for the *max cut* problem of size $n = 200$ varying with the average connectivity of the graph. This time the weights in the gadgets are reduced as determined in figure 6

have very limited data dependencies. But this is only correct if we use the gadget to map an arbitrary quadratic problem on a bilinear one. We have proved that minimizing the pseudo energy is NP-hard and so it might be possible to use the pseudo energy itself without the way around over the quadratic form. Because we could not find any real optimization problem formulated as symmetric bilinear $(-1, 1)$-program we created an artificial one to demonstrate that the connectivity dependent slowdown is just a problem with the gadget construction and is not intrinsic to the parallel Boltzmann machine. We build an artificial problem by choosing every connection in the Boltzmann machine randomly and independent out of the interval $[-5, 5]$. The computing times are compared to the sequential algorithm where we mapped the bilinear problem on a quadratic one via the padding described in section 3. Figure 9 shows the measured speedups. The speedup scales linear with the problem size and the efficiency of 2 is explained by

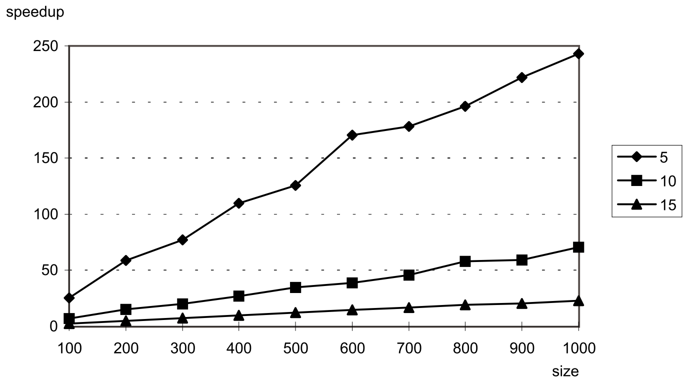

Figure 8: Speedups for the *max cut* problem with reduced gadget weights.

the padding technique. These measurements make it reasonable that it is possible to exploit full parallel dynamics also for highly connected Boltzmann machines.

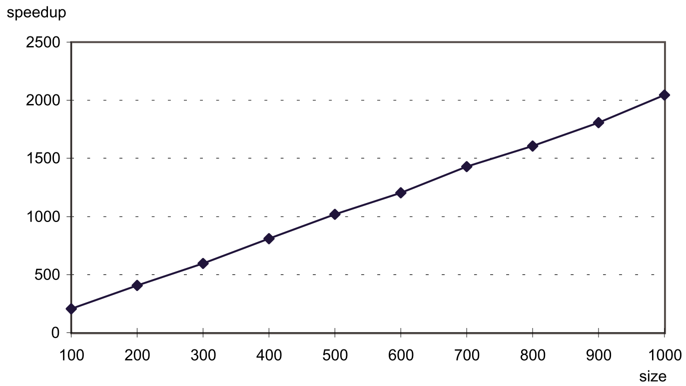

Figure 9: Speedup for the bilinear problem

5 Conclusion

We have proved that minimizing the pseudo energy is NP-hard and therefore that full parallel Boltzmann machines can also be used for NP-hard optimization tasks. To do so we have given a simple and effective mapping scheme to transform an arbitrary quadratic problem into a bilinear one that corresponds to minimizing the pseudo energy. The speedups that we can achieve by this method are only acceptable for limited data dependencies up to an expected vertex degree of 15. But for sparse connected Boltzmann machines our approach is comparable to the parallelization based on parallel updates of independent sets [1] with the

advantage that we do not need a global preprocessing step. The gadget that we use in our approach can be completely computed locally while the task to determine the maximal independent sets is itself an NP-hard optimization problem. So we think that our approach is a real alternative to the method proposed by AARTS et al. [1]. Usually Boltzmann machines are of very limited practical use, especially Boltzmann machines with limited connectivity. But there may be a broad field of real applicability in the field of image processing using the concept of Markov random fields [11]. In these applications we have only local connections that correspond to certain templates.
Another challenging task would be to formulate problems directly in the bilinear philosophy what would allow us to use full parallel Boltzmann machines as a real-time optimization tool.

References

[1] E.H.L. Aarts, J. Korst, *"Simulated Annealing and Boltzmann Machines"*, JOHN WILEY & SONS, 1989

[2] A. Bertoni, P. Campadelli, F. Grassani, *"Full Parallelism in Boltzmann Machines"*, Neuro Nimes 89, 1989, 361-370

[3] J. Bruck, *"On the Convergence Properties of the Hopfield Model"*, Proceedings of the IEEE, vol.78, no.10, 1990, 1579-1584

[4] A.F. Casotto, A.L. Sangiovanni-Vincentelli, *"A parallel simulated annealing algorithm for the placement of macro-cells"*, IEEE Transactions on Computer-Aided Design, 6, 1987, 838-847

[5] S. Gelfand, S. Mitter, *"Simulated Annealing with Noisy or Imprecise Energy Measurements"*, Technical Report, Massachusetts Institute of Technology, Cambridge, MA, 1987

[6] R.J. Glauber, *"Time-Dependent Statistics of the Ising Model."*, Journal of Mathematical Physics 4, 1963, 294-307

[7] D.R. Greening, *"A Taxonomy of Parallel Simulated Annealing Techniques"*, Research Report 66639, IBM T.J. Watson Research Center, Yorktown Heights, New York, 1989

[8] W.A.Little, G.L. Shaw, *"Analytic Study of the Memory Storage Capacity of a Neural Network"*, Mathematical Biosciences 39, 1978, 281-290

[9] P. Peretto, *"Collective Properties of Neural Networks: A Statistical Physics Approach"*, Biological Cybernetics 50, 1984, 51-62

[10] F. Romeo, A. Sangiovanni-Vincentelli, *"A Theoretical Framework for Simulated Annealing"*, Algorithmica, 6, 1991, 302-345

[11] G. Winkler, *"Image Analysis, Random Fields and Dynamic Monte Carlo Methods"*, Springer-Verlag Berlin Heidelberg 1995

[12] P.J. Zwietering, E.H.L. Aarts, *"The convergence of parallel Boltzmann machines"*, Parallel Processing in Neural Systems and Computers, North-Holland Amsterdam, 1990, 277-280