Some Afterthoughts on Hopfield Networks

Šíma, Jiří
1999

INSTITUTE OF COMPUTER SCIENCE

ACADEMY OF SCIENCES OF THE CZECH REPUBLIC

Some Afterthoughts on Hopfield Networks

Jiří Šíma Pekka Orponen Teemu Antti-Poika

Technical report No. 781

May 1999

Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod vodrenskou v 2, 182 07 Prague 8, Czech Republic
phone: (+4202) 6605 3030 fax: (+4202) 85 85 789
e-mail: sima@uivt.cas.cz

INSTITUTE OF COMPUTER SCIENCE

ACADEMY OF SCIENCES OF THE CZECH REPUBLIC

Some Afterthoughts on Hopfield Networks

Jiří Šíma [1] Pekka Orponen [2] Teemu Antti-Poika [3]

Technical report No. 781
May 1999

Abstract

The present paper investigates four relatively independent issues, each in one section, which complete our knowledge regarding the computational aspects of popular Hopfield nets [9]. In Section 1, the computational equivalence of convergent asymmetric and Hopfield nets is proved with respect to the network size. In Section 2, the convergence time of Hopfield nets is analyzed in terms of bit representations. In Section 3, a polynomial time approximate algorithm for the minimum energy problem is shown. In Section 4, the Turing universality of analog Hopfield nets is studied.

Keywords

Hopfield networks, computational power, convergence time, minimum energy problem, analog networks

[1] Department of Theoretical Computer Science, Institute of Computer Science, Academy of Sciences of the Czech Republic. Research supported by GA ČR Grant No. 201/98/0717.

[2] Department of Mathematics, University of Jyväskylä, P. O. Box 35, FIN-40351 Jyväskylä, Finland, e-mail: orponen@math.jyu.fi. Research partially supported by the Academy of Finland.

[3] Department of Mathematics, University of Jyväskylä, P. O. Box 35, FIN-40351 Jyväskylä, Finland, e-mail: anttipoi@math.jyu.fi.

1 A Size-Optimal Simulation of Asymmetric Networks

In his 1982 paper [12], John Hopfield introduced a very influential associative memory model which has since come to be known as the discrete-time Hopfield (or symmetric) network. He has shown that any symmetric network is governed by a bounded Liapunov, or 'energy' function defined on its state space which is properly decreasing along any nonconstant computation path (productive computation). Hence the existence of Liapunov function implies that under sequential update any (in general asynchronous) productive computation in the underlying model with symmetric weights (and nonnegative feedbacks) converges from any initial state towards some stable final state. An analogous result can be shown for parallel update where a cycle of length at most two different states may appear [22]. Thus Hopfield nets compared with general asymmetric networks have favorable convergence properties. Part of the appeal of Hopfield nets also stems from their natural hardware implementation (e.g. Ising spin glasses [3], optical computers [7], etc.). Besides associative memory, the proposed uses of Hopfield networks include, e.g., fast approximate solution of combinatorial optimization problems [13].

It is known [20] that under fully parallel mode any convergent asymmetric discrete-time recurrent neural network with n neurons can be simulated by a symmetric network of quadratic size $O(n^2)$. This also means that the infinite polynomial-size increasing sequences of discrete symmetric networks are computationally equivalent to (nonuniform) polynomially space-bounded Turing machines, i.e. they compute the complexity class PSPACE/poly or P/poly when polynomial weights are considered. The idea behind this simulation is that each directed edge is implemented by a small symmetric subnetwork which receives energy support from a symmetric clock subnetwork (a binary counter) [11] in order to propagate a signal in the right direction.

In this section the construction from [20] will be improved by reducing the number of neurons in the simulating symmetric network to the linear size $6n + 2$ which is asymptotically optimal. This is achieved by simulating the neurons (instead of edges) whose states are updated by means of the clock technique. A similar idea was used for an analogous continuous-time simulation [27]. This result can be interpreted in the sense that convergent asymmetric networks are computationally equivalent with symmetric ones to a greater degree when considering also the network size.

We will first briefly specify the model of a finite *discrete recurrent neural network*. The network consists of n simple computational *units* or *neurons*, indexed as $1, \ldots, n$, which are connected into a generally cyclic oriented graph or *architecture* in which each edge (i, j) leading from neuron i to j is labelled with an integer *weight* $w(i, j) = w_{ji}$. The absence of a connection within the architecture corresponds to a zero weight between the respective neurons, and vice versa. Special attention will be paid to *Hopfield (symmetric) networks*, whose architecture is an undirected graph with symmetric weights $w(i, j) = w(j, i)$ for every i, j.

The synchronous computational dynamics of the network, working in *fully parallel mode*, determines the evolution of the *network state* $\mathbf{y}^{(t)} = (y_1^{(t)}, \ldots, y_n^{(t)}) \in \{0, 1\}^n$ for all discrete time instants $t = 0, 1, \ldots$ as follows. At the beginning of the computation, the network is placed in an *initial state* $\mathbf{y}^{(0)}$ which may include an external input. At discrete time $t \geq 0$, each neuron $j = 1, \ldots, n$ collects its binary *inputs* from the *states* (*outputs*) $y_i^{(t)} \in \{0, 1\}$ of incident neurons i. Then its integer *excitation* $\xi_j^{(t)} = \sum_{i=0}^{n} w_{ji} y_i^{(t)}$ $(j = 1, \ldots, n)$ is computed as the respective weighted sum of inputs including an integer *bias* w_{j0} which can be viewed as the weight of the formal constant unit input $y_0^{(t)} = 1$. At the next instant $t + 1$, an *activation function* σ is applied to $\xi_j^{(t)}$ for all neurons $j = 1, \ldots, n$ in order to determine the new network state $\mathbf{y}^{(t+1)}$ as follows:

$$y_j^{(t+1)} = \sigma\left(\xi_j^{(t)}\right) \quad j = 1, \ldots, n \tag{1.1}$$

where a *binary-state* neural network employs the *hard limiter* (or *threshold*) activation function

$$\sigma(\xi) = \begin{cases} 1 & \text{for } \xi \geq 0 \\ 0 & \text{for } \xi < 0\,. \end{cases} \tag{1.2}$$

Now we will show the following theorem concerning the computational equivalence of asymmetric and symmetric networks:

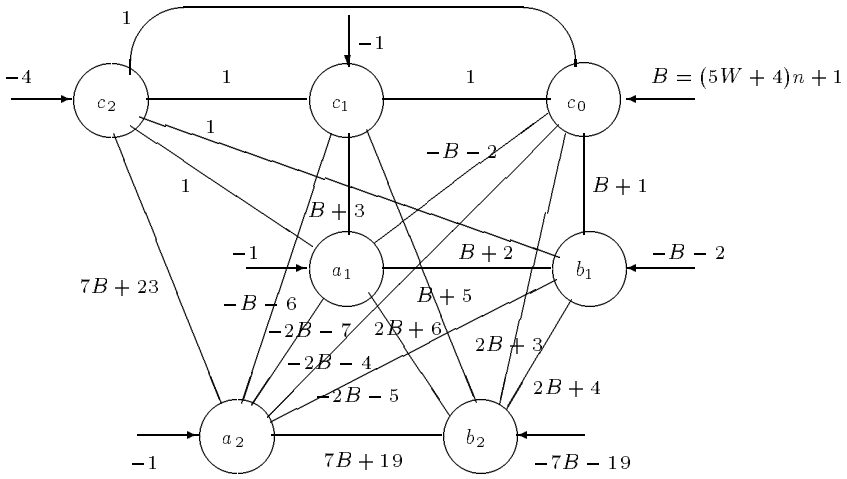

Figure 1.1: A 3-bit counter network

Theorem 1 *Any fully parallel computation by a recurrent neural network of n binary neurons, with generally asymmetric weights, which converges within $t^\star$ discrete updates can be simulated by a Hopfield net with $6n + 2$ neurons within $4t^\star$ discrete-time steps.*

Proof: Observe, first, that any converging computation by an asymmetric network of n binary neurons must terminate within $t^\star \leq 2^n$ steps. A basic technique used in our proof is the construction of an $(n + 1)$-bit symmetric clock subnetwork (a binary counter) which, using $3n + 1$ units being initially at the zero state, produces a sequence of 2^n well-controlled oscillations before it converges. This sequence of clock pulses is used to drive the rest of the network where each neuron is simulated by a symmetric subnetwork of 3 units. The construction of the $(n+1)$-bit binary counter will be described by induction for n. An example of a 3-bit counter network is presented in Figure 1.1, where the symmetric connections between neurons are labelled with corresponding weights, and the biases are indicated by the edges drawn without an originating unit. In the sequel the symmetric weights in the Hopfield net will be denoted by w whereas w' denotes the original asymmetric weights.

The counter bit c_0 with the bias $w(0, c_0) = (5W + 4)n + 1$ denoted also by B in Figure 1.1 where

$$W = \max_{j=1,\ldots,n} \sum_{i=0}^{n} |w'_{ji}| \,, \tag{1.3}$$

which is initially *passive* (i.e. its state is zero), will *fire* or be *active* (i.e. its state is 1) at the next time instant according to (1.2) since its excitation is positive.

For the induction step suppose that the counter has been constructed up to the first $k < n + 1$ counter bits $c_0, \ldots, c_{k-1}$ and denote by V_k the set of all its $n_k = 3(k - 1) + 1$ neurons, including the auxiliary ones labelled a_ℓ, b_ℓ for $\ell = 1, \ldots, k-1$. Then the counter unit c_k is connected to all n_k neurons $v \in V_k$ via unit weights $w(v, c_k) = 1$ which, together with its bias $w(0, c_k) = -n_k$, make c_k fires when all these units are active. This includes the first k active counter units $c_0, \ldots, c_{k-1}$ which means that counting from 0 to $2^k - 1$ has been accomplished. In addition, unit c_k is connected to a_k which is further linked to b_k so that these auxiliary neurons are, one by one, activated after c_k fires. This is implemented by the following weights $w(c_k, a_k) = W_k > 0$ (specified below), $w(a_k, b_k) = W_k - n_k$, and the biases $w(0, a_k) = -1$, $w(0, b_k) = n_k - W_k$. Neuron a_k resets all the units in V_k to their initial zero states. For this purpose, a_k is further connected to each $v \in V_k$ via a sufficiently large negative weight $w(a_k, v) < 0$ such that $-w(a_k, v) > 1 + \sum_{u \in V_k \cup \{0\} : w(u,v) > 0} w(u, v)$ exceeds their mutual positive influence (including the weight $w(c_k, v) = 1$ and possibly its positive bias $w(0, v) > 0$). This also determines the above-mentioned large positive weight parameter $W_k = 1 - \sum_{v \in V_k} w(a_k, v)$ which makes the state of a_k (similarly for b_k) independent on the outputs from $v \in V_k$. Finally, the unit b_k balances the influence of a_k on V_k so that the first k counter bits can again count from 0 to $2^k - 1$ but

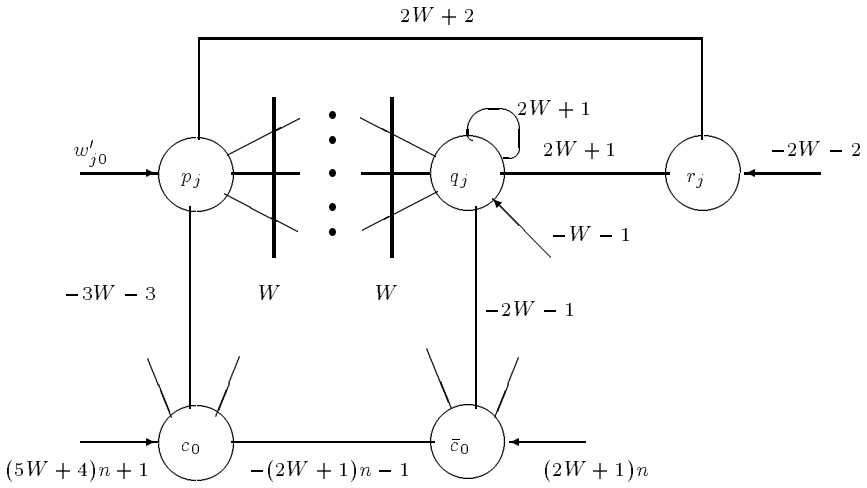

Figure 1.2: Symmetric simulation of neuron j

now with c_k being active. This is achieved by the weight $w(b_k, v) = -w(a_k, v) - 1$ for each $v \in V_k$ in which -1 compensates $w(c_k, v) = 1$. This completes the induction step.

Now, the symmetric clock subnetwork, particularly the counter unit c_0 which outputs the state sequence $(0111)^{2^n}$ during the computation, will be used to proceed the respective simulation of an asymmetric network. For this purpose, neuron $\bar{c}_0$ is added which computes the negation of c_0 output. Then for each neuron j from the asymmetric network, 3 units p_j, q_j, r_j are introduced in the Hopfield net so that p_j represents the new (current) state $y_j^{(t)}$ of j at time $t \geq 1$ while q_j stores the old state $y_j^{(t-1)}$ of j from the preceding time instant $t-1$, and r_j is an auxiliary neuron realizing the update of the old state. The corresponding symmetric subnetwork simulating one neuron j is depicted in Figure 1.2. The total number of units simulating the asymmetric network is $3n + 1$ (including $\bar{c}_0$) which, together with the clock size $3n + 1$, gives the desired $6n + 2$ neurons of the Hopfield net.

At the beginning of the simulation all the neurons in the Hopfield net are initially passive except for units q_j corresponding to the original initially active neurons j, i.e. $y_j^{(0)} = 1$. Then an asymmetric network update at time $t \geq 1$ is simulated by a cycle of four steps in the Hopfield net as follows. In the first step, unit c_0 fires and remains active until its state is changed by the clock since its large positive bias makes it independent on all the n neurons p_j. Also the unit $\bar{c}_0$ fires because it computes the negation of c_0 that was initially passive. At the same time each neuron p_j computes the new state $y_j^{(t)}$ from the old ones $y_i^{(t-1)}$ which are stored in corresponding units q_i. Thus each neuron p_j is connected with units q_i via the original weights $w(q_i, p_j) = w'(i, j)$ and also its bias $w(0, p_j) = w'(0, j)$ is preserved. So far, unit q_j keeps the old state $y_j^{(t-1)}$ due to its feedback. In the second step, the new state $y_j^{(t)}$ is copied from p_j to r_j, and the active neuron c_0 makes each neuron p_j passive by means of a large negative weight which exceeds the positive influence from units q_i $(i = 1, \ldots, n)$ including its bias $w(0, p_j)$ according to (1.3). Similarly, the active neuron $\bar{c}_0$ erases the old state $y_j^{(t-1)}$ from each neuron q_j by making it passive with the help of a large negative weight which exceeds its feedback and the positive influence from units p_i $(i = 1, \ldots, n)$. Finally, also neuron $\bar{c}_0$ becomes passive since c_0 was active. In the third step, the current state $y_j^{(t)}$ is copied from r_j to q_j since all the remaining incident neurons p_i and $\bar{c}_0$ are and remain passive due to c_0 being active. Therefore also unit r_j becomes passive. In the fourth step, c_0 becomes passive and the state $y_j^{(t)}$, being called old from now on, is stored in q_j. Thus the Hopfield net finds itself at the starting point of the next asymmetric network update simulation at time $t + 1$ which proceeds in the same way. Hence the whole simulation is achieved within $4t^\star$ discrete-time steps. □

2 Convergence Time Analysis

In this section the *convergence time* in Hopfield networks, which is the number of discrete updates until the network converges, will be analyzed. We will consider only the worst case bounds while the average-case analysis can be found in [17]. Obviously, there are exactly 2^n different states in a network with n binary neurons which yields trivial 2^n upper bound on the convergence time in symmetric networks of size n. On the other hand, the symmetric clock network [11] which was used in the proof of Theorem 1 represents an explicit example of a Hopfield net whose convergence time is exponential with respect to n. Namely, this gives $\Omega(2^{n/3})$ lower bound on the convergence time of Hopfield nets since the respective $(k+1)$-bit binary counter requires $n = 3k+1$ neurons.

However, the above-mentioned bounds do not take the weight size into account. The corresponding upper bound can be derived by the energy function argument which also witnesses the convergence property of symmetric nets. Namely, for a sequential computation of a Hopfield net, for the simplicity, with zero feedbacks $w_{jj} = 0$ and biases $w_{j0} = 0$ $(j = 1, \ldots, n)$, an *energy* associated with state $y^{(t)}$ at time $t \geq 0$ can be defined as follows:

$$E\left(y^{(t)}\right) = E(t) = -\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n} w_{ji} y_i^{(t)} y_j^{(t)} \tag{2.1}$$

for which Hopfield showed that $E(t) \leq E(t-1) - 1$ for every $t \geq 1$ of a productive computation [12]. Moreover, the energy function (2.1) is bounded, i.e. $|E(t)| \leq W$ where

$$W = \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n} |w_{ji}| \tag{2.2}$$

is called the *weight* of the network. Hence, the computation must converge within time $O(W)$ which can even be made more accurate by using a slightly different energy function [8]. This yields the polynomial upper bound on the convergence time of Hopfield nets with polynomial weights. Similar arguments can be used for fully parallel updates.

In the following theorem these results will be translated into the convergence time bounds with respect to the length of bit representations of Hopfield nets. Namely, for a symmetric network which is described within M bits, the convergence-time lower and upper bounds, $2^{\Omega(M^{1/3})}$ and $2^{O(M^{1/2})}$, respectively will be shown. It is an open problem whether these upper or lower bounds can be improved. This is an important issue since the convergence-time results for binary-state networks can be compared with those for analog-state (or even continuous-time) networks in which the precision of real weight parameters (i.e. the representation length) plays an important role. For example, there exists an analog-state symmetric network with an encoding size of M bits that converges after $2^{\Omega(g(M))}$ continuous-time units, where $g(M)$ is an arbitrary continuous function such that $g(M) = o(M)$, $g(M) = \Omega(M^{2/3})$, and $M/g(M)$ is increasing [27]. From the result presented here it follows that the computation of this analog symmetric network terminates later than that of any other discrete Hopfield net of the same representation size. This approach also appears to be more rigorous since we express the convergence time with respect to the full descriptional complexity of the Hopfield net instead of to the number of neurons which captures its computational sources only partially.

Theorem 2 *There exists a Hopfield network with an encoding size of M bits that converges after $2^{\Omega(M^{1/3})}$ updates and any computation of a symmetric network with a binary representation of M bits terminates within $2^{O(M^{1/2})}$ discrete computational steps.*

Proof: For the underlying lower bound the clock network with parameter $B = 1$ from the proof of Theorem 1 can again be exploited. It is sufficient to estimate its representation length. By induction, the maximum weight in the $(k+1)$-bit counter with $n = 3k+1$ neurons is of order $2^{O(n)}$. This corresponds to $O(n)$ bits per weight which is repeated $O(n^2)$ times, and thus yields at most $M = O(n^3)$ bits in the representation. Hence, the convergence-time lower bound $2^{\Omega(n)}$ of the clock network can be expressed as $2^{\Omega(M^{1/3})}$.

On the other hand, consider a Hopfield network with an M-bit representation that converges after $T(M)$ updates. A major part of this M-bit representation consists of m binary encodings of weights $w_1, \ldots, w_m$ of the corresponding lengths $M_1, \ldots, M_m$ where $\sum_{r=1}^{m} M_r = \Theta(M)$. Clearly, there must be at least $T(M)$ different energy levels corresponding to the states visited during the computation. Thus the underlying weights must produce at least $S \geq T(M)$ different sums $\sum_{r \in A} w_r$ for $A \subseteq \{1, \ldots, m\}$ where w_r for $r \in A$ agrees with w_{ji} for $y_i = y_j = 1$ in (2.1). So, it is sufficient to upper bound the number of different sums over m weights whose binary representations form a $\Theta(M)$-bit string altogether.

For $m = \Theta(M^{1/2})$ consider m weights $1, 2, 4, 8, \ldots, 2^{\Theta(M^{1/2})}$, each with a representation length of order $M_r = O(M^{1/2})$, which generate $S = 2^{\Theta(M^{1/2})}$ different sums. For $m = o(M^{1/2})$ we have only $2^m = 2^{o(M^{1/2})}$ possible sums, which is less than that in the previous case. On the contrary, suppose that $m = \omega(M^{1/2})$. We define the maximum number of nonzero weights similarly as in the first case so that each nonzero weight representation consists of zeros except for one bit in which the remaining weight representations have zeros. Thus p such nonzero weights altogether require at least $1 + 2 + \ldots + p = \Omega(p^2)$ bits for their representation which must be M at most and hence $p = O(M^{1/2})$ which yields at most $2^{O(M^{1/2})}$ different sums. It follows from the weight definition that the bits in the representations of the remaining zero weights can arbitrarily be generated by summing appropriate nonzero weights, otherwise p would not be the maximum. An alternative definition of weights can possibly increase the number of different sums by exploiting at most $O(\log m)$ carry bits that are produced by summing m weights. However, this yields only factor $O(m)$ and hence the number of different sums $S \leq m 2^{O(M^{1/2})} = 2^{O(M^{1/2})}$ from $m \leq M$. Thus $T(M) \leq 2^{O(M^{1/2})}$. □

3 Approximating the Minimum Energy Problem

Another important issue in Hopfield nets is the *MIN ENERGY* or *GROUND STATE problem* of finding a network state with minimal energy (2.1) for a given symmetric neural network. Remember that in (2.1) it is assumed, for reasons of simplicity, that $w_{jj} = 0$ and $w_{j0} = 0$ for $j = 1, \ldots, n$. In addition, without loss of generality [21], we will work throughout this section with frequently used bipolar states $-1, 1$ of neurons instead of binary ones $0, 1$ introduced in (1.2) where 0 is now replaced by -1. This problem appears to be of a special interest since many hard combinatorial optimization problems have been heuristically solved by minimizing the energy in Hopfield nets [1, 13]. This issue is also important in statistical physics which originally inspired the Hopfield net models, e.g. Ising spin glasses [3].

Unfortunately, the decision version of the MIN ENERGY problem, i.e. whether there exits a network state having an energy less than the prescribed value, is NP-complete. This can be observed from the above-mentioned reductions of hard optimization problems to MIN ENERGY. For an explicit NP-completeness proof see e.g. [29] where a reduction from SAT is exploited. On the other hand there is a MIN ENERGY polynomial algorithm for special cases of Hopfield nets whose architectures are planar lattices [6] or planar graphs [3].

Perhaps, the most direct and frequently used reduction to MIN ENERGY is from the *MAX CUT problem* (see e.g. [4]) which, given an undirected graph $G = (V, E)$ with an integer edge evaluation $c : E \longrightarrow \mathcal{Z}$, is the issue of finding a *cut* $V_1 \subseteq V$ which maximizes the *cut size*

$$c(V_1) = \sum_{\{i,j\} \in E, i \in V_1, j \notin V_1} c(\{i,j\}) - \sum_{\{i,j\} \in E, c(\{i,j\}) < 0} c(\{i,j\}) \,. \tag{3.1}$$

In fact, this is a generalized version of MAX CUT that allows negative edge evaluations necessary for the opposite reduction from MIN ENERGY to MAX CUT. Recently, a new randomized approximation algorithm with a high performance guarantee $\alpha = 0.87856$ for this MAX CUT formulation has been proposed [10] and later derandomized [19] which we will exploit for approximating the MIN ENERGY problem. Namely, we will show the theorem that MIN ENERGY can be approximated in a polynomial time within the absolute error less than $0.243W$ where W is the network weight (2.2). For $W = O(n^2)$ which is satisfied by e.g. Hopfield nets with n neurons and constant weights, this result matches the lower bound $\Omega(n^{2-\varepsilon})$ which cannot be guaranteed by any approximate polynomial time MIN ENERGY

algorithm for every $\varepsilon > 0$ [4], unless $P = NP$. In addition, an approximate polynomial time MIN ENERGY algorithm with absolute error $O(n/\log n)$ is also known in a special case of Hopfield nets whose architectures are two-level grids [5].

Theorem 3 *The MIN ENERGY problem for Hopfield nets can be approximated in a polynomial time within the absolute error less than* $0.243W$ *where* W *is the network weight (2.2).*

Proof: We will first recall the well-known simple reduction between MIN ENERGY and MAX CUT problems. For a Hopfield network with architecture G and weights $w(i,j)$ we can easily define the corresponding instance $G = (V, E); c$ of MAX CUT with the edge evaluation $c(\{i,j\}) = -w(i,j)$ for $\{i,j\} \in E$ (i.e. for $w(i,j) \neq 0$). We will show that any cut $V_1 \subseteq V$ of G corresponds to a Hopfield net state $\mathbf{y} \in \{-1,1\}^n$ where $y_i = 1$ if $i \in V_1$ and $y_i = -1$ for $j \in V \setminus V_1$, so that the respective cut size $c(V_1)$ is related to the underlying energy $E(\mathbf{y})$. Thus the energy function (2.1) can be expressed in terms of cut size (3.1) as follows:

$$\begin{aligned}
E(\mathbf{y}) &= -\frac{1}{2}\sum_{i,j\in V} w(i,j)y_iy_j = \\
&= -\frac{1}{2}\sum_{y_i=y_j} w(i,j) + \frac{1}{2}\sum_{y_i\neq y_j} w(i,j) + \frac{1}{2}\sum_{y_i\neq y_j} w(i,j) - \frac{1}{2}\sum_{y_i\neq y_j} w(i,j) = \\
&= -\frac{1}{2}\sum_{i,j\in V} w(i,j) + \sum_{y_i\neq y_j} w(i,j) = -\frac{1}{2}\sum_{w(i,j)<0} w(i,j) - \frac{1}{2}\sum_{w(i,j)>0} w(i,j) + \\
&\quad + \sum_{y_i\neq y_j} w(i,j) + \frac{1}{2}\sum_{w(i,j)>0} w(i,j) - \frac{1}{2}\sum_{w(i,j)>0} w(i,j) = \\
&= -\frac{1}{2}\sum_{w(i,j)<0} w(i,j) + \frac{1}{2}\sum_{w(i,j)>0} w(i,j) - \sum_{w(i,j)>0} w(i,j) + \sum_{y_i\neq y_j} w(i,j) = \\
&= W + 2\sum_{\{i,j\}\in E, c(\{i,j\})<0} c(\{i,j\}) - 2\sum_{\{i,j\}\in E, i\in V_1, j\notin V_1} c(\{i,j\}) = \\
&= W - 2c(V_1)\,. \qquad (3.2)
\end{aligned}$$

It follows from (3.2) that the minimum energy state corresponds to the maximum cut.

Now, the approximate polynomial time algorithm from [10] can be employed to solve instance $G = (V, E); c$ of the MAX CUT problem which provides a cut V_1 whose size $c(V_1) \geq \alpha c^\star$ is guaranteed to be at least $\alpha = 0.87856$ times the maximum cut size $c^\star$. Let cut V_1 correspond to the Hopfield network state $\mathbf{y}$ which implies $c(V_1) = 1/2(W - E(\mathbf{y}))$ from (3.2). Hence, we get a guarantee $W - E(\mathbf{y}) \geq \alpha(W - E^\star) = \alpha W - \alpha E^\star$ where $E^\star$ is the minimum energy corresponding to the maximum cut $c^\star$ which implies $E(\mathbf{y}) \leq (1-\alpha)W + \alpha E^\star$. By adding $-E^\star$, this inequality can be rewritten as $E(\mathbf{y}) - E^\star \leq (1-\alpha)(W - E^\star)$. Since $|E^\star| \leq W$, we obtain the desired guarantee for the absolute error $E(\mathbf{y}) - E^\star \leq (1-\alpha)2W < 0.243W$. □

4 Turing Universality of Finite Analog Hopfield Nets

In this section we will deal with the computational power of finite *analog-state* discrete-time recurrent neural networks which, instead of the threshold activation function (1.2), employ e.g. the *saturated-linear* sigmoid activation function

$$\sigma(\xi) = \begin{cases} 1 & \text{for } \xi > 1 \\ \xi & \text{for } 0 \leq \xi \leq 1 \\ 0 & \text{for } \xi < 0\,. \end{cases} \qquad (4.1)$$

Hence the states of analog neurons are real numbers within the interval $[0,1]$, and similarly the weights (including biases) are allowed to be reals.

The computational power of asymmetric analog networks is known to increase with the Kolmogorov complexity of real weights [2]. With integer weights such networks are equivalent to finite automata [14, 15, 28], while with rational weights arbitrary Turing machines can be simulated [15, 24]. With arbitrary real weights the network can even have 'super-Turing' computational capabilities, e.g. polynomial time computations correspond to the complexity class P/poly and all languages can be recognized within exponential time [23]. On the other hand, it is known that any amount of analog noise reduces the computational power of this model to that of finite automata [18].

For finite symmetric networks, only the computational power of binary-state Hopfield nets is fully characterized. Namely, they recognize the so-called Hopfield languages [25] which establish a proper subclass of regular languages and hence, they are less powerful than finite automata. Hopfield languages can also be faithfully recognized by analog symmetric neural networks [18, 26] and this provides the lower bound on their computational power. A natural question arises whether the finite analog Hopfield nets are Turing universal, i.e. whether a Turing machine simulation can be achieved with rational weights similarly as in the asymmetric case [15, 24]. The main problem is that under fully parallel update any analog Hopfield net with rational weights converges to a limit cycle of length at most two [16]. Thus the only possibility of simulating Turing machines is to exploit a sequence of rational network states converging to this limit cycle which seems to be tricky if possible at all. A more reasonable approach is to supply an external clock that produces an infinite sequence of binary pulses providing the symmetric network with an energy support, e.g. for simulating an asymmetric analog network similarly as in Theorem 1. In this way the computational power of the analog Hopfield nets with an external clock is proved to be the same as that of the asymmetric analog networks. Especially for rational weights, this implies that they are Turing universal. The following theorem also completely characterizes the infinite binary sequences by the external clock, which prevent the Hopfield network from converging.

Theorem 4 *Any fully parallel computation by an analog-state recurrent neural network with real weights and n neurons can be simulated by an analog Hopfield net with real weights of the same Kolmogorov complexity having $3n+8$ units and one external input producing an infinite binary sequence including the infinite number of strings of the form $bx\bar{b} \in \{0,1\}^3$ where $b \neq \bar{b}$, which are necessary to prevent the symmetric network from converging.*

Proof: First observe that an infinite binary sequence produced by the external input (clock) c that does not satisfy the assumption of the theorem must be of the form $\mathbf{u}(b_1b_2)^\star$ where $b_1, b_2 \in \{0,1\}$, and $\mathbf{u} \in \{0,1\}^\star$ is a prefix which clearly cannot prevent the network from converging to a limit cycle of length two. On the other hand we will prove that if the sequence meets the respective condition, then it must contain an infinite number of strings of the form $1x0$ ($x \in \{0,1\}$) since these strings necessarily accompany the infinite number of strings $0x1$. Thus consider two subsequent occurrences of $0x1$. For $x = 1$, after 011 possibly followed by several 1's, at least one 0 must appear due to the next $0x1$. For $x = 0$, the string 001 is followed either by 1 which means the previous case with 011 applies or by 0 possibly succeeded by several occurrences of 10, which is followed either by a desired 0 or by 11 which again leads to 011.

The symmetric simulation of the asymmetric analog network is very similar to the discrete one (see Theorem 1). Thus each neuron j in the original analog asymmetric network with n neurons is simulated by 3 units p_j, q_j, r_j which are controlled by other 3 central neurons f, g, h (corresponding to $c_0, \bar{c}_0$ in the discrete case) whose binary states are generated by a small symmetric subnetwork of 5 auxiliary units a, d, e, r, s transforming the binary external input signal from c to a well-controlled binary sequence. This gives the desired $3n+8$ units of the simulating analog Hopfield net. The situation is depicted in Figure 4.1 including the definition of symmetric weights where W is introduced in (1.3) and $B = (4W+1)n + 7$.

Every binary external input bit is copied from clock c to r and further to s and therefore the states of these neurons s, r, c store the last 3 bits of the input sequence, respectively. Neuron a detects the underlying strings of the form $1x0$ ($x \in \{0,1\}$) at the input, i.e. it is active iff s is active and c is passive. It may happen that two such strings follow each other immediately (also notice that it is impossible for such string to appear again the next step but one). This case is indicated by the activity of both neurons a, d where d copies the state from a. Thus unit e copies the state 1 from a iff d is

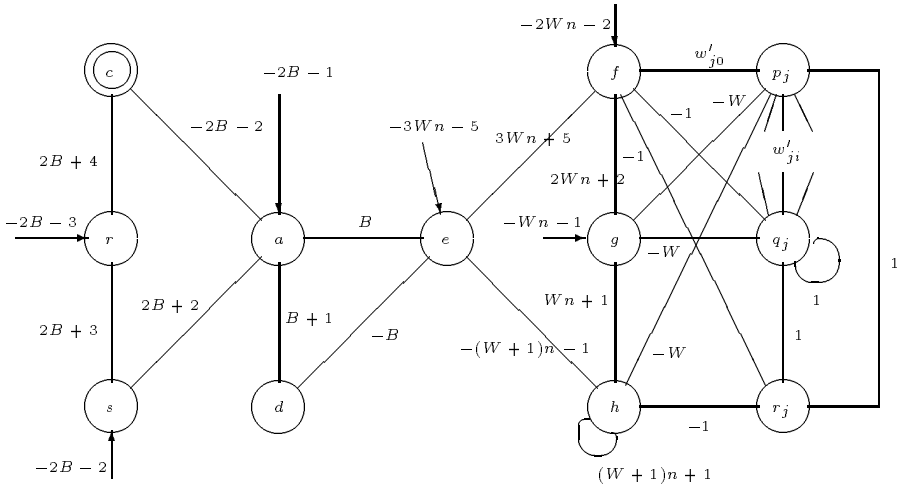

Figure 4.1: Analog symmetric simulation of analog neuron j

passive. Hence, the neuron e produces a 'noiseless' sequence of zeros containing an infinite number of strings 100, each being exploited for the simulation of one computational step of the original analog asymmetric network similarly as in the proof of Theorem 1.

Namely, q_j stores the old analog state $y_j^{(t-1)}$ by its unit feedback according to (4.1) and also h remains initially active preserving the stability of p_j, q_j, r_j. After e fires, the unit h becomes passive and f is active which controls the computation of a new analog state $y_j^{(t)}$ by p_j via the original weights $w(q_i, p_j) = w'(i, j)$ and the bias $w(f, p_j) = w'(0, j)$. Then the control signal is further copied from f to g, which enables r_j to receive the state $y_j^{(t)}$. Finally, h becomes active and the state of q_j is updated by $y_j^{(t)}$ until e fires again to proceed the next simulation step. □

Bibliography

[1] Aarts, E., Korst, J. *Simulated Annealing and Boltzmann Machines.* Wiley and Sons, 1989.

[2] Balcázar, J. L., Gavaldà, R., Siegelmann, H. T. Computational power of neural networks: A characterization in terms of Kolmogorov complexity. *IEEE Transactions of Information Theory*, **43**, 1175–1183, 1997.

[3] Barahona, F. On the computational complexity of Ising spin glass models. *Journal of Physics A*, **15**, 3241–3253, 1982.

[4] Bertoni, A., Compadelli, P. On the approximability of the energy function. In *Proceedings of the ICANN'94 conference*, 1157–1160, Springer-Verlag, 1994.

[5] Bertoni, A., Campadelli, P., Gangai, C., Posenato, R. Approximability of the ground state problem for certain Ising spin glasses. *Journal of Complexity*, **13**, 323–339, 1997.

[6] Bieche, I., Maynard, R., Rammal, R., Uhry, J. P. On the ground states of the frustration model of a spin glass by a matching method of graph theory. *Journal of Physics A*, **13**, 2553–2576, 1980.

[7] Farhat, N. H., Psaltis, D., Prata, A., Paek, E. Optical implementation of the Hopfield model. *Applied Optics*, **24**, 1469–1475, 1985.

[8] Floreén, P. Worst-case convergence times for Hopfield memories. *IEEE Transactions on Neural Networks*, **2**, 533–535, 1991.

[9] Floreén, P., Orponen, P. Complexity issues in discrete Hopfield networks. Research Report A–1994-4, Department of Computer Science, University of Helsinki, 1994.

[10] Goemans, M. X., Williamson, D. P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, **42**, 1115–1145, 1995.

[11] Goles, E., Martínez, S. Exponential transient classes of symmetric neural networks for synchronous and sequential updating. *Complex Systems*, **3**, 589–597, 1989.

[12] Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. In *Proceedings of the National Academy of Sciences*, vol. **79**, 2554–2558, 1982.

[13] Hopfield, J. J., Tank, D. W. "Neural" computation of decisions in optimization problems. *Biological Cybernetics*, **52**, 141-152, 1985.

[14] Horne, B. G., Hush, D. R. Bounds on the complexity of recurrent neural network implementations of finite state machines. *Neural Networks*, **9**, 243–252, 1996.

[15] Indyk, P. Optimal simulation of automata by neural nets. In *Proceedings of the 12th Annual Symposium on Theoretical Aspects of Computer Science*, vol. **900** of LNCS, 337-348, Springer-Verlag, Berlin, 1995.

[16] Koiran, P. Dynamics of discrete time, continuous state Hopfield networks. *Neural Computation*, **6**, 459–468, 1994.

[17] Komlós, P., Patury, R. Convergence results in an associative memory model. *Neural Networks*, **1**, 239–250, 1988.

[18] Maass, W., Orponen, P. On the effect of analog noise in discrete-time analog computations. *Neural Computation*, **10**, 1071–1095, 1998.

[19] Mahajan, S., Ramesh, H. Derandomizing semidefinite programming based approximation algorithms. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science.*, 162–163, IEEE, Los Alamitos, California, 1995.

[20] Orponen, P. The computational power of discrete Hopfield nets with hidden units. *Neural Computation*, **8**, 403–415, 1996.

[21] Parberry, I. A primer on the complexity theory of neural networks. In *Formal Techniques in Artificial Intelligence: A Sourcebook*, editor R. B. Banerji, 217–268, Elsevier, North-Holland, Amsterdam, 1990.

[22] Poljak, S., Sůra, M. On periodical behaviour in societies with symmetric influences. *Combinatorica*, **3**, 119–121, 1983.

[23] Siegelmann, H. T., Sontag, E. D. Analog computation via neural networks. *Theoretical Computer Science*, **131**, 331–360, 1994.

[24] Siegelmann, H. T., Sontag, E. D. Computational power of neural networks. *Journal of Computer System Science*, **50**, 132–150, 1995.

[25] Šíma, J. Hopfield languages. In *Proceedings of the SOFSEM Seminar on Current Trends in Theory and Practice of Informatics*, LNCS **1012**, 461–468, Springer-Verlag, 1995.

[26] Šíma, J. Analog stable simulation of discrete neural networks. *Neural Network World*, **7**, 679–686, 1997.

[27] Šíma, J., Orponen, P. A continuous-time Hopfield net simulation of discrete neural networks. Technical report V-773, ICS ASCR, Prague, January, 1999.

[28] Šíma, J., Wiedermann, J. Theory of neuromata. *Journal of the ACM*, **45**, 155–178, 1998.

[29] Wiedermann, J. Complexity issues in discrete neurocomputing. *Neural Network World*, **4**, 99–119, 1994.