ELSEVIER

BioSystems 58 (2000) 167–176

BioSystems

www.elsevier.com/locate/biosystems

Probabilistic computation by neuromine networks

R.D. Hangartner [a,*], P. Cull [b]

[a] *Cray Inc., Seattle, WA 98104, USA*
[b] *Department of Computer Science, Oregon State University, Corvallis, OR 97331, USA*

Abstract

In this paper, we address the question, can biologically feasible neural nets compute more than can be computed by deterministic polynomial time algorithms? Since we want to maintain a claim of plausibility and reasonableness we restrict ourselves to algorithmically easy to construct nets and we rule out infinite precision in parameters and in any analog parts of the computation. Our approach is to consider the recent advances in randomized algorithms and see if such randomized computations can be described by neural nets. We start with a pair of neurons and show that by connecting them with reciprocal inhibition and some tonic input, then the steady-state will be one neuron *ON* and one neuron *OFF*, but which neuron will be *ON* and which neuron will be *OFF* will be chosen at *random* (perhaps, it would be better to say that microscopic noise in the analog computation will be turned into a megascale random bit). We then show that we can build a small network that uses this random bit process to generate repeatedly random bits. This random bit generator can then be connected with a neural net representing the deterministic part of randomized algorithm. We, therefore, demonstrate that these neural nets can carry out probabilistic computation and thus be less limited than classical neural nets.

Keywords: Analog neural network; Stochastic neural network; Randomized algorithms; Probabilistic computation

1. Introduction

From their beginning with McCulloch and Pitts, artificial neural nets (ANNs) have been used as models of computation. In the hands of von Neumann, neural nets became the logical basis for describing computing machinery. Kleene (1956) showed that the computational power of a discrete-time threshold neural net was equivalent to the computational power of a finite state machine. Over the last several years researchers have begun more intensively studying the formal computational power of more biologically-inspired ANNs. Most of this work deals with discrete-time networks, but a few results have appeared for analog systems including a demonstration that idealized discrete-time deterministic networks with infinite precision weights are more powerful than deterministic Turing machines (TMs) (Siegelmann and Sontag, 1992). However, once physically realistic limitations are imposed on the weights, these networks are no more powerful than deterministic TMs (Abu-Mostafa, 1986; Siegelmann and Sontag, 1991). Incorporating stochastic elements in

* Corresponding author.
E-mail addresses: hangarr@pdx.or.com (R.D. Hangartner), pc@cs.orst.edu (P. Cull).

0303-2647/00/$ - see front matter
PII: S0303-2647(00)00120-9

physically realistic discrete-time networks makes them equivalent to probabilistic TMs and thus meaningfully augments their capabilities (Parberry and Schnitger, 1989).

For analog networks it appears likely that idealized deterministic analog networks are much more powerful than TMs (Moore, 1993). As with discrete networks, however, deterministic analog networks with physically realistic limitations on bandwidth and precision appear to be no more powerful than TMs (Vergis et al., 1986). Similar limitations in computational power appear to apply for deterministic spiking networks, which encode information in spike timing (Maass, 1994) when one applies radar and sonar pulse detection theory to these models.

The literature also contains a few complexity results for discrete-time ANNs, which most likely also hold for continuous-time ANNs. It is known that symmetric threshold gate networks (synchronous or asynchronous) cannot solve NP-hard problems unless $co\text{-}NP = NP$ (Bruck and Goodman, 1990) nor even approximate solutions unless $co\text{-}NP = NP$ (Yao, 1992) and, for some strongly NP-complete problems like TSP, unless $P = NP$ (Bruck and Goodman, 1990). Furthermore, it seems learning cannot increase the recognition power of discrete-time or continuous-time ANNs (Blum and Rivest, 1988).

Although these results are highly suggestive, at this point it appears that our knowledge of the computational power of biologically-inspired — and biologically plausible — stochastic analog networks is still fragmentary. In the hope of partially filling this gap, this work examines the computational power of one such model derived by Hangartner (Hangartner, 1994) as an approximation for the dynamics of a biologically inspired, stochastic pulse neuromime network (SPNN) model. The results show that while families of these networks do not have any greater recognition power than a deterministic TM, they do have practical space–time complexity advantages over conventional digital computers.

It is important to note how this work differs from that of Sato (1978) and others on mathematically sound stochastic models for single neurons (and some specific small networks), as well as more recent work in the spirit of Turova (1997). In contrast to the former, this research focuses on the problem of assessing the computational capabilities of arbitrarily large neural networks composed of neurons described by a slightly more sophisticated stochastic single-unit model. And unlike the latter, the fundamental neuromine model incorporates an arguably more realistic, but less mathematically idealized, non-uniform renewal sequence model for neural spike trains along with explicit axonal delays. These latter features lead to important differences between the network dynamics of the SPNN model and other network models that are key to the computational complexity results presented here. More specifically, in Hangartner (1994) it was also shown that a rather robust mapping existed between the biologically-inspired pulse (spiking) network model and a model for the mean dynamics of the network. Furthermore, a robust mapping also exists between this mean dynamics model and a ternary logic based analysis model, which supports extensive study into the formal computational capabilities of these networks.

2. The stochastic analog network model

The model derived in Hangartner (1994) approximates the mean dynamics of a biological-inspired, homogeneous recurrent stochastic pulse neuromime network by a stochastic analog network described by a noise-driven differential-delay equation:

$$\begin{aligned}\frac{\mathrm{d}\vec{y}(t)}{\mathrm{d}t} &= -\frac{1}{\tau_s}\vec{y}(t) + \frac{1}{\tau_s}\vec{g}(\xi[B\vec{h}(\vec{y}(t-\tau_a)) + \vec{f}]) \\ &\quad + \vec{q}(t) = -\frac{1}{\tau_s}\vec{y}(t) + \frac{1}{\tau_s}\vec{r}(\vec{y}(t-\tau_a)) + \vec{q}(t)\end{aligned} \tag{1}$$

(Henceforth **boldface** quantities represent stochastic processes.) In Eq. (1), the scalar τ_s represents the time-constant in a leaky-integrator model for the dendritic tree, the state vector $\vec{y} \in \boldsymbol{R}^n$ denotes the mean membrane depolarization in the leaky-integrator model, while the soft-limiter function $\vec{g}(\vec{y})$ approximates the effect of dendritic reverse synaptic potentials. Similarly, B represents the synaptic connection strengths and $\vec{f}$ represents a

net tonic input, the scalar ξ denotes an explicit gain parameter, τ_a represents the axonal propagation delay, and $\vec{h}(\vec{y})$ approximates the sigmoidal Hopfield–Tank activation function relating the cell input to the output firing rate. Finally, $\vec{\boldsymbol{q}}(t)$ is an stochastic process, which models the various random properties of biological networks; for present purposes, it is only necessary to assume that this is a bounded-variation process with zero-mean and short correlation time.

As Hale (1997) has demonstrated, systems described by deterministic and stochastic differential-delay equations have significantly different dynamics from those described by their pure delay counterparts. As the analyses of these systems are correspondingly more complex, the remainder of this section summarizes just the results from Hangartner (1994), and Hangartner and Cull (1995) needed for the complexity analyses in the following sections.

2.1. Ternary logic model for characterizing network equilibria

The equilibria of the deterministic part of Eq. (1), (which henceforth we shall just refer to as the equilibria of Eq. (1)) correspond to the fixed points of the *state transition mapping*.

$$\vec{z} = \vec{\psi}(\xi, \vec{z}) = \vec{h}\vec{g}(\xi\vec{\phi}(\vec{z})) \tag{2}$$

where $\vec{\phi}(\vec{z}) = B\vec{z} + \vec{f}$ and $\vec{z} = \vec{h}(\vec{y})$ (note that the mapping between the equilibria of Eq. (1) and the fixed points of Eq. (2) may be many-to-one if, for one or more i, $\mathrm{d}h_i(y_i)/\mathrm{d}y_i = 0$ over some range of y_i). Now consider the three-level quantization function $\breve{\mathrm{g}}$: $\boldsymbol{R} \to \mathscr{T}$, $\mathscr{T} = \{0, \chi, 1\}$ defined componentwise as:

$$\mathrm{g}_i(z_i) = \begin{cases} 0 & z_i^{[l]} \le z_i \le z_i^{[0]} \\ x & z_i^{[0]} \le z_i \le z_i^{[1]} \\ 1 & z_i^{[1]} \le z_i \le z_i^{[u]} \end{cases} \tag{3}$$

It can be shown that for most networks a *consistent ternary quantization* exists in the sense that there is a minimum gain ξ_0 such that the state transition mapping takes binary-valued vectors into binary-valued vectors if $\xi > \xi_0$, the same results hold for all networks if slight perturbations of $\vec{f}$ are allowed. As a result, the state transition mapping has a *Boolean model* $\vec{\Psi}$: $\mathscr{B}^n \to \mathscr{B}^n$ for $\vec{z}$ such that $\breve{\mathrm{g}}(\vec{z}) \in \mathscr{B}^n$, where $\mathscr{B} = \{0, 1\}$.

The Boolean model is extended into a ternary model $\hat{\vec{\Psi}}$: $\mathscr{T}^n \to \mathscr{T}^n$ by redefining χ to simultaneously denote the quantized representation for intermediate valued components of $\vec{z}$ *and* a representation for uncertainty as to the binary value of components of $\vec{\Psi}(\vec{z})$. This latter notion, derived from Brzozowski and Seger (1989) formal theory of asynchronous sequential networks, proposes that the value χ be assigned to $\hat{\vec{\Psi}}(\vec{z})$ if different Boolean assignments can be found for the χ components of $\vec{z}$ such that $\vec{\Psi}(\vec{z})$ evaluates to both 0 and 1, i.e.

$$\hat{\Psi}_i(\check{\mathrm{z}}) = \begin{cases} \chi & \check{\mathrm{z}} \notin \mathscr{B}^n; \\ & \exists \check{\mathrm{p}},\ \check{\mathrm{q}} \in \mathscr{B}^n,\ \Psi_i(\check{\mathrm{p}}) = \Psi_i(\check{\mathrm{q}}), \\ & \check{\mathrm{p}} \ne \check{\mathrm{q}}, \quad \mathrm{p}_i = \mathrm{q}_i = \mathrm{z}_i \ \text{ for } \ \mathrm{z}_i \in \mathscr{B} \\ \Psi_i(\check{\mathrm{z}}) & \text{otherwise} \end{cases}$$

For present purposes, the most important property of this ternary model $\hat{\vec{\Psi}}(\check{\mathrm{z}})$ is that it can be used to localize the fixed points $\check{\mathrm{z}}^\infty$ of the state transition mapping.

Theorem 2.1. *Suppose the state transition mapping* $\vec{\psi}(\xi, \vec{z})$ *for an instance of* Eq. (1) *has a consistent ternary quantization* $\breve{\mathrm{g}}(\vec{z})$ *and a Boolean model with corresponding ternary model* $\hat{\vec{\Psi}}(\check{\mathrm{z}})$. *Suppose further that*

$$\check{\mathrm{z}}^\infty = \hat{\vec{\Psi}}(\check{\mathrm{z}}^\infty) \in \mathscr{B}^n \tag{4}$$

and let

$$\mathscr{Y} = \{\vec{y} \,|\, \check{\mathrm{z}}^\infty = \breve{\mathrm{g}}(\vec{h}(\vec{y}))\}$$

Then there exists a ξ_0 such that the SPNN has an asymptotically stable equilibrium $\vec{y}^\infty \in \mathscr{Y}$ for any $\xi > \xi_0$. As is shown in Hangartner (1994), this result is a unique consequence of the differential-delay network model and the threshold value ξ_0 is strongly inversely related to the average axonal delay τ_a.

It would be nice if the converses of this theorem also held, i.e. that the network equilibria corresponding to non-binary fixed points of $\hat{\vec{\Psi}}(\check{\mathrm{z}})$ were unstable, but this does not appear to be true. Fortunately, it turns out that the bounded stabil-

ity region means that a slightly qualified version does hold:

Theorem 2.2. *Suppose the state transition mapping* $\vec{\psi}(\xi, \vec{z})$ *for an instance of* Eq. (1) *has a consistent ternary quantization* $\hat{\mathrm{g}}(\vec{z})$ *and a Boolean model with corresponding ternary model* $\hat{\hat{\Psi}}(\hat{\mathrm{z}})$. *Suppose further that*

$$\hat{\mathrm{z}}^{\infty} \leq \hat{\hat{\Psi}}(\hat{\mathrm{z}}^{\infty}) \in \mathscr{T}^n - \mathscr{B}^n \tag{5}$$

and let

$$\mathscr{Y} = \{\vec{y} \,|\, \hat{\mathrm{z}}^{\infty} = \hat{\mathrm{g}}(\vec{h}(\vec{y}))\}$$

If $J(\vec{y}^{\infty}, \xi)$, *the Jacobian, is not nilpotent everywhere in the closure* $\mathscr{Y}^{\mathrm{c}}$ *of* $\mathscr{Y}$, *then there exists a* ξ_0 *such that any equilibrium* $\vec{y}^{\infty} \in \mathscr{Y}$ *of the network is unstable for any* $\xi > \xi_0$. *(Here* $\leq$ *denotes a partial order relation in which* $0 \leq x,\ 1 \leq 0$.*)*

The preceding two theorems show that the ternary logic model can be used to locate and characterize the equilibria of the autonomous part of Eq. (1) for a broad class of instances. It follows for this class of networks that sufficiently small perturbations $\hat{\mathbf{q}}(t)$ will not destabilize the asymptotic equilibria in the sense that once the network state trajectory enters a bounded neighborhood of some minimum radius it will not exit. Similarly, the trajectory should exit any connected unstable limit set.

2.2. NOR networks

Although these results apply for networks with a wide variety of interconnection matrices B and tonic vectors f, the remainder of this paper will study one particularly important class of networks. The networks in this class are distinguished by the property that the ternary model for the state transition mapping $\vec{\psi}(\vec{y}, \xi)$ is the ternary NOR function

$$\hat{\Phi}(\hat{\mathrm{z}}) = \begin{cases} 0 & \text{if } \exists i\ \mathrm{z}_i = 1 \\ 1 & \text{if } \forall i\ \mathrm{z}_i = 0 \\ \chi & \text{otherwise} \end{cases} \tag{6}$$

and hence are dubbed *NOR networks*. Clearly, any network with a state transition mapping that has a Boolean model can be represented by a NOR network having an equivalent Boolean model.

3. Mutual inhibition networks

In many biological neural networks, *mutual inhibition* (the generalized case of recurrent inhibition) between neurons similar to that shown in Fig. 1 plays a key role. In the discussion to follow, we assume that all units have quasi-tonic excitatory inputs but we will only depict the inhibitory inputs that are the focus of our discussion. Although the two units in this network have a common external input x_1, the network is not of the form Eq. (1), but it suffices to consider the network behavior when x_1 is inactive. Recurrent inhibition in combination with other network features is identified frequently as the mechanism underlying oscillatory dynamics in many small invertebrate neural networks (e.g. the *Tritonia d.* swim CPG Getting, 1988). Recurrent inhibition has also been found in structures of the human brain such as the neostratium (Wilson, 1990) and the pyriform cortex (Haberly, 1990) where it frequently is proposed as a mechanism for enhancing the difference between neurons with high and low firing rates.

Most, if not all, of the previous work on mutual inhibition has focused on the autonomous dynamics produced by this configuration. In contrast, this section shows that under certain circumstances mutual inhibition can also produce stochastic dynamics. Although the results are rigorous for the SPNN model studied here, it remains an open question whether biological

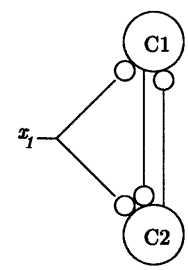

Fig. 1. Mutual inhibition network.

instances have the necessary characteristics to exhibit stochastic dynamics.

The mutual-inhibition network has three fixed points:

$$\check{z}^{\infty} \in \{01,\ 10,\ \chi\chi\}$$

According to Theorem 2.1, the binary fixed points $\check{z} = 01$ and $\check{z} = 10$ correspond to asymptotically stable equilibria of the network if the gain ξ is large enough. Similarly, Theorem 2.2 demonstrates that any equilibrium corresponding to the fixed point $\check{z} = \chi\chi$ is unstable for large enough values of ξ. A version of a theorem by Roska et al. (1992), based on a more powerful result by Smith (1987), shows that the non-constant periodic limit set of the network is unstable.

An informal argument suggests how noise affects the dynamics of a mutual-inhibition network. Because the non-constant periodic limit cycle is unstable, relatively low intensity noise is sufficient to drive the network trajectory into the attraction basin for one of the two asymptotically stable equilibria. These basins are relatively deep so that the trajectory will eventually enter a small neighborhood of the equilibrium state (perhaps after bounding between the two basins for a short time). Since neither outcome is preferred over the other, the mutual-inhibition network in effect executes a computation analogous to a 'coin-flip' when operated in this mode. Of course, slight differences between the units in a biological or artificial network will bias the coin-flip but this will not effect the complexity results presented below.

4. A stochastic oscillator

The probabilistic behavior of mutual-inhibition pairs is just an idle curiosity unless it can be shown that larger networks incorporating these pairs can have dynamics, which express their probabilistic behavior in a meaningful way. If the network trajectory is viewed as a computation, as this work proposes, then the dynamics must induce a 'coin-flip' at appropriate points in the trajectory so that the entire trajectory represents a meaningful probabilistic computation. Fig. 2 depicts the simplest member in one class of SPNNs that meets this requirement. As the rest of this section shows, this network has a well-defined but stochastic trajectory. Using this result, the following sections show that networks in this class have dynamics, which can be consistently interpreted as probabilistic digital computation.

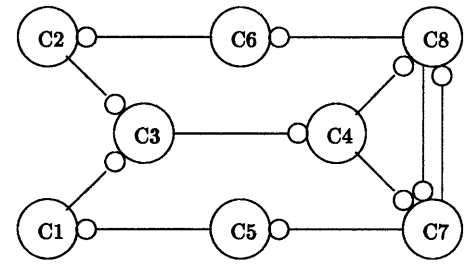

Fig. 2. The stochastic oscillator.

The behavior of the net in Fig. 2 can be informally understood by recognizing that units C1–C6 form a two-input OR gate, the inputs of which are driven by the outputs of the C7, C8 mutual-inhibition pair and the output in turn drives the input of the C7, C8 pair. Now, suppose initially that the input to the C7, C8 pair goes active forcing both units in the pair to go inactive. This in turn drives the output of the C1–C6 OR-gate active. At this point both units in the C7, C8 pair start to go active and, if the network were deterministic, both units eventually would reach a high activity level. Since the feedback path through the C1–C6 OR-gate involves four units, the feedback delay is significantly longer that the unit time constants and both units in the C7, C8 pair would inhibit each other before the feedback would drive the C1–C6 OR-gate active. At that point both units would again go active and the cycle $\ldots 00 \rightarrow 11 \rightarrow 00\ldots$ would repeat ad infinitum. However, the previous section shows this limit cycle is unstable in the SPNN; this implies that the C7, C8 pair converges towards a neighborhood of one of the two asymptotically stable equilibrium. Since neither equilibrium is preferred over the other, the network makes the probabilistic transition $00 \rightarrow 10$ or $00 \rightarrow 01$. Either case forces the C1–C6 OR-gate active which, in turn, forces the C7, C8 pair inactive and the entire cycle just described begins anew. Since this pattern repeats ad infinitum, the

network functions as a stochastic oscillator, or in computational terms, as a repeated 'coin-flip'.

5. Generalizing the stochastic oscillator

As mentioned earlier, the stochastic oscillator is the simplest member in a class of networks whose dynamics represent meaningful probabilistic computations. This section focuses on generalizing the stochastic oscillator and characterizing the equilibris of the resulting networks. Using those results, the remainder of the paper investigates the computational interpretation for the dynamics of these networks.

To frame the proposed classes of networks, observe that the stochastic oscillator includes one mutual-inhibition pair and two (disjoint) feedback paths as defined by the number of inputs to unit C3. In addition, observe that the outputs of both units in the mutual-inhibition pair feedback to C3 and that the feedback paths are disjoint. A natural generalization of this network then is the network with p mutual-inhibition pairs and q non-disjoint feedback paths. Each unit receives no external input except for the constant tonic input so subsequent analyses can omit explicit references to the network inputs. Throughout the sequel, the family of SPNNs whose connection matrices meet these requirements will be denoted by $\mathscr{S}$. Furthermore, it simplifies the following discussion to define the ternary variables

$$\vec{z} = \vec{g}(\vec{h}(\vec{y}))$$

where $\vec{g}(\vec{z})$is the ternary quantization function for the network, and then partition $\vec{z}$ into several subvectors:

$$c_1, \ldots, c_q = z_1, \ldots, z_q \tag{7}$$

$$d = z_{q+1} \tag{8}$$

$$w = z_{g+2} \tag{9}$$

$$a_1, b_1, \ldots, a_p, b_p = z_{g+3}, \ldots, z_{2p+q+2} \tag{10}$$

$$u_1, v_1, \ldots, u_p, v_p = z_{2p+q+3}, \ldots, z_{4p+q+2} \tag{11}$$

The discussion in the rest of this paper refers to these variables, the state vector $\vec{y}$, and the quantized state vector $\vec{z}$ synonymously, choosing in any particular situation the one which best facilitates the analysis.

5.1. Equilibria of networks in $\mathscr{S}$

As in the previous examples, the significant dynamics of the networks in $\mathscr{S}$ can be inferred from the equilibria. For these networks to be useful computational devices, only the binary equilibria should be asymptotically stable. The next theorem shows this is the case; the key idea used in the proof is a simple lemma on nonnegative matrices:

Lemma 5.1. *A nonnegative $n \times n$ matrix A is not nikpotent iff the corresponding graph has a cycle.*

From this lemma and the structure of connection matrices which arise in the networks in $\mathscr{S}$, it follows:

Theorem 5.1. *There exists a ξ_0 such that, for any gain $\xi > \xi_0$, any equilibrium state $\vec{y}^\infty$ of a network $N \in \mathscr{S}$ is asymptotically stable if and only if*

$$\vec{z}^\infty = \vec{g}(\vec{h}(\vec{y}^\infty)) \in \mathscr{B}^n \tag{12}$$

The most immediate and useful consequence of this theorem is that all of the asymptotically stable equilibria of any network in $\mathscr{S}$ can be located by finding the fixed points of the Boolean model $\vec{\Psi}(\vec{z})$ for the network. Recall that for networks in $\mathscr{S}$, each component $\Psi_i(\vec{z})$ of the state transition is just a ternary NOR function (complemented disjunctive clause). Using the variables defined in Eqs. (7)–(11), any equilibrium must satisfy the set of Boolean expressions

$$u_i = \overline{w \vee v_i} \quad i = 1, \ldots, p$$

$$v_i = \overline{w \wedge u_i} \tag{13}$$

$$a_i = \overline{u_i} \quad i = 1, \ldots, p$$

$$b_i = \overline{v_i} \tag{14}$$

$$c_j = \Psi_j(\vec{a}, \vec{b}) \quad j = 1, \ldots, q \tag{15}$$

$$d = \overline{\bigvee_{j=1,\ldots,q} c_j} \tag{16}$$

$$w = \bar{d} \tag{17}$$

where each Ψ_j is a complemented disjunctive clause. As a consequence of the proof of Theorem 5.1 in Hangartner (1994), $d = 1$ and $w = 0$ for any asymptotically stable equilibrium. This implies that $c_j = 0$ for all j and that $u_i = \bar{v}_i$ (and therefore, $a_i = \bar{b}_i$) for all i. Since each Ψ_j is complemented disjunctive clause in the variables u_i and their negations, the asymptotically stable equilibria correspond one for one with the satisfying assignments of the CNF Boolean formula:

$$f(\vec{u}) = \bigwedge_{j=1,\ldots,p} \bar{\Psi}_j(\vec{a}, \bar{\vec{a}}) \tag{18}$$

since $\vec{a} = \bar{\vec{u}}$ and $\vec{b} = \bar{\vec{v}} = \vec{u} = \bar{\vec{a}}$.

5.2. Probabilistic computation by networks in $\mathscr{S}$

With the equilibria characterized, the dynamics of networks in $\mathscr{S}$ can now be informally interpreted. Drawing on the discussion in Section 4 of the stochastic oscillator and the discussion of Theorem 5.1, this section investigates the asynchronous switching network (ASN) (Brzozowski and Seger, 1989) models for the networks in $\mathscr{S}$. The results show that these networks can be viewed as devices, which effect robust probabilistic digital computation.

As the discussion in Section 4 and of Theorem 5.1 suggests, the ASN model for the dynamics of these networks can be understood by assuming first that $w = 1$. This forces the units in the mutual-inhibition pairs to be reset, i.e. $\vec{u} = \vec{v} = \vec{0}$. As a result, $\vec{a} = \vec{b} = \vec{1}$, $\vec{c} = \vec{0}$, and $d = 1$. This causes $w = 0$ releasing the reset condition on the mutual-inhibition pairs.

The results in Section 3 show that at this point each mutual-inhibition pair will execute a 'coin-flip'. Suppose first that as a result of this probabilistic operation the vectors $\vec{u}$ and $\vec{v}$ are such that $\vec{c} = \vec{1}$ for at least one j. This causes both d and w to toggle, i.e. $d = 0$ and $w = 1$; and the entire cycle repeats. On the other hand, if the probabilistic operation regults in a choice of vectors $\vec{u}$ and $\vec{v} = \bar{\vec{u}}$ such that $\vec{c} = \vec{0}$, then the values $d = 1$ and $w = 0$ remain unchanged. This means that the mutual-inhibition pairs hold their current value and the network has settled to a stable-state, i.e. a satisfying assignment for the CNF Boolean formula Eq. (18).

Thus each cycle of the network represents a probabilistic digital computation with two steps. The first step represents a probabilistic guess in the sense that the outcome of the 'coin-flip' is determined by the stastistics of the noise in the network. The second step checks the guess and halts the computation if it is correct, otherwise the guess–check cycle starts anew. If the 'coin-flips' are unbiased and if the CNF corresponding to a network has n variables and m satisfying assignments, then the probability that the network trajectory settles to a neighborhood of an asymptotically stable equilibria corresponding to a satisfying assignment in k-iterations is:

$$p(k) = 1 - \left(\frac{2^n - m}{2^n}\right)^k$$

Straightforward manipulation of this expression shows that the probability that the network has not converged after n^l iterations is less than p where

$$l > \frac{\log\log 1/(1-p) - \log(n - \log(2^n - m))}{\log n} \tag{19}$$

It should be clear that l is constant as n increases if and only if m is a function of n. The sequel will be particularly concerned with networks in which over half the possible assignments are satisfying ($m > 2^{n-1}$) since Eq. (19) shows these networks find a satisfying solution in constant time ($l = \mathcal{O}(1)$) for any probability p.

6. Complexity results for NOR networks

The set of asymptotically stable equilibria of an NOR network in $\mathscr{S}$ corresponds one-to-one with the set of satisfying assignments for the CNF Boolean formula Eq. (18). This section examines the implications of that correspondence using a few ides from probabilistic and nonuniform circuit complexity theory. In particular, it is shown first that there can exist no polynomial time algorithm for finding the equilibria of NOR networks if $P \neq NP$. After that it is shown that the

family $\mathscr{S}$ has the same computational power as a polynomial-time bonded probabilistic Turing machine.

The restrictions on the feedback structure of the members of $\mathscr{S}$ means that $\mathscr{S}$ contains a NOR network instance for every CNF Boolean formula in which each variable and its complement appears in at least one clause but no clause includes both. Since the size of the NOR network is polynomial in the size of the CNF formula, this implies trivially that

Theorem 6.1. *There exists no efficient algorithm for finding the fixed points of the ternary model for NOR networks, and hence the equilibria of NOR networks, if* $P \neq NP$.

The second result, namely that the family $\mathscr{S}$ has the same computational power as a polynomial-time bounded probabilistic Turing machine (PTM), depends on showing that the algorithm executed by $\mathscr{S}$ can be simulated in polynomial time by a PTM and that $\mathscr{S}$ and a PTM can accept the same languages in polynomial time. Recall that Gill (1977) defined a PTM as a machine that operates like a deterministic Turing machine (DTM) with just two differences. First, since the transition function associates two possible outcomes with any current state, if the two outcomes are the same, the transition is *deterministic* and the machine executes the transition just like a deterministic machine, but if the two outcomes are not the same, the machine executes the transition by 'flipping' an un-biased coin to choose one of the two outcomes. The other difference between a PTM and a DTM is that the PTM has a finite running time defined by the constructible clock function $c(n)$. In contrast to a DTM, a PTM always halts after executing $c(n)$ steps. This implies that the computation can be represented as a binary tree of depth $c(n)$. The input is *accepted* after $c(n)$ steps if over half the final states represented by the leaves of the computation tree for the input are *accepting*, the input is *rejected* otherwise.

By convention, the family of languages accepted by polynomial-time bounded PTMs (i.e. $c(n)$ is a polynomial in n) is referred to as *PP*. It is known (Balcázar et al., 1988) that the set MAJ, defined as the set of CNF Boolean formulas satisfied by over half of the possible assignments, is *PP*-complete. Thus, the computational power of any machine which accepts MAJ equals or exceeds the computational power of a PTM.

Now, Eq. (19) shows that the set $\mathscr{S}$ is a family of NOR networks each of which probabilistically computes a satisfying assignment (if one exists) for the associated CNF in finite time. More specifically, if the CNF associated with a particular NOR network in $\mathscr{S}$ has a satisfying assignment, then for any probability p there exists a finite k such that the probability the network has not found a satisfying a satisfying solution after k trials is less than p. This property leads to the following definition for the family of languages accepted by $\mathscr{S}$ for different time bounds:

Definition 6.1. *Let C be an arbitrary time function class and let SC denote the set of languages accepted by* $\mathscr{S}$ *with time-bound C. A set* $\mathscr{F}$ *of CNF Boolean formulas is a language in SC if, for any probability p, there exists a function* $c(n) \in C$ *such that for any* $F \in \mathscr{F}$, *there exists a NOR network* $N(F) \in \mathscr{S}$ *which converges to a satisfying solution of F with probability at least p in time* $c(|F|)$.

Based on this definition, a computational model for $\mathscr{S}$ can be constructed from the family of NOR networks in $\mathscr{S}$. For a given input CNF, the ternary model for the NOR network can be constructed in linear time. The network can then be simulated in linear time by a PTM using the description given in Section 5.2. This model can be reduced even further to the probabilistic **net_SAT** algorithm given below:

```
algorithm net_SAT(𝒱, 𝒞, c(n))
   returns ACCEPT or REJECT
inputs: Boolean variables 𝒱 = {v_1, ..., v_p)
   propositional clauses 𝒞 = Φ_1, ..., Φ_q)
      on variables in 𝒱
   polynomial time constructible
      clock function c(n)
output: binary value ACCEPT or REJECT
local: iteration counter k
   clause values f_1, f_2, ..., f_q
01 for each Φ_j ∈ 𝒞
```

02 $f_j \leftarrow FALSE$;
03 $k \rightarrow 0$;
04 while at least one $f_j = FALSE$
and $k < c(|\mathscr{V}|) + |\mathscr{C}|$ do steps 05–09
05 for each $v_j \in \mathscr{V}$
06 $v_j \leftarrow$ coin_flip(); (*probabilistic step*)
07 for each $\Phi_j \in \mathscr{C}$
08 $f_j \leftarrow \Phi_j(v_1, v_2, \ldots, v_p)$;
09 $k \rightarrow k + 1$
10 if every $f_j = TRUE$
11 return *ACCEPT*;
12 else
13 return *REJECT*;
end net_SAT

Algorithm 1. Probabilistic algorithm modeling $\mathscr{E}$

Obviously, the most interesting class of languages is $\mathscr{S}P$, the class of CNF Boolean formulas accepted by $\mathscr{S}$ in polynomial time. It is straightforward to show that

Theorem 6.2. $\mathscr{S}P = PP$.

As a result since, $\mathscr{N}P \subseteq PP$.

7. Conclusion

This paper has studied the computational capabilities of an interesting family $\mathscr{S}$ of probabilistic neuromime networks proposed as a model for biological neural networks. It was shown that due to the presence of system noise, the NOR networks in $\mathscr{S}$ function as probabilistic computing derives and the family can be viewed as a computing machine where the nth network in the family handles all size n instances of a PP-complete problem. A constructive proof is given, which shows that the nth network has $\mathscr{O}(n)$ space complexity and $\mathscr{O}(1)$ probabilistic-time complexity in the size of the input. Consequently, although this family has the same recognition capability as a deterministic TM, probabilistic features endow S with a space–time advantage over conventional digital computers. This result suggests that mutual-inhibition could be an important computational mechanism in biological neural networks and that biological networks may have a similar space–time advantage over conventional digital computers.

References

Abu-Mostafa, Y.S., 1986. Neural networks for computing? In: Denker, J.S. (Ed.), Neural Networks for Computing. Snowbird 1986. AIP Conf. Proc. 151, 53–58. American Institute of Physics.

Balcázar, J.L., Díaz, J., Gabarró, J., 1988. Structural Complexity I. Springer, Berlin.

Blum, A., Rivest, R.L., 1988. Training a 3-node neural network is NP-complete. In: Touretsky, D.S. (Ed.), Advances in Neural Information Processing, vol. I. Morgan Kaufmann, San Mateo, CA, pp. 494–501.

Bruck, J., Goodman, J.W., 1990. On the power of neural networks for solving hard problems. J. Complexity 6, 129–135.

Brzozowski, J.A., Seger, C.-J., 1989. A unified framework for race analysis of asynchronous networks. J. Assoc. Comput. Machine. 36, 20–45.

Getting, P.A., 1988. Comparative analysis of invertebrate central pattern generators. In: Cohen, A.H., Rossignol, S., Grillner, S. (Eds.), Neural Control of Rhythmic Movements in Vertebrates. Wiley, New York, pp. 101–127.

Gill, J., 1977. Computational complexity of probabilistic Turing machines. SIAM J. Comput. 6 (4), 675–695.

Haberly, L.B., 1990. Olfactory cortex. In: Shepherd, G.M. (Ed.), The Synaptic Organization of the Brain. Oxford University Press, Oxford, pp. 317–345.

Hale, J., 1997. Theory of Functional Differential Equations. Springer, New York.

Hangartner, R.D., 1994. Probabilistic Computation in Stochastic Pulse Neuromime Networks. Ph.D. Dissertation. Computer Science Department, Oregon State University.

Hangartner, R., Cull, P., 1995. A ternary logic model for recurrent neuromime networks with delay. Biol. Cybern. 73, 177–188.

Kleene, S.C., 1956. Representation of events in nerve nets end finite automata. In: Automata Studies (Annals of Math Studies, 34). Princeton University Press, Princeton, NJ.

Maass, W., 1994. Lower bounds for the computational power of networks of spiking neurons. Technical report TR94-019, Electronic Colloquium on Computational Complexity, http:/www.eccc.uni-trier.de/eccc/.

Moore, C., 1993. Real-valued continuous-time computers: a model of analog computation, Part I. Unpublished report, Sante Fe Institute.

Parberry, I., Schnitger, G., 1989. Relating Boltzmann machines to conventional models of computation. Neural Networks 2, 59–67.

Roska, T., Wu, C.W., Balsi, M., Chua, L.O., 1992. Stability and dynamics of delay-type general and cellular neural networks. IEEE Trans. Circuits Syst. — I. Fundam. Theory Appl. 30, 487–490.

Sato, S., 1978. On the moments of the firing interval of the diffusion approximated model neuron. Math. Biosci. 39, 53–70.

Siegelmann, H.T., Sontag, E.D., 1991. On the computational power of neural nets. Report SYCON-91-11. Rutgers University Center for Systems and Control.

Siegelmann, H.T., Sontag, E.D., 1992. Some recent results on computing with neural nets. In: Proceedings of the IEEE INNS international Conference on Decision and Control, pp. 1476–1481.

Smith, H., 1987. Monotone semiflows generated by functional differential equations. J. Differential Equations 66, 420–442.

Turova, T.S., 1997. Stochastic dynamics of a neural network with inhibitory and excitatory connections. BioSystems 40, 197–202.

Vergis, A., Stieglitz, K., Dickinson, B., 1986. The complexity of analog computation. Math. Comput. Simul. 28, 91–113.

Wilson, C.J., 1990. Basal ganglia. In: Shepherd, G.M. (Ed.), The Synaptic Organization of the Brain. Oxford University Press, Oxford, pp. 279–316.

Yao, X., 1992. Finding approximate solutions to NP-hard problems by neural networks is hard. Inform. Process. Lett. 41, 93–98.