# Hopfield Models as Nondeterministic Finite-State Machines

Marc F.J. Drossaers

Computer Science Department,
University of Twente,
P.O Box 217, 7500 AE Enschede,
The Netherlands,
email: mdrssrs@cs.utwente.nl.

## Abstract

The use of neural networks for integrated linguistic analysis may be profitable. This paper presents the first results of our research on that subject: a Hopfield model for syntactical analysis. We construct a neural network as an implementation of a bounded push-down automaton, which can accept context-free languages with limited center-embedding. The network's behavior can be predicted a priori, so the presented theory can be tested. The operation of the network as an implementation of the acceptor is provably correct. Furthermore we found a solution to the problem of spurious states in Hopfield models: we use them as dynamically constructed representations of sets of states of the implemented acceptor. The so-called neural-network acceptor we propose, is fast but large.

## 1 Introduction

Neural networks may be well suited for integrated linguistic analysis, as Waltz and Pollack [10] indicate. An integrated linguistic analysis is a parallel composition of several analyses, such as syntactical, semantical, and pragmatic analysis. When integrated, these analyses constrain each other interactively, and may thus suppress a combinatoric explosion of sentence structure and meaning representations.

This paper presents the first results of our research into the use of neural networks for integrated linguistic analysis: a Hopfield model for syntactical analysis. Syntactical analysis in the context of integration with other analyses boils down to the decision whether a sentence is an element of a language. A parse tree is superfluous here as an intermediary representation, since it will not be finished before the complete integrated analysis is. This fact allows us to deal with the problem of a restricted length of the sentences a neural parser can handle, see e.g. [5],[7], a problem that could not be elegantly solved, see e.g. [6],[3].

In this paper we propose a formal model that recognizes syntactically correct sentences (section 2), a Hopfield model onto which we want to map this formal model (section 3), the parameters that makes the network operate as intended (section 4), and a way to map the formal model onto the Hopfield model, including a correctness result for the latter (section 5). The theoretically predicted behavior of the so-obtained network has been verified, and a simple example provides the taste of it (section 6). We also consider complexity aspects of the model (section 7). Section 8 consists of concluding remarks.

## 2 A Bounded Push-Down Automaton

Although it is not an established fact, it is assumed here that natural languages are context-free, and consequently that sentences in a natural language can be recognized, by a push-down automaton (PDA). However, we are not interested in modeling the competence of natural language users, but in modeling their performance. The human performance in natural language use is also characterized by a very limited degree of center-embedding. In terms of PDAs this means that there is a bound on the number of items on the stack of a PDA for a natural language. A bounded push-down automaton $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is a PDA that has an upper limit $k \in \mathbb{N}$ on the number of items on its stack, i.e. $|\sigma| \leq k$ for every instantaneous description (ID) $(q, w, \sigma)$ of $M$. The set of stack states of this PDA is defined to be: $Q_{ST} = \{\sigma \mid (q_0, w, Z_0) \vdash_M^* (q, \varepsilon, \sigma)\}$. $Q_{ST}$ is finite: $|Q_{ST}| \leq (|\Gamma|)^k$, therefore we may define a nondeterministic finite-state acceptor (NDA) $M'$ that has $Q_{ST}$ as its set of states.

The class of PDAs of which we would like to map bounded versions onto NDAs is constrained, among others to the class of $\varepsilon$-free PDAs. By this constraint we anticipate the situation that grammars are stored in a neural network by self-organization. In that situation a neural network will store $\varepsilon$-productions only if examples of applications of $\varepsilon$-productions are repeatedly presented to it. This requires $\varepsilon$ to have a representation in the machine, in which case it fails to accommodate its definition.

Another restriction we would like to introduce is to grammars in 2-standard form with a minimal number of quadratic productions: productions of the form

$A \rightarrow bCD$ where $b$ is a terminal and $C$ and $D$ are variables. Such a grammar can be seen as a minimal extension of a right-linear grammar. Within such grammars, quadratic productions provide for the center-embedding. Since such grammars have a minimal number of quadratic productions, acceptance by a PDA defined for such grammars requires a minimal use of (stack) memory, and thus generates a minimal $Q_{ST}$. To maintain this minimal use of memory a restriction to one-state PDAs that accept by empty stack is also required: when a PDA is mapped onto an NDA, the information concerning its states is lost, unless it was stored on the stack.

An $\varepsilon$-free, one-state PDA that simulates a context-free grammar in 2-standard form with a minimal number of quadratic productions (and that accepts by empty stack) satisfies all our criteria. For every such PDA we can define an NDA, for which we can prove [4] that it accepts the same language as the PDA does.

**Definition 2.1**
Let $M = (\{*\}, \Sigma, \Gamma, \delta, *, Z_0, \emptyset)$ be an $\varepsilon$-free bounded PDA with bound $k$. *An NDA defined for $M$* is an NDA $M' = (Q', \Sigma', \delta', Q'_0, F')$ such that:
$Q' = Q_{ST}$, as defined above;
$\Sigma' = \Sigma$;
$\delta' : Q_{ST} \times \Sigma \rightarrow 2^{Q_{ST}}$ is defined by:
$\delta'(\sigma_1, x) = \{\sigma_2 \mid (*, xw, \sigma_1) \vdash_M (*, w, \sigma_2)\}$
$Q'_0 = \{Z_0\}$; and
$F' = \{\varepsilon\}$ (empty stack).

**Theorem 2.2** *(correctness of the NDA)*
*Let $M$ be an $\varepsilon$-free one-state PDA with bound $k$, if $M'$ is an NDA as defined in definition 2.1, then $M$ accepts a string by empty stack if and only if $M'$ accepts it by accepting state.*

In as far as a natural language is context-free, we claim that there is an instance of our acceptor that recognizes it.

## 3 An Input-Driven Sequencing Hopfield Model

In this section a noiseless Hopfield model is proposed that is tailored to implement NDAs on. The model is based on the associative memory described by Buhmann et al. [2] and the theory of delayed synapses from [8]. We chose the Hopfield model because of its analytical transparency, and its capability of sequence traversing, which agrees well with the sequential nature of language use at a phenomenological level. The Hopfield model proposed is a memory for temporal transitions extended with external-input synapses. Figure 1 shows the architecture involved.

In this network only those neurons are active upon which a combined local field operates that transcends the threshold. The activity generated by such a local field is the active overlap of the temporal image of past activity provided by so-called temporal synapses, and the image of input external activity provided by so-called input synapses. By the temporal synapses, this activity will later generate another (subthreshold) temporal image, so network activity may be considered a transition mechanism that brings the network from one temporal image to another. Active overlaps are unique with high probability if the activity patterns are chosen at random and represent low mean network activity. This uniqueness makes the selectivity of the network very plausible: if an external activity pattern is presented that does not match the current temporal image, then there will not be activity of any significance; the input is not recognized.

When an NDA is mapped onto this network, pairs of NDA-state $q$ and input-symbol $x$, such that $\delta(q, x) \neq \emptyset$, are mapped onto activity patterns. Temporal relations in the network then serve to implement NDA transitions. Note that single NDA transitions are mapped onto single network transitions. This results in complex representations of the NDA states and the input symbols. An NDA state is represented by all activity patterns that represent a pair containing that state, and input patterns are represented by a component-wise OR over all activity patterns containing that input symbol. A consequence is that mixed temporal images, the subthreshold analogue of mixture states, are a very natural phenomenon in this network, because the temporal image of an active overlap comprises at least all activity patterns representing a successor state. But this is not all. Also the network will act as if it implements the deterministic equivalent of the NDA, i.e. it will trace all paths through state space the input allows for, concurrently. The representations of the states of this deterministic finite-state automaton (FSA) are dynamically constructed along the way; they are mixed temporal images. The concept of a "dynamically constructed representation" is borrowed from Touretzky [9], who, by the way, argued that they could not exist in the current generation of neural networks, such as Hopfield models.

A time cycle of the network can be described as follows:

1. The network is allowed to evolve into a stable activity pattern that is the active overlap of a temporal image of past activity, and the input image of external input for a period $t_r$ (= relaxation time), when an external activity pattern is presented to the network;

2. After some time the network reaches a state of stable activity and starts to construct a new temporal image. It is allowed to do this for a period $t_a$ (= active time);

3. Then the input is removed, and the network evolves towards inactivity. This takes again about a period $t_r$;

4. Not before a period $t_d$ (= delay time) has passed, a new input arrives. The new temporal image is forwarded by the slow synapses during a period $t_a + t_r$, starting when $t_d$ ends. The slow synapses have forgotten the old temporal image while the network was in its $t_d$.

The synapses modeled in the network collect the incoming activity over a period $t_a + t_r$, and emit the time average over again a period $t_a + t_r$ after having waited a period $t_d + t_r$. In the network this is modeled by computing a time average over prior neuronal activity, and then multiply it by the synaptic efficacy. The time average ranges over a period $(2t_a + t_d + 3t_r)/N - (t_a + t_r)/N$. The first argument is the total time span in the network, covering two active periods and an intervening delay time, including their transition times. The second argument is the current period of network is activity, activity that cannot directly interfere with the network's dynamics.

More formally the network can be described as follows:

$$
\begin{aligned}
S_i &\in \{0,1\}, \text{ where } i = 1,\ldots,N, \\
S_i(t+1) &= \begin{cases} 1 & \text{if } h_i(t) > U \\ 0 & \text{if } h_i(t) \le U, \end{cases} \\
h_i(t) &= h_i^e(t) + h_i^t(t), \\
h_i^t(t) &= \sum_{j=1}^{N} J_{ij}^t \overline{S}_j(t), \\
&\quad \text{the temporal transition term,} \\
J_{ij}^t &= \frac{\lambda^t}{a(1-a)N} \sum_{\mu=1}^{p} (\xi_i^{\mu+1} - a)(\xi_j^\mu - a), \\
&\quad \text{with } J_{ii} = 0, \\
\overline{S_j}(t) &\equiv \frac{\tau - \theta}{\theta} \int_0^\infty S_j(t - t')w(t')dt', \text{ where} \\
w(t) &= \begin{cases} \frac{1}{\tau - \theta} & \text{if } \theta < t < \tau \\ 0 & \text{otherwise ,} \end{cases} \\
h_i^e(t) &= \lambda^e(S_i'(t) - a), \\
&\quad \text{the external input term,} \\
J_{ij}^e &= \lambda^e \delta_{ij}.
\end{aligned}
$$

The $S_i$ are neuronal variables ($S_i'$ is a neuron in another network), $h_i$ is the total input on $S_i$, $U$ is a threshold value which is equal for all $S_i$, $J_{ij}$ is the synaptic efficacy of the synapse connecting $S_j$ to $S_i$, and $\lambda$ is the relative magnitude of the synapses. The average at time $t$ is expressed by $\overline{S}_j(t)$, where $\tau \equiv (2t_a + t_d + 3t_r)/N$ and $\theta \equiv (t_a + t_r)/N$. The function $w(t)$ determines over which period activity is averaged. The input synapses are nonzero only in case $i = j$. These synapses carry a negative ground signal $-\lambda^e a$, which is equivalent to an extra threshold generated by the input synapses. The activity patterns $\{\xi^\mu\}$ ($\{\xi^\mu\} \equiv (\xi_1^\mu, \xi_2^\mu, \ldots, \xi_N^\mu)^T$) are statistically independent, and satisfy the same probability distribution as the patterns in the model of Buhmann et al. [2]:

$$
P(\xi_i^\mu) = a\delta(\xi_i^\mu - 1) + (1-a)\delta(\xi_i^\mu), \text{ where}
$$
$$
\delta(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise.} \end{cases}
$$

If $a \neq \frac{1}{2}$ the pattern is biased. For $N \to \infty$, $1/N \sum_{i=1}^{N} \xi_i^\mu \to a$. The updating process is a Monte Carlo random walk.

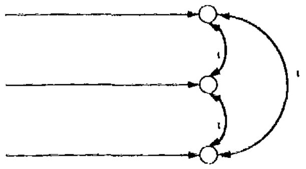


Figure 1: *The model for $N = 3$. Usually Hopfield models consist of very many neurons. The arced arrows denote temporal synapses. The straight arrows denote input synapses.*

## 4 Estimation of Parameters

A number of system parameters need to be related in order to make the model work correctly.

Timing is fairly important in this network. The time the network is active ($t_a$) should not exceed the delay time $t_r$. If it does then $t_a + t_r > t_d + t_r$, and since no average is computed over a period $t_a + t_r$ back in time, not the full time average of the previous activity need to be computed, consequently we choose $t_a \le t_d$. The choice for a transition time $t_r$ depends on the probability with which one requires the network to get in the next stable activity state. This subject will be dealt with in section 7.

In the estimates of $\lambda^t$ and the storage capacity below, an expression of the temporal transition term in terms of the overlap parameter $m^\mu$ is used, which will be introduced here first. The overlap parameter $m^\mu(t)$ measures the overlap of a network state $\{S\} \equiv (S_1, S_2, \ldots, S_N)^T$ at time $t$ with stored pattern $\{\xi^\mu\}$, and is defined by:

$$
m^\mu(t) = \frac{1}{aN} \sum_{i=1}^{N} (\xi_i^\mu - a)S_i(t).
$$

$m^\mu \in [-a, 1-a]$. The expression for the temporal transition term is:

$$
h_i^t(t) = \sum_{j=1}^{N} J_{ij}^t \overline{S}_j(t).
$$

Assuming that $N \to \infty$ while $p$ remains fixed this is, after expansion of the $J_{ij}$ and ignoring infinitesimal terms, approximated by:

$$\begin{aligned} h_i^t(t) &= \frac{\lambda^t}{a(1-a)N}\sum_{\mu=1}^{p}(\xi_i^{\mu+1}-a)\sum_{j=1}^{N}(\xi_j^{\mu}-a)\overline{S}_j(t) \\ &= \frac{\lambda^t}{(1-a)}\sum_{\mu=1}^{p}(\xi_i^{\mu+1}-a)\overline{m}^{\mu}(t), \text{ where} \end{aligned}$$

$$\overline{m}^{\mu}(t) \equiv \frac{\tau-\theta}{\theta}\int_0^{\infty} m^{\mu}(t-t')w(t')dt'.$$

If the temporal image is $\{\xi^{\nu}\}$ then $h_i^t$ is about ($N \to \infty$):

$$h_i^t(t) = \frac{\lambda^t(\xi_i^{\nu+1}-a)(\tau-\theta)}{\theta}\int_0^{\infty} w(t)dt.$$

If a number of patterns in a mixture state have the same successor, that pattern may be activated. To prevent this $\lambda^t$ will be chosen such that the slow synapses do not induce activity in the network autonomously, not even if all the neurons in the network are active. On average, the activity in the network is given by the parameter $a$. The total activity in a network is a quantity $x$ such that $x = 1/a$, so what we require is that $xh_i^t < U$, i.e. that:

$$\frac{\lambda^t(\xi_i^{\nu+1}-a)(\tau-\theta)}{a\theta}\int_0^{\infty} w(t)dt < U.$$

The interesting case is $\xi_i^{\nu+1} = 1$. Since the integral is at most $\theta/(\tau-\theta)$ which is the strongest condition on the left side, the left expression can be written as $\lambda^t(1-a)/a$. It was earlier demanded that only a combined local field can transcend the threshold, which implies that external input $\lambda^e(1-a) < U$, so we can take $\lambda^t \leq \lambda^e a$ safely. This is small because $a$ is small.

Next a value for the threshold that optimizes storage capacity is estimated by signal-to-noise ratio analysis, following [1] and [2], for $N, p \to \infty$. Temporal effects are neglected because they effect signal and noise equally. It is also assumed that external input is present, so that the critical noise effects can be studied. In this model the external input synapses do not add noise, they do not contain any information apart from the magnitude of the incoming signal. Now suppose the system is in state $\{S\} = \{\xi^1\}$. The signal is that part of the input that excites the current pattern:

$$S = \lambda^t(\xi_i^1 - a).$$

The noise is the part of the input that excites other patterns. It can be seen as a random variable with zero mean and it is estimated by its variance: $\lambda^t\sqrt{\alpha a}$ where $\alpha = p/N$. We want that given the right input $h_i > U$, if both the temporal and the external input excite $S_i$, and that $h_i < U$ if the temporal input does not excite $S_i$. This gives signal-to-noise ratios:

$$\begin{aligned} \rho_1 &= \frac{(\lambda^e+\lambda^t)(1-a)-U}{\lambda^t\sqrt{\alpha a}}, \text{ and} \\ \rho_0 &= \frac{U+\lambda^t a-\lambda^e(1-a)}{\lambda^t\sqrt{\alpha a}}. \end{aligned}$$

Recall is optimal in case $\rho_0 = \rho_1$ which is true for a threshold:

$$U_{opt} = \lambda^e(1-a)+\lambda^t(\tfrac{1}{2}-a).$$

Substituted in either $\rho_0$ or $\rho_1$ it results in $\rho_{opt} = \frac{1}{\sqrt{4\alpha a}}$. This result is the same as obtained by Buhmann et al. [2], and they found a storage capacity $\alpha_c \approx -(a\ln a)^{-1}$ where $\alpha_c = p_{max}/N$. The storage capacity is large for small $a$, so $a$ will be chosen $a << 0.5$. A last remark concerns an initial input for the network. In case there has been no input for the network for some time, it does not contain a temporal image anymore, and consequently has to be restarted. This can be done by preceding a sequence by an extra strong first input, a kind of warning signal of magnitude e.g. $\lambda^e + \lambda^t$. This input creates a memory of previous activity and serves as a starting point for the temporal sequences stored in the network.

## 5 Neural-Network Acceptors

In this section it is shown how NDAs from definition 2.1 can be mapped onto networks as described in sections 3 and 4. Such networks can only be used for cyclic recognition runs. Where "cyclic" indicates that both the initial and the accepting state of an NDA are mapped onto the same network state. If this were not done, the accepting state is not assigned an activity vector, since no transition departs from it, see definition 5.2 below. Cyclic recognition in its turn can only be correctly done for grammars that generate end-of-sentence markers. Any grammar can be extended to do this.

An NDA is related to a network by a parameter list.

**Definition 5.1**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA. *A parameter list defined for an NDA M* is a list of the form $(a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$, where:

1. $a \in [0,1] \subset \mathbb{R}$;
2. $t_a \leq t_d$, with $t_a, t_d \in \mathbb{N}$;
3. $\lambda^e \in \mathbb{R}^+$ where $\mathbb{R}^+ = \{x \mid x \in \mathbb{R} \wedge x > 0\}$;
4. $0 < \lambda^t \leq \lambda^e a$;
5. $p = \sum_{q,q' \in Q} \mid \{x \in \Sigma \mid q' \in \delta(q,x)\} \mid \times \mid \{y \in \Sigma \mid \delta(q',y) \neq \emptyset\} \mid$;
6. $p_{max} \geq p$;
7. $N \geq (-a\ln a)p_{max}$;

8. $t_r$: see section 7;

9. $U = \lambda^e(1-a) + \lambda^t(\frac{1}{2} - a)$.

Note that there are an infinite number of parameter lists for each NDA.

The mapping of an NDA onto a network starts by mapping basic entities of the NDA onto activity patterns. Such a pattern is called a code.

**Definition 5.2**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1. The *coding function* $c$ is given by:

$$c \quad : \quad Q \times \Sigma \rightarrow \{0,1\}^N,$$

such that for $q \in Q$, and $x \in \Sigma$:

1. if $\delta(q, x) \neq \emptyset$:

   $$c(q, x) \quad = \quad \{\xi\},$$

   where $\xi_i$ is chosen at random from $\{0,1\}$ with probability distribution

   $$P(\xi_i) \quad = \quad a\delta(\xi_i - 1) + (1-a)\delta(\xi_i), \text{ and}$$

2. undefined otherwise.

The set of codes is then partitioned: into sets of activity patterns corresponding to NDA states, and into sets of patterns corresponding to input symbols.

**Definition 5.3**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1.
*The set $P_q$ of activity patterns for $q \in Q$* is:

$$P_q \quad = \quad \{c(q, x) \mid x \in \Sigma\}.$$

*The set $P_x$ of activity patterns for $x \in \Sigma$* is:

$$P_x \quad = \quad \{c(q, x) \mid q \in Q\}.$$

Then a network transition is defined as a matrix operator specified by the network's storage prescription, and related to NDA transitions using the previously defined partition of the set of codes.

**Definition 5.4**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1. and let $\boldsymbol{a}$ be an $N$-dimensional vector, with each component $a$. *The set $Tr$ of network transitions* is:

$$Tr \quad = \quad \{J^t_{(q',q,x)} \mid J^t_{(q',q,x)} = \frac{\lambda^t}{a(1-a)N} \sum_{c(q',y) \in P_{q'}} (c(q', y) - \boldsymbol{a})(c(q, x) - \boldsymbol{a})^T \wedge q' \in \delta(q, x)\},$$

where each $J^t$ is an $N \times N$ matrix.

This suffices to define a neural-network acceptor.

**Definition 5.5**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1. *A **neural-network acceptor (NNA)** defined for an NDA $M$* that takes its parameters from $P$ is a quadruple $H = (T, f, U, S)$, where:

1. the topology $T$, a list of neurons per layer, is: $(N)$ ;

2. the activation function $f$ is given by: $S_i = \begin{cases} 1 & \text{if} \quad \sum_j J^t_{ij}\overline{S}_j + \lambda^e(S'_i - a) > U \\ 0 & \text{if} \quad \sum_j J^t_{ij}\overline{S}_j + \lambda^e(S'_i - a) \leq U. \end{cases}$

3. the update procedure is a Monte Carlo random walk.

4. the synaptic coefficients are given by:

   $$J^t \quad = \quad \sum J^t_{(q',q,x)} \in Tr, \text{ and}$$
   $$J^e \quad = \quad \lambda^e I \text{ where } I \text{ is the identity matrix.}$$

In order to construct activity patterns that can serve as external input $x$ for the network a component-wise OR operation is performed over the set $P_x$ as defined in definition 5.3.

**Definition 5.6**
*The OR operation over a set $P_x$ of activity patterns* is specified by:

1. $\text{OR}(\{\xi^\mu\}, \{\xi^\nu\}) = \{\text{OR}(\xi^\mu_i, \xi^\nu_i)\}$ ;

2. $\text{OR}(\{\xi^1\}, \ldots, \{\xi^n\}) = \text{OR}(\{\xi^1\}, \text{OR}(\{\xi^2\}, \ldots, \{\xi^n\}))$ ; and

3. $\text{OR}(P_x) = \text{OR}(\{\xi^1\}, \ldots, \{\xi^n\})$ if $\cdot P_x = \{\{\xi^1\}, \ldots, \{\xi^n\}\}$.

At last a *formal* definition can be given of a temporal image as the set of all activity patterns for which there is a network input that makes such an activity pattern the network's next quasi-stable state.

**Definition 5.7**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1, and let $H = (T, f, U, S)$ be an NNA defined according to definition 5.5 that takes its parameters from $P$. *A **temporal image*** is a set:

$$\{c(q, x) \mid \text{ input OR}(P_x) \text{ for } H \text{ implies } m^{c(q,x)} = 1 - a\},$$

a set $P_{q'}$ is ***a temporal image of a quasi-stable state*** $\{S\} = c(q, x)$ *of* $H$ if and only if $J^t_{(q',q,x)}$ is a transition of $H$.

Now that we have a neural-network acceptor, we may also want to use it to judge the legality of strings against a given grammar with it.

**Definition 5.8**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1, let $H = (T, f, U, S)$ be an NNA defined according to definition 5.5 that takes its parameters from $P$, and let $a_i \in \Sigma$, $q \in Q_0$, and $q' \in F$. *H is said to accept a string* $w = a_1 \cdots a_n$ if and only if $H$ evolves through a series of temporal images that ends at $P_{q'}$ if started at $P_q$, if $\mathrm{OR}(P_{a_1}), \ldots, \mathrm{OR}(P_{a_n})$ appears as external input for the network.

Next the correctness of an NNA is to be proven. Since an NNA is essentially a stochastic machine, this raises some extra problems. What we propose is to let various network parameters approach values for which the NNA has only exact properties, and prove that the network is correct in the limit. Those "various network parameters" are defined below.

**Definition 5.9**
Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definition 5.1. *A list of large parameters* is a parameter list $P$ such that:

1. $a \equiv c_1/N$ where $c_1 << N$ is a constant;
2. $\lambda^e \equiv c_2 N/c_1$, where $c_2$ is a small constant;
3. $0 < \lambda^t \leq \lambda^e a$, i.e. $0 < \lambda^t \leq c_2$;
4. $p_{max} \equiv -(a \ln a)^{-1} N$;
5. $N \rightarrow \infty$;
6. $t_r/N \rightarrow \infty$.

The following lemma states that for neural-network acceptors that take their parameters from a list of large parameters both the probability that the network reaches the next stable state within relaxation time, and the probability that only the patterns that are temporally related to the previous activity pattern will become active, tend to unity. Essentially it means that such networks operate exactly as prescribed by the synapses. Such networks are intrinsically correct.

**Lemma 5.10** *intrinsical correctness*
*Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definitions 5.1 and 5.9, and let $H = (T, f, U, S)$ be an NNA defined according to definition 5.5 that takes its parameters from $P$, then $H$ is such that:*

1. *for all neurons $S_i$ in $H$, $P(S_i$ is selected$) \rightarrow 1$ during network evolution; and*
2. *for all activity patterns $\{\xi\} \in \bigcup P_y$, $y \in Q \cup \Sigma$, if $\mu \neq \nu$, then $P(\xi_i^\mu = \xi_i^\nu = 1) \rightarrow 0$, where $i = 1, \ldots, N$.*

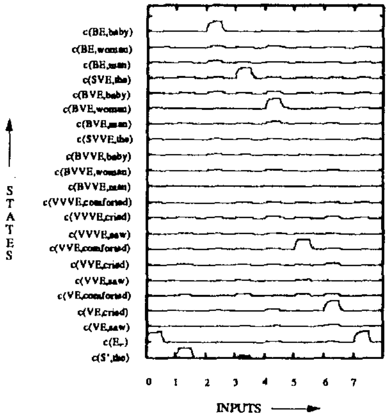
Then the correctness of an NNA follows.

**Theorem 5.11** *(correctness of the NNA)*
*Let $M = (Q, \Sigma, \delta, Q_0, F)$ be an NDA, let $P = (a, \lambda^e, \lambda^t, N, p, p_{max}, t_a, t_d, t_r, U)$ be a parameter list defined according to definitions 5.1 and 5.9, let $H = (T, f, U, S)$ be an NNA defined according to definition 5.5 that takes its parameters from $P$, and let $w \in \Sigma^+$, then the probability that $M$ accepts a string $w$ if and only if $H$ accepts $w$, tends to unity.*

The proof of the theorem is given in [4].

## 6 Simulation Results

As an example we constructed an NNA that accepts the language generated by a grammar with productions:
$S' \rightarrow$ the$BE$,
$B \rightarrow$ man$SV$ | woman$SV$ | baby$SV$ | man$V$ | woman$V$ | baby$V$,
$S \rightarrow$ the$B$,
$E \rightarrow \bullet$,
$V \rightarrow$ saw | cried | comforted.
It takes its parameters from the list: (0.05,1.5,0.07,800,64,6.68N,5,5,5,1.46). It was tested with the sentence ". the baby the woman comforted cried ." The preceding full stop is a first input that awakens the network. The graph below shows the time evolution of the network.



*Plot of the system dynamics.*

## 7 Complexity Aspects

If a neural-network acceptor has to process a sequence of $n$ input patterns, it (worst case) first has to construct its initial temporal image, when awakened by an initial input (that is not considered a part of the sequence), and then has to build $n$ further temporal images. The time required to process a string of length $n$ as a function of the length of the input sequence is thus $(\tau - \theta)(n + 1)$. The constant $\tau$ also depends on $t_r$ which is chosen to let the network satisfy a certain probability that it reaches the next state in relaxation. This probability is given by $(1 - \frac{1}{N})^B$ where $B = t_r/N$. The time complexity of the neural-network acceptor is $O(n)$.

The upper limit on the number $p$ of stored temporal relations between single activity patterns is $| Q |^2 \times | \Sigma |^2$. The number of neurons in a network is then $c \times | Q |^2 \times | \Sigma |^2$, where $c$ depends on the storage capacity and the chosen (low) probability that selection errors occur. The randomly chosen activity patterns overlap, so if a large number of patterns is active they may constitute, by overlap, other unselected activity patterns that will create their own causal consequences. This is called a selection error. The probability that this can happen can be estimated by $P_{error}(n) \equiv 1 - P(S_n = 0)$, where the latter is:

$$P\left(\frac{-1-2np}{2\sqrt{npq}} \leq S_n \leq \frac{1-2np}{2\sqrt{npq}}\right) \approx \frac{1}{\sqrt{2\pi}} \int_{\frac{-1-2np}{2\sqrt{npq}}}^{\frac{1-2np}{2\sqrt{npq}}} e^{-x^2/2} dx.$$

In this expression, $p \equiv (c-m)/v$ where $v \equiv \binom{N}{aN}$, $c$ is the number of activity patterns stored in the network, and $m$ is the number of patterns that were supposed to be present in the mixture state. The probability $q = 1 - p$, and $n \equiv \binom{x}{aN}$ is the number of patterns that can be constructed from the active neurons in the mix. $S_n$ is the number of wrongly selected activity patterns for a given $n$. $P_{error}(n)$ decreases with increasing $N$ if the other parameters remain fixed.

The space complexity of the network, expressed as the number of neurons, and as a function of the number of NDA states is $O(| Q |^2)$. This is large because $Q = Q_{ST} \leq | \Gamma |^k$ for some PDA $M$. However things could have been worse. Not using mixed temporal images to represent FSA states would necessitate the use of a number of temporal images of order $2^{|Q|^2}$. So compared to a more conventional use of Hopfield models, this approach yields a reduction of the space complexity of the network.

## 8 Conclusions

We proposed an acceptor for all context-free languages with limited center-embedding, and a suitable variant of the Hopfield model. The formal model was implemented on the Hopfield model, and a correctness theorem for the latter was given. Simulation results provided initial corroboration of our theory. The obtained neural-network acceptor is fast but large.

Continuation of this research in the near future consists of the design of an adaptive variant of this model, one that learns a grammar from examples in an unsupervised fashion.

## Acknowledgements

I am very grateful for the indispensable support of Mannes Poel and Anton Nijholt in making this paper. Special thanks go out to Bert Helthuis who wrote the very beautiful simulation program with which the theory was corroborated.

## References

[1] D.J. Amit. *Modeling Brain Function*. Cambridge University Press, Cambridge, 1989.

[2] J. Buhmann, R. Divko, and K. Schulten. Associative memory with high information content. *Physical Review A*, 39(5):2689–2692, 1989.

[3] E. Charniak and E. Santos. A connectionist context-free parser which is not context-free, but then it is not really connectionist either. In *Proceedings 9th Ann. Conf. of the Cognitive Science Society*, pages 70–77, 1987.

[4] M.F.J. Drossaers. Neural-network acceptors. Technical Report Memoranda Informatica 92-36, University of Twente, Enschede, 1992.

[5] M.A. Fanty. Context-free parsing in connectionist networks. Technical Report TR 174, University of Rochester, Rochester, NY, 1985.

[6] H. Nakagawa and T. Mori. A parser based on connectionist model. In *Proceedings of COLING '88*, pages 454–458, 1988.

[7] B. Selman and G. Hirst. Parsing as an energy minimization problem. In E. Davis, editor, *Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence*, pages 141–154, Los Altos, Calif., 1987. Morgan Kaufman Publishers.

[8] H. Sompolinsky and I. Kanter. Temporal association in asymmetric neural networks. *Physical Review Letters*, 57(22):2861–2864, 1986.

[9] D.S. Touretzky. Connectionism and compositional semantics. Technical Report CMU-CS-89-147, Carnegie Mellon University, Pittsburgh, 1989.

[10] D.L. Waltz and J.B. Pollack. Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive Science*, pages 51–74, 1985.