Permutation parity machines for neural synchronization

2009 J. Phys. A: Math. Theor. 42 195002

(http://iopscience.iop.org/1751-8121/42/19/195002)

IOP PUBLISHING JOURNAL OF PHYSICS A: MATHEMATICAL AND THEORETICAL

J. Phys. A: Math. Theor. **42** (2009) 195002 (20pp) doi:10.1088/1751-8113/42/19/195002

Permutation parity machines for neural synchronization

O M Reyes[1,2], **I Kopitzke**[1] **and K-H Zimmermann**[1]

[1] Institute of Computer Technology, Hamburg University of Technology, D-21071 Hamburg, Germany
[2] Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones, Universidad Industrial de Santander, Bucaramanga, Colombia

E-mail: omreyes@uis.edu.co and k.zimmermann@tuhh.de

Received 10 November 2008, in final form 31 March 2009
Published 21 April 2009
Online at stacks.iop.org/JPhysA/42/195002

Abstract
Synchronization of neural networks has been studied in recent years as an alternative to cryptographic applications such as the realization of symmetric key exchange protocols. This paper presents a first view of the so-called permutation parity machine, an artificial neural network proposed as a binary variant of the tree parity machine. The dynamics of the synchronization process by mutual learning between permutation parity machines is analytically studied and the results are compared with those of tree parity machines. It will turn out that for neural synchronization, permutation parity machines form a viable alternative to tree parity machines.

PACS numbers: 05.45.Xt, 84.35.+i, 87.18.Sn, 89.70.−a

1. Introduction

Synchronization is a phenomenon that can be observed in several physical and biological systems [1]. Recently, it was shown that artificial neural networks can synchronize too [2]. Artificial neural networks were first developed to study and simulate the behavior of biological neurons. But it was soon recognized that they can be used to solve artificial intelligence problems in the fields of speech recognition, image analysis and adaptive control [3].

Two artificial neural networks can synchronize by mutual learning. For this, they start with randomly chosen weights. In each step, they receive common inputs, calculate their outputs and communicate them to each other. Depending on the learning rule, the weights are updated. In the case of discrete weights, this process eventually leads to full synchronization in a finite number of steps.

Tree parity machines are artificial neural networks that can synchronize by mutual learning. However, they exhibit a new phenomenon. Synchronization by mutual learning

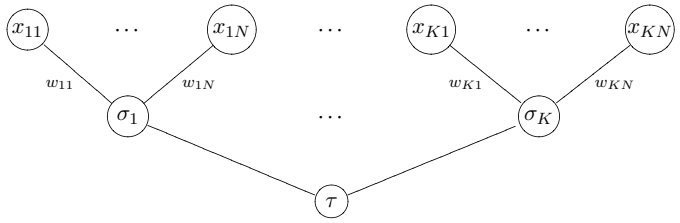

Figure 1. General structure of a tree parity machine.

is much faster than learning by adapting to examples generated by other networks [2, 4, 5]. In this way, active and passive participants can be distinguished in the learning process. This observation is key to using neural synchronization for cryptographic key exchange protocols [6, 7].

This paper introduces a binary version of the tree parity machine whose learning rule is based on weight permutation. In section 2, the tree parity machine and its mutual learning process are reconsidered. Section 3 introduces the permutation parity machine, its learning rule and order parameters that describe the correlation between permutation parity machines during the mutual learning process. Section 4 studies the dynamics of the mutual learning process in dependence of the order parameters. Finally, tree and permutation parity machines are compared with respect to the mutual learning process.

2. Tree parity machines

Tree parity machines are multi-layer feed-forward networks that contain one hidden layer (figure 1) [2].

Let K, L and N be positive integers. A tree parity machine consists of K hidden units that are perceptrons with independent receptive fields. Each unit has N input neurons and one output neuron. All input values are binary,

$$x_{ij} \in \{-1, +1\}, \qquad 1 \leqslant i \leqslant K, \quad 1 \leqslant j \leqslant N, \tag{1}$$

and the weights weightening the connections between input and hidden units are integral numbers between $-L$ and $+L$,

$$w_{ij} \in \{-L, \ldots, 0, \ldots, +L\}, \qquad 1 \leqslant i \leqslant K, \quad 1 \leqslant j \leqslant N, \tag{2}$$

where L denotes the synaptic depth of the network.

The output of a hidden unit is determined by the weighted sum over the current input values:

$$h_i = \frac{1}{\sqrt{N}} \boldsymbol{w}_i \cdot \boldsymbol{x}_i = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} w_{ij} x_{ij}, \qquad 1 \leqslant i \leqslant K. \tag{3}$$

The output of the ith hidden unit is given by the sign of the random field h_i:

$$\sigma_i = \operatorname{sgn}(h_i), \qquad 1 \leqslant i \leqslant K. \tag{4}$$

The special case, $h_i = 0$, is mapped to $\sigma_i = -1$ in order to ensure binary output. A hidden unit is considered to be active if the weighted sum over its inputs is positive ($\sigma_i = +1$); otherwise, it is inactive ($\sigma_i = -1$).

The output of a tree parity machine is defined by the parity of the hidden units:

$$\tau = \prod_{i=1}^{K} \sigma_i. \tag{5}$$

The output indicates whether the number of inactive hidden units is even ($\tau = +1$) or odd ($\tau = -1$). There are 2^{K-1} internal representations $(\sigma_1, \ldots, \sigma_K)$ of the hidden units that lead to the same output value τ.

Two interacting neural feed-forward networks can synchronize by mutual learning. For this, let A and B denote two tree parity machines with identical parameters K, L and N. Both networks start with randomly chosen weight vectors $\boldsymbol{w}_i^A$ and $\boldsymbol{w}_i^B$, $1 \leqslant i \leqslant K$. In each time step, both machines receive common input vectors $\boldsymbol{x}_i$, $1 \leqslant i \leqslant K$, and compute the corresponding outputs τ^A and τ^B. Then both networks exchange their output bits. If the output bits disagree ($\tau^A \neq \tau^B$), the weights remain unchanged. Otherwise, the weights are updated according to a suitable learning rule.

For instance, the *Hebbian learning rule* says that in the case of agreeing output bits ($\tau^A = \tau^B$), the weights of hidden units that have the same output as the communicated output are updated. The new weights are given by

$$w_{ij} := \begin{cases} g(w_{ij} + x_{ij} \cdot \sigma_i) & \text{if} \quad \sigma_i = \tau^A, \\ w_{ij} & \text{otherwise,} \end{cases} \qquad 1 \leqslant i \leqslant K, \quad 1 \leqslant j \leqslant N, \tag{6}$$

where the function $g(w)$ guarantees that the new weights stay in the allowed range by bouncing back values to the nearest boundary value:

$$g(w) = \begin{cases} L & \text{if} \quad w > L, \\ w & \text{if} \quad -L \leqslant w \leqslant L, \\ -L & \text{otherwise.} \end{cases} \tag{7}$$

This synchronization step can be repeated until the corresponding weights in both networks eventually are in synch; that is, the networks have equal weights: $\boldsymbol{w}_i^A = \boldsymbol{w}_i^B$, $1 \leqslant i \leqslant K$. Once the two machines are synchronized they will remain synchronized since the movements of the weights only depend on the inputs and weights which are already identical.

Instead of communicating single bits during the mutual learning phase, a block mode called bit-packaging was proposed [7]. In this mode, a sequence of $B \geqslant 1$ synchronization steps is performed and the corresponding output bits are stored in registers. After B synchronization steps, the contents of the registers are exchanged between the networks and the networks perform B consecutive steps to adapt their weights.

3. Permutation parity machines

Permutation parity machines are multi-layer feed-forward networks proposed as a variant of tree parity machines with binary weights [8].

3.1. Structure

Let G, K and N be positive integers. A *permutation parity machine* can be considered as a neural network with K hidden units that are perceptrons with independent receptive fields (figure 2).

Each unit has N input neurons and one output neuron. All input values are binary,

$$x_{i,j} \in \{0, 1\}, \qquad 1 \leqslant i \leqslant K, \quad 1 \leqslant j \leqslant N. \tag{8}$$

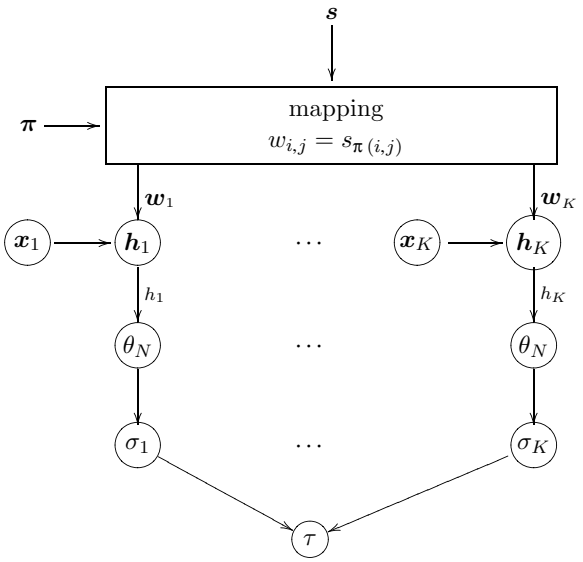

Figure 2. General structure of a permutation parity machine.

Unlike the tree parity machine, the synaptic weights are drawn from a pool of binary data given by a so-called *state vector* $\boldsymbol{s} \in \{0,1\}^G$. More specifically, take an $K \times N$ matrix $\boldsymbol{\pi} = (\pi_{ij})$ whose entries are given as the images of the one-to-one mapping $\boldsymbol{\pi} : \{1,\ldots,K\} \times \{1,\ldots,N\} \to \{1,\ldots,G\} : (i,j) \mapsto \pi_{i,j}$. Then assign the weights by taking the entries of the state vector $\boldsymbol{s}$ according to the positions given by the matrix entries,

$$w_{i,j} = s_{\pi_{i,j}}, \qquad 1 \leqslant i \leqslant K, \quad 1 \leqslant j \leqslant N. \tag{9}$$

Since the mapping should be one to one, the length of the state vector must satisfy $G \geqslant K \cdot N$. This mapping can be implemented by using a linear feedback shift register with period G for addressing the components of the state vector.

The output of the ith hidden unit requires to determine the component-wise exclusive disjunction (exclusive or) between weights and inputs:

$$\boldsymbol{h}_i = \boldsymbol{x}_i \oplus \boldsymbol{w}_i = (x_{i,j} \oplus w_{i,j})_j, \qquad 1 \leqslant i \leqslant K. \tag{10}$$

The vectorized random field $\boldsymbol{h}_i$ provides the number of positions at which inputs and weights differ:

$$h_i = |\{j \mid h_{ij} = x_{i,j} \oplus w_{i,j} = 1, 1 \leqslant j \leqslant N\}|, \qquad 1 \leqslant i \leqslant K, \tag{11}$$

where $|\cdot|$ denotes the size of a set.

The output of the ith hidden unit yields a threshold value for the random field h_i. It equals 1 if the random field is larger than $N/2$, and equals 0 otherwise; that is,

$$\sigma_i = \theta_N(h_i), \qquad 1 \leqslant i \leqslant K, \tag{12}$$

where for each non-negative integer h,

$$\theta_N(h) = \begin{cases} 1, & h > N/2, \\ 0, & h \leqslant N/2. \end{cases} \tag{13}$$

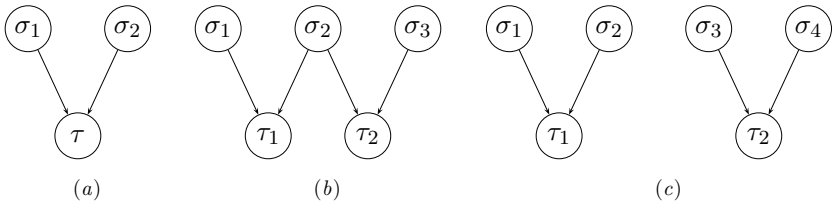

Figure 3. Some possible output layer configurations for permutation parity machines: (*a*) single-bit output with two hidden units, (*b*) two-bit output with three hidden units and (*c*) two-bit output with four hidden units.

The output of a permutation parity machine is given by the parity of the hidden units:

$$\tau = \bigoplus_{i=1}^{K} \sigma_i . \tag{14}$$

The output τ indicates whether the number of active hidden units ($\sigma_i = 1$) is even ($\tau = 0$) or odd ($\tau = 1$).

Several configurations of the output layer depending on the number of hidden units were proposed by Kopitzke [8] (figure 3). However, configurations with two or more output bits increase the machine complexity and require a re-definition of the learning rule. Therefore, we only analyze permutation parity machines with two hidden units ($K = 2$) and a single output bit; the case $K = 2$ already allows a direct comparison with the tree parity machine.

3.2. Learning rule

Two interacting permutation parity machines can synchronize by mutual learning. For this, let A and B denote two permutation parity machines with identical parameters G, K and N. The learning process involves two kinds of rounds, inner and outer rounds. An *outer round* consists of several inner rounds. A series of inner rounds is employed to fill an initially empty buffer of length G position by position in each network. When the buffers are completely filled, an outer round replaces each state vector by the respectively filled buffer.

The machines A and B start with randomly chosen state vectors $\boldsymbol{s}^A$ and $\boldsymbol{s}^B$ and empty buffers $\boldsymbol{b}^A$ and $\boldsymbol{b}^B$ of length G, respectively. Each *inner round* consists of the following steps:

- Choose uniformly at random a $K \times N$ matrix $\boldsymbol{\pi}$ and a binary input vector $\boldsymbol{x} = (x_{ij})$ of length $K \cdot N$; these data are known to both networks.
- Generate the weights w_{ij} according to (9).
- Compute the output bits τ^A and τ^B according to (14). If the output bits are equal, then in each machine the output of the first perceptron, σ_1, is stored in the next empty position of the corresponding buffer. Otherwise, the buffers remain unchanged.

We speak of a *synchronization step* if during the learning phase the output bits τ^A and τ^B are equal; in this case, the buffers are filled by one bit. However, the output of the first perceptron, σ_1^A, in machine A can be different from the output of the first perceptron, σ_1^B, in machine B. The inner rounds are repeated until the buffers are completely filled. Then the state vectors are updated with their buffers (i.e., $\boldsymbol{s}^A := \boldsymbol{b}^A$ and $\boldsymbol{s}^B := \boldsymbol{b}^B$), and the outer round ends. A new outer round starts with emptying the buffers and providing a new series of inner rounds.

This learning process can be repeated until the networks are eventually synchronized. That means, the state vectors are aligned in such a way that they remain aligned even if they are updated by further applications of the learning rule. We will later see that this learning process can be described by a first-order Markov chain so that synchronization amounts to stationarity of the chain.

3.3. Order parameters

Order parameters are used to describe the correlation between two permutation parity machines during the mutual learning process. For this, the probability distributions of the state vectors and the weights in corresponding hidden units are studied. First, take the probability of finding a weight with $w_{ij}^A = a$ in the ith hidden unit of machine A and a corresponding weight with $w_{ij}^B = b$ in the ith hidden unit of machine B, $1 \leqslant i \leqslant K$:

$$p_{a,b}^i = P\left(w_{ij}^A = a, w_{ij}^B = b\right), \qquad 1 \leqslant j \leqslant N, a, b \in \{0, 1\}. \tag{15}$$

Second, consider the probability of finding an entry with $s_l^A = a$ in the state vector of machine A and an associated entry with $s_l^B = b$ in the state vector of machine B:

$$p_{a,b} = P\big(s_l^A = a, s_l^B = b\big), \qquad 1 \leqslant l \leqslant G, a, b \in \{0, 1\}. \tag{16}$$

In simulations these probabilities can be estimated by relative frequencies:

$$p_{a,b}^i = \frac{1}{N}\big|\big\{j \,\big|\, w_{ij}^A = a, w_{ij}^B = b, 1 \leqslant j \leqslant N\big\}\big|, \tag{17}$$

$$p_{a,b} = \frac{1}{G}\big|\big\{l \,\big|\, s_l^A = a, s_l^B = b, 1 \leqslant l \leqslant G\big\}\big|. \tag{18}$$

The level of synchronization can be estimated by using the expected values of these probabilities. Such kinds of expectations are generally studied in online learning processes [9]. The *normalized overlap* between the ith hidden units is given by

$$\rho_i^{AB} = p_{0,0}^i + p_{1,1}^i, \tag{19}$$

while the *normalized overlap* between the state vectors is defined as

$$\rho^{AB} = p_{0,0} + p_{1,1}. \tag{20}$$

In practice, these overlaps can be calculated by using Hamming distances:

$$\rho_i^{AB} = 1 - \frac{1}{N}\mathrm{d_H}\big(\boldsymbol{w}_i^A, \boldsymbol{w}_i^B\big), \qquad 1 \leqslant i \leqslant K, \tag{21}$$

and

$$\rho^{AB} = 1 - \frac{1}{G}\mathrm{d_H}(\boldsymbol{s}^A, \boldsymbol{s}^B). \tag{22}$$

For further reference, we also consider the corresponding *non-normalized overlaps*,

$$r_i = N \cdot \rho_i = N - \mathrm{d_H}\big(\boldsymbol{w}_i^A, \boldsymbol{w}_i^B\big) = N - \mathrm{d_H}\big(\boldsymbol{h}_i^A, \boldsymbol{h}_i^B\big), \qquad 1 \leqslant i \leqslant K, \tag{23}$$

where the latter equality follows from the definitions, and

$$R = G \cdot \rho = G - \mathrm{d_H}(\boldsymbol{s}^A, \boldsymbol{s}^B). \tag{24}$$

Clearly, uncorrelated hidden units have overlap $\rho_i = 0.5$, while the overlaps $\rho_i = 0$ and $\rho_i = 1$ indicate anti-parallel $\big(\boldsymbol{w}_i^A = \overline{\boldsymbol{w}_i^B}\big)$ and parallel weights $\big(\boldsymbol{w}_i^A = \boldsymbol{w}_i^B\big)$, respectively. The situation is similar for state vectors.

The overlap R between the state vectors and the overlap r between hidden units are related by the hypergeometric distribution. For this, note that the hidden units' weights are drawn uniformly at random from the state vectors. The probability that exactly r positions of the weight vector of a hidden unit are drawn from R overlapping positions of the corresponding state vector is given as

$$f_{r;G,R,N} = \frac{\binom{R}{r}\binom{G-R}{N-r}}{\binom{G}{N}}, \qquad 0 \leqslant r \leqslant N. \tag{25}$$

In the case of $G \gg N$ such that the ratio $R/G = \rho$ and N are held constant, the hypergeometric distribution approaches the binomial distribution [10]. That is,

$$f_{r;G,R,N} \approx \binom{N}{r} \rho^r (1-\rho)^{N-r}. \tag{26}$$

We next study the outputs of hidden units. To this end, consider the ith hidden unit in two permutation parity machines A and B with identical parameters, and let $q^a_{r,N}$ denote the probability that these hidden units with non-normalized overlap r both provide output $a \in \{0, 1\}$, $1 \leqslant i \leqslant K$. First, the probability that the overlap contains m zeros is given by

$$P_m = \frac{1}{2^r}\binom{r}{m}, \qquad 0 \leqslant m \leqslant r. \tag{27}$$

Moreover, the probability that the non-overlapping section contains n zeros amounts to

$$Q_n = \frac{1}{2^{N-r}}\binom{N-r}{n}, \qquad 0 \leqslant n \leqslant N-r. \tag{28}$$

Since the overlap r satisfies (23), we may either consider the weights $\boldsymbol{w}^A$ and $\boldsymbol{w}^B$ or the corresponding vectorized random fields $\boldsymbol{h}^A$ and $\boldsymbol{h}^B$ in the respective hidden units of the machines A and B. Thus if the vectorized random field $\boldsymbol{h}^A$ has m zeros in the overlap and n zeros in the non-overlapping section, then the vectorized random field $\boldsymbol{h}^B$ must have m zeros in the overlap and $N-r-n$ zeros in the non-overlapping part:

$$\begin{array}{lcccc} \boldsymbol{h}^A: & \overbrace{0\ldots0}^{m} & \overbrace{1\ldots1}^{r-m} & \overbrace{0\ldots0}^{n} & 1\ldots1 \\ \boldsymbol{h}^B: & 0\ldots0 & 1\ldots1 & 1\ldots1 & \underbrace{0\ldots0}_{N-r-n}. \end{array}$$

Let h^A and h^B denote the random fields corresponding to the vectorized random fields $\boldsymbol{h}^A$ and $\boldsymbol{h}^B$ in corresponding hidden units, respectively. We have $\theta_N(h^A) = 0$ and $\theta_N(h^B) = 0$ if and only if the number of zeros in $\boldsymbol{h}^A$ and $\boldsymbol{h}^B$ is at least $N/2$. Equivalently, $h^A = m + n \geqslant N/2$ and $h^B = m + N - r - n \geqslant N/2$. It follows that $m \geqslant r/2$ and $N/2 - m \leqslant n \leqslant N/2 - r + m$. But the probabilities P_m and Q_n are independent and so the probability $q^0_{r,N}$ can be calculated as

$$q^0_{r,N} = \frac{1}{2^N} \sum_{m=\lceil \frac{r}{2} \rceil}^{r} \sum_{n=\lceil \frac{N}{2}-m \rceil}^{\lfloor \frac{N}{2}-r+m \rfloor} \binom{r}{m}\binom{N-r}{n}. \tag{29}$$

In view of the probability $q^1_{r,N}$, observe that the arguments of the threshold function θ_N are uniformly distributed since they are given by the randomly chosen matrix $\boldsymbol{\pi}$ and the input vector $\boldsymbol{x}$. But if the parameter N is odd, the threshold function θ_N maps $(N+1)/2$ values to 0 and $(N+1)/2$ values to 1. So the outputs 0 and 1 are obtained with the same probability, and

hence we have $q^0_{r,N} = q^1_{r,N}$. However, if the parameter N is even, the threshold function θ_N maps $N/2$ values to 0 but only $N/2-1$ values to 1. In this case, we obtain

$$q^1_{r,N} = \frac{1}{2^N} \sum_{m=\lceil \frac{r+1}{2} \rceil}^{r} \sum_{n=\lceil \frac{N+1}{2}-m \rceil}^{\lfloor \frac{N-1}{2}-r+m \rfloor} \binom{r}{m}\binom{N-r}{n}. \tag{30}$$

Let $q_{r,N}$ denote the probability that two corresponding hidden units in the networks A and B have non-normalized overlap r and yield the same output. Clearly, $q_{r,N} = q^0_{r,N} + q^1_{r,N}$ and thus

$$q_{r,N} = \begin{cases} 2 \cdot q^0_{r,N}, & \text{if} \quad N \text{ odd}, \\ q^0_{r,N} + q^1_{r,N}, & \text{otherwise}. \end{cases} \tag{31}$$

From (29)–(31), it follows that

$$q_{r,N} = \begin{cases} 1 - q_{N-r,N}, & \text{if} \quad N \text{ odd}, \\ 1 - q_{N-r-1,N}, & \text{otherwise}. \end{cases} \tag{32}$$

Thus if the parameter N is odd, then the probability $q_{r,N}$ as a function of the normalized overlap $\rho = r/N$ has odd symmetry at $\rho = 0.5$ (figure 4). Therefore, (29) and (31) imply

$$q_{0,N} = 0 \qquad \text{and} \qquad q_{N,N} = 1, \quad N \text{ odd}. \tag{33}$$

On the other hand, if the parameter N is even, then (32) implies that the probability $q_{r,N}$ has no such symmetry. In particular, from (29) we find that $q^0_{0,N} = \frac{1}{2^N}\binom{N}{N/2}$ and $q^0_{N,N} = \frac{1}{2} + \frac{1}{2^{N+1}}\binom{N}{N/2}$, and from (30) that $q^1_{0,N} = 0$ and $q^1_{N,N} = \frac{1}{2} - \frac{1}{2^{N+1}}\binom{N}{N/2}$. Thus by (31),

$$q_{0,N} = \frac{1}{2^N}\binom{N}{N/2} \qquad \text{and} \qquad q_{N,N} = 1, \quad N \text{ even}. \tag{34}$$

Finally, it is well known that in the thermodynamic limit, when N approaches infinity, the error probability $1 - q_{r,N}$ tends to the *generalization error* between two perceptrons [9]:

$$\epsilon = \frac{1}{\pi} \arccos(2\rho - 1), \qquad -1 \leqslant 2\rho - 1 \leqslant 1, \tag{35}$$

where ρ is the normalized overlap between the perceptrons. The error probabilities given by $\epsilon_{r,N} = 1 - q_{r,N}$ are good finite approximations of the generalization error as illustrated in figure 4. In this figure, the solid line indicating the generalization error ($N \to \infty$) was computed from (35), while the symbols were computed from (29) to (31) by using (23) to obtain the normalized overlap ρ from the non-normalized overlap r.

4. Dynamics of the mutual learning process

First, the effects of the learning rule on the inner rounds will be considered as a function of the overlap. Second, the learning process is described as a Markov chain in order to find necessary conditions for achieving synchronization.

As mentioned in section 3.1, the mutual learning process will be studied in the case of two hidden units per machine ($K = 2$), although several of the expressions developed will also hold in the general case.

4.1. Effects of learning rule

During the mutual learning process of two identical permutation parity machines A and B, a synchronization step occurs when the outputs of both machines, τ^A and τ^B, are equal. Then the buffers of both machines are updated. Two situations can occur during a synchronization

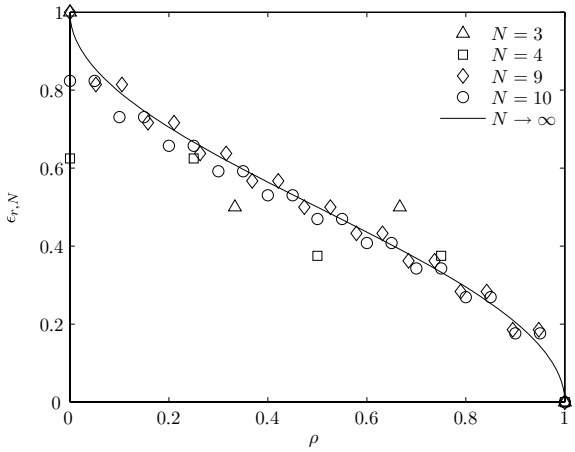

Figure 4. Error probabilities between two perceptrons in dependence of normalized overlap. The solid line indicates the generalization error.

step:

- The outputs of the first hidden units are equal $\left(\sigma_1^A = \sigma_1^B\right)$ and so the overlap between the buffers increases.
- Otherwise, $\sigma_1^A \neq \sigma_1^B$ and thus the overlap between the buffers decreases.

The first step is called *increasing*, while the second step is called *decreasing*. A sequence of only increasing steps eventually leads to a *parallel alignment* such that the overlap ρ becomes 1 and thus $\boldsymbol{s}^A = \boldsymbol{s}^B$. Similarly, a series of only decreasing steps eventually provides an *anti-parallel alignment* in which the overlap becomes 0 and thus $\boldsymbol{s}^A = \overline{\boldsymbol{s}^B}$, where the binary complement is taken component-wise.

The probability that the ith hidden units produce the same output when R is the overlap of the state vectors is given by

$$P_R\left(\sigma_i^A = \sigma_i^B\right) = \sum_{r=0}^{N} q_{r,N} \cdot f_{r;G,R,N}, \qquad 1 \leqslant i \leqslant K, \tag{36}$$

while the probability that they produce different outputs can be obtained as

$$P_R\left(\sigma_i^A \neq \sigma_i^B\right) = \sum_{r=0}^{N} (1 - q_{r,N}) \cdot f_{r;G,R,N}, \qquad 1 \leqslant i \leqslant K, \tag{37}$$

where $q_{r,N}$ and $f_{r;G,R,N}$ are given by (31) and (25), respectively. We have $P_R\left(\sigma_i^A = \sigma_i^B\right) = 1 - P_R\left(\sigma_i^A \neq \sigma_i^B\right)$. Furthermore, if the parameter N is odd, then by using (32), (37) and the symmetry of the hypergeometric distribution, $f_{r;G,R,N} = f_{N-r;G,G-R,N}$, we obtain

$$P_R\left(\sigma_i^A \neq \sigma_i^B\right) = P_{G-R}\left(\sigma_i^A = \sigma_i^B\right). \tag{38}$$

In the case of $G \gg N$, it follows from (26) that the probabilities $P_R\left(\sigma_i^A = \sigma_i^B\right)$ and $P_R\left(\sigma_i^A \neq \sigma_i^B\right)$ are functions of the normalized overlap $\rho = R/G$ and the number of inputs per hidden unit N. Moreover, when G tends to infinity, the above probabilities are continuous and the hidden units become independent, and when N also goes to infinity, the probability $P_R\left(\sigma_i^A \neq \sigma_i^B\right)$ corresponds to the generalization error ϵ given by (35) (figure 5). The symbols in this figure were computed from (37) in dependence of the normalized overlap $\rho = R/G$, where $0 \leqslant R \leqslant G$.

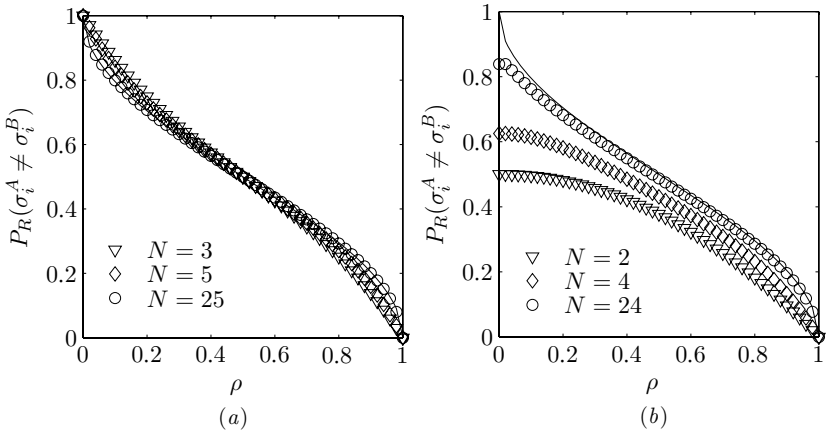

Figure 5. Error probability of corresponding hidden units in two permutation parity machines in dependence of the normalized overlap for the parameter $G = 64$. The continuous line indicates the generalization error: (*a*) N odd, (*b*) N even.

During the mutual learning process, the buffers are updated in the case of $\tau^A = \tau^B$. Thus the probability of a synchronization step for $K = 2$ is given by

$$P_R(\tau^A = \tau^B) = P_R\left[\left(\sigma_1^A = \sigma_1^B \wedge \sigma_2^A = \sigma_2^B\right) \vee \left(\sigma_1^A \neq \sigma_1^B \wedge \sigma_2^A \neq \sigma_2^B\right)\right]$$
$$= P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) + P_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right). \tag{39}$$

Moreover, by (36),

$$P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) = \sum_{r_1=0}^{N}\sum_{r_2=0}^{N} q_{r_1,N} \cdot f_{r_1;G,R,N} \cdot q_{r_2,N} \cdot f_{r_2;G-N,R-r_1,N} \tag{40}$$

and

$$P_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right)$$
$$= \sum_{r_1=0}^{N}\sum_{r_2=0}^{N} (1 - q_{r_1,N}) \cdot f_{r_1;G,R,N} \cdot (1 - q_{r_2,N}) \cdot f_{r_2;G-N,R-r_1,N}. \tag{41}$$

We next study the behavior of the probability of a synchronization step as a function of the normalized overlap. First, we infer from (26) that for a fixed overlap ρ, the influence of the length G of state vectors on the probability of synchronization steps with mutual interaction is rather weak especially when $G \gg N$. This behavior is illustrated in figure 6. In contrast to this, the parameter N significantly affects this probability. In particular, if the parameter N is odd, then by using (32) and (41), and taking into account the symmetry of the hypergeometric distribution, $f_{r;G,R,N} = f_{N-r;G,G-R,N}$, we obtain

$$P_{G-R}\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right) = P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right). \tag{42}$$

Therefore, we derive

$$P_R(\tau^A = \tau^B) = P_{G-R}(\tau^A = \tau^B). \tag{43}$$

In this case, the probability of synchronization steps has even symmetry around the normalized overlap $\rho = 0.5$. From the definitions we can find the probability $P_G(\tau^A = \tau^B)$. For this, note that from (25)

$$f_{r;G,G,N} = \begin{cases} 1 & \text{if} \quad r = N, \\ 0 & \text{otherwise.} \end{cases}$$

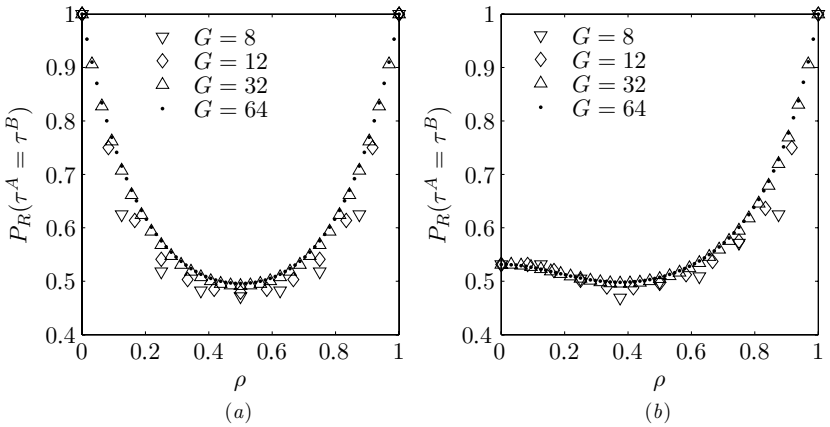

Figure 6. Probability of synchronization steps with mutual interaction in dependence of normalized overlap. (*a*) $N = 3$, (*b*) $N = 4$.

Thus (40) and (41) respectively become

$$P_G\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) = q_{N,N}^2$$

and

$$P_G\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right) = (1 - q_{N,N})^2.$$

By (33) and (34), we have $q_{N,N} = 1$ for all N. Thus it follows from the last two equations that

$$P_G\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) = 1 \qquad \text{and} \qquad P_G\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right) = 0. \tag{44}$$

Consequently, we find that $P_G(\tau^A = \tau^B) = 1$. By the symmetry given in (43) we deduce that $P_0(\tau^A = \tau^B) = 1$ as well. This means that if the parameter N is odd and a parallel ($\rho = 1$) or an anti-parallel ($\rho = 0$) alignment is reached, then the networks always produce equal outputs: $\tau^A = \tau^B$. Moreover, this probability becomes minimal when the correlation between the state vectors becomes minimal ($\rho = 0.5$) such that $P_R(\tau^A = \tau^B)|_{\rho=0.5} = 0.5$. Figure 7(*a*) illustrates the probability $P_R(\tau^A = \tau^B)$ as a function of the overlap ρ for some odd values of the parameter N.

Similarly, when the parameter N is even and a parallel alignment ($\rho = 1$) is given, we find that $P_G(\tau^A = \tau^B) = 1$. However, if the overlap is 0, we obtain from (34) that $0 < q_{0,N} < 1$. Moreover, it follows from (40) and (41) that the probability of a synchronization step is $P_0(\tau^A = \tau^B) = (q_{0,N})^2 + (1 - q_{0,N})^2 = 1 - 2q_{0,N}(1 - q_{0,N}) < 1$. Nevertheless, this probability is increasing when the parameter N (even) is increasing (figure 7(*b*)).

In the mutual learning process, the probability of increasing steps corresponds to the conditional probability that a synchronization step occurs in which the outputs of the perceptrons are equal:

$$P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = \frac{P_R\left(\sigma_1^A = \sigma_1^B, \tau^A = \tau^B\right)}{P_R\left(\tau^A = \tau^B\right)}, \tag{45}$$

and the probability of decreasing steps is given as

$$P_R\left(\sigma_1^A \neq \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = 1 - P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right). \tag{46}$$

In particular, in the case of $K = 2$, we obtain from (39) that

$$P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = \frac{P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right)}{P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) + P_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right)}. \tag{47}$$

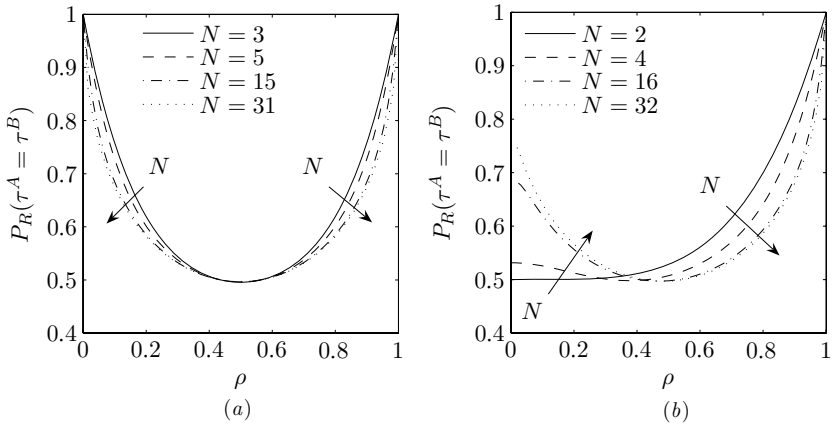

Figure 7. Probability of synchronization steps with mutual interaction in dependence of normalized overlap for the parameter $G = 64$. (*a*) N odd, (*b*) N even.

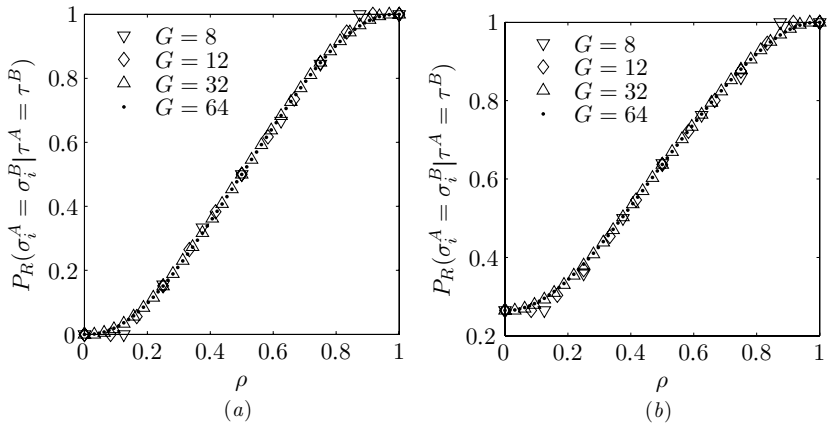

Figure 8. Probability of increasing steps for synchronization with mutual interaction under the condition $\tau^A = \tau^B$ and in dependence of normalized overlap. (*a*) $N = 3$, (*b*) $N = 4$.

In the thermodynamic limit, when N approaches infinity, this probability can be written in terms of the generalization error as

$$P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = \frac{(1-\epsilon)^2}{(1-\epsilon)^2 + \epsilon^2}. \tag{48}$$

It follows that neither the probability of increasing steps nor that of decreasing steps is significantly affected by the length G of the state vectors similar to the probability of synchronization steps (figure 8). Moreover the probability of increasing (decreasing) steps is an increasing (decreasing) function of the normalized overlap ρ (figures 8 and 9).

If the parameter N is odd, then in view of (42) the probability of increasing (decreasing) steps, which are symmetric around the overlap $\rho = 0.5$, amounts to

$$\begin{aligned} P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) &= 1 - P_{G-R}\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) \\ &= P_{G-R}\left(\sigma_1^A \neq \sigma_1^B \,\middle|\, \tau^A = \tau^B\right), \qquad N \text{ odd}. \end{aligned} \tag{49}$$

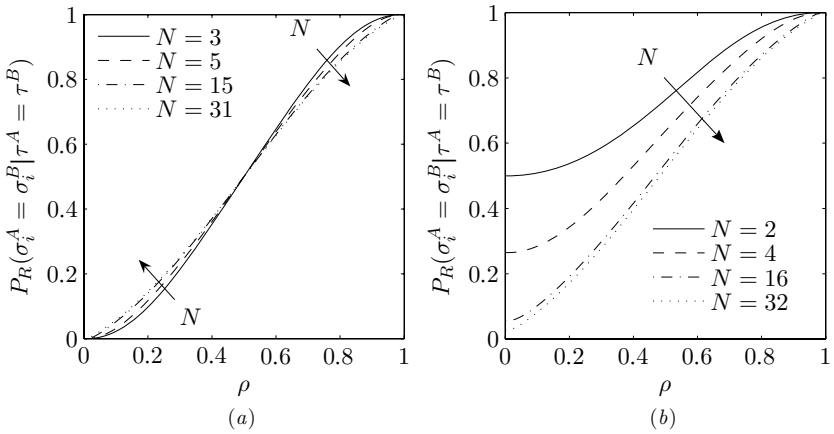

Figure 9. Probability of increasing steps for synchronization with mutual interaction under the condition $\tau^A = \tau^B$ and in dependence of normalized overlap for the parameter $G = 64$. (*a*) N odd, (*b*) N even.

Using this symmetry and (44) we find that

$$P_G\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = P_0\left(\sigma_1^A \neq \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = 1, \tag{50}$$

as illustrated in figure 9(*a*). This means that once the state vectors are parallel (anti-parallel), not only the output bits in all subsequent inner rounds are the same ($\tau^A = \tau^B$), but also a sequence of only increasing (decreasing) steps is always performed preserving the alignment of the state vectors during the outer rounds. Therefore, an odd number of inputs per hidden unit eventually produces a *bivergent behavior*, in which there are two possible alignments for the state vectors that attain synchronization: parallel ($\boldsymbol{s}^A = \boldsymbol{s}^B$) or anti-parallel ($\boldsymbol{s}^A = \overline{\boldsymbol{s}^B}$), and both states are equally likely.

On the other hand, if the parameter N is even, we obtain from (44) and (47) that the probability of increasing steps becomes 1 when the overlap ρ equals 1: $P_G\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = 1$. However, for the overlap $\rho = 0$, the probability of decreasing steps becomes

$$P_0\left(\sigma_1^A \neq \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) = \frac{(1 - q_{0,N})^2}{(q_{0,N})^2 + (1 - q_{0,N})^2}. \tag{51}$$

Since $0 < q_{0,N} < 1$, it follows that this probability is smaller than 1 (figure 9(*b*)). Thus in the case of N even, only increasing steps eventually lead to full synchronization.

The increasing and decreasing steps are related to the attractive and repulsive steps during the mutual learning process between tree parity machines [11]. However, attractive steps have the property to diminish the distance between corresponding weights, while repulsive steps increase this distance.

4.2. Synchronization

The results obtained in the previous section can be used to show that during the mutual learning process synchronization can be achieved, with high probability, in a finite number of rounds. Moreover, we estimate the synchronization time of the mutual learning process.

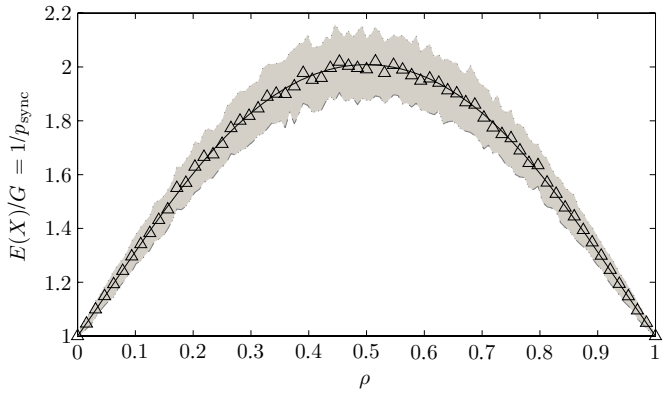

Figure 10. Expected value of the inner round duration in dependence of normalized overlap for the parameters $G = 128$ and $N = 9$. The shaded area denotes the standard deviation measured via 100 simulations.

First, the number of inner rounds performed during an outer round is estimated. This requires to study how the buffers are filled during mutual learning taking into account the variation of the overlap between the state vectors.

Let p_{sync} denote the probability that a synchronization step between the networks occurs. When a synchronization step takes place, the machines fill the buffers by the output bits of their first perceptrons. The probability of a synchronization step with the given overlap R between the state vectors was stated in (39). This probability does not change during a series of inner rounds since the overlap R remains constant.

Let X denote the random variable that provides the number of times the outputs of the networks are exchanged before the buffers are filled; that is, the number of inner rounds during an outer round. By definition, the variable X takes on values $\geqslant G$. The random variable X follows a negative binomial distribution [12], and the probability that the variable X takes on the value T is given by

$$P(X = T) = \binom{T-1}{G-1} (p_{\text{sync}})^G (1 - p_{\text{sync}})^{T-G}. \tag{52}$$

The expected value of the random variable X is given by

$$E(X) = \frac{G}{p_{\text{sync}}} \tag{53}$$

and illustrated in figure 10 as a function of the normalized overlap $\rho = R/G$. The solid line in this figure is computed from (53), while the symbols are denoting averages obtained by 100 simulations with the parameters $G = 128$ and $N = 9$. If the state vectors are aligned in parallel or anti-parallel, the number of inner rounds become minimal; while if the state vectors are uncorrelated ($\rho = 0.5$), the expected number of inner rounds becomes maximal.

After each outer round, the state vectors are replaced by the buffers of the respective machines. It follows that the learning process given by the sequence of outer rounds $(R(t) \mid t \in \mathbb{N}_0)$ can be described as a first-order Markov chain [13]. The transition probabilities of this Markov chain correspond to the conditional probabilities that given an overlap $R(t) = R$ between the state vectors, the overlap in the next outer round is $R(t+1) = R'$:

$$p_{R,R'} = P(R(t+1) = R' \mid R(t) = R), \qquad R, R' \in \mathcal{S}, \tag{54}$$

where $\mathcal{S} = \{n \in \mathbb{N}_0 \mid 0 \leqslant n \leqslant G\}$ is the state space of the Markov chain. The overlap R' in the outer round $t+1$ is given by the number of increasing steps during the current round t. Thus $p_{R,R'}$ is the probability of performing R' increasing steps in a sequence of G synchronization steps during one outer round with the overlap $R(t) = R$. This probability follows a binomial distribution:

$$p_{R,R'} = \binom{G}{R'} (p_{R,\mathrm{inc}})^{R'} (1 - p_{R,\mathrm{inc}})^{G-R'}, \tag{55}$$

where $p_{R,\mathrm{inc}}$ denotes the probability that an increasing step takes place when the overlap between the state vectors equals R; this probability was already given in (47). Since the probability $p_{R,\mathrm{inc}}$ is independent of round t, the sequence $(R(t) \mid t \in \mathbb{N}_0)$ amounts to a finite homogeneous Markov chain with state space $\mathcal{S}$ [14].

The transition probabilities of the Markov chain are given by the $(G+1) \times (G+1)$ matrix $\boldsymbol{P} = (p_{R,R'})$. As stated before, the probability of increasing steps $p_{R,\mathrm{inc}}$ depends on the ratio R/G. Moreover, if the parameter G is large enough (while $p_{R,\mathrm{inc}}$ stays fixed), the binomial distribution can be approximated by the normal distribution with mean $G \cdot p_{R,\mathrm{inc}}$ and variance $G \cdot p_{R,\mathrm{inc}} \cdot (1 - p_{R,\mathrm{inc}})$ [12].

Second, the convergence of the Markov chain is investigated. For this, we first consider the case when the parameter $N \geqslant 3$ is odd. From the definitions we find that $p_{0,\mathrm{inc}} = 0$ and $p_{G,\mathrm{inc}} = 1$ (figure 9(*a*)). Therefore, it follows from (55) that $p_{0,0} = (1 - p_{0,\mathrm{inc}})^G = 1$ and $p_{G,G} = (p_{G,\mathrm{inc}})^G = 1$ as well. These states are called absorbing, because once the Markov chain reaches an absorbing state, it cannot leave it [14]. These states provide synchronization that result in anti-parallel ($\rho = 0$) or parallel ($\rho = 1$) alignment of the state vectors. Claim that the remaining states $R, 0 < R < G$, are transient; that is, given that the Markov chain starts in the state R there is a non-zero probability of never returning to state R.

First, consider the cases $R = 1$ and $R = G - 1$. From the definitions, we obtain that $P_1\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) = 0$. It follows from (47) that $p_{1,\mathrm{inc}} = 0$ and, by the symmetry given in (49), $p_{G-1,\mathrm{inc}} = 1$. By (55), we have that $p_{1,0} = 1 = p_{G-1,G}$. Thus the states $R = 1$ and $R = G - 1$ are transient, since they always lead to the absorbing states $R' = 0$ and $R' = G$, respectively.

Second, consider the case $2 \leqslant R \leqslant G - 2$. Claim that the probabilities $P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right)$ and $P_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right)$ are larger than 0. Indeed, it is sufficient to show that for some values of r_1 and r_2, the term $q_{r_1,N} \cdot f_{r_1;G,R,N} \cdot q_{r_2,N} \cdot f_{r_2;G-N,R-r_1,N}$ of $P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right)$ is larger than 0, and the term $(1 - q_{r_1,N}) \cdot f_{r_1;G,R,N} \cdot (1 - q_{r_2,N}) \cdot f_{r_2;G-N,R-r_1,N} \mathrm{P}_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right)$ is larger than 0, too.

For this, note that $f_{r;G,R,N} > 0$ if $0 \leqslant r \leqslant R \leqslant G - N + r$. Thus $f_{r_1;G,R,N} \cdot f_{r_2;G-N,R-r_1,N} > 0$ if $0 \leqslant r_1 + r_2 \leqslant R \leqslant G - 2N + r_1 + r_2$. Moreover, from (29) and (31) it follows that $q_{r,N} > 0$ if $0 < r \leqslant N$ and $1 - q_{r,N} > 0$ if $0 \leqslant r < N$. Thus $q_{r_1,N} \cdot f_{r_1;G,R,N} \cdot q_{r_2,N} \cdot f_{r_2;G-N,R-r_1,N} > 0$ and $(1-q_{r_1,N}) \cdot f_{r_1;G,R,N} \cdot (1-q_{r_2,N}) \cdot f_{r_2;G-N,R-r_1,N} > 0$ provided that $r_1 + r_2 \leqslant R \leqslant G - 2N + r_1 + r_2$ and $0 < r_1, r_2 < N$. Then $r_1 + r_2 \geqslant 2$ and, since by definition $G \geqslant 2N$, $G - 2N + r_1 + r_2 \leqslant G - 2$, hence the values r_1 and r_2 will exist. The claim follows.

Using this result, the probability of increasing steps (47) satisfies

$$0 < p_{R,\mathrm{inc}} = P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) < 1, \qquad 2 \leqslant R \leqslant G - 2.$$

Thus the transition probability $p_{R,R'}$ is larger than 0 for $2 \leqslant R \leqslant G - 2$ and $0 \leqslant R' \leqslant G$. That is, there is a non-zero probability of reaching an absorbing state from the state $R, 2 \leqslant R \leqslant G - 2$, which implies a non-zero probability of never returning to the state R.

Therefore, the Markov chain only contains absorbing and transient states and is thus called *absorbing* [14]. It follows that regardless of the initial state, the probability of reaching an absorbing state after t rounds tends to 1 as t tends to infinity [15]. Consequently, the Markov chain always converges when the parameter N is odd.

On the other hand, when the parameter $N \geqslant 2$ is even, we have already seen from (44) and (47) that $p_{G,\mathrm{inc}} = 1$ and thus $p_{G,G} = 1$; that is, the state $R = G$ is absorbing. We shall show that the remaining states $R, 0 \leqslant R < G$ are transient. First, consider the case $R = G - 1$. By (34), we have $q_{N,N} = 1$. Thus it follows from the definitions that $P_{G-1}\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right) > 0$ and $P_{G-1}\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right) = 0$. Thus the probability of increasing steps (47) equals $p_{G-1,\mathrm{inc}} = 1$ and hence by (55) the transition probability $p_{G-1,G}$ is 1. Therefore, the state $R = G - 1$ is transient, since it always leads to an absorbing state.

Second, consider the case $0 \leqslant R \leqslant G - 2$. Claim that the probabilities $P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right)$ and $P_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right)$ are greater than 0. Indeed, it is sufficient to prove that for some values of r_1 and r_2, the term $q_{r_1,N} \cdot f_{r_1;G,R,N} \cdot q_{r_2,N} \cdot f_{r_2;G-N,R-r_1,N}$ of $P_R\left(\sigma_1^A = \sigma_1^B, \sigma_2^A = \sigma_2^B\right)$ and the term $(1 - q_{r_1,N}) \cdot f_{r_1;G,R,N} \cdot (1 - q_{r_2,N}) \cdot f_{r_2;G-N,R-r_1,N}$ of $P_R\left(\sigma_1^A \neq \sigma_1^B, \sigma_2^A \neq \sigma_2^B\right)$ are larger than 0. For this, we already mentioned that $f_{r_1;G,R,N} \cdot f_{r_2;G-N,R-r_1,N} > 0$ if $0 \leqslant r_1 + r_2 \leqslant R \leqslant G - 2N + r_1 + r_2$. Moreover, it follows from (31) and (34) that $q_{r,N} > 0$ if $0 \leqslant r \leqslant N$ and $1 - q_{r,N} > 0$ if $0 \leqslant r < N$. Thus the terms $q_{r_1,N} \cdot f_{r_1;G,R,N} \cdot q_{r_2,N} \cdot f_{r_2;G-N,R-r_1,N}$ and $(1 - q_{r_1,N}) \cdot f_{r_1;G,R,N} \cdot (1 - q_{r_2,N}) \cdot f_{r_2;G-N,R-r_1,N}$ are larger than 0 provided that $r_1 + r_2 \leqslant R \leqslant G - 2N + r_1 + r_2$ and $0 \leqslant r_1, r_2 < N$. Then $r_1 + r_2 \geqslant 0$ and, since by definition $G \geqslant 2N$, $G - 2N + r_1 + r_2 \leqslant G - 2$, it follows that the values r_1 and r_2 will exist. The claim follows.

Using this result, the probability of increasing steps (47) satisfies

$$0 < p_{R,\mathrm{inc}} = P_R\left(\sigma_1^A = \sigma_1^B \,\middle|\, \tau^A = \tau^B\right) < 1, \qquad 0 \leqslant R \leqslant G - 2.$$

Therefore the transition probability $p_{R,R'}$ is larger than 0 for $0 \leqslant R \leqslant G-2$ and $0 \leqslant R' \leqslant G$. This means, there is a non-zero probability of reaching the absorbing state from the state $R, 0 \leqslant R \leqslant G - 2$, and therefore there is a non-zero probability of never returning to the state R. Thus, if N is even, the Markov chain is absorbing and always converging on the only absorbing state.

In order to illustrate the stationarity of the Markov chain, the chain is described by the sequence of probability distributions $\boldsymbol{q}(t) = (q_0(t), \ldots, q_G(t))$ for all outer rounds $t \geqslant 0$, where the entry $q_R(t)$ denotes the probability that the state vectors have overlap R during outer round t. We may assume that at the beginning of the learning phase, the probability of increasing steps $p_{R,\mathrm{inc}}$ equals 0.5. The initial distribution of the Markov chain then amounts to

$$q_R(0) = P(R(0) = R) = \frac{1}{2^G}\binom{G}{R}, \qquad R \in \mathcal{S}. \tag{56}$$

The time evolution of the Markov chain is given by the equation $\boldsymbol{q}(t) = \boldsymbol{P}^t \boldsymbol{q}(0)$ for each outer round $t \geqslant 0$. In particular, the time evolution of the probability distribution $\boldsymbol{q}(t)$ in dependence of the normalized overlap is shown in figure 11. When the parameter N is odd, the Markov chain converges to one of two absorbing states $R = 0$ and $R = G$, both with probability $q_0(t) = q_G(t) = 0.5$ (figure 11(*a*)). When the parameter N is even, the Markov chain converges to the absorbing state $R = G$ with probability $q_G(t) = 1$ (figure 11(*b*)).

Third, we study the number of outer rounds required to achieve synchronization; that is, the number of steps until the Markov chain reaches an absorbing state. Let $\mathcal{T}$ denote the set of transient states. By the proof of convergence of the Markov chain, $\mathcal{T} = \{1, \ldots, G - 1\}$ if

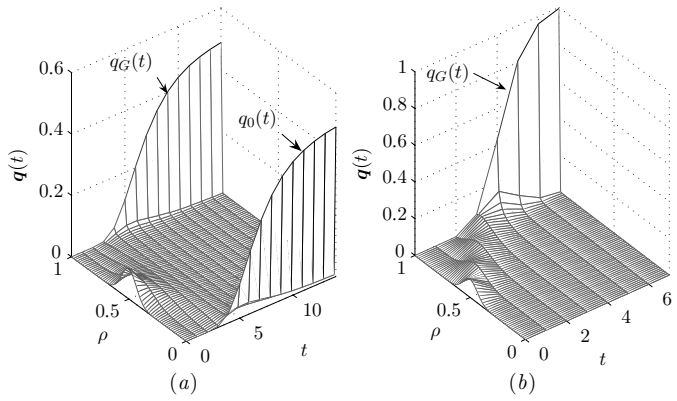

Figure 11. Time evolution of the Markov chain in dependence of normalized overlap for the parameter $G = 64$. (a) $N = 3$, (b) $N = 4$.

the parameter N is odd, and $\mathcal{T} = \{0, \ldots, G-1\}$ otherwise. Define the transient matrix of the Markov chain as $\boldsymbol{U} = (u_{R,R'})_{R,R' \in \mathcal{T}}$, where $u_{R,R'} = p_{R,R'}$ denotes the transition probability.

Let ν_R denote the random variable that describes the number of steps in which the Markov chain is a transient state starting from the initial state R. For an absorbing Markov chain, the expected value of this variable is given by [14, section 3]

$$E(\nu_R) = \sum_{R' \in \mathcal{T}} l_{R,R'}, \qquad R \in \mathcal{T}, \tag{57}$$

where $\boldsymbol{L} = (l_{R,R'})_{R,R' \in \mathcal{T}} = (\boldsymbol{I} - \boldsymbol{U})^{-1}$ and $\boldsymbol{I}$ is the identity matrix.

This expectation is illustrated in figure 12 as a function of the normalized overlap $\rho = R/G$ for some values of the parameter N. When the parameter N is odd, the expectation $E(\nu_R)$ is symmetric around the overlap $\rho = 0.5$, $E(\nu_R) = E(\nu_{G-R})$, due to the symmetry of the probability of increasing steps (49). Moreover, the expectation $E(\nu_R)$ attains the maximum at the overlap $\rho = 0.5$, and becomes minimal when the state vectors have maximal ($\rho = 1$) or minimal ($\rho = 0$) overlap (figure 12(a)).

When the parameter N is even, the expectation attains the minimum at $\rho = \frac{G-1}{G}$ with a minimum value $E(\nu_{G-1}) = 1$. Furthermore, the parameter N and the expectation increase with the same order of magnitude. In particular, the expectation for small overlaps becomes very large, when compared with the case of N odd (figure 12(b)). The reason is that when N is large, then by (34) $q_{0,N}$ tends to 0 and thus the probability of decreasing steps (51) tends to 1. It follows that the transition probability $p_{0,0}$ tends to 1, too. That is, the transient state $R = 0$ tends to become an absorbing state such that the states near $R = 0$ almost form a so-called closed subset $\mathcal{M} \subset \mathcal{S}$ with the property $\sum_{R' \in \mathcal{M}} p_{R,R'} \approx 1$ for each $R \in \mathcal{M}$; the probability of leaving $\mathcal{M}$ given that the chain started in a state belonging to the closed set $\mathcal{M}$ is almost 0 [14].

Let X_{sync} denote the random variable which describes the number of outer rounds required to achieve synchronization given an initial distribution $\boldsymbol{q}(0)$ of the chain. The expected value of the random variable X_{sync} is given by

$$E(X_{\text{sync}}) = \sum_{R \in \mathcal{T}} E(\nu_R) \cdot q_R(0), \tag{58}$$

where $q_R(0)$ is the probability to start in the initial state R. The expectation $E(X_{\text{sync}})$ behaves logarithmically with respect to the parameter G (figure 13). In contrast to this,

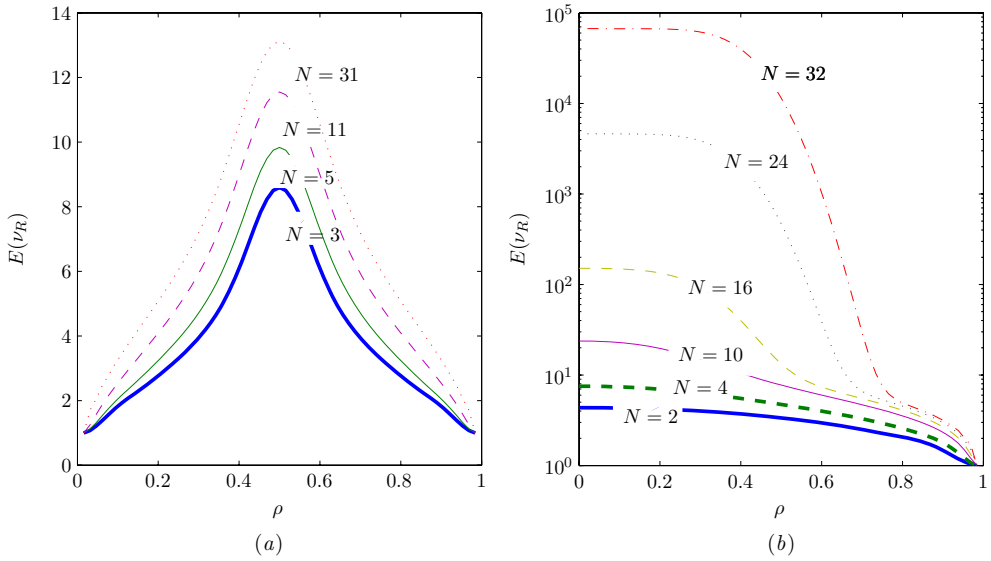

Figure 12. Expected value of the number of outer rounds as a function of the initial overlap, for the parameter $G = 64$. (*a*) N odd, (*b*) N even.

(This figure is in colour only in the electronic version)

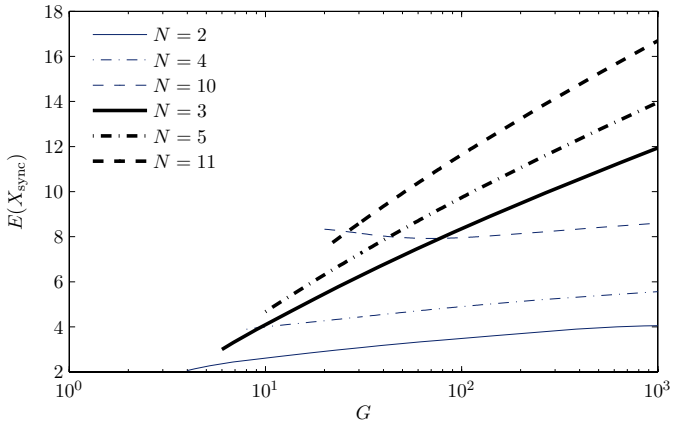

Figure 13. Expected value of synchronization time $E(X_{\text{sync}})$ as a function of the parameter G.

the synchronization time of the learning process between tree parity machines is proportional to both, $\ln K N$ and L^2 for $K \leqslant 3$ [16] (section 2).

Fourth, we need a criterion showing convergence of the mutual learning process, given the fact that the partners cannot directly compare their state vectors in order to verify alignment. When the networks are synchronized, the probability of synchronization steps p_{sync} becomes 1 and from (52) it follows that $P(X = T) = 1$ in the case of $T = G$. Thus when the networks exchange their outputs exactly G times to fill their buffers, we can assume that the mutual learning process has converged. In contrast to this, the learning process between two tree parity machines makes use of an empirical criterion which implies that the outputs of the

networks must agree during a certain number of consecutive learning steps (usually 100–400 steps) in order to presume synchronization [2, 7].

Finally, in the case of an anti-parallel alignment of the state vectors when the number of inputs N is odd, an additional learning step is required to make both state vectors equal. This step can be provided by a simple parity verification which can be performed without an additional exchange of information between the networks. For this, take a mapping $f : \{0, 1\}^G \to \{0, 1\}^G$ with the property

$$f(\overline{\boldsymbol{s}}) = \overline{f(\boldsymbol{s})}, \qquad \boldsymbol{s} \in \{0, 1\}^G, \tag{59}$$

where the binary complement is taken component-wise. Such a mapping is called *even*. If $\boldsymbol{s}^A = \overline{\boldsymbol{s}^B}$, then we have

$$\boldsymbol{s}^A \oplus f(\boldsymbol{s}^A) = \overline{\boldsymbol{s}^B} \oplus f(\overline{\boldsymbol{s}^B}) = \overline{\boldsymbol{s}^B} \oplus \overline{f(\boldsymbol{s}^B)} = \boldsymbol{s}^B \oplus f(\boldsymbol{s}^B). \tag{60}$$

Consequently, if the state vectors $\boldsymbol{s}$ and their images $f(\boldsymbol{s})$ are added (exclusive or), the derived vectors $\boldsymbol{s} \oplus f(\boldsymbol{s})$ will be the same in both networks. A typical even function is given by the assignment $f : (s_1, \ldots, s_G) \mapsto (s_1 \oplus s_G, \ldots, s_G \oplus s_G)$; in this case, the last bit in each state vector becomes 0.

5. Conclusions

We introduced the permutation parity machine as a binary variant of the tree parity machine. Its learning rule involves inner and outer rounds that makes the permutation parity machines suitable for bit-packaging implementations without affecting the dynamics of the learning process.

Neural synchronization of permutation parity machines was proved for networks with two hidden units, regardless of the number of inputs and the length of the state vectors. The dynamics of the learning process between two permutation parity machines significantly depends on whether the number of inputs is even or odd; it crucially affects the probability distribution of the state vectors after synchronization. In both cases, this process is a result of competing stochastic forces given by increasing and decreasing steps, but while two permutation parity machines with an even number of inputs achieve synchronization by the dominance of the increasing steps, if the number of inputs is odd, the process exhibits a bivergent behavior, in which synchronization is attained in one of two possible alignments, both with the same probability.

The dynamics of the mutual learning process of permutation parity machines with two hidden units ($K = 2$) is not significantly influenced by the length G of the state vectors, especially in the case of $G \gg N$. The process mainly depends on the number of inputs per hidden unit and on the overlap.

The synchronization time of the mutual learning process depends on the parameters G and N and whether N is even or odd. The length G of the state vectors directly affects the number of inner rounds required to fill the buffer, while the number of outer rounds is influenced by the number of inputs N and increases proportional to $\ln G$. In particular, if the number of inputs per hidden unit N is even and large, the synchronization time significantly increases.

Permutation parity machines make use of a more complex learning rule than that of tree parity machines, especially due to the necessity to extract entries from the respective state vectors at random. This issue becomes particularly important when the machines will be implemented in hardware. However, this step can be easily implemented by using a linear feedback shift register. Nevertheless, the simplicity of the network compensates for the complexity of the learning rule. Moreover, the synchronization of two permutation parity

machines can be detected with certainty by counting the number of learning steps, while there is no such simple criterion for tree parity machines.

In view of neural synchronization, permutation parity machines form a viable alternative to tree parity machines. In a forthcoming paper, we will study the application of permutation parity machines to symmetric key exchange.

Acknowledgments

O M Reyes acknowledges the support from German Academic Exchange Service (DAAD) and Colombian Institute for the Development of Science and Technology, 'Francisco José de Caldas' – Colciencias.

References

[1] Pikovsky A, Rosenblum M and Kurths J 2001 *Synchronization: A Universal Concept in Nonlinear Sciences* (Cambridge: Cambridge University Press)
[2] Kanter I, Kinzel W and Kanter E 2002 *Europhys. Lett.* **57** 141–7
[3] Haykin S 1998 *Neural Networks* (New York: Prentice-Hall)
[4] Rosen-Zvi M, Kanter I and Kinzel W 2002 *J. Phys. A: Math. Gen.* **35** L707–13
[5] Rosen-Zvi M, Klein E, Kanter I and Kinzel W 2002 *Phys. Rev.* E **66** 066135
[6] Behroozi N 2005 Realisierung eines Embedded Systems zur Integration eines Schlüsselaustauschverfahrens mittels Tree Parity Machines in Wireless LAN *Master's Thesis* Hamburg University of Technology, Hamburg
[7] Volkmer M and Wallner S 2005 *IEEE Trans. Comput.* **54** 421–7 (ISSN 0018-9340)
[8] Kopitzke I 2007 Modifizierung der Tree Parity Machine für den Schlüsselaustausch *Master's Thesis* Hamburg University of Technology, Hamburg
[9] Engel A and den Broeck C V 2004 *Statistical Mechanics of Learning* (Cambridge: Cambridge University Press)
[10] Feller W *An Introdution to Probability Theory and its Applications* 3rd edn (New York: Wiley)
[11] Ruttor A, Kinzel W, Shacham L and Kanter I 2004 *Phys. Rev.* E **69** 046110
[12] Shao J 1999 *Mathematical Statistics* (New York: Springer)
[13] Yeung R 2008 *Information Theory and Network Coding* 2nd edn (Boston, MA: Springer)
[14] Iosifescu M 1980 *Finite Markov Processes and Their Application* (Chichester: Springer)
[15] Kemeny J and Snell J 1983 *Finite Markov Chains* 3rd edn (New York: Springer)
[16] Mislovaty R, Kanter I and Kinzel W 2002 *Phys. Rev.* E **66** 066102