A High-Storage Capacity Content-Addressable Memory and Its Learning Algorithm

MICHEL VERLEYSEN, MEMBER, IEEE, BRUNO SIRLETTI, MEMBER, IEEE, ANDRÉ VANDEMEULEBROECKE, AND PAUL G. A. JESPERS, FELLOW, IEEE

Abstract —**Hopfield's neural networks show retrieval and speed capabilities that make them good candidates for content-addressable memories (CAM's) in problems such as pattern recognition and optimization. This paper presents a new implementation of a VLSI fully interconnected neural network with only two binary memory points per synapse (the connection weights are restricted to three different values: $+1$, 0 and -1). The small area of single synaptic cells (about 10^4 μm^2) allows the implementation of neural networks with more than 500 neurons. Because of the poor storage capability of Hebb's learning rule, especially in VLSI neural networks where the range of the synapse weights is limited by the number of memory points contained in each connection, a new algorithm is proposed for programming a Hopfield neural network as a high-storage capacity CAM. The results of the VLSI circuit programmed with this new algorithm are very promising for pattern recognition applications.**

I. Introduction

CONNECTIONIST architectures offer very interesting possibilities to solve a large class of associative memory and pattern recognition problems. Hopfield [1] in 1982 proposed a simplified model of the human brain's structure based on the approach of McCullogh and Pitts [2]. In this model, each processing element (neuron) can be connected to the other neurons through a resistive coupling network. The connections, called synapses, can be either excitatory (positive weight) or inhibitory (negative weight). All the information stored in the network lies in the connection values and is consequently distributed within the whole system; the neurons only perform the sum of all their inputs (which are the outputs of the other neurons weighed by the synaptic strengths).

Hopfield's model led a new class of VLSI neural associative memories using Hebb's learning rule [3] to record patterns (i.e., to set the correct connection values). Although this rule is very simple to implement the results are not completely satisfactory: the storage capacity is poor compared to the number of memory points contained in the network; Hopfield showed that only about 0.15n n-bit patterns could be correctly stored in a n-neuron network. Beyond this limit, poor retrieval occurs and even instability is experienced for input patterns close to stored patterns. Instability appears when we suppress Hopfield's hypothesis that two changes in the neuron values cannot occur at the same time; this is indeed impossible to foretell in a VLSI neural network where all the neuron values are computed simultaneously [4]. Moreover, below the 0.15n limit, stored patterns have to be strongly different from each other to ensure a good learning behavior. When an input pattern is presented to the network, three different behaviors may be observed: convergence to the closest learned pattern, convergence to an unwanted pattern or instability (no convergence at all). This is illustrated in Fig. 1 for Hebb's rule (the X-axis represents the Hamming distance, i.e., the number of different bits between the input patterns and the closest learned pattern, while the Y-axis shows the percentage of each of the three observed states for three recorded patterns in a 12-bit Hopfield network).

Another problem arises when such a content-addressable memory (CAM) is implemented within a VLSI chip. A synapse weight computed accordingly to Hebb's rule may take any integer value between p and $-p$, where p is the number of stored patterns [5]; since there are $2p+1$ different synaptic values, a memory with $\log_2(2p+1)$ bits is necessary in each synapse. Hence the area of a synaptic cell grows quickly with the number of recorded patterns since p can be considerable in large networks. Furthermore, in a fully interconnected VLSI neural network, the main part of the circuit area consists of synapses. To increase the number of neurons on a single chip one must therefore be able to decrease the area of synaptic cells. A good tradeoff between the requirements of Hebb's rule and the restricted available area on the chip is thus required.

The objective of this paper is to propose a complete CAM system using a neural network; it is divided into two parts: the first one presents a VLSI implementation of a Hopfield fully interconnected neural network, with only three different weight values stored in each synapse $(+1, 0, -1)$, and the second describes a new learning method to record patterns in the CAM.

Manuscript received June 30, 1988; revised October 24, 1988 and December 15, 1988. M. Verleysen and B. Sirletti was supported by an IRSIA fellowship. A. Vandemeulebroecke was supported by a FNRS fellowship. This paper was recommended by Guest Editors R. W. Newcomb and N. El-Leithy.
The authors are with the Université Catholique de Louvain, Laboratoire de microélectronique, 1348 Louvain-la-Neuve, Belgium.
IEEE Log Number 8826707.

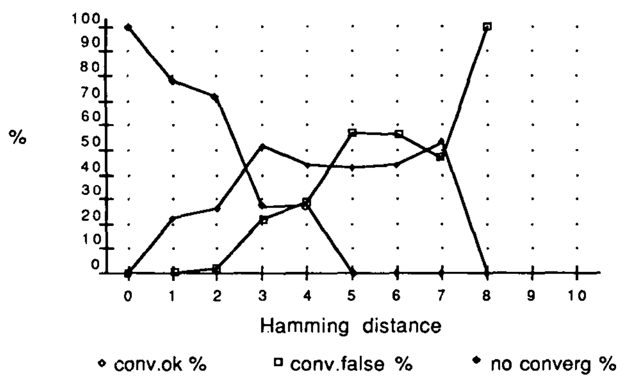

Fig. 1. Hebb's rule: simulation results. Three 12-bit patterns have been stored in a 12-neuron network using Hebb's rule. All the 2^{12} possible patterns have been introduced at the network inputs; the diagram shows the percentage of well-retrieved, nonretrieved and unstable patterns versus the Hamming distance (number of different bits) between the input pattern and the closest learned pattern.

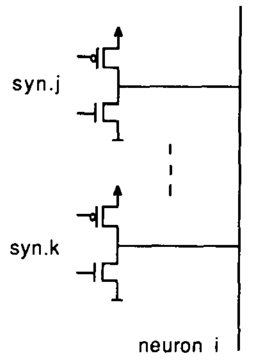

Fig. 2. *P*- and *n*-type transistor synapses.

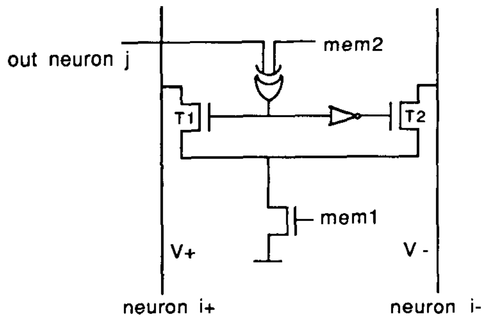

Fig. 3. Synapse between neurons *i* and *j*.

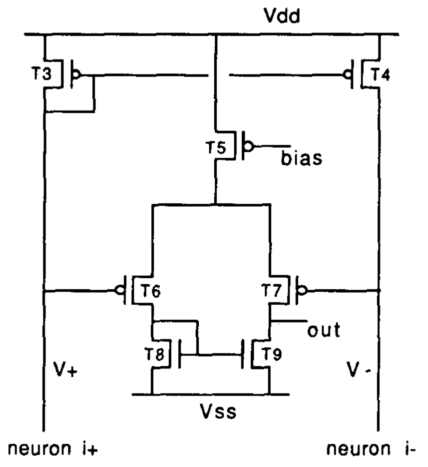

Fig. 4. Neuron with differential amplifier.

II. VLSI Hopfield Neural Network

The purpose of the proposed circuit is to allow the realization of very large neural networks. In the first implementations of fully interconnected neural networks, synapses consisted only of resistive connections between neurons [6]. Such a class of circuits introduces special technological requirements and therefore is not often used. In more recent applications [7], synapses consist of current sources controlled by the output of the connected neuron: the excitatory currents come from *p*-type transistors and the inhibitory currents from *n*-type ones (see Fig. 2).

One of the problems that arises with this kind of synapse is that it is difficult to make the current injected by the *p*-type transistors equal to the one sunk by the *n*-type transistors. The mobility differences between the two types of transistors can be compensated by using a width scaling; however, threshold voltage variations due to the technological processes involved in integrated circuit fabrication make it impossible to exactly compensate these differences by different transistor dimensions between p- and n-MOS's (a factor of 2.5 to 3 is usually used, but must be empirically determined). Since the function of each neuron is to detect the sign of the sum of the other neuron values, weighed by the synapses strengths, the possible mismatches between the sourced and sunk currents are summed with an increasing number of synapses. In order to make a neuron able to discriminate the sign of its inputs even when the difference between excitatory and inhibitory currents equals only a single synaptic current, the latter must exceed *n* times the difference between the *p*- and *n*-type current sources; this limits considerably the size of the network.

This paper proposes a new architecture for neural associative memories in which the sourced and sunk currents are summed separately on two different lines. Each synapse is a current source (Fig. 3) programmed by *mem1*; if *mem1* = 0, neurons *j* and *i* are not connected together.

Mem2 determines the sign of the connection, i.e., if the current must be sourced or sunk. In the first case *T1* derives current from the line $i+$, in the second one *T2* derives it from the line $i-$. The function of the neuron is thus to compare the total currents on lines $i+$ and $i-$; this is done by means of the current mirror described in Fig. 4. The currents on these lines are converted into voltages across transistors *T3* and *T4*; these voltages are themselves compared in the differential amplifier formed by transistors *T5* to *T9*. Because of the two-stage architecture of the neuron, its gain is very important and the output (*out*) is either V_{dd} if the current in *neuron* $i-$ is greater than the one in *neuron* $i+$, or V_{ss} in the opposite case.

With an increasing number of connected synapses, voltage drops in the neuron on lines $i+$ and $i-$ ($V+$ and $V-$ in Fig. 4) of course tend to decrease. Hence the drains of *T1* and *T2* may influence to some extent the currents

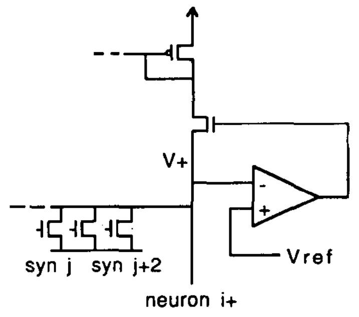

Fig. 5. Feedback loop to avoid current decrease.

injected in the $i+$ and $i-$ lines. To avoid this effect, a feedback loop is introduced as shown in Fig. 5. The voltages $V+$ and $V-$ are thus fixed to V_{ref}, and the currents sunk in all synapses no longer vary. Since no high gain is needed for the feedback loop, the amplifier shown in Fig. 5 can be very simple; SPICE simulations showed that with a single transistor for the feedback loop, voltage $V+$ is fixed with an accuracy of more than 95 percent with up to 500 synapses connected; moreover, with such accuracy, the drain voltages of *T1* and *T2* will no longer significantly influence the current sourced in the synapses. The same feedback loop is inserted in line $i-$ to keep voltage V-fixed.

SPICE simulations of this circuit were carried out to determine the number of neurons that can be connected together. The criterion was that the output of the neuron (Fig. 4) must be either high or low enough to drive correctly a buffer inserted between the output of the neuron and the inputs of the synapses. Voltages of 3 and 2 V were chosen respectively as lower bound for output voltage of the neuron if current in $i+$ is greater than current in $i-$ line, and as upper bound for output voltage in the opposite case; an inverter placed at the output of the neuron can indeed be easily designed to have its switching voltage between 2 and 3 V. The results of the SPICE simulations were satisfactory: about 110 neurons can be connected together without feedback loop, and about 520 with a feedback loop.

In order to prove the simulation results and to measure some interesting properties of this architecture (synaptic currents, maximum number of connected neurons), a test chip with 14 neurons and 196 synapses has been realized in a CMOS 3-μm technology. In this circuit, the single synaptic current equals 10 μA; to achieve this, the synaptic transistor connected to *mem1* (*T0* in Fig. 3) is long ($W/L = 0.1$), while *T1* and *T2* are minimal ($W/L = 1.5$). Such current of 10 μA is acceptable in a neural network with a restricted number of neurons; in larger networks, it has to be reduced for power density reasons. For example, the power dissipated by a 128-neuron network with synaptic currents equal to 1 μA will be about 100 mW; such circuit can be made in a 64-mm^2 chip with a CMOS 2-double metal technology (power density is about 1 mW/mm^2).

Speed properties are one of the most interesting features of Hopfield neural networks. Tests showed that a change in a synapse value introduces a change in the corresponding neuron value and is fed back in the synapse in about 30 ns. Practically, this means that it takes about 120–150 ns for a 128-neuron network to converge to a stable state.

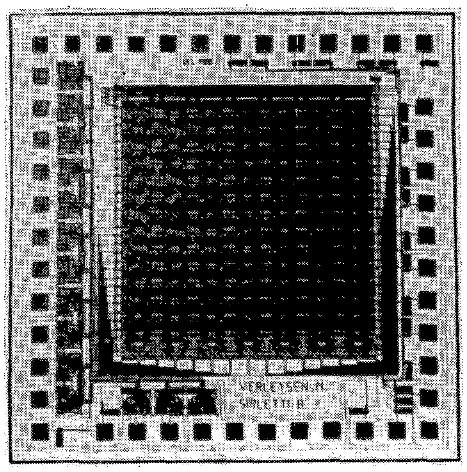

Fig. 6. Photomicrograph of the chip.

Since this circuit is fully programmable (each connection strength can be programmed to +1, 0, and −1 by setting appropriate values in *mem1* and *mem2* of Fig. 3), it can be programmed to solve optimization problems, or can be used as a content-addressable memory as described below. Fig. 6 shows a photomicrograph of the chip; its size is 3 mm×3 mm. The central and main part of the chip contains the 196 synapses (115×108 μm^2 each); below the synapses are the 14 neurons (115×75 μm^2 each), and above the synapses is a decoder used to program the RAM contained in the synapses (*mem1* and *mem2* of Fig. 3). The technology used is CMOS 3-μm with single metal and single poly.

III. The Learning Algorithm for the CAM

A new way to compute the connection strengths is proposed, using a linear algebra optimization method (Simplex) in order to maximize the stability of the recorded patterns. In Hopfield's model, each neuron is connected to every other neuron. The connections between neurons can thus be represented by a $(n \times n)$ matrix where element T_{ij} is the value of the connection between neuron i and neuron j. The proposed algorithm allows the matrix to be asymmetric ($T_{ij} \neq T_{ji}$). If V_i is the state of the ith neuron (which is supposed to be Boolean in our model) and Θ the threshold value, the dynamic behavior of the network can be described by

$$V_i(t+\Delta t) = \operatorname{sign}\left(\sum T_{ij} V_j(t) - \Theta\right).$$

The network reaches a stable state when

$$V_i(t+\Delta t) = V_i(t) \qquad \forall\, i.$$

The stable states are programmed into the network by setting appropriate connection strengths (each synapse can be programmed via *mem1* and *mem2*).

To compute the connection values in a n-neuron Hopfield network where k n-bit patterns are to be stored, the different columns of the matrix are computed separately,

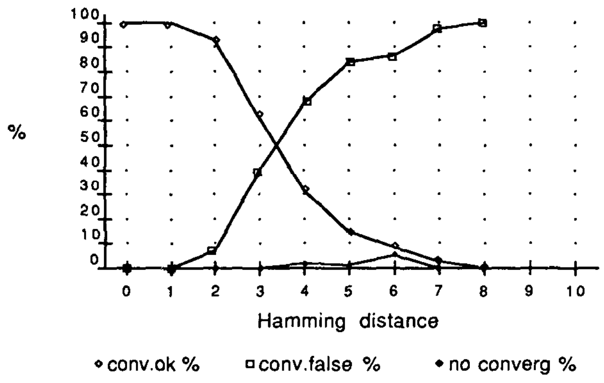

Fig. 7. Optimization method: simulation results (same simulation as in Fig. 1).

and the following procedure shows how to compute the first one (i.e., the values of the connections between any neuron and the first one); the same procedure applies to the other columns. The following notations will be used.

V_{ij}	value of the ith neuron from the jth pattern to memorize ($1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant k$, $V_{ij} = 1$ or $V_{ij} = -1$),
T_{r1}	value of the connection between neurons r and 1 ($1 \leqslant r \leqslant n$),
S_{1k}	$\Sigma T_{r1} V_{rk}$ input of the first neuron when the network outputs correspond to the training pattern k.

In this model, each neuron acts as a Boolean threshold function whose output is 1 in the case of a postive input and -1 in the other cases (Θ is thus set to 0). The essential feature of this method is to choose the set T_{r1} in order to maximize the difference between S_{1k} and the threshold of the neuron. If the sign of S_{1k} is forced to be the same as the one of V_{1k}, the highest stability for bit 1 of pattern k is obtained. To ensure the right sign to S_{1k}, the quantity to maximize is $Z_{1k} = S_{1k} V_{1k}$. This has to be done simultaneously for all values of k. The equation solved by the Simplex method is then

$$\text{maximize } M \text{ where } M = \min(Z_{1k}) \text{ for all values of } k.$$

To avoid unbounded solutions ($M \to \infty$), T_{r1} are limited by the inequalities:

$$-1 \leqslant T_{r1} \leqslant 1.$$

Practical algorithms able to solve such Simplex problems are available in the literature [8, pp. 20–46].

The results of this method can be compared with those of the Hebb's rule shown previously. Fig. 7 illustrates the convergence results for the same three memorized patterns in the same 12-bit network as in Fig. 1. The two diagrams can easily be compared: while a small enhancement in the number of unrecognized patterns can be noticed (i.e., input patterns which converge to an unwanted stable state), the percentage of well recognized patterns is strongly increased, and the number of unstable patterns (i.e., input patterns which never converge) is reduced to less than 5 percent. If the observations are restricted to the left side of the diagram (in pattern recognition problems, the Hamming distance between the input and recorded patterns is supposed to be short), the conclusion is that all patterns are correctly retrieved, while unstable states no more exist. This is not true for Hebb's rule.

Conclusion

This paper describes a system to realize a CAM with neural networks. A VLSI fully interconnected neural network has been designed, where the connection weights are restricted to only three different values; this has been done to reduce the area of a synaptic cell and thus to increase the number of neurons which can be put together in a network. The architecture proposed in this paper can be used for networks of hundreds of neurons: its only limitation is the size of the chip. A test chip with 14 neurons and 196 synapses has been realized in CMOS 3-μm technology.

A learning algorithm suitable for this architecture is also described. The convergence results obtained when programming the chip with the proposed algorithm correspond to the theoretical results of Fig. 7: the storage density is much better than the one obtained with Hebb's rule, even with only three different weights for the connections.

This algorithm used with the described circuit allows the realization of a high-storage CAM; this memory shows enhanced speed and retrieval properties due to the adequate use of neural networks.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," in *Proc. National Academy of Sciences USA*, vol. 79, no. 8, pp. 2554–2558, Washington DC, Apr. 1982.

[2] W. S. McCullogh and W. Pitts, *Bull. Math. Biophys.*, vol. 5, pp. 115–133, 1943.

[3] D. Hebb, *The Organization of Behavior.* New York: Wiley, 1949.

[4] L. Personnaz, "Etude de réseaux de neurones formels: conception, propriétés et applications," Ph.D. dissertation, University Paris, France, 1986.

[5] J. J. Hopfield and D. W. Tank, "Computing with neural circuits: a model," *Science*, vol. 233, pp. 625–633, Aug. 1986.

[6] R. Howard *et al.*, "An associative memory based on an electronic neural network architecture," *IEEE Trans. Electron Devices*, vol. ED-34, pp. 1553–1556, July 1987.

[7] H. P. Graf and P. de Vegvar, "A CMOS associative memory chip based on neural networks," in *Proc. ISSCC 1987*, New York.

[8] L. S. Lasdon, *Optimization Theory for Large Systems.* London, England: Collier-MacMillan, 1970.

Michel Verleysen (M'86) received the engineering degree from the Université Catholique de Louvain, Louvain-la-Neuve, Belgium, in 1987.

Since 1987, he was granted an IRSIA fellowship while working towards the Ph.D. degree in the field of neural networks, at the Microelectronics Laboratory of the Université Catholique de Louvain. His research activities and interest are VLSI realization of neural networks, content-addressable-memories, and analog integrated circuits and systems.

Bruno Sirletti (S'86–M'87) received the engineering degree in electronic engineering at the Université Catholique de Louvain (Belgium) in 1987.

Since 1987 he has been with Microelectronics Lab of the UCL where he is involved in the field of analog circuits and artificial neural networks with a fellowship from IRSIA.

André M. Vandemeulebroecke received the engineering degree from the Université Catholique de Louvain, Louvain-la-Neuve, Belgium, in 1983. Since 1985, he was granted an FNRS fellowship while working towards the Ph.D. degree in the field of the theory and applications of redundant numbers systems.

From 1983 to 1985, having been granted an IRSIA fellowship he worked in the field of Silicon Compilation for digital signal processors. His research activities and interest are centered around computational techniques for full custom integrated circuits, and applications to A/D conversion, cryptography, image processing and neural networks.

Paul G. A. Jespers (M'60–SM'65–F'82) received the engineering degree from the Université Libre de Bruxelles in 1953, and the Ph.D. degree from the Université Catholique de Louvain, Louvain-la-Neuve, Belgium, in 1958.

He first joined the Laboratoire Central d'Electricité, Brussels, working in RFI measurements until 1959. Since then he has been with the Department of Electrical Engineering, Université Catholique de Louvain, heading the Laboratoire de Microélectronique. He was a Visiting Professor at Stanford University, Stanford, CA, from September 1967 to January 1968. His current interest is in MOS integrated circuits and systems.

Dr. Jespers is Vice-Chairman of the Steering Committee of the European Solid-State Circuits Conference. He was appointed IEEE Regional Director of Region 8 from 1971 to 1972.