E1.4 Digital integrated circuit implementations

Valeriu Beiu

Abstract

This section considers some of the alternative approaches towards modeling biological functions by digital circuits. It starts by introducing some circuit complexity issues and arguing that there is considerable computational and physiological justification that shallow threshold gate circuits are computationally more efficient than classical Boolean circuits. We comment on the tradeoff between the depth and the size of a threshold gate circuit, and on how design parameters like fan-in, weights and thresholds influence the overall area and time performances of a digital neural chip. This is followed by briefly discussing the constraints imposed by digital technologies and by detailing several possible classification schemes as well as the performance evaluation of such neurochips and neurocomputers. Lastly, we present many typical and recent examples of implementation and mention the 'VLSI-friendly learning algorithms' as a promising direction of research.

E1.4.1 Introduction

The research on neural networks goes back to the early 1940s (see Section A1.1). The seminal year for the development of the 'science of mind' was 1943 when several articles were published (McCulloch and Pitts 1943, Craik 1943, Rosenbleuth *et al* 1943, 1949, Landahl *et al* 1943). A1.1

Almost immediately different approaches to neural network simulation started to be developed. Typical of that era was the development of the first neurocomputer *Snark* (Minsky 1954). It was in fact an electromechanical neurocomputer which was shortly followed by the *Perceptron Mark I* (Rosenblatt 1958). Both were using resistive circuits (motor-driven potentiometers) for implementing the weights. Another successful neurocomputer that used resistive weights was Bernard Widrow's *adaline* and, later, *madaline*. They used a type of electronically adjustable resistor called a memistor. Widrow even founded the first neurocomputer company: the Memistor Corporation. It actually produced neurocomputers during the early and mid-1960s. More details can be found in Nilsson (1965), Anderson and Rosenfeld (1988) and Hecht-Nielsen (1989). The neurocomputer industry was born. C1.1.3 C1.1.4

In the last decade the tremendous impetus of VLSI technology has made neurocomputer design a really lively research topic. Hundreds of designs have already been built, and some are available as commercial products. However, we are far from the main objective as can be clearly seen from figure E1.4.1. Here the horizontal axis represents the number of synapses (number of connections), while the vertical axis represents the processing speed in 'connections per second' (CPS). The drawing shows a crude comparison of the computational potential of different neural network hardware 'technologies'. It becomes clear that biological neural networks are far ahead of digital, analog and even future optical implementations of artificial neural networks.

Focusing only on digital implementations, this section will firstly introduce some circuit complexity issues (section E1.4.2) and comment on the constraints imposed by digital technologies (section E1.4.3). Several possible classification schemes for digital implementations of artificial neural networks will also be discussed in the last and most detailed part (section E1.4.4) which will briefly present many different implementations.

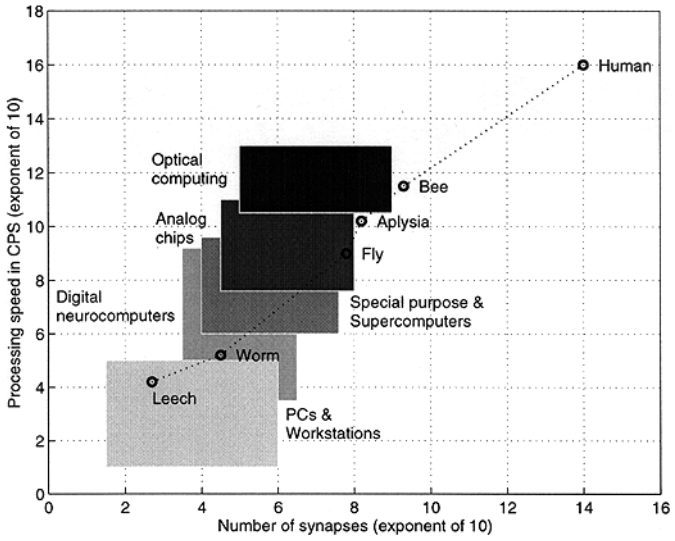

Figure E1.4.1. Different hardware alternatives for implementing artificial neural networks, an enhanced and updated version from Glesner and Pöchmüller (1994) and Iwata (1990).

E1.4.2 Circuit complexity issues

One main line of theoretical research has concentrated on the approximation capabilities of feedforward networks. It was started in 1987 by Hecht-Nielsen (1987) and Lippmann (1987) who were probably the first to point to Kolmogorov's theorem (Kolmogorov 1957), together with Le Cun (1987). The first nonconstructive proof that neural networks are universal approximators was given the following year by Cybenko (1988, 1989) using a continuous activation function. Thus, the fact that neural networks are computationally universal—with more or less restrictive conditions—when modifiable connections are allowed, was established. These results have been further enhanced by Funahashi (1989), Funahashi and Nakamura (1993), Hornik (1991, 1993), Hornik *et al* (1989, 1990), Koiran (1993) and Leshno *et al* (1993). All these results—with the partial exception of Koiran (1993)—were obtained 'provided that sufficiently many hidden units are available'. This means that no claim on the minimality of the resulting network was made, the number of neurons needed to make a satisfactory approximation being in general much larger than the minimum needed.

The other line of research was to find tight bounds, and the problem can be stated as finding the smallest network (i.e., smallest number of neurons) which can realize an arbitrary function given a set of m vectors (examples, or points) in n dimensions. If the function takes as output just 0 or 1, then it is called a dichotomy. This aspect of the smallest network is of great importance when thinking of hardware implementations. The networks considered are feedforward neural networks with threshold activation function. This is probably due to the fact that this line of research was continuing on from the rigorous results already obtained in the literature dealing with threshold logic from the 1960s (Cameron 1969, Cohen and Winder 1969, Cover 1965, Dertouzos 1965, Fischler 1962, Hu 1965, Kautz 1961, Klir 1972, Lewis and Coates 1967, Lupanov 1973, Minnick 1961, Minsky and Papert 1969, Muroga 1959–1979, Muroga *et al* 1961, Neciporuk 1964, Nilsson 1965, Red'kin 1970, Sheng 1969, Winder 1962–1971). The best result was that a *multilayer perceptron* with only one hidden layer having $m-1$ nodes could compute an arbitrary dichotomy (sufficient condition). The main improvements since then have been as follows: C1.2

- Baum (1988b) presented a network with one hidden layer having $\lceil m/n \rceil$ neurons capable of realizing an arbitrary dichotomy on a set of m points in general position in $\mathbb{R}^n$; if the points are on the corners of the n-dimensional hypercube (i.e., binary vectors), $m-1$ nodes are still needed (the general position condition is now special and strict).
- Huang and Huang (1991) proved a slightly tighter bound: only $\lceil 1+(m-2)/n \rceil$ neurons are needed in the hidden layer for realizing an arbitrary dichotomy on a set of m points which satisfy a more relaxed topological assumption as only the points forming a sequence from some subsets are required to be in general position; also the $m-1$ nodes condition was shown to be the least upper bound needed.

- Arai (1993) recently showed that $m-1$ hidden neurons are necessary for arbitrary separability (any mapping between input and output for the case of binary-valued units), but improved the bound for the two-category classification problem to $m/3$ (without any condition on the inputs).

A study which somehow tries to unify these two lines of research has been published by Bulsari (1993) who gives practical solutions for one-dimensional cases including an upper bound on the number of nodes in the hidden layer(s). Extensions to the n-dimensional case using three- and four-layer solutions are derived under piecewise constant approximations having constant or variable width partitions and under piecewise linear approximations using ramps instead of sigmoids.

To strengthen such claims, we shall go briefly through some basic circuit complexity results (Papadopoulos and Andronikos 1995, Parberry 1994, Paterson 1992, Pippenger 1987, Roychowdhury *et al* 1991a, b, 1994b, Siu *et al* 1994) and argue that there is considerable computational and physiological justification that shallow (i.e., having relatively few layers) threshold gate circuits are computationally more efficient than classical Boolean circuits. When considering computational complexity, two classes of constraints could be thought of:

- Some arising from the physical constraints (related to the hardware in which the computations are embedded) and including time constants, energy limitations, volumes, geometrical relations and bandwidth capacities.
- Others are logical constraints: (i) computability constraints and (ii) complexity constraints which give upper and/or lower bounds on some specific resource (e.g., size and depth required to compute a given function or class of functions).

The first aspect when comparing Boolean and threshold logic is that they are equivalent in the sense that any Boolean function can be implemented using either logic in a circuit of depth-2 and exponential size (simple counting arguments show that the fraction of functions requiring a circuit of exponential size approaches one as $n \rightarrow \infty$ in both cases). Yet, threshold logic is more powerful than Boolean logic as a Boolean gate can compute only one function whereas a threshold gate can compute up to the order of $2^{\alpha n^2}$ functions by varying the weights, with $1/2 \leq \alpha \leq 1$ (see Muroga 1962 for the lower bound, and Muroga 1971 and Winder 1962, 1963 for the upper bound). An important result which clearly separates threshold and Boolean logic is due to Yao (1985) (see also Håstad 1986 and Smolensky 1987) and states that in order to compute a highly oscillating function like PARITY in a constant depth circuit, at least $\exp[c(n^k)^{1/2}]$ Boolean gates with unbounded fan-in are needed (Furst *et al* 1981, Paturi and Saks 1990). In contrast, a depth-2 threshold gate circuit for PARITY has linear size.

Another interesting aspect is the tradeoff between the depth and the size of a circuit (Beiu 1994, 1997, Beiu and Taylor 1996a, Beiu *et al* 1994c, Siu and Bruck 1990c, Siu *et al* 1991b). There exists a very strong bias in favor of shallow circuits (Judd 1988, 1992) for several reasons. First, for a fixed size, the number of different functions computable by a circuit of small depth is larger than the number of those computed by a deeper circuit. Second, it is obvious that such a circuit is also faster, as having a small(er) depth. Finally, one should notice that biological circuits must be shallow—at least within certain modules like the cortical structures—as the overall response time (e.g., recognizing a known person from a noisy image) of such slow devices (the response time of biological neurons being at least in the 10-ms range due to the refractory period) is known to be in the few hundred millisecond range. Other theoretical results (Abu-Mostafa 1988a, b) also support the shallow architecture of such circuits.

A lot of work has been devoted to finding minimum size and/or minimum constant-depth threshold gate circuits (Hajnal *et al* 1987, Hofmeister *et al* 1991, Razborov 1987, Roychowdhury *et al* 1994a, Siu and Bruck 1990a, Siu *et al* 1990, 1993b, Siu and Roychowdhury 1993, 1994) but little is known about tradeoffs between those two cost functions (Beiu *et al* 1994c, Siu *et al* 1991b), and even less about how design parameters like fan-in, weights and thresholds influence the overall area and time performances of a digital neural chip. Since for the general case only existence exponential bounds are known (Bruck and Smolensky 1992, Siu *et al* 1991b), it is important to isolate classes of functions whose implementations are simpler than that of others (e.g., shallow depth and polynomial size (Rief 1987)). Several of the corner-stone results obtained so far have been gathered in table E1.4.1. Here n is the number of input variables, and the nomenclature commonly in use is (see Amaldi and Mayoraz 1992, Papadopoulos and Andronikos 1995, Parberry 1994, Roychowdhury *et al* 1994b, Siu *et al* 1994, Wegener 1987):

- AC^k represents the circuits of polynomial size with AND and OR unbounded fan-in gates and depth $\boldsymbol{O}(\log^k n)$

- NC^k is the class of Boolean functions with bounded fan-in, and having size n^c (polynomial) and depth $\boldsymbol{O}(\log^k n)$
- TC^0 the family of functions realized by polynomial size threshold gate circuits with unbounded fan-in and constant depth
- $LT_1(\widehat{LT_1})$ is the class of Boolean functions computed by linear threshold gates with real weights (bounded by a polynomial in the number of inputs $|w_i| \leq n^c$ (Bruck 1990))
- $\widehat{LT_k}$ is the class of Boolean functions computed by a polynomial size, depth-k circuit of $\widehat{LT_1}$ gates (Bruck 1990, Siu and Bruck 1990b)
- PT_1 is the class of Boolean functions that can be computed by a single threshold gate in which the number of monomials is bounded by a polynomial in n (Bruck 1990, Bruck and Smolensky 1992)
- PT_k is the class of Boolean functions computed by a polynomial size, depth-k circuit of PT_1 gates
- PL_1 is the class of Boolean functions for which the spectral norm L_1 is bounded by a polynomial in n (Bruck and Smolensky 1989)
- PL_∞ is the class of Boolean functions with the spectral norm L_∞^{-1} bounded by a polynomial in n (Bruck and Smolensky 1989)
- MAJ_1 is the class of Boolean functions computed by linear threshold gates having only ± 1 weights (Mayoraz 1991, Siu and Bruck 1990c)
- MAJ_k is the class of Boolean functions computed by a polynomial size, depth-k circuit of MAJ_1 gates (Albrecht 1992, Mayoraz 1992, Siu and Bruck 1993).

Recently three complexity classes for sigmoid feedforward neural networks have been defined and linked with the (classical) above-mentioned ones:

- NN^k is defined (Shawe-Taylor *et al* 1992) to be the class of functions which can be computed by a family of polynomially sized neural networks with weights and threshold values determined to b bits of precision (accuracy), fan-in equal to Δ and depth h, satisfying $\log \Delta = \boldsymbol{O}[(\log n)^{1/2}]$, $b \log \Delta = \boldsymbol{O}(\log n)$ and $h \log \Delta = \boldsymbol{O}(\log^k n)$
- $NN^k_{\Delta,\varepsilon}$ is defined (Beiu *et al* 1994e, Beiu and Taylor 1996b) to be the class of functions which can be computed by a family of polynomially sized neural networks which satisfies slightly less restrictive conditions for fan-in and accuracy: $\log \Delta = \boldsymbol{O}(\log^{1-\varepsilon} n)$ and $b = \boldsymbol{O}(\log^{1-\varepsilon} n)$
- NN^k_Δ is defined (Beiu *et al* 1994d, Beiu and Taylor 1996b) to be the class of functions which can be computed by a family of polynomially sized neural networks having linear fan-in and logarithmic accuracy ($\Delta = \boldsymbol{O}(n)$ and $b = \boldsymbol{O}(\log n)$).

Still, in many situations one is concerned by the values of a function for just a vanishing small fraction of the 2^n possible inputs. Such functions can also be implemented in poly-size shallow circuits (the size and depth of the circuit can be related to the cardinal of the interesting inputs (Beiu 1996b, Beiu and Taylor 1996a, Beiu *et al* 1994a, Tan and Vandewalle 1992, 1993). Such functions are also appealing from the learning point of view: the relevant inputs being nothing else but the set of training examples (Beiu 1996b, Beiu and Taylor 1995b, Linial *et al* 1989, Takahashi *et al* 1993).

Circuit complexity has certain drawbacks which should be mentioned:

- The extension of the poly-size results to other functions and to the continuous domain is not at all straightforward (Maass *et al* 1991, Siu 1992)
- Even the known bounds (for the computational costs) are sometimes weak
- Time (i.e., delay) is not properly considered
- All complexity results are asymptotic in nature and may not be meaningful for the range of a particular application.

But the scaling of some important parameters with respect to some others represents quite valuable results:

- Area of the chip (wafer) grows like the cube of the fan-in
- Area of the digital chip (wafer) grows exponentially with accuracy.

Furthermore, it was shown recently that the fan-in and the accuracy are linearly dependent parameters. If the number of inputs to one neuron is n, the reduction of the fan-in by decomposition techniques has led to the following results:

- If the fan-in is reduced to (small) constants, the size grows slightly faster than the square of the number of inputs (i.e., $n^2 \log n$) while the depth growth is lower than logarithmic (i.e., $\log n / \log \log n$)

Table E1.4.1. Circuit complexity results.

Author(s)	Result(s)	Remark(s)
Neciporuk (1964)	Lower bound on the *size* of a threshold circuit for 'almost all' n-ary Boolean functions.	$size \geq 2 \cdot (2^n/n)^{1/2}$
Lupanov (1973)	Upper bound for the *size* of a threshold circuit for 'almost all' n-ary Boolean functions.	$size \leq 2 \cdot (2^n/n)^{1/2} \times \{1 + \Omega[(2^n/n)^{1/2}]\}$ $depth = 4$
Yao (1989)	There are Boolean functions for which *depth*-$(k-1)$ threshold gate circuit of unbounded fan-in (i.e., TC^0 circuits) require exponential size *depth*-k Boolean circuits of unbounded fan-in.	Conjecture: $TC^0 \neq NC^1$
Allender (1989)	Any Boolean function computable by a polynomial size constant-*depth* logic circuit with unbounded fan-in (i.e., AC^0) is also computable by a *depth*-3 neural network (threshold gate circuit) of superpolynomial size: $AC^0 \neq TC^0$.	$size = n^{O(\log n)}$ $depth = 3$ *fan-in* unbounded
Immermann and Landau (1989)	Conjecture: $TC^0 = NC^1$	
Bruck (1990)	$LT_1 \subset PT_1 \subset LT_2$	Existence proofs
Siu *et al* (1991a)	$\mathrm{MUL}(x, y) \in \widehat{LT}_4$ $X \bmod p$, $X^n \bmod p$, $c^X \bmod p \in \widehat{LT}_2$ $X^n, c^X \in \widehat{LT}_5$ $\mathrm{DIV}(x, y) \in \widehat{LT}_5$ $\mathrm{MUL}(x_1, \ldots, x_n) \in \widehat{LT}_6$	$depth = \text{const}$ $size \leq n^c$ $weights \leq n^c$ *fan-in* unbounded Partly constructive
Siu *et al* (1991b)	Upper bound on the *size* or implementing any Boolean function. Lower bound on the *size* or implementing any Boolean function.	$depth = 3$ $size = \boldsymbol{O}(2^{n/2})$ *fan-in* unbounded Existence proofs $size = \Omega(2^{n/3})$ *fan-in* unbounded Existence proofs
Bruck and Smolensky (1992)	$PL_1 \subset PT_1 \subset PL_\infty$ $PL_1 \subset PT_1 \subset MAJ_2$ $AC^0 \not\subset PL_1$ $AC^0 \not\subset PL_\infty$ $AC^0 \not\subset MAJ_2$	Existence proofs
Siu and Bruck (1992)	$LT_1 \subset \widehat{LT}_3$ $LT_d \subseteq \widehat{LT}_{2d+1}$ $\mathrm{MUL}(x, y) \in \widehat{LT}_4$ $\mathrm{MAX}(x_1, \ldots, x_n) \in \widehat{LT}_3$ $\mathrm{SORT}(x_1, \ldots, x_n) \in \widehat{LT}_4$	$depth = \text{const}$ $size \leq n^c$ $weights \leq n^c$ *fan-in* unbounded Existence proofs
Albrecht (1992)	*Depth*-2 threshold circuits require superpolynomial *fan-in*. Polynomial threshold circuits have more than two layers.	$depth = 2$ $size = (1 \pm \varepsilon) \cdot 2^{n-1}$ $weights \in \{-1, 0, +1\}$ *fan-in* unbounded
Shawe-Taylor *et al* (1992)	$NC^k \subset NN^k \subseteq AC^k$	Partly constructive
Beiu *et al* (1994d)	$NN^k \subset NN^k_{\Delta,\varepsilon} \subset \begin{cases} NC^{k+1} \\ NN^k_\Delta \subset NC^{k+2} \end{cases}$	Constructive proofs (based on binary trees of Boolean gate adders
Beiu *et al* (1994e)	$\begin{array}{ccc} NN^k_{\Delta,\varepsilon} & \subset & NN^k_\Delta \subset \begin{cases} LT_{\log^{k+\varepsilon} n} \\ \widehat{LT}_{\log^{k+1} n} \end{cases} \\ \cap & & \cap \\ MAJ_{\log^{k+1} n} & \subset & MAJ_{\log^{k+2} n} \end{array}$	Constructive proofs (based on binary trees of threshold gate adders)

Table E1.4.1. Continued.

Author(s)	Result(s)	Remark(s)
Goldmann and Karpinski (1994)	$LT_d \subset MAJ_{d+1}$ (improves on Siu and Bruck 1992 and implies: $NN^k_\Delta \subset \left\{ \begin{array}{l} LT_{\log^{k+\varepsilon} n} \subset MAJ_{1+\log^{k+\varepsilon} n} \\ \widehat{LT}_{\log^{k+1} n} \end{array} \right)$	Existence proof. An important aspect is that such a simulation is possible even if the depth d grows with the number of variables n
Beiu and Taylor (1996b)	$NN^k_\Delta \subset \left\{ \begin{array}{ll} NC^{k+1} & \\ MAJ_{\log^{k+1} n} & \subset \widehat{LT}_{\log^{k+1} n} \end{array} \right.$	Constructive proofs (based on carry save addition)

- Boolean decomposition can be used for reducing the fan-in, but at the expense of a superpolynomial increase in size ($(n^{\log n})^{1/2}$) and a double logarithmic increase in depth ($\log^2 n$).

Much better results can be achieved for a particular function.

Due to such scaling problems, theoretical results show that we can implement (as digital chips or wafers) only neural networks having (sub) logarithmic accuracy and (sub) linear fan-in (with respect to the number of inputs n). From the practical point of view (the two parameters being dependent) these should be translated to (sub) logarithmic both for accuracy and for fan-in. The main conclusion is that full parallel digital implementations of neural networks (as chips or wafers) are presently limited to artificial neural networks having 10^2–10^3 inputs and about 10^3–10^4 neurons of 10^2–10^3 inputs each. As will be seen later, these values are in good accordance with those from chips and wafers which stick as much as possible to a parallel implementation. Although we do expect that technological advances will push these limits, they cannot be spectacular—at least in the near future.

Such drastic limitations have forced designers to approach the problem from different angles:

- By using time-multiplexing
- By building arrays of (dedicated) chips working together and exploiting as much as possible (in one way or another) the architectural concept of pipe-lining
- By using non-conventional techniques such as: stochastic processing (Gorse and Taylor 1989a, Köllmann *et al* 1996), sparse memory architecture (Aihara *et al* 1996) or spike processing (Jahnke *et al* 1996).

These allow the simulation of far larger neural networks, by mapping them onto the existent (limited) hardware.

E1.4.3 Digital VLSI

Digital neurochips (and, thus, neurocomputers) benefit from the legacy of the most advanced human technology—digital information processing. VLSI technology is the main support for electronic implementations. It has been mature for many years, and allows a large number of processing elements to be mapped onto a small silicon area. That is why it has attracted many researchers (Alla *et al* 1990, Alspector *et al* 1988, Barhen *et al* 1992, Beiu 1989, Beiu and Rosu 1985, Boser *et al* 1992, Del Corso *et al* 1989, Disante *et al* 1989, 1990a, b, Faggin 1991, Fornaciari *et al* 1991b, Holler 1991, Jackel 1992, Mackie *et al* 1988, Personnaz *et al* 1989, Tewksbury and Hornak 1989, Treleaven *et al* 1989, Weinfeld 1990).

The main constraints of VLSI come from the fact that the designer has to implement the processing elements on a two-dimensional limited area and—even more—connect these elements by means of a limited number of available layers. This leads to limited interconnectivity as has been discussed in Akers *et al* (1988), Baker and Hammerstrom (1988), Hammerstrom (1988), Reyneri and Filipi (1991), Szedegy (1989) and Walker *et al* (1989) and limited precision (higher precision requires larger area—both due to storing and processing—leading to fewer neurons per chip (Dembo *et al* 1990, Denker and Wittner 1988, Myhill and Kautz 1961, Obradovic and Parberry 1990, Stevenson *et al* 1990, Walker and Akers 1992)). The shallowness of slow biological neural networks has to be traded off for (somehow) deeper

networks made of higher speed elements. Beside these, the power dissipation might impose another severe restriction (especially for wafer scale integration—WSI). The tradeoff is either to reduce the number of neurons per chip (working at high speed) or reduce the clock rate (while having more neurons). Lastly, the number of available pins to get the information on and off the chip is another strong limitation.

From the biological point of view, synapses have to be restricted on precision and range to some small number of levels (Baum 1988a, Baum and Haussler 1989). Lower bounds on the size of the network have been obtained both for the networks with real valued synaptic weights and for the networks where the weights are limited to a finite number of possible values (Siu and Bruck 1990a). These bounds differ only by a logarithmic factor, but to achieve near optimal performance $O(m)$ levels are required (Baum 1988b)—where m is the number of training examples given. A similar logarithmic factor has been proven in Hong (1987), Raghavan (1988) and Sontag (1990) when replacing real weights by integers. Some results concerning the needed number of quantization levels have already been presented in Section E1.2.2 and can be supplemented by many references. For example, Baker and Hammerstrom (1988), Hollis *et al* (1990), Höhfeld (1990), Shoemaker *et al* (1990), Allipi (1991), Asanović and Morgan (1991), Holt and Hwang (1991, 1993), Nigri (1991) and Xie and Jabri (1991) argue that the execution phase needs roughly 8 bits (6 . . . 10), while learning demands about 16 bits (14 . . . 18). There are few exceptions: Halgamuge *et al* (1991) being the only pessimistic one claiming that 32 bits are needed, and Reyneri and Filipi (1991) claiming that 20 . . . 22 bits are needed *in general*, but explicitly mentioning that this value can be reduced to 14 . . . 15 bits or even lower by properly choosing the learning rate (for backpropagation). New weight discretization learning algorithms can go much lower: to just several bits (see Section E1.2.4). This makes them ideal candidates for digital implementations.

Today, the digital VLSI design is still the most important design style. The advantages of the dominant CMOS technology are small feature sizes, lower power consumption and a high signal-to-noise ratio. For neural networks these are supplemented by the following advantages of digital VLSI design styles (see Glesner and Pöchmüller 1994 and Hammerstrom 1995 for more details):

- Simplicity (an important feature for the designer)
- High signal-to-noise ratio (one of the most important advantages over analog designs)
- Circuits are easily cascadable (as compared to analog designs)
- Higher flexibility (digital circuits in general can solve many tasks)
- Reduced fabrication price (certainly of interest for customers)
- Many CAD (computer aided design) systems are available to support a designer's work
- Reliable (as fabrication lines are stable).

Digital VLSI implementations of a neural network are based on several building blocks:

- Summation can easily be realized by adders (many different designs are possible and well-known: combinatorial, serial, dynamic, carry look ahead, manchester, carry select, Wallace tree)
- Multiplication is usually the most area-consuming operation and in many cases a multiplier is time-multiplexed (classical solutions are serial, serial/parallel and fully parallel, each of which differ in speed, accuracy and area)
- Nonlinear transfer function (very different nonlinear activation functions (Das Gupta and Schnitger 1993) can be implemented by using circuits for full calculations, but most digital designs use either a small lookup table (Nigri 1991, Nigri *et al* 1991) or—for even lower area and higher precision—a dedicated circuit for a properly quantized approximation, as can be seen in table E1.4.2 and also in Murtagh and Tsoi (1992), Sammut and Jones (1991)
- Storage elements (are very common—either static or dynamic—from standard RAM cells)
- Random number generators (are normally realized by shift registers with feedback via XOR-gates).

E1.4.4 Different implementations

E1.4.4.1 General comments

As the different number of proposed architectures or fabricated chips, boards and dedicated computers reported in the literature is on the order of hundreds, we cannot mention all of them here. Instead, we shall try to cover important types of architectures by several representation implementations—although certain readers could disagree sometimes with our choice. For a deeper insight the reader is referred to the following books: Eckmiller and von der Malsburg (1988), Eckmiller *et al* (1990), Souček and Souček

Table E1.4.2. Digital implementations of the sigmoid—alternatives to lookup tables.

Author(s)	Result(s)	Remark(s)
Myers and Hutchinson (1989)	Approximation of an A-law sigmoid-like function.	7 segments piecewise. Error $\leq \pm 4.89\%$ for $[-8, 8]$.
Alippi *et al* (1990a)	Approximations of a classical sigmoid function by sum of 1–5 steps.	Error $\leq \pm 13.1\%$ with 5 steps. Four comparators and several logic gates.
Alippi *et al* (1990b)	Approximations of a classical sigmoid function.	Sum of 1–5 steps (Alippi *et al* 1990a).
Pesulima *et al* (1990)	Approximation of the classical sigmoid by two exponentials.	Digital implementation: LFSR. Error $\leq \pm 2.45\%$ for $[-8, 8]$.
Saucier and Ouali (1990)	Silicon compilation.	Approximation by Taylor series.
Alippi *et al* (1991a)	Relations between convergence of learning and precision.	Introduces a general class of nonlinear functions.
Alippi and Storti-Gajani (1991)	Approximations of a classical sigmoid function.	Piecewise by the set of points $(\pm n, 1/2^{n+1})$.
Höhfeld and Fahlman (1991)	Probabilistic weight updates (down to 4 bits).	Needed precision for sigmoid 4–6 bits.
Krikelis (1991)	Approximation of the classical sigmoid function.	Piecewise linearization with 3 segments in $[-4, 4]$. Errors $\leq \pm 5.07\%$.
Nigri (1991)	Precision required for backpropagation.	Look-up table for 8 bits; 'exact' only for $[-2, 2]$.
Siggelkow *et al* (1991)	Analyzes *accuracy* and shows that a problem-dependent synthesis is required.	Piecewise linearization of the sigmoid with 5 segments. No hardware suggested.
Spaanenburg *et al* (1991)	Bit-serial approximation of binary logarithmic computations (*problem dependent complex parameters*).	Piecewise linearization of the sigmoid with 5 segments (4–6 bits). Errors around $\pm 10\%$. No hardware suggested.
Beiu (1992)	Sum of steps approximation of a particular sigmoid.	Six 'threshold gates' solution with weights $\{-1, 1, 2\}$. Error $\leq \pm 8.1\%$.
Deville (1993)	General method for piecewise linearization. Highest precision ($\leq \pm 1.14\%$).	*Requires 10 floating-point numbers and 5 multiplications!*
Beiu *et al* (1993, 1994b)	Piecewise approximation of the classical sigmoid.	Errors $\leq \pm 1.9\%$ using only a shift register and several logic gates.

(1988), Sami (1990), Zornetzer *et al* (1990), Antognetti and Milutinovic (1991), Ramacher and Rückert (1991), Sanchez-Sinencio and Lau (1992), Hassoun (1993), Przytula and Prasama (1993), Delgado-Frias and Moore (1994) and Glesner and Pöchmüller (1994) together with the references therein. Several overview articles or chapters can also be recommended: Alspector and Allen (1987), Mackie *et al* (1988), Jackel *et al* (1987), Jackel (1991), Przytula (1988), DARPA (1989), Del Corso *et al* (1989), Denker (1986), Goser *et al* (1989), Personnaz and Dreyfus (1989), Treleaven (1989), Treleaven *et al* (1989), Schwartz (1990), Burr (1991, 1992), Nordström and Svensson (1991), Graf *et al* (1991—having many references, 1993), Hirai (1991), Holler (1991), Ienne (1993a, b), Lindsey and Lindblad (1994) and the recent ones—Heemskerk (1995), Hammerstrom (1995) and Morgan (1995). The proceedings of MicroNeuro (International Conference on Microelectronics for Neural Networks) would also prove useful for those readers wishing to find latest details on different implementations or the most recent proposals. Many other conferences on neural networks have special sessions on hardware implementations: NIPS (Neural Information Processing Systems), IJCNN (International Joint Conference on Neural Networks), ICANN (International Conference on Artificial Neural Networks), WCNN (World Congress on Neural Networks), IEEE ICNN (IEEE International Conference on Neural Networks) just to mention some of the most widely known.

One of the difficult problems when discussing dedicated architectures for artificial neural networks is how to classify them. There are many different ways of classifying such architectures, and we shall mention here some which have already been presented and used in the literature.

- A first classification can be made based on the division of computer architectures due to Flynn (1972): single instruction stream, single datastream (SISD); single instruction stream, multiple datastreams (SIMD); multiple instruction streams, single datastream (MISD)—which does not make too much sense; multiple instruction streams, multiple datastreams (MIMD). Most of the architectures proposed for implementing neural networks belong to the SIMD class, and thus the group should be further subdivided into: systolic arrays, processor arrays (linear, mesh, multidimensional) and even pipelined vector processors.
- Another classification has been based on 'how many and how complex' processing elements are (Nordström and Svensson 1991). Computer architectures can be characterized by the level of parallelism which can be: moderately parallel (16 to 256 processors), highly parallel (256 to 4096 processors) or massively parallel (more than 4096 processors). As a coarse measure of the 'complexity' of the processing elements, the bit-length (i.e., the precision) of a processing element has been used.
- A much more simple classification of neurocomputers has been suggested by Heemskerk (1995): those consisting of a conventional computer and an accelerator board; those built from general purpose processors; and those built from dedicated neurochips.
- A completely different classification was suggested by Glesner and Pöchmüller (1994) based on the following three criteria: biological evidence (mimicking biological systems; mimicking on a higher level; or without biological evidence), mapping onto hardware (network-oriented; neuron-oriented; or synapse-oriented) and implementation technology (digital; analog; or mixed).

Only for digital electronic implementations a simple three-class subclassification scheme—somehow similar to that of Heemskerk (1995)—could be the following (Beiu 1994).

- *Dedicated digital neural network chips* (Kung 1989, Kung and Hwang 1988, 1989a), Wawrzynek *et al* (1993) can reach fantastic speeds of up to 1G connections per second. Several examples of such chips are: L-Neuro from Philips (Duranton 1996, Duranton *et al* 1988, Duranton and Maudit 1989, Duranton and Sirat 1989, 1990), X1 and N64000 of Adaptive Solutions (Adaptive Solutions 1991, 1992, Hammerstrom 1990), Ni1000 from Intel (Scofield and Reilly 1991, Holler *et al* 1992), MA16 from Siemens (Ramacher 1990, 1992, Ramacher and Rückert 1991, Ramacher *et al* 1991a, b, 1993), p-RAM from King's College London (Clarkson and Ng 1993, Clarkson *et al* 1989–1993) and Hitachi's WSI (Yasunaga *et al* 1989, 1990) and the 1.5-V chip (Watanabe *et al* 1993), SMA from NTT (Aihara *et al* 1996), NESPINN from the Institute of Microelectronics, Technical University of Berlin (Jahnke *et al* 1996), or SPERT from the International Computer Science Institute, Berkeley (Asanović *et al* 1992, 1993d, Warwzynek 1993, 1996).
- *Special purpose digital coprocessors* (sometimes called neuroaccelerators) are special boards that can be connected to a host computer (PCs and/or workstations) and are used in combination with a neurosimulator program. Such a solution tries to take both advantages: accelerated speed and flexible and user-friendly environment. Well-known are the delta Floating Point Processor from SAIC (DARPA 1989) which can be connected to a PC host, and the ones produced by Hecht-Nielsen Computers (Hecht-Nielsen 1991): ANZA, Balboa. Their speed is in the order of 10M connections per second improving tenfold on a software simulator. Some of them are using conventional RISC microprocessors, some use DSPs or transputers, while others are built with dedicated neurochips.
- *Digital neurocomputers* can be considered the massively data-parallel computers. Several neurocomputers are: WARP (Arnould 1985, Kung and Webb 1985, Annaratone *et al* 1987), CM (Means and Hammerstrom 1991), RAP (Morgan *et al* 1990, Beck 1990), SANDY (Kato *et al* 1990), MUSIC (Gunzinger *et al* 1992, Müller *et al* 1995), MIND (Gamrat *et al* 1991), SNAP (Hecht-Nielsen 1991, Means and Lisenbee 1991), GF-11 (Witbrock and Zagha 1990, Jackson and Hammerstrom 1991), Toshiba (Hirai 1991), MANTRA (Lehmann and Blayo 1991, Lehmann *et al* 1993), SYNAPSE (Ramacher 1992, Ramacher *et al* 1991a, b, 1993, Johnson 1993a), HANNIBAL (Myers *et al* 1993), BACCHUS and PAN IV (Huch *et al* 1990, Pöchmüller and Glesner 1991, Palm and Palm 1991), PANNE (Milosavlevich *et al* 1996), 128 PE RISC (Hiraiwa *et al* 1990), RM-nc256 (Erdogan and Wahab 1992), CNAPS (Adaptive Solutions 1991, 1992, Hammerstrom 1990), Hitachi WSI (Boyd

1990, Yasunaga *et al* 1989–1991), MasPar MP-1 (Grajski *et al* 1990, MasPar 1990a–c, Nickolls 1990), and CNS-1 (Asanović *et al* 1993a)—just to mention only the most well-known.

But even such a subclassification is not very clear cut, as in too many cases there are no borders. For example, many neurocomputers have been assembled based on identical boards built with custom designed neurochips: SNAP uses the HNC 100 NAP chip; MANTRA uses the GENES IV and the GACD1 chips; HANNIBAL uses the HANNIBAL chip; SYNAPSE uses the MA 16 chip; MasPar MP-1 uses the MP-1 chip; CNAPS uses the X1 or the N64000 chip; CNS-1 will use the Torrent and Hydrant chips. That is why we have decided in this section to use a more detailed classification which starts with the first historical neurocomputers and continues through acceleration boards, slice architectures, arrays of DSPs (digital signal processors), arrays of transputers, arrays of RISC processors, SIMD and systolic arrays built of dedicated processing elements and continuing with several other alternatives and ending with some of the latest implementations.

Beside classification and classification criteria, another problem when dealing with neurocomputers and neurochips is their performance evaluation. While the performance of a conventional computer is usually measured by its speed and memory, for neural networks 'measuring the computing performance requires new tools from information theory and computational complexity' (Abu-Mostafa 1989). Although the different solutions presented here will be assessed for size, speed, flexibility and cascadability, great care should be taken especially when considering speed. Hardware approaches are very different, thus making it almost impossible to run the same benchmark on all systems. Even for machines which support backpropagation (which is commonly used as a benchmark), the average number of weight updates per second or CUPS (connection updates per second) reported in publications shows different computational power—even for the same machine! This is due to: different precision of weights; the use of fixed point representation in some cases and the size of the network to be simulated (larger networks may be implemented more efficiently). A typical example of two different *backpropagation* implementations on WARP can be found in Pomerleau *et al* (1988). For architectures which do not support learning, the number of synaptic multiplications per second or CPS (connections per second) will be mentioned, but the same caution should be taken due to different word lengths (precision of computation) and network architectures. Normalizing the CPS value by the number of weights leads to CPS per weight or CPSPW, and was suggested as a better way to indicate the processing power of a chip (Holler 1991). Precision can also be included in the processing performance by considering a connection primitive per second (CPPS) which is CPS multiplied by bits of precision and by bits for representing the inputs (van Keulan *et al* 1994). Another reason for taking such speed measurements with a lot of care is that some of the articles report only on a small test chip (and the results reported are extrapolations to a future full-scale chip or to a board of chips and/or neurocomputer), or that only peak values are given.

C1.2.3

Finally, for neurochips and neurocomputers which are dedicated to a certain neural architecture (e.g., the *Boltzmann machine* (Murray *et al* 1992, 1994); Kohonen's *self-organizing feature maps* (Hochet *et al* 1991, Goser *et al* 1989, Rüping and Rückert 1996, Tryba *et al* 1990, Thiran 1993, Thiran *et al* 1994, Thole *et al* 1993); *Hopfield networks* (Blayo and Hurat 1989, Gascuel *et al* 1992, Graf and de Vegvar 1987a, b, Graf *et al* 1987, Savran and Morgül 1991, Sivilotti *et al* 1986, Weinfeld 1989, Yasunaga *et al* 1989, 1990); *Neocognitron* (Trotin and Darbel 1993, White and Elmasry 1992); *radial basis functions* and *restricted coulomb energy* (LeBouquin 1994, Scofield and Reilly 1991)), or for those which are built as *stochastic devices* (Clarkson and Ng 1993, Clarkson *et al* 1993a, b, Köllmann *et al* 1966), it is almost impossible to assess their speed. It should be mentioned that due to such unsurmountable problems there is usually little if any information on benchmarks.

C1.4, C2.1.1
C1.3.4
C2.1.3, C1.6.2
C1.6.3.1
C1.4

E1.4.4.2 Typical and recent examples

We shall firstly mention Mark III and IV from a historical point of view.

- *Mark III* was built at TRW, Inc., during 1984 and 1985 Hecht-Nielsen (1989). The design used eight Motorola M68010-based boards running at 12 MHz, with 512 kbytes of DRAM memory each. The software environment used was called ANSE (Artificial Neural Systems Environment). The original Mark III had a capacity of approximately 8000 processing elements (neurons) and 480 000 connections, and had a speed of 380 000 CPS (large instar network using Grossberg learning).
- *Mark IV* was also built at TRW, Inc., but under funding from the Defense Science Office of the Defense Advanced Research Projects Agency (DARPA). A detailed description is given by Hecht-Nielsen (1989) who, together with Todd Gutschow, was one of the designers. It was capable of

implementing as many as 262 144 processing elements and 5.5 M connections, and had a sustained speed of 5 MCPS, whether or not learning was taking place Kuczewsk *et al* (1988). It had a mass of 200 kg and drew 1.3 kW of power. The basic computing unit was a 16-bit Texas Instruments TMS32020 DSP. The idea was that Mark IV would be a node of a larger neurocomputer (which was never intended to be constructed).

In the meantime most of the neural network simulations have been performed on sequential computers. The performance of such software simulation was roughly between 25 000 and 250 000 CPS in 1989 (DARPA 1989). Fresh results show impressive improvements on computers having just one processing element.

- *IBM 80486/50MHz* exhibits 1.1 MCPS and 0.47 MCUPS (Müller *et al* 1995).
- *Sun* (Sparcstation 10) has 3.0 MCPS and 1.1 MCUPS Müller *et al* (1995).
- *NEC SX-3* (supercomputer) achieves 130 MCUPS (the implementation was presented by Koike from NEC at the Second ERH-NEC Joint Workshop on Supercomputing 1992 Zürich, but no published English reference seems to be available). As NEC SX-3 has 5.9 Gflops it is expected that a similar performance would be obtained on a Cray Y-MP/8 (which has 2.5 Gflops).

Similar results have been reported for Hypercube FPS 20 (Roberts and Wang 1989, Neibur and Brettle 1992) and CM (Deprit 1989, Zhang *et al* 1990). At least one order of magnitude increase can be expected on Fujitsu, Intel Paragon or on the NEC SX-4.

As a first alternative and aimed at increasing the speed of simulations on PCs and workstations, *special acceleration boards* have been developed Williams and Panayotopoulos (1989).

- *Delta Floating Point Processor* from the Science Application International Corporation (SAIC), has separate addition and multiplication parts; it runs at 10 MCPS and 1–2 MCUPS (Souček and Souček 1988, Works 1988).
- SAIC later developed *SIGMA-1* which has a 3.1 M virtual interconnections and has reached 11 MCUPS (Treleaven 1989).
- *ANZA Plus* from Hecht-Nielsen Computers (Hecht-Nielsen 1988) has a 4-stage pipelines Harvard architecture. It can go up to 1 M virtual processing elements, 1.8 MCUPS (Atlas and Suzuki 1989) and 6 MCPS (Treleaven 1989).
- Intel i860 RISC processor is used in the *Myriad MC860* board and in the *Balboa* board from HNC, showing around 7 MCUPS (Hecht-Nielsen 1991).

Many other accelerator boards are mentioned in a tabular form by Lindsey and Lindblad (1994).

One simple way to increase performance even more is to use processors in parallel. A classical design style was used for *slice architectures*, and several representative models are detailed.

- Micro Devices have introduced the NBS (Neural Bit Slice) chip MD1220 (Micro Devices 1989a–c, 1990). The chip has eight processing elements with hard-limit thresholds and eight inputs (Yestrebsky *et al* 1989). The architecture is suited for multiplication of a 1-bit synapse input with a 16-bit weight. The chip only allows for hard-limiting threshold functions. The weights are stored in standard RAM, but only eight external weights per neuron and seven internal weights per neuron are supported. Such a reduced fan-in (maximum 15 synapses per neuron) is quite a drastic limitation. This can be avoided by additional external circuits, but increasing the fan-in decreases the accuracy (as the 16-bit accumulator can overflow). The chip has a processing rate of 55 MIPS which roughly would correspond to 8.9 MCPS.
- A similar chip is the *Neuralogix NLX-420* Neural Processor Slice from Neuralogix (1992), which has 16 processing elements. A common 16-bit input is multiplied by a weight in each processing element in parallel. New weights are read from off-chip. The 16-bit weights and inputs can be user selected as 16 1-bit, 4 4-bit, 2 8-bit or 1 16-bit value(s). The 16 neuron sums are multiplexed through a user-defined piecewise continuous threshold function to produce a 16-bit output. Internal feedback allows for multilayer networks.
- The Philips *L-Neuro 1.0* chip (Duranton and Maudit 1989, Duranton and Sirat 1989, Theeten *et al* 1990, Maudit *et al* 1992) was designed to be easily interfaced to transputers. It also has a 16-bit processing architecture in which the neuron values can be interpreted as 8 2-bit, 4 4-bit, 2 8-bit or 1 16-bit value(s). Unlike the NLX-420, there is a 1 kbyte on-chip cache to store the weights. The chip has 32 inputs and 16 output neurons and only the loop on the input neurons is parallelized (weight parallelism). This chip has on-chip learning with an adjustable learning rate. The transfer function is

computed off-chip. This allows for multiple chips to provide synapse-input products to the neurons and, thus, to build very large networks. An experiment with 16 L-Neuro 1.0 (Maudit *et al* 1991) was able to simulate networks with more than 2000 neurons and reached 19 MCPS and 4.2 MCUPS. The work has been continued: a *L-Neuro 2.0* architecture was reported (Dejean and Caillaud 1994), followed recently (Duranton 1996) by *L-Neuro 2.3* (see the paragraph on the latest implementations).

- *BACCHUS* is another slice architecture which was designed at Darmstadt University of Technology. There have been three successive versions I, II, and III (Huch *et al* 1990, Pöchmüller and Glesner 1991). The neurons perform only a hard-limiting threshold function. The final version was designed as a sea-of-gates in 1.5-μm CMOS (Glesner *et al* 1989, Glesner and Pöchmüller 1991). The chip contains 32 neurons and runs at 32 MCPS (but for 1-bit interconnections!). An associative system PAN IV, based on BACCHUS III chips has been built (Palm and Palm 1991). It has eight BACCHUS III chips (for a total of 256 simple processors) and 2 Mbytes of standard RAM. The system was designed only as a binary correlation matrix memory.

For even higher performances the designers have used SIMD arrays (various one- or two-dimensional systolic architectures (Kung 1988, Kung and Hwang 1988, 1989a, 1989b, Kung and Webb 1985), made of DSPs (digital signal processors), RISC processors, transputers or dedicated chips.

Many neuroprocessors have been built as *arrays of DSPs*.

- One of the first array-processors proposed for neural network simulation was built at IBM Palo Alto Scientific Center (Cruz *et al* 1987). The building block was the *NEP* (Network Emulation Processor) board able to simulate 4000 nodes (neurons) with 16 000 links (weights) and a speed of between 48 000 and 80 000 CUPS. Up to 256 NEPs could be cascaded (through a NEPBUS communication network), thus allowing for networks of 1 million nodes and 4 million links.
- Another DSP neuroprocessor called *SANDY* emerged from Fujitsu Laboratories (Kato *et al* 1990). The DSP used was the Texas Instruments TMS320C30 connected in a SIMD array. SANDY/6 (with 64 processors) was benchmarked on NETtalk (Sejnowski and Rosenburg 1986) at 118 MCUPS and 141 MCPS. SANDY/8 with 256 processors was expected to work at 583 MCUPS (Yoshizawa *et al* 1991).
- The *RAP* (Ring Array Processor) developed at the International Computer Science Institute (ICSI, Berkeley) is an array of between 4 and 40 Texas Instruments TMS320C30 DSPs containing 256 kbytes of fast static RAM and 4 Mbytes of dynamic RAM each (Morgan *et al* 1990, 1992, 1993, Kohn *et al* 1992). These chips are connected via a ring of Xilinx programmable gate arrays, each implementing a simple two register data pipeline and running at the DSP clock speed of 16 MHz. A single board can perform 57 MCPS and 13.2 MUCPS, with a peak performance for a whole system reaching 640 MCPS (tested at 570 MCPS) and 106 MCUPS.
- At the Swiss Federal Institute of Technology in Zürich, a 63-processor system named *MUSIC* (Multiprocessor System with Intelligent Communication) has been developed (Müller *et al* 1992, 1994, 1995). The architecture is similar to that of RAP but differs in the communication interface. Three Motorola 96002 DSPs (32-bit floating-point) are mounted on one board, each one with a Xilinx LCA XC3090 programmable gate array and an Inmos T805 transputer. Up to 21 boards (i.e., 63 processors) fit into a standard 19-inch rack. A global 5-MHz ring connects the nodes and communication can be overlapped with computation. The complete system has achieved 817 MCPS and 330 MCUPS (for a 5000-1575-63 two-layer perceptron), but the peak performance is 1900 MCPS. A fully equipped system consumes 800 W.
- *PANNE* (Parallel Artificial Neural Network Engine) has been designed at the University of Sydney (Milosavlevich *et al* 1996) and exploits the many specialized features of the TMS320C40 DSP chip. One board contains two DSPs together with 32 Mbytes of DRAM and 2 Mbytes of high speed SRAM. These are accessed through a dedicated local bus. Apart from this local bus, each board has a global bus and six programmable unidirectional 8-bit ports specially designed to allow connections of neighboring DSPs at 20 Mbytes per second. The system has up to eight boards and is estimated at 80 MCUPS.

Different solutions have been implemented on *arrays (networks) of transputers* (Ernst *et al* 1990, Murre 1993). Ernoult (1988) reported that a network of 2048 neurons with 921 600 connections running on a 16-transputer system (T800) has reached 0.57 MCUPS. A Megaframe *Hypercluster* from Parsytec (Achen, Germany), having 64 transputers (T800) and implementing backpropagation, runs at 27 MCPS

and 9.9 MCUPS (Mühlbein and Wolf 1989). This performance should increase tenfold on the Parsytec's *Gigacluster* which uses T9000 transputers.

Instead of transputers some researchers have used RISC processors and here are some of the neurocomputers built as *arrays of RISC processors*.

- One solution was to design a RISC processor (dedicated for simulating neural networks) and assembling several of them in SIMD arrays. Here we can mention the 16-bit *Neural RISC* developed at University College London (Pacheco and Treleaven 1989, Treleaven *et al* 1989, Treleaven and Rocha 1990). Several neural RISCs have been connected in a linear array. A linear array interconnecting scheme has several advantages: simplified wiring and ease of cascadability. Several arrays are linked by an interconnecting module (Pacheco and Treleaven 1992). This allows for different topologies (rings, meshes, cubes) and is expandable up to a maximum of 65 536 processors. The flexibility is high as the computer is of the MIMD type (multiple instructions multiple data).
- *REMAP*^{3} was an experimental neurocomputing project (Bengtsson *et al* 1993, Linde *et al* 1992) with its objective being to develop a parallel reconfigurable SIMD computer using FPGAs. The performance was estimated to be between 100 and 1000 MCUPS.
- Another solution is to use a standard RISC processor. An example is the *128 PE RISC* which uses the Intel 80860 (Hiraiwa *et al* 1990). 128 processors are connected in a two level pipeline array where the horizontal mesh connections serve for information exchange (weights) and vertical meshes share dataflow. For a 256-80-32 network and 5120 training set vectors, the performance is around 1000 MCUPS.
- *BSP400* from Brain Style Processor (Heemskerk *et al* 1991, Heemskerk 1995) used low-cost commercial microprocessors MC68701 (8-bit microprocessor). Due to the low speed of the processor used (1 MHz!) the overall performance reached only 6.4 MCUPS when 400 processors were used.

Because both DSP and RISC processors are too powerful and flexible for the task of simulating neural networks, a better alternative is to use smaller and more specific (less flexible) dedicated processing elements. This can increase the computational power and also maintain a very small cost. The trend has been marked by the use of *SIMD arrays* (Single Instruction Multiple Data) and especially *systolic arrays* (Kung and Hwang 1988) of dedicated chips. Systolic arrays are a class of architecture where the processing elements and the interconnecting scheme can be optimized for solving certain classes of algorithms. Matrix multiplication belongs to this class of algorithms (Leiserson 1982), and it is known that neural network simulation relies heavily on matrix multiplication (Beiu 1989, Kham and Ling 1991, Kung and Hwang 1989b). The SIMD arrays are similar structures, the main difference being that the elementary processing elements have no controllers and that a central controller is in charge of supervising the activity of all the elementary processing elements.

- The *WARP* array was probably the earliest systolic one (Kung and Webb 1985, Arnould 1985, Annaratone *et al* 1987). Although built primarily for image processing, it has also been used for neural network simulation (Pomerleau *et al* 1988). It is a ten (or more) processor programmable systolic array. The system can work either in a systolic mode, or in a local mode (each processor works independently). A performance of 17 MCUPS was obtained on a 10-processor WARP.
- *ARIANE* chip (Gascuel *et al* 1992) is a 64-neuron implementation in a 1.2-μm CMOS of the architecture first proposed by Weinfeld (1989). The chip—having 420 000 transistors in 1 cm^2—implements a fully digital Hopfield-type network, thus continuing on the lines of other Hopfield-type implementations (Sivilotti 1986, Graf *et al* 1986). All operations are performed by a 12-bit adder/subtracter. There are 64 connections per neuron, making it possible to store 4096 weights. The reported speed is 640 MCUPS, but this figure cannot be compared to standard CUPS as the chip does not implement backpropagation. The main drawback is that the chip is not easily cascadable (however, a four chip board has been designed).
- *SNAP* (SIMD Neurocomputer Array Processor) from Hecht-Nielsen Computers, Inc is based on HNC 100 NAP chips (Neural Array Processor). The chip is a one-dimensional systolic array of four arithmetic cells forming a ring (Hecht-Nielsen 1991, Means and Lisenbee 1991) and implementing IEEE 32-bit floating-point arithmetic. Each arithmetic cell contains a 32-bit floating point multiplier, floating point ALU and integer ALU, and runs at 20 MHz with all instructions being executed in one clock cycle. Four NAPs are linked on one SNAP board. SNAP has either 32 (SNAP-32) or 64 (SNAP-64) processors (i.e., either two or four boards). The SNAP-32 performed at 500 MCPS (peak

performance being 640 MCPS) and 128 MCUPS. Although the system performs lower than CNAPS (described below), we have to mention that SNAP uses 32-bit floating point arithmetic.

- The *APLYSIE* chip is a two-dimensional systolic array dedicated for Hopfield-type networks (Blayo and Hurat 1989). Since the outputs are only $+1$ and -1, the synaptic multiplication can be performed by an adder/subtracter (like in Weinfeld's 1989 solution). The weights are limited to 8-bit and the partial product is computed by a 16-bit register. The adder/subtracter is of the serial type for minimizing the area, but is also thought for the serial interconnecting scheme used. An advantage of such a solution is its cascadability.
- The *GENES* chip is a generalization of APLYSIE and it was implemented at the Swiss Federal Institute of Technology (Lausanne) as part of the *MANTRA* project (Lehmann and Blayo 1991, Lehmann *et al* 1993, Viredaz *et al* 1992). It is based on the same recurrent systolic array as APLYSIE, but it has been enhanced to simulate several neural network architectures. The first chip of the family was GENES HN8 implementing each synapse as a serial-parallel multiplier. Two versions have been fabricated: 2×2 array of processors and 4×4 array of processors. Weights and inputs are represented on 8 bits. The partial sum is calculated on 24 bits. A full board, GENES SY1, was built as a 9×8 array of GENES HN8 2×2 chips (18×16 synapses) and was able to reach 110 MCPS. A GENES IV chip was later designed as an upgrade of GENES HN8 (Lehmann *et al* 1993, Viredaz *et al* 1992). It has 16-bit inputs and synaptic weights and uses 39 bits for the partial sum. The chip was designed with standard cells in a 1-μm CMOS technology on a 6.2×6.2 mm^2 area. Together with another chip, GACD1 (dedicated to the error computation for delta rule and backpropagation), it was used to build the first MANTRA neurocomputer as a 40×40 array of processing elements. The speed is estimated at 500 MCPS and 160 MCUPS.
- A low-cost high-speed neurocomputer system has recently been proposed (Strey *et al* 1995) and implemented (Avellana *et al* 1996). The system is based on a dedicated AU chip which has been designed so as to *dynamically adapt the internal parallelism to data precision*. It tends to achieve an optimal utilization of the available hardware resources. The AU chip is organized as a pipeline structure where the data path can be adapted dynamically to the encoding of the data values. The chip has been realized in 0.7 μm and has 80 mm^2. Four chips are installed on a board together with: a Motorola DSP96002 (used for the management of the local bus, computation of the sigmoid function, error calculation, winner calculation and convergence check); an FPGA for communication; local weight memories; central memory; and FIFO memory. Several boards can be used together. For 16-bit weights and with only one board the estimated performance is 480 MCPS and 120 MCUPS.
- *TNP* (Toroidal Neural Processor) is a linear systolic neural accelerator engine developed at Loughborough University of Technology (Jones and Sammut 1993, Jones *et al* 1990, 1991). The system is still under development although several prototype chips have been successfully fabricated and tested.
- *HANNIBAL* (Hardware Architecture for Neural Networks Implementing Backpropagation Algorithm Learning) was built at British Telecom. A dedicated HANNIBAL chip contains eight processing elements (Myers *et al* 1991, Orrey *et al* 1991, Naylor *et al* 1993), each one with 9216 bits of local memory (configurable as 512 17-bit words, or 1024 9-bit words). Such a chip allows for high fan-in neurons to be implemented; up to four lower fan-in neurons can be mapped onto one processing element. The neuron activation function is realized by a dedicated approximation for area saving reasons. The chip uses reduced word length (8-bit in the recall phase and 16-bit when learning (Vincent and Myers 1992) and it was fabricated in a 0.7-μm CMOS technology. This has led to 750 000 transistors in a 9×11.5 mm^2 area. The clock frequency is 20 MHz and a single chip can reach 160 MCPS.
- *MM32K* (Glover and Miller 1994) is a SIMD having 32 768 simple processors (bit serial). A custom chip contains 2048 processors. The bit serial architecture allows for the variation of the number of bits (variable precision). The processors are interconnected by a 64×64 full crossbar switch with 512 processors connected to each port of the switch.
- *SYNAPSE 1* and *SYNAPSE X* (Synthesis of Neural Algorithms on a Parallel Systolic Engine) from Siemens (Ramacher 1990, 1992, Ramacher *et al* 1991b, 1993) are dedicated to operation on matrices based on the MA16 chip (Beichter *et al* 1991), which has four systolic chains (of four multipliers and four adders each). The chip runs at 25 MHz and was fabricated in 1.0-μm CMOS. Its 610 000 transistors occupy 187 mm^2. The MA16 alone has 800 MCPS when working on 16-bit weights. SYNAPSE neurocomputer is nothing else but a two-dimensional systolic array of MA16 chips

arranged in two rows by four columns. The weights are stored off chip in local memories. Both processor rows are connected to the same weight bus which excludes the operation on different input patterns. The MA16s in a row form a systolic array where input data as well as intermediate results are propagated for obtaining the total weighted sum. Multiple standard 68040s and additional integer ALUs are used as general purpose processors which complement the systolic processor array. The standard configuration has eight MA16s, two MC68040 for control and 128 Mbytes of DRAM. It performs at 5100 MCPS and 133 MCUPS.

- *CNAPS* (Connected Network of Adaptive Processors) is a SIMD array from Adaptive Solutions, Inc (Adaptive Solutions 1991, 1992, Hammerstrom 1990). X1 is a neural network dedicated chip with on-chip learning. It consists of a linear array of elementary processors, each one having a 32-bit adder and a 24-bit multiplier (fixed-point). The structure of an elementary processor is such that it can work with three different weight lengths: 1-bit, 8-bit and 16-bit weights (Hammerstrom 1990, Hammerstom and Nguyen 1991). X1 chips are fully cascadable, allowing the construction of linear arrays having arbitrary many elementary processors. Another chip, the N64000, was produced in 0.8-μm CMOS and 80 elementary processors have been embedded in this design. N64000 is a large chip (one square inch) containing over 11 million transistors (Griffin *et al* 1991) and due to defects in the fabrication process only 64 functioning processing elements are used from one chip (the 16 more being redundant). The same idea will be used at a higher level for the Hitachi's WSI (wafer scale integration) to be discussed later. The maximum fan-in of one neuron is 4096 and there are 256K programmable synapses on the 26.2×27.5 mm^2 chip. The chip alone can perform 1600 MCPS and 256 MCUPS for 8- or 16-bit weights (12 800 MCPS for 1-bit weight). The CNAPS has four N64000 chips running at 20 MHz on one board (256 processing elements). The maximum performance of the system is quite impressive: 5700 MCPS and 1460 MCUPS (Adaptive Solutions 1991, 1992, McCator 1991), but these values are for 8- and 16-bit weights! Hammerstrom and Nguyen (1991) have also compared a Kohonen self-organizing map implemented on the CNAPS: 516 MCPS and 65 MCUPS, with the performance on a SPARC station: 0.11 MCPS and 0.08 MCUPS.
- *MasPar MP-1* is a SIMD computer based on the MP-1 chip (Blank 1990, MasPar 1990). It is a general purpose parallel computer but it exhibits excellent performances when simulating neural networks. The core chip is MP-1 which has 32 processing elements working on 32-bit floating point numbers (each processing element can be viewed as a small RISC processor). MP-1 was fabricated in 1.6-μm CMOS on an area of 11.6×9.5 mm^2 and has 450 000 transistors. The chip works at a moderate clock frequency of only 14 MHz for minimizing the dissipated power. One board uses 32 MP-1 chips, thus having 1024 processing elements which are arranged in a two-dimensional array. The connection scheme is different from others: 16 processing elements are configured as a 4×4 array with an X-net mesh and form a 'processor element cluster'. These clusters are again connected as an X-net mesh of clusters. The processors are connected together from the edges to form a torus. On top of that, a global communication between processing elements is realized by a dedicated 1024×1024 crossbar interconnecting network having three stages for routing. MasPar can have from 1 to 16 boards. The largest configuration has 16 384 processing elements. Grajski *et al* (1990) have simulated neural networks on a MasPar MP-1 with 4096 processing elements (MasPar MP-1 1100). A 900-20-17 backpropagation network obtained 306 MCUPS, but on the largest MasPar MP-1 1200 (16 384 processing elements) performance is expected to be on the order of GCUPS.

Many *other alternatives* have also been presented and we shall shortly enumerate some of them here.

- *WISARD* belongs to the family of weightless neural networks or the RAM model (Aleksander and Morton 1990) and has been used in image recognition. C1.5.4
- The *pRAM* (probabilistic RAM) is a nonlinear stochastic device (Gorse and Taylor 1989a, b, 1990a, 1991a, c) with neuron-like behavior which—as opposed to the simple RAM model—can implement nonlinear activation functions and can generalize after training (Clarkson *et al* 1993a). It is based on a pulse-coding technique and several chips have been fabricated. The latest digital pRAM has 256 neurons per chip. The 16-bit 'weights' (probabilities) are stored in an external RAM in order to keep the costs at a minimum. Up to 1280 neurons can be interconnected by combining five chips. Learning (Clarkson and Ng 1993, Clarkson *et al* 1991a, b, 1992b, 1993b, c, Gorse and Taylor 1990b, 1991b, Guan *et al* 1992) is performed on-chip. The pRAM uses a 1-μm CMOS gate-array with 39 000 gates. A PC board has been designed and tested. A VMEbus-based neural processor board (using the pRAM-256) has also been recently built (El-Mousa and Clarkson 1996). The current VMEbus C1.5.2

version is being used for studying the various different architectures and advantages of hardware-based learning using pRAM artificial neural networks. For this purpose, the board relies heavily on the use of in-system programmable logic devices (ISPLD) to facilitate changing the support hardware logic associated with the actual neural processor without the need to rewrite and/or exchange parts of it.

- Intel has several neural network solutions (Intel 1992a, b). Two commercial chips are dedicated to radial basis functions (Watkins *et al* 1992): the *IBMZISC036* (LeBouquin 1994) and *Ni1000* (Scofield and Reilly 1991) build in cooperation with Nestor. The *ZISC036* (from Zero Instruction Set Computer) contains 36 prototype neurons, where the vectors have 64 8-bit elements and can be assigned to categories from 1 to 16 384 (i.e., the first layer has 36 neurons fully connected by 8-bit weights to the 64 neurons of the second layer). Multiple ZISC036 chips can be easily cascaded to provide additional prototypes, while the distance norm is selectable between city-block (Manhattan) or the largest element difference. The ZISC036 implements a region of influence (ROI) learning algorithm (Verleysen and Cabestany 1994) using signum basis functions with radii of 0 to 16 383. Recall is either according to the ROI identification, or via the nearest-neighbor readout, and takes 4 μs for a 250-K sec-pattern presentation rate.

  The *Ni1000* was developed jointly by Intel and Nestor and contains 1024 prototypes of 256 5-bit elements (i.e., the first layer has 256 neurons, while the second layer is fully connected to the first layer by 5-bit weights and has 1024 neurons). The distance used is the city-block (Manhattan) distance. The third layer has 64 neurons working in a sequential way, but achieving higher precision. All the weights and the threshold are stored on board in a nonvolatile memory, as the chip is implemented in Intel's 0.8-μm EEPROM process. On the same chip a Harvard RISC is used to accelerate learning (Johnson 1993b), and increases the overall number of transistors to 3.7 million. The chip implements two on-chip learning algorithms: restricted coulomb energy or RCE (Reilly *et al* 1982) and probabilistic neural networks or PNN (Specht 1988). Other algorithms can be microcoded. In a pattern processing application the chip can process 40 000 patterns per second (Holler *et al* 1992).
- A *generic neural architecture* was proposed by Vellasco and Treleaven (1992). The idea is to tailor the hardware to the neural network to be simulated. This can increase the performance at the expense of reduced flexibility. The aim of such an approach is to automatically generate application-specific integrated circuits (ASICs). Several chips have been fabricated. Other authors have been working on similar approaches (Disante *et al* 1990b, Fornaciari *et al* 1991a, b), or have tried a mapping onto FPGAs (Beiu and Taylor 1995c, Botros and Abdul-Aziz 1994, Gick *et al* 1993, Nigri *et al* 1991, Nijhuis *et al* 1991, Rossmann *et al* 1996, Rückert *et al* 1991).
- Several implementations of the *Boltzmann machine* have also been reported. A high-speed digital one is that of Murray *et al* (1992, 1994). The chip, realized in a 1.2-μm CMOS technology, has 32 neural processors and four weight update processors supporting an arbitrary topology of up to 160 functional neurons. The 9.5×9.8 mm^2 area hosts 400 000 transistors. This includes the 20 480 5-bit weights stored in a dynamic RAM (the activation and temperature memories are static). Although clocked at 125 MHz, the chip dissipates less than 2 W. The theoretical maximum learning rate is 350 MCUPS and the recall rate is typically 1200 patterns per second. An SBus interface board was developed using several reconfigurable Xilinx FPGAs. C1.4
- *ArMenX* is a distributed computer architecture (Poulain Maubant *et al* 1996) articulated around a ring of FPGAs acting as routing resources as well as fine grain computing resources (Léonhard *et al* 1995). This allows for a high degree of flexibility. Coarse grain computing relies on transputers and DSPs. Each ArMenX node contains an FPGA (Xilinx 4010) tightly coupled to an Inmos T805 transputer and a Motorola DSP56002, but other processors could be used. The node has 4 Mbytes of transputer RAM and 384 Kbytes of DSP RAM and the FPGA connects to the left and right neighboring nodes. The sustained performance of a node is about 5 MCPS and 1.5 MCUPS, and it is expected that the scale-up will be linear for a 16-node machine: 80 MCPS and 24 MCUPS.
- A solution which uses *on-line arithmetic* has been proposed in Girau and Tisserand (1996) and should be implemented on an FPGA. A redundant number representation allows very fast arithmetic operations, the estimated speed being 5.2 MCUPS per chip.
- The use of *stochastic arithmetic* computing for all arithmetic operations of training and processing backpropagation networks has also been considered (Köllmann *et al* 1996). Arithmetic operations become quite simple. The main problem in this case is the generation of numerous independent random generators. The silicon reported uses a decentralized pseudorandom generator based on the

principle of shifting the turn-around code for parities formed on partial stages of a feedback shift register. A 3.5×2.8 mm^2 silicon prototype has been implemented in 1-μm CMOS technology. The prototype delivers a theoretical performance of 400 MCUPS for 12-bit weight length and 15-bit momentum length. It is estimated that a state-of-the-art 0.25-μm process would allow 4K synapses and 64 neurons should fit into 160 mm^2 if standard cells are used; a custom design should increase these values to: 16K synapses and 128 neurons.

Some of the latest implementations are pushing the performances even further and we shall mention here the most promising ones, even if by our classification some of them might also fall in another class.

- The *RM-nc* is a reconfigurable machine for massively parallel-pipelined computations and has been proposed in Erdogan and Wahab (1992). The reconfigurability is not only in the domain of communication and control, but also in the domain of processing elements. A fast floating point sum-of-products circuit using special carry-save multipliers (with built-in on-the-fly shifting capability and extensive pipelining) has been proposed and has to be implemented on FPGAs. The performance of an RM-nc256 machine (with 256 processing FPGAs) has been estimated for NETtalk (203-60-26 network with 13 826 connections) at a speed of 2000 MCUPS. No implementation has yet been reported.
- One interesting development is based on *WSI* (Mann *et al* 1987, Rudnik and Hammerstrom 1988, Tewksbury and Hornak 1989). A first neural network WSI has been developed by Hitachi (Yasunaga *et al* 1989, 1990). This first version was designed only for Hopfield networks without learning. *Hitachi's WSI* has 576 neurons with a fan-in of 64. Weights are represented on 10 bits. If larger fan-in is required, three neurons can be cascaded to increase the fan-in to 190 (this reduces the number of available neurons). A 'small' 5-inch wafer and a 0.8-μ CMOS technology has been used to realize the designed 19 million transistors. The wafer has 64 chips of 12 neurons each; one redundant chip (Zorat 1987) is used to replace faulty neurons from the other chips. Up to 37K synapses are available on chip. For controlling the neurons and the buses there are eight more chips on the wafer. The only way to keep the power to a reasonable 5 W is a quite-slow clock rate: 2.1 MHz, but the actual performance is still around 138 MCPS.

  The same idea has been used by Hitachi (Boyd 1990) to design a WSI for multilayer feedforward networks including the backpropagation algorithm. The weights' accuracy has been increased to 16 bits to cope with the required precision of on-chip learning. One wafer has 144 neurons and eight wafers have been stacked together to form a very small neurocomputer with 1152 neurons (Yasunaga *et al* 1991). The reported speed is 2300 MCUPS. Using a similar architecture and the present day state-of-the-art 0.3-μm CMOS technology it becomes clear that we can expect to have 10 000 neurons WSI in the very near future.
- For portable applications *Hitachi* has also developed a *1.5 V digital chip* with 1 048 576 synapses (Watanabe *et al* 1993). The chip can emulate 1024 fully connected neurons (fan-in of 1024 each) or three layers of 724 neurons. An on-chip DRAM cell array is used to store the 8-bit weights. A 256 parallel circuit for summing product (Baugh parallel multiplier) pushes the processing speed to 1370 MCPS. A scaled-down version of the chip was fabricated using a 0.5-μm CMOS design rule. It allowed an estimation of the full-scale chip: 15.4×18.6 mm^2 and 75 mW.
- The new *L-neuro 2.3* (Duranton 1996) is a fully programmable vectorial processor in a highly parallel chip composed of an array of twelve DSPs which can be used not only for neurocomputing, but also for fuzzy logic applications, real-time image processing and digital signal processing. Beside the twelve DSPs, the chip contains: a RISC processor, a vector-to-scalar unit, a 32-bit scalar unit, an image addressing module and several communication ports. All the DSPs are linked together: by a broadcast bus connecting all DSPs; by two shift chains linking the DSPs as a systolic ring; by fast neighbor-to-neighbor connections existing between adjacent DSPs; and also to an I/O port. All the internal buses are connected together through a programmable crossbar switch. The RISC processor of one chip can be used to control several other L-Neuro chips, allowing an expansion in a hierarchical fashion. The chip was fabricated in 0.6-μm technology and has 1.8 million transistors clocked at 60 MHz. It can implement different learning algorithms such as backpropagation, Kohonen features map, radial basis functions and neural trees (Sirat and Nadal 1990). The peak performance is estimated at 1380 MCUPS and 1925 MCPS but no tests have yet been reported.
- One very interesting approach is the novel *SMA* (Sparse Memory Architecture) neurochip (Aihara *et al* 1996) which uses specific models to reduce neuron calculations. SMA uses two key techniques

to achieve extremely high computational speed without an accuracy penalty: 'compressible synapse weight neuron calculation' and 'differential neuron operation'. The compressible synapse weight neuron calculation uses the transfer characteristics of the neuron to stop the calculation for the sum if it is determined that the final sum will be in the saturation region. This also cancels subsequent memory accesses for the synapse weights. The purpose of differential neuron operation is to do calculations only on those inputs whose level has changed. A dedicated processing unit having a 22-bit adder, a 16-bit shifter, an EX-OR gate and two 22-bit registers has been designed. A test chip having 96 processing units has been fabricated in 0.5-μm CMOS and has 16.5×16.7 mm^2. It runs at 30 MHz and dissipates 3.2 W. The chip can store 12 228 16-bit synapse weights and has a peak performance of 30 GCPS (tested at 18 GCPS).

- *SPERT* (from Synthetic Perceptron Testbed) (Asanović *et al* 1992, 1993d, Wawrzynek *et al* 1993, 1996) is a fully programmable single chip neuromicroprocessor which borrowed heavily from the experience gained with RAP (Morgan *et al* 1990, 1992, 1993). It combines a general purpose integer data path with a vector unit of SIMD arrays optimized for neural network computations and with a wide connection to external memory through a single 128 VLIW instruction format. The chip is implemented in 1.2-μm CMOS and runs at 50 MHz. It has been estimated at a peak performance of 350 MCPS and 90 MCUPS. The chip is intended to be a test chip for the future Torrent chip: the basic building block of CNS-1 (see below). Recent developments have led to SPERT-II (Wawrzynek *et al* 1996) which has a vector instruction set architecture (ISA) based on the industry standard MIPS RISR scalar ISA.
- *NESPINN* (Neurocomputer for Spike-Processing Neural Networks) is a mixed SIMD/dataflow neurocomputer (Roth *et al* 1995, Jahnke *et al* 1996). It will allow the simulation of up to 512K neurons with up to 10^4 connections each. NESPINN consists of the spike-event list (the connectivity of sparsely connected networks is performed by the use of lists), two connectivity units containing the network topology (a regular and a nonregular connection unit), a sector unit controlling the processing of sectors and the NESPINN chip. The chip has a control unit and eight processing elements; each processing element has 2 Kbytes of on-chip local memory and an off-chip neuron state memory. The chip has been designed and simulated and will be implemented in 0.5-μm CMOS. It will operate at 50 MHz in either SIMD or dataflow mode. The estimated performance of the system with one NESPINN chip for a model network with 16K neurons of 83 connections each is 10^{11} CUPS.
- *CNS-1* from University of California Berkeley is the acronym from Connectionist Network Supercomputer-1 (Asanović *et al* 1993a–c, 1994) and is currently under development. It is targeted for speech and language processing as well as early and high-level vision and large conceptual knowledge representation studies. The CNS-1 is similar to other massively parallel computers with major differences in the architectural details of the processing nodes and the communication mechanisms. Processing nodes will be connected in a mesh topology and operate independently in a MIMD style. The processor node, named Torrent, includes: an MIPS CPU with a vector coprocessor running at 125 MHz, a Rambus external memory interface, and a network interface. The design is scalable up to 1024 Torrent processing nodes, for a total of up to 2 TeraOps and 32 Gbytes of RAM. The host and other devices will connect to CNS-1 through custom VLSI I/O nodes named Hydrant connected to one edge of the mesh and allowing up to 8 Gbytes of I/O bandwidth. A sketch of the future CNS-1 can be seen in figure E1.4.2. The goal set ahead is to be able to evaluate networks with one million neurons and an average of one thousand connections per unit (i.e., a total of a billion connections) at a rate of 100 times per second, or 10^{11} CPS and 2×10^{10} CUPS.

Several of the implementations presented here have been plotted in figures E1.4.3 (digital neurochips) and E1.4.4 (dedicated neurocomputers and supercomputers). As can be seen from these two figures, some architectural improvements are to be expected from the techniques used in designs like the SMA and the NESPINN, which could reach speed performances similar to the CNS-1.

We shall not end this section before mentioning an interesting alternative that has recently emerged. To cope with the limited accuracy, new learning algorithms with quantized weights have started to appear (see also Section E1.2.4). One might call them 'VLSI-friendly learning algorithms', which was the topic covered in MicroNeuro'94. Such algorithms could be used to map neural networks onto FPGAs or to custom-integrate circuits. The first such learning algorithm (Armstrong and Gecsie 1979, 1991) is in fact synthesizing Boolean functions using adaptive tree networks whose elements—after training and elimination of redundant elements—perform classical (Boolean) logical operations (AND and OR). This

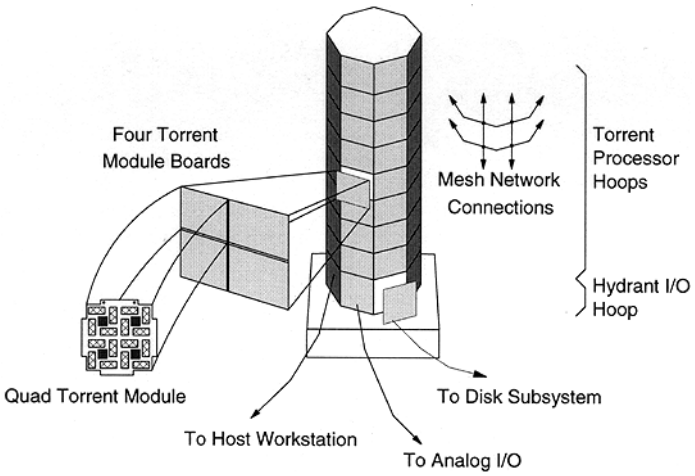

Figure E1.4.2. Connectionist Network Supercomputer CNS-1 (adapted from Asanović *et al* 1993b).

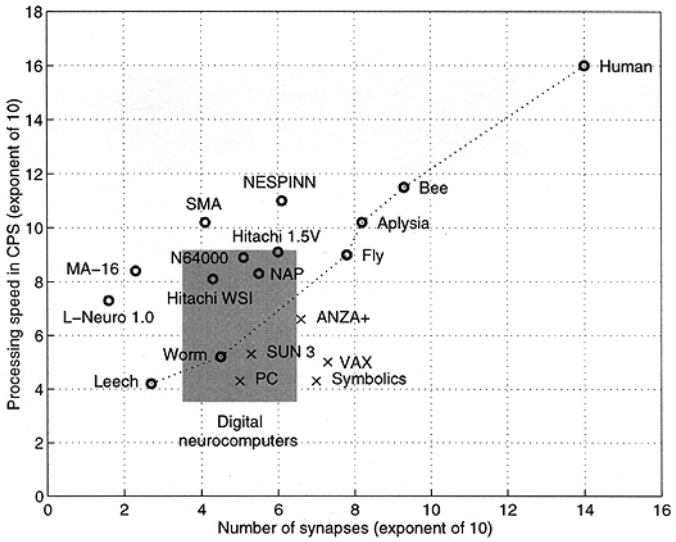

Figure E1.4.3. Different neurochips (circles) and classical computers (crosses) used for implementing artificial neural networks.

line of research has been extended by using a combination of AND and OR gates after an initial layer of threshold gates (Ayestaran and Prager 1993, Bose and Garga 1993). New learning algorithms have been developed by quantizing other learning algorithms (Höhfeld and Fahlman 1991, 1992, Jabri and Flower 1991, Makram-Ebeid *et al* 1989, Pérez *et al* 1992, Sakaue *et al* 1993, Shoemaker *et al* 1990, Thiran *et al* 1991, 1993) or by devising new ones (Fiesler *et al* 1990, Höhfeld and Fahlman 1991, Hollis and Paulos 1994, Hollis *et al* 1991, Mézard and Nadal 1989, Nakayama and Katayama 1991, Oliveira and Sangiovanni-Vincentelli 1994, Walter 1989, Xie and Jabri 1992), a particular class being the one dealing with threshold gates (Beiu *et al* 1994a, Beiu and Taylor 1995a, b, 1996a, Diederich and Opper 1987, Gruau 1993, Krauth and Mézard 1987, Kim and Park 1995, Littlestone 1988, Raghavan 1988, Roy *et al* 1993, Tan and Vandewalle 1992, 1993, Venkatesh 1989). Four overviews have compared and discussed such constructive algorithms (Śmieja 1993, Fiesler 1994, Moerland and Fiesler 1996, Beiu 1996c).

The main conclusion is that a lot of effort and creativity has been used recently to improve digital solutions for implementing artificial neural networks. The many designs proposed over the years make this area a lively topic confirming its huge interest. Fresh proposals together with estimates and/or results already show impressive performances competing with analog chips and reaching towards an area which, not so long ago, was considered accessible only for future optical computing.

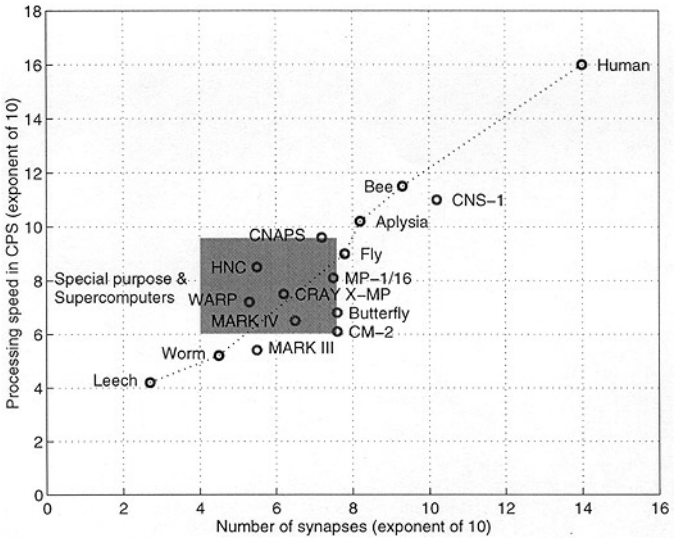

Figure E1.4.4. Different neurocomputers and supercomputers used for implementing artificial neural networks.

References

Abu-Mostafa Y S 1988a Connectivity versus entropy *Proc. Conf. on Neural Information Processing Systems* pp 1–8

——1988b Lower bound for connectivity in local-learning neural networks *J. Complexity* **4** 246–55

——1989 The Vapnik-Chervonenkis dimension information versus complexity in learning *Neural Comput.* **1** 312–7

Adaptive Solutions 1991 *CNAPS Neurocomputing Information Sheet on the CNAPS Neurocomputing system* (Adaptive Solutions Inc, 1400 NW Compton Drive Suite, 340 Beaverton, OR 97006, USA)

——1992 *CNAPS Server Preliminary Data Sheet* (Adaptive Solutions Inc, 1400 NW Compton Drive Suite, 340 Beaverton, OR 97006, USA)

Aihara K, Fujita O and Uchimura K 1996 A digital neural network LSI using sparse memory access architecture *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 139–48

Akers L A, Walker M R, Ferry D K and Grondin R O 1988 Limited interconnectivity in synthetic neural systems *Neural Computers* eds R Eckmiller and C von der Malsburg (Berlin: Springer) pp 407–16

Albrecht A 1992 On bounded-depth threshold circuits for pattern functions *Proc. Int. Conf. on Artificial Neural Networks (1992)* (Amsterdam: Elsevier) pp 135–38

Aleksander I and Morton H B 1990 An overview of weightless neural nets *Proc. Int. Joint Conf. on Neural Networks (Washington, 1990)* vol II pp 499–502

Alippi C 1991 Weight representation and network complexity reductions *The Digital VLSI Implementation of Neural Nets Research Note RN/91/22* Department of Computer Science University College, London, February

Alippi C, Bonfanti S and Storti-Gajani G 1990a Some simple bounds for approximations of sigmoidal functions *Layered Neural Nets Report No 90-022* (Department of EE, Polytechnic of Milano)

——1990b Approximating sigmoidal functions for VLSI implementation of neural nets *Proc. Int. Conf. on Microelectronics for Neural Networks (1990)* pp 165–170.4

Alippi C and Nigri M 1991 Hardware requirements for digital VLSI implementation of neural networks *Proc. Int. Joint Conf. on Neural Networks (1991)* pp 1873–8

Alippi C and Storti-Gajani G 1991 Simple approximation of sigmoidal functions realistic design of digital neural networks capable of learning *Proc. Int. Symp. on Circuits and Systems (Singapore, 1991)* (Los Alamitos, CA: IEEE Computer Society Press) pp 1505–8

Alla P Y, Dreyfus G, Gascuel J D, Johannet A, Personnaz L, Roman J and Weinfeld M 1990 Silicon integration of learning algorithm and other auto-adaptive properties in a digital feedback neural network *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 341–6

Allender E 1989 A note on the power of threshold circuits *IEEE Symp. on the Foundation of Computer Science* p 30

Alon N and Bruck J 1991 Explicit construction of depth-2 majority circuits for comparison and addition *Research Report RJ 8300 (75661)* (IBM Almaden, San Jose, CA)

Alspector J and Allen R B 1987 Neuromorphic VLSI Learning System *Advanced Research in VLSI, Proc. 1987 Stanford Conf.* ed P Losleben (Cambridge MA: MIT Press)

Alspector J, Allen R B, Hu V and Satyanaranayana S 1988 Stochastic learning networks and their electronic implementation *Proc. Conf. on Neural Information Processing Systems (1987)* pp 9–21

Amaldi E and Mayoraz E (eds) 1992 *Mathematical Foundations of Artificial Neural Networks (Summer School, Sion, September 1992)* (Swiss Federal Institute of Technology–Lausanne and Kurt Bösch Academic Institute–Sion 1992)

Annaratone M, Arnould E, Gross T, Kung H T, Lam M, Menzilcioglu O and Webb J A 1987 The WARP computer architecture implementation and performance *IEEE Trans. Comput.* **36** 1523–38

Anderson J A and Rosenfeld E 1988 *Neurocomputing: Foundations of Research* (Cambridge, MA: MIT Press)

Antognetti P and Milutinovic V (eds) 1991 *Neural Networks: Concepts Applications and Implementations* vol 2 (Englewood Cliffs, NJ: Prentice Hall)

Arai M 1993 Bounds on the number of hidden units in binary-valued three-layer neural networks *Neural Networks* **6** 855–60

Armstrong W W and Gecsei J 1979 Adaption algorithms for binary tree networks *IEEE Trans. Syst. Man Cybern.* **9** 276–85

Armstrong W W, Dwelly A, Liang J, Lin D and Reynolds S 1991 Some results concerning adaptive logic networks *Technical Report* (Department of Computer Science, University of Alberta, Edmonton, Canada)

Arnould E 1985 A systolic array computer *Proc. IEEE Int. Conf. on Application Specific Signal processing (Tampa, FL, 1985)* pp 232–5

Asanović K, Beck J, Callahan T, Feldman J, Irissou B, Kingsbury B, Kohn P, Lazzaro J, Morgan N, Stoutamire D and Wawrzynek J 1993a *CNS-1 Architecture Specification, Technical Report TR-93-021* (International Computer Science Institute and Unviersity of California, Berkeley)

Asanović K, Beck J, Feldman J, Morgan N and Wawrzynek J 1993b Development of a connectionist network supercomputer *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 253–62

——1993c Designing a Connectionist Network Supercomputer *Int. J. Neural Systems* **4** 317–26

——1994 A supercomputer for neural computation *Proc. IEEE Int. Conf. on Neural Networks* vol 1 (Los Alamitos, CA: IEEE Computer Society Press) pp 5–9

Asanović K, Beck J, Kingsbury B E D, Kohn P, Morgan N and Wawrzynek J 1992 SPERT: A VLIW/SIMD neuro-processor *Proc. Int. Joint Conf. on Neural Networks (1992)* vol II pp 577–82

Asanović K and Morgan N 1991 Experimental determination of precision requirements for back-propagation training of artificial neural networks *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 9–15

Asanović K, Morgan N and Wawrzynek J 1993d Using simulations of reduced precision arithmetic to design a neuro-microprocessor *J. VLSI Signal Processing* **6** 3–44

Atlas L E and Suzuki Y 1989 Digital systems for artificial neural networks *IEEE Circuits and Devices Mag.* **5** 20–4

Avellana N, Strey A, Holgado R, Fernández J A, Capillas R and Valderrama E 1996 Design of a low-cost and high-speed neurocomputer system *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 221–6

Ayestaran H E and Prager R W 1993 *The Logical Gates Growing Network Technical Report 137* (Cambridge University Engineering Department, F-INFENG, July)

Baker T and Hammerstrom D 1988 Modifications to artificial neural network models for digital hardware implementation *Technical Report CS/E 88-035* (Department of Computer Science and Engineering, Oregon Graduate Center)

Barhen J, Toomarian N, Fijany A, Yariv A and Agranat A 1992 New directions in massively parallel neurocomputing *Proc. NeuroNimes '92* pp 543–54

Baum E B 1988a Supervised learning of probability distributions by neural networks *Proc. Conf. on Neural Information Processing Systems (1987)* pp 52–61

——1988b On the capabilities of multilayer perceptrons *J. Complexity* **4** 193–215

Baum E B and Haussler D 1989 What size net gives valid generalization? *Neural Comput.* **1** 151–60

Beck J 1990 The ring array processor (RAP) hardware *Technical Report TR-90-048* (International Computer Science Institute, Berkeley, CA, September)

Beichter J, Bruels N, Meister E, Ramacher U and Klar H 1991 Design of a general-purpose neural signal processor *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 311–5

Beiu V 1989 From systolic arrays to neural networks *Sci. Ann. Informatics* **35** 375–85

——1994 Neural networks using threshold gates a complexity analysis of their area- and time-efficient VLSI implementations *PhD Dissertation* Katholieke Universiteit, Leuven, Belgium, x-27-151779-3

——1996a Constant fan-in digital neural networks are VLSI-optimal *1st Int. Conf. on Mathematics of Neural Networks and Applications (Oxford, 1995)* (*Ann. Math. Artif. Intell.* to appear)

——1996b Entropy bounds for classification algorithms *Neural Network World* **6** 497–505

——1996c Optimal VLSI implementation of neural networks: VLSI-friendly learning algorithm *Neural Networks and Their Applications* ed T G Taylor (Chichester: Wiley) pp 255–76

——1997 VLSI components *Complexity of Discrete Neural Networks* (New York: Gordon and Breach) accepted for publication

Beiu V, Peperstraete J A and Lauwereins R 1992 Using threshold gates to implement sigmoid nonlinearity *Proc. Int. Conf. on Artificial Neural Networks (1992)* vol II pp 1447–50

Beiu V, Peperstraete J, Vandewalle J and Lauwereins R 1993 Close approximations of sigmoid functions by sum of steps for VLSI implementation of neural networks *Proc. Romanian Symp. on Computer Science (Jassy, Romania 1993)* pp 31–50

——1994a Learning from examples and VLSI implementation of neural networks *Cybernetics and Systems '94, Proc. 12th Euro. Meeting on Cybernetics and Systems Research (Vienna, 1994)* vol II ed R Trappl (Singapore: World Scientific) pp 1767–74

——1994b VLSI Complexity reduction by piece-wise approximations of the sigmoid function *Proc. Euro. Symp. on Artificial Neural Networks (Brussels)* ed M Verleysen (Brussels: De facto) pp 181–6

——1994c Area-time performances of some neural computations *Proc. IMACS Int. Sump. on Signal Processing Robotics and Artificial Neural Networks (Lille, France)* ed P Borne, T Fukuda and S G Tzafestas (Lille: GERF EC) pp 664–8

——1994d On the circuit complexity of feedforward neural networks *Proc. Int. Conf. on Artificial Neural Networks (1994)* pp 521–4

——1994e Placing feedforward neural networks among several circuit complexity classes proceedings *World Congr. on Neural Networks (San Diego, CA, 1994)* vol II (Lawrence Erlbaum Associates/INNS Press) pp 584–9

Beiu V and Rosu I 1985 VLSI implementation of a self-testable real content addressable memory *Proc. 6th Int. Conf. on Control System and Computer Science (Bucharest, Romania, 1985)* vol 2 pp 400–5

Beiu V and Taylor J G 1995a VLSI optimal learning algorithm ed D W Pearson, N C Steele and R F Albrecht *Artificial Neural Nets and Genetic Algorithms, Proc. Int. Conf. on Artificial Neural Networks and Genetic Algorithms (Alès, France, 1995)* (Berlin: Springer) pp 61–4

——1995b Area-efficient constructive learning algorithm *Proc. 10th Int. Conf. on Control Systems and Computer Science (Bucharest, Romania, 1995)* vol 3 pp 293–310

——1995c Optimal mapping of neural networks onto FPGAs–a new constructive algorithm *From Natural to Artificial Neural Computations Lecture Notes in Computer Science* vol 930 eds J Mira and F Sandoval (Berlin: Springer) pp 822–9

——1996a Direct synthesis of neural networks *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 257–64

——1996b On the circuit complexity of sigmoid feedforward neural networks *Neural Networks* accepted

Bengtsson L, Linde A, Svensson B, Taveniku M and Ehlander A 1993 The REMAP massively parallel computer platform for neural computations *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 47–62

Blank T 1990 The MasPar MP-1 architecture *Proc. 35th IEEE Computer Society Int. Conf., Spring COMPCON '90 (San Francisco)* pp 20–4

Blayo F and Hurat P 1989 A VLSI systolic array dedicated to Hopfield neural networks *VLSI for Artificial Intelligence* ed J G Delgado-Frias and W R Moore (New York: Kluwer)

Bose N K and Garga A K 1993 Neural network design using Voronoi diagrams *IEEE Trans. Neural Networks* **4** 778–87

Boser B E, Sackinger E, Bromley J, le Cun Y and Jackel L D 1992 Hardware requirements for neural network pattern classifiers *IEEE Micro Mag.* **12** 32–40

Botros N M and Abdul-Aziz M 1994 Hardware implementation of an artificial neural network using field programmable gate arrays (FPGA's) *IEEE Trans. Indust. Electron.* **41** 665–8

Boyd J 1990 Hitachi's neural computer *Electronic World News* 10 December, 6–8

Bruck J 1990 Harmonic analysis of polynomial threshold functions *SIAM J. Discrete Math.* **3** 168–77

Bruck J and Smolensky R 1989 Polynomial threshold functions, AC^0 functions and spectral norms *Research Report RJ 7410 (67387)* (IBM Yorktown Heights, New York)

——1992 Polynomial threshold functions AC^0 functions and spectral norms *SIAM J. Comput.* **21** 33–42

Bulsari A 1993 Some analytical solutions to the general approximation problem for feedforward neural networks *Neural Networks* **6** 991–6

Burr J B 1991 Neural network implementations *Neural Networks Concepts: Applications and Implementations* vol 2 ed P Antognetti and V Milutinovic (Englewood Cliffs, NJ: Prentice Hall)

——1992 Digital neurochip design *Digital Parallel Implementations of Neural Networks* ed K W Przytula and V K Prasanna (Englewood Cliffs, NJ: Prentice Hall)

Cameron S H 1969 An estimate of the complexity requisite in a universal decision network *Bionics Symp. (Wright Airforce Development Division WADD Report 60-600)* pp 197–212

Clarkson T G, Gorse D and Taylor J G 1989 Hardware realisable models of neural processing *Proc. 1st IEE Int. Conf. on Artificial Neural Nets, IEE Publication 313* (London: IEE) pp 242–6

——1990 pRAM automata *Proc. IEEE Int. Workshop on Cellular Neural Networks and Their Applications (Budapest, 1990)* pp 235–43

——1991a Biologically plausible learning in hardware realisable nets *Proc. Int. Conf. on Artificial Neural Networks (1991)* pp 195–9

——1992a From wetware to hardware reverse engineering using probabilistic RAMs *J. Intell. Syst.* **2** 11–30

Clarkson T G, Gorse D, Taylor J G and Ng C K 1992b Learning probabilistic RAM nets using VLSI structures *IEEE Trans. Computer* **41** 1552–61

Clarkson T G, Guan Y, Taylor J G and Gorse D 1993a Generalization in probabilistic RAM nets *IEEE Trans. Neural Networks* **4** 360–3

Clarkson T G and Ng C K 1993 Multiple learning configurations using 4th generation pRAM modules *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 233–40

Clarkson T G, Ng C K, Gorse D and Taylor J G 1991b A serial update VLSI architecture for the learning probabilistic RAM neuron *Proc. Int. Conf. on Artificial Neural Networks (1991)* pp 1573–6

Clarkson T G, Ng C K and Guan Y 1993b The pRAM An adaptive VLSI chip *IEEE Trans. Neural Networks* **4** 408–12

Cohen S and Winder R O 1969 Threshold gate building blocks *IEE Trans. Computer* **18** 816–23

Cover T M 1965 Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition *IEEE Trans. Electron. Computer* **14** 326–34

Craik K J W 1943 *The Nature of Explanation* (Cambridge: Cambridge University Press)

Cruz C A, Hanson W A and Tam J Y 1987 Neural network emulation hardware design considerations *Proc. Int. Joint Conf. on Neural Networks (1987)* vol III pp 427–34

Cybenko G 1988 Continuous valued neural networks with two hidden layers are sufficient *Technical Report* (Tufts University)

——1989 Approximations by superpositions of a sigmoidal function *Math. Control Signal Syst.* **2** 303–14

DARPA 1989 DARPA neural network study final report October 1987–February 1988 *Technical Report 840* (Lincoln Laboratory, MIT)

Das Gupta B and Schnitger G 1993 The power of approximating a comparison of activation functions *Conf. on Neural Information Processing Systems (1992)* pp 615–22

Dejean C and Caillaud F 1994 Parallel implementations of neural networks using the L-Neuro 2.0 architecture *Proc. 1994 Int. Conf. on Solid State Devices and Materials (Yokohama Japan)* pp 388–90

DelCorso D, Grosspietch K E and Treleaven P 1989 European approaches to VLSI neural networks *IEEE Micro Mag.* **9**

Delgado-Frias J and Moore W R 1994 *VLSI for Neural Networks and Artificial Intelligence, An Edited Selection of the Papers Presented at the Int. Workshop on VLSI for Neural Networks and Artificial Intelligence (Oxford, 2–4 September, 1992)* (New York: Plenum)

Dembo A, Siu K-Y and Kailath T 1990 Complexity of finite precision neural network classifier *Proc. Conf. on Neural Information Processing Systems (1989)* pp 668–75

Denker J S (ed) 1986 Neural network for computing *Proc. AIP Conf. on Neural Networks for Computing (Snowbird, Utah, 1986)* (New York: American Institute of Physics)

Denker J S and Wittner B S 1988 Network generality training required and precision required *Proc. Conf. on Neural Information Processing Systems (1987)* pp 219–22

Deprit E 1989 Recurrent backpropagation on the connection machine *Neural Networks* **2** 295–314

Dertouzos M L 1965 *Threshold Logic: A Synthesis Approach* (Cambridge, MA: MIT Press)

Deville Y 1993 A Neural implementation of complex activation functions for digital VLSI *Neural Networks Microelectron. J.* **24** 259–62

Diederich S and Opper M 1987 Learning of correlated patterns in spin-glass networks by local learning rules *Phys. Rev. Lett.* **58** 949–52

Disante F, Sami M G, Stefanelli R and Storti-Gajani G 1989 Alternative approaches for mapping neural networks onto silicon *Proc. Int. Workshop on Artificial Neural Networks (Vietri sul Mare, Italy, 1989)* pp 319–28

——1990a A configurable array architecture for WSI implementation of neural networks *Proc. IEEE IPCC* Phoenix AZ March

——1990b A compact and fast silicon implementation for layered neural nets *Proc. Int. Workshop on VLSI for Artificial Intelligence and Neural Networks (Oxford)*

Duranton M 1996 L-Neuro 2.3: a VLSI for image processing by neural networks *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 157–60

Duranton M, Gobert J and Maudit N 1989 A digital VLSI module for neural networks *Neural Networks from Models to Applications, Proc. nEuro '88 (Paris, June 1988)* (Paris: IDSET) pp 720–4

Duranton M and Maudit N 1989 A general purpose digital architecture for neural network simulation *Proc. IEE Int. Neural Network Conf. (1989)* (London: IEE) pp 62–66

Duranton M and Sirat J A 1989 A general purpose digital neurochip *Proc. Int. Joint Conf. on Neural Networks (Washington, 1989)*

——1990 Learning on VLSI: a general-purpose digital neurochip *Philips J. Res.* **45** 1–17

Eckmiller R, Hartman G and Hauske G (eds) 1990 *Parallel Processing in Neural Systems and Computers* (Amsterdam: North-Holland)

Eckmiller R and von der Malsburg C (eds) 1988 Neural computers *Proc. NATO Advanced Research Workshop on Neural Computers (Neuss, Germany)* (Berlin: Springer)

El-Mousa A H and Clarkson T G 1996 Multi-configurable pRAM based neurocomputer *Neural Network World* **6** 587–96

Erdogan S S and Wahab A 1992 Design of RM-nc a reconfigurable neurcomputer for massively parallel-pipelined computations *Proc. Int. Joint Conf. on Neural Networks (1992)* vol II pp 33–8

Ernoult C 1988 Performance of backpropagation ona parallel transputer-based machine *Proc. Neuro Nimes '88 (Nimes, France)* pp 311–24

Ernst H P, Mokry B and Schreter Z 1990 A transputer based general simulator for connectionist models *Parallel Processing in Neural Systems and Computers* ed G Hartmann and G Hauske (Amsterdam: North-Holland) pp 283–6

Faggin F 1991 Hardware (VLSI) Implementations of Neural Networks. Tutorial 3(b) *Int. Conf. on Artificial Neural Networks 1991)*

Faggin F and Mead C A 1990 VLSI Implementation of neural networks *An Introduction to Neural and Electronic Networks* eds S F Zornetzer, J L Davis and L Clifford (San Diego, CA: Academic Press) pp 275–92

Fiesler E 1994 Comparative bibliography of ontogenetic neural networks *Proc. Int. Conf. on Artificial Neural Networks (1994)* vol I pp 793–6

Fiesler E, Choudry A and Caulfield H J 1990 A universal weight discretization method for multi-layer neural networks *IEEE Trans. Syst. Man Cybern.* accepted (see also Fiesler E, Choudry A and Caulfield H J 1990 A weight discretization paradigm for optical neural networks *Proc. Int. Congr. on Optical Science and Engineering (Bellingham, Washington)* SPIE vol 1281 (SPIE) pp 164–73

Fischler M A 1962 Investigations concerning the theory and synthesis of linearly separable switching functions *PhD Dissertation* Department EE, Stanford University, USA

Flynn M J 1972 Some computer organization and their effectiveness *IEEE Trans. Comput.* **21** pp 948–60

Fornaciari W, Salice F and Storti-Gajani G 1991a Automatic synthesis of digital neural architectures *Proc. Int. Joint Conf. on Neural Networks (1991)* pp 1861–6

——1991b A formal method for automatic synthesis of neural networks *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 367–80

Funahashi K-I 1989 On the approximate realization of continuous mapping by neural networks *Neural Networks* **2** 183–92

Funahashi K-I and Nakamura Y 1993 Approximation of dynamic systems by continuous time recurrent neural networks *Neural Networks* **6** 801–6

Furst M, Saxe J B and Sipser M 1981 Parity circuits and the polynomial-time hierarchy *Proc. IEEE Symp. on Foundations of Computer Science* **22** 260–70 (also in 1984 *Math. Syst. Theory* **17** 13–27

Gamrat C, Mougin A, Peretto P and Ulrich O 1991 The architecture of MIND neurocomputers *Proc. MicroNeuro Int. Conf. on Microelectronics for Neural Networks (1991)* pp 463–9

Gascuel J-D, Delaunay E, Montoliu L, Moobed B and Weinfeld M 1992 A custom associative chip used as building block for a software reconfigurable multi-networks simulator *Proc. 3rd Int. Workshop on VLSI for Artificial Intelligence and Neural Networks (Oxford)*

Gick S, Heusinger P and Reuter A 1993 Automatic synthesis of neural networks to programmable hardware *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 115–20

Girau B and Tisserand A 1996 On-line arithmetic-based reprogrammable hardware Implementation of multilayer perceptron back-propagation *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 168–75

Glesner M, Huch M, Pöchmüller W and Palm G 1989 Hardware implementations for neural networks *Proc. IFIP Workshop on Parallel Architectures on Silicon (Grenoble, France)* pp 65–79

Glesner M and Pöchmüller W 1991 *Circuit Diagrams and Timing Diagrams of BACCHUS III* (Darmstadt University of Technology, Institute for Microelectronic Systems, Karlstraße 15, D-6100 Darmstadt, Germany)

——1994 *Neurocomputers–An Overview of Neural Networks in VLSI* (London: Chapman and Hall)

Glover M A and Miller W T 1994 A massively-parallel SIMD processor for neural networks and machine vision applications *Proc. Conf. on Neural Information Processing Systems (1993)* pp 843–49

Goldmann J and Karpinski M 1994 Simulating threshold circuits by majority circuits *Technical Report TR-94-030* (International Computer Science Institute, Berkeley, California) (a preliminary version appeared in 1963 *Proc. 25th ACM Symp. on Theory of Computation* (New York: ACM) pp 551–60)

Gorse D and Taylor J G 1989a On the identity and properties of noisy neural and probabilistic RAM nets *Phys. Lett.* A **131** 326–32

——1989b An analysis of noisy RAM and neural nets *Physica* D **34** 90–114

——1990a A general model of stochastic neural processing *Biol. Cybern.* **63** 299–306

——1990b Hardware-Realisable Learning Algorithms *Proc. Int. Conf. on Neural Network (Paris, 1990)* (Dordrecht: Kluwer) pp 821–4

——1991a Universal associative stochastic learning automata *Neural Network World* **1** 192–202

——1991b Learning sequential structure with recurrent pRAM nets *Proc. Int. Joint Conf. on Neural Networks (1991)* vol II pp 37–42

——1991c A continuous input RAM-based stochastic neural model *Neural Networks* **4** 657–65

Goser K, Hilleringmann U, Rückert U and Schumacher K 1989 VLSI Technologies for artificial neural networks *IEEE Micro Mag.* **9** 28–44

Graf H P and de Vegvar P 1987a A CMOS implementation of a neural network model *Advanced Research in VLSI, Proc. Stanford Conf. on Advanced Research on VLSI* ed P Losleben (Cambridge, MA: MIT Press) pp 351–67

——1987b A CMOS associative chip based on neural networks *Proc. IEEE Int. Solid-State Circuits Conf. (New York, 1987)* pp 304, 305 and 437

Graf H P, Hubbard W, Jackel L D and de Vegvar P 1987 A CMOS associative memory chip *Proc. Int. Joint Conf. on Neural Networks (1987)* vol III pp 461–8

Graf H P, Jackel L D, Howard R E, Straughn B, Denker J S, Hubbard W, Tennant D M and Schwartz D 1986 Implementation of a neural network memory with several hundreds of neurons *Neural Network for Computing, Proc. AIP Conf. on Neural Networks for Computing (Snowbird, Utah)* ed J S Denker (New York: American Institute of Physics) pp 182–7

Graf H P, Sackinger E, Boser B and Jackel L D 1991 Recent developments of electronic neural nets in USA and Canada *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 471–88

Graf H P, Sackinger E and Jackel L D 1993 Recent developments of electronic neural nets in North America *J. VLSI Signal Processing* **6** 19–31

Grajski K A, Chinn G, Chen C, Kuszmaul C and Tomboulian S 1990 *Neural Network Simulation on the MasPar MP-1 Massively Parallel Computer, MasPar information sheet TW007 0690* (MasPar Computer Corporation, 749 North Mary Avenue, Sunnyvale, CA 94086, USA)

Griffin M, Tahara G, Knorpp K, Pinkham P and Riley B 1991 An 11 million transistor neural network execution engine *Proc. IEEE Int. Solid-State Circuits Conf. (San Francisco, CA, 1991)* pp 180–1

Gruau F 1993 Learning and pruning algorithm for genetic boolean neural networks Proc. Euro. Symp. on Artificial Neural Networks (Brussels, 1993) ed M Verleysen (Brussels: de facto) pp 57–63

Guan Y, Clarkson T G, Gorse D and Taylor J G 1992 The application of noisy reward/penalty learning to pyramidal pRAM structures *Proc. Int. Joint Conf. on Neural Networks (1992)* vol III pp 660–5

Gunzinger A, Müller U, Scott W, Bäumle B, Kohler P and Guggenbühl W 1992 Architecture and realization of a multi signal processor system *Proc. Application Specific Array Processors (1992)* ed J Fortes, E Lee and T Meng (Los Alamitos, CA: IEEE Computer Society Press) pp 327–340.2

Hajnal A, Maass W, Pudlák P, Szegedy M and Turán G 1987 Threshold circuits of bounded depth *Proc. IEEE Symp. on Foundations of Computer Science* **28** 99–110 (also in 1993 *J. Computing System Science* **46** 129–54)

Halgamuge S K, Pöchmüller W and Glesner M 1991 Computational hardware requirements for the backpropagation algorithm *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 47–52

Hammerstrom D 1988 The connectivity analysis of simple associations–or–how many connections do you need *Proc. Conf. on Neural Information Processing Systems (1987)* pp 338–47

——1990 A VLSI Architecture for high-performance low-cost on-chip learning *Proc. Int. Joint Conf. on Neural Networks (1990)* vol II pp 537–43

——1995 Digital VLSI for neural networks *The Handbook of Brain Theory and Neural Networks* ed M A Arbib (Cambridge, MA: MIT Press) pp 304–9

Hammerstrom D and Nguyen N 1991 An implementation of Kohonen's self-organizing map on the adaptive solution neurocomputer *Proc. Int. Conf. on Artificial Neural Networks (1991)* vol I pp 715–20

Hassoun M H (ed) 1993 *Associative Neural Memories Theory and Implementation* (New York: Oxford University Press)

Håstad J 1986 Almost optimal lower bounds for small depth circuits *Proc. ACM Symp. on Theory of Computing (1986)* vol 18 pp 6–20

Heemskerk J N H Neurocomputers for brain-style processing. Design, implementation and application *PhD Thesis* Leiden University, The Netherlands (Chapter 3: 'Overview of Neural Hardware' is available via ftp from: ftp.mrc-apu.cam.ac.uk/pub/nn)

Heemskerk J N H, Murre J M J, Hoekstra J, Kemna L H J G and Hudson P T W 1991 The BSP400: a modular neurocomputer assembled from 400 low-cost microprocessors *Proc. Int. Conf. on Artificial Neural Networks (1991)* vol I pp 709–14

Hecht-Nielsen R 1987 Kolmogorov's mapping neural network existence theorem *Proc. Int. Joint Conf. on Neural Networks (1987)* vol III pp 11–13

——1988 Neurocomputing picking the human brain *IEEE Spectrum* **25** 36–41

——1989 *Neurocomputing* (Reading, MA: Addison Wesley)

——1991 *Computers Information sheet on HNC neural network products* (HNC Inc., 5501 Oberlin Drive, San Diego, CA 92121, USA)

Hirai Y 1991 Hardware implementation of neural networks in Japan *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 435–46

Hiraiwa A, Kurosu S, Arisawa S and Inoue M 1990 A two level pipeline RISC processor array for ANN *Proc. Int. Joint Conf. on Neural Networks (1990)* vol II pp 137–40

Hochet B, Peiris V, Abdo S and Declercq M 1991 Implementation of a learning Kohonen neuron *IEEE J. Solid-State Circuits* **26** 262–7

Hofmeister T, Hohberg W and Köhling S 1991 Some notes on threshold circuits and multiplication in depth 4 *Info. Processing Lett.* **39** 219–26

Höhfeld M 1990 Fixed point arithmetic in feedforward neural networks *Technical Report FKS3-108* (Siemens AG, Münich)

Höhfeld M and Fahlman S E 1992 Probabilistic rounding in neural network with limited precision *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 1–8 (also in 1992 *Neurocomputing* **4** 291–9)

——1992 Learning with limited numerical precision using the cascade-correlation algorithm *Technical Report CMU-CS-91-130* (School of Computer Science, Carnegie Mellon) (also in *IEEE Trans. Neural Networks* **3** 602–11)

Holler M A 1991 VLSI implementation of learning and memory systems: a review *Proc. Conf. on Neural Information Processing Systems (1990)* pp 993–1000

Holler M A, Park C, Diamond J, Santoni U, The S C, Glier M, Scofield C L and Núñez L 1992 A high performance adaptive classifier using radial basis functions *Proc. Government Microcircuit Application Conf. (Las Vegas, Nevada)*

Hollis P W, Harper J S and Paulos J J 1990 The effects of precision constraints in a backpropagation learning network *Neural Comput.* **2** 363–73

Hollis P W and Paulos J J 1994 A neural network learning algorithm tailored for VLSI implementation *IEEE Trans. Neural Networks* **5** 784–91

Hollis P W, Paulos J J and D'Costa C J 1991 An optimized learning algorithm for VLSI implementation *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 121–6

Holt J L and Hwang J-N 1991 Finite precision error analysis of neural network hardware implementations *Technical Report FT-10* (University of Washington, Seattle)

——1993 Finite precision error analysis of neural network hardware implementations *IEEE Trans. Computer* **42** 281–90

Hong J 1987 On connectionist models *Technical Report* (Department of Computer Science, University of Chicago)

Hornik H 1991 Approximation capabilities of multilayer feedforward networks *Neural Network* **4** 251–7

——1993 Some new results on neural network approximation *Neural Network* **6** 1069–72

Hornik K, Stinchcombe M and White H 1989 Multilayer feedforward neural networks are universal approximators *Neural Network* **2** 359–66

——1990 Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks *Neural Network* **3** 551–60

Hu S 1965 *Threshold Logic* (Berkeley and Los Angeles: University of California Press)

Huang S-C and Huang Y-F 1991 Bounds on number of hidden neursons of multilayer perceptrons in classification and recognition *IEEE Trans. Neural Networks* **2** 47–55

Huch M, Pöchmüller W and Glesner M 1990 Bacchus: a VLSI architecture for a large binary associative memory *Proc. Int. Conf. on Neural Networks (Paris, 1990)* vol II pp 661–4

Hush D R and Horne B G 1993 Progress in supervised neural networks *IEEE Signal Proc. Mag.* **10** 8–39

Ienne P 1993a Quantitative comparison of architectures for digital neuro-computers *Proc. Int. Joint Conf. on Neural Networks (Nagoya, 1993)* pp 1987–1990

——1993b Architectures for neuro-computers: review and performance evaluation *Technical Report no 21/93* (Swiss Federal Institute of Technology, Lausanne)

Immerman N and Landau S 1989 The complexity of integrated multiplication *Proc. Structure in Complexity Theory Symp.* pp 104–111

Intel Corporation 1992a *Intel Neural Network Solutions Order Number 296961-002* (Intel Corporation, 2200 Mission College Boulevard, Mail Stop RN3–17, Santa Clara, CA 95052-8119, USA)

——1992b *Intel Neural Network Products Price and Availability Order Number 296961-002* (Intel Corporation, 2200 Mission College Boulevard, Mail Stop RN3-17, Santa Clara, CA 95052-8119, USA)

Iwata A 1990 Neural devices and networks *Sixth German–Japanese Forum on Information Technology (Berlin, 1990)*

Jabri M A and Flower B 1991 Weight perturbation: an optimal architecture and learning technique for analog VLSI feedforward and recurrent multi-layer networks *SEDAL Technical Report* Department of EE, University of Sydney (1992 *IEEE Trans. Neural Networks* **3** 154–7)

Jackel L D 1991 Practical issues for electronic neural-nets hardware–tutorial notes *Conf. on Neural Information Processing Systems (1991)*

——1992 Neural nets hardware. Tutorial 4 *CompEuro '92 (The Hague, The Netherlands, 1992)*

Jackel L D, Graf H P and Howard R E 1987 Electronic neural network chips *Appl. Opt.* **26** 5077–80

Jackson D and Hammerstrom D 1991 Distributed back propagation networks over the Intel iPSC/860 hypercube *Proc. Int. Joint Conf. on Neural Networks (1991)* vol I pp 569–74

Jahnke A, Roth U and Klar H 1996 A SIMD/DataflowArchitecture for a neurocomputer for spike-processing neural networks (NESPINN) *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 232–7

Johnson R C (ed) 1993a Siemens shows off its first neural network chip *Cognizer Report* **4** 9–11 (Frontline Strategies, 516 S E Chkalov, Drive Suite 164, Vancouver, WA 98684, USA)

——(ed) 1993b Intel/Nestor Announce delivery of chip to DARPA *Cognizer Report* **4** 17–19 (Frontline Strategies, 516 S E Chkalov, Drive Suite 164, Vancouver, WA 98684, USA)

Jones S R and Sammut K 1993 Learning in systolic neural network engines *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 175–85

Jones S R, Sammut K and Hunter J 1990 Toroidal neural network processor architecture operation performance *Proc. Int. Conf. on Microelectronics for Neural Networks (1990)* pp 163–9

Jones S R, Sammut K, Nielsen C and Staunstrup J 1991 Toroidal neural network processor architecture and processor granularity *VLSI Design of Neural Networks* ed U Ramacher and U Rückert (New York: Kluwer) pp 229–44

Judd J S 1988 On the complexity of loading shallow neural networks *J. Complexity* **4** 177–92

——1990 *Neural network design and the complexity of learning* (Cambridge, MA: MIT Press)

——1992 Constant-time loading of shallow 1-dimension networks *Proc. Conf. on Neural Information Processing Systems (1991)* pp 863–70

Kato H, Yoshizawa H, Iciki H and Asakawa K 1990 A parallel neurocomputer architecture toward billion connection updates per second *Proc. Int. Joint Conf. on Neural Networks (1990)* vol II pp 47–50

Kautz W 1961 The realization of symmetric switching functions with linear-input logical elements *IRE Trans. Electron. Computer* **10**

Kham E R and Ling N 1991 Systolic Architectures for artificial neural nets *Proc. Int. Joint Conf. on Neural Networks (1991)* vol 1 pp 620–7

Kim J H nad Park S-K 1995 The geometrical learning of binary artificial neural networks *IEEE Trans. Neural Networks* **6** 237–47

Klir G J 1972 *Introduction to the Methodology of Switching Circuits* (New York: Van Nostrand)

Kohn P, Bilmes J, Morgan N and Beck J 1992 Software for ANN training on a ring array processor *Proc. Conf. on Neural Information Processing Systems (1991)* pp 781–8

Koiran P 1993 On the complexity of approximating mappings using feedforward networks *Neural Networks* **6** 649–53

Kolmogorov A N 1957 On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition *Dokl Akad Nauk SSSR* **114** 679–81 (Engl. transl. 1963 *Math. Soc. Transl.* **28** 55–59

Köllmann K, Reimschneider K-R and Zeidler H C 1996 On-chip backpropagation training using parallel stochastic bit streams *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 149–56

Krauth W and Mézard M 1987 Learning algorithms with optimal stability in neural networks *J. Phys A: Math. Gen.* **20** L745–52

Krikelis A 1991 A novel massively parallel associative processing architecture for the implementation of artificial neural networks *Proc. Int. Conf. on Acoustics, Speech and Signal Processing (Toronto, 1991)* vol II (Los Alamitos, CA: IEEE Computer Society Press) pp 1057–60

Kuczewsk R, Meyers M and Crawford W 1988 Neurocomputer workstation and processors approaches and applications *Proc. Int. Joint Conf. on Neural Networks (1988)* vol III pp 487–500

Kung S Y 1989 *VLSI Array Processors (Prentice Hall Information and System Sciences Series)* (Englewood Cliffs, NJ: Prentice Hall)

Kung S Y and Hwang J-N 1988 Parallel architectures for artificial neural nets *Proc. Int. Joint Conf. on Neural Networks (1988)* vol II pp 165–72

——1989a Digital VLSI architectures for neural networks *Proc. IEEE Int. Symp. on Circuits and Systems (Portland, Oregon, 1989)* vol I (Los Alamitos, CA: IEEE Computer Society Press) pp 445–8

——1989b A unified systolic architecture for artificial neural networks *J. Parallel Distrib. Comput.* **6** 358–87

Kung H T and Webb J A 1985 Global operations on a systolic array machine *Proc. IEEE Int. Conf. on Computer Design VLSI in Computers (Port Chester, New York, 1985)* pp 165–71

Landahl H D, McCulloch W S and Pitts W 1943 A statistical consequence of the logical calculus of nervous system *Bull. Math. Biophysiology* **5** 135–7

Le Bouquin J-P 1994 IBM Microelectronics ZISC, zero instruction set computer *Proc. World Congr. on Neural Networks (San Diego, CA, 1994) (supplement)*

Le Cun Y 1985 A learning procedure for asymmetric threshold networks *Proc. Cognitiva '85* pp 599–604

——1987 Models connexionistes de l'apprentisage *MSc thesis* Université Pierre et Marie Curie, Paris

Lehmann C and Blayo F 1991 A VLSI Implementation of a generic systolic synaptic building block for Neural Networks *VLSI for Artificial Neural Networks* ed J G Delgado-Frias and W R Moore (New York: Plenum) pp 325–34

Lehmann C, Viredaz M and Blayo F 1993 A generic systolic array building block for neural networks with on-chip learning *IEEE Trans. Neural Networks* **4** 400–7

Leiserson C E 1982 *Area-Efficient VLSI Computation* (Cambridge, MA: MIT Press)

Leshno M, Lin V Y, Pinkus A and Schocken S 1993 Multilayer feedforward neural networks with a nonpolynomial activation function can approximate any function *Neural Networks* **6** 861–7

Lewis P M II and Coates C L 1967 *Threshold Logic* (New York: Wiley)

Léonhard G, Cousin E, Laisne J D, Le Drezen J, Ouvradou G, Poulain Maubant A and Thépaut A 1995 ArMenX: a flexible platform for signal and image processing *Field Programmable Gate Arrays (FPGAs) for Fast Board Development and Reconfigurable Computing* ed J Schewel vol 2607 (SPIE)

Linde A, Nordström T and Taveniku M 1992 Using FPGAs to implement a reconfigurable highly parallel processor *Proc. 2nd Int. Workshop on Field Programmable Logic and Applications (Vienna)*

Lindsey C S and Lindblad T 1994 Review of hardware neural networks: a user's perspective plenary talk given at the *Third Workshop on Neural Networks: From Biology to High Energy Physics (Isola d'Elba, Italy, 1994)* (see also the following two WWW sites: http://www1.cern.ch/NeuralNets/nnwInHep.html and also http://www1.cern.ch/NeuralNets/nnwInHepHard.html)

Linial N, Mansour Y and Nisan N 1989 Constant depth circuits Fourier transforms and learnability *Proc. IEEE Symp. on Foundations of Computer Science* p 30

Lippmann R P 1987 An introduction to computing with neural nets *IEEE ASSP Mag.* **4** 4–22

Littlestone N 1988 Learning quickly when irrelevant attributes abound a new linear-threshold algorithm *Machine Learning* **2** 285–318

Losleben P (ed) 1987 Advanced research in VLSI *Proc. Stanford Conf. on Advanced Research on VLSI* (Cambridge, MA: MIT Press)

Lupanov O B 1973 The synthesis of circuits from threshold elements *Problemy Kibernetiki* **20** 109–40

Maass W, Schnitger G and Sontag E 1991 On the computational power of sigmoid versus Boolean threshold circuits *IEEE Symp. on Foundation of Computer Science (1991)*

Mackie S, Graf H P, Schwartz D B and Denker J S 1988 Microelectronic implementations of connectionist neural networks *Proc. Conf. on Neural Information Processing Systems (1987)* pp 515–23

Makram-Ebeid S, Sirat J-A and Viala J-R 1989 A rationalized error back-propagation learning algorithm *Proc. Int. Joint Conf. on Neural Networks (1989)*

Mann J, Berger B, Raffel J, Soares A and Gilbert S 1987 A generic architecture for wafer-scale nuromorphic systems *Proc. Int. Joint Conf. on Neural Networks (1987)* vol IV pp 485–93

MasPar 1990a MasPar 1100 series computer systems *Information sheet PL003 0490* (MasPar Computer Corporation, 749 North Marry Avenue, Sunnyvale, CA 94086, USA)

——1990b MasPar 1200 series computer systems *Information sheet PL004 0490* (MasPar Computer Corporation, 749 North Marry Avenue, Sunnyvale, CA 94086, USA)

——1990c The MP-1 family data-parallel computer *Information sheet PL006 0490* (MasPar Computer Corporation, 749 North Marry Avenue, Sunnyvale, CA 94086, USA)

Maudit N, Duranton M, Gobert J and Sirat J A 1991 Building up neuromorphic machines with L-Neuro 1.0 *Proc. Int. Joint Conf. on Neural Networks (1991)* pp 602–7

——1992 L-Neuro 1.0: a piece of hardware LEGO for building neural network systems *IEEE Trans. Neural Networks* **3** 414–22

Mayoraz E 1991 On the power of networks of majority functions *Proc. Int. Workshop on Artificial Neural Networks (1991)* (Berlin: Springer) pp 78–85

——1992 Representation of Boolean functions with democratic networks *Internal Report* (ÉPFL, Lausanne)

McCator H 1991 Back propagation Implementation on the Adaptive Solution CNAPS neurocomputer chip *Proc. Conf. on Neural Information Processing Systems (1990)* pp 1028–31

McCulloch W S and Pitts W 1943 A logical calculus of the ideas immanent in nervous activity *Bull. Math. Biophysiol.* **5** 115–33

Means E and Hammerstrom D 1991 Piriform model execution on a neurocomputer *Proc. Int. Joint Conf. on Neural Networks (1991)* vol I pp 575–80

Means R W and Lisenbee L 1991 Extensible linear floating point SIMD neurocomputer array processor *Proc. Int. Joint Conf. on Neural Networks (1991)* vol I pp 587–92

Mézard M and Nadal J-P 1989 Learning in feedforward layered networks the tiling algorithm *J. Phys. A: Math. Gen.* **22** 2191–203

Micro Devices 1989a *Data Sheet MD1220* (Micro Devices 5695B Beggs Road, Orlando, FL 32810-2603, USA)

——1989b Neural bit slice *Data Sheet no DS102300P on circuit MD1200* (Micro Devices 5695B Beggs Road, Orland, FL 32810-2603, USA)

——1989c *Design Manual for the NBS Part No DM102500* (Micro Devices 5695B Beggs Road, Orland, FL 32810-2603, USA)

——1990 Neural bit slice *Data sheet no DS102301 on circuit MD120* (Micro Devices 5695B Beggs Road, Orland, FL 32810-2603, USA)

Milosavlevich I Z, Flower B G and Jabri M A 1996 PANNE: a parallel computing engine for connectionist simulation *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 363–8

Minnick R C 1961 Linear-Input Logic IRE *Trans. Electron. Comput.* **10** 6–16

Minsky M L 1954 Neural nets and the brain–model problem *PhD Dissertation* (Princeton, NJ: Princeton University Press)

Minsky M L and Papert S A 1969 *Perceptron: An Introduction to Computational Geometry* (Cambridge, MA: MIT Press)

Moerland P D and Fiesler E 1996 hardware-friendly algorithms for neural networks: an overview *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 117–24

Morgan N 1995 Programmable neurocomputing systems *The Handbook of Brain Theory and Neural Networks* ed M A Arbib (Cambridge, MA: MIT Press) pp 264–8

Morgan N, Beck J, Kohn P and Bilmes J 1993 Neurocomputing on the RAP *Parallel Digital Implementations of Neural Networks* ed K W Przytula and V K Prasanna (Englewood Cliffs, NJ: Prentice Hall)

Morgan N, Beck J, Kohn P, Bilmes J, Allman E and Beer J 1990 The RAP: a ring array processor for layered network calculations *Proc. IEEE Int. Conf. on Application Specific Array Processes* (Los Alamitos, CA: IEEE Computer Society Press) pp 296–308

——1992 The ring array processor (RAP) a multiprocessing peripheral for connectionist applications *J. Parallel Distrib. Comput.* **14** 248–59

Mühlbein H and Wolf K 1989 Neural network simultation on parallel computers *Parallel Computing (1989)* ed D J Evans, G G Joubert and F J Peters (Amsterdam: North-Holland) pp 365–74

Müller U A, Bäumle B, Kohler P, Gunzinger A and Guggenbühl W 1992 Achieving supercomputer performance for neural net simulation with an array of digital signal processors *IEEE Micro Mag.* **12** 55–65

Müller U A, Gunzinger A and Guggenbühl W 1995 Fast neural net simulation with a DSP processor array *IEEE Trans. Neural Networks* **6** 203–13

Müller U A, Kocheisen M and Gunzinger A 1994 High performance neural net simulation on a multiprocessor system with 'intelligent' communication *Proc. Conf. on Neural Information Processing Systems (1993)* pp 888–95

Muroga S 1959 The principle of majority decision logic elements and the complexity of their circuits *Proc. Int. Conf. on Information Processing (Paris)*

——1961 Functional forms of majority decision functions and a necessary and sufficient condition for their realizability In switching circuit theory and logical design *AIEE Special Publication S134* pp 39–46

——1962 Generation of self-dual threshold functions and lower bounds of the number of threshold functions and a maximum weight in switching circuit theory and logical design *AIEE Special Publication S134* pp 170–84

——1971 *Threshold Logic and Its Applications* (New York: Wiley)

——1979 *Logic Design and Switching Theory* (New York: Wiley) ch 5

Muroga S, Toda I and Takasu S 1961 *Theory of Majority Decision Elements Journal* vol 271 (Franklin Institute) pp 376–418

Murray M, Burr J B, Stork D G, Leung M-T, Boonyanit K, Wolff G J and Peterson A M 1992 Deterministic Boltzmann machine VLSI can be scaled using multi-chip modules *Proc. Int. Conf. on Application Specific Array Processors (Berkeley, CA)* (Los Alamitos, CA: IEEE Computer Society Press) pp 206–17

Murray M, Leung M-T, Boonyanit K, Kritayakirana K, Burr J B, Wolff G J, Watanabe T, Schwartz E and Stork D G 1994 Digital Boltzmann VLSI for constraint satisfaction and learning *Proc. Conf. on Neural Information Processing Systems (1993)* pp 896–903

Murre J M J 1993 Transputers and neural networks an analysis of implementation constraints and performance *IEEE Trans. Neural Networks* **4** 284–92

Murtagh P and Tsoi A C 1992 Implementation issues of sigmoid function and its derivative for VLSI digital neural networks *IEE Proc.-E Computer and Digital Techniques* **139** 207–14

Myers D J and Hutchinson R A 1989 Efficient implementation of piecewise linear activation function for digital VLSI neural networks *Electron. Lett.* **25** 1662–3

Myers D J, Vincent J M and Orrey D A 1991 HANNIBAL A VLSI building block for neural networks with on-chip backpropagation learning *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 171–81

——1993 HANNIBALL A VLSI building block for neural networks with on-chip backpropagation learning *Neurocomputing* **5** 25–37

Myhill J and Kautz W H 1961 On the size of weights required for linear-input switching functions *IRE Trans. Electron. Comput.* **10**

Nakayama K and Katayama H 1991 A low-bit learning algorithm for digital multilayer neural networks applied to pattern recognition *Proc. Int. Joint Conf. on Neural Networks (1991)* pp 1867–72

Naylor D, Jones S, Myers Dand Vincent J 1993 Design and application of a real-time neural network based image processing system *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 137–47

Neciporuk E I 1964 The synthesis of networks from threshold elements *Problemy Kibernetiki II* 49–62 (Engl. transl. 1964 *Automation Express* **7** 35–9 and **7** 27–32

Neibur E and Brettle D 1994 Efficient simulation of biological neural networks on massively parallel supercomputers with hypercube architecture *Proc. Conf. on Neural Information Processing Systems (1993)* pp 904–10

Neuralogix 1992 *NLX420 Data Sheet Neurologix Inc* (800 Charcot Avenue Suite, 112 San Jose, California)

Nickolls J R 1990 The design of the MasPar MP-1: a cost effective massively parallel computer *Proc. 35th IEEE Computer Society Int. Conf. Spring COMPCON '90 (San Francisco, CA)* pp 25–8

Nigri M E 1991 Hardware emulation of back-propagation neural networks *Research Note RN/91/21* (Department of Computer Science, University College London)

Nigri M E, Treleaven P and Vellasco M 1991 Silicon compilation of neural networks *CompEuro '91* ed Pröebster W E and Reiner H (Los Alamitos, CA: IEEE Computer Society Press) pp 541–6

Nijhuis J, Höfflinger B, Neußer S, Siggelkow A and Spaanenburg L 1991 A VLSI implementation of a neural car collision avoidance controller *Proc. Int. Joint Conf. on Neural Networks (1991)* vol 1 pp 493–9

Nilsson N J 1965 *Learning Machines* (New York: McGraw-Hill)

Nordström T and Svensson B 1991 Using and designing massively parallel computers for artificial neural networks *Technical Report TULEA 91:1* (Division of Computer Engineering LuleåUniversity of Technology S-95187 LuleåSweden) (also in *J. Parallel Distrib. Comput.* **14** 260–85)

Obradovic Z and Parberry I 1990 Analog neural networks of limited precision I: computing with multilinear threshold functions *Proc. Conf. on Neural Information Processing Systems (1989)* pp 702–9

Oliveira A L and Sangiovanni-Vincentelli A 1994 Learning complex Boolean functions algorithms and applications *Proc. Conf. on Neural Information Processing Systems (1993)* pp 911–8

Orrey D A, Myers D J and Vincent J M 1991 A high performance digital processor for implementing large artificial neural networks *Proc. IEEE Custom Integrated Circuits Conf. (San Diego, CA)*

Pacheco M and Treleaven P 1989 A VLSI word-slice architecture for neurocomputing *Proc. 1989 Int. Symp. on Computer Architecture and Digital Signal Processing (Hong Kong)* (IEEE)

——1992 Neural-RISC a processor and parallel architecture for neural networks *Proc. Int. Joint Conf. on Neural Networks (1992)* vol II pp 177–82

Palm G and Palm M 1991 Parallel associative networks the PAN-System and the BACCHUS-Chip *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 411–6

Papadopoulos C V and Andronikos T S 1995 Modelling the complexity of parallel and VLSI computations with Boolean circuits *Microprocess. Microsyst.* **19** 43–50

Parberry I 1994 *Circuit Complexity and Neural Networks* (Cambridge, MA: MIT Press)

Paterson M S (ed) 1992 Boolean function complexity *London Mathematical Society Lecture Notes Series* ***169*** (Cambridge: Cambridge University Press)

Paturi R and Saks M 1990 On threshold circuits for parity *Proc. IEEE Symp. on Foundation of Computer Science (1990)*

Pérez C J, Carrabina J and Valderrama E 1992 Study of a learning algorithm for neural networks with discrete synaptic couplings *Network* **3** 165–76

Personnaz L and Dreyfus G (eds) 1989 Neural networks from models to applications *Proc. nEuro '88 (Paris, 1988)* (Paris: IDSET)

Personnaz L, Johannet A and Dreyfus G 1989 Problems and trends in integrated neural networks *Connectionism in Perspective* eds R Pfeifer, Z Schreter and F Fogelman-Soulié (Amsterdam: Elsevier)

Pesulima E E, Pandya A S and Shankar R 1990 Digital implementation issues of stochastic neural networks *Proc. Int. Joint Conf. on Neural Networks (1990)* vol II pp 187–90

Pippenger N 1987 The complexity of computations by networks *IBM J. Res. Dev.* **31** 235–43

Pöchmüller W and Glesner M 1991 A cascadable architecture for the realization of large binary associative networks *VLSI for Artificial Intelligence and Neural Networks* ed J G Delgado-Frias and W R Moore (New York: Plenum) pp 265–74

Pomerleau D A, Gusciora G L, Touretzky D S and Kung H T 1988 Neural network simulation at warp speed how we got 17 million connections per second *Proc. Int. Joint Conf. on Neural Networks (1988)* (Los Alamitos, CA: IEEE Computer Society Press) vol II pp 143–50

Poulain Maubant A, Autret Y, Léonhard G, Ouvradoui G and Thépaut A 1996 An efficient handwritten digit recognition method on a flexible parallel architecture *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 355–62

Przytula K W 1988 A survey of VLSI implementations of artificial neural networks *VLSI Signal Processing III* ed R W Brodersen and H S Moscovitz (New York: IEEE Computer Society Press) pp 221–31

Przytula K W and Prasanna V K 1993 *Parallel Digital Implementations of Neural Networks* (Englewood Cliffs, NJ: Prentice Hall)

Raghavan P 1988 Learning in threshold networks: a computational model and applications *Technical Report RC 13859* (IBM Research July 1988) (also in 1988 *Proc. Workshop on Computational Learning Theory* (Cambridge, MA: Cambridge) pp 19–27

Ramacher U 1990 The VLSI Kernel of neural algorithms *Proc. 1st Int. Workshop on Cellular Neural Networks and their Applications (Budapest, 1990)* pp 185–96

——1992 SNAPSE–a neurocomputer that synthesizes neural algorithms on a parallel systolic engine *J. Parallel Distrib. Comput.* **14** 306–18

Ramacher U, Beichter J and Brüls N 1991a Architecture of a general-purpose neural signal processor *Proc. Int. Joint Conf. on Neural Networks (1991)* vol I pp 443–6

Ramacher U, Raab W, Anlauf J, Hachmann U, Beichter J, Brüls N, Weßeling M and Sicheneder E 1993 Multiprocessor and memory architecture of the neurocomputer SYNAPSE-1 *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 227–31

Ramacher U, Raab W, Anlauf J, Hachmann U and Weßeling M 1991b SYNAPSE-X a general-purpose neurocomputer *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 401–9 (also in *Proc. Int. Joint Conf. on Neural Networks (1991)* pp 2168–76

Ramacher U and Rückert U (eds) 1991 *VLSI Design of Neural Networks* (New York: Kluwer)

Razborov A A 1987 Lower bounds for the size of circuits of bounded depth with basis $\{\wedge, \oplus\}$ *Math. Not. Acad. Sci. USSR* **41** 333–8

Red'kin N P 1970 Synthesis of threshold circuits for certain classes of Boolean functions *Kibernetika* **5** 6–9 (Engl. transl. 1970 *Cybernetics* **6** 540–4)

Reilly D L, Cooper L N and Elbaum C 1982 A neural model for category learning *Biol. Cybern.* **45** 35–41

Reyneri L M and Filipi E 1991 An analysis on the performance of silicon implementations of backpropagation algorithms for artificial neural networks *IEEE Trans. Comput.* **40** 1380–9

Rief J H 1987 On threshold circuits and polynomial computations *Proc. 2nd Annual Structure in Complexity Theory Symp.* pp 118–23

Roberts F and Wang S 1989 Implementation of neural networks on a hypercube FPS T20 *Parallel Processing* ed M Cosnard M, M H Barton and M Vanneschi (Amsterdam: North-Holland) pp 189–200

Rosenblatt F 1958 The perceptron a probabilistic model for information storage and organization *Brain Psych. Revue* **62** 386–408

——1961 Principles of neurodynamics *Perceptrons and the Theory of Brain Mechanism* (Washington, DC: Sparton Press)

Rosenbleuth A, Wiener N and Bigelow J 1943 Behaviour, purpose and teleology *Phil. Sci.* **10** 18–24

Rosenbleuth A, Wiener N, Pitts W and Garcia Ramos J 1949 A statistical analysis of synaptic excitation *J. Cell Comput. Physiol.* **34** 173–205

Rossmann M, Hesse B, Goser K, Bühlmeier and Manteuffel G 1996 Implementation of a biologically inspired neuron-model in FPGA *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 322–9

Roth U, Jahnke A and Klar H 1995 Hardware requirements for spike-processing neural network 1995 *From Natural to Artificial Neural Computations Lecture Notes in Computer Science* vol 930 ed J Mira and F Sandoval (Berlin: Springer) pp 720–7

Roy A, Kim L S and Mukhopadhyay S 1993 A polynomial time algorithm for the construction and training of a class of multilayer perceptrons *Neural Networks* **6** 535–45

Roychowdhury V P, Orlitsky A and Siu K-Y 1994a Lower bounds on threshold and related circuits via communication complexity *IEEE Trans. Info. Theory* **40** 467–74

Roychowdhury V P, Siu K-Y and Orlitsky A (eds) 1994b *Theoretical Advances in Neural Computation and Learning* (Boston: Kluwer)

Roychowdhury V P, Siu K-Y, Orlitsky A and Kailath T 1991a A geometric approach to threshold circuit complexity *Proc. Workshop on Computational Learning Theory COLT (Santa Cruz, CA, 1991)* pp 97–111

——1991b On the circuit complexity of neural networks *Proc. Conf. on Neural Information Processing Systems (1990)* pp 953–59

Rückert U, Kleerbaum C and Goser K 1991 Digital VLSI implementations of an associative memory based on neural networks 1991 *VLSI for Artificial Intelligence and Neural Networks* ed J G Delgado-Frias and W R Moore (New York: Plenum) pp 275–84

Rudnick M and Hammerstrom D 1988 An interconnecting structure for wafer scale neurocomputers 1988 *Connectionist Models Summer School 1988 Proc.* ed D S Touretzky and G Hinton (San Mateo, CA: Morgan Kaufmann)

Rüping S and Rückert U 1996 A scalable processor array for self-organizing feature maps *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 285–91

Sanchez-Sinencio E and Lau C (eds) 1992 Artificial neural networks *Paradigms Applications and Hardware Implementations* (New York: IEEE Computer Society Press)

Saucier G and Ouali J 1990 Silicon compiler for neuron ASICs *Proc. Int. Joint Conf. on Neural Networks (1990)* vol II pp 557–61

Sakaue S, Kohda T, Yamamoto H, Maruno S and Shimeki Y 1993 Reduction of required precision bits for back-propagation applied to pattern recognition *IEEE Trans. Neural Networks* **4** 270–5

Sami M (ed) 1990 *Workshop on Silicon Architectures for Neural Nets (St Paul de Venice, France)* (Amsterdam: Elsevier)

Sammut K and Jones S R 1991 Implementing non-linear activation functions in neural network emulators *Electron. Lett.* **27** 1037–8

Savran M E and Morgül Ö 1991 On the associative memory design for the Hopfield neural network *Proc. Int. Joint Conf. on Neural Networks (1991)* vol II pp 1166–71

Schwartz T J 1990 *A Neural Chips Survey AI Expert* **5** 34–9

Scofield C L, Reilly D L 1991 Into silicon real time learning in a high density RBF neural network *Proc. Int. Joint Conf. on Neural Networks (1991)* vol I pp 551–6

Sejnowski T J and Rosenberg C R 1986 NETtalk A parallel network that learns to read aloud *Technical Report JHU/EECS-86/01* (Johns Hopkins University, Electrical Engineering and Computer Science, Baltimore)

Shawe-Taylor J S, Anthony M H G and Kern W 1992 Classes of feedforward neural networks and their circuit complexity *Neural Networks* **5** 971–7

Sheng C L 1969 *Threshold Logic* (New York: Academic)

Shoemaker P A, Carlin M J and Shimabukuro R L 1990 Back-Propagation learning with coarse quantization of weight updates *Proc. Int. Joint Conf. on Neural Networks (1990)* vol I pp 573–6

Siggelkow A, Nijhuis J, Neußer S and Spaanenburg L 1991 Influence of hardware characteristics on the performance of a neural system *Proc. Int. Conf. on Artificial Neural Networks (1991)* vol 1 pp 697–702

Singer A 1990a Exploiting the inherent parallelism of artificial neural networks to achieve 1300 million interconnects per second *Proc. INNC '90 (Paris)* pp 656–60

——1990b Implementations of artificial neural networks on the connection machine *Parallel Comput.* **14** 305–15

Sirat J A and Nadal J-P 1990 *Neural trees: a new tool for classification network: computation in neural systems* **1** 423–8

Siu K-Y 1992 On the complexity of neural networks with sigmoid units *Neural Networks for Signal Processing II. Proc. IEEE-SP Workshop on Neural Networks and Signal Processing (1992)* ed S Y Kung, F Fallside, J Aa Sorenson and C A Kamm (Helsingöer, Denmark) (Los Alamitos, CA: IEEE Computer Society Press) pp 23–28

Siu K-Y and Bruck J 1990a On the dynamic range of linear threshold elements *Research Report RJ 7237* (IBM, Yorktown Heights, New York)

——1990b Neural computation of arithmetic functions *Proc. IEEE* **78** 166–75

——1990c On the power of threshold circuits with small weights *Research Report RJ 7773 (71890)* (IBM, Yorktown Heights, New York) (see also *SIAM J. Discrete Math.* **4** 423–35 1991)

——1992 Neural computing with small weights *Proc. Conf. on Neural Information Processing Systems (1991)* pp 944–9

——1993 On the dynamic range of linear threshold elements *SIAM J. Discrete Math.* to appear

Siu K-Y, Bruck J and Kailath T 1991a Depth efficient neural networks for division and related problems *Research Report RJ 7946 (72929)* (IBM, Yorktown Heights, New York) (see also Siu 1993b)

Siu K-Y, Bruck J, Kailath T and Hofmeister T 1993a Depth-efficient neural networks for division and related problems *IEEE Trans. Info. Theory* **39** 946–56

Siu K-Y and Roychowdhury V P 1993 Optimal depth neural networks for multiplication and related problems *Proc. Conf. on Neural Information Processing Systems (1992)* pp 59–64

——1994 On optimal depth threshold circuits for multiplication and related problems *SIAM J. Discrete Math.* **7** 284–92

Siu K-Y, Roychowdhury V and Kailath T 1990 Computing with almost optimal size threshold circuits *Technical Report* (Information System Laboratory, Stanford University) (also in *Proc. IEEE Int. Symp. on Information Theory (Budapest, 1991)*)

——1991b Depth-size tradeoffs for neural computations *IEEE Trans. Comput.* **40** 1402–12

——1993b Computing with almost optimal size neural networks *Proc. Conf. on Neural Information Processing Systems (1992)* pp 19–26

——1994 *Discrete Neural Computation: A Theoretical Foundation* (Englewood Cliffs, NJ: Prentice-Hall)

Sivilotti M A, Emerling M R and Mead C A 1986 VLSI architectures for implementation of neural networks *Neural Networks for Computing* (New York: American Institute of Physics) pp 408–13

Smolensky R 1987 Algebraic methods in the theory of lower bounds for Boolean circuit complexity *Proc. ACM Symp. on Theory of Computing (1987)* vol 19 pp 77–82

Sontag E D 1990 On the recognition capabilities of feedforward nets *Report SYCON* (Rutgers Center for System and Control, 90-03 Department of Mathematics, Rutgers University, New Brunswick, NJ 08903, USA)

Souček B and Souček M 1988 *Neural and Massively Parallel Computers—the Sixth Generation* (New York: Wiley)

Spaanenburg L, Hoefflinger B, Neußer S, Nijhuis J A G and Siggelkow A 1991 A multiplier-less digital neural network *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 281–9

Specht D F 1988 Probabilistic neural networks for classification, mapping, or associative memory *Proc. Int. Joint Conf. on Neural Networks (1988)* vol I pp 525–32

Stevenson M, Winter R and Widrow B 1990 Sensitivity of feed-forward neural networks to weight errors *IEEE Trans. Neural Networks* **1** 71–80

Strey A, Avellana N, Hogado R, Fernández J A, Capillas R and Valderrama E 1995 A massively parallel neurocomputer with a reconfigurable arithmetical unit 1995 *From Natural to Artificial Neural Computations Lecture Notes in Computer Science* ed J Mira and F Sandoval vol 930 (Berlin: Springer) pp 800–6

Szedegy M 1989 Algebraic methods in lower bounds for computational models with limited communication *PhD Dissertation* University of Chicago

Śmieja F 1993 Neural network constructive algorithm trading generalization for learning efficiency? *Circuits, Syst. Signal Processing* **12** 331–74

Takahashi H, Tomita E and Kawabata T 1993 Separability of internal representations in multilayer perceptrons with application to learning *Neural Networks* **6** 689–703

Tan S and Vandewalle J 1992 Efficient algorithm for the design of multilayer feedforward neural networks *Proc. Int. Joint Conf. on Neural Networks (1992)* vol II pp 190–5

——1993 On the design of feedforward neural networks *Technical Report* (National University of Singapore, Department of EE) (also in *Neurocomputing* **6** 565–82)

Tewksbury S K and Hornak L A 1989 Wafer level system integration: a review *IEEE Circuits and Devices Mag.* **5** 22–30

Theeten J B, Duranton M, Maudit N and Sirat J A 1990 The L-Neuro chip: a digital VLSI with on-chip learning mechanism *Proc. INNC '90 (Paris)* ed B Angeniol and B Widrow (Dordrecht: Kluwer) pp 593–6

Thiran P 1993 Self-organization of a Kohonen network with quantized weights and an arbitrary one-dimensional stimuli distribution *Proc. Euro. Symp. on Artificial Neural Networks (Brussels)* ed M Verleysen (Brussels: de facto) pp 203–8

Thiran P, Peiris V, Heim P and Hochet B 1994 Quantization effects in digitally behaving circuit implementations of Kohonen networks *IEEE Trans. Neural Networks* **5** 450–8

Thole P, Speckmann H and Rosenstiel W 1993 A hardware supported system for Kohonen's self-organizing map *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 29–34

Treleaven P C 1989 Neurocomputers international *J. Neuro-computing* **1** 4–31

Treleaven P C, Pacheco M and Vellasco M 1989 VLSI architectures for neural networks *IEEE Micro Mag.* **9** 8–27

Treleaven P C and Rocha P V 1990 Towards a general-purpose neurocomputing system *Workshop on Silicon Architectures for Neural Nets (St Paul de Venice, France, 1990)* ed M Sami (Amsterdam: Elsevier)

Trotin A and Darbel N 1993 A neocognitron for digits classification on a VLSI chip *Proc. Int. Conf. on Microelectronics for Neural Networks (1993)* pp 21–8

Tryba V, Speckmann H and Goser K 1990 A digital hardware implementation of a self-organizing feature map as a neural coprocessor to a von Neumann computer *Proc. Int. Conf. on Microelectronics for Neural Networks (1990)* pp 177–86

van Keulan E, Colak S, Withagen H and Hegt H 1994 Neural network hardware performance criteria *Proc. IEEE Conf. on Neural Networks (1994)* vol III (Los Alamitos, CA: IEEE Computer Society Press) pp 1885–8

Vellasco M and Treleaven P C 1992 A VLSI architecture for the automatic generation of neuro-chips *Proc. Int. Joint Conf. on Neural Networks (1992)* vol II pp 171–6

Venkatesh S S 1989 A new linear threshold algorithm for learning binary weights *On-Line Workshop on Neural Network for Computing (Snowbird, Utah, 1989)*

Verleysen M and Cabestany J 1994 Project ESPRIT ELENA *Realisation VLSI de reseaux de neurones VLAGO, ISSN 1243-4835 No 94-1: Les processeurs neuronaux 1994*

Vincent J and Myers D 1992 Weight dithering and wordlength selection for digital backpropagation networks *BT Technology J.* **10** 124–33

Viredaz M A, Lehmann C, Blayo F and Ienne P 1992 MANTRA a multi-model neural network computer *Proc. 3rd Int. Workshop on VLSI for Neural Networks and Artificial Intelligence (Oxford)*

Walker M R and Akers L A 1992 Information-theoretic analysis of finite register effects in neural networks *Proc. Int. Joint Conf. on Neural Networks (1992)* vol II pp 666–71

Walker M R, Haghighi S, Afgan A and Akers L A 1989 Training a limited-interconnect synthetic neural IC *Proc. Conf. on Neural Information Processing Systems (1988)* pp 777–84

Watanabe T, Kimura K, Aoki M, Sakata T and Ito K 1993 A single 1 5-V digital chip for a 10^6-Synapse neural network *IEEE Trans. Neural Networks* **4** 387–93

Watkins S S, Chau P M and Tawel R 1992 Different approaches to implementing a radial basis function neurocomputer *Proc. RNNS/IEEE Symp. on Neuroinformatics and Neurocomputing (Rostov-on-Don, Russia)* pp 1149–55

Wawrzynek J, Asanović K, Kingsbury B, Beck J, Johnson D and Morgan N 1996 SPERT-II: a vector microprocessor system and its applications to large problems in backpropagation training *Proc. Int. Conf. on Microelectronics for Neural Networks (1996)* pp 227–31

Wawrzynek J, Asanović K and Morgan N 1993 The Design of a neuro-microprocessor *IEEE Trans. Neural Networks* **4** 394–9

Wegener I 1987 *The Complexity of Boolean Functions* (Chichester: Wiley)

Weinfeld M 1989 A fully digital integrated CMOS Hopfield network including the learning algorithm *VLSI for Artificial Intelligence* ed Delgado-Frias J G and Moore W R (Boston: Kluwer) pp 169–78

——1990 Integrated artificial neural networks components for higher level architectures with new properties *NATO Advance Workshop on Neurocomputing* ed Fogelman-Soulié F and Hérault J (Berlin: Springer)

White B and Elmasry M 1992 The digi-neocognitron: a digital neocognitron neural network model for VLSI *IEEE Trans. Neural Networks* **3** 73–85

Williams P and Panayotopoulos G 1989 Tools for neural network simulation *Report ANNR04 from ESPRIT project 2092 (ANNIE)*

Winder R O 1962 Threshold logic *PhD Dissertation* Mathematics Department, Princeton University, Princeton, NJ

——1963 Bounds on threshold gate realizability *IRE Trans. Electron. Comput.* **12** 561–4

——1969a *Fundamentals of Threshold Logic* AAT pp 235–318

——1969b The status of threshold logic *RCA Review* **30** 62–84
——1971 Chow parameters in threshold *J. ACM* **18** 265–89
Witbrock M and Zagha M 1990 An implementation of backpropagation learning on GF11 a large SIMD parallel computer *Parallel Comput.* **14** 329–46
Works G 1988 The creation of delta: a new concept in ANS processing *Proc. Int. Joint Conf. on Neural Networks (1988)* vol II pp 159–64
Xie Y and Jabri M A 1991 Analysis of the effect of quantization in multi-layer neural networks using statistical model *SEDAL Technical Report 1991-8-2* (Department of EE, University of Sydney, Australia)
——1992 Training algorithms for limited precision feedforward neural networks *SEDAL Technical Report 1991-8-3* (Department of EE, University of Sydney, Australia) (also in *Proc. Australian Conf. on Neural Networks (Canberra, Australia, 1992)* pp 68–71
Yao A C 1985 Separating the polynomial-time hierarchy by oracles *Proc. IEEE Symp. on Foundations Computer Science (1985)* vol 26 pp 1–10
——1989 On ACC and threshold circuits *Proc. ACM Symp. on Theory of Computing* pp 186–96
Yasunaga M, Masuda N, Asai M, Yamada T, Masaki A and Hirai Y 1989 A wafer scale integration neural network utilizing completely digital circuits *Proc. Int. Joint Conf. on Neural Networks (1989)* vol II pp 213–7
Yasunaga M, Masuda N, Yagyu M, Asai M, Yamada T and Masaki A 1990 Design fabrication and evaluation of a 5-inch wafer scale neural network LSI composed of 576 digital neurons *Proc. Int. Joint Conf. on Neural Networks (1990)* vol II pp 527–35
——1991 A self-learning neural net composed of 1152 digital neurons in wafer-scale LSIs *Proc. Int. Joint Conf. on Neural Networks (1991)* vol III pp 1844–9
Yestrebsky J, Basehore P and Reed J 1989 Neural bit-slice computing element information *Sheet No TP102600* (Micro Devices, 5695B Beggs Road, Orlando, FL, 32810-2603, USA)
Yoshizawa H, Ichiki H K H and Asakawa K 1991 A highly parallel architecture for back-propagation using ring-register data path *Proc. Int. Conf. on Microelectronics for Neural Networks (1991)* pp 325–32
Zhang X, Mckenna M, Mesirov J P and Waltz D L 1990 An Efficient Implementation of the back-propagation algorithm on the connection machine CM-2 *Technical Report RL-89-1* (Thinking Machines Corp., 245 First St., Cambridge, MA 02114, USA) (also in *Proc. Conf. on Neural Information Processing Systems (1989)* pp 801–9)
Zorat A 1987 Construction of a fault-tolerant grid of processors for wafer-scale integration *Circuits, Syst. Signal Processing* **6**
Zornetzer S F, Davis J L and Clifford L (eds) 1990 *An Introduction to Neural and Electronic Networks* (San Diego, CA: Academic)

