Digital Systems for Artificial Neural Networks

Les E. Atlas and Yoshitake Suzuki

This is the last of a trilogy of papers on implementation of artificial neural networks. The first two papers dealt with analog electronic and analog optronic implementation of neural networks. This final overview paper surveys the current art of digital electronic implementation of neural networks. Indeed, certain neural network paradigms, such as error back propagation training, require the accuracy available only from digital implementation. Atlas and Suzuki give a clear and complete compendium of this important mode of neural network implementation.

*Robert J. Marks II**

Abstract

A tremendous flurry of research activity has developed around artifical neural systems. These systems have also been tested in many applications, often with positive results. Most of this work has taken place as digital simulations on general-purpose serial or parallel digital computers. Specialized neural network emulation systems have also been developed for more efficient learning and use. Dedicated digital VLSI integrated circuits offer the highest near-term future potential for this technology.

Introduction

Two recent publications have inspired much of the research community in engineering, computer science, cognitive sciences and physics and biophysics to take on a new direction in information processing and modeling. These two papers, the first about the Hopfield model [1] and the second about back-propagation [2] suggest approaches to pattern recognition that were trained by examples and based on relatively large networks of neuron-like processors. Even though this notion of collective computation had been heavily discussed in the past (Minsky and Papert [3] have a good review) this most recent incarnation coincided with readily available and inexpensive computing power. Thus, any researcher with computer access can quickly run experiments that involve at least millions of multiplications and additions. This research is now progressing to the point where personal computers, workstations, mainframe computers, and, in some cases, even supercomputers are inadequate. As has been discussed by past papers in this magazine, analog electronic [4] and optical [5] techniques offer huge potential for these artificial neural systems. This paper will concentrate on the more near-term digital solutions and will show some projected limits of different digital technologies.

Some reasons that artificial neural system simulations are computationally intensive are:

1. *Massive interconnection:* Most of the architectures used involve tens or hundreds of neuron-like units where all units can be connected to each other. Each connection usually requires a multiplication and each unit can require a sum of hundreds (or more) inputs.
2. *Learning:* Many of the problems studied with artificial neural systems involve large data sets. The learning algorithms, which can adjust the weights for the multiplies in the interconnections, have very slow convergence. Thus, many iterations are required where each iteration involves a considerable size set of data.
3. *Flexibility:* Algorithms and architectures for artificial neural systems are continuously evolving, and both researchers and users require the ability to change the simulations.
4. *Trial and error:* Many of the artificial neural system algorithms do not guarantee convergence at a global minimum. This characteristic can sometimes be reduced by repeating training runs with different initial random weights. The weights from the training run with the lowest final error rate are then used in the chosen network.

This paper will summarize several techniques for digital implementation of neural networks. We will concentrate on trainable architectures and report other researchers' recent results. The work we report is intended to be representative and not exhaustive and we apologize to any researchers whose work is not reported in our summary.

General-Purpose Parallel Computers for Neural Network Simulations

General-purpose parallel computers (as distinct from vector-oriented supercomputers) are composed of a large number of processors cooperating on the same task. Each processor has independent memory and data paths and instructions for all processors can be independent for MIMD (Multiple Instruction Multiple Data). The connections between processors can either be through a single high speed data path or via short point-to-point links between proces-

*R.J. Marks II is the current Chairman of the IEEE Neural Networks Committee. He is also co-founder and first Chairman of the IEEE Circuits & Systems Society Technical Committee on Neural Systems and Applications. He is Professor of Electrical Engineering, Seattle. Professor Marks was recently asked by *Circuits and Devices Magazine* to be the guest editor for a series of three articles on neural networks.

sors. Many parallel computers are now commercially available and are currently used for many simulation applications.

The interconnection needs for ANNs pose a special challenge for parallel processors. Another difficulty is the inconsistency of the need for flexibility and the difficulty of efficiently programming parallel processors. We will describe two studies that adapted parallel computers to ANN simulations. The first study, by Forrest et al. [6], made separate use of a Distributed Array Processor (DAP) [7] and a MIMD array of transputers. [8] The second study, by Pomerleau et al. [9], made use of the Warp machine, which is a systolic array of processors. [10]

Forrest et al. applied a DAP, which is a 2-D grid of 4096 processors, to a Hopfield net [1] and to a distributed image restoration algorithm. [11] For the Hopfield net, it was found that the DAP could perform 25 million conditioned adds per second. The image restoration algorithm was able to perform 100 iteration updates per second for a 64×64 image on the DAP. There is no comparison made to supercomputer or serial computer implementations, but the authors conclude "It is our view that the software effort expended in the first place to implement these simulations on the hardware described is well justified by the increase gained in performance; in fact, in some cases it is clear that the use of these parallel machines was essential for the simulations to be done at all in a feasible amount of time."

The Warp machine is quite different from the previously discussed parallel computers. The architecture used by Pomerleau et al.'s study [9] was based on a systolic array of 10 cells. Each cell consists of an adder, multiplier, and ALU. Communication is possible at high bandwidth with a cell's left and right neighbors. Programs for the Warp machine are written in a Pascal-like language called W2 and an optimizing compiler gives high efficiency in execution time.

The ANN algorithm that was simulated by the Warp machine was back-propagation. [2] The researchers initially partitioned the neurons into different processor cells. They later found that this partitioning scheme became troublesome for large ANNs. In particular, the size of the cell memories of the Warp machine limited the number of interconnect weights and hence the size of the network. Pomerleau et al. then devised a data partitioning scheme that divided the training data between the cells. Their technique allowed weights to be stored in the 39 Mbyte cluster memory and weight changes to be propagated at high speed between processor cells.

The Warp machine ANN was able to compute approximately 17 million connection updates per second for the training of a large back-propagation network. The authors of the Warp machine study also compared their systems performance to Convex C-1 and 16K Connection machine ANN simulators and found speed advantages of a factor of 9.4 and 6.5, respectively.

Special Purpose Processors for Neural Network Simulations (Neurocomputers)

The name "neurocomputer" has been applied to special boards or other attached systems for high speed ANN simulations. Several companies, such as Hecht-Nielsen Neurocomputers, Science Applications International Corporation, and TRW, have products which are based upon their own designs (some are proprietary) of boards and systems. Many of these boards utilize combinations of general-purpose microprocessors and/or digital signal processing integrated circuits. Other more research level ideas also show promise for special-purpose ANN systems. In particular, Bell Labs' Graph Search Machine and INMOS's transputer integrated circuit have been proposed and designed into ANN systems.

A transputer system was used by Feild and Navlakha to implement a Hopfield network. [12] This system consisted of two INMOS boards connected to an IBM PC/XT which acted as a host. These two boards contained a total of five transputer chips and the system could easily be expanded for more parallelism. The authors did not report on the speed of their simulation, but they did provide descriptions of the software for their parallel system.

A back-propagation model was implemented on a larger network of transputers by Beynon. [13] This study made use of 40 transputers, each with 2 MBytes of dedicated memory, and compared the training speed with a single Sun-3 workstation. In all cases the transputer array was faster, and it is most notable that the transputer had the best relative performance (about 13 times faster than the workstation) when the number of neural network weights was the largest (51,200 interconnection weights). The author attributed this effect to the high relative communication overhead for the smaller networks. It was also found that graph theory provided useful techniques for minimizing the longest software communications path length between transputers [14], thereby reducing communication overhead. Beynon concluded that while transputers are not the best parallel systems for the global communications found in fully interconnected ANNs, the arrays provide a good test bed for simulation.

Another specialized integrated circuit is the Graph Search Machine (GSM) developed at Bell Labs. [15] This VLSI circuit is a reduced-instruction set architecture that is specially optimized for pattern matching. The chip also has a 32 word instruction cache, thus allowing for fastest execution of short, modular programs. Na and Glinski made use of a single GSM processor for Hopfield's ANN and found considerable advantages over a mainframe computer system. [16] For training, recognition, and control, seven short programs were needed. The authors predicted that after training, the GSM processor could recognize one image of 234 pixels every 0.45 seconds. This was approximately 25 times faster than their mainframe (the type was not specified) simulation. GSM processors can also be connected together in arrays allowing for faster simulations of larger networks.

Many ANN operations consist of sums of products and are quite similar to some filtering operations in digital signal processing. This similarity suggests that much of the DSP (digital signal processor) technology could be applied to accelerate ANN operations. Researchers from Texas Instruments have applied their TMS32020 DSP to the recall of a 256 component vector. [17] An inner product operation was 2½ times faster on the TMS32020 than on a Digital Equipment Corp. VAX 8600. One advantage of DSP systems is that many of the chips can be built into a system. The Texas Instruments researchers also designed a mapping scheme for multiple DSPs. For matrix-vector calculations of size $N \times N$, $(N/256)^2$ TMS32020 DSPs can be used to achieve large speed improvements relative to more conventional serial machines. For example, a 1000×1000 matrix-vector multiply would require 16 DSPs, effecting a speed gain of 40 times the speed of a VAX 8600. The TMS32020 is a fixed-point processor, hence some of the ANN systems could be difficult to develop on this architecture. However, very fast floating-point DSPs, such as Texas Instrument's TMS320C30 and AT&T's DSP32C, are now becoming available.

Several manufacturers have designed and developed board-level or larger systems for ANN simulations. Three of the companies that have been most visible are TRW, Science Applications International Corporation, and Hecht-Neilsen Neurocomputers. All of these companies sell boards and software systems for VME- or PC-based host computers.

The TRW products include a Mark III and a Mark IV neurocomputer. [18] The Mark III system consists of up to 15 slave processors operating on a single VME bus. Each processor module consists of a Motorola 68020 microprocessor with a 68881 floating point co-processor. Enough memory is provided in each module to store a significant portion of the interconnect weights, thus minimizing communication on the single VME bus. The Mark III can process up to 450,000 interconnections per second. TRW's Mark IV system uses dedicated hardware for an even higher speed of 5,000,000 interconnections per second. Both of these systems make use of a virtual PE concept where, at any one time, the computer physically contains only a subset of the ANN model. Other neurons are "swapped in" as processing progresses, analogous to the use of physical memory in virtual memory computers. TRW's virtual PE concept allows the simulation of very large ANNs. For example, the Mark IV can support an ANN with 256,000 neurons.

The ANN system developed by Science Applications International Corporation (SAIC) is called the Delta Floating Point Processor. This system consists of a set of boards (and software) that interface to an IBM PC. SAIC's design approach was described by Works. [19] The designers decided that they required floating-point operations, but they ruled out commercially available array processors since the memory and speed were deemed inadequate for their projected applications. The system that was designed made use of very fast (35 nsec) static column mode memories, an ECL floating-point chip set (from Bipolar Integrated Technologies in Oregon), and a reduced instruction set computer (RISC) architecture. Since the SAIC designers were interested in simulating many ANN paradigms, the RISC architecture was found to provide a good compromise between efficiency and flexibility. The speeds claimed for the Delta Processor are 2,000,000 connections per second during learning and 10,000,000 connections per second when weights are not updated. It is notable that the Delta Processor achieved this speed with no parallel processing—a single fast special purpose processor was used.

The last commercially available neurocomputer we describe is the neurocomputing co-processor from Hecht-Nielsen Neurocomputers. There is a circuit card that is plug-in compatible with a PC-AT (the ANZA Plus) and another card (the ANZA Plus/VME) which is configured for a VME bus. Both of these systems have similar hardware and specifications. The architecture of these boards is based on a 4-stage pipelined Harvard architecture where data and instruction paths are kept separate for efficiency. The processor used is the Weitek XL floating-point chip set. For both the VME and the PC-AT version, 1,800,000 interconnections per second are claimed during learning iterations. For non-learning mode (where the weights are not updated) 6,000,000 sustained and 10,000,000 peak interconnections per second can be calculated. Hecht-Nielsen Neurocomputers also distributes ANN development software for use in conjunction with these boards.

Dedicated Digital ANN VLSI Circuits

There has been much work in the design of VLSI ICs that are specially designed for ANNs. Many of these systems are analog or hybrid analog/digital and have been covered elsewhere. [20] We will thus stress systems that are solely digital. Our descriptions will start with some recent publications of other researchers and will finish with some results of our own research.

Rasure et al. [21] at the Department of Electrical and Computer Engineering at the University of New Mexico designed a VLSI-based 3-layer feed forward ANN. This network is intended to classify handwritten numerals and consisted of 50 neurons and 6688 interconnections. The training for this network is done off-line, i.e., the interconnection weights are not determined by the VLSI system but are instead kept fixed. The VLSI layout (using a 2-micron CMOS process) was found to occupy a 7900 by 9200 micron die. The chip simulation results predict that a new input could be classified every 0.4 milliseconds.

Another system that is based on custom digital VLSI designs was described by Garth at Texas Instruments in Bedford, UK. [22] This system is intended to accelerate training of neural networks. The author takes the approach that there are several key aspects of a trainable simulator: 1) a very large address space, 2) a small number of needed instructions, 3) the pipelining of repetitive operations, and 4) adequacy of relatively slow memory. He proposes a 3-D mesh of "NETSIM" cards, each of which contains communications, control, memory, and a specialized custom co-processor. This co-processor takes on the bulk of the computational load and consists of a math processor, address controller, and a memory controller. The calculations are based on 16-bit interconnection weights with a 24-bit accumulator. The author projects that a system consisting

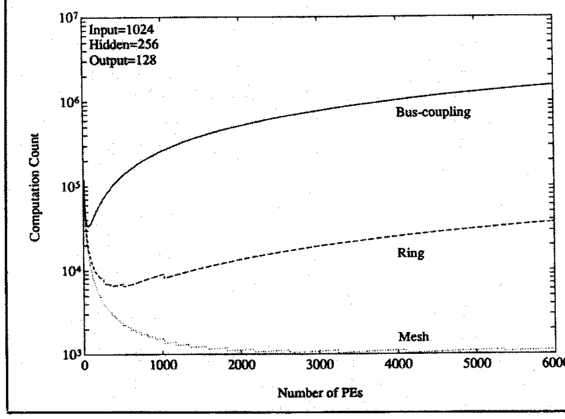

Fig. 1. A comparison of computation count for three different parallel topologies.

of 125 NETSIM cards would operate at 90,000,000 interconnections per second during learning.

Suzuki and Atlas have recently completed a study in which they determined a mapping of an ANN to an array of custom processors. [23,24] This mapping was optimized for the training phase of the back propagation algorithm. In order to find a minimum number of transmissions among processor elements (PEs), several mapping schemes from NN units to PEs were considered. We compared bus-coupling, ring, and mesh topologies, theoretically analyzing the required data transmission count and calculation count for one iteration of training for an ANN with one hidden layer. Our equation for the total computation count (sum of the needed data transmission cycles and the calculation cycles) was given as a function of the number of neural units in each layer and the total number of PEs. For the data transmission count an optimal number of PEs exists in the case of the mesh, whereas this count increases monotonically in the case of the bus-coupling and the ring. The calculation count decreases as the number of PEs increases for all three topologies.

A comparison of computation counts for one full NN training update is shown in Fig. 1. This count gives an indication of the total number of machine cycles for a single ANN learning iteration. For an ANN with 1024 input, 256 hidden, and 128 output units, a computation count of about 1020 is obtained by the mesh with 4096 PEs. This computation count is about 16 percent of that seen for the minimum case of the ring which consists of 512 PEs, and about three percent of the best bus-coupling result which is obtained with 64 PEs. A similar result can be obtained for an ANN with two hidden layers. An important point is that the lowest computation count occurs with many more PEs for the case of the mesh. This means that a much finer grain regular processor system can be realized by using the mesh topology.

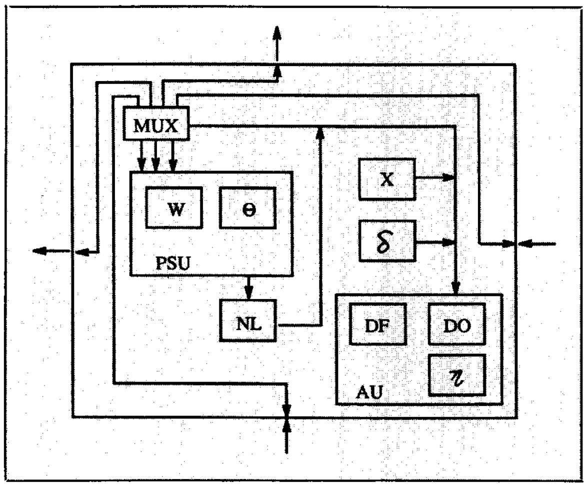

Fig. 2. The structure for one processing element. The blocks represent special operations (as described in the text) of the artificial neural net update equations.

An example of a proposed processing element structure is shown in Fig. 2. This processing element contains two calculation units. One is the product-sum unit (PSU) which calculates and accumulates part of a neuron's inputs. Another is the arithmetic unit (AU) where almost all the other calculations are performed. A partial matrix of weights W and a partial vector of the threshold θ for each layer are stored in the PSU. A nonlinear table (NL) is placed to perform sigmoidal or arbitrary nonlinearities. In the AU there exist memories for the back propagated derivative of nonlinear function (DF), desired outputs (DO), and a coefficient for weight adaption (η). Memories for input data and output values of neural units in each layer (X) and a memory for the error value δ are attached to the internal bus so they can be accessed easily by both the PSU and the AU. For smooth data transmission the multiplexer (MUX) is placed between external links and internal bus.

TABLE I.

A comparison of speed for several digital artificial neural network architectures. All speeds are in interconnects per second during training of a back propagation ANN.

ANN Architecture	Learned Connections Per Second
Warp machine [9]	1.7×10^{7}
TRW Mark III [18]	4.5×10^{5}
TRW Mark IV [18]	5.0×10^{6}
SAIC Delta [19]	2.0×10^{6}
HNC ANZA Plus	1.8×10^{6}
NETSIM [22]	9.0×10^{7}*
Pipelined Mesh [23]	6.9×10^{11}*

*These figures are projected from analysis or simulation.

Conclusions

ANN design and development is heavily dependent on appropriate computational tools. Many of the researchers who are investigating ANNs for real-world applications are faced with needs that go beyond conventional computing systems. In order to compare the available digital ANN systems we have put together Table 1. This table, which is by no means complete, lists the expected speed of the system during learning for a back propagation algorithm. Since the total learning time is problem-dependent, these figures are only for comparison. Also note that important issues such as flexibility, word size, and cost are not included and that the fastest architectures could be difficult to adapt to new algorithmic developments in ANNs.

We conclude that specialized digital systems are advantageous for ANN simulations. It is also apparent that the utmost in speed will require custom and dedicated digital integrated circuits. This paper has reviewed some recent contributions to this rapidly expanding area and we would expect that many more companies, large and small, are already developing systems that will fit advantageously into Table 1.

Acknowledgments

Les Atlas was supported by the National Science Foundation Center. The authors would also like to acknowledge the support of Drs. Kawashima and Nakatsu at NTT Human Interface Laboratories.

References

[1] J. J. Hopfield, "Neural Networks and Physical Systems with Emergent Computational Abilities," *Proc. of the National Academy of Sciences*, vol. 79, pp. 2554–2558, 1982.

[2] D. E. Rumelhart, G. F. Hinton, and R. J. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, pp. 533–536, 1986.

[3] M. L. Minsky and S. A. Papert, *Perceptrons* (2nd ed.), Cambridge, MA: MIT Press, 1988.

[4] H. P. Graf and L. D. Jackel, "Analog Electronic Neural Network Circuits," *IEEE Circuits and Devices Magazine*, vol. 5, pp. 44–49, 1989.

[5] N. H. Farhat, "Optoelectronic Neural Networks and Learning Machines," *IEEE Circuits and Devices Magazine*, vol. 5, pp. 32–41, 1989.

[6] B. M. Forrest, D. Roweth, N. Stround, D. J. Wallace, and G. V. Wilson, "Implementing Neural Network Models on Parallel Computers," *Computer Journal*, vol. 30(5), pp. 413–419, 1987.

[7] S. F. Reddaway, "DAP-A Distributed Array Processor," *Proc. 1st Annual Symp. on Computer Architecture (IEEE/ACM)*, Florida, pp. 61–65, 1973.

[8] K. C. Bowler, R. D. Kenway, G. S. Pawley, and D. Roweth, "An Introduction to Occam 2 and the Meiko Computing Surface," Physics Department, University of Edinburgh, 1987.

[9] D. A. Pomerleau, G. L. Gusciora, D. S. Touretzky, H. T. Kung, "Neural Network Simulation at Warp Speed: How We Got 17 Million Connections Per Second," *IEEE Int. Conf. on Neural Networks*, II-143–II-150, 1988.

[10] Annartone, E. Arnould, T. Gross, H. T. Kung, M. Lam, O. Menzilcioglu, and J. A. Webb, "The Warp Computer: Architecture, Implementation and Performance," *IEEE Trans. on Computers*, pp. 1523–1538, December, 1987.

[11] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images," *IEEE Trans. PAMI* vol. 5, pp. 721–741, 1984.

[12] W. B. Feild and J. K. Navlakha, "Transputer Implementation of Hopfield Neural Network," *IEEE Int. Conf. on Neural Networks* (poster session), 1988.

[13] T. Beynon, "A Parallel Implementation of the Back-Propagation Algorithm on a Network of Transputers," *IEEE Int. Conf. on Neural Networks* (poster session), 1988.

[14] J. Bermond, C. Delorme, and J. Quisquater, "Strategies for Interconnection Networks: Some Methods from Graph Theory," *J. Parallel and Distributed Computing*, vol. 3, pp. 433–449, 1986.

[15] S. Glinski, T. Lalumia, D. Cassiday, T. Koh, C. Gerveshi, G. Wilson, and J. Kumar, "The Graph Search Machine: A VLSI Architecture for Connected Word Speech Recognition and Other Applications," *Proc. IEEE*, vol. 75, pp. 1172–1184, 1987.

[16] H. Na and S. Glinski, "Neural Net Based Pattern Recognition on the Graph Search Machine," *Proc. IEEE Int. Conf. Acoust., Speech, Sig. Proc.*, pp. 2168–2171, 1988.

[17] P. A. Penz and R. Wiggins, "Digital Signal Processor Accelerators for Neural Network Simulations," in J. S. Denker (ed.), *Neural Networks for Computing*, American Institute of Physics, New York, 1986.

[18] R. Kuczewsk, M. Myers, and W. Crawford, "Neurocomputer Workstations and Processors: Approaches and Applications," *IEEE Int. Conf. on Neural Networks*, III-487–III-500, 1988.

[19] G. Works, "The Creation of Delta: A New Concept in ANS Processing," *IEEE Int. Conf. on Neural Networks*, II-159–II-164, 1988.

[20] H. Graf, L. Jackel, and W. Hubbard, "VLSI Implementation of a Neural Network Model," *IEEE Computer*, pp. 41–49, March 1988.

[21] J. Rasure, D. Hush, J. Salas, and M. Newell, "A VLSI Three Layer Artificial Neural Network for Binary Image Classification," *Proceedings of the International Neural Network Society* (poster session), 1988.

[22] S. Garth, "A Chipset for High Speed Simulation of Neural Network Systems," *IEEE Int. Conf. on Neural Networks*, III–443–III-452, 1987.

[23] Y. Suzuki and L. Atlas, "A Study of Regular Architectures for Digital Implementation of Neural Networks, "*Proc. IEEE Int. Symposium on Circuits and Systems*, Portland, May 9-11, 1989.

[24] Y. Suzuki and L. Atlas, "A Comparison of Processor Topologies for a Fast Trainable Neural Network for Speech Recognition," *Proc. IEEE Int. Conf. on Acoust., Speech, and Sign. Proc.*, Glasgow, May 23-26, 1989.

Yoshitake Suzuki received the B.E. and M.E. degrees from Waseda University, Tokyo, Japan. He joined Nippon Telegraph and Telephone Corp. in 1981 and is currently a senior research engineer at NTT Human Interface Laboratories. He is currently working to design architectures for speech recognition systems.

Mr. Suzuki was a visiting scholar at the University of Washington in 1988, where his study included designing digital architectures for artificial neural networks.

Les E. Atlas (Member, IEEE) received his B.S.E.E. degree from the University of Wisconsin and his M.S. and Ph.D. degrees from Stanford University.

He joined the University of Washington in 1984 and is currently an Associate Professor of Electrical Engineering. He co-founded the Interactive Systems Design Laboratory at the University of Washington and he is currently doing research in speech processing and recognition, neural network classifiers, and biologically-inspired signal processing algorithms and architectures. His research in these areas is funded by the National Science Foundation, the Office of Naval Research, and the Washington Technology Center.

Dr. Atlas was a 1985 recipient of a National Science Foundation's Presidential Young Investigator Award.