ANNSiS: A Circuit Level Simulator for Analog Neural Networks

İsmet Bayraktaroğlu
Electrical & Electronic Dept.
Boğaziçi University,
TR 80815, İstanbul
bayraki@boun.edu.tr

Sina Balkır
Electrical & Electronic Dept.
Boğaziçi University
TR 80815, İstanbul
balkir@boun.edu.tr

Günhan Dündar
Electrical & Electronic Dept.
Boğaziçi University
TR 80815, İstanbul
dundar@boun.edu.tr

Abstract. A tool for the simulation of analog neural networks based on circuit partitioning techniques is realized. The simulator is tested with different analog multilayer perceptron neural network structures and the results are compared with that of SPICE2G6 for accuracy and speed.

1. Introduction

Analog implementations of neural networks have many advantages such as small size and high speed. Synapses, which are the most common elements in a neural network, can be represented at the circuit level by multipliers. Parallel digital multipliers of 8x8 input word lengths have transistor counts on the order of at least several thousand [1]. Analog multipliers of comparable precision use less than 20 transistors. The speed of an analog multiplier is limited only by its bandwidth and can go up to the GHz range. When one looks at neurons, a similar picture can be seen. Again, an adder and the nonlinearity can be realized by less than 20 transistors, whereas the same operations require thousands of transistors in the digital domain [2]-[5].

However, analog neural network implementations have been rather *ad hoc* in that very few, if any, have explored the constraints that circuit non-idealities — like nonlinear synapses [6,7], neurons that deviate from ideal functions, or errors and limitations in storing weights [8] — bring about. It has been shown in [6] and [7] that multiplier nonlinearity can be a very severe problem for nonlinearity factors of less than 10% for many applications. This problem can be avoided by the use of sufficiently linear multipliers; nevertheless, their big size defeats the purpose of using analog neural networks. Limited precision in storing weights has also proven to be a crucial problem in analog neural network design. The work in this area has been mostly limited to predicting these effects either through simulation or through theoretical analysis and developing some methods to overcome these problems partially. In [7], the effects of some non-idealities have been studied through circuit simulation with SPICE and the importance of circuit level simulation in analog neural network design has been shown.

This work was supported by TÜBITAK under grant number EEEAG-183.

SPICE is actually not a very suitable environment to perform circuit level simulation of neural networks due to several reasons. One of these reasons is the sheer size of practical neural network circuits. The simulation time of SPICE on such large circuits can be extremely long and may make such simulations impractical. Furthermore, neural network simulations rarely require any other simulation type than DC simulation so that using SPICE may be a "luxury" in terms of simulation time and memory requirements. Therefore, a circuit level simulator specifically designed for analog neural networks will facilitate the design of this type of circuits greatly. In this paper, we present novel methodologies for simulating analog multilayer neural networks.

2. Simulating Neural Networks

In Figure 1, the structure of a multilayer neural network is shown. The circles denote neurons whereas synapses are represented by the interconnections between the neurons. The circles in the input layer are just input nodes and do not perform a neuron function. This structure is a regular structure where every neuron is connected to every other neuron in the previous layer through synapses. Therefore, it yields itself easily to partitioning and automatic netlist generation.

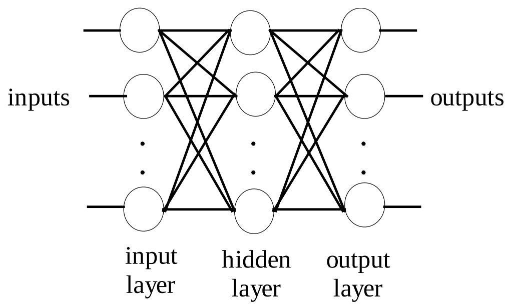

Figure 1. General structure of a multilayer neural network

If the speed of the analog synapses and neurons are known, the speed of the overall circuit can be estimated to a good degree of approximation. This means that for most applications, the DC transfer characteristics of the neural networks will be the important issue for the designer. Therefore, DC simulation will be sufficient for most cases. A simulator designed specifically for this purpose can be applicable to analog neural networks. For DC analysis, if the layers are completely decoupled, by which we mean that the outputs of the neurons are not loaded by the inputs of the synapses of the next layer, the circuit can be partitioned into decoupled blocks which can be simulated separately starting from the input layer.

Having the netlists of the synapse and neuron blocks, the netlist of the whole circuit can easily be generated automatically. Circuits created this way could be simulated with SPICE. The simulation time for the whole circuit is very long and depends quadratically on the circuit size as will be shown later. However, if the circuit is partitioned into decoupled blocks, the simulation time will vary linearly with the number of blocks and hence with the circuit size. Also, simulation for partitioned circuits is easily parallelizable. Netlists for partitioned circuits can also be created automatically and these can also be simulated with SPICE; however, feeding the results of a layer to the next layer should be done manually.

3. Description of the Simulator ANNSIS

The circuit level Analog Neural Network Simulation System (ANNSiS) written in C is a software package that is suitable for circuit level analog neural network simulations. ANNSiS has a circuit simulator based on the data structures of SPICE version 2G6. It has provisions for incorporating training into the circuit level simulation.

SPICE2G6 written in FORTRAN uses modified nodal analysis for the solution of node voltages, and uses LU factorization for the solution of modified node equations [9]. SPICE is a general purpose program for circuit simulation hence, it is not fully optimized for DC operating point solutions. We are only interested in finding the DC solution of the circuits. Therefore, we optimized SPICE for DC solutions and improved the speed of the simulator.

Solution of a circuit with nonlinear elements requires the solution of *N* (number of nodes in the circuit) nonlinear equations simultaneously. One of the well known algorithms for the solution of nonlinear equation is the Newton-Raphson algorithm, which is an iterative algorithm and requires linearization and simultaneous solution of *N* equations at every iteration [10].

Newton-Raphson algorithm for the solution of a set of nonlinear equations is as follows. Start for an arbitrary $x_{o,}$ which is a blind estimate of the solution, and find x_1 using equation (1), and then find $x_{2,}$ x_3, ... until x converges to the solution. Newton-Raphson algorithm converges quadratically in most cases, and given as

$$x_{n+1} = x_n - \left[J(x_n) \right]^{-1} f(x_n) \quad \text{where } J(x) = \begin{bmatrix} \dfrac{\partial f_1(x)}{\partial x_1} & \dfrac{\partial f_1(x)}{\partial x_2} & \cdots & \dfrac{\partial f_1(x)}{\partial x_n} \\ \vdots & & & \\ \dfrac{\partial f_n(x)}{\partial x_1} & \dfrac{\partial f_n(x)}{\partial x_2} & \cdots & \dfrac{\partial f_n(x)}{\partial x_n} \end{bmatrix} \text{ and}$$

f(x) is the set of nonlinear circuit equations.

Solution of *N* equations requires $N^3/3$ multiplications for large *N* if all coefficients of the equations are assumed to be nonzero, which means that the equation matrix is dense. With large circuits, solution of the equation with dense matrix is extremely time consuming job, if not impossible. SPICE2G6 employs sparse matrix techniques in which only non-zero coefficients are stored and LU factorization for the solution linearized of equations. With sparse matrix techniques, the time complexity is remarkably decreased. The circuit matrices for large circuit are sparse; typically more than 95% of the matrix entries are zero. Therefore, we need to handle less than 5% of the matrix entries, which reduces the time required for the convergence of the solution tremendously.

The following steps are followed during the solution of modified node equations:

1. Read the input file and store all the device and model parameters.
2. Renumber the node numbers and expand subcircuit definitions so that node numbers are continuous starting from ground node 0 to number of nodes.
3. Process the model parameters of the devices to ease the linearization of equations.
4. Add internal nodes of the devices whenever they exist in device models.

5. Create modified nodal equations' matrix by reserving all branches on the matrix.
6. Reorder the equations to eliminate singularity problems which may occur during LU decomposition.
7. Reorder the node numbers to minimize number of fills which are the nonzero coefficients created during LU decomposition [10].
8. Load the matrix with linearized device parameters.
9. Solve the equations for node voltages.
10. Check for convergence, if not converged yet, goto 9.
11. Display node voltages if required by user.

ANNSiS takes as the input, the SPICE netlist of the synapse and neuron circuit and a neural network configuration file and creates circuit netlists for all partitions of the network. It also creates the netlist file for the whole circuit with the given weights to verify the results with SPICE. Then, it simulates all partitions in the first layer at the circuit level and finds the outputs of the neurons. The output values of the first layer are then applied to the next layer as independent voltage sources being input to the synapses. Thus, the input is propagated to the output just like in traditional neural network simulators.

4. Simulations

SPICE netlists for different analog MLP (Multilayer Perceptron) structures were generated by ANNSiS using a four-quadrant gilbert multiplier given in figure 2 as synapse and a general purpose opamp given in figure 3 with a nonlinear feedback as a neuron. Three simulations are done for all structures; with SPICE2G6, ANNSiS without partitioning, and ANNSiS with partitioning. Network structures, solution times, memory usage and number of synapses plus neurons are given in table 1.

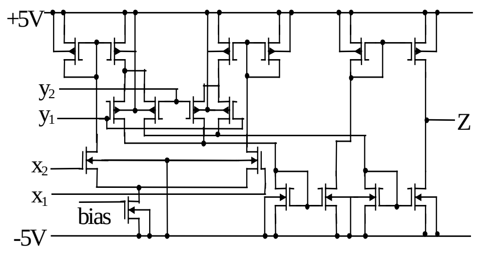

Figure 2. Four-quadrant Gilbert multiplier

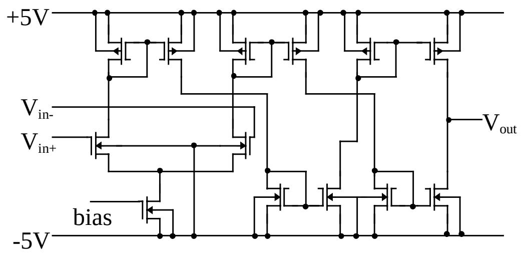

Figure 3. General purpose opamp

Simulation results of SPICE2G6 and ANNSiS are compared for accuracy and found out to be exactly matching. For the 2x5x3 MLP structure and the succeeding structures SPICE2G6 was not able to converge. There is about four times improvement in the solution speed without partitioning, which shows that ANNSiS optimized for DC solutions gives better performance when compared with the SPICE2G6.

Table 1. Simulation results

				SPICE2G6		Without partitioning		With partitioning	
Structure	synapses +opamps	Fets	Nodes	Memory (k)	Time (sec)	Memory (k)	Time (sec)	Memory (k)	Time (k)
2x1	3	49	148	672	3.6	448	0.8	560	0.8
2x2x1	9	147	426	908	13.3	596	2.7	812	1.7
2x3x1	13	213	611	980	19.6	672	4.3	956	2.4
2x4x1	17	279	796	1040	26.3	748	6.6	1084	3.0
2x5x1	21	345	981	1100	31.9	820	8.8	1208	3.7
2x5x2	27	445	1258	1212	49.3	928	14.5	1400	5.1
2x5x3	33	545	1535	*	*	1028	20.6	1564	6.3
2x5x4	39	645	1812	*	*	1128	26.6	1696	7.0
2x5x5	45	745	2089	*	*	1264	32.3	1912	8.8
4x8x7	103	1721	4766	*	*	2136	128.0	3112	19.4

* SPICE2G6 did not converge up to the predefined number of iterations of SPICE.

When the analog neural network circuit was partitioned, simulation time decreased remarkably. As seen in figures 4 and 5, simulation time increases almost linearly for the partitioned case and almost quadratically for the non-partitioned case. This means that simulation time only increases linearly with the number of synapses plus neurons when we partition the circuit, whereas in non-partitioned case simulation time increase more than linearly each time a synapse or neuron is added to the circuit.

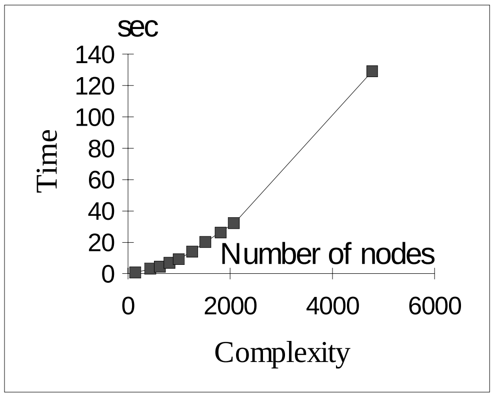

Figure 4. Simulation time for different MLP structures (without partitioning)

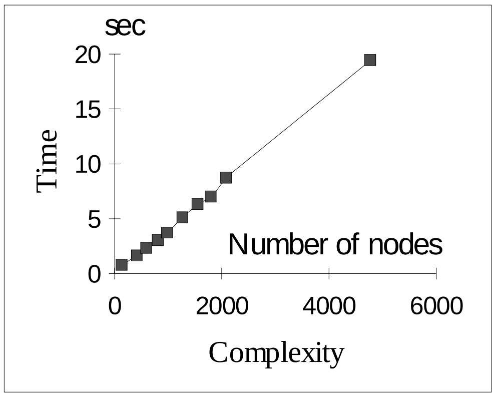

Figure 5. Simulation time for different MLP structures (with partitioning)

Remarkable decrease in simulation time is not the only advantage of partitioning. Without partitioning it is not possible to simulate large analog neural network structures which is necessary in most applications. With partitioning there is almost no limit for the size of network structure. Even if we could not handle the overall structure in the memory at a time, we can overcome this problem by generating the partitions just when we are going to simulate them and use the same memory locations for all the partitions. This will increase the simulation time, but we would still simulate the circuit, which is not possible with the traditional simulation techniques.

5. Conclusion

A circuit level simulator for analog neural networks was developed. This simulator is based on partitioning concepts and brings other improvements over the standard SPICE2G6 simulator. Many examples have been tried with this simulator and a superior performance has been observed for all examples.

While using the partitioning approach, not all of the circuit should be simulated when the user changes some inputs or weights and the iterations for a new solution are started from the last solution points which pulls down the number of iterations for the solution below 5 for most cases, which is around 20-30 for blind starting points. The reason for changing the weight is to incorporate learning algorithms into the program, which is a current research subject. To ease determination of network weights, we are also currently working on modeling the synapses and neurons as proposed in [7] for improving the performance of learning algorithms.

References

[1] H. Binici, G. Dündar, and S. Balkır, “A new multiplier architecture based on radix-2 conversion scheme,” *Proceedings of ECCTD ‘95*, pp 439-442, Istanbul, 1995.

[2] P. Treleaven, M. Pacheco, and M. Vellasco, “VLSI architectures for neural networks,” *IEEE Micro. Mag*., pp 8-27, Dec 1989.

[3] Intel 80170NX ETANN Data Sheets, Feb. 1991.

[4] O. Rossetto et. al., “Analog VLSI synaptic matrices as building blocks for neural networks,” *IEEE Micro. Mag*., pp 56-63, Dec 1989.

[5] G. Dündar and K. Rose, “Analog neural network circuits for ASIC fabrication,” *Proc. of the 5th. IEEE ASIC Conference*, Rochester, 1992, pp 419-422.

[6] G. Dündar, F-C. Hsu, and K. Rose, “Effects of nonlinear synapses on the performance of multilayer neural networks,” *Neural Computation*, in press.

[7] A. Şimþek, M. Civelek, and G. Dündar, “Study of the effects of nonidealities in multilayer neural networks with circuit level simulation,” to be presented in Melecon‘1996.

[8] G. Dündar and K. Rose, “The effects of quantization on multilayer neural networks,” *IEEE Transactions on Neural Networks*, Vol.6, No. 6, pp. 1446-1451, Nov. 1995.

[9] L. W. Nagel, “SPICE2: A computer program to simulate semiconductor circuits,” *MEMO ERL-M520, Univ. Calif. Berkeley, Electronic Research Lab*, May 1975.

[10] L.O. Chua and P-M. Lin, *Computer aided analysis of electronic circuits: algorithms and computational techniques*, Prentice Hall, Inc., 1975, New Jersey.