Microwave Circuit Analysis and Design by a Massively Distributed Computing Network

Mankuan Vai, *Member, IEEE,* and Sheila Prasad, *Senior Member, IEEE*

Abstract—**The advances in microelectronic engineering have rendered massively distributed computing networks practical and affordable. This paper describes one application of this distributed computing paradigm to the analysis and design of microwave circuits. A distributed computing network, constructed in the form of a neural network, is developed to automate the operations typically performed on a normalized Smith chart. Examples showing the use of this computing network for impedance matching and stabilizing are provided.**

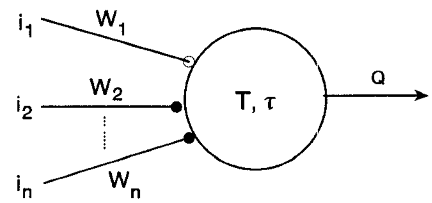


Fig. 1. The conceptual structure of a neuron.

I. Introduction

THE MANUAL analysis and design of microwave circuits are generally tedious and error prone. Recently, computer-aided design (CAD) methodology has established its indispensable role in microwave circuit engineering activities. Following the advances in microelectronic engineering technologies (e.g., VLSI), parallel processing has become a practical and affordable way to conduct microwave CAD operations. Massively distributed computing networks, also called artificial neural networks, have been developed as a special form of parallel processing [1], [2]. A preliminary description of our first attempt to apply such a computing network to microwave design and analysis problems has been reported in [5]. This paper further elaborates the details of this work.

The objective of this paper is to investigate the application of a massively distributed computing network to microwave engineering. The vehicle for this investigation is the development of a massively distributed computing network to perform typical Smith chart operations. Typical problems that can be analyzed using a Smith chart include impedance matching, stability analysis, etc. [3]. A Smith chart represented by a numerical matrix has been developed for conventional computing environments [4]. The Smith chart was selected as a research vehicle here because it is simple yet it provides an important tool for microwave analysis and design.

Section II of this paper provides a brief background of massively distributed computing networks. Section III presents a scheme to map a graphical Y-Z Smith chart onto a massively distributed computing network. Section IV describes the programming methodology of this computing network for microwave engineering problems. The distributed computing network is first programmed to mimic the operations of a designer to provide a conceptual framework for its functionality. This is followed by a modification, described in Section V, which takes practical implementation into consideration. Examples of the use of this computing network to design impedance matching circuits and stabilizing circuits are given.

Manuscript received October 21, 1992; revised September 26, 1994. This work was supported in part by ARPA administered through ONR under Contract N00014-94-1-0687.

The authors are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA.

IEEE Log Number 9410332.

II. Massively Distributed Computing Network

The following description provides the necessary background to understand the technique developed in this paper. A general description of the distributed computing methodology can be found in [1], [2]. Massively distributed computing networks are a specific form of a nonlinear system that maps an input to an output. A distributed computing network can be considered as an asynchronous array processor with very simple processing elements (i.e., neurons). Fig. 1 shows a typical processing element, referred to as a neuron in the following discussion, with n inputs ($i_1 - i_n$) and one output (Q). An input can be excitatory (indicated by a solid circle) or inhibitory (indicated by a hollow circle) and is assigned a weighting factor W_j. A threshold value, T, is associated with the neuron. The function of a neuron can be described by the following equation which combines inputs $i_1 - i_n$ to form an overall input value I

$$I = \sum_{j=1}^{n} W_j \times i_j, \tag{1}$$

where W_j is positive for an excitatory input and negative for an inhibitory input. If the overall input value I is above the threshold value T associated with the neuron, the neuron fires and an output of $Q = 1$ is produced. Otherwise, the output remains at $Q = 0$. A neuron is also associated with a time constant (τ) that determines its output response time.

While the operation of a conventional computer is controlled by a series of instructions, called a program, a massively distributed computing network is programmed by wiring up a set of neurons and setting the weights of these interconnections.

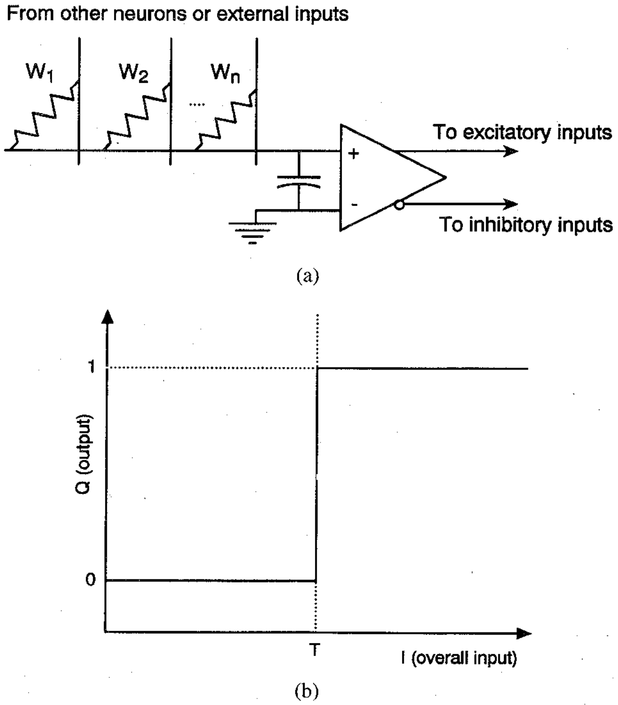


Fig. 2. The physical implementation of a distributed computing network: (a) a neuron implemented by an operational amplifier; (b) the neuron transfer function.

The function of a distributed computing network can only be determined by considering the network as an integrated entity. No meaningful information can be extracted by examining a neuron isolated from its colleagues.

A distributed computing network is typically implemented by a hardware analog circuit [2]. Fig. 2 shows the use of an operational amplifier configured as an integrating adder to carry out the function of a neuron. As shown in Fig. 2(a), the input weighting of such a neuron can be controlled by choosing appropriate resistance values connecting its inputs to the outputs of other neurons. The time constant (τ) of this neuron is determined by the capacitance connected at the operational amplifier input. Fig. 2(b) shows the input-output transfer function of a neuron.

III. Mapping of a Y-Z Smith Chart

A Y-Z Smith chart is shown in Fig. 3. For clarity, only constant-resistance and constant-conductance circles are provided in Fig. 3. Resistance and conductance circles are shown with solid and dotted lines, respectively. It should be noted that the pure resistance/conductance *line* (indicated in the figures by a bold line), which is a special case of the constant-resistance/conductance circles (zero reactance/susceptance), is also included in this Y-Z Smith chart. Though only a sufficient number of circles for the explanation of this novel computing paradigm are provided in Fig. 3, the scheme to be described can be readily extended to any desired precision by adding circles to the Y-Z Smith chart.

Since most operations performed with a Smith chart are carried out by tracing circles and reading the impedance/admittance values at their intersections, the distributed computing network must be designed to perform these activities. As seen in Fig. 3, the resistance and conductance circles intersect each other and form a set of cross-over points on the Y-Z Smith chart. The impedance/admittance values at these cross-over points can be found by identifying their locations on the Y-Z Smith chart. A neuron, called an intersection neuron in the following discussion, is placed on each of these cross-over points. For the desired precision, it is reasonable to assume that there are identical numbers of resistance and conductance circles. The modification of this mapping scheme is straightforward if this assumption has to be removed for any reasons.

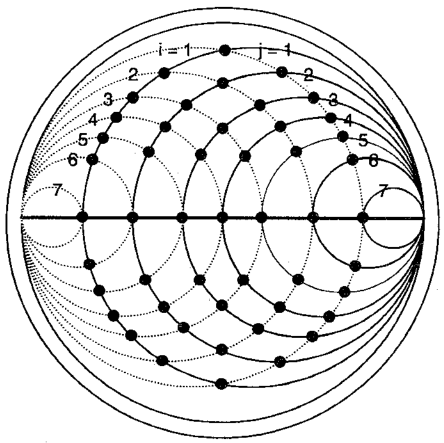


Fig. 3. An example Y-Z Smith chart showing the intersection neuron placement.

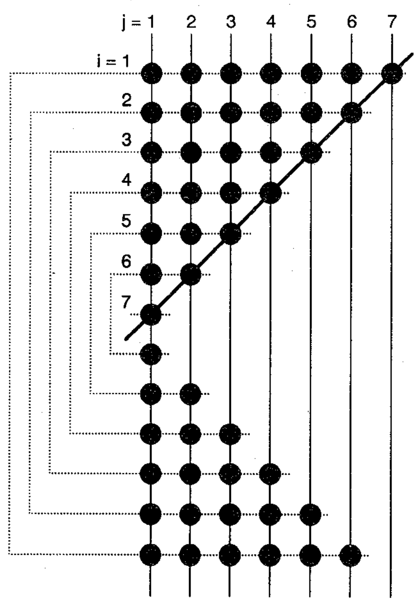
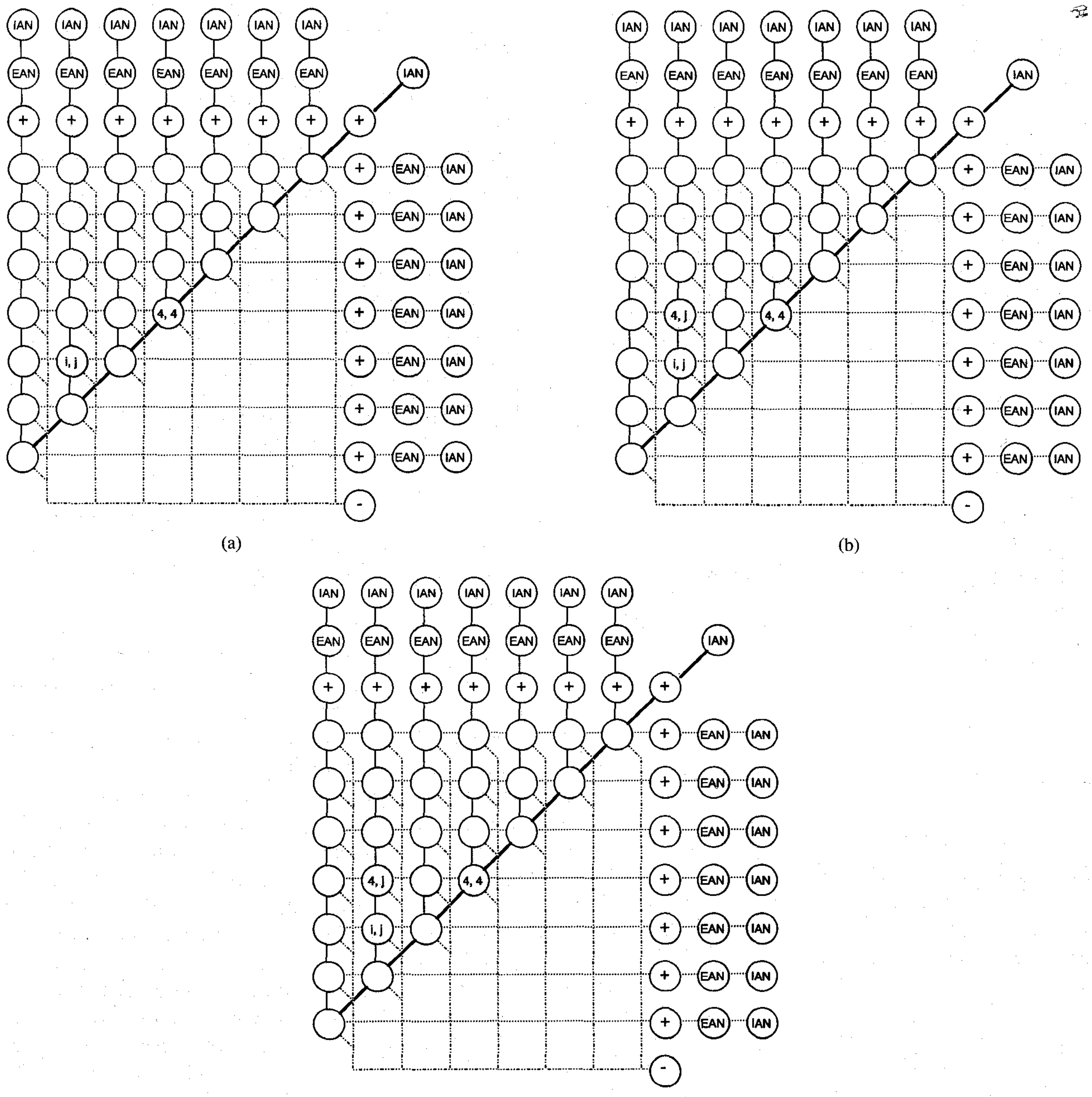


Fig. 4. An unfolded rectilinear version of the Y-Z Smith chart mapped onto a distributed computing network.

In order to clearly show the structure of the distributed computing network and its detailed interconnections, it is unfolded into a rectilinear graph as shown in Fig. 4. As shown in Fig. 3, if the conductance and resistance circles are labeled, beginning with the largest circles, i and j, $1 \leq i, j \leq n$ ($n = 7$ in Fig. 3), respectively, a cross-over point then acquires a pair of coordinates $[i, j]$. This coordinate system is retained



(a) (b) (c)

Fig. 5. A conceptual framework of a distributed computing network showing the snapshots of its operations: (a) $t = \tau$; (b) $t = 2\tau$; (c) $t = 3\tau$.

when the chart is unfolded in Fig. 4. It can be observed from Fig. 3 that, the points on the pure resistance/conductance line are created by two circles tangent to each other. However, a pair of resistance and conductance circles cross over at two points, and thus there are two impedance/admittance values (and neurons too) associated with each pair of coordinates $[i, j]$. In the following discussion, the companion intersection neurons on the upper and lower half planes of the chart are distinguished by identifying them as $N[i, j]$ and $N'[i, j]$, respectively.

IV. Conceptual Framework

Based on the general topology in Fig. 4, the distributed computing network is programmed to mimic the way that a designer uses a Smith chart. This configuration, as shown in Fig. 5(a)–(c), is important in that it directly performs the operations and thus provides a conceptual framework for programming a distributed computing network. It will show how this development will evolve into a practical implementation. For simplicity, Fig. 5 only shows the upper half plane of the Y-Z Smith chart. The distributed computing network is first modified as follows to provide a selection of multiple functions (e.g., impedance matching and stabilizing). In addition to the intersection neurons shown in Fig. 4, an excitatory auxiliary neuron (EAN) and an inhibitory auxiliary neuron (IAN) are added to each row and each column. An inhibitory auxiliary neuron is also added to the pure resistance/conductance line of the network. Neurons are interconnected by solid lines (columns) or dotted lines

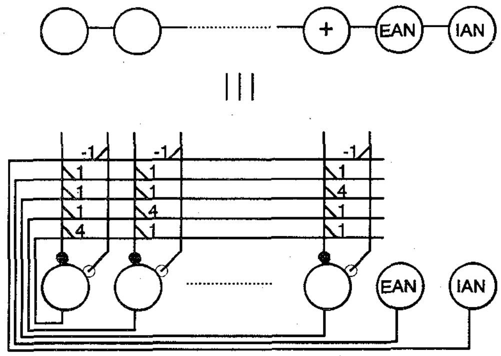


Fig. 6. The relationship between a summing unit and its associated neurons.

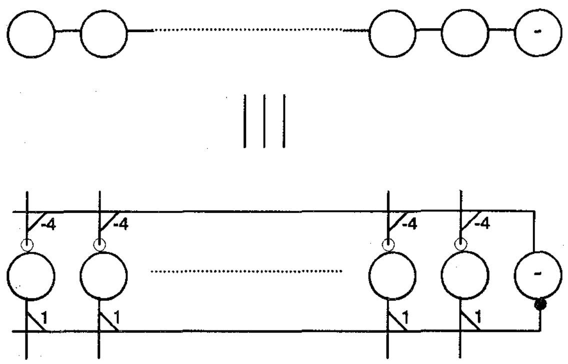


Fig. 7. The relationship between an inhibitory neuron and its associated neurons.

TABLE I
PARAMETERS FOR THE DISTRIBUTED COMPUTING NETWORK SHOWN IN FIG. 5

Thresholds	
Intersection, inhibitory or excitatory auxiliary neuron	1.5
Global inhibitory neuron	4.5
Time Constants	
Intersection neuron	1τ
Inhibitory or excitatory auxiliary neuron	0.5τ
Global inhibitory neuron	0.5τ
Weighting Factors	
External input → intersection neuron	3
External input → companion neuron (full Smith chart only)	-1
External input → inhibitory or excitatory auxiliary neuron	2
Intersection neuron → intersection neuron (except itself)	1
Intersection neuron → itself	4
Excitatory auxiliary neuron → intersection neuron	1
Inhibitory auxiliary neuron → intersection neuron	-1
Intersection neuron → global inhibitory neuron	1
Global inhibitory neuron → intersection neuron	-4

(rows). For clarity, a group of interconnected neurons in which a neuron excites (or inhibits, in a negative sense) other neurons is symbolically shown in Fig. 5 by connecting them to a summing unit (+). The physical implementation of this relationship is provided in Fig. 6 by showing their interconnections and associated weights. It should be noted that the output of a neuron is also fed back to itself for self-support. The reason for this will be discussed later. Each neuron also has an external input (not shown) which can be set by the user. Furthermore, a global inhibitory neuron (−), which receives the outputs of all intersection neurons and sends back an inhibitory signal to them when its threshold is exceeded, is included. The relationship between the global inhibitory neuron and its associated neurons is provided in Fig. 7.

The next step in programming the distributed computing network is to set its parameters, which are the threshold values of the neurons, the time constants of the neurons, and the interconnection weights. It will be obvious in the following discussions that the relative magnitudes of these parameters, rather than their real values, are critical for the operation of this distributed computing network.

The parameters for the distributed computing network in Fig. 5 are summarized in Table I. The inhibitory auxiliary neuron of the pure resistance/conductance line is turned on by its external input for the impedance matching operation. It will be shown later that because of the connection of this inhibitory auxiliary neuron to the neurons in the pure resistance/conductance line (see Fig. 6), the pure resistance/conductance line does not participate in the impedance matching operation. This is needed because the participation of the resistance/conductance line in an impedance matching process will result in a lossy circuit which is normally undesirable.

Two neurons, $N[i, j]$ and $N[4, 4]$, representing the original impedance and the impedance to which it is to be matched, respectively, are turned on by applying external inputs to them. The threshold value of an intersection neuron and the weighting factor of its external input are thus set to 1.5 and 3, respectively. After the time constant (1τ) of the intersection neurons, they fire and produce outputs of 1. A snapshot of the distributed computing network at $t = 1\tau$ is shown in Fig. 5(a), in which a shaded circle indicates a firing neuron. The firing of neuron $N[i, j]$ provides an excitatory signal of 1 to all neurons in row i and column j. Meanwhile, similar effects are imposed on all neurons in row 4 and column 4 by the firing of neuron $N[4, 4]$. Some thinking will reveal that the output of neuron $N[4, 4]$ to the intersection neurons associated with the pure resistance/conductance line will be canceled by the inhibitory input from its inhibitory auxiliary neuron. Because of the threshold value of the neurons, which is 1.5, the excitatory signals have only an effect on neuron $N[4, j]$, which will receive a total input of two. The threshold of neuron $N[4, j]$ is exceeded and it fires at $t = 2\tau$ (Fig. 5(b)). The firing of neuron $N[4, j]$ boosts the inputs of the intersection neurons at column j and row 4 to two. All the intersection neurons on row 4 (a conductance circle) and column j (a resistance circle) are thus turned on at $t = 3\tau$ (Fig. 5(c)). This operation is comparable to what a designer will do on a Smith chart for impedance matching.

An impedance matching circuit can be determined by tracing the highlighted circles at this moment. However, it can be easily seen from Fig. 5(c) that every intersection neuron now has a minimum input value of two and hence all of them will be turned on eventually. The global inhibitory neuron is provided to solve this problem. The threshold value of the global inhibitory neuron is selected so that it will remain idle when less than five neurons are turned on. The multiple neurons on the firing column and row in Fig. 5(c) raise the global inhibitory neuron input to a value beyond its threshold which is 4.5. The inhibitory neuron then fires and sends an inhibitory signal, which has a weight of −4, to all intersection neurons. Because of the smaller time constant (0.5 τ) of this global inhibitory neuron, the inhibitory signal will stop the neurons, which remained off to this time, from being turned on. This inhibitory signal has no effect on the neuron already turned on since it is canceled out by self-supporting excitatory signals generated from themselves (see Fig. 6). The distributed computing network will thus stay at this stable state until it is reset.

Tracing the path consisting of the neurons in the firing column j and row 4 will identify neurons $N[4, j]$ as the cross-over point of the resistance and conductance circles. The location of neuron $N[4, j]$ on the Smith chart determines the topology and value of impedance/admittance that should be added to the circuit. A detailed description showing the simple steps in using such a result to form the matching circuit can be found in [3] and will not be elaborated here.

The distributed computing network shown in Fig. 5 performs impedance matching by locating the cross-over point(s) on the Y-Z Smith chart between a resistance circle and a conductance circle. This capability can be extended to perform the design of a stabilizing circuit. Given a certain circuit with $K < 1$ (i.e., potentially unstable), the calculation of a stability circle can be used to identify a resistance circle or a conductance circle as the boundary of the unconditionally stable region.

Referring to Fig. 8, suppose a resistance circle j is selected, the excitatory auxiliary neuron of column j and the neuron $N[4, 4]$ ($Z = 1$) are turned on by their external inputs, respectively. On the other hand, the inhibitory auxiliary neurons of row 4, which corresponds to the conductance circle that goes through $Z_o = 1$, are turned on by its external input. The process of the distributed computing network moving into a stable state is similar to what was described for impedance matching. The only difference is that the network now finds the cross-over neuron between column j and the pure resistance/conductance line instead of a conductance circle. Tracing the firing column and resistance/conductance line will find the value indicating the resistance that is required to be added to stabilize the circuit. The final status of the computing network is shown in Fig. 8.

The distributed computing network shown above has a number of shortcomings and is only useful as a conceptual framework. First, the neuron at the intersection of the firing column and row needs to be located manually by tracing them. In addition, a global inhibitory neuron is needed; this makes the implementation difficult. Furthermore, the fact that only half a Smith chart is represented limits its uses. The network in Fig. 5 can be readily expanded to include the lower half plane of the Smith chart by adjusting the parameters in Table I. However, it can be observed that, when a full Smith chart (Fig. 4) is considered, a neuron $N[i, j]$ ($N'[i, j]$), when excited by an external input, sends two excitatory signals to its companion neuron $N'[i, j]$ ($N[i, j]$), one through the row interconnection and one through the column interconnection. A solution is provided here for the completeness of the discussion. In order to remove the redundant excitatory signal

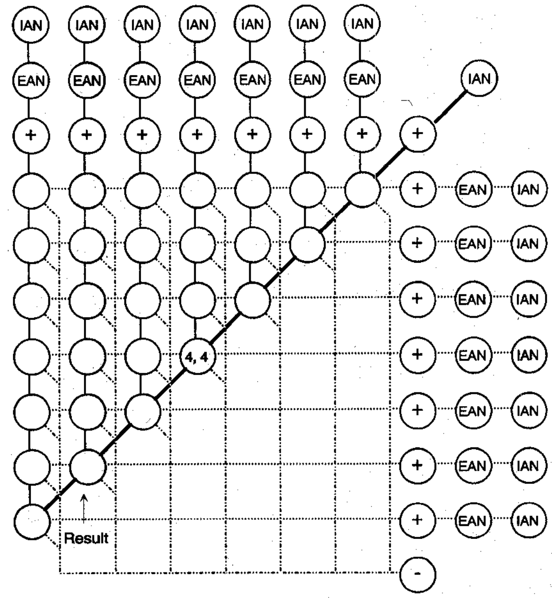


Fig. 8. The computing network of Fig. 5 programmed to design a stabilizing circuit showing its final status.

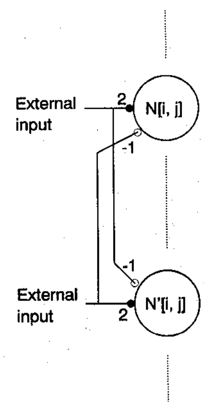


Fig. 9. The input relationship between a pair of companion intersection neurons.

between a pair of companion neurons, the external input of a neuron $N[i, j]$ ($N'[i, j]$) is also sent to $N[i, j]$ ($N'[i, j]$) through a connection weighted negative one (−1) so that the effect of the redundant excitatory signal is canceled. This is demonstrated for a pair of neurons $N[i, j]$ and $N'[i, j]$ in Fig. 9.

V. Practical Implementation

The drawbacks of the computing network shown in Section IV can be removed by a simple modification. In Fig. 5, a neuron has two responsibilities, calculation and result indication, which are distributed to two separate neurons in this modification. In the network shown in Fig. 10, each intersection neuron consists of a display neuron (D) and a calculation neuron (C). All the C-neuron outputs in a row/column are connected to a summation unit (Σ) which sends the result to all D-neurons in the same row/column as an excitatory signal. As shown in Fig. 11, this is equivalent to exciting the receivers of the summation unit by all its

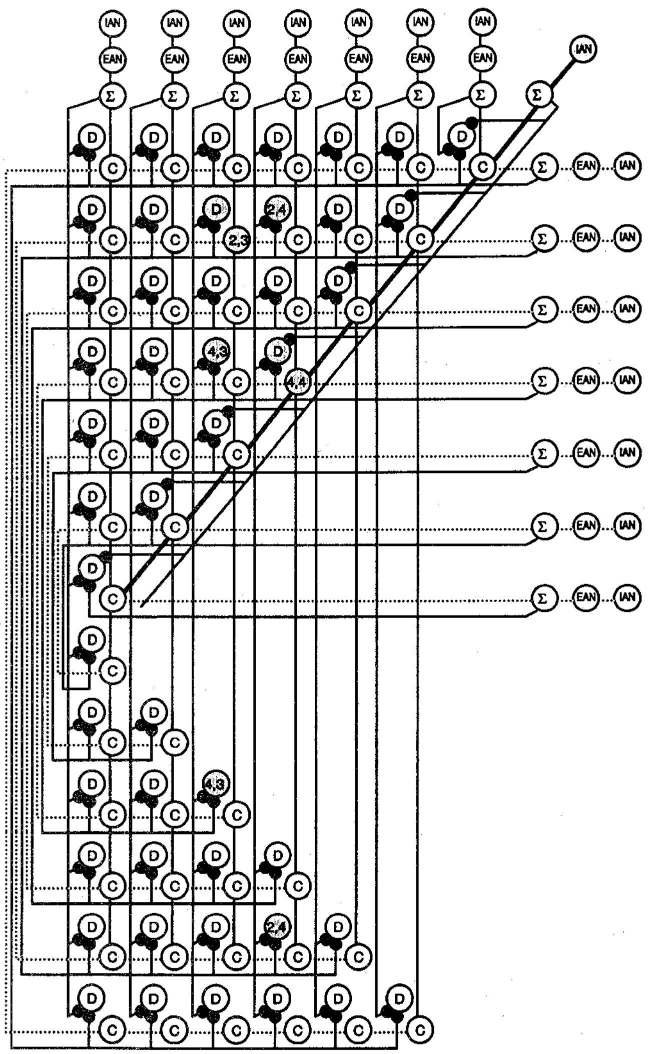


Fig. 10. A practical implementation of a distributed computing network showing the result of designing an impedance matching circuit.

contributing C-neurons. Each pair of D- and C-neurons at the same location share a common external input (not shown) which can be set by a user. Other than these modifications, the distributed computing network follows the topology shown in Fig. 4. The parameters for programming this distributed computing network are given in Table II. The important feature of this implementation is that the separation of the calculation and the display functions into two neurons eliminates the need of tracing. The other shortcomings of the network described in Section IV are also eliminated.

The distributed computing network of Fig. 10 is used to solve an impedance matching problem given in Fig. 12. It is required to match a circuit with an intrinsic impedance of Z_i to $Z_o = 1$ using the T-network shown. A suitable inductance L_1 is first selected, and the impedance $j\omega L_1 + Z_i$ is represented by, say neuron $C[2, 3]$. As explained in Section IV, the inhibitory auxiliary neuron in the pure resistance/conductance line is turned on by an external input for an impedance matching operation. The D- and C-neurons at both locations $C[2, 3]$ (the original impedance) and $C[4, 4]$ ($Z_o = 1$) are turned on at $t = \tau$ by their external inputs. As shown in

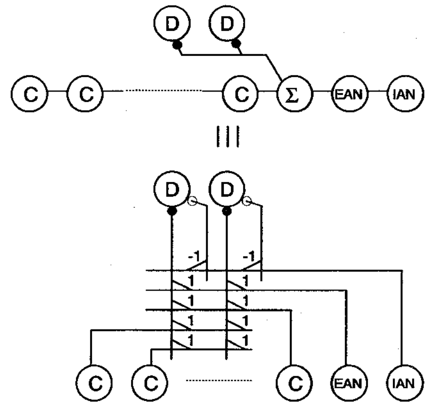


Fig. 11. The relationship between a summation unit and its associated neurons.

TABLE II
PARAMETERS FOR THE DISTRIBUTED COMPUTING NETWORK SHOWN IN FIG. 8

Thresholds	
D-neuron, C-neuron, inhibitory, or excitatory auxiliary neuron	1.5
Time Constants	
D-neuron or C-neuron	1τ
Inhibitory or excitatory auxiliary neuron	0.5τ
Weighting Factors	
External input → D-neuron, C-neuron	3
External input → inhibitory or excitatory auxiliary neuron	2
Excitatory auxiliary neuron → D-neuron	1
Inhibitory auxiliary neuron → D-neuron	-1
C-neuron → D-neuron	1

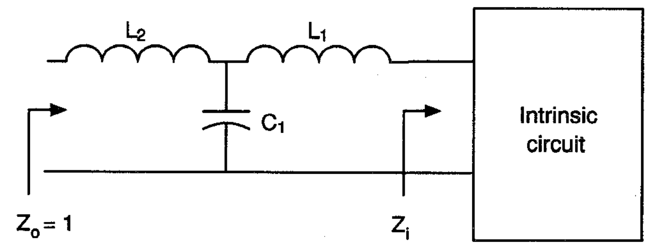


Fig. 12. An example impedance matching problem.

Fig. 10, the summation units of row 4 and column 2 will turn on the D-neurons $D[4, 3]$, $D'[4, 3]$, $D[2, 4]$, and $D'[2, 4]$ at $t = 2\tau$. The impedance/admittance values at these four intersection points give four different impedance matching circuits. Under the topology constraint set by the T-network, only $D'[2, 4]$ is the valid solution and the values of C_1 can be determined by the susceptance difference between the admittance values at $D[2,3]$ and $D'[2, 4]$ and L_2 can be determined by the reactance difference between the impedance values at $D'[2, 4]$ and $D[4, 4]$.

Alternatively, the distributed computing network can be used to design a stabilizing circuit. In Fig. 13(a), suppose that the resistance circle $j = 2$ is the boundary of the unconditionally stable region. The inhibitory auxiliary neuron of row 4, which corresponds to the conductance circle that goes through $Z_o = 1$, is turned on by its external input. The excitatory auxiliary neuron of column 2 and neuron $N[4, 4]$ ($Z_o = 1$) are turned on by their external inputs. Neuron $D[6,$

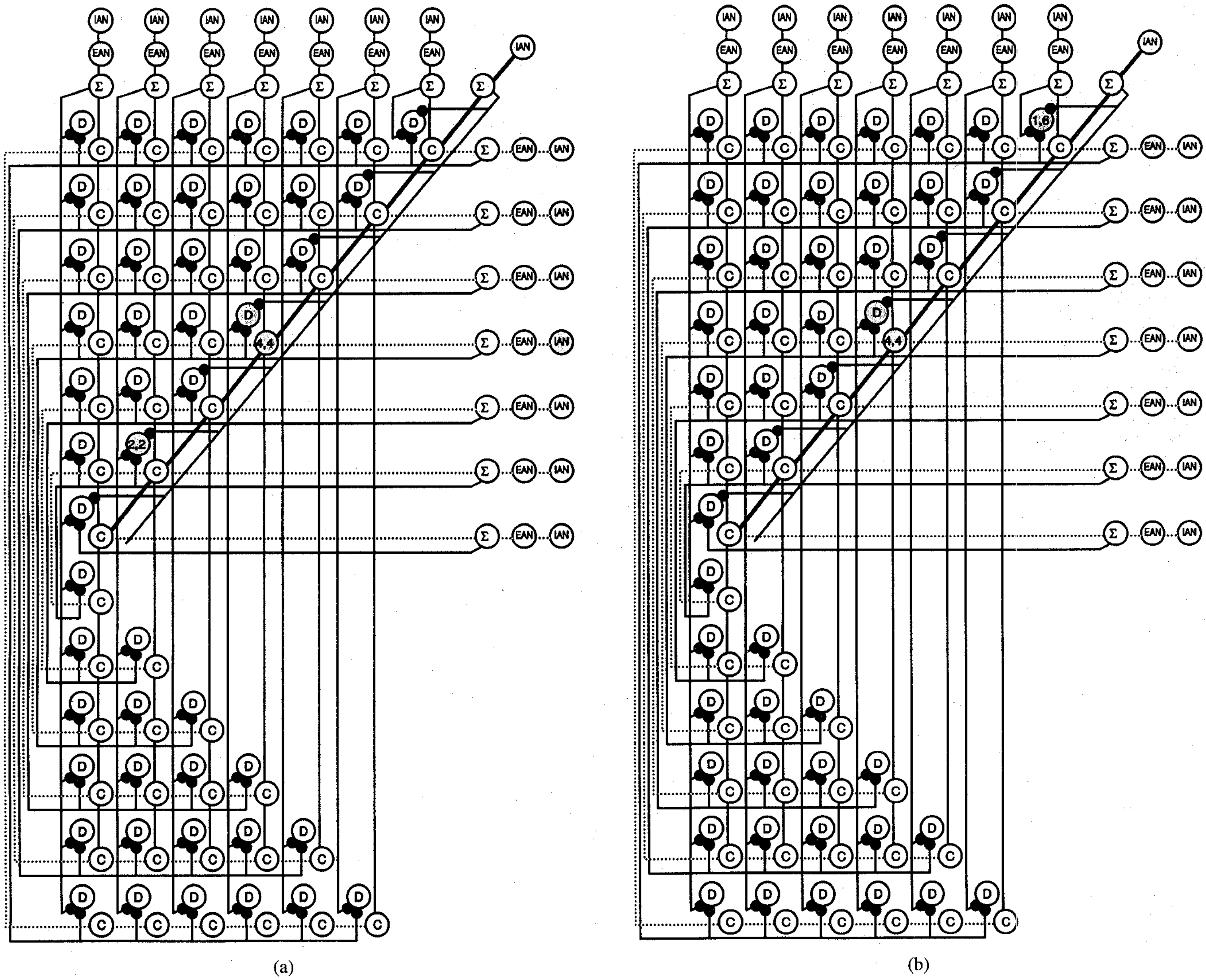


Fig. 13. The result of using the computing network of Fig. 10 to design a stabilizing circuit: (a) a resistance circle is selected; (b) a conductance circle is selected.

2] is turned on at $t = \tau$ by the summation units of column 2 and the resistance/conductance line. The resistance difference between the impedance values at $D[4, 4]$ and $D[6, 2]$ indicates the resistance that is needed to be added (in series) for the purpose of stabilization.

Similarly, assume instead, that a conductance circle $i = 1$ is identified as the boundary of the stable region. In Fig. 13(b), the inhibitory auxiliary neuron of column 4, which corresponds to the resistance circle that goes through $Z_o = 1$, is turned on by its external input. The excitatory auxiliary neuron of row 1 and neuron $C[4, 4]$ ($Z_o = 1$) are turned on by their external inputs. Neuron D [1, 6] is turned on at $t = \tau$ by the summation units of row 1 and the resistance/conductance line. The conductance difference between the admittance values at $D[4, 4]$ and $D[1, 6]$ indicates the conductance that is needed to be added (in parallel) for the purpose of stabilization.

VI. Summary

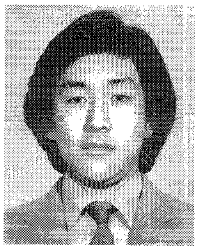
In summary, this paper describes the first attempt at applying a massively distributed computing network to microwave analysis and design. An intuitively simple mapping scheme is provided to program a distributed computing network to perform typical operations on a Smith chart. Examples to show how this computing network can be programmed to design impedance matching circuits and stabilizing circuits are given. The major benefit of this approach is the speed brought by its parallelism when it is implemented on a VLSI application specific processor. Ongoing research motivated by the encouraging results of this paper includes the extension of this technique to perform an in-circuit, real-time adjustable broadband impedance matching. In addition, other design and analysis tasks such as optimizing the noise figure and the gain of an amplifier are also being investigated.

References

[1] R. P. Lippmann, "An introduction to computing with neural nets," *IEEE ASSP Mag.*, pp. 4–22, Apr. 1987.

[2] J. J. Hopefield and D. W. Tank, "'Neural' computation of decisions in optimization problems," *Biological Cybernetics,* vol. 52, pp. 141–152, 1985.

[3] G. Gonzalez, *Microwave Transistor Amplifiers Analysis and Design.* Englewood Cliffs, NJ: Prentice-Hall, 1987, pp. 42–80, 95–102.
[4] M. Vai and S. Prasad, "Computer-aided microwave circuit analysis by a computerized numerical Smith chart," *IEEE Microwave and Guided Wave Lett.,* vol. 2, no. 7, pp. 294–296, 1992.
[5] M. Vai, S. Prasad, and H. Wang, "A Smith chart represented by a neural network and its applications," in *MTT-S Int. Microwave Symp. Dig.,* 1992, pp. 1565–1568.



Mankuan Vai (S'84–M'87) received the B.S. degree from the National Taiwan University, Taipei, Taiwan, in 1979, and the M.S. and Ph.D. degrees from Michigan State University, East Lansing, MI, in 1985 and 1987, respectively, all in electrical engineering.

He is currently associate professor of Electrical and Computer Engineering at Northeastern University, Boston, MA. He has worked and published in microelectronics and computer engineering education.

Sheila Prasad (M'80–SM'82) received the B.Sc. degree from the University of Mysore, IN, and the S.M. and Ph.D. degrees in applied physics from Harvard University.

She has been on the faculty at New Mexico State University, Las Cruces, the American University in Cairo, Egypt, and the Birla Institute of Technology and Science, Pilani, India. She is presently on the faculty of the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA. Her areas of research include electromagnetics, microwave semiconductor devices, and circuits. She is coauthor, with Prof. R. W. P. King, of the book *Fundamental Electromagnetic Theory and Applications.*

Dr. Prasad is a member of Sigma Xi.