[12] R. B. Craven, "An integrated circuit 12-Bit D/A converter," *ISSCC Digest Tech. Papers*, pp. 40–41, 1975.
[13] J. A. Schoeff, "An inherently monotonic 12b DAC," *IEEE J. Solid State Circuits*, vol. SC-14, pp. 904–911, 1979.
[14] M. Shoji, "Elimination of process dependant clock skew in CMOS VLSI," *IEEE J. Solid State Circuits*, vol. SC-21, pp. 875–880, 1986.
[15] K. R. Lakshmikumar, R. A. Hadaway, M. A. Copeland, and M. I. H. King, "A high-speed 8-bit current steering CMOS DAC," in *Proc. IEEE 1985 Custom Integrated Circuits Conf.*, pp. 156–159, 1985.
[16] P. H. Saul, D. W. Howard, and C. J. Greenwood, "An 8b CMOS video DAC," *ISSCC Digest Tech. Papers*, pp. 32–33, 1985.

Artificial Neural Networks for Four-Coloring Map Problems and K-Colorability Problems

Yoshiyasu Takefuji and Kuo Chun Lee

Abstract —**The computational energy is presented for solving a four-coloring map problem. The map-coloring problem is defined that one wants to color the regions of a map in such a way that no two adjacent regions (that is, regions sharing some common boundary) are of the same color. This paper presents a parallel algorithm based on the McCulloch–Pitts binary neuron model and the Hopfield neural network. It is shown that the computational energy is always guaranteed to monotonically decrease with the Newton equation. A $4 \times n$ neural array is used to color a map of n regions where each neuron as a processing element performs the proposed Newton equation. The capability of our system is demonstrated through a large number of simulation runs. The parallel algorithm is extended for solving the K-colorability problem.**

I. Introduction

In 1943, mathematical models based on biological computation were proposed by McCulloch and Pitts [1]. They attempted to take advantage of the elegant natural biological computation in the brains of animals and human beings. Hebb presented the learning theory for realizing the associative memory where the strengths of the existing synaptic connections between neurons are modified by the input pattern [2]. Widrow at Stanford University demonstrated adaptive switching circuits in 1960 [3]. In 1961, Rosenblatt at Cornell University presented perceptrons and the theory of brain mechanisms in his book [4]. In 1969, Minsky and Papert at MIT showed the limitation of perceptrons in their book [5]. Negative results against the artificial neural network computing had caused less support and interest from governments/industries and consequently shrank the scale of neural network study and the number of investigators. However, a small number of researchers such as Amari, Cooper, Fukushima, and Grossberg studied the neural network computing during the 1960's and 1970's. In the 1970's, Anderson and Kohonen developed mathematical models of associative memory. The new discovery in neurobiology and the explosive interest in parallel computation along with the inexpensive VLSI technology have caused a dramatic resurgence.

The mathematical model of the artificial neural network consists of two components: neurons and synaptic links. The output signal transmitted from a neuron propagates to other neurons through the synaptic links. The state of the input signal of a neuron is determined by the linear sum of weighted input signals from the other neurons where the respective weight is the strength of the synaptic links. In our parallel algorithm the McCulloch–Pitts binary neuron model is used. The McCulloch–Pitts input/output function is given by $V_i = f(U_i) = 1$, if $U_i > 0$ and 0 otherwise where V_i and U_i are the output and input of the ith neuron, respectively. U_i is given by $U_i = \sum_k W_{ki} V_k$ where W_{ki} is the strength of the synaptic link from the kth neuron to the ith neuron. The synaptic links and the strength of the synaptic links are given by the motion equation or the Newton equation.

The first neural network for combinatorial optimization problems was introduced by Hopfield and Tank in 1985 [6]. They use the predefined energy function E, which follows the quadratic form

$$E = \sum_{i=1}^{N} \sum_{j=1}^{N} G_{ij} V_i V_j + \sum_{i=1}^{N} V_i I_i \tag{1}$$

where V_i is the output of the ith neuron and G_{ij} is the conductance between the ith and the jth neurons. Note that I_i is the constant bias of the ith neuron.

Hopfield gives the motion equation of the ith neuron [6]:

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau} - \frac{\partial E}{\partial V_i} \tag{2}$$

where the output follows the continuous nondecreasing function (sigmoid):

$$V_i = f(U_i) = \frac{1}{2}(\tanh(\lambda_0 U_i) + 1) \tag{3}$$

where λ_0 is the gain of the sigmoid function.

Wilson and Pawley strongly criticized the Hopfield and Tank neural network for the travelling salesman problem [7]. Unfortunately, Wilson and Pawley did not know what causes the problem. The use of the decay term $(-U_i/\tau)$ in (2) sometimes increases the computational energy. The following shows why the use of the decay term may increase the computational energy.

Proof: Consider the derivatives of the computational energy E with respect to time t:

$$\frac{dE}{dt} = \sum_i \frac{dV_i}{dt}\frac{dE}{dV_i} = \sum_i \frac{dV_i}{dt}\left(-\frac{U_i}{\tau} - \frac{dU_i}{dt}\right),$$

$$\text{where } dE/dV_i \text{ is replaced by } \left(-\frac{U_i}{\tau} - \frac{dU_i}{dt}\right) \text{ from (2)}$$

$$= -\sum_i \frac{dV_i}{dt}\frac{U_i}{\tau} - \sum_i \frac{dV_i}{dt}\frac{dU_i}{dt}$$

$$= -\sum_i \frac{dV_i}{dt}\frac{U_i}{\tau} - \sum_i \left(\frac{dU_i}{dt}\frac{dV_i}{dU_i}\right)\left(\frac{dU_i}{dt}\right)$$

$$= -\sum_i \frac{dV_i}{dt}\frac{U_i}{\tau} - \sum_i \left(\frac{dV_i}{dU_i}\right)\left(\frac{dU_i}{dt}\right)^2.$$

The first term $-\sum_i (dV_i/dt)(U_i/\tau)$ is positive, negative, or zero. The second term $-\sum_i (dV_i/dU_i)(dU_i/dt)^2$ is always negative or zero, because the output $V_i = f(U_i)$ is a nondecreasing function. The following condition can be true: $-\sum_i (dV_i/dt)(U_i/\tau) - \sum_i (dV_i/dU_i)(dU_i/dt)^2 > 0$ only if one of the following conditions is satisfied: ($U_i > 0$ and $(dV_i/dt) < 0$) or ($U_i < 0$ and $(dV_i/dt) > 0$). Under such a condition the derivatives of E with respect to time t may be positive; $dE/dt > 0$. Q.E.D.

Manuscript received April 2, 1990; revised October 4, 1990. This paper was recommended by Associate Editor C. Lau.

The authors are with the Department of Electrical Engineering and Applied Physics, Case Western Reserve University, Cleveland, OH 44106.

IEEE Log Number 9041765.

It is easy to show that the Newton equation always forces the energy function to decrease monotonically. The change of the input state of the ith neuron is given by the partial derivatives of the computational energy E with respect to the output of the ith neuron where E is an n-variable function: $E(V_1, V_2, \cdots, V_n)$. It is given by

$$\frac{dU_i}{dt} = -\frac{\partial E(V_1, V_2, \cdots, V_n)}{\partial V_i}. \tag{4}$$

Whatever the computational energy function E is given, the Newton equations force it to monotonically decrease. The following proof shows that the Newton equations force the state of the system to converge to the local minimum [8].

Proof: Consider the derivatives of the computational energy function E with respect to time t:

$$\frac{dE}{dt} = \sum_i \frac{dV_i}{dt}\frac{dE}{dV_i}$$

$$= \sum_i \frac{dV_i}{dt}\left(-\frac{dU_i}{dt}\right),$$

where the Newton equation replaces dE/dV_i by $\left(-\frac{dU_i}{dt}\right)$

$$= -\sum_i \left(\frac{dU_i}{dt}\frac{dV_i}{dU_i}\right)\left(\frac{dU_i}{dt}\right)$$

$$= -\sum_i \left(\frac{dV_i}{dU_i}\right)\left(\frac{dU_i}{dt}\right)^2 \leqslant 0.$$

As long as the input/output function of the neurons obeys the continuous nondecreasing function, dV_i/dU_i must be positive so that dE/dt is negative or zero. Therefore, the state of the system is always guaranteed to converge to the local minimum. Q.E.D.

In the network the state of each neuron is influenced by the states of itself and other neurons. The strength of synaptic interconnections actually determines the motion of the neurodynamical system. The Newton equation of the neurons describes the neurodynamics on how the state of the corresponding neuron will be influenced by the other neurons. It is important to note that one of the significant advantages in neural network parallel computing lies in parallel asynchronization. Each neuron communicates with other neurons without any clock or handshaking synchronization. The state of each neuron will be asynchronously updated by observing the neighboring states of other neurons linked to it.

The following shows why the Newton equation of the discrete McCulloch–Pitts binary neurons monotonically decreases the computational energy function.

Proof: Consider the derivatives of the computational energy E with respect to time t.

$$\frac{dE}{dt} = \sum_i \frac{dV_i}{dt}\frac{dE}{dV_i}$$

$$= \sum_i \frac{dV_i}{dt}\left(-\frac{dU_i}{dt}\right), \quad \text{where } dE/dV_i \text{ is replaced by } \left(-\frac{dU_i}{dt}\right)$$

$$= -\sum_i \left(\frac{dU_i}{dt}\frac{dV_i}{dU_i}\right)\left(\frac{dU_i}{dt}\right)$$

$$= -\sum_i \left(\frac{dV_i}{dU_i}\right)\left(\frac{dU_i}{dt}\right)^2.$$

Let dU_i/dt be $U_i(t+\Delta t) - U_i(t)$. Let dV_i/dU_i be $(V_i(t+\Delta t) - V_i(t))/(U_i(t+\Delta t) - U_i(t))$. It is necessary and sufficient to consider the following seven cases:

1) $U_i(t+\Delta t) > U_i(t)$, $U_i(t+\Delta t) < 0$
2) $U_i(t+\Delta t) > U_i(t)$, $U_i(t+\Delta t) \geqslant 0$, and $U_i(t) < 0$
3) $U_i(t+\Delta t) > U_i(t)$, $U_i(t+\Delta t) > 0$, and $U_i(t) \geqslant 0$
4) $U_i(t+\Delta t) < U_i(t)$, $U_i(t+\Delta t) \geqslant 0$
5) $U_i(t+\Delta t) < U_i(t)$, $U_i(t+\Delta t) < 0$, and $U_i(t) \geqslant 0$
6) $U_i(t+\Delta t) < U_i(t)$, $U_i(t+\Delta t) < 0$, and $U_i(t) < 0$
7) $U_i(t+\Delta t) = U_i(t)$.

If the condition 7) is satisfied, then dU_i/dt must be zero so that $dE/dt = 0$. If condition 1), 3), 4), or 6) is satisfied, then dV_i/dU_i must be zero, because $dV_i/dU_i = (V_i(t+\Delta t) - V_i(t)/(U_i(t+\Delta t) - U_i(t) = 0/(\text{nonzero number})$, so that $dE/dt = 0$. If condition 2) is satisfied, then dV_i/dU_i must be positive, because $dV_i/dU_i = (V_i(t+\Delta t) - V_i(t)/(U_i(t+\Delta t) - U_i(t) = 1/(\text{positive number})$, so that $dE/dt < 0$. If condition 5) is satisfied, then dV_i/dU_i must again be positive, because $dV_i/dU_i = (V_i(t+\Delta t) - V_i(t))/(U_i(t+\Delta t) - U_i(t)) = -1/(\text{negative number})$, so that $dE/dt < 0$.

We can conclude that the energy function E monotonically decreases as long as the Newton equation of the binary neurons is given by $dU_i/dt = -\partial E/\partial V_i$. Q.E.D.

II. Four-Coloring Map Problems

A map maker colors adjacent countries with different colors so that they may be easily distinguished. This is not a problem as long as one has a large number of colors. However, it is more difficult with a constraint that one must use the minimum number of colors required for a given map. It is still easy to color a map with a small number of regions. In the early 1850's, Francis Guthrie was interested in this problem, and he brought it to the attention of Augustus De Morgan. Since then many mathematicians, including Arthur Kempe, Peter Tait, Percy Heawood, and others tried to prove the problem that any planar simple graph can be colored with four colors. A four-color problem is defined that one wants to color the regions of a map in such a way that no two adjacent regions (that is, regions sharing some common boundary) are of the same color. In August 1976, Appel and Haken presented their work to members of the American Mathematical Society [9]. They showed a computer-aided proof of the four-color problem. However, their coloring was based on the sequential method so that it took many hours to solve a large problem. Their computation time may be proportional to $O(n^2)$ where n is the number of regions.

Few parallel algorithms have been reported. Dahl [10], Moopenn *et al.* [15], and Thakoor *et al.* [16] have presented the first neural network for map K-colorability problems [10]. In our algorithm, the four-color problem is solved by a $4 \times n$ two-dimensional neural array where n is the number of regions to be colored.

Four colors can be simply expressed by four neurons. For example, red, yellow, blue, and green are represented by 1000, 0100, 0010, and 0001, respectively. A single region requires four neurons for the single-color assignment. Therefore, a $4 \times n$ two-dimensional neural array is needed for coloring n regions of a map. For example, five regions are colored by four colors as shown in Fig. 1. The neural representation for the five-region map problem is given in Fig. 2 where a 4×5 neural array is used. Fig. 3 shows the adjacency matrix of the five-region map, which gives the boundary information between regions.

In order to consider in such a way that no two adjacent regions are of the same color, the basic energy function is given

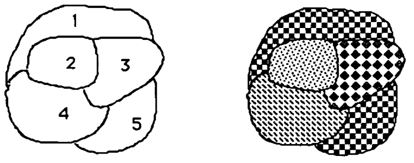


Fig. 1. A five-region map and a four-colored map.

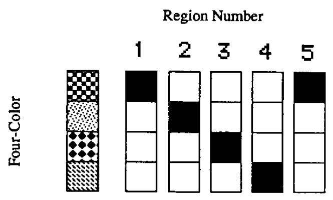


Fig. 2. Neural representation for the five-region map.

	1	2	3	4	5
1		1	1	1	
2	1		1	1	
3	1	1		1	1
4	1	1	1		1
5			1	1	

Fig. 3. An adjacency matrix of the map.

by

$$E=\frac{A}{2}\sum_{X=1}^{n}\left(\sum_{i=1}^{4}V_{Xi}-1\right)^{2}+B\sum_{X=1}^{n}\sum_{\substack{Y=1\\Y\neq X}}^{n}\sum_{i=1}^{4}d_{XY}V_{Xi}V_{Yi} \tag{5}$$

where d_{XY} is 1 if regions X and Y are adjacent to each other, and 0 otherwise. Note that A and B are constant. The adjacency matrix is an $n\times n$ array where d_{XY} is a single element. Note that V_{Xi} is the output of the ith neuron in the X region. The first term corresponds to the column constraint in the neural array which forces one and only one neuron's output to be nonzero. The last term describes the boundary violation between regions. If X and Y regions have a common boundary ($d_{XY}=1$), then X and Y regions should not have the same color i.

The Newton equation of the ith color of the Xth region is given by

$$\frac{dU_{Xi}}{dt}=-\frac{\partial E}{\partial V_{Xi}}. \tag{6}$$

Equations (5) and (6) generate the equation of the ith color of the Xth region by

$$\frac{dU_{Xi}}{dt}=-A\left(\sum_{j=1}^{4}V_{Xj}-1\right)-B\sum_{\substack{Y=1\\Y\neq X}}^{n}d_{XY}V_{Yi}. \tag{7}$$

We have introduced the hill-climbing term, which allows the state of the system to escape from the local minimum and to converge to the global minimum. The hill-climbing term has been successfully used for solving graph planarization problems

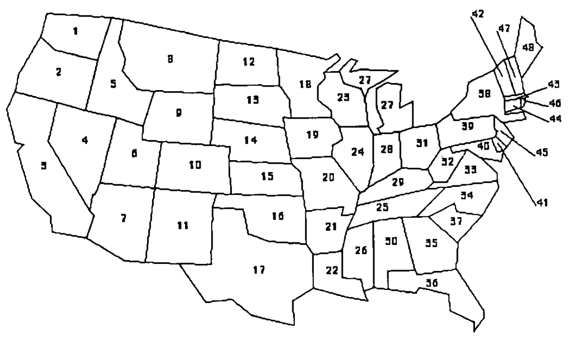


Fig. 4. A 48-state map of the continental United States.

[11], tiling problems [12], and secondary structure prediction problems in ribonucleic acids [13]. The final Newton equation of the ith color of the Xth region is given by

$$\frac{dU_{Xi}}{dt}=-A\left(\sum_{j=1}^{4}V_{Xj}-1\right)-B\sum_{\substack{Y=1\\Y\neq X}}^{n}d_{XY}V_{Yi}\sum_{K=1}^{n}d_{YK}$$
$$+C\cdot h\left(\sum_{j=1}^{4}V_{Xj}\right)\left(C_1\sum_{K=1}^{n}d_{XK}+C_2\frac{\sum_{K=1}^{n}\sum_{Y=1}^{n}d_{XY}d_{YK}}{\sum_{K=1}^{n}d_{XK}}\right). \tag{8}$$

The coefficient of the second term is normalized. Note that C, C_1, and C_2 are constant. The last term is the hill-climbing term where $h(x)$ is 1 if $x=0$, and 0 otherwise. It performs the excitatory force only when all of V_{Xj} are zeros. The hill-climbing term allows the state of the system to escape from the local minimum and to converge to the global minimum.

III. The Algorithm

The following procedure describes the proposed algorithm to the four-coloring problem.

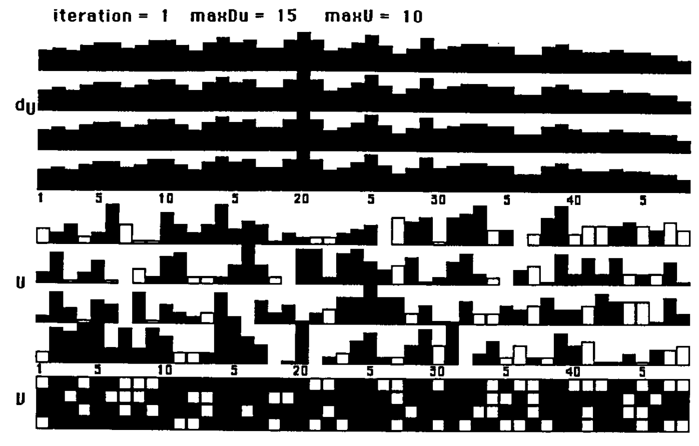
0) Set $t=0$, $\Delta t=1$, and $A=B=C=C_1=C_2=1$.
1) The initial values of $U_{Xi}(t)$ where $X=1,\cdots,n$ and $i=1,2,3,4$ are randomized.
2) Evaluate values of $V_{Xi}(t)$ based on the binary function where $X=1,\cdots,n$ and $i=1,2,3,4$.

$$V_{Xi}(t)=f(U_{Xi}(t))=\begin{cases}1, & \text{if } U_{Xi}(t)>0\\ 0, & \text{otherwise.}\end{cases}$$

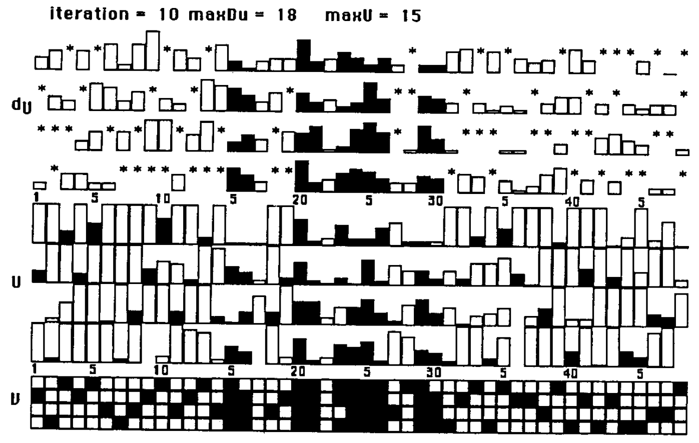
3) Use the motion equation in (3) to compute $\Delta U_{Xi}(t)$.

$$\Delta U_{Xi}(t)=-A\left(\sum_{j=1}^{4}V_{Xj}(t)-1\right)$$
$$-B\sum_{\substack{Y=1\\Y\neq X}}^{n}d_{XY}V_{Yi}(t)\sum_{K=1}^{n}d_{YK}+C\cdot h\left(\sum_{j=1}^{4}V_{Xj}(t)\right)$$
$$\cdot\left(C_1\sum_{K=1}^{n}d_{XK}+C_2\frac{\sum_{K=1}^{n}\sum_{Y=1}^{n}d_{XY}d_{YK}}{\sum_{K=1}^{n}d_{XK}}\right),$$

for $X=1,\cdots,n$ and $i=1,2,3,4$.



(a)



(b)

Fig. 5. (a) The convergence of the 48-state U.S. map neural network to a four-color solution. (a) At the intermediate state of 192 neurons after the first iteration. (b) At the intermediate state of 192 neurons after the tenth iteration. (c) At the intermediate state of 192 neurons after the 30th iteration. (d) At the final state of 192 neurons after the 35th neuron. (The linear dimension of each rectangle is proportional to the value of ΔU_{xi}, U_{xi}, or V_{xi}. Black and white rectangles indicate positive and negative, respectively.)

4) Compute $U_{Xi}(t+1)$ using the first-order Euler method:

$$U_{Xi}(t+1) = U_{Xi}(t) + \Delta U_{Xi}(t)\Delta t,$$

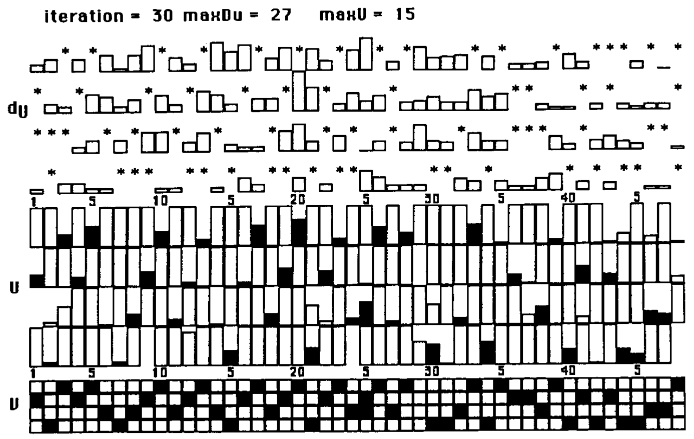
$$\text{for } X = 1,\cdots,n \text{ and } i = 1,2,3,4.$$

5) Increment t by 1. Set $C = 5$ if $((t+15) \bmod 25) < 5$ and $t \geqslant 60$, 1 otherwise. If at least one among $\Delta U_{X1}(t)$, $\Delta U_{X2}(t)$, $\Delta U_{X3}(t)$, and $\Delta U_{X4}(t)$ is equal to zero for $X = 1,\cdots,n$, then terminate the procedure; else go to step 2.

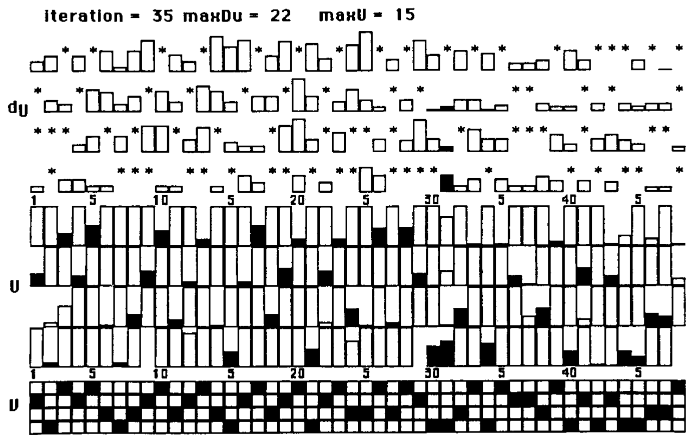
IV. Simulation Results

Fig. 4 shows the U.S. continental map, which consists of 48 states where the adjacency matrix is given by a 48×48 array. Fig. 5(a), (b), (c), and (d) show the convergence of the 48-state U.S. map neural network to a four-color solution, respectively. Fig. 6 depicts the solution that is decoded from the state of 192 neurons in Fig. 5(d). Fig. 7 describes one of the solutions for a 210-country map four-coloring problem where the problem is taken from the example of Appel and Haken's experiments. The state of the system always converged to the global minimum as far as we have observed through more than 1000 simulation runs. Fig. 8 shows the frequency versus the number of iteration steps using several hundred simulation runs in the 210-country map problem. The average number of iteration steps is 820. Fig. 9 shows one of the solutions for a 211-country map problem where the 210-country is surrounded by ocean. The number of iteration steps for the 211-country map problem is similar to the result of the 210-country map problem. Fig. 10 shows one of the solutions for a 430-country map problem. Fig. 11 depicts the frequency versus the number of iteration steps using several hundred simulation runs for the 430-country map problem. The average number of iteration steps is 1000. Figs. 8 and 11 indicate that the problem size does not strongly reflect the number of iteration steps.

Through more than 1000 sets of simulation runs, we have observed that the problem size does not strongly reflect the number of iteration steps to converge to the global minimum.



(c)



(d)

Fig. 5. (*Continued*)

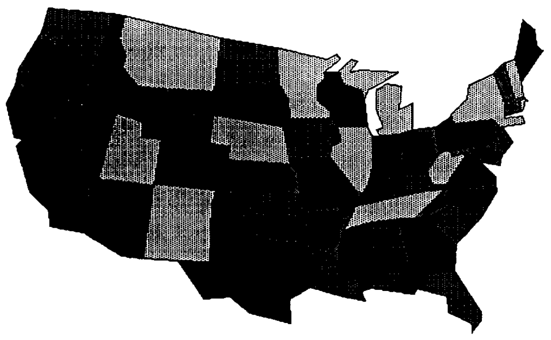


Fig. 6. The solution of the 48-state U.S. map four-color problem, illustrating how the final state V_{xi} in Fig. 5(d) is decoded into a solution.

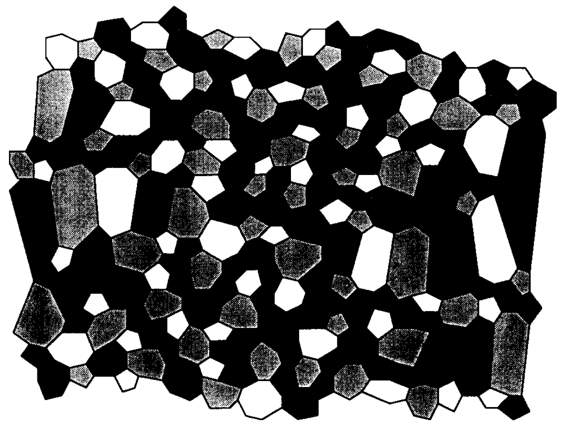


Fig. 7. The solution of the 210-country map problem.

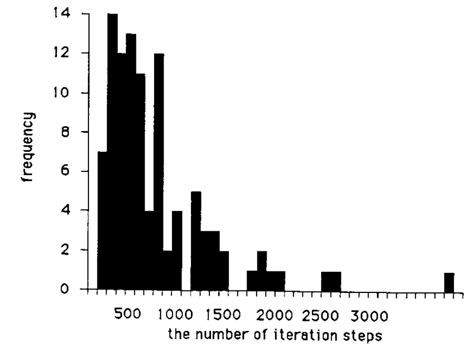


Fig. 8. The relationship between frequency and the number of iteration steps to converge to the global minimum for the 210-country map problem.

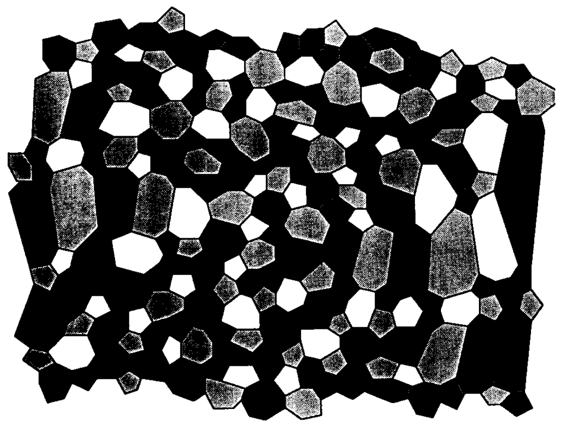


Fig. 9. The solution of the 211-country map problem (210-country is surrounded by white-colored ocean).

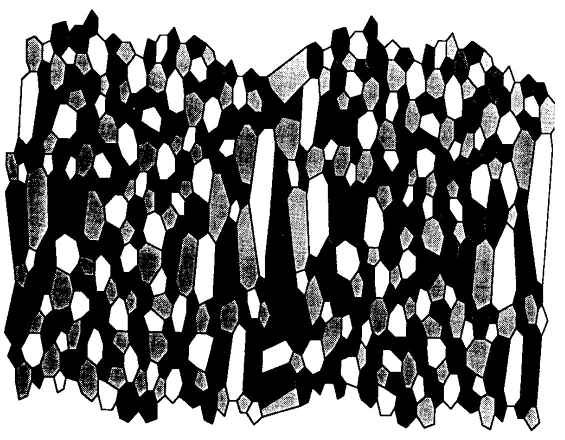


Fig. 10. The solution of the 430-country map problem.

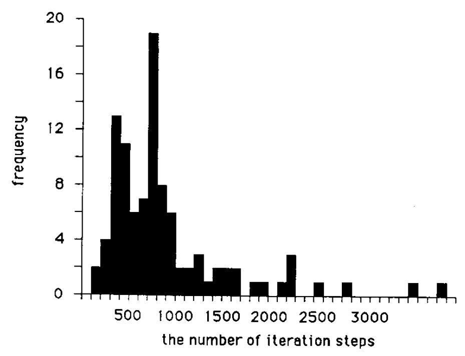


Fig. 11. The relationship between frequency and the number of iteration steps to converge to global minimum for the 430-country map problem.

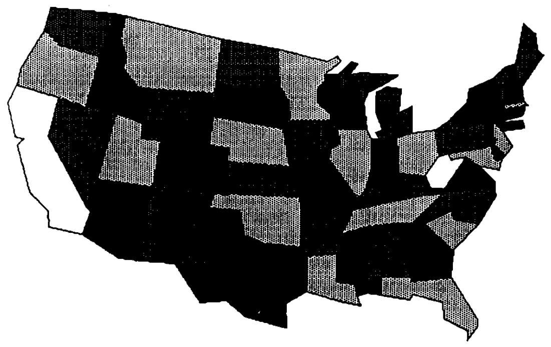


Fig. 12. A three-colorability problem of the 48-state map.

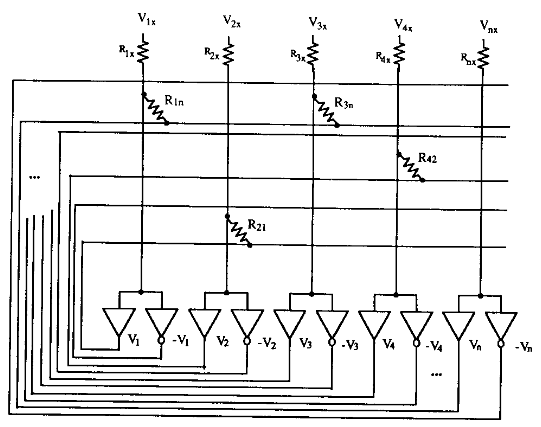


Fig. 13. A circuit diagram of the four-color neural network.

The proposed scaling method for the coefficient in the motion equation is quite effective, which allows the state of the system to escape from the local minimum and to converge to the global minimum.

V. K-Colorability Problem

The Newton equation in (8) is modified for the three-color map problem. For the three-color map problem, a $3 \times n$ neural

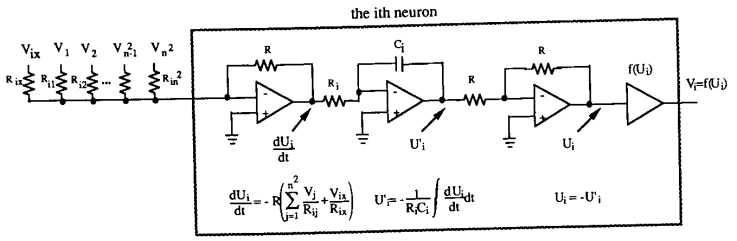

the ith neuron
V_{ix} V_1 V_2 V_{n-1}^2 V_n^2
R_{ix} R_{i1} R_{i2} ... R_{in^2}
R
R_i
C_i
R
R
f(U$_i$)
V_i=f(U$_i$)
$\frac{dU_i}{dt}$
U'$_i$
U$_i$
$\frac{dU_i}{dt} = -R\left(\sum_{j=1}^{n^2} \frac{V_j}{R_{ij}} + \frac{V_{ix}}{R_{ix}}\right)$
$U'_i = -\frac{1}{R_iC_i}\int \frac{dU_i}{dt}dt$
$U_i = -U'_i$

Fig. 14. An analog circuit of the ith neuron.

array is prepared. It is given by

$$\frac{dU_{Xi}}{dt} = -A\left(\sum_{j=1}^{3} V_{Xj} - 1\right) - B\sum_{\substack{Y=1\\Y\neq X}}^{n} d_{XY}V_{Yi}\sum_{K=1}^{n} d_{YK}$$

$$+C\cdot h\left(\sum_{j=1}^{3} V_{Xj}\right)\left(C_1\sum_{K=1}^{n} d_{XK} + C_2\frac{\sum_{K=1}^{n}\sum_{Y=1}^{n} d_{XY}d_{YK}}{\sum_{K=1}^{n} d_{XK}}\right). \quad (9)$$

Fig. 12 shows three-colorability for the 48-state U.S. map problem. Our simulation result shows that there are two conflict states (California and West Virginia) which cannot be colored. Our simulator always indicates where the conflicts exist in a map: a conflict around California and the other conflict around West Virginia. In general, an $m \times n$ neural array is required for the m-color problem where n is the number of regions. K-colorability problems belong to the NP-complete problem [14]. The Newton equation of the ith color of the Xth region for the m-color-n-region problem is given by

$$\frac{dU_{Xi}}{dt} = -A\left(\sum_{j=1}^{m} V_{Xj} - 1\right) - B\sum_{\substack{Y=1\\Y\neq X}}^{n} d_{XY}V_{Yi}\sum_{K=1}^{n} d_{YK}$$

$$+C\cdot h\left(\sum_{j=1}^{m} V_{Xj}\right)\left(C_1\sum_{K=1}^{n} d_{XK} + C_2\frac{\sum_{K=1}^{n}\sum_{Y=1}^{n} d_{XY}d_{YK}}{\sum_{K=1}^{n} d_{XK}}\right). \quad (10)$$

VI. Analog Neural Network

Fig. 13 shows the analog neural network where each neuron performs the Newton equation given by $dU_i/dt = -\partial E/\partial V_i$. In Fig. 14 the detailed neuron circuit is described where the operational amplifier on the first stage sums all of inputs which satisfies the following equation:

$$\frac{dU_i}{dt} = -R\left(\sum_{j=1}^{n^2} \frac{V_j}{R_{ij}} + \frac{V_{ix}}{R_{ix}}\right). \quad (11)$$

The derivatives of the four-color computational energy E with respect to the pth output is given by

$$\frac{dU_p}{dt} = -\sum_{q=1}^{n^2} G_{pq}V_q - I_p \quad (12)$$

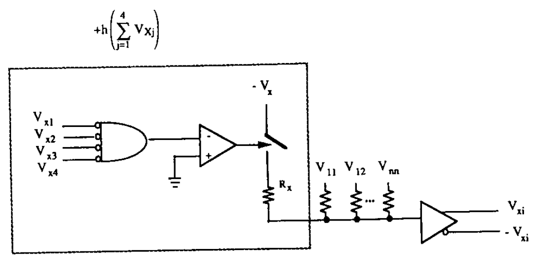

$+h\left(\sum_{j=1}^{4} V_{Xj}\right)$
$-V_x$
V_{x1}
V_{x2}
V_{x3}
V_{x4}
R_x
V_{11} V_{12} V_{nn}
V_{xi}
$-V_{xi}$

Fig. 15. The circuit of the ith neuron in the X state for the non-quadratic function.

where p and q are given by $(4(X-1)+i)$ and $(4(Y-1)+j)$, respectively. G_{pq} is given by

$$G_{pq} = A\delta_{XY}(1-\delta_{ij}) + B\delta_{XY} + Cd_{XY}\delta_{ij}(1-\delta_{XY})$$

$$\text{where } \delta_{rs} = \begin{cases} 1, & \text{if } r = s \\ 0, & \text{otherwise.} \end{cases} \quad (13)$$

From (11) and (12), G_{pq} and I_p in (12) are determined by R/R_{ij} and RV_{ix}/R_{ij}, respectively.

The second operational amplifier in Fig. 14 performs integration of dU_i/dt, which is given by the following equation:

$$U_i' = -\frac{1}{R_iC_i}\int\left(\frac{dU_i}{dt}\right)dt. \quad (14)$$

The third operational amplifier generates U_i, $U_i = -U_i'$. The last component must perform a nondecreasing function such as the sigmoid or the binary function.

The hill-climbing term in the Newton equation can be easily implemented by the circuit described in Fig. 15 when binary neurons are used. If and only if $V_{x1} = V_{x2} = V_{x3} = V_{x4} = 0$, $-V_x$ will be given to the input of the ith color of the X state through the analog switch and the resistor R_x; otherwise, the circuit of the resistor will be cut off.

VII. Conclusion

We introduced the computational energy function for the four-color map problem and the k-color map problem. It was proven that the state of the system with the Newton equation is guaranteed to converge to the local minimum. With the hill-climbing term, the proposed Newton equation forces the state of the system to escape from the local minimum and to converge to the global minimum. A large number of simulation runs was performed to verify our algorithm. The simulation result supports that the problem size does not influence the number of iteration steps. The analog circuit for the four-color map problem was presented in this paper.

References

[1] W. S. McCulloch and W. H. Pitts, "A logical calculus of ideas imminent in nervous activity," *Bull. Math. Biophys.*, vol. 5, 1943.
[2] D. O. Hebb, *The Organization of Behavior.* New York: Wiley, 1949.
[3] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *IREWESCON Convention Rec.*, pp. 25–39, Sept. 1960.
[4] F. Rosenblatt, *Principles of Neurodynamics.* New York: Spartan, 1962.
[5] M. Minsky and S. Papert, *Perceptrons.* Cambridge, MA: MIT Press, 1969.
[6] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Bio. Cybern.*, vol. 52, 141–152, 1985.
[7] G. V. Wilson and G. S. Pawley, "On stability of the travelling salesman problem algorithm of Hopfield and Tank," *Bio. Cybern.*, vol. 58, pp. 58–70, 1988.
[8] Y. Takefuji and K. C. Lee, "A super parallel sorting algorithm based on neural networks," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 1425–1429, Nov. 1990.
[9] K. Appel and W. Haken, "The solution of the four-color-map problem," *Scientific American*, pp. 108–121, Oct. 1977.
[10] E. D. Dahl, "Neural network algorithm for an NP-Complete problem: Map and graph coloring," in *Proc. First Int. Conf. on Neural Networks*, vol. III, pp. 113–120, 1987.
[11] Y. Takefuji and K. C. Lee, "A near-optimum parallel planarization algorithm," *Science*, vol. 245, pp. 1221–1223, Sept. 1989.
[12] ——, "A parallel algorithm for tiling problems," *IEEE Trans. Neural Networks*, vol. 1, pp. 143–145, Mar. 1990.
[13] Y. Takefuji, C. W. Lin, and K. C. Lee, "A parallel algorithm for estimating the secondary structure in ribonucleic acids," *Bio. Cybern.*, vol. 63, pp. 337–340, 1990.
[14] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* San Francisco, CA: Freeman, 1979.
[15] A. Moopenn *et al.*, "A neurocomputer based on an analog-digital hybrid architecture," in *Proc. First Int. Conf. Neural Networks*, vol. III, pp. 479–486, 1987.
[16] A. P. Thakoor *et al.*, "Electronic hardware implementations of neural networks," *Appl. Opt.*, vol. 26, pp. 5085–5092, 1987.

On the Largest Modulus of Polynomial Zeros

Ezra Zeheb

Abstract —**A result by Cauchy is extended in two directions, providing two bounds for the moduli of the zeros of a polynomial. One of these pertains to real polynomials, while the other pertains to polynomials with complex coefficients.**

I. Introduction

The problem of the relation between the zeros of a polynomial and its coefficients is a very old one, but continuously vital. Both facts can be deduced by glancing at the references in the comprehensive book by Marden [1] and by noticing the abundance of recent publications on the subject. The relation between this problem and the analysis, design, and stability considerations of engineering systems is well known. In addition to the classic results by Routh, Hurwitz, Hermite, Schur–Cohn, and Jury–Marden, which provide necessary and sufficient conditions for stability, there are numerous other such conditions or necessary conditions or sufficient conditions. (See references in [2].) The need for simple necessary conditions and for simple sufficient conditions is important where the design calls for quick assessment of stability, or stability margins, sometimes in many iterative steps. It becomes particularly important where multidimensional systems (related to multivariable polynomials) are concerned. Here, the "dimensionality curse" almost prevents the use of necessary and sufficient conditions. For but a few recent necessary or sufficient conditions, or simple necessary and sufficient conditions for special cases, see [3]–[13].[1]

In this note we extend a result by Cauchy [14] in two directions, providing two circular bounds for the zeros of a polynomial.

Manuscript received March 23, 1990. This paper was recommended by Associate Editor P. H. Bauer.

E. Zeheb is with the Electrical Engineering Department, Technion, Israel Institute of Technology, Haifa 32000, Israel.

IEEE Log Number 9041766.

II. A Bound for the Zeros of a Real Polynomial

Let

$$P(z) = z^n + \sum_{i=0}^{n-1} a_i z^i, \qquad a_i \in \mathbb{R} \tag{1}$$

be a real polynomial. A special case[2] of a result due to Cauchy [14] states that all the zeros of $P(z)$ lie in the circle

$$|z| < 1 + A \tag{2}$$

where

$$A = \max_i \{|a_i|\}, \qquad i = 0, \cdots, n-1. \tag{3}$$

The following is an extension to this result.

Theorem 1: All the zeros of $P(z)$ in (1) lie in the circle

$$|z| < 1 + \max\{A_{ij}\} \tag{4}$$

where

$$A_{ij} = \frac{|a_i a_{j-1} - a_j a_{i-1}|}{|a_i| + |a_j|}, \qquad i,j = 0,\cdots,n;\ j > i \tag{5.1}$$

$$a_n \triangleq 1, \qquad a_{-1} \triangleq 0. \tag{5.2}$$

Remarks:

1) In (5.1), the understanding is that i and j take on the values $0,\cdots,n$ $(j > i)$, including values where coefficients may be zero (nonexisting), but excluding values where both $a_i = a_j = 0$.
2) Though Theorem 1 cannot be used to ascertain stability of discrete systems, it can be used to get a quick "feeling" about stability or instability of such a system. Other results of this type are given in [15]–[17].

[1] It is almost impossible, and certainly not intended, to provide a comprehensive list.
[2] Cauchy's result pertains to complex coefficients $a_i \in \mathbb{C}$ as well.