References

[1] B. Widrow *et al.*, "Adaptive noise cancelling: Principles and applications," *Proc. IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

[2] S. F. Boll and D. C. Pulsipher, "Suppression of acoustic noise in speech using two microphone acoustic noise cancellation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 752–753, Dec. 1980.

[3] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 113–120, Apr. 1979.

[4] M. G. Amin, "Adaptive noise cancelling in the spectrum domain," in *Proc. ICASSP*, pp. 2959–2962, 1986.

[5] M. R. Sambur, "Adaptive noise cancelling for speech signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-26, pp. 419–423, Oct. 1978.

[6] W. A. Harrison, J. S. Lim, and E. Singer, "A new application of adaptive noise cancellation," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 21–27, 1986.

[7] W. A. Gardner and B. G. Agee, "Two-stage adaptive noise cancellation for intermittent-signal applications," *IEEE Trans. Inform. Theory*, vol. IT-26, pp. 746–750, Nov. 1980.

[8] W. B. Mikhael and P. D. Hill, "Acoustic noise cancellation in a multiple noise source environment," in *Proc. ISCAS*, pp. 2399–2402, 1988.

[9] M. Feder, A. V. Oppenheim, and E. Weinstein, "Methods for noise cancellation based on the EM algorithm," in *Proc. ICASSP*, pp. 201–202, 1987.

[10] ——, "Maximum likelihood noise cancellation using the EM algorithm," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-37, pp. 204–216, Feb. 1989.

[11] R. A. Goubran, R. Hebert, and H. M. Hafez, "Acoustic noise suppression using regressive adaptive filtering," in *Proc. IEEE Vehicular Tech. Conf.*, pp. 48–53, 1990.

[12] R. A. Goubran and H. M. Hafez, "Background acoustic noise reduction in mobile telephony," in *Proc. IEEE Vehicular Tech. Conf.*, pp. 72–75, 1986.

[13] R. Hebert, R. A. Goubran, and H. M. Hafez, "Analysis of background noise in cars," in *Proc. Can. Conf. Elec. Comp. Eng.*, pp. 21.4.1–4, 1990.

[14] M. M. Goulding and J. S. Bird, "Speech enhancement in small noisy reverberant enclosures," in *Proc. Can. Conf. Elec. Comp. Eng.*, pp. 219–222, 1988.

[15] I. Lecomte *et al.*, "Car noise processing for speech input," in *Proc. ICASSP*, pp. 512–515, 1989.

Neural Network Parallel Computing for BIBD Problems

Takakazu Kurokawa and Yoshiyasu Takefuji

***Abstract*—Neural network parallel computing for balanced incomplete block design (BIBD) problems is presented in this paper. A design in which all the blocks contain the same number of varieties, and all the varieties occur in the same number of blocks, is called a block design. A block is said to be incomplete if it does not contain all the varieties. If a design is balanced, we call it a balanced incomplete block design. BIBD problems are very important for solving problems in experimental design, material relating design, and coding theory. Two methods for BIBD problems have been proposed. One uses the notion of the finite fields, and the other uses the notion of the difference sets. In general, the conventional algorithms are only able to solve the problems that satisfy an affine plane or a finite projective plane. The proposed algorithm is able to solve BIBD problems regardless of the condition of an affine plane or a finite projective plane. The proposed algorithm requires $\binom{v}{k}$ processing elements, or artificial neurons to solve the $[k, 1; v]$-design problem in parallel. The proposed algorithm was verified by a large number of simulation runs. The simulation results demonstrated that the number of iteration steps for the system to converge to the solution increases slightly with the problem size.**

I. Introduction

Balanced incomplete block design (BIBD) [1] plays an important role in experimental design and the modern theory of statistical analysis [2]. It has also been used for solving material relating design and coding theory problems [3]. There is a famous mathematics problem called Kirkman's schoolgirl problem, proposed by T. P. Kirkman in 1850 [4]. Kirkman's original problem is given as follows: A school teacher is in the habit of taking her fifteen girls $V = \{1, 2, \cdots, 15\}$ for a daily walk, always arranging them three abreast in what is called a block and is denoted by B_i for $i = 1, 2, \cdots, 35$ in five rows [4]. The problem is to arrange them so that for seven consecutive days no girl will walk more than once in the same row with any other girl [4]. Fig. 1 shows a solution of Kirkman's problem. The following properties must be satisfied for a collection $\{B_1, B_2, \cdots, B_{35}\}$ in a BIBD, or (15, 35, 7, 3, 1)-design [2]:

(a) no variety appears more than once in a block where every block contains three varieties;
(b) every variety appears equally often;
(c) every pair of varieties appears in the same number of blocks.

This paper introduces a new parallel algorithm for solving a (v, b, r, k, λ)-BIBD design problem where the existing algorithms cannot solve it if neither an affine plane nor a finite projective plane is satisfied. The affine plane and the finite projective plane are explained in Section II. The existing algorithms may have to use a prohibitively expensive exhaustive search technique on the condition. Our algorithm uses $\binom{v}{k}$ processing elements called the McCulloch–Pitts binary neurons. The input/output function of the ith McCulloch–Pitts neuron at time t $V_i(t)$ is given by

$$\begin{aligned} V_i(t) &= 1, \qquad \text{if } U_1(t) > 0 \\ &\ 0, \qquad \text{otherwise for } i = 1, 2, \cdots, \binom{v}{k}. \end{aligned} \tag{1}$$

Note that $U_i(t)$ is the input of the ith neuron. The first artificial neural network for solving combinatorial optimization problems was introduced by Hopfield and Tank [5] where they used the sigmoid neurons. It has been empirically demonstrated that the convergence speed of the McCulloch–Pitts neural network is relatively faster than that of the sigmoid neural network by solving graph planarization problems [6], tiling problems [7], sorting problems [8], [9], predicting the secondary structure in ribonucleic acids [10], [11], finding the maximum independent set [11], searching problems [12], four-coloring and k-coloring problems [13], spare allocation problems [14], channel routing problems [15], crossbar switch scheduling problems [16], knights tour problems [17], Hip games [18], traffic control problems [19], [20], and broadcast scheduling problems [21].

It is proven that the state of the McCulloch–Pitts binary neural network is always allowed to converge to the local minimum as long as the motion equation of the McCulloch–Pitts ith neuron is given by $dU_i/dt = -dE/dV_i$ [9]. In other words, it is shown that the

Manuscript received July 2, 1990; revised June 9, 1991. This paper was recommended by Associate Editor A. Kuh.

T. Kurokawa is with the Department of Computer Science, National Defense Academy, Yokosuka 239, Japan.

Y. Takefuji is with the Department of Electrical Engineering and Applied Physics, Case Western University, Cleveland, OH 44106.

IEEE Log Number 9107121.

1057-7130/92$03.00 © 1992 IEEE

Sun.		Mon.		Tues.		Wed.		Thurs.		Fri.		Sat.	
1 2 3	B_1	1 4 5	B_6	1 6 7	B_{11}	1 8 9	B_{16}	1 10 11	B_{21}	1 12 13	B_{26}	1 14 15	B_{31}
4 8 12	B_2	2 8 10	B_7	2 9 11	B_{12}	2 12 15	B_{17}	2 13 14	B_{22}	2 4 6	B_{27}	2 5 7	B_{32}
5 10 14	B_3	3 13 15	B_8	3 12 14	B_{13}	3 5 6	B_{18}	3 4 7	B_{23}	3 9 10	B_{28}	3 8 11	B_{33}
6 11 13	B_4	6 9 14	B_9	4 10 15	B_{14}	4 11 14	B_{19}	5 9 12	B_{24}	5 11 15	B_{29}	4 9 13	B_{34}
7 9 15	B_5	7 11 12	B_{10}	5 8 13	B_{15}	7 10 13	B_{20}	6 8 15	B_{25}	7 8 14	B_{30}	6 10 12	B_{35}

Fig. 1. A solution of Kirkman's schoolgirl problem.

computational energy function $E = E(V_1, V_2, \cdots, V_{\binom{v}{k}})$ monotonically decreases as long as the ith neuron in the McCulloch–Pitts neural network obeys $dU_i/dt = -dE/dV_i$ [13].

The mathematical proof of the global minimum convergence of the McCulloch–Pitts neural network has never been given although the massive empirical simulation experiments in many problems have shown promising result [6]–[21]. Interconnection weights or synaptic weights between neurons are simply determined by the computational energy function $E(V_1, V_2, \cdots, V_{\binom{v}{k}})$. In order to obtain the energy function $E(V_1, V_2, \cdots, V_{\binom{v}{k}})$, we can use the RHS of the motion equation:

$$\int (\text{RHS of the motion equation})\, dV_i = \int (-dE/dV_i)\, dV_i = -\int dE = -E.$$

If E follows the quadratic form, then the interconnection weight T_{ij} between the ith and the jth neurons will be given by

$$E = \sum_{i=1}^{\binom{v}{k}} \sum_{j=1}^{\binom{v}{k}} T_{ij} V_i V_j + \sum_{i=1}^{\binom{v}{k}} I_i V_i.$$

II. The BIBD Problem

In general, three properties must be satisfied for (v, b, r, k, λ)-design.

a) Each block B_i $(i = 1, 2, \cdots, b)$ contains k varieties where k is a fixed constant;

b) each variety $x \in V$ appears $r (\leq b)$ times in the blocks;

c) every pair of varieties appears $\lambda (\leq b)$ times in the blocks.

Two relationships among the five parameters v, b, r, k, and λ are given by [2]:

1) $$v \cdot r = b \cdot k$$

2) $$\lambda \cdot (v - 1) = r \cdot (k - 1).$$

A (v, b, r, k, λ)-design can be expressed by a $[k, \lambda; v]$-design in an abbreviation form. Fisher and Yates have studied the design of experiments in the modern theory of statistical analysis [22]–[24]. Bose presented several mathematical methods to construct block design [25]–[27]. Grimaldi [2] determined the necessary condition for $[k, \lambda; v]$-design and showed how such designs could be generated by using finite fields. All of these research results are summarized in [28].

Over more than half a century, two basic deterministic methods have been proposed for BIBD problems. One uses the notion of the finite fields, and the other uses the notion of the difference sets. In general, the existing methods are only able to solve problems that satisfy an affine plane or a finite projective plane. If a problem satisfies neither an affine plane nor a finite projective plane, an exhaustive search must be performed. The method using the affine plane $AG(n, q)$ is able to solve the $[q, 1; q^n]$-design problem where q must be a power of a prime number and n must be a positive integer. Another method using the finite projective plane $PG(n, q)$ is able to solve the $[\mathbf{q + 1, 1}; (q^{n+1} - 1)/(q - 1)]$-design problems. In both methods, points in planes must accord with varieties and lines must accord with blocks of a design. If k and v satisfy neither the AG plane nor the PG plane, the $[\boldsymbol{k}, \lambda; \boldsymbol{v}]$-design problem cannot be solved by the conventional methods although solutions may exist.

For example, it is impossible to solve the $[\mathbf{3, 1; 13}]$-design problem with the finite fields, because it satisfies neither the affine plane nor the finite projective plane. Another method using the notion of the difference set provides us the existence of a solution. But it does not give us the solution. The conventional algorithms require exhaustive search to find an initial set of the rotation sets [1]. As far as we know, no parallel algorithms have been proposed for BIBD design problems. The proposed heuristic algorithm is able to solve the BIBD problem regardless of the condition of the affine plane or the finite projective plane.

III. BIBD Neural Network

To solve a $[k, \lambda; v]$-design problem, $\binom{v}{k}$ binary neurons are required in our system. Each neuron corresponding to a block candidate represents a possible combination of k elements out of v varieties. Consider an example of the BIBD problem: a [3, 1; 7]-design problem. There are seven varieties, which we denote $1, 2, \cdots, 7$ here. The number of possible candidates to make a block is given by $\binom{7}{3} = 35$. The state of a single neuron with a black square or a white square in Fig. 2 indicates the selection of the corresponding block. One of the solutions in the [3, 1; 7]-design problem is (1, 2, 3) (1, 4, 7) (1, 5, 6) (2, 4, 6) (2, 5, 7) (3, 4, 5) (3, 6, 7), and the selected blocks are shown by the black squares in Fig. 2.

A circuit diagram of the neural network for the [3, 1; 7]-design problem is depicted in Fig. 3. In Fig. 3, each neuron is represented by an inverter symbol. The output of each neuron performs an inhibitory force to avoid the other candidates with the same pair of varieties. The input of a neuron is connected to the outputs of those neurons that have at least the same pair of varieties as shown in Fig. 3. For example, the input of the leftmost #1 neuron is connected to the outputs of #2, #3, $\cdots$, #9, #16, #17, #18, #19 neurons, because all of these neurons have at least the same pair of varieties, such as (1, 2),(1, 3), or (2, 3).

The output of the ith neuron V_i for $i = 1, 2, \cdots, \binom{v}{k}$ is given by

$$\frac{dU_i}{dt} = -A\left(\sum_{\substack{j=1\\ j\neq i}}^{\binom{v}{k}} f(N_j, N_i) \cdot V_j - (\lambda - 1)\right) + Bh\left(\sum_{\substack{j=1\\ j\neq i}}^{\binom{v}{k}} f(N_j, N_i) \cdot V_j - (\lambda - 1)\right) \qquad (2)$$

where the function $f(x, y)$ is 1 if an x-block and a y-block have the same pair of varieties, and 0 otherwise. A and B are constant integers. The function $h(x)$ performs deterministic hill-climbing where the function is given by

$$h(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{otherwise.} \end{cases} \qquad (3)$$

The deterministic hill-climbing function has been used in many applications [6]–[21]. The hill-climbing term was first introduced in

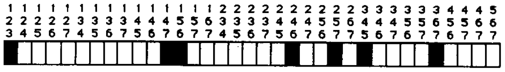


Fig. 2. Neural network representation for a [3, 1; 7]-design problem.

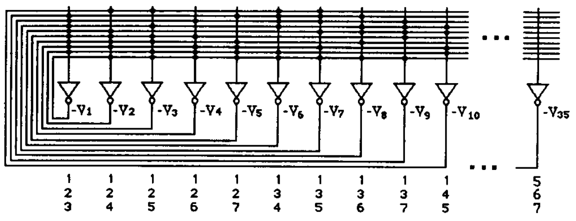


Fig. 3. A circuit diagram of a [3, 1,; 7]-design problem.

tiling problems [7]. It has been empirically demonstrated that the hill-climbing term not only helps the state of the system to escape from the local minimum and increases the frequency to converge to the global minimum [7], [13]. The first term in (1) is an inhibitory force in order to avoid any ($\lambda + 1$) blocks with the same pair of varieties to be selected in a design.

The second term is an excitatory force to select the ith block if the combination of k elements in the ith blocks causes no conflict with the selected λ blocks in a design. Fig. 4 shows a complete circuit diagram of #1 neuron as shown in Fig. 3 to solve the [3, 1; 7]-design problem.

The entire circuit is composed of four components. Fig. 4(a) shows a circuit performing a summing operation

$$\sum_{\substack{j=1\\ j\neq 1}}^{35} f(N_j, N_1) \cdot V_j.$$

The function $f(N_j, N_1)$ is realized by the connectivity of the output V_j through a 10-kΩ resistor. The operational amplifiers perform a summing operation to determine the output voltage V_o. V_o is given by $V_o = V_2 + V_3 + \cdots + V_{19}$. In Fig. 4(b), the operational amplifiers subtract the value ($\lambda - 1$) from V_o and generate the calculation result given by V_o'. In Fig. 4(c), a circuit to compute the value of dU_1/dt discussed in (2) is shown. The input of this circuit is V_o' from the circuit as shown in Fig. 4(b). An inverting amplifier shown in the upper left part in Fig. 4(c) multiplies its input by A. The lower left components composed of two inverting amplifiers, two diodes, a transistor, and seven resistors perform the binary function in (3). The output of this component is multiplied by B through a noninverting amplifier. Two outputs of an inverting and a noninverting amplifier are summed by the rightmost two operational amplifiers. The output of this circuit generates dU_1/dt. In Fig. 4(d), the first operational amplifier performs integration that is given by

$$-\frac{1}{CR}\int\left(\frac{dU_1}{dt}\right)dt = -\frac{U_1}{CR} = -\frac{U_1}{47}.$$

The second operational amplifier generates U_1: $-(-U_1/CR)R' = U_1$. The last operational amplifier in Fig. 4(d) performs the binary function in (1).

The following procedure describes the proposed parallel algorithm.

[*Step 0*] Set $t = 0$.

[*Step 1*] The states of inputs $U_i(t)$ for $i = 1, 2, \cdots, \binom{v}{k}$ are set to the small negative number or randomized.

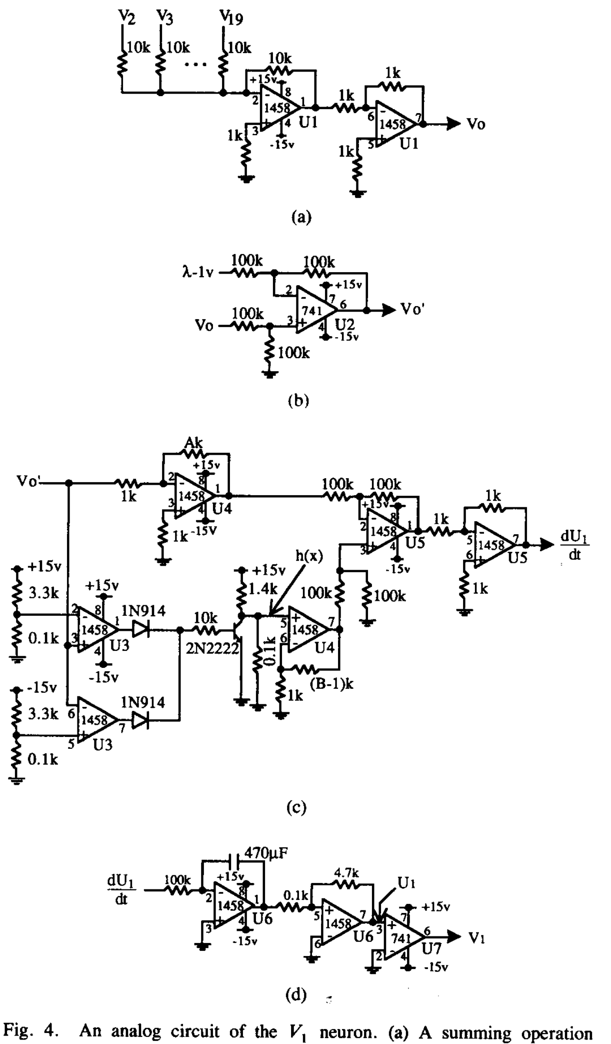


Fig. 4. An analog circuit of the V_1 neuron. (a) A summing operation circuit. (b) A difference operation circuit. (c) A dU_1/dt operation circuit. (d) An integrator and a $V1$ operation circuit.

[*Step 2*] Evaluate values of $V_i(t)$ for $i = 1, 2, \cdots, \binom{v}{k}$ based on the binary function in (1).

[*Step 3*] Use the motion equation in (2) to compute $\Delta U_i(t)$.

$$\Delta U_i(t) = -A\left(\sum_{\substack{j=1\\ j\neq i}}^{\binom{v}{k}} f(N_j, N_i) \cdot V_j - (\lambda - 1)\right) + Bh\left(\sum_{\substack{j=1\\ j\neq i}}^{\binom{v}{k}} f(N_j, N_i) \cdot V_j - (\lambda - 1)\right).$$

[*Step 4*] Compute $U_i(t+1)$ on the basis of the first-order Euler method:

$$U_i(t+1) = U_i(t) + \Delta U_i(t), \qquad \text{for } i = 1, 2, \cdots, \binom{v}{k}.$$

[*Step 5*] Increment t by 1. If the number of fired neuron equals to $b(= \lambda v(v-1)/k(k-1))$ and if the generated

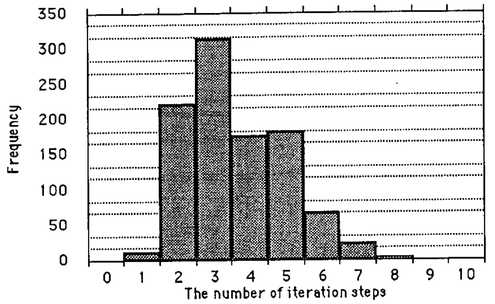


Fig. 5. The relationship between the frequency and the number of iteration steps to converge to global minimum in the [3, 1; 7]-design problem.

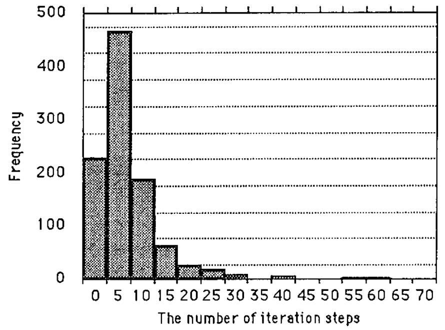


Fig. 6. The relationship between the frequency and the number of iteration steps to converge to global minimum in the [3, 1; 9]-design problem.

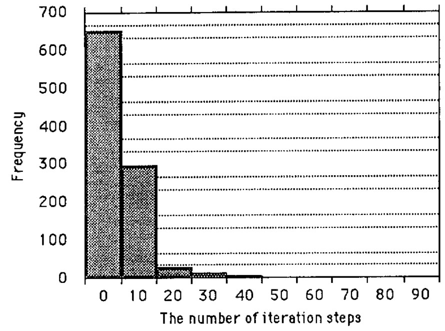
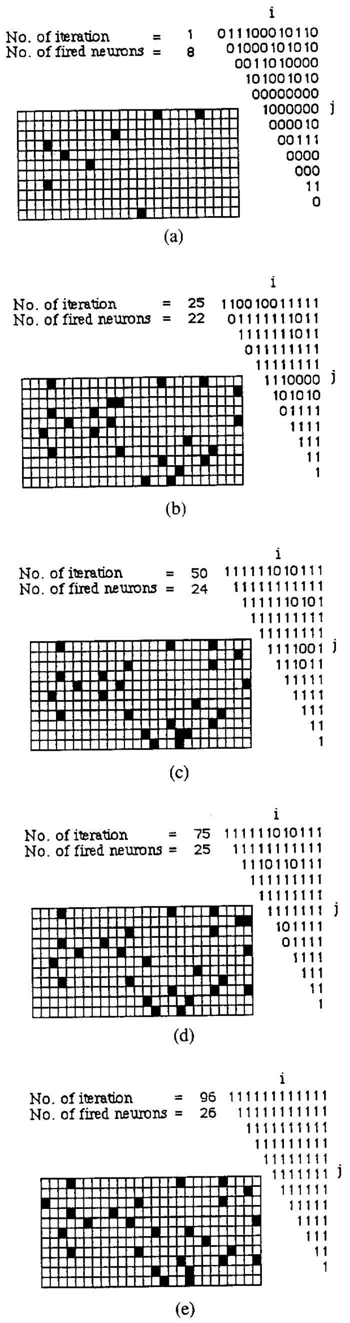


Fig. 7. The relationship between the frequency and the number of iteration steps to converge to global minimum in the [4, 1; 13]-design problem.



```
                                   i
No. of iteration     =  1   011100010110
No. of fired neurons =  8    01000101010
                              0011010000
                               101001010
                                00000000
                                 1000000  j
                                  000010
                                   00111
                                    0000
                                     000
                                      11
                                       0
```

(a)

```
                                   i
No. of iteration     = 25   110010011111
No. of fired neurons = 22    01111111011
                              1111111011
                               011111111
                                11111111
                                 1110000  j
                                  101010
                                   01111
                                    1111
                                     111
                                      11
                                       1
```

(b)

```
                                   i
No. of iteration     = 50   111111010111
No. of fired neurons = 24    11111111111
                              1111110101
                               111111111
                                11111111
                                 1111001  j
                                  111011
                                   11111
                                    1111
                                     111
                                      11
                                       1
```

(c)

```
                                   i
No. of iteration     = 75   111111010111
No. of fired neurons = 25    11111111111
                              1110110111
                               111111111
                                11111111
                                 1111111  j
                                  101111
                                   01111
                                    1111
                                     111
                                      11
                                       1
```

(d)

```
                                   i
No. of iteration     = 96   111111111111
No. of fired neurons = 26    11111111111
                              1111111111
                               111111111
                                11111111
                                 1111111  j
                                  111111
                                   11111
                                    1111
                                     111
                                      11
                                       1
```

(e)

Fig. 8. The simulation result of a [3, 1; 13]-design problem. (a) The state of the neurons after the first iteration. (b) The state of the neurons after the 25th iteration. (c) The state of the neurons after the 50th iteration. (d) The state of the neurons after the 75th iteration. (e) The state of the neurons in the final stage.

element pairs are all unique, then terminate this procedure, else go to step 2.

The simulator has been developed on a Macintosh SE/30. The relationships between the frequency and the number of iteration steps to converge to the global minimum for [3, 1; 7], [3, 1; 9], and [4, 1; 13]-design problems are shown in Figs. 5, 6, and 7, respectively. One thousand simulation runs were performed for each one of three problems. In each one of the simulation runs the initial state of the system was randomly generated. Our simulation result depicts that the number of iteration steps and the solution are affected by the initial state of the system, while the state of the system always converges to the solution within 100 iteration steps. The number of possible candidates or the complexity of the searching space of three problems is given by $\binom{35}{7} = 6.7 \times 10^6$, $\binom{84}{12} = 1.1 \times 10^{14}$, and $\binom{715}{13} = 1.8 \times 10^{27}$, respectively. The following parameters were used in our simulations: $A = 1$, $B = k - 1$ if the number of blocks selected in the design is greater than $2/3$ of b, otherwise $B = k - 2$.

Our algorithm solved the [3, 1; 15]-design problem in 159 iteration steps, which is the famous Kirkman's schoolgirl problem. The number of possible candidates, the complexity of the searching space is $\binom{455}{35} = 2.7 \times 10^{52}$. Note that the mass of a mountain is about 10^{43} times that of an electron [29]. Furthermore, the [3, 1; 13]-design problem (which cannot be solved by the conventional algorithms using finite fields) was solved by our simulator. Fig. 8 shows the simulation result of the problem. In Fig. 8, the state of 286 neurons is depicted by a 11×26 rectangular array where the black squares indicate the selected blocks. In the triangular array, the coordinate (i, j) for $i = 2, 3, \cdots, 13$, and $j = 1, 2, \cdots, 12$ indicates whether a pair of variety i and j exists ("1") or not ("0"). One of the solutions of this problem is (1, 2, 6) (1, 3, 9) (1, 4, 5) (1, 7, 13) (1, 8, 10) (1, 11, 12) (2, 3, 11) (2, 4, 10) (2, 5, 7) (2, 8, 12) (2, 9, 13) (3, 4, 13) (3, 5, 8) (3, 6, 12) (3, 7, 10) (4, 6, 8) (4, 7, 12) (4, 9, 11) (5, 6, 9) (5, 10, 11) (5, 12, 13) (6, 7, 11) (6, 10, 13) (7, 8, 9) (8, 11, 13) (9, 10, 12), where the variety is numbered from 1 to 13. The complexity of the searching space is $\binom{286}{26} = 5.6 \times 10^{36}$. Note that the mass of a man is about 10^{31} times that of an electron [29].

IV. Conclusion

A new parallel algorithm to solve the BIBD problems using the McCulloch–Pitts binary neurons and its analog circuits were presented. Through a large number of simulation runs, the state of the system converges to the solution within several hundred iteration steps. The simulation results show that the number of iteration steps increases slightly with the increase of problem size. A [3, 1; 13]-BIBD problem (which cannot be solved by the conventional methods) was solved by our simulator. We believe that the neural network approach provides one of the most excellent tools for searching a solution among a large space.

References

[1] A. P. Street and D. J. Street, *Combinatorics of Experimental Design*. Oxford: Clarendon Press, 1987.

[2] R. P. Grimaldi, *Discrete and Combinatorial Mathematics*. Reading, MA: Addison-Wesley, 1985.

[3] A. P. Street and W. D. Wallis, *Combinatorial Theory: An Introduction*. Winnipeg, Canada: Charles Babbage Research Center, 1977.

[4] M. Gardner, *New Mathematical Diversions from Scientific American*. Chicago: University of Chicago Press, 1966.

[5] J. Hopfield and D. Tank, "Neural Computation of decisions in optimization problems," *Bio. Cybernet.*, vol. 52, pp. 141-152, 1985.

[6] Y. Takefuji and K. C. Lee, "A Near-optimum parallel planarization algorithm," *Science*, vol. 245, pp. 1221-1223, Sept. 1989.

[7] ——, "A parallel algorithm for tiling problems," *IEEE Trans. Neural Networks*, vol. 1, pp. 143-145, Mar. 1990.

[8] ——, "A two-step parallel sorting algorithm based on neural networks," *J. Neural Network Computing*, 1990.

[9] ——, "A super parallel sorting algorithm based on neural networks," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 1425-1429, Nov. 1990.

[10] Y. Takefuji, C. W. Lin, and K. C. Lee, "A parallel algorithm for estimating the secondary structure in ribonucleic acids," *Biol. Cybern.*, vol. 63, pp. 337-340, 1990.

[11] Y. Takefuji, L. L. Chen, K. C. Lee, and J. Huffman, "Parallel algorithms for finding a near-maximum independent set of a circle graph," *IEEE Trans. Neural Networks*, vol. 1, pp. 263-267, Sept. 1990.

[12] Y. Takefuji, T. Tanaka, and K. C. Lee, "A parallel string search algorithm," *IEEE Trans. Systems, Man, Cybern.*, to be published.

[13] Y. Takefuji and K. C. Lee, "Artificial neural networks for four-coloring map problems and *K*-colorability problems," *IEEE Trans. Circuits Syst.*, vol. 38, pp. 326-333, Mar. 1991.

[14] N. Funabiki and Y. Takefuji, "A parallel algorithm for spare allocation problems," *IEEE Trans. Reliability*, vol. 40, pp. 338-346, 1991.

[15] ——, "A parallel algorithm for channel routing problems," *IEEE Trans. Computer-Aided Design*, to be published.

[16] Y. Takefuji and K. C. Lee, "An artificial hysteresis binary neuron: A model suppressing the oscillatory behaviors of neural dynamics," *Bio. Cybern.*, vol. 64, pp. 353-356, 1991.

[17] T. Kurokawa *et al.*, "CMOS layout design of the hysteresis McCulloch-Pitts neuron," *Electron. Lett.*, vol. 26, pp. 2093-2095, Dec. 1990.

[18] N. Funabiki and Y. Takefuji, "A parallel algorithm for solving "Hip" games," *Neurocomputing*, vol. 3, pp. 97-106, 1991.

[19] ——, "A parallel algorithm for traffic control problems in TDM hierarchical switching systems," *IEEE Trans. Commun.*, to be published.

[20] ——, "A parallel algorithm for traffic control problems in three-stage connecting networks," *J. Parallel Distrib. Comput.*, to be published.

[21] ——, "A parallel algorithm for broadcast scheduling problems in packet radio networks," *IEEE Trans. Commun.*, to be published.

[22] R. A. Fisher, "The arrangement of field experiments," *J. Minis. Agric.*, vol. 33, pp. 503-513, 1926.

[23] R. A. Fisher and F. Yates, *Statistical Tables for Biological, Agricultural and Medical Research*. Oliver and Boyd, 1938.

[24] R. A. Fisher, "An examination of the different possible solutions of a problem in incomplete blocks," *Ann. Eugen*, vol. 10, pp. 52-75, 1940.

[25] R. C. Bose, "On the construction of balanced incomplete block designs," *Ann. Eugenics*, vol. 9, pp. 353-399, 1939.

[26] ——, "On some new series of balanced incomplete block designs," *Bull. Calcutta Math. Soc.*, vol. 34, pp. 17-31, 1942.

[27] R. C. Bose and S. S. Shrikhande, "On the composition of balanced incomplete block designs," *Canad. J. Math.*, vol. 12, pp. 177-188, 1960.

[28] I. Anderson, *Combinatorial Designs: Construction Methods*. Ellis Horwood Limited, 1990.

[29] R. K. Adair, *The Great Design: Particles, Fields, and Creation*. London: Oxford University Press, 1989.

Dynamic Hysteresis of the RTD Folding Circuit and its Limitation on the A/D Converter

Sen-Jung Wei, Hung Chang Lin, Robert C. Potter, and Dave Shupe

Abstract—**Dynamic hysteresis of the resonant tunneling diode (RTD) folding circuit is analyzed. From the study of the dynamic hysteresis effect, the limiting factors for folding type A/D converters are investigated. A 4-bit A/D converter using 4-peak RTD's is simulated. By designing the RTD folding circuit carefully, our simulations show that a 4-bit A/D converter could have a 2-GHz signal bandwidth.**

I. Introduction

Due to their very fast intrinisic response, resonant tunneling diodes (RTD's) have been considered for high-speed applications.

Manuscript received February 25, 1991; revised May 29, 1991. This work was supported in part by DARPA through the Office of Naval Research under Contract N00014-90-C-0208, and by Maryland Industrial Partnerships. This paper was recommended by Associate Editor R. K. Hester.

S.-J. Wei and H. C. Lin are with the Department of Electrical Engineering, University of Maryland, College Park, MD 20472.

R. C. Potter and D. Shupe are with Allied-Signal Aerospace Technology Center, Columbia, MD 21045.

IEEE Log Number 9107708.

1057-7130/92$03.00 © 1992 IEEE