Neurocomputing 3 (1991) 97–106
Elsevier

A parallel algorithm for solving the 'Hip' games

Nobuo Funabiki* and Yoshiyasu Takefuji
Department of Electrical Engineering and Applied Physics, Case Western Reserve University, Cleveland, Ohio 44106, USA

Abstract

Funabiki, N. and Y. Takefuji, A parallel algorithm for solving the 'Hip' games, Neurocomputing 3 (1991) 97–106

A parallel algorithm for solving the 'Hip' games based on an artificial neural network model is presented in this paper. The game of 'Hip' is named because of the hipster's reputed disdain for 'squares'. The rule of the game is to place the counters on a checkerboard so that four of them do not mark the corners of a square. The square may be of any size and be tipped at any angle. Normally this game is played by two players, where the game on a six-by-six checkerboard is the maximum size for the solution. The solution means that every player can place all the counters on the checkerboard without violations. In other words, the goal of our algorithm is to find the pattern of a draw game between players where they should not mark the corners of a square. In order to enlarge the size of the checkerboard where a solution exists, we modified the game as $n/2$ players play on an n-by-n checkerboard where n is an even number. The proposed parallel algorithm requires $m \times n^2$ processing elements for the m-player-n-by-n-checkerboard game to find the solution of the 'Hip' games. The algorithm is verified by solving six games where the size of the checkerboard is varied from 4 to 12.

Keywords. Parallel algorithm; artificial neural network; modified McCulloch-Pitts neuron model; Hip game: draw game pattern.

1. Introduction

Martin Gardner introduced the game of 'Hip' where each player should place the counters on a checkerboard so that any four of them do not mark the corners of a square [1]. The square may be of any size and be tipped at any angle. *Figure 1* shows four of the squares on a six-by-six checkerboard where there are 105 possible squares. The number of different squares on an n-by-n checkerboard is given by $(n^4 - n^2)/12$ [2]. If a player makes a square, he loses the game. The goal of our algorithm is to find the solution of this game. The solution means that each player can place all the counters on a checkerboard without making a square. In other

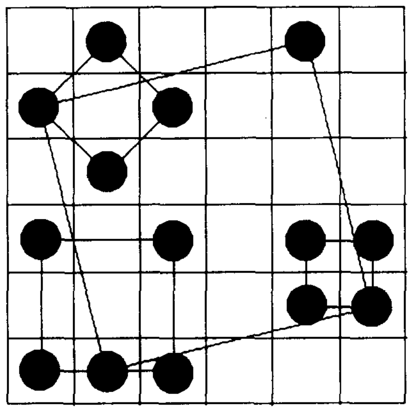


Fig. 1 Four of the 105 squares on 6-by-6 checkerboard.

* Funabiki is also with the system engineering division, Sumitomo Metal Industries, Ltd., Ibaraki, Japan 314.

words, the goal of our algorithm is to find the pattern of a draw game between players where they should not mark the corners of a square. Normally this game is played by two players, but in this case it was proved by R.I. Jewett in 1960 that the game on a six-by-six checkerboard is the maximum size where a solution exists. Therefore we modified the game so that larger than six-by-six checkerboards have a solution. The modified game is that $n/2$ players play on an n-by-n checkerboard where n is an even number. We propose a parallel algorithm which generates solutions in the conventional two-player games and in the modified games.

The first solution or the draw game pattern for the two-player-six-by-six-checkerboard game was discovered by C.M. McLaury. Then D.H. Templeton proposed a simple symmetry strategy for the second player to win this game or to find the draw game pattern. The strategy is that the second player should place his counter on the cell which is rotated by 90 degrees around the center of the checkerboard from the cell on which his opponent placed his last counter. In 1963 W.W. Massie devised an algorithm for the solution of the 'Hip' game by using this strategy. In the algorithm he used a random number to choose the cell on which the first player should place his counter. Therefore the algorithm does not guarantee to find the solution. No parallel algorithm has been reported in the last three decades.

In this paper we propose a parallel algorithm to find the solutions in the 'Hip' games which is based on an artificial neural network model. The artificial neural network model uses a large number of simple processing elements which are called neurons because they perform the function of the simplified biological neurons. The artificial neural network model for solving combinatorial optimization problems was first introduced by Hopfield and Tank [3]. The artificial neural network model has been successfully applied for several NP-complete and optimization problems [4–14].

The output V_{ijk} of the ijkth processing element based on the modified McCulloch–Pitts neuron model [15] follows:

$$V_{ijk} = 1 \text{ if } U_{ijk} > 0 \text{ and } U_{ijk} = \max\{U_{ijr}\} \quad \text{for } r = 1, \ldots, m$$
$$= 0 \text{ otherwise}, \tag{1}$$

where U_{ijk} is the input of the ijkth processing element and m is the number of players. The change of the input U_{ijk} is given by the partial derivatives of the computational energy E with respect to the output V_{ijk}. E is an $n^2 \times m$-variable function: $E(V_{111}, V_{112}, \ldots, V_{nnm})$ where n is the size of the checkerboards. The equation is called a motion equation or a Newton equation. It is given by:

$$\frac{dU_{ijk}}{dt} = -\frac{\partial E(V_{111}, V_{112}, \ldots, V_{nnm})}{\partial V_{ijk}} . \tag{2}$$

In whatever form the computational energy function E is given, the motion equation forces it to monotonically decrease. The following proof shows that the motion equation forces 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.

$$\frac{dE}{dt} = \sum_i \sum_j \sum_k \frac{dV_{ijk}}{dt} \frac{\partial E}{\partial V_{ijk}}$$
$$= \sum_i \sum_j \sum_k \frac{dV_{ijk}}{dt} \left(-\frac{dU_{ijk}}{dt}\right)$$

where the motion equation replaces $(\partial E/\partial V_{ijk})$

by

$$\left(-\frac{\mathrm{d}U_{ijk}}{\mathrm{d}t}\right),$$

$$\frac{\mathrm{d}E}{\mathrm{d}t} = -\sum_i\sum_j\sum_k\left(\frac{\mathrm{d}U_{ijk}}{\mathrm{d}t}\,\frac{\mathrm{d}V_{ijk}}{\mathrm{d}U_{ijk}}\right)\left(\frac{\mathrm{d}U_{ijk}}{\mathrm{d}t}\right)$$

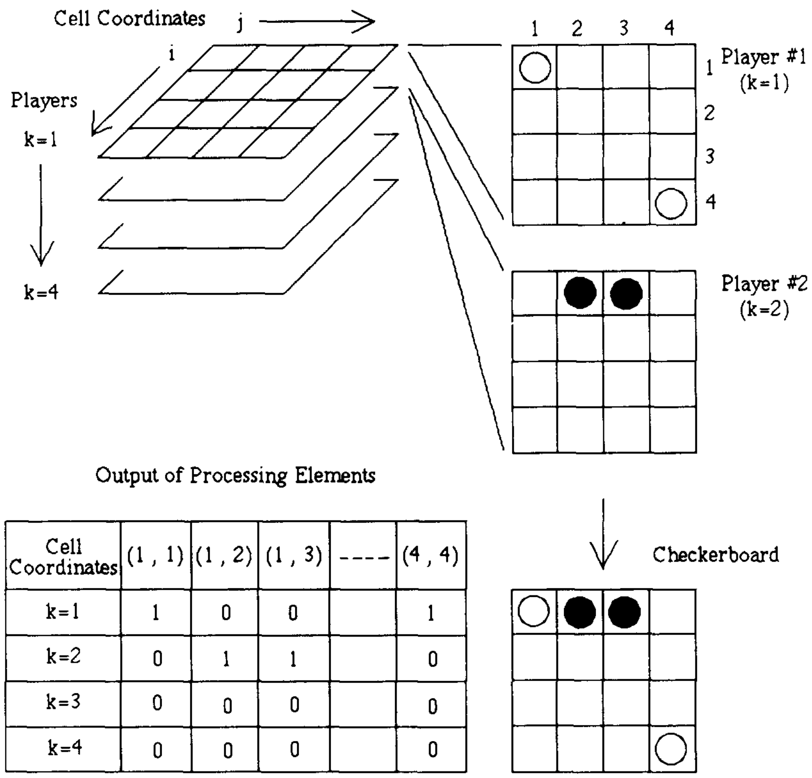
$$= -\sum_i\sum_j\sum_k\left(\frac{\mathrm{d}V_{ijk}}{\mathrm{d}U_{ijk}}\right)\left(\frac{\mathrm{d}U_{ijk}}{\mathrm{d}t}\right)^2 \leqslant 0\,. \qquad (3)$$

As long as the input/output function of the processing elements obeys the nondecreasing function, $\mathrm{d}V_{ijk}/\mathrm{d}U_{ijk}$ must be positive or zero so that $\mathrm{d}E/\mathrm{d}t$ is negative or zero. Therefore the state of the system is always guaranteed to converge to the local minimum. □

We verified our parallel algorithm through solving the six problems where the size of the checkerboard is varied from 4 to 12. The simulation results are also shown and discussed in this paper.

2. System representation

Figure 2 shows the system representation to find the solutions in the 'Hip' game by four players on a four-by-four checkerboard. Four processing elements are used to describe which player should occupy one cell on the checkerboard in this game. Generally m processing elements are used to represent m players for one cell on the checkerboard. The total number of required processing elements is $m \times n^2$ for the m-player-n-by-n-checkerboard game. One and only one processing element among m processing elements for one cell should have nonzero out-



Cell Coordinates	(1, 1)	(1, 2)	(1, 3)	----	(4, 4)
k=1	1	0	0		1
k=2	0	1	1		0
k=3	0	0	0		0
k=4	0	0	0		0

Fig. 2. System representation for the 'Hip' game.

put to choose a player from m players to occupy the cell on the checkerboard. The nonzero output means that the corresponding player to the processing element should place his counter on the cell. *Figure 2* shows that the cells $(1,1)$ and $(4,4)$ are occupied by the player #1, and the cells $(1,2)$ and $(1,3)$ are occupied by the player #2.

Figure 3 shows the general form of the violation conditions for this game. The four cells corresponding to the four coordinates mark the corners of a square. The kth player processing element of the cell (i, j) should not have nonzero output if the kth player processing elements of the other three cells, $(i+p, j+q)$, $(i-q, j+p)$, and $(i+p-q, j+p+q)$, have nonzero output simultaneously. Therefore the violation conditions are given by the following concise function:

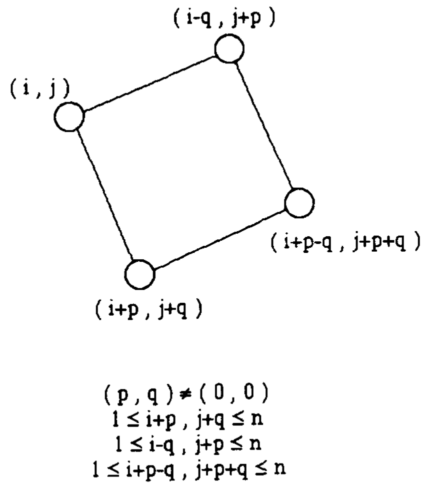


Fig. 3. Violation conditions for the 'Hip' game.

$$\sum_{\substack{p,q \\ (p,q)\neq(0,0) \\ 1\leq i+p,\, j+q\leq n \\ 1\leq i-q,\, j+p\leq n \\ 1\leq i+p-q,\, j+p+q\leq n}} \sum V_{i+p\ j+q\ k} V_{i-q\ j+p\ k} V_{i+p-q\ j+p+q\ k} \tag{4}$$

This function is nonzero if the kth player occupies the three corners of a square.

The motion equation of the kth player processing element of the cell (i, j) for the m-player-n-by-n-checkerboard game is given by:

$$\begin{aligned} \frac{\mathrm{d}U_{ijk}}{\mathrm{d}t} = &-A\left(\sum_{r=1}^{m} V_{ijr} - 1\right) \\ &- B \sum_{\substack{p,q \\ (p,q)\neq(0,0) \\ 1\leq i+p,\, j+q\leq n \\ 1\leq i-q,\, j+p\leq n \\ 1\leq i+p-q,\, j+p+q\leq n}} \sum V_{i+p\ j+q\ k} V_{i-q\ j+p\ k} V_{i+p-q\ j+p+q\ k} \\ &- Cg\left(\sum_{p=1}^{n}\sum_{q=1}^{n} V_{pqk} - \frac{n^2}{m}\right) \\ &+ Dh\left(\sum_{r=1}^{m} V_{ijr}\right). \end{aligned} \tag{5}$$

The first term (A-term) in Eq. (5) forces one and only one processing element to have nonzero output for the cell (i, j) on the n-by-n checkerboard. The second term (B-term) performs the inhibitory force. The B-term discourages the

ijkth processing element to have nonzero output if the kth player occupies the other three cells of a square. The third term (C-term) adjusts the number of the counters which kth player places on the checkerboard so that each player should place the same number of counters on the checkerboard. In this equation each player should place n^2/m counters on the checkerboard. The function $g(x)$ is x if $|x| \leq 3$, 3 if $x > 3$, -3 otherwise. The last term (D-term) provides the hill-climbing which allows the state of the system to escape from the local minimum and to converge to the global minimum – the solutions. The function $h(x)$ is 1 if $x = 0$, 0 otherwise. A, B, C, and D are constant coefficients.

The symmetry strategy which was given by Templeton is useful to obtain a solution. The strategy can be said that all the counters should be placed symmetrically around the center of the checkerboard. Therefore we always make the values of the processing elements symmetric around the center of the checkerboard by using the following procedure:

$$U_{n+1-i\,n+1-j\,k} = U_{ijk} \quad \text{for } i = 1, \ldots, n/2, \quad j = 1, \ldots, n, \text{ and } k = 1, \ldots, m. \tag{6}$$

3. Parallel algorithm for the 'Hip' game

The following procedure describes the proposed algorithm based on the first order Euler method for the m-player-n-by-n-checkerboard game. It decides which player should place his counter on the cell of the checkerboard without making a square by his counters.

0. Set $t = 0$, $A = B = C = 1$, $D = 5$, $U\_max = 20$, and $U\_min = -20$.
1. The initial values of $U_{ijk}(t)$ for $i = 1, \ldots, j = 1, \ldots, n$, and $k = 1, \ldots, m$ are randomized between 0 and U_min.
2. Evaluate values of $V_{ijk}(t)$ for $i = 1, \ldots, n$, $j = 1, \ldots, n$, and $k = 1, \ldots, m$ based on the conditional binary function.

$$V_{ijk}(t) = 1 \text{ if } U_{ijk}(t) > 0 \text{ and } U_{ijk}(t) = \max\{U_{ijr}(t)\} \text{ for } r = 1, \ldots, m$$
$$= 0 \text{ otherwise}. \tag{7}$$

3. Use the motion equation in Eq. (5) to compute $\Delta U_{ijk}(t)$ for $i = 1, \ldots, n$, $j = 1, \ldots, n$, and $k = 1, \ldots, m$. If $(t \bmod 10) < 2$ then

$$\begin{aligned}\Delta U_{ijk}(t) = &-A\Big(\sum_{r=1}^{m} V_{ijr}(t) - 1\Big) \\ &- B \sum_{\substack{p,q \\ (p,q)\neq(0,0) \\ 1\leq i+p,\, j+q\leq n \\ 1\leq i-q,\, j+p\leq n \\ 1\leq i+p-q,\, j+p+q\leq n}} V_{i+p\;j+q\;k}(t) V_{i-q\;j+p\;k}(t) V_{i+p-q\;j+p+q\;k}(t) \times 3 \\ &- Cg\Big(\sum_{p=1}^{n}\sum_{q=1}^{n} V_{pqk}(t) - \frac{n^2}{m}\Big) \times 3 \\ &+ Dh\Big(\sum_{r=1}^{m} V_{ijr}(t)\Big)\end{aligned} \tag{8}$$

else

$$\Delta U_{ijk}(t) = -A\Big(\sum_{r=1}^{m} V_{ijr}(t) - 1\Big) - B \sum_{\substack{p,q \\ (p,q)\neq(0,0) \\ 1\le i+p,\, j+q \le n \\ 1 \le i-q,\, j+p \le n \\ 1 \le i+p-q,\, j+p+q \le n}} V_{i+p\ j+q\ k}(t) V_{i-q\ j+p\ k}(t) V_{i+p-q\ j+p+q\ k}(t) - Cg\Big(\sum_{p=1}^{n}\sum_{q=1}^{n} V_{pqk}(t) - \frac{n^2}{m}\Big) + Dh\Big(\sum_{r=1}^{m} V_{ijr}(t)\Big). \quad (9)$$

4. Compute $U_{ijk}(t+1)$ for $i = 1, \ldots, n$, $j = 1, \ldots, n$, and $k = 1, \ldots, m$ based on the first order Euler method.

$$U_{ijk}(t+1) = U_{ijk}(t) + \Delta U_{ijk}(t). \quad (10)$$

5. If $U_{ijk}(t+1) > U\_max$ then $U_{ijk}(t+1) = U\_max$ for

$$i = 1, \ldots, n,\ j = 1, \ldots, n, \text{ and } k = 1, \ldots, m. \quad (11)$$

If $U_{ijk}(t+1) < U\_min$ then $U_{ijk}(t+1) = U\_min$ for

$$i = 1, \ldots, n,\ j = 1, \ldots, n, \text{ and } k = 1, \ldots, m. \quad (12)$$

6. Use the symmetry strategy in Eq. (6).

$$U_{n+1-i\ n+1-j\ k}(t) = U_{ijk}(t) \text{ for } i = 1, \ldots, n/2,\ j = 1, \ldots, n, \text{ and } k = 1, \ldots, m. \quad (13)$$

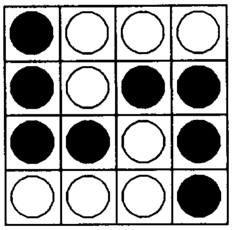
7. If all conflicts are resolved then terminate this procedure else increment t by 1 and goto step 2.

The modified motion equations in step 3 empirically improve the convergence frequency to the global minimum [5].

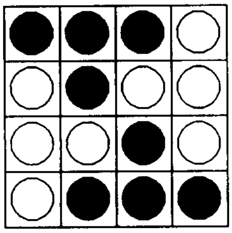
The simulator based on the proposed procedure has been developed on a Macintosh SE/30 in order to verify our algorithm. The following six games were simulated:

(i) Game #1: two-player game on a four-by-four checkerboard.
(ii) Game #2: two-player game on a six-by-six checkerboard.
(iii) Game #3: three-player game on a six-by-six checkerboard.
(iv) Game #4: four-player game on an eight-by-eight checkerboard.
(v) Game #5: five-player game on a ten-by-ten checkerboard.
(vi) Game #6: six-player game on a twelve-by-twelve checkerboard.

Figures 4–9 show two of the solutions or the draw game patterns for the respective games which our simulator found. Our simulator found several solutions in the same games from the different initial values of $U_{ijk}(t)$. *Table 1* shows the frequency of the convergence to solutions and the average numbers of iteration steps where 100 simulation runs were performed for each one of the six games. *Figure 10* shows the relationship between the frequency and the number of iteration steps to converge to the solutions in Game #3 and Game #4.

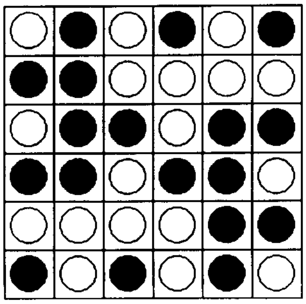


(a) A Solution #1

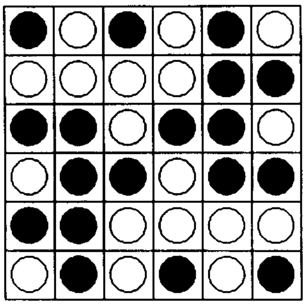


(b) A Solution #2

Fig. 4. Simulation results for game #1.

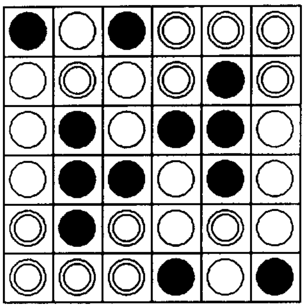


(a) A Solution #1

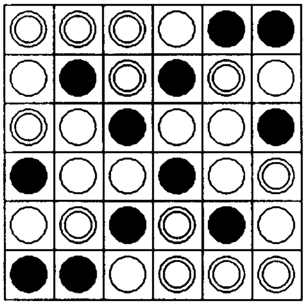


(b) A Solution #2

Fig. 5. Simulation results for game #2.

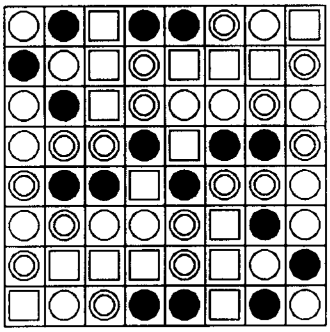


(a) A Solution #1

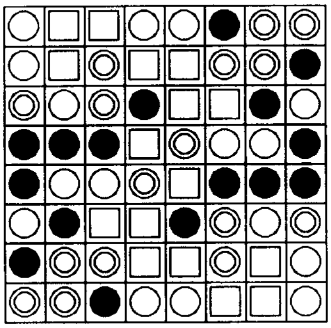


(b) A Solution #2

Fig. 6. Simulation results for game #3.

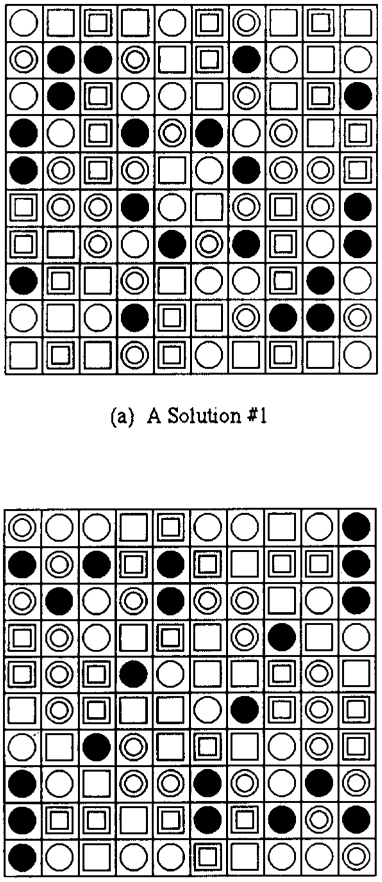


(a) A Solution #1



(b) A Solution #2

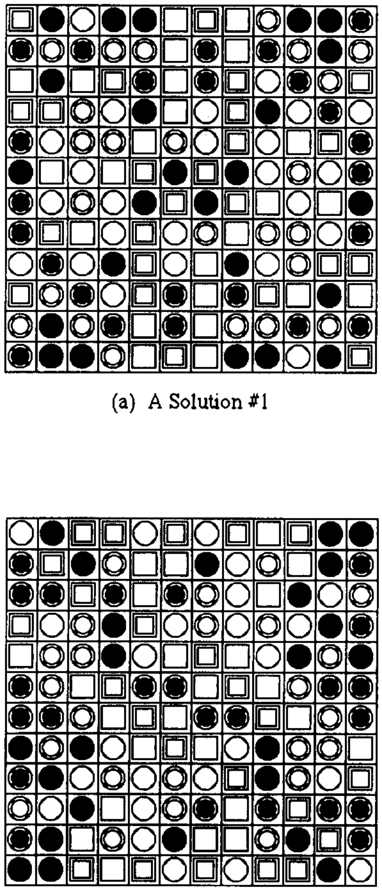
Fig. 7. Simulation results for game #4.



(a) A Solution #1

(b) A Solution #2

Fig. 8. Simulation results for game #5.



(a) A Solution #1

(b) A Solution #2

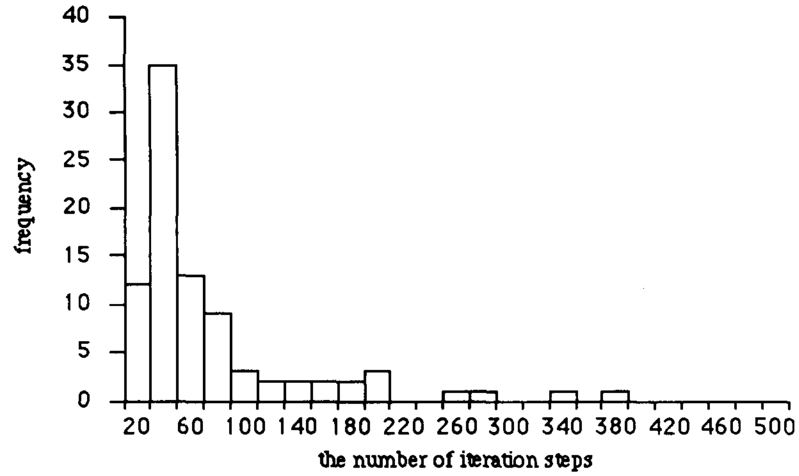
Fig. 9. Simulation results for game #6.

Table 1
Summary of simulation results.

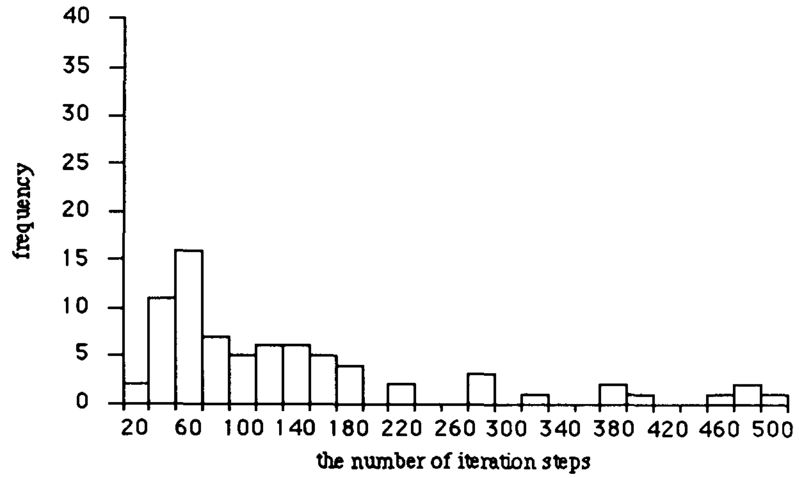
Game no.	Average iteration steps to solutions	Convergence frequency to solutions
Game #1	25.6	85%
Game #2	85.2	27%
Game #3	65.8	87%
Game #4	128.7	75%
Game #5	147.5	30%
Game #6	234.1	10%

4. Conclusion

This paper proposed the parallel algorithm for solving both the normal 'Hip' games and the modified 'Hip' games. It uses $m \times n^2$ processing elements for the m-player-n-by-n checkerboard games. Based on the algorithm the frequency that the state of the system converged to a solution was 10% or more and the average numbers of iteration steps were in a range of 25.6 to



(a) The Game #3



(b) The Game #4

Fig. 10. The relationship between the frequency and the number of iteration steps to converge to the solutions in games #3 and #4.

234.1 in our simulation. With a slight modification the proposed algorithm can be used for finding Ramsey graphs which have been extensively studied by mathematicians [16, 17]. The algorithm for the Ramsey graphs is under investigation.

References

[1] M. Gardner, *New Mathematical Diversions from Scientific American* (University of Chicago Press, 1983).

[2] H. Langman, *Play Mathematics* (Hafner, New York, 1962).

[3] J.J. Hopfield and D.W. Tank, Neural computation of

decisions in optimization problems, *Biol. Cybernet.* 52 (1985) 141–152.

[4] Y. Takefuji and K.C. Lee, A near-optimum parallel planarization algorithm, *Science* 245 (Sept. 1989) 1221–1223.

[5] Y. Takefuji and K.C. Lee, A parallel algorithm for tiling problems, *IEEE Trans. Neural Networks* 1 (1) (1990) 143–145.

[6] Y. Takefuji, C.W. Lin and K.C. Lee, A parallel algorithm for estimating the secondary structure in ribonucleic acids, *Biol. Cybernet.* 63 (5) (1990) 337–340.

[7] 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* 1 (3) (1990) 263-267.

[8] Y. Takefuji and K.C. Lee, A super parallel sorting algorithm based on neural networks, *IEEE Trans. Circuits and Systems* 37 (11) (1990) 1425–1429.

[9] Y. Takefuji and K.C. Lee, Artificial neural networks for four-coloring map problems and K-colorability problems, *IEEE Trans. Circuits and Systems* 38 (3) (1991) 326–333.

[10] N. Funabiki and Y. Takefuji, A parallel algorithm for spare allocation problems, to appear in *IEEE Trans. Reliability*.

[11] N. Funabiki and Y. Takefuji, A parallel algorithm for channel routing problems, to appear in *IEEE Trans. CAD*.

[12] N. Funabiki and Y. Takefuji, A parallel algorithm for traffic control problems in TDM hierarchical switching systems, to appear in *IEEE Trans. Commun.*

[13] N. Funabiki and Y. Takefuji, A parallel algorithm for traffic control problems in three-stage connecting networks, to appear in *J. Parallel Distrib. Comput.*

[14] T. Kurokawa et al., CMOS layout design of the hysteresis McCulloch–Pitts neuron, *Electronics Lett.* 26 (25) (1990) 2093-2095.

[15] W.S. McCulloch and W.H. Pitts, A logical calculus of ideas immanent in nervous activity, *Bull. Math. Biophysics* 5 (1943) 115–133.

[16] R.L. Graham, B.L. Rothschild and J.H. Spencer, *Ramsey theory*, *Scientific American* (July 1990) 112–117.

[17] R.L. Graham, B.L. Rothschild and J.H. Spencer, *Ramsey Theory* (Wiley, New York, 1980 and 1990).

Mr. Funabiki is a graduate student of electric engineering at Case Western Reserve University working toward the PhD degree in electrical engineering. He is also working in Sumitomo Metal Ind., Ltd. (Japan). He received his BS in Mathematical Engineering and Information Physics from the University of Tokyo (Japan) in 1984. His research interests include channel routing problems, traffic control problems in three-stage/multistage connecting networks, time slot assignment problems in TDM hierarchical switching systems, and broadcast scheduling problems. He is a student member of the IEEE Computer Society and the American Association for the Advancement of Science.

Dr. Takefuji is an assistant professor of electrical engineering at Case Western Reserve University. Before joining Case Western in 1988, he taught at the University of South Florida and the University of South Carolina. He received his BS (1978), MS (1980), and PhD (1983) in Electrical Engineering from Keio University (Japan). His current research interests focus on neural network parallel computing for solving real-world problems. He is also interested in VLSI applications and silicon architecture. He received the National Science Foundation/Research Initiation Award in 1989 and is an NSF advisory panelist. A member of the IEEE Computer Society, ACM, International Neural Network Society, and American Association for the Advancement of Science, he received the Information Processing Society of Japan's best paper award in 1980. He coauthored two books, *Digital Circuits* and *Neural Network Computing* (in Japanese). He will publish a new book entitled *Neural Network Parallel Computing* in 1991. He is an editor of the *Journal of Neural Network Computing* and an associate editor of *IEEE Trans. on Neural Networks*. He has published more than 70 technical papers.