NEUROCOMPUTING

ELSEVIER Neurocomputing 8 (1995) 43–49

A neural network algorithm for the no-three-in-line problem [☆]

Kazuhiro Tsuchiya [a,b,*], Yoshiyasu Takefuji [a,c]

[a] *Department of Electrical Engineering and Applied Physics, Case Western Reserve University, Cleveland, OH 44106, USA*

[b] *Fuji Electric Co., Ltd., Tokyo 100, Japan*

[c] *Faculty of Environmental Information, Keio University, Fujisawa 252, Japan*

Received 24 March 1993; accepted 3 January 1994

Abstract

The no-three-in-line problem is one of unsolved problems in number theorem. The goal of the no-three-in-line problem is to locate $2N$ points on an $N \times N$ squares array where no three points are in line. The proposed algorithm uses N^2 hysteresis McCulloch-Pitts neurons as the processing elements for the $N \times N$ array problem. Our neural network algorithm has discovered several different solutions for up to $N = 25$.

Keywords: Neural network algorithm; Unsolved problem; No-three-in-line; Hysteresis McCulloch-Pitts neuron

1. Introduction

The no-three-in-line problem is one of unsolved problems in number theorem and originally presented by H.E. Dudeney in 1906 [1]. The goal of the problem is to locate $2N$ points on an $N \times N$ squares array with no three points in line. Note that the 'line' means not only any vertical/horizontal/diagonal line but also any straight line. Because of the characteristics of the problem, we must locate two and only two points per row and per column. Fig. 1 shows that four black squares (locations of the points) in the 7×7 array problem form six lines where we cannot locate any point on the shaded squares.

[☆] This work was partially supported by Fuji Electric Co., Ltd. and TEPCO Science Foundation.

[*] Corresponding author. Email: tsuchiya@axon.eeap.cwru.edu

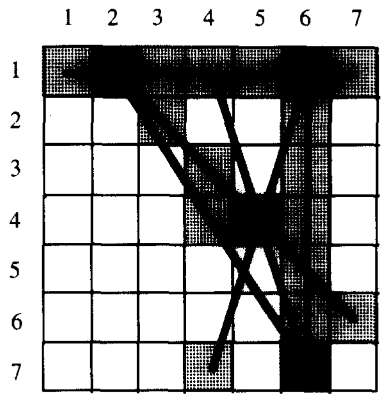


Fig. 1. Definition of the line.

The no-three-in-line problem has been studied by many mathematicians. Guy and Kelly have presented a probabilistic argument supporting the conjecture that for large N, the maximum number of points or $M(N)$ is less than cN, where $c = (2\pi^2/3)^{1/3} \approx 1.87$ [2]. Hall et al. have shown that, given $\varepsilon > 0$, $M(N) \geq (3/2 - \varepsilon)N$ for all sufficiently large N [3]. The maximum number of N satisfying $M(N) = 2N$ is still unknown [4]. Only the examples have proved $M(N) = 2N$ for $N \leq 20$ [5–7]. Many of the solutions, especially for bigger N, were solved by a computer search [8]. In a recent paper [9], the computer search proved $M(N) = 2N$ for $N \leq 32$ where the configuration having $\pi/2$ rotational symmetry or reflection symmetry in two main diagonals was mainly used to reduce the searching space.

Hopfield and Tank proposed the first neural network approach to optimization problems [10]. They applied the sigmoid neural network to the traveling salesman problem. Szu used the McCulloch-Pitts neural network for the same problem [11]. To suppress the oscillatory behavior, the hysteresis neuron model has been introduced [12]. In this paper, the hysteresis McCulloch-Pitts neural network was used for the no-three-in-line problem.

2. Neural representation

The hysteresis McCulloch-Pitts neuron model is shown in Fig. 2 and the input/output function of the i, jth hysteresis neuron is given by:

$$\begin{aligned} V_{i,j} &= 1 \text{ if } U_{i,j} > \text{UTP (Upper Trip Point)} \\ &= 0 \text{ if } U_{i,j} < \text{LTP (Lower Trip Point)} \\ &\quad \text{unchanged otherwise} \end{aligned} \tag{1}$$

where $V_{i,j}$ and $U_{i,j}$ are the output and the input of the i, jth neuron respectively, and UTP is always larger than LTP.

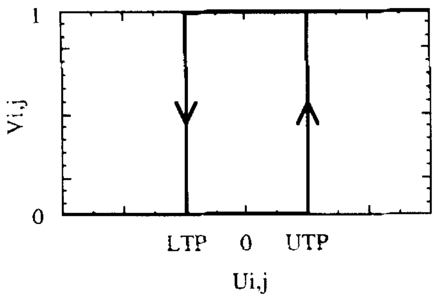


Fig. 2. Hysteresis McCulloch-Pitts input/output function.

The synaptic links between the i, jth neuron and other neurons are determined by the motion equation. The motion equation of the i, jth neuron is given by:

$$\frac{\mathrm{d}U_{i,j}}{\mathrm{d}t}=\frac{\partial E(V_{1,1},V_{1,2},\dots,V_{N,N})}{\partial V_{i,j}} \tag{2}$$

where E is the computational energy function. Convergence theorems/proofs of the artificial neural network is shown in [13]. The motion equation in Eq. (2) can be constructed by considering the necessary and sufficient constraints and/or the cost function from the given problem because the condition of the constraints or the violations gives the interconnections of the artificial neural network.

An $N\times N$ neural array is prepared for the no-three-in-line problem. The output state of the i,jth neuron gives the location of a point in the ith row and the jth column. In other words, $V_{i,j}=1$ means that the point in the ith row and the jth column should be located. $V_{i,j}=0$ means that no point is located in the ith row and the jth column. In order to locate four black squares (points) in Fig. 1, $V_{1,2}=V_{1,6}=V_{4,5}=V_{7,6}=1$ must be satisfied.

It is well understood that two and only two points must be located per row and per column so as to locate $2N$ points on an $N\times N$ squares array with no-three-in-line. The motion equation of the i,jth neuron for the no-three-in-line problem is given by:

$$\begin{aligned}\frac{\mathrm{d}U_{i,j}}{\mathrm{d}t}=&-A\left(\sum_{k=1}^{N}V_{i,k}-2\right)-A\left(\sum_{k=1}^{N}V_{k,j}-2\right)\\&-Bd\left(\sum\sum\sum_{\substack{1\le i-nk,j-mk\le N\\k,m,n\neq 0}}V_{i-nk,j-mk}\right)\\&-Bd\left(\sum\sum\sum_{\substack{1\le i-nk,j+mk\le N\\k,m,n\neq 0}}V_{i-nk,j+mk}\right)\\&\text{for } i,\ j=1,\dots,N\end{aligned} \tag{3}$$

where $\mathrm{d}(x) = x - 1$ if $x \geq 2$, and 0 otherwise. Note that A and B are parameters and that m and n should not be a multiple of the same prime number except 1. In Eq. (3) the first term gives the row (horizontal) constraint such that two and only two points/neurons should be located/fired in the ith row. The second term describes the column (vertical) constraint such that two and only two points/neurons should be located/fired in the jth column. If there is only one neuron fired in the ith row/jth column, the neurons in the ith row/jth column must be encouraged to fire. The third and the fourth terms represent all directional constraints except horizontal and vertical ones such that no three points should be in the same line. When $m = n = 1$, the third and the fourth term represent diagonal constraints.

In order to accelerate the calculation and escape from the local minimum [14], the following equation was used instead of Eq. (3):

If (t mod 10) < 7 then

$$\frac{\mathrm{d}U_{i,j}}{\mathrm{d}t} = -A\left(\sum_{k=1}^{N} V_{i,k} - 2\right) - A\left(\sum_{k=1}^{N} V_{k,j} - 2\right)$$
$$- Bd\left(\sum\sum\sum_{\substack{1 \leq i-nk,\, j-mk \leq N \\ k,m,n \neq 0}} V_{i-nk,j-mk}\right) V_{i,j}$$
$$- Bd\left(\sum\sum\sum_{\substack{1 \leq i-nk,\, j+mk \leq N \\ k,m,n \neq 0}} V_{i-nk,j+mk}\right) V_{i,j}$$

else

$$\frac{\mathrm{d}U_{i,j}}{\mathrm{d}t} = -A\left(\sum_{k=1}^{N} V_{i,k} - 2\right) - A\left(\sum_{k=1}^{N} V_{k,j} - 2\right)$$
$$- Bd\left(\sum\sum\sum_{\substack{1 \leq i-nk,\, j-mk \leq N \\ k,m,n \neq 0}} V_{i-nk,j-mk}\right)$$
$$- Bd\left(\sum\sum\sum_{\substack{1 \leq i-nk,\, j+mk \leq N \\ k,m,n \neq 0}} V_{i-nk,j+mk}\right)$$

for $i, j = 1, \ldots, N$ (4)

where t is the number of iteration steps. The third and the fourth terms in the first equation are activated only when the i, jth neuron generates nonzero output. In the local minimum, no neurons/points are fired/located on the ith row or the jth column, or only the limited neurons/points are fired/located. In order to increase the global minimum convergence, the following two terms were also added to Eq. (4) [13]:

$$+Ch\left(\sum_{k=1}^{N} V_{i,k}\right) + Ch\left(\sum_{k=1}^{N} V_{k,j}\right) \tag{5}$$

where $h(x)$ is 1 if $x < 2$, and 0 otherwise. Note that C is a parameter.

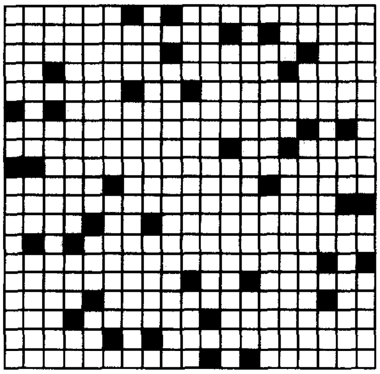
3. Neural network algorithm

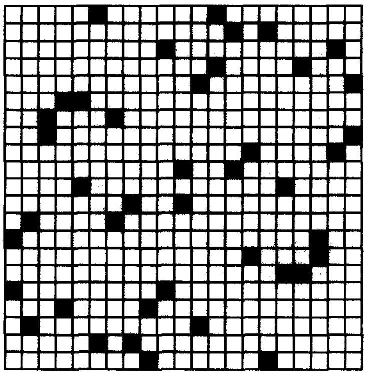
Each neuron's output is updated by the motion equation. In order to numerically solve the motion equation, the first order Euler method was used. The following procedure describes the proposed algorithm.

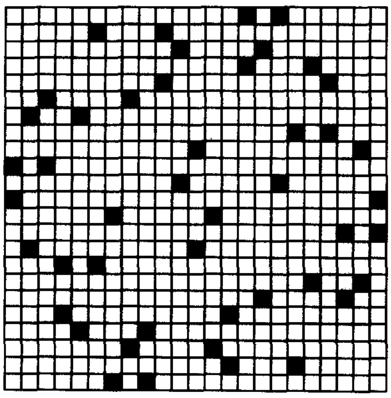
0. Set $t=0$, $A=B=C=1$, UTP $=1$, LTP $=-1$, and the range for $U_{i,j}$, $U\_\text{max} = 5$ and $U\_\text{min} = -10$.
1. Assign randomly and uniformly generated numbers between 0 and U_min to the initial values of $U_{i,j}(t)$ for $i, j = 1, \ldots, N$.
2. Set $V_{i,j}(t) = 0$ for $i, j = 1, \ldots, N$.
3. Iterate the following four steps sequentially for $i, j = 1, \ldots, N$:

3.1. Compute Eq. (4) and (5) to obtain $\Delta U_{i,j}(t)$:

$$\Delta U_{i,j}(t) = \frac{\mathrm{d}U_{i,j}(t)}{\mathrm{d}t} \tag{6}$$


A solution for N=19


A solution for N=21


A solution for N=23

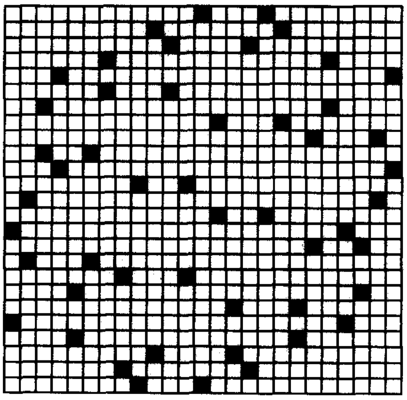

A solution for N=25

Fig. 3. Different solutions for the no-three-in-line problem.

3.2. Update $U_{i,j}(t+1)$ based on the first-order Euler method:

$$U_{i,j}(t+1) = U_{i,j}(t) + \Delta U_{i,j}(t) \tag{7}$$

3.3. If $U_{i,j}(t+1) > U\_\text{max}$ then $U_{i,j}(t+1) = U\_\text{max}$ and if $U_{i,j}(t+1) < U\_\text{min}$ then $V_{i,j}(t+1) = U\_\text{min}$.

3.4. Update $V_{i,j}(t+1)$ by Eq. (1).

4. If $V_{i,j}(t) = 1$ and $\Delta U_{i,j}(t) = 0$ for $i, j = 1, \ldots, N$ or $t = t\_\text{limit}$, then terminate this procedure.

5. Increment t by 1 and go to step 3.

The termination condition in step 4 implies that the required constraints are all satisfied.

4. Results

The proposed algorithm was implemented on a Macintosh PowerBook 170 and a DEC 3100 computer. Generally the neural network program running on a sequential machine requires $O(N^2)$ time for solving the no-three-in-line problem and nearly $O(1)$ time on a parallel machine using N^2 processing elements.

Our simulator has found several solutions for up to $N = 25$ so far. Fig. 3 shows solutions for $N = 19$, 21, 23, and 25. Note that one and only one solution has been shown for $N = 23$ and only two solutions for $N = 25$ have been found, and that all of those configurations have reflection symmetry in two main diagonals [9]. It took about 10 minutes to obtain the solution on $N = 25$ in the DEC machine.

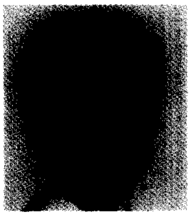
5. Conclusion

We have shown several solutions of the no-three-in-line problem, an unsolved problem in number theorem. Our neural network algorithm uses an $N \times N$ neural array for the $N \times N$ array problem. The simulation results of our algorithm show that our approach is promising for finding the solutions for the larger scale problems.

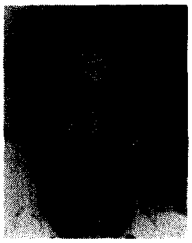
References

[1] H.E. Dudeney, *Amusements in Mathematics* (Dover, 1958).
[2] R.K. Guy and P.A. Kelly, *Canadian Math. Bull.* 11 (1968) p. 527.
[3] R.R. Hall, T.H. Jackson, A. Sudbery and K. Wild, *J. Combinatorial Theory* 18 (1975) 336.
[4] R.K. Guy, *Unsolved Problems in Number Theory* (Springer-Verlag, 1981) 133.
[5] M. Gardner in *Scientific American* 236 (1977) 139.
[6] D.B. Anderson, *J. Combinatorial Theory* 27 (1979) 365.
[7] H. Harborth, P. Oertel and T. Prellberg, *Discrete Math.* 73 (1988/89) 89.
[8] M.A. Adena, D.A. Holton and P.A. Kelly in *Combinatorial Mathematics* (D.A. Holton, ed), 403 (1974) 6.
[9] A. Flammenkamp, *J. Combinatorial Theory* 60 (1992) 305.

[10] J. Hopfield and D. Tank, *Biol. Cybern.*, 52 (1985) 141.
[11] H. Szu in *Proc. Int. Conf. on Neural Networks* (IEEE, Piscataway, NJ, 1988) 259.
[12] Y. Takefuji and K.C. Lee, *Biol. Cybern.* 64 (1991) 353.
[13] Y. Takefuji, *Neural Network Parallel Computing* (Kluwer, MA, 1992).
[14] Y. Takefuji and K.C. Lee, *IEEE Trans. Neural Networks* 1 (1) (1990) 143.



Kazuhiro Tsuchiya has been a research associate with Dr. Y. Takefuji at the Department of Electrical Engineering and Applied Physics at Case Western Reserve University since 1991. He received his B.S. (1983) in chemical engineering and M.S. (1985) in material chemistry from Tohoku University (Japan). He has worked for Fuji Electric Co., Ltd. in Japan since 1985. His research interests focus on neural network parallel computing for solving real-world problems such as graph theory problems, operations research problems, management science problems, and VLSI design. He is also interested in VLSI applications and silicon architecture.



Yoshiyasu Takefuji is a tenured associate professor on faculty of environmental information at Keio University since April 1992 and also on faculty of Electrical Engineering at Case Western Reserve University since 1988. Before joining Case, he taught at the University of South Florida for two years and the University of South Carolina for three years. He received his BS (1978), MS (1980), and Ph.D. (1983) from Electrical Engineering from Keio University under the supervision of Professor Hideo Aiso. His research interests focus on neural network parallel computing for solving real-world problems. He is also interested in VLSI applications and silicon architecture.