Paper

A Saturation Computation Method of Artificial Binary Neural Networks for Combinatorial Optimization Problems

Rong Long Wang* Student Member
Zheng Tang* Non-member
Qi Ping Cao** Non-member

In this paper, we propose a saturation computation method of neural networks for efficiently solving combinatorial optimization problems. In this computation method, once the neuron is in excitatory state, then its input potential is considered to be in positive saturation where the input potential can only be reduced but cannot be increased, and once the neuron is in inhibitory state, then its input potential is considered to be in negative saturation where the input potential can only be increased but cannot be reduced. The proposed method is applied to N-Queens problem. The performance is evaluated through simulations where the results show that the saturation method improves the searching capability of neural networks and shortens the computation time. Particularly, the simulation results show that the performance of the proposed method surpasses the exiting methods for N-queens problem in synchronous parallel model.

Keywords: combinatorial optimization problems, Hopfield neural network, N-Queens problem, synchronous parallel model, saturation computation method

1. Introduction

A number of commonly encountered problems in mathematics, computer science, molecular biology, management science, seismology, communications, and operation research belong to a class of combinatorial optimization problems [1]. The goal of combinatorial optimization problems is to minimize or maximize a cost function subject to constraints. Most of practical combinatorial optimization problems are of the NP-complete. Because it is unlikely that there exist efficient algorithms for NP-completeness problems, many polynomial time approximation algorithms have been proposed to solve the NP-completeness problems. The inherent parallelism in neural network provides a promising alternative for solving these problems. Since McCulloch and Pitts proposed a simplified artificial neuron model in 1943 [2], several neuron models have been investigated. Hopfield and Tank [3] [4] were the first to propose a neural network called the Hopfield network for solving combinatorial optimization problems. The Hopfield network has a large number of neurons or processing elements where neuron $\#i$ has an input U_i and an output V_i. It uses the gradient descent method to seek the local minimum of the Liapunov energy function E by using motion equations of these neurons. The energy function is given by the constraints and the objective function in the problem. Hopfield proposed a motion equation with a partial derivation term of the energy function and a decay term with a time constant τ[3] [4]. Since Hopfield and Tank's work [3], [4], there has been growing interest in the Hopfield network because of its advantages over other approaches for solving optimization problems. The advantages include massive parallelism, convenient hardware implementation of the neural network architecture, and a common approach for solving various optimization problems.

However the work by Wilson and Pawley [5] showed that the Hopfield network often failed to converge to valid solutions. Even when it converged, the obtained solution was often far from the optimal solution. Since then, various modifications have been proposed to improve the convergence of the Hopfield network. Takefuji et al. showed that the decay term in Hopfield-type neural network increased the energy function under some conditions [6]. They modified the motion equation in order to guarantee the local minimum convergence. However, the Hopfield-type neural network was forced to converge to the local minimum. Therefore, several neuron models and heuristics such as hysteresis binary neuron model [7], neuron filter [8], the hill-climbing term and omega function [9], Lagrange relaxation [10] and pots spin [11] have been proposed to improve the convergence of the networks. Despite the improvement of the performance of the Hopfield network over the past decade, this model still has some basic problems [12], [13]. To improve the solution quality is still a question of both practical and theoretical interests.

In Neural netwoek, there are two kinds of methods to compute the input potential of neurons: the time-independent one in which the input potential of neurons at time t+1 does not directly depend on their value

*Faculty of Engineering, Toyama University, Toyama-shi, Japan 930-8555
**Tateyama Systems Institute, Toyama-shi, Japan 930-0001

at time t[14][15] and the time-dependent one in which the input potential of neurons at time $t+1$ depends on their value at time t[6]. In this paper, we propose a new method to compute the input potential of neurons named saturation computation method. This method offers a new idea to compute the input potential of neurons to improve the global convergence quality and shorten the convergence time. In the proposed computation method, once the neuron is in excitatory state, its input potential is considered to be in positive saturation and the input potential can only be reduced but cannot be increased, and once the neuron is in inhibitory state, its input potential is considered to be in negative saturation and the input potential can only be increased but cannot be reduced. The effectiveness of the saturation computation method is demonstrated by simulating N-Queens problem. The simulation results show that the proposed saturation computation method improves the searching capability of neural networks and shortens the computation time.

2. Hopfield Neural Network for Combinatorial Optimization

The Hopfield neural network model for combinatorial optimization problems consists of two elements named "neuron unit" and "motion equation". The neuron unit is a collection of simple processing elements called neurons. Each neuron has an input potential U_i and an output potential V_i. The dynamic behavior of the network is described by the following motion equation with a partial derivation term of the energy function and a decay term with a time constant τ[3][4].

$$\frac{dU_i(t)}{dt} = -\frac{\partial E(V_1, V_2, ..., V_n)}{\partial V_i} - \frac{U_i}{\tau} \cdots\cdots\cdots (1)$$

Takefuji et al. showed that the decay term increases the energy function under some conditions [6]. They modified the motion equation in order to guarantee the local minimum convergence.

$$\frac{dU_i(t)}{dt} = -\frac{\partial E(V_1, V_2, ..., V_n)}{\partial V_i} \cdots\cdots\cdots\cdots\cdots (2)$$

There are two kinds of methods to compute the input potential of neurons:

(1) The time-independent one in which the input potential of neurons at time $t+1$ does not directly depend on the value at time t[14][15]. Thus the input potential is simply the partial derivation term of the energy function.

$$U_i = -\frac{\partial E(V_1, V_2, ..., V_n)}{\partial V_i} \cdots\cdots\cdots\cdots\cdots\cdots (3)$$

(2)The time-dependent one in which the input potential of neurons at time $t+1$ depends on the value at time t[6].

$$U_i(t+1) = U_i(t) + \frac{dU_i(t)}{dt} \cdots\cdots\cdots\cdots\cdots\cdots (4)$$

The output is updated from U_i using a non-linear function called neuron model. The follow two binary neuron models have been used for optimization problems:

(1) The McCulloch-Pitts binary neuron model [2]

$$V_i = \begin{cases} 1 & if\ U_i > 0 \\ 0 & otherwise \end{cases} \cdots\cdots\cdots\cdots\cdots\cdots\cdots (5)$$

(2) The hysteresis McCulloch-Pitts neuron model [7]

$$V_i = \begin{cases} 1 & if\ U_i > UTP \\ 0 & if\ U_i < LTP \\ \text{unchanged} & if\ LTP < U_i < UTP \end{cases} \qquad (6)$$

Where, UTP and LTP are constant parameters satisfying $UTP > LTP$.

Each neuron updates its input potential according to the computation rule (Eq.(3) or Eq.(4)) and sends its output state in response to the input according to the input/output function (Eq.(5) or Eq.(6)). All neurons operate in parallel and each adjusts its own state to the states of all the others; in consequence, the whole network converges to a final configuration. The structure of combinatorial optimization problems can be mapped onto the structure of a Hopfield network by deciding the connection weights between the neurons. In this way, we can find the solution to a problem simply by observing the final configuration that the network reaches.

3. Saturation Computation Method of Neural Networks

In this paper, a saturation computation method is proposed to improve the global convergence quality and shorten the convergence time. The proposed saturation computation method consists of two main rule:

(1) Once a neuron is in excitatory state, then its input potential is considered to be in positive saturation. In the positive saturation, the input potential U_i can only be reduced but cannot be increased.

For the case of $V_i = 1$:
if $dU_i(t)/dt < 0$

$$U_i(t+1) = U_i(t) + \frac{dU_i(t)}{dt} \cdot \Delta t \cdots\cdots\cdots\cdots (7)$$

else

$$U_i(t+1) = U_i(t)$$

(2) Once the neuron is in inhibitory state, then its input potential is considered to be in negative saturation. In the negative saturation, the input potential can only be increased but cannot be reduced.

For the case of $V_i = 0$
if $dU_i(t)/dt > 0$

$$U_i(t+1) = U_i(t) + \frac{dU_i(t)}{dt} \cdot \Delta t \cdots\cdots\cdots\cdots (8)$$

else

$$U_i(t+1) = U_i(t)$$

The output of each neuron at iteration $(t+1)$ is updated according to the updating rule specified by Eq.(6).

We analyze the details of the time-independent, the time-dependent and the proposed saturation computation method and explain their differences in performance.

In the time-independent computation method, the input potential of neuron updates according to computation rule (Eq.(3)). Because the input potential of neuron at time $t+1$ does not directly depend on its value at time t, the network usually causes undesirable oscillation [16].

The time-dependent computation method is the most popular method. This computation method has been successfully used for several optimization problems [17]-[20]. However this computation method is not as good as expected. In the time-dependent computation method, the input potential $U_i(t)$ is updated according to Eq.(4) unconditionally. We consider the case of $U_i(t) > UTP$, $V_i(t) = 1$ and $(-\partial E/\partial V_i) > 0$ in the time-dependent computation method. Because $Vi(t) = 1$ and $(-\partial E/\partial V_i) > 0$, the neuron dose not change its output, but the input $U_i(t+1)$ will be increased, and as time goes on, the input $U_i(t+1)$ may become so large that the neuron is insensitive to its input. Similarly in the case of $U_i(t) < LTP$, $V_i(t) = 0$ and $(-\partial E/\partial V_i) < 0$ in the time-dependent computation, the input potential U_i may become so small that the neuron is not sensitive to its input. Once it happens, the convergence will become very slow and the states of network may be very difficult to be changed. The solution obtained under this condition is less likely to have high quality.

In contrast to the time-independent and the time-dependent computation method, the proposed saturation computation method uses the updating conditions (Eq.(7) and Eq.(8)) to update the input potential of each neuron. According to the updating conditions (Eq.(7) and Eq.(8)), when the output state of neuron is 0, its input potential can only be increased but cannot be decreased and once the input potential exceed the UTP, the output state of the neuron becomes 1 and the input potential will be in positive saturation. Thereafter the input potential will not be increased until the input potential falls into a negative saturation. Thus, different neuron may have different range for the input potential, and the same neuron may have different range for the input potential when the network is in different configuration. This kind of variation behavior of input potential in the proposed saturation computation method can restrain the undesirable oscillation of network and the insensitivity to the input of neurons, and may result in better performance for the network.

4. Algorithm

The following procedure describes the synchronous parallel algorithm for solving combinatorial optimization problems based on the proposed saturation computation method. Note that *targ_cost* is the target total cost set by user as an expected total cost. *t_limit* is the maximum number of iteration step allowed by user.

(1) Set $t = 0$, Δt and set *targ_cost*, *t_limit*, and other constants.

(2) Initialize value of U_i for $i = 1, \cdots, N$ in the range from LTP to UTP randomly.

(3) For $i = 1, \cdots, N$, Evaluate $V_i(t)$ using binary neuron model.

(4) Check the network. If *targ_cost* is reached, then terminate this procedure.

(5) Increment t by 1. If $t > t\_limit$, then terminate this procedure.

(6) For i=1,⋯,N

(a) Compute Eq.(2) to obtain $\Delta U_i(t)$.

$$\Delta U_i(t) = \frac{dU_i(t)}{dt} \cdots\cdots\cdots\cdots \quad (9)$$

(b) Update $U_i(t+1)$ using updating conditions (Eq.7 and Eq.8).

If $(V_i = 1$ and $dU_i(t)/dt < 0)$ or $(V_i = 0$ and $dU_i(t)/dt > 0)$

$$U_i(t+1) = U_i(t) + \Delta U_i(t) \cdot \Delta t \cdots\cdots \quad (10)$$

else

$$U_i(t+1) = U_i(t) \cdots\cdots\cdots\cdots \quad (11)$$

(7) If $U_i(t+1) = U_i(t)$ for $i = 1, \cdots, N$, then terminate this procedure

(8) Go to the step 3.

5. Application to N-Queens Problem

In this section, we evaluate the performance of the proposed saturation computation method through simulations, and compare it with two traditional computation methods and other improved methods. The evaluation in our experiments is based on N-Queens problem.

5.1 The N-Queens Problem N-Queens problem is classic of difficult optimization. The task is given a standard chessboard and N chess queens, to place them on the board so that no queen is on the line of attack of any other queen. The problem can be solved by constructing an appropriate energy function and minimizing the energy function to zero ($E = 0$) using an $N \times N$ two-dimensional binary Hopfield-type neuron network[21].

The objective energy function of N-Queens problem is given by:

$$E = \frac{A}{2}\sum_{i=1}^{N}(\sum_{k=1}^{N} y_{ik} - 1)^2 + \frac{A}{2}\sum_{j=1}^{N}(\sum_{k=1}^{N} y_{kj} - 1)^2 + \frac{B}{2}(\sum_{i=1}^{N}\sum_{j=1}^{N} y_{ij}(\sum_{1 \le i-k, j-k \le N, k \ne 0} y_{i-k,j-k} + \sum_{1 \le i-k, j+k \le N, k \ne 0} y_{i-k,j+k})) \cdots\cdots \quad (12)$$

Where A, B are coefficients, the output $y_{ij} = 1$ represents that a queen is placed at i-th row j-th column on the chessboard, and output $y_{ij} = 0$ represents no placement there. The first term becomes zero if one queen is placed in every row. The second term becomes zero if one queen is placed in every column and the third term becomes zero if no more than one queen is placed on any diagonal line.

The motion equation for the ij-th neuron is given by:

$$\begin{aligned} E = & -A\left(\sum_{k=1}^{N} y_{ik} - 1\right) - A\left(\sum_{k=1}^{N} y_{kj} - 1\right) \\ & -B\left(\sum_{1 \leq i-k, j-k \leq N, k \neq 0} y_{i-k,j-k} \right. \\ & \left. + \sum_{1 \leq i-k, j+k \leq N, k \neq 0} y_{i-k,j+k} \right) \cdots\cdots\cdots \quad (13) \end{aligned}$$

5.2 **Simulations and Discussions** The proposed saturation computation method was implemented and simulations were performed in C++ on PC Station (PentiumIII 800MHz). In simulations, the parameters A and B were set to 2 and 1. The maximum updating step was set to 1000. The hysteresis binary neuron model was used to update the output of neurons.

According to the input potential updating conditions (Eq.(7) and Eq.(8)) of the proposed saturation computation method, we can see that the band size (LTP and UTP) of the hysteresis binary neuron is an important parameter in the proposed saturation computation method. When LTP and UTP are near zero, undesirable oscillation similar to that in the time-independent computation method will appear. When LTP and UTP is too large (negative large value for LTP), the network will become insensitive to the input of neurons, which is similar to the behavior of the time-dependent computation method. To see this characteristic, we simulated a 20-Queens problem using different band size of the hysteresis binary neuron. Figure 1 shows the relation between the percentage of valid solution and the band size (LTP and UTP) of the hysteresis binary neuron. Note that 100 simulation runs with different initial states of neurons were performed to obtain the percentage of valid solution per pair of LTP and UTP. From Fig.1 we can see that when LTP and UTP are near zero or very large ($LTP < -20$ and $UTP > 20$), the rate to find a valid solution is very low. Especially we can see that when LTP and UTP are zero, the network did not find a valid solution. The reason is that when LTP and UTP is zero, the proposed method is nearly as same as the time-independent computation method, and undesirable oscillation appears. We can also see from this figure that when the LTP is in the range from -2 to -20 and the UTP is in the range from 2 to 20, the rate to find a valid solution is very high. Figure 2 shows the relation between the average number of iteration steps to find valid solution and the band size of the hysteresis binary neuron. From Fig.2 we can see that the larger is the band size of the hysteresis binary neuron, the large is the number of iteration steps to find valid solution. The reason of this relation is that for the larger band size of the hysteresis binary neuron, the network becomes insensitive to the input of neurons, which is similar to the

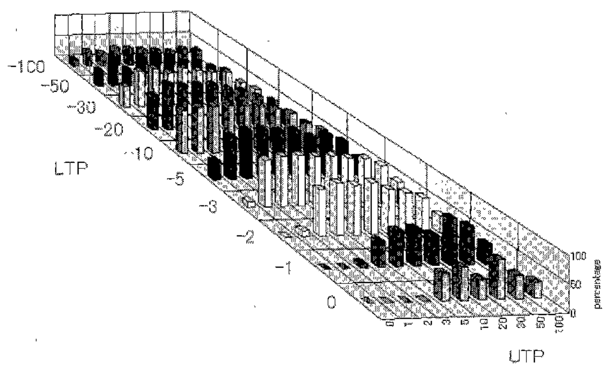


Fig. 1. The relation between the percentage of valid solution and the band size (LTP and UTP) of hysteresis binary neuron.

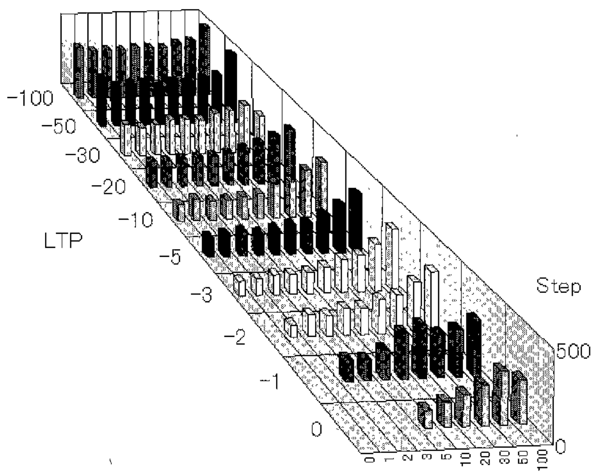


Fig. 2. The relation between the average number of iteration steps to find valid solution and the band size of hysteresis binary neuron.

behavior of the time-dependent computation method. From these simulation results, we can summarized that the proposed saturation computation method could restrain the undesirable oscillation of the network and the insensitivity of the network to the input of neurons by selecting a reasonable pair of band size (LTP and UTP) of the hysteresis binary neuron. Furthermore, our simulations found that the range of the reasonable pair of the band size of the hysteresis binary neuron is very large; it is easy to select the reasonable pair of the band size of the hysteresis binary neuron.

In order to widely verify the proposed method, we tested the proposed algorithm with a large number of instances. The band size UTP and LTP of the hysteresis binary neuron were set to 3 and -3 respectively. The time-independent and the time-dependent computation methods were also executed for comparison. Three computation methods are all executed in synchronous parallel model. In simulating 20-queens problem, we recorded the variation process of the input potential of neurons in three computation methods. Figure 3 shows a variation of the input potential of one neuron in the time-independent method. From Fig.3 we can see that in the

Table 1. Computational results.

Queens	Propose Method		Time-independent		Time-dependent		Takefuji's Network[22]		Maximum Neural[22]	
	Conv.	Steps	Conv.	Steps	Conv.	Steps	Conv.	Steps	Conv.	Steps
20	100	148	0	-	29	310	57	169	84	129
30	98	168	0	-	47	338	54	381	96	172
50	100	191	0	-	53	395	62	107	97	154
100	100	242	0	-	53	676	44	82	98	199
150	100	298	0	-	46	757	13	103	99	245
200	100	326	0	-	13	771	0	-	93	271
300	100	415	0	-	8	918	0	-	94	335

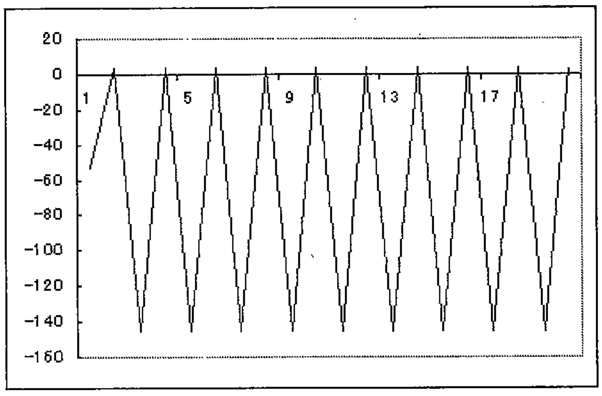


Fig. 3. An example of the variation process of input potential of neuron under the time-independent computation method.

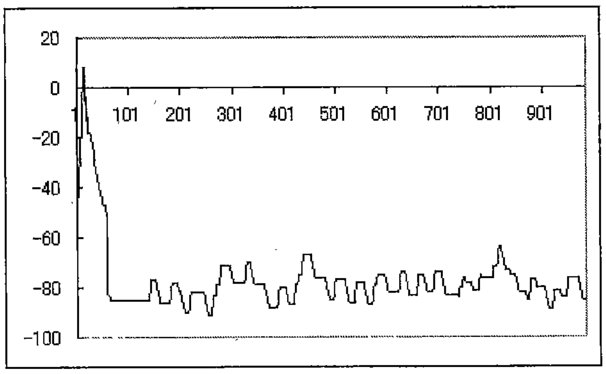


Fig. 4. An example of the variation process of input potential of neuron under the time-dependent computation method.

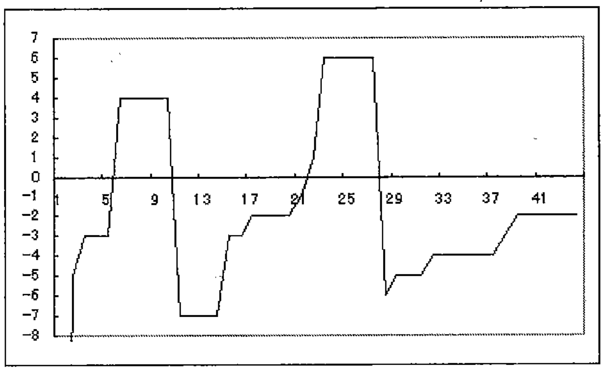


Fig. 5. An example of the variation process of input potential of neuron under the proposed saturation computation method.

time-independent computation method, after the second updating, the input potential caused oscillation. Figure 4 shows the variation of the input potential of a neuron in time-dependent computation method. From this figure we can see that after the 100th updating, the input potential U_i became very small and the neuron was not sensitive to its input. The variation of the input potential under the proposed saturation computation method is shown in Fig.5. We can see from this figure that in the proposed saturation computation method, the variation of the input potential was very calm due to the saturation of the input potential of neuron. Thus, the undesirable oscillation of the network and the insensitivity of the network to the input of neurons were restrained. The detail simulation results were shown in Table 1, where the convergence rates and the average numbers of iteration steps required for the convergence were summarized. The simulation results showed that the proposed saturation computation method could increase the percentage of valid solutions and reduce the average numbers of time steps as compared with the other two computation methods. We also compared our results with these produced by other improved neural network methods. Takenaka et al. [22] simulated N-Queens problem using Takefuji's network and maximum neural network in synchronous parallel model. Table 1 also showed the simulation results reported by Takenaka et al. [22]. From Table.1 we could see that the proposed saturation computation method performed better than Takefuji's neural network and maximum neural network in terms of the solution quality for N-queens problem.

Focusing on combinatorial optimization problems, some other optimization techniques have been proposed. Simulated Annealing (SA) [23] is a widely used meta-heuristic. It could be described as a randomized scheme, which reduces the risk of getting trapped in local minima by allowing moves to inferior solution. Simulated annealing is a powerful method for solving combinatorial optimization problems, but it always requires more iterations than exhaustive search to find a good solution [24]. To further evaluate the performance of the proposed method, simulated annealing was also executed for comparison. We used the annealing algorithm given by Johnson et al [25]. It started with a randomly generated assignment; repeatedly picked a random variable, and computed the change (Δ) of energy when the state of that variable was flipped. If $\Delta \geq 0$, it made the

Table 2. Comparison with the simulated annealing.

Queens	Propose Method		Simulated Annealing	
	Conv.	CPU(S)	Conv.	CPU(S)
20	100	0.05	98	18.57
30	98	0.13	90	88.01
50	100	2.63	78	1448.34
80	100	11.89	25	8245.63

flip. Otherwise, it flipped the variable with probability $e^{-\Delta/T}$. We slowly decreased the temperature (T) from 3.0 to 0.00001. Because of the unreasonable computation time for $N > 80$ queens, we executed the simulated annealing only for 20, 30, 50 and 80 Queens. 100 simulation runs were performed for 20, 30 and 50 queens, 50 simulation runs were performed for 80 queens. To compare the proposed method with the simulated annealing, the rate to find the valid solution and the average computation time were shown in Table 2. From this table, we can see that the proposed method worked better than simulated annealing in terms of the solution quality and the computation time.

6. Conclusion

A saturation computation method in binary Hopfield-type neural networks for efficiently solving combinatorial optimization problems was proposed in this paper. The differences in the performance among the traditional two computation methods and the proposed saturation computation method were analyzed. The effectiveness of the proposed saturation computation method was demonstrated by simulating N-Queens problem. The simulation results showed that the proposed saturation computation method could improve the global convergence quality and shorten the convergence time. The simulation results also showed that the proposed saturation computation method was better than the other improved methods.

Acknowledgment

This work was supported in part by the grant-in-aid for Science Research of the Ministry of Education, Science and Culture of Japan under Grant (C)(2)12680392.

(Manuscript received November 14, 2001, revised July 22, 2002)

References

(1) R. Garey and S. Johnson: Computers and Intractability, a guid to theory of NP-completeness, Freeman and Company, (1991)

(2) W. S. McCulloch and W. H. Pitts: "A logical calculus of ideas immanent in nervous activity", *Bull. Math. Biophys*, Vol.5 pp.115-133 (1943)

(3) J. J. Hopfield and D.W.Tank: " 'Neural' computation of decisions in optimization problems", *Biol. Cybern.*, No.52, pp.141-152 (1985)

(4) J.J. Hopfield and D.W.Tank: " Computing with neural circuits: a model". *Science*, No.233, pp.625-633 (1986)

(5) G. V. Wilson and G. S. Pawley: "On the stability of the traveling salesman problem algorithm of Hopfield and Tank". *Boil. Cybern.*, Vol.58, pp.63-70 (1988)

(6) Y. Takefuji and K. C. Lee: "Artificial neural networks for four-coloring map problems and K-colorability problems", *IEEE Trans. Circuits Syst.*, Vol.38, No. 3, pp.326-333 (1991-3)

(7) Y. Takefuji and K. C. Lee: "An artificial hysteresis binary neuron: A model suppressing the oscillatory behaviors of neural dynamics", *Biol. Cybern.*, Vol.64, pp.353-356 (1991)

(8) Y. Takenaka, N. Funabiki, and T. Higashino: "A proposal of neuron filter: A constraint resolution scheme of neural networks for combinatorial optimization problems", *IEICE Trans. Fundamentals*, Vol.E83-A, No.9, pp.1815-1823 (2000)

(9) Y. Takefuji and K. C. Lee: "A parallel algorithm for tiling problems", *IEEE Trans. Neural networks*, Vol.1, No.1, pp.143-145 (1990-3)

(10) Z. L. Stan: "Improving convergence and solution quality of Hopfield-type neural networks with augmented Lagrange multipliers", *IEEE Trans. Neural Networks*, Vol.7, No.6, pp.1507-1516 (1996)

(11) S. Ishii and S. Masaaki: "Chaotic potts spin model for combinatorial optimization problems", *Neural Networks*, Vol.10, pp.941-963 (1997)

(12) B. S. Cooper: Higher order neural networks-can they help us optimize? In Proceedings of the Sixth Australian Conference on neural networks (ACNN'95) pp.29-32 (1995)

(13) D. E. Bout and T. K. Miller: "Improving the performance of the Hopfield-Tank neural network through normalization and annealing". *Biol. Cybern.*, Vol.62, pp.129-139 (1989)

(14) J. Mandziuk and B. Macukow: "A neural network designed to solve the N-Queens problem", *Boil. Cybern*, Vol 66, pp.375-379 (1992)

(15) J. Mandziuk: "Solving the N-Queens problem with a binary Hopfield-type network", *Boil.Cybern*, Vol 72, pp.439-445 (1995)

(16) H. Yoshio, T. Baba, N. Funabiki, and S. Nishikawa: "A proposal of the N-Parallel computation method of the neural network for N-Queens problems", *IEICE Trans. Fundamentals*, Vol.J80-A, No.1, (1997)

(17) Y. Takefuji, L. L. Chen, K. C. Lee, and J. Huffman: "Parallel algorithm for finding a near-maximum independent set of a circle graph", *IEEE Trans. Neural Networks*, Vol.1, pp.263-267 (1990)

(18) N. Funabiki and Y. Takefuji: "A parallel algorithm for spare allocation problems", *IEEE Trans. Reliability*, Vol.40, No.3, pp.338-346 (1991)

(19) K.C. Lee, N. Funabiki, and Y. Takefuji: "A parallel improvement algorithm for the bipartite subgraph problem", *IEEE Trans. Neural Networks*, Vol.3, No.1 pp.139-145 (1992)

(20) Y.Takefuji and K.C.Lee: "A near-optimum parallel planarization algorithm", *Science*, Vol.245, No.4922, pp.1221-1223 (1989)

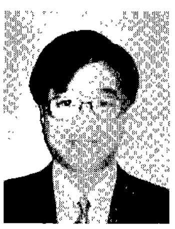
(21) Y. Takefuji: "Neural network parallel computing", Kluwer Academic Publishers, (1992)

(22) Y. Takenaka, N. Funabiki, and S. Nishikawa: "Maximum neural network algorithms for N-Queen problems", *J. IPSJ* Vol.37, No.10, pp.1781-1788 (1996)

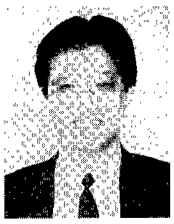
(23) S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi: "Optimisation by Simulated Annealing", *Science*, Vol.220, pp.671-680 (1983)

(24) P. J. M. van Laarhoven and E. H. L. Aarts: Simulated Annealing: Theory and Applications. Kluwer, Dordrecht, (1988)

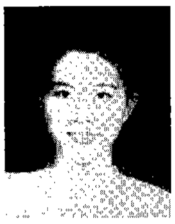
(25) D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon: "Optimization by simulated annealing: An experimental evaluation; Part 1, graph partitioning", *Oper. Res.*, Vol.37, No.6, pp.865-892 (1989)



Rong Long Wang (Student Member) received a B.S. degree from Hangzhe teacher's college, Zhejiang, China and an M.S. degree from Liaoning University, Liaoning, China in 1987 and 1990, respectively. Since 1990 he has been an Instructor in Benxi University, Liaoning, China. Now he is working toward the Ph.D degree at Toyama University, Japan. His main research interests are neural networks and optimization problems.



Zheng Tang (Non-member) received a B.S. degree from Zhejiang University, Zhejiang, China in 1982 and an M.S. degree and a D.E. degree from Tsinghua University, Beijing, China in 1984 and 1988, respectively. From 1988 to 1989, he was an Instructor in the Institute of Microelectronics at Tsinhua University. From 1990 to 1999, he was an Associate Professor in the Department of Electrical and Electronic Engineering, Miyazaki University, Miyazaki, Japan. In 2000, he joined Toyama University, Toyama, Japan, where he is currently a Professor in the Department of Intellectual Information Systems. His current research interests include intellectual information technology, neural networks, and optimizations.



Qi Ping Cao (Non-member) received a B.S. degree from Zhejiang University, Zhejiang, China and an M.S. degree from Beijing University of Posts and Telecommunications, Beijing, China in 1983 and 1986, respectively. From 1987 to 1989, she was an Assistant Professor in the Institute of Microelectronics at Shanghai Jiaotong University, Shanghai, China. From 1989 to 1990, she was a Research Student in Nagoya University, Nagoya, Japan. From 1990 to 2000, she was a Senior Engineer in Sanwa Newtech Inc., Japan. In 2000, she joined Tateyama Systems Institute, Japan. Her current research interests include multiple-valued logic, neural networks, and optimizations.