An Improved Maximum Neural Network with Stochastic Dynamics Characteristic for Maximum Clique Problem

Gang Yang  Non-member  (Toyama University, yanggang1979@hotmail.com)
Zheng Tang  Non-member  (Toyama University, tang@iis.toyama-u.ac.jp)
Hongwei Dai  Memer  (Toyama University, dai@hi.iis.toyama-u.ac.jp)

Keywords : neural network, maximum neural network, maximum clique problem, annealing strategy

A very powerful neural network approach called maximum neural network (MNN) for combinatorial optimization problems has been presented by Takefuji et al. Because of the neural network only using one term of the energy function and steepest descent method, it need not suffer the parameter-tuning to get good solutions. But it suffers the flaw of getting into local minima easily. In this paper, we overcome the flaw and propose an improved stochastic MNN for maximum clique problem.

For solving the maximum clique problem, a new graph $G_{G}$ is constructed by adding a vertex #0 that connects with all the other vertices in the complemental graph of $G$. $G$ is the graph that wants to be calculated maximum clique. The $n$-vertex MCP can be mapped onto the MNN with $2n$ neurons, where it consists of $n$ clusters of two neurons each. The $i$th neuron in $x$th cluster has input $u_{xi}$ and output $v_{xi}(x=0,...,n, i=1,2)$.

By our analysis on the experiment data and energy function, we redefine the weight formula of the added vertex #0 as follows:

$$d_{0i} = d_{0} = \frac{1}{4} \left( \sum_{j=1}^{n} u_{ji} - p \right), \quad i = 1, \ldots, n \quad \text{(1)}$$

where parameter $p$ determines the weights of the added vertex #0 with other vertices. The network can escape the local minima easily by tuning the parameter $p$. As $p$ becomes large, the added vertex #0 will have a powerful influence to the other vertices and make solution clique include more possible vertices. So the size of solution clique can be modified by tuning parameter $p$ according different instances of graphs.

In order to escape local minima, we propose a stochastic maximum neural network through adding a stochastic nonlinear self-feedback to MNN, and combine it with a flexible annealing strategy we propose. The updating function is redefined as follows:

$$\delta = \text{random}(0 \sim 1) \quad \text{(2)}$$

$$u_{xi}(t+1) = u_{xi}(t) + \psi \Delta u_{xi}(t) - T(t)(v_{xi}(t) - \delta) \quad \text{(3)}$$

$$T(t) = \frac{1}{\beta + \gamma \left( \beta + \gamma \tanh(\alpha t) \right)} T(t-1), \quad t = 1,2, \ldots \quad \text{(4)}$$

$$\beta = \beta(1 - \theta) \quad \text{(5)}$$

Where $\delta$ is a random variable producing rich stochastic dynamics by acting with the $v_{xi}(t)$. By embedding the random parameter $\delta$ in Eqs(2)(3), the improved MNN has more efficient stochastic dynamics to skip local minima.

The flexible annealing strategy in Eqs(4)(5) is embedded in the maximum neural network to control the stochastic dynamics wholly, which makes our improved MNN obtain the entire advantages of the annealing strategy. The flexible annealing strategy creates a stable and slowly decreasing temperature at the beginning that means the network with this strategy getting rich stochastic dynamics in the beginning stage. Then the temperature quickly decreases to weaken the dynamics and makes the network convergent. In Fig.1 the characteristic of our annealing strategy with different parameter values displays:

Some simulations in k random graphs and some graphs of the DIMACS clique instances in the second DIMACS challenge show that our improved network is superior to other algorithms in light of the solution quality and CPU time.
An Improved Maximum Neural Network with Stochastic Dynamics Characteristic for Maximum Clique Problem

Gang Yang∗ Non-member
Zheng Tang∗ Non-member
Hongwei Dai∗ Member

Through analyzing the dynamics characteristic of maximum neural network with an added vertex, we find that the solution quality is mainly determined by the added vertex weights. In order to increase maximum neural network ability, a stochastic nonlinear self-feedback and flexible annealing strategy are embedded in maximum neural network, which makes the network more powerful to escape local minima and be independent of the initial values. Simultaneously, we present that solving ability of maximum neural network is dependence on problem. We introduce a new parameter into our network to improve the solving ability. The simulation in k random graph and some DIMACS clique instances in the second DIMACS challenge shows that our improved network is superior to other algorithms in light of the solution quality and CPU time.

Keywords: neural network, maximum neural network, maximum clique problem, annealing strategy

1. Introduction

Maximum clique problem is one of the most important combinatorial optimization graph problem, which can be utilized in many important applications in project scheduling, cluster analysis, facility location problems, and other problems from operations research. The problem is computationally intractable, even to be approximated with certain absolute performance bounds. It is generally believed that the computational power needed to solve it grows exponentially with the number of vertices and edges density. To seek an optimal solution, a lot of exact methods have been so far proposed to deal with the combinatorial optimization problem, such as cutting plane methods, branch-and-bound methods, and so on. Unfortunately, no exact polynomial algorithm, however, has been found for any NP-hard problem up to now. So many researchers have been paying attention, instead, on approximate or heuristic algorithms such as neural network algorithms which seek near-optimal solutions at a reasonable computational cost without ensuring optimality or feasibility.

Neural networks have been shown to be a powerful tool for combinatorial optimization problems, especially for NP-hard problems. A lot of algorithms based on neural network method are presented to solve the maximum clique problem, including many Hopfield-type network algorithms and maximum neural network algorithms. Lai et al. use a Hopfield network to solve the problem, but their energy representation is different from those used by Ramanujam and Sadayappan and are based on logical functions.

Jagota has considered the maximum clique problem alone and presents several energy minimizing dynamics of a Hopfield network, both discrete and continuous. Funabiki compares some energy-descent optimization algorithms for maximum clique problem and proposes an efficient binary neural network which suits for solving k random graphs. The first parallel algorithm using a maximum neural network proposed by Lee et al. The maximum neural network always guarantees a valid solution and reduces the search space without a burden on the parameter-tuning. Unfortunately, the maximum neural network easily converges to an oscillation state of local minimum because it is based on the steepest descent method and has not powerful ability to escape the trap of local minima. So some researchers embed additional dynamics into neural network to increase network dynamics characteristic and avoid the minimum problem. Just like transient chaotic neural network(TCNN), Wang et al. propose an improved maximum neural network which has nonlinear self-feedback to escape local minima. But the network has not enough dynamics at the beginning, and its annealing strategy is unfit for controlling dynamics. So in this paper, we propose an improved maximum neural network with stochastic dynamics to increase the network ability of solving maximum clique problem.

In this paper, we propose an improved parallel algorithm that can help the maximum neural network escape from local minima and has powerful ability of searching the globally optimal or near-optimal solution for the maximum clique problem. A stochastic nonlinear self-feedback is embedded in the maximum neural network, which creates more efficient stochastic dynamics to the network. We analyze the influence on weight setting among the added vertex and other vertices, introduce a
new parameter to control the weight initial setting. In order to make the maximum neural network have enough random dynamics characteristic at the beginning, a flexible annealing strategy is introduced to the improved network to control the dynamics. With the novel strategy, the improved network can balance the stochastic dynamics and neurodynamics flexibly, which makes the improved network escape local minimum efficiently and converge quickly. The simulation in k random graph and the DIMACS clique instances in the second DIMACS challenge verifies our proposed algorithm.

This paper is organized as follows: In the next section, the maximum clique problem is described. The improved maximum neural network is presented in section 3. In section 4, the simulation on maximum clique problem in k random graph and benchmark graphs is shown. Finally we give the solution about our paper.

2. Maximum Clique Problem

In a graph with undirected edges, a clique is a set of vertices such that every pair of connected by an edges. A clique is maximal if no strict superset of it is also a clique. A k clique is a clique of size k. A clique is maximum if it is the largest clique. Let $G = (V, E)$ be an arbitrary undirected graph, where $V = \{v_1 \cdots v_n\}$ is the vertex set of G, and $E \subseteq V \times V$ is the edge set of G. $A = (a_{ij})_{n \times n}$ is the adjacency matrix of G, where $a_{ij} = 1$ if $(i, j) \in E$, and $a_{ij} = 0$ if $(i, j) \notin E$. Given a subset $S \subseteq V$, we call $G(S) = (S; E \cap S \times S)$ the subgraph induced by S. A graph $G = (V, E)$ is complete if all its vertices are pairwise adjacent. The MCP requires a clique that has the maximum cardinality. As above definition, the binary variables $d_{ij}$ are defined as follows.

$$d_{ij} = \begin{cases} 1 & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The state of neuron $v_i$ is determined by

$$v_i = \begin{cases} 1 & \text{if the } i\text{-th vertex in the clique} \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The maximum clique problem is one of the first problems which has been proved to be NP-complete (5). Moreover, even its approximations within a constant factor are NP-hard (6). However this problem is quite important because it appears in a lot of real world problems. Many important intractable problems turn out to be easily reducible to MCP, for example, the Boolean satisfiability problem, the independent set problem, the subgraph isomorphism problem, and the vertex covering problem. Moreover, the maximum clique problem has many important practical applications, especially in information retrieval, economics, VLSI design and computer vision.

3. The Improved Maximum Neural Network for MCP

A very powerful neural network approach called maximum neural network (MNN) for combinatorial optimization problems has been presented by Takefuji et al. It has been proved that the algorithm performs better than the best-known algorithm in solving some other optimization problems (14, 15). Because of the neural network only using one term of the energy function and steepest descent method, it need not suffer the parameter-tuning to get good solutions.

In Ref. (19), Barahona et al. showed how an unconstrained quadratic 0-1 programming problem is equivalent to minimize the weight summation over the same partition in a newly constructed graph $G_M$ with vertex set $V_M = \{0, 1, 2, \ldots, n\}$. As shown in Fig. 1, a new graph $G_M$ is constructed by adding a vertex #0 that connects with all the other vertices of the complemental graph of G. G is the graph that wants to be calculated maximum clique. The n-vertex MCP can be mapped onto the MNN with $2 \times n$ neurons, where it consists of n clusters of two neurons each. The ith neuron in xth cluster has input $u_{xi}$ and output $v_{xi}(x = 0, \ldots, n, i = 1, 2)$. Lee and Takefuji (14, 15) formulated the MCP problem as the global minimization of the function:

$$e = \sum_{x=0}^{n} \sum_{y=0}^{n} \sum_{i=1}^{2} d_{xy}v_{xi}v_{yi} \quad \quad (3)$$

The weight matrix is defined by:

$$d_{0i} = d_{0} = \frac{1}{4} \left( \sum_{j=1}^{n} a_{ij} - 1 \right), i = 1, \ldots, n \quad \quad (4)$$

$$d_{ii} = 0, d_{ij} = \frac{1}{4} a_{ij} \quad \forall i \neq j, i, j = 1, \ldots, n \quad \quad (5)$$

where $\{a_{ij}\}$ is the adjacency matrix of the complement of G. In practice, the motion equation of the ith neuron
in the xth cluster without the decay term to minimize the summation of the weights is given by

$$\Delta u_{xi} = - \sum_{y=0}^{n} d_{xy} v_{yi}$$  \hspace{1cm} (6)

So the updating function is \( u_{xi}(t+1) = u_{xi}(t) + \Delta u_{xi}(t) \). The input/output function of the xth neuron in the xth cluster is given by

$$v_{xi} = 1 \quad \text{if} \quad u_{xi} = \max\{u_{x1}, u_{x2}\}; 0 \quad \text{otherwise} \quad \hspace{1cm} (7)$$

The function \( \max\{\} \) returns the first argument with the maximum value. Due to vertex \#0 always belonged to the clique, the value of \( v_{x0} \) is set 1 at all times. Because MNN is based on the steepest descent method, it has a tendency to easily converge to a local minimum. So we propose an improved maximum neural network through adding a stochastic nonlinear self-feedback to the MNN, and combine it with a flexible annealing strategy we present. The updating function is redefined as follows:

$$\delta = \text{random}(0 \sim 1) \quad \hspace{1cm} (8)$$

$$u_{xi}(t+1) = u_{xi}(t) + \psi \Delta u_{xi}(t) - T(t)(v_{xi}(t) - \delta) \quad \hspace{1cm} (9)$$

$$T(t) = \frac{1}{\beta + \gamma \left[ \beta + \gamma \tanh(\alpha t) \right]} [T(t-1)], \quad t = 1, 2, \ldots \quad \hspace{1cm} (10)$$

$$\beta = \beta(1 - \theta) \quad \hspace{1cm} (11)$$

Where \( \delta \) is a random variable producing rich dynamics by acting with the \( v_{xi}(t) \). By embedding the random parameter \( \delta \), our improved MNN has more efficient dynamics to skip local minima than MNN and the other improved MNN methods\(^{(14),(16)}\). The original MNN does not have the self-feedback dynamics and it is easy to get stuck into local minima. Although the improved method by wang et al. has some kind chaotic dynamics to escape local minima, its chaotic dynamics is not obvious to solve the problem. By using our improved network, the solution quality is no longer determined by the initial state selection of neuron inputs obviously, because the stochastic dynamics make network get adaptive values of \( u_{xi} \) at the beginning. The variable \( T(t) \) can be interpreted as the strength of negative self-feedback connection of each neuron. The stochastic dynamics on selected vertices are shown in Fig.2, in which the improved maximum neural network solve the instance c-fat200-1 of maximum clique problem. In order to describe the stochastic dynamics of competitive internal value, Fig.2 show the difference of \( u_{x1} \) and \( u_{x2} \) which decides the output value. As we can see from Fig.2, in the beginning the stochastic dynamics is well-distributed, which means the network does global search. With the network proceeding, the intensity of stochastic dynamics decreases and the difference of \( u_{x1} \) and \( u_{x2} \) gradually change slightly. When the difference reaches zero or unidirectional changing value, the vertex reaches a saturated state. The saturate state \( u_{x1} - u_{x2} \geq 0 \) means the vertex belong to the maximum clique. Otherwise, it is not the maximum clique vertex.

In order to analyze MNN character, we formulate Eq.(3) as follows:

$$e = \sum_{x=0}^{n} \sum_{y=0}^{n} \sum_{l=0}^{2} d_{xy} v_{xi} v_{yi}$$

$$= \sum_{x=0}^{n} \sum_{y=0}^{n} d_{xy} v_{x1} v_{y1} + \sum_{x=0}^{n} \sum_{y=0}^{n} d_{xy} v_{x2} v_{y2}$$

$$= \sum_{x=1}^{n} \sum_{y=1}^{n} d_{xy} v_{x1} v_{y1} + \sum_{x=1}^{n} \sum_{y=1}^{n} d_{xy} v_{x2} v_{y2}$$

$$+ \sum_{x=0}^{n} d_{x0}(v_{x1} v_{01} + v_{x2} v_{02})$$

$$+ \sum_{y=0}^{n} d_{0y}(v_{01} v_{y1} + v_{02} v_{y2})$$

\( \therefore \quad v_{01} \equiv 1, \quad v_{02} \equiv 0 \)

$$= \sum_{x=1}^{n} \sum_{y=1}^{n} d_{xy} v_{x1} v_{y1}$$

$$+ \sum_{x=1}^{n} (d_{x0} v_{x1}) + \sum_{y=1}^{n} (d_{0y} v_{y1})$$

$$= e' + \sum_{x=1}^{n} (d_{x0} v_{x1}) + \sum_{y=1}^{n} (d_{0y} v_{y1}) \hspace{1cm} (12)$$
From Eq. (12) we find that if there is not have the adding vertex effect ε′ will be a constant value to a special instance. It is dependence on problem. For solving a designated instance, we should tune $\sum_{i=1}^{n} a_{ij} - p$, $i = 1, ..., n \cdots (13)$

Parameter $p$ determines the weights of the added vertex #0 with other vertices. With setting a suitable value to parameter $p$, the network can escape local minima easily. Roughly speaking, as $p$ becomes large, the added vertex #0 would have a powerful influence to the other vertices and make solution clique include more possible vertices. So the size of solution clique can be modified by tuning parameter $p$ according different instances of graphs. The energy convergence status with different value of $p$ are shown in Fig. 3 at the instance Johnson16-2-4.

In Fig. 3(a), we can see that the energy value will maintain some iterations to get convergence at the end phase with parameter $p$ set as 1. But it is still efficient to get the near-optimal solution. The added vertex has not enough power to make good competition among vertices. So until the competing internal values $u_{xi} (i = 1, 2)$ are equal to each other, the network will convergent the saturate point. When $u_{x1} = u_{x2} (i = 1, 2)$, we set $v_{x1}=1$ and $v_{x2}=0$. When parameter $p$ is set as 4 in Fig. 3(b), the competing degree is increased and the network has enough ability to converge quickly and get the optimal result directly. So at the end phase the network converge quickly. It is proved that the maximum neural network has feasible ability to converge and find the optimal or near-optimal result, and the tuning parameter $p$ makes the solution quality controllable.

We propose a flexible annealing strategy in Eqs (10-11) to control the stochastic dynamics wholly, which makes the improved MNN obtain the entire advantages of the annealing strategy. The flexible annealing strategy creates a stable and slowly decreasing temperature at the beginning that means the network with this strategy getting rich stochastic dynamics in the beginning stage. Then the temperature quickly decreases to weaken the stochastic dynamics and makes the network converge. In Fig. 4 the character of our annealing strategy displays:

In Eq. (9), the parameter $\psi$ represents also the influence of the energy function on the neurodynamics, or balance between the self-feedback term inducing the mechanism of escaping from local minima and the gradient term $\Delta u_{xi}(t)$ inducing the convergent dynamics.

The improved maximum neural network with flexible annealing strategy has rich dynamics with various coexisting attractors. The new updating function with stochastic dynamic can help the maximum neural network escape from local minima and converge to the global-minimum or near-global minimum. Given randomly generated numbers for the initial state of $u_{xi}(0)$ and a befitting large initial self-feedback $T(0)$, the competition and state transient of neurons are drastic when iteration runs as Eqs. (8-11), which breaks the monotonic gradient descent dynamic and can help the network to escape from the local minima. When there are no neurons changing the output, the algorithm will be terminated and the total number of vertices in the maximum clique is given by $\sum v_{x1}, x = 1, 2 \cdots n$.

An adequately strong stochastic dynamics in the beginning stage is important to get good solutions. The proposed improved MNN has flexible rich dynamics by tuning the parameters of the annealing strategy, which makes the network’s solution be independent on the ini-
tial neurons’ values. The flexible annealing strategy gives the network enough stochastic dynamics to prevent it to get stuck at local minima, and then the stochastic dynamics decreases quickly. When the stochastic dynamics vanishes, the proposed algorithm is then fundamentally reigned by the gradient descent dynamics and usually converges to a stable equilibrium point like the MNN. So the proposed algorithm has the advantages of both MNN and stochastic networks.

The evolution of energy function is drastic because of its rich stochastic dynamics, which sufficiently proves that the improved MNN is powerful to escape local minima and search near-optimal and optimal solutions. The energy evolutions of three different algorithms in solving Hamming6-2 graph are shown in Fig.5. Original MNN algorithm suffers the choice of right initial values and the local minima, which displays the lightly oscillation of energy in Fig.5 (a) Although it converge quickly, it cannot find the optimal result in our tests with limited iterations. Fig.5 (b) shows the energy evolution of the improved algorithm proposed by Wang et al. Although a nonlinear self-feedback is added in the updating function, it is not enough to make the network independent on the initial values and escape the local minima. Moreover, the annealing strategy used in Ref. (16). is proposed by Kirkpatrick et al (20). The annealing strategy is not good because its temperature does not permeate the whole domain of the annealing process (21). Our improved maximum neural network has stochastic negative self-feedback to increase network dynamics, and has a flexible annealing strategy to control and balance neurodynamics and stochastic dynamics. The energy of our algorithm has smarter oscillation and succeeds in skipping local minima and shaking off the burden of initial neurons values obviously. Moreover the convergence rate is very faster than Wang, et al algorithms as Fig.5 (c)(d) shows. The effect of our flexible annealing strategy is clear in our improved MNN. Due to introduce stochastic dynamics self-feedback into our network, our improved neural network have more dynamics than the other two networks.

4. Simulations and Results

The algorithm discussed earlier is implemented on solving the maximum clique problem to test its efficiency and feasibility in P4 3.0G 1G RAM. In accordance with Jagota (12) (13), our algorithm compared with several other algorithms is tested in a difficult type of random graphs: k random graphs. A k random graph is a graph which is a union of randomly generated k cliques of various sizes. It is usually difficult to obtain the maximum clique for k random clique graphs. And the k random graph is more suitable than p random graph to evaluate algorithms’ efficiency. Thus, through solving k random clique graphs, it is possible to separate poor algorithms from good ones for MCP (11).

In Fig.6, the probability of successfully solving some challenges of second DIMACS benchmark is shown. We set parameter p as 2 and set random initial values in our test. In this test it is about 100 percents to get the near-optimal and optimal result by our improved maximum neural network. It is revealed that our proposed neural network is not depended on the initial values obviously. In order to evaluate algorithms’ character, we simulate and compare these algorithms in some k random graphs. We execute every algorithm 30 times on different graphs, and the average results are summarized in Table1. RaCLIQUE (18), Binary neural network(BNN) (11), the improved maximum neural network by Wang et al. (16), and our proposed maximum neural network are compared in result sizes and executive time to evaluate their efficiency. In Table 1, the column “Avg” denotes the average sizes of results found by the relative algorithm. The column “Time” expresses the average cost time of every algorithm in the test. The columns “Nodes”, “K”, “Edges” and “Density” represent the number of its vertices, the random graph number, total edges of the graph, and its edge density. In every time the random initial values are set to the network. In the test we modify parameter p until to find the best solution. The comparing algorithms use the parameter setting like references (11) (16) (18), the other fixed parameters are set as follows:

![Fig. 5. The comparison of energy oscillation of different algorithms for Hamming6-2 graph. (a) MNN proposed by Takefuji et al. (b) CMNN proposed by Wang et al. (c) Our improved MNN with γ=1 and θ=0.01. (d) Our improved MNN with γ=1 and θ=0.05.](image-url)
In the test, if we increase the convergent rate of wang’s algorithm\(^{(16)}\), the solution quality is affect obviously because the beginning dynamics disappear too quickly to escape the local minima in time. Our improved algorithm is clearly nicer in this aspect.

In order to evaluate not just the relative, but also the absolute performance quality, we tested and compared these algorithms on the second DIMACS benchmark graphs. From the Table 2, we can see that it is efficient to get the optimum solutions by our improved MNN. It cost a few steps to get more feasible result. Here one step means one cyclic updating of all neurons. In the test, the parameters are tuned to make our algorithm search larger domain in order to find the optimal solution. The results of numerical experiments suggest that the improved MNN is superior to RaCLIQUE, BNN,TCNN and the MNN improved by Wang et al. in the absolute performance quality, we tested and compared these algorithms on the second DIMACS benchmark graphs.
5. Conclusions

The paper presented an improved stochastic maximum neural network for approximating the maximum clique problem. A stochastic dynamic mechanism is embedded in the network, which makes the maximum neural network escape the local minima efficiently and get the near-optimal or optimal results successfully. The proposed network has a more flexible annealing strategy to control the stochastic dynamics. By using the strategy, the network converges quickly and retain the dynamics character. Simultaneously, we present that solving ability of maximum neural network is dependence on problem. We introduce a new parameter into our network to improve the solving ability. The simulations with comparing different algorithms on maximum clique problem in $k$ random graphs and hard DIMACS instances show that our improved maximum neural networks can get efficient results with good quality in less time and iteration steps.

(Manuscript received Nov. 22, 2006, revised May 18, 2007)

References


Gang Yang (Non-member) received the B.S. degree and M.S. degree from Shandong University, Shandong, China in 2002 and 2005 respectively. Now he is working toward the Ph.D degree at University of Toyama, Japan. His main research interests are artificial neural network, pattern recognition and optimizations.

Zheng Tang (Non-member) received the B.S. degree from Zhejiang University, Zhejiang, China in 1982 and an M.S. degree and a D.E. degree from Taihinghu University, Beijing, China in 1984 and 1988 respectively. From 1988 to 1989, he was an Instructor in the Institute of Microelectronics at Tshinhua 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 artificial information technology, neural networks, and optimizations.

Hongwei Dai (Member) received the B.S. degree and M.S. degree from Xi’an Jiaotong University, Shaanxi, China in 1999 and 2002 respectively. He joined the Institute of Mechatronics and Information System at Xi’an Jiaotong University in 2002. Now, he is working toward the Ph.D degree at Toyama University, Toyama, Japan. His main research interests are artificial immune system and artificial intelligence.