An Improved Chaotic Maximum Neural Network for Maximum Clique Problem

Gang Yang†, Zheng Tang†, and Junyan Yi†

† Faculty of Engineering, Toyama University, Toyama, 930-8555 Japan

Summary
We propose an improved chaotic maximum neural network to solve maximum clique problem. Through analyzing the character of maximum neural network with an added vertex in maximum clique problem, we find that the quality and size of clique solution can be modified by tuning a parameter about vertex weight. Based on the analysis, a random 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. The simulation in *k* random graph and some graphs of the 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.

Key words:
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 [5]. It is generally believed that the computational power needed to solve it grows exponentially with the number of vertices and edges density[7],[8].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.[4]

Neural networks have been shown to be a powerful tool for combinatorial optimization problems, especially for NP-hard problems [1],[2],[3]. 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. [9] use a Hopfield network to solve the problem, but their energy function representations are different from those used by Ramanujam and Sadayappan [10] and are based on logical functions. Jagota[12],[13] has considered the maximum clique problem alone and presents several energy minimizing dynamics of a Hopfield network, both discrete and continuous. Funabiki[11] 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.[14],[15] 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 the 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.[17] So some researchers embed chaotic dynamics into neural network to avoid the minimum problem. Just like transient chaotic neural network, Wang et al.[16] propose a chaotic maximum neural network which has self-feedback character. But it has not enough chaotic dynamics at the beginning, and the annealing strategy is unfit for controlling chaotic dynamics. So in this paper, we propose an improved maximum neural network to increase the network ability of solving the maximum clique problem.

In this paper, based on maximum neural network, 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 random nonlinear self-feedback is embedded in the maximum neural network, which creates more efficient chaotic dynamics to the network. We analyze the influence on weight setting among the added vertex and other vertices, and introduce a new parameter to control the

Manuscript received February 5, 2007.
Manuscript revised February 25, 2007.

weight initial setting. In order to make the maximum neural network have enough chaos at the beginning, a flexible annealing strategy is introduced to the improved network to control the chaotic dynamics. With the strategy, the improved network can balance the chaotic 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*. $A=(d_{ij})\ n \times n$ is the adjacency matrix of *G*, where $d_{ij}=1$ if $(i,j) \in E$, and $d_{ij}=0$ 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.

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

The state of neuron v_i is determined by

$$v_i = \begin{cases} 1 & if\ the\ \#i\ vertex\ in\ the\ clique \\ 0 & otherwise \end{cases} \tag{2}$$

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.

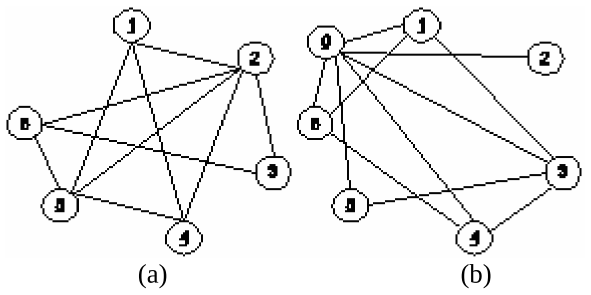


Fig.1 The graph *G* and G_M with added vertex #0. The clique of vertices 1, 2, 4 and 5 is the maximum clique. (a).A graph *G* with 6 vertices and 10 edges (b).The graph G_M of *G* with the added vertex #0.

In Ref.[19], Barahona et al. show 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 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 $2 \times n$ 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). Lee and Takefuji [14][15] formulated the MCP problem as the global minimization of the function:

$$e=\sum_{x=0}^{n}\sum_{y=0}^{n}\sum_{i=1}^{2} d_{xy} V_{xi} V_{yi} \tag{3}$$

The weight matrix is defined by:

$$d_{0i} = d_{i0} = \frac{1}{4}\left(\sum_{j=1}^{n}\overline{a_{ij}} - 1\right), i = 1,\dots, n \quad (4)$$

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

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

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

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

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

The function *max*{} returns the first argument with the maximum value. Due to vertex #0 always belonged to the clique, the value of v_{x1} 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 a chaotic maximum neural network through adding a random chaotic nonlinear self-feedback to the MNN, and combine it with a flexible annealing strategy we propose. The updating function is redefined as follows:

$$\delta = random\ (0 \sim 1) \quad (8)$$

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

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

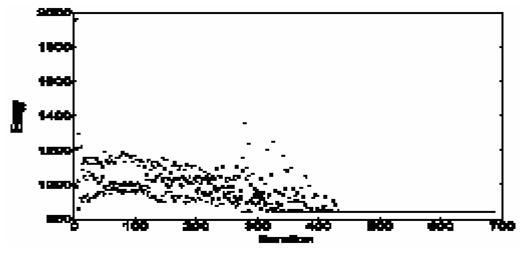
$$\beta = \beta(1-\theta) \quad (11)$$

Where δ is a random variable producing rich chaotic dynamics by acting with the $v_{xi}(t)$. By embedding the random parameter δ, the chaotic MNN has more efficient chaotic dynamics to skip local minima than MNN and other improved MNN methods.[14][16] The solution quality is no longer determined by the initial state selection of neuron inputs obviously, because chaotic dynamics make the network get random and 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.

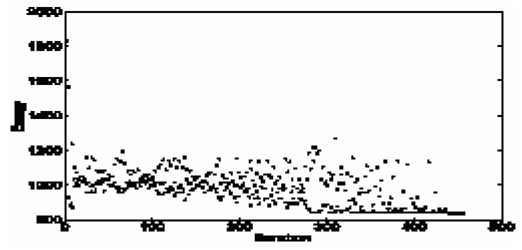
Based on experiment and method calculation, we find the Eq.(4) should be modified as follows:

$$d_{0i} = d_{i0} = \frac{1}{4}\left(\sum_{j=1}^{n}\overline{a_{ij}} - p\right),\ \ i = 1,\cdots,n \quad (12)$$

Because parameter *p* determines the weights of the added vertex #0 connected with other vertices. The network can escape the local minima easily by tuning the parameter *p*. With the increase of *p* value, the added vertex #0 will have a powerful influence to the other vertices and make solution clique include more possible vertices. So the quality and size of solution clique can be modified by tuning parameter *p* according different instances of graphs. The energy convergence statuses with different value of *p* are shown in Fig.2 at the instance of Johnson16-2-4 graph.



(a)Parameter *p*=1



(b)Parameter *p*=4

Fig.2. The energy evolvement of our proposed neural network with different parameter *p*

In Fig.2(a), we can see that the network will contain many iterations with same energy value to reach the convergence status at the end phase of network convergence 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 overwhelming competition among vertices. So until the competing internal values u_{xi} (i =1, 2) are equal to each other, the network converges the saturate point. The process will cost much time, which should be improved. When parameter *p* is set as 4 in Fig.2(b), the intense competition phase at the end of convergence vanishes and the network can converge and get the optimal result directly. 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 chaotic dynamics wholly, which makes the chaotic 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 chaotic dynamics in the beginning stage. Then the temperature quickly decreases to weaken the chaotic dynamics and makes the network convergent. In Fig.3 the character of our annealing strategy displays:

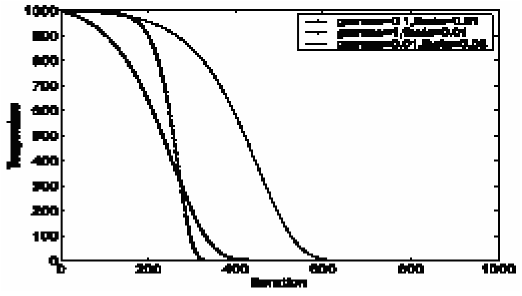
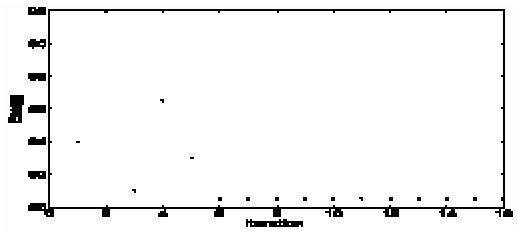


Fig.3. The temperature evolvement of our annealing strategy

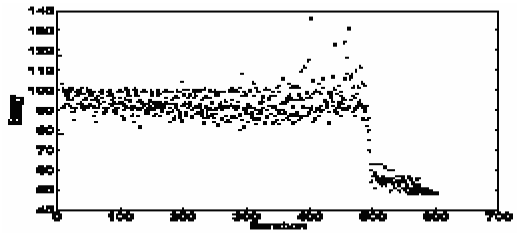
In Eq.(9),the parameter ψ represents also the influence of the energy function on the neurodynamics, or balance between the self-feedback term inducing the mechanism of escaping from local minima and the gradient term $\Delta u_{xi}(t)$ inducing the convergent dynamics.

The improved random chaotic neural network with flexible annealing strategy has rich dynamics with various coexisting attractors not only of fixed points but also periodic and even chaotic attractors. The new updating function with chaotic 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 $\sum v_{x1}, \quad x = 1,2 \cdots n.$.

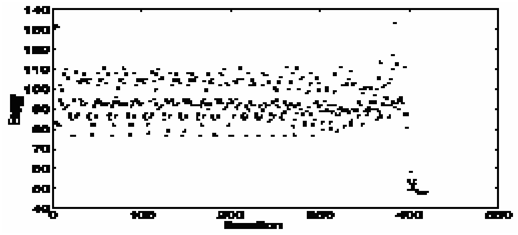
An adequately strong chaotic dynamics in the beginning stage is important to get good solutions. The proposed chaotic MNN has flexible rich dynamics by tuning the parameters of the annealing strategy, which makes the network's solution be independent on the initial neurons' values. The flexible annealing strategy gives the network enough chaos to prevent it to get stuck at local minima, and then chaotic dynamics decreases quickly. When the chaotic dynamics vanishes, the proposed algorithm is then fundamentally reined by the gradient descent dynamics and usually converges to a stable equilibrium point like the MNN which doesn't have the burden on the parameter-tuning. So the proposed algorithm has the advantages of both MNN and chaotic networks.



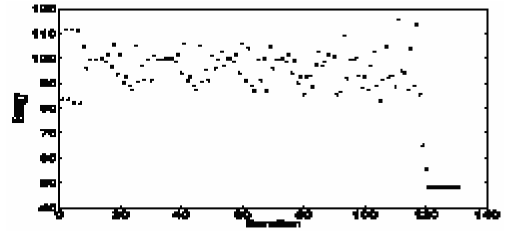
(a).MNN proposed by Takefuji et al.



(b).CMNN proposed by Wang et al.



(c).Our proposed improved MNN with γ=1 and θ=0.01



(d).Our proposed improved MNN with γ=1 and θ=0.05

Fig.4. The comparison of energy oscillation of different algorithms for Johnson16-2-4 graph.

The evolution of energy function is drastic because of its rich chaotic dynamics, which sufficiently proves that the chaotic MNN is powerful to escape the local minima and search the near-optimal and optimal solutions. The energy evolutions of three different algorithms in solving Hamming6-2 graph are shown in Fig.4. The original MNN algorithm suffers the choice of right initial values and the local minima, which displays the lightly oscillation of energy in Fig.4.(a). Fig.4.(b) shows the energy evolution of the improved algorithm proposed by Wang. et al. Although some chaotic dynamics is added in their updating function, it is not enough to make the network independent on the initial values and escape the local minima efficiently. Moreover, the annealing strategy used in Ref.[16] is proposed by Kirkpatrick et al.[20] The annealing strategy is not good to balance chaos because the annealing temperature does not permeate the whole domain of the annealing process.[21] Our new chaotic maximum neural network has random negative self-feedback to increase the chaotic dynamics, and has the flexible annealing strategy to control and balance the neurodynamics and chaotic dynamics. As Fig.4(c)(d) shown, the energy of our algorithm has smarter oscillation and succeeds in skipping the local minima and shaking off the burden of initial neurons values obviously. Moreover the convergence rate is very faster than Wang. et al algorithms . The effect of our flexible annealing strategy is clear in the chaotic MNN.

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 test 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 order to evaluate algorithms' character, we simulate and compare several 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. The testing parameters are set as follows:

β =500, γ =1, α=0.9998, ψ =0.01, $T(0)$=0.15, k=0.9, ε =0.004, θ =0.01,

In the *k* random graphs test, our improved MNN can find more feasible solutions with less time than MNN proposed by Wang et al. and RaCLIQUE. In Ref.[11], BNN was presented as the best synchronous algorithm to solve *k* random graph. But in our simulation, we find although BNN sometimes costs less time to converge to a stable status than our improved network, it's solution quality is not as good as our algorithm further. Our proposed MNN can find more valuable results by introducing random self-feedback and the flexible annealing strategy. It means that the improved MNN is efficient and feasible in terms of the solution quality and cost time. Moreover, the exact parameter *p* is easy to find, which can increase the solution quality and adaptability. If we increase the convergent rate of Wang's algorithm [16], the solution quality is affect obviously because the chaotic 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 Table2, 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. So 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 light of the solution size and CPU time.

5. Conclusion

The paper presented an improved chaotic maximum neural network for approximating the maximum clique problem. A random 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. The proposed network has a more flexible annealing strategy to control the chaotic dynamics. By using the strategy, the network convergent quickly and retain the dynamics character. The simulations with comparing different algorithms on maximum clique problem in *k* random graphs and hard DIMACS instances show that our proposed chaotic maximum neural networks can get efficient results with good quality in less time and iteration steps.

Table 1. Simulation results on the *k* random graphs

Nodes	K	Edges	Density	RaCLIQUE		BNN		MNN wang et al.		Our proposed MNN	
				Avg	Time(s)	Avg	Time(s)	Avg	Time(s)	Avg	Time(s)
100	10	1570	0.317	32	5.297	32	0.016	32	2.14	32	0.427
100	10	2605	0.526	36	5.024	35.4	0.016	36	2.69	36	0.432
100	20	3728	0.753	39	3.172	39	1.376	45	2.19	46	0.403
100	20	3803	0.768	40	2.962	39.2	2.590	44	2.04	45	0.419
100	50	4564	0.922	55	3.094	55	1.211	62	3.24	62	0.483
100	50	4535	0.916	58	2.995	58	2.909	61	3.56	61	0.482
200	10	7881	0.396	70	46.01	70	0.016	70	3.67	70	0.775
200	10	8916	0.448	77	48.88	77	0.016	77	2.42	77	0.848
200	20	15283	0.768	80	46.718	73.2	1.695	76	3.33	76	0.784
200	20	14667	0.737	79	50.52	69	3.739	72	3.89	72	0.890
200	50	18833	0.946	109	51.30	109	4.503	119	6.87	119	1.218
200	50	19301	0.970	122	48.44	122	4.157	134	7.79	144	1.321
300	10	19039	0.425	105	249.4	105	0.047	105	5.89	105	1.818
300	10	17594	0.392	122	258.8	122	0.044	122	5.72	122	2.724
300	20	32938	0.734	116	247.9	105	0.658	102	6.78	102	2.890
300	20	34054	0.759	112	255.5	108	0.784	107	8.62	107	2.750
300	50	43165	0.962	172	228.9	171	6.369	187	7.84	187	2.290
300	50	43647	0.973	183	239.1	179	8.421	208	8.86	208	2.800

Table 2. Simulation results on the second DIMACS benchmark graphs

Name	N	Density	C(G)	TCNN		MNN wang et al.		Our proposed MNN	
				Size	Steps	Size	Steps	Size	Steps
MANNa9	45	0.927	16	15	704	15.4	1084	15.8	460
Hamming6-2	64	0.905	32	32	603	32	1030	32	448
Hamming6-4	64	0.349	4	4	594	3.8	1074	4	446
Hamming8-2	256	0.969	128	128	642	128	1029	128	449
c-fat200-1	200	0.077	12	12	528	12	1021	12	448
c-fat200-2	200	0.163	24	19	615	24	1021	24	447
c-fat200-5	200	0.426	58	58	621	58	1024	58	446
c-fat500-1	500	0.036	14	12	609	14	1123	14	442
c-fat500-2	500	0.073	26	23	624	26	1021	26	448
c-fat500-5	500	0.183	64	64	636	64	1152	64	423
Johnson8-2-4	28	0.556	4	4	597	3.8	1117	4	442
Johnson8-4-4	70	0.768	14	12	599	14	1017	14	447
Johnson16-2-4	120	0.765	8	7	622	8	1054	8	451
Johnson32-2-4	496	0.879	16	15	617	15.9	1028	16	462
P-hat300-1	300	0.244	8	8	608	7.6	1018	8	458
P-hat300-2	300	0.489	25	23	612	24.8	1486	25	479
P-hat300-3	300	0.744	36	32	623	33.6	1364	35.8	472
P-hat500-1	500	0.253	9	8	592	9	1997	9	587
P-hat500-2	500	0.505	36	33	603	36	1987	36	532
P-hat500-3	500	0.752	>=49	44	632	48	1587	49.5	651
P-hat700-1	700	0.249	11	7	647	9	1782	10.2	722
P-hat700-2	700	0.498	44	43	632	44	1980	44	674
P-hat700-3	700	0.748	>=62	57	662	59.8	1898	61.2	760

References

[1] J.J. Hopfield and D.W. Tank: "Neural computation of decisions in optimization problems", Biological Cybernetics, Vol.52, pp. 141-152 (1985).

[2] J.J. Hopfield and D.W. Tank: "Computing with neural circuits: a model", Science, Vol.233, pp.625-633 (1986)

[3] D.E.Vanden Bout and T.K.Miller: "Improving the performance of the Hopfield-Tank neural network through normalization and annealing", Biological Cybernetics, Vol62,pp. 129-139(1989).

[4] L. Chen and K. Aihara, Chaotic simulated annealing by a neural networkmodel with transient chaos, Neural Networks, Vol.8, pp.915–930(1995).
[5] R.M.Karp: "Reducibility among Combinatorial Problems", in: R.E. Miller and J.W.Thatcher(ed.), Complexity of Computer Computation, pp.85-103, Plenum Press, New York(1972)
[6] U. Feige,S. Goldwasser,S. Safra,L. Lovasz,and M. Szegedy : "Approximating clique is almost NP-complete",In Proc. of the 32*nd* Annual IEEE Symposium on the Foundations of Computer Science,pp.2-12(1991).
[7] M.R. Garey,D.S.Johnson and L.J.Stockmeyer,Some simplified NP-complete graph problem, Theor.Comput.Sci.,1(1976)237-267.
[8] S.Even and Y.Shiloach, NP-completeness of several arrangement problems, Technical Report 4, Department of Computer Science, Technion(Haifa Israel, 1975).
[9] J.S.Lai,S.Y.Kuo, and I.Y.Chen, Neural Networks for Optimization Problems in Graph Theory, Proceedings IEEE International Symposium on Circuits and Systmes 6,269-272
[10] J.Ramanujam and P.Sadayappan,Mapping Combinatorial Optimization Problems onto Neural Networks, Information Sciences 82,239-255(1995)
[11] N. Funabiki and S. Nishikawa, Comparisons of Energy-Descent Optimization Algorithms for Maximum Clique Problems, IEICE Trans. Fundamentals, E79-A,4(452-460)1996.
[12] A. Jagota, "Approximating maximum clique with a Hopfield network," IEEE Trans. Neural Networks, vol.6, no.3,pp.724-735,May 1995.
[13] A. Jagota, L. Sanchis, and R. Ganesan, "Approximately solving maximum clique using neural networks and related heuristics," In Cliques, Coloring, and Satisfiability, D. Johnson and M. Trick, Eds. Providence, RI: American Mathematical Society, 1996, pp. 169-204.
[14] K.C. Lee, N.Funabiki and Y. Takefuji, A parallel improvement algorithm for the bipratite subgraph problem, IEEE Trans.Neural Networks 3(1) (1992)139-145
[15] Y. Takefuji, K.Lee and H.Aiso, An artificial maximum neural network: A winner-take-all neuron model forcing the state of the system in a solution domain, Biol.Cybern.67(3) (1992)243-251.
[16] J Wang,Z Tang,R Wang, Maximum neural network with nonlinear self-feedback for maximum clique problem. Neurocomputing 57, pp.485-492(2004).
[17] Y.Takenaka, N.Funabiki and T.Higashino, A proposal neural filter: A constraint resolution scheme of neural networks for combinatorial optimization problems, IEICE Trans. Fundamentals E83-A(9)(2000)1815-1823.
[18] Y.Yamada, E.Tomita, and H.Takahashi, A randomized algorithm for finding a near-maximum clique and its experimental evaluations, Trans.IEICE, Vol.J76-D-I,no.2,pp.46-53(1993).
[19] F.Barahona, M.Junger, G.Reinelt, Experiments in quadratic 0-1 programming, Math. Programming 44(2)127-137(1989).
[20] S.Kirkpatrick,C.D.Gelatt Jr.,M.P.Vecchi,Optimization by simulated annealing, Science N.Y.220(1983)671
[21] Jzau-sheng Lin, Annealed chaotic neural network with nonlinear self-feedback and its application to clustering problem, Pattern recognition,34,pp.1093-1104(2001).

Gang Yang 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 neural network, pattern recognition and optimizations

Zheng Tang received the B.S. degree from Zhejiang University, Zhejiang, China in 1982 and an M.S. degree and a D.E. degree from Tshinghua 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 intellectual information technology, neural networks, and optimizations.

Junyan Yi received the B.S. degree and M.S. degree from Shandong University, Shandong, China in 2001 and 2005 respectively. She had worked in Huawei company for one year. Now she is working toward the Ph.D degree at University of Toyama, Japan. Her main research interests are neural network, brain-wave, pattern recognition and optimizations.