Maximum Neural Network with Nonlinear Self-Feedback and Its Application to Maximum Independent Set Problem

Jiahai Wang* Non-member
 Zheng Tang* Non-member
 Xinshun Xu* Member

In this paper, based on the maximum neural network, we propose a new parallel algorithm that can escape from local minima and has powerful ability of searching the globally optimal or near-optimum solution for the maximum independent set problem (MISP). Given a graph, the aim of the MISP is to find the largest set of vertices such that no two vertices in the set are connected by an edge. The MISP is a classic optimization problem in computer science and in graph theory with many real-world applications, and is also known to be NP-complete. By adding a nonlinear self-feedback to the maximum neural network, we proposed a parallel algorithm that introduces richer and more flexible nonlinear dynamics and can prevent the network from getting stuck at local minima. After the nonlinear dynamics has vanished, the proposed algorithm then is fundamentally reined by the gradient descent dynamics and usually converges to a stable equilibrium point. A large number of instances have been simulated to verify the proposed algorithm.

Keywords : maximum independent set problem, maximum neural network, nonlinear self-feedback, NP-complete problem

1. Introduction

Consider an undirected graph $G(V, E)$ where V is the set of vertices and E is the set of edges in G . An independent set is a set of vertices of G so that no two vertices of the set are adjacent; that is, no two vertices are joined by an edge of G . An independent set is said to be maximal if the set is not a proper subset of any other independent set. Furthermore, an independent set is said to be maximum if there is no independent set with a greater number of vertices. The maximum independent set problem (MISP) is to find the number of vertices in a maximum independent set for a given graph. The opposite concept of an independent set is a clique. A clique of G is a subset of V such that all pairs of vertices are connected by an edge in E . The maximum clique problem (MCP) is to find a clique of maximum size of a graph G . It is quite obvious that an independent set of a graph G corresponds to a clique of the graph $\bar{G}$ and vice versa, where $\bar{G}$ is the graph complementary to G .

The MISP is a very practical problem which can be applied to coding theory, geometry, VLSI design, molecular biology, scheduling, information retrieval fault detection and recovery. The problem of finding a maximum independent set for arbitrary graphs is NP-complete⁽¹⁾⁽²⁾ which means there is no polynomial time optimization algorithm to solve it to date. Generally, heuristics are applied to real problems and suboptimal solutions are accepted. Gravril developed an algorithm for finding a near maximum independent set in a circle graph⁽³⁾. Supowit proposed a algorithm in the circle graph⁽⁴⁾. Hsu gave an algorithm on the planar perfect graphs⁽⁵⁾. Choukhmane⁽⁶⁾ and Burns⁽⁷⁾ proposed an algorithm on cubic plannar graphs. Masuda gave an algorithm on circle graphs⁽⁸⁾. Takefuji et al. proposed parallel algorithms for finding a near-maximum independent set of a circle graph⁽⁹⁾.

Serveral parallel algorithms have been proposed by Karp⁽¹⁰⁾, Luby⁽¹¹⁾ and Goldberg⁽¹²⁾.

However, these specific algorithms just can get optimal solution for special graphs, for example chords in a circle, circular-arc graph and cubic planar graph. It is desirable that method can find optimal solution on general graph. There do exist graph optimization techniques, such as stochastic and simulated annealing, but their time complexity is too high. One possible and very promising approach to combinatorial optimization problem is to apply artificial neural network. In 1996, Lee and Takefuji⁽¹³⁾ proposed a different and very powerful parallel neural network for solving the MCP, which was based on a competitive Hopfield-type network called maximum neural network (MNN)⁽¹⁴⁾. It presented the following advantages in solving the MCP: (1) coefficient-parameter tuning was not needed while most the Hopfield neural networks suffer from it and (2) the equilibrium state was clearly defined in order to terminate the algorithm while the existing neural networks did not usually have the clear definitions. Moreover, this neural algorithm for large-size problems outperformed the best-known algorithms in terms of computation time with the same solution quality⁽¹³⁾. Unfortunately, the MNN that was based on the steepest descent method sometimes caused the excessive fixation of the state of the neural network and easily converged to the oscillation state of local minimum because of the limitation of its ability⁽¹⁵⁾⁽¹⁶⁾.

Yuan et al⁽¹⁷⁾. proposed an efficient technique to map the minimum vertex cover and two closely related problems, maximum independent set and maximum clique, onto the Hopfield neural networks, and a systematic way of deriving energy functions was described. They claimed that their algorithm could be used to find near-optimum solutions for these problems in parallel and the performance of their algorithms was better than that of the well-known sequential greedy algorithms. Sakamoto et al. also proposed a powerful genetic algorithm for MISP⁽¹⁸⁾.

* The Faculty of Engineering, Toyama University
 Toyama-shi 930-8555 Japan

While most of the conventional neural networks fundamentally utilize gradient descent dynamics, a number of artificial neural network with chaotic dynamics have been investigated towards more complex neurodynamics⁽¹⁹⁾⁻⁽²⁶⁾. Especially, Chen and Aihara⁽²²⁾ proposed a transiently chaotic neural network (TCNN) based on the continuous Hopfield network with a decay term. By adding a negative self-feedback to a “transiently chaotic neural network” (TCNN) and gradually removing this negative self-feedback, Chen and Aihara used the transient chaos for searching and self-organizing, thereby achieving remarkable improvement over other neural network approaches to optimization problems with or without simulated annealing⁽²³⁾, and they theoretically proved that the TCNN had global searching ability⁽²⁴⁾⁽²⁵⁾.

However, since this method⁽²²⁾ is based on the continuous Hopfield neural network (CHNN), satisfying constraints is as difficult as searching good solutions, so it often offers no-feasible solutions⁽²⁷⁾. A number of network parameters must be carefully chosen so as to guarantee the convergence of the TCNN and its minimization of the energy upon the removal of the transient chaos⁽²³⁾⁽²⁴⁾. The parameter of membrane term and the gain in the sigmoid function as the input/output function for neurons in the TCNN also must be carefully selected with respect to the synaptic weight matrix according to their stability theorems⁽²³⁾⁽²⁸⁾. Furthermore, the decay term included in the motion equation of TCNN increases the energy function under certain conditions⁽¹³⁾⁽¹⁴⁾, so it disturbs the convergence of the system.

In this paper, based on the MNN, we propose a new parallel algorithm that can help the MNN escape from local minima and has powerful ability of searching the globally optimal or near-optimum solution for the MISP. By adding a nonlinear self-feedback to the MNN, the proposed algorithm has richer and more flexible dynamics and can prevent the network from getting stuck at local minima. After the nonlinear 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 always guarantees the valid solution. The performance of the proposed algorithm is evaluated by simulating a number of instances.

2. MISP and Its Neural Representation

Let $G=(V, E)$ be an arbitrary undirected graph, where $V=\{1, \dots, n\}$ 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 $(i, j) \notin E$. An independent set is a subset of V , whose elements are pairwise nonadjacent. The MISP asks for an independset of maximum cardinality.

In general, the n -vertex MISP can be mapped onto the Hopfield neural network. An objective function can be formulated for this optimization problem whose minimum value corresponds to optimal solution. In a reasonable formulation there are two components to the objective function: goal term which is to realize the number of vertices in clique is maximum, and constrain term which is to satisfy that in clique every pair of vertices is connected by an edge. The MISP can be mathematically stated as finding the minimum of following Hopfield neural network energy function which is equivalent to minimizing a unconstrained quadratic zero-one programming problem⁽²⁹⁾:

$$e = A \sum_{i=1}^n \sum_{j=1}^n a_{ij} y_i y_j - B \sum_{i=1}^n y_i \dots\dots\dots (1)$$

where the output y_i is 1 if vertex i is in the independent set, 0 otherwise. A and B are constant coefficients. The A -term represents the constraint of the MISP. If only if all pair of vertices selected in an independent set are pairwise nonadjacent, the A -term has the minimum value 0. Otherwise, it has a positive value. The B -term represents the goal function of the problem. When the number of selected vertices in maximized, the B -term is minimized. The motion equation in the Hopfield neural network is given by⁽³⁰⁾:

$$\frac{dx_i}{dt} = -\frac{x_i}{\tau} - \frac{\partial e}{\partial y_i} \dots\dots\dots (2)$$

The equivalence between quadratic 0-1 optimization and the max-cut problem has been pointed out by Hammer⁽³¹⁾. Barahona et al⁽³²⁾, showed how an unconstrained quadratic 0-1 programming problem is equivalent to minimizing the weight summation over summation over the same partition in a newly constructed graph G_M with vertex set $V_M = \{0, 1, 2, \dots, n\}$. By applying these results, the n -vertex MISP can be mapped onto the MNN with $2 \times n$ neurons where it consists of n clusters of two neurons, and each neuron has an output v_{xi} and an input u_{xi} , respectively. The energy function of the MISP is defined by:

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

the weight matrix is defined by:

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

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

In this formulation, the first step in solving the MISP is to construct a graph G_M by adding a vertex 0 to G . Figure 1 shows a graph G with 10 vertices, 24 edges and the graph G_M . Two maximum independent set of the graph G are shown in Fig.2 by black point. The weight matrix of G_M is given by (4), where d_{ij} represents the weight of an edge between vertices i, j . The second step is to minimize the summation of the weights whose vertices belong to the same partition set. Lee et al⁽¹³⁾⁽¹⁴⁾, proved that the decay term $-x_i/\tau$ disturbs the convergence of the system because it increases the energy function under certain conditions. Therefore the motion equation of the i -th neuron in the x -th cluster without the decay term to minimize the energy function is given by:

$$\frac{du_{xi}}{dt} = -\frac{\partial e}{\partial v_{xi}} = -\sum_{y=0}^n d_{xy} v_{yi} \dots\dots\dots (5)$$

However, in practice, they are approximated by the first-order Euler method in the form:

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

Therefore, the input u_{xi} for each neuron of the MNN is updated using the following equation based on the first-order Euler method:

Fig. 1. (a) A graph G with 10 vertices and 24 edges, (b) The graph G_M of (a).

$$u_{xi}(t+1) = u_{xi}(t) + \Delta u_{xi}(t), \dots\dots\dots (7)$$

until the network reaches an equilibrium state, and the input/output function of the i -th neuron in the x -th cluster is given by:

$$v_{xi} = 1 \text{ if } u_{xi} = \max_{j=1,2} \{u_{xj}\}; 0 \text{ otherwise.} \dots\dots\dots (8)$$

If there is more than one neuron with the maximum input in any cluster, the neuron with the smallest subscript has nonzero output. The outputs of the other neuron in the same cluster become zero. The binary output $v_{xi}=1$ means the x vertex belongs to the independent set and $v_{xi}=0$ means the x vertex is not included in the independent set. Note, the 0 vertex is always assigned to independent set, that is, we must impose $v_{01}=1$. The total number of vertices in the final solution is given by $\sum_{x=1}^n v_{x1}$. Since the outputs of all neurons are all binary (0 or 1), the MISP is considered as a 0-1 combinatorial optimization problem whose energy function is defined by Eq.(3). The system can minimize the energy function Eq.(3) and reach stable state of the energy function finally with time evolution according to Eq.(7) and Eq.(8). This convergence property of the MNN is proved and given in

Fig. 2. Two maximum independent set of the graph G depicted in Fig.1 (a).

Ref.(13)(14). The advantage of this formulation of the MISP applied to the MNN is the constraint always automatically satisfied. In this way, the MISP reduces to obtain a minimum of the energy function without the parameters affecting the global minimum search.

3. Description of the Proposed Algorithm

The maximum neural model always guarantees the valid solution and greatly reduces the search space without a burden on the parameter-tuning. However, the model has a tendency to converge to local minimum easily because it is based on the steepest descent method. Here we propose a new parallel algorithm by adding a negative self-feedback to the MNN, as defined below:

$$u_{xi}(t+1) = u_{xi}(t) - \alpha \Delta u_{xi}(t) - z(t)(v_{xi}(t) - \frac{1}{2}) \dots\dots\dots (9)$$

$$z(t+1) = (1 - \beta)z(t) \dots\dots\dots (10)$$

where α is a positive scaling parameter for inputs, and $z(t)$ is a self-feedback connection weight ($z(t) \geq 0$). The same $z(t)$ value is used for all neurons at t during the updating process. β is a

damping factor of the time-dependent $z(t)$ ($0 \leq \beta \leq 1$).

The form of Eq.(9) is similar to the mathematical statement which was used in reference⁽²²⁾, but Eq.(9) uses the maximum neuron model and has no decay term. Chen and Aihara⁽²²⁾ analyzed the system dynamics by orbital instability quantitatively of the TCNN with indexes of Lyapunov exponents which were commonly used to analyze chaotic systems. Adachi and Aihara⁽¹⁹⁾ pointed out that it was impossible to analyse them if the output of each neuron in the network like the proposed algorithm had alternative discrete values of 0 or 1. Thus, we analyze the proposed algorithm as following, which indicates how the proposed system evolves and can escapes from local minima. In the proposed algorithm, the updating of input u_{xi} for each neuron is related to the output v_{xi} of itself. The self-feedback term, $-z(t)(v_{xi}(t) - \frac{1}{2})$, becomes negative after the xi -neuron firing at time t (that is, $v_{xi}(t) = 1$). The negative self-feedback inhibits the growth of the input value of neuron at next time, $t+1$. On the other hand, the self-feedback term becomes positive if the xi -neuron is unfired at time t (that is, $v_{xi}(t) = 0$). The positive self-feedback promotes the growth of the input value of neuron at next time, $t+1$. Thus, we can say that the proposed algorithm includes refractory effect, which is one of the characteristics of real biological neurons⁽²⁷⁾. Each neuron with refractory effect becomes difficult to fire after previous firings. In the maximum neuron model as described by Eq.(8), the input values of two neurons in the same cluster are compared with each other to extract the maximum one. In other words, neurons in the same cluster compete with each other to be fired. Since the self-feedback can inhibit or promote the growth of the input value of neurons in the way mentioned above, it encourages this competition and easily leads to the transition of output state of all neurons. This mechanism contributes to the blocking of continuous firings and encourages other neurons to be eventually fired. Therefore the network can easily escape from a fixed point or a local minimum by changing the output state of neurons in the MNN.

In the motion equation (9), the α term means decreasing of the objective function value, which yields a better solution that is closer to a local minimum. The self-feedback term induces a jump from a searching region to other regions in order to avoid getting trapped in a single local minimum; thus the system can generate a new solution. The parameter α represents the influence of the energy function on the neurodynamics, or the balance between the self-feedback term inducing the mechanism of escaping from local minima and the gradient term Δu_{xi} inducing the convergent dynamics. The parameter β can be considered as a damping speed of the nonlinear self-feedback connection $z(t)$.

Given randomly generated numbers for the initial state of $u_{xi}(0)$ and a sufficiently large initial self-feedback $z(0)$, the competition or the state transition of neurons is drastic when iterating Eq.(8) and Eq.(9), which breaks the monotonic gradient descent dynamic and can help the network to escape from local minima. When the self-feedback connection weight $z(t)$ tends toward zero with time evolution according to Eq.(10), the network eventually reduces to the MNN without the self-feedback connection, which is always allowed to converge to a valid and good solution.

The proposed algorithm can be terminated in the equilibrium

state where no neuron changes the output⁽¹³⁾⁽¹⁴⁾.

4. Simulation Results

In order to assess the effectiveness of the proposed algorithm, extensive simulations were implemented in C++ on AOPEN-PC (PentiumIII 866 MHz) to a number of randomly generated graphs of various sizes. The parameter values used in the simulations were

$$\alpha = 0.015, \quad z(0) = 0.11, \quad \text{and} \quad \beta = 0.003.$$

All the graphs are randomly generated, thus, any specific algorithm based on special subclass graph characteristics may fail. Yuan et al. in⁽¹⁷⁾ proposed three methods, FGDN1, FGDN2 and CHA, for MISP which can be applied to any kind of graphs. Experimental results showed that the performance of their algorithms were better than that of the well-known sequential greedy algorithms. But they only evaluated their algorithms on small size graphs (the largest graph had only 50 vertices). We found that FGDN2 and CHA could not find the maximal independent set when the graph had more than 200 vertices. In FGDN2, when the number of vertex of the graph was larger than 200, the coefficient γ became so small that the solution was bad. Thus, we used the FGDN1 and MNN (which can be applied to any kind of graphs just as our algorithm does) for comparison. All of the algorithms were run 25 times. The best results and the average results with average computation time are recorded to evaluate the algorithms.

At first, we tested the proposed algorithm on 24 random graphs defined in terms of two parameters, n and p . The parameter n specified the number of vertices in the graph; the parameter p , $0 < p < 1$, specified the probability that any given pair of vertices constituted an edge. The size of graphs was from 200 vertices to 600 vertices and up to 44925 edges. Table 1 indicates the information of the 24 random graphs. Table 2 is the solutions produced by the MNN, FGDN1 and the proposed algorithm. The results of Table 2 show that the proposed algorithm has the most powerful performance for the MISP in term of best results and average results. Table 3 shows the comparison of average computation time. We denote the number of iteration as the number of updating all the neuron output values. The MNN converged to solution during 500 iterations, while the proposed

Table 1. 24 simulated graph instances.

Problem	Vertices	Probability	Edges
1	200	0.05	995
2	200	0.15	2985
3	200	0.25	4975
4	250	0.05	1556
5	250	0.15	4669
6	250	0.25	7781
7	300	0.05	2243
8	300	0.15	6728
9	300	0.25	11213
10	350	0.05	3054
11	350	0.15	9161
12	350	0.25	15269
13	400	0.05	3990
14	400	0.15	11970
15	400	0.25	19950
16	450	0.05	5051
17	450	0.15	15154
18	450	0.25	25256
19	500	0.05	6238
20	500	0.15	18713
21	500	0.25	31187
22	600	0.05	8985
23	600	0.15	26955
24	600	0.25	44925

algorithm converged to solution during 1200 iterations. The FGDNI algorithm converged to solution during 3000 iterations because it used very small time step in the motion equation and used sigmoid function as input/output function of the neurons. The computation time of the proposed algorithm was longer than that of the MNN because the proposed algorithm spent many time in annealing the nonlinear feedback to escape from local minima. The computation time of the proposed algorithm is almost the same as that of the FGDNI algorithm.

In order to evaluate not just the relative, but also the absolute performance equality, we also evaluated experimentally the performance of the proposed algorithm on the DIMACS benchmark graphs⁽³⁴⁾. The powerful genetic algorithm proposed by Sakamoto et al⁽¹⁸⁾, was also executed on the benchmark graphs for comparison. The DIMACS benchmark graphs were originally considered for the MCP. These graphs provided a valuable source for testing the performance of algorithms for the MCP, because they arised from different areas of application. For example the

Hamming graphs were from coding theory problem⁽³⁵⁾, the Keller graphs were based on Keller's conjecture on tilings using hypercubes⁽³⁶⁾. We tested the proposed algorithm on 17 graphs in size from 28 vertices to 1024 vertices and up to 518656 edges. Table 4 is the experimental results for DIMACS benchmark graphs. The first three columns indicate the graph data and graph size (including the number of vertices and edges). The other columns indicate global optimality, the average and best solutions found by the proposed algorithm and the genetic algorithm, respectively. The average computation times of the algorithms are also given in Table 4. Note that the solutions and average computation times of the genetic algorithm is directly quoted from Ref. (18) where the genetic algorithm was coded in C and run on a workstation Sun SPARC LX station. From Table 4, we can conclude that the proposed algorithm is comparable withthe genetic algorithm. Furthermore, the results of the proposed algorithm were the same as the global optimality found by all 15 heuristic algorithms presented at the second DIMACS Challenge⁽³⁴⁾. All 15 DIMACS heuristic algorithms were very

Table 2. Solutions produced by the MNN, FGDNI and the proposed algorithm.

Problem	MNN		FGDNI		Proposed algorithm	
	Best	Av.	Best	Av.	Best	Av.
1	55	50.96	61	57.71	61	61
2	24	21.84	24	22.31	30	30
3	18	14.84	18	14.67	20	19
4	59	52.36	59	51.26	67	66.12
5	28	23.92	27	24.78	32	30.12
6	19	14.96	17	14.71	22	20
7	57	54.20	65	58.73	71	71
8	27	24.06	31	27.21	35	34.1
9	18	14.84	18	15.11	22	21.96
10	65	58.28	69	63.21	76	75.4
11	27	24.92	29	27.13	37	35.84
12	18	16.24	19	17.12	23	22.1
13	65	60.84	70	64.51	81	80.4
14	29	25.68	33	29.34	37	36.6
15	20	16.52	20	16.61	25	25
16	68	62.56	78	75.78	81	81
17	29	26.64	31	30.11	38	37.6
18	20	17.12	20	18.04	25	23.04
19	68	62.74	77	75.78	88	86.28
20	31	27.8	32	29.31	40	39.12
21	21	17.52	20	18.17	27	24.8
22	76	69.64	86	84.21	95	92.96
23	32	28.36	34	31.46	41	40
24	22	17.88	22	21.11	26	24.24

Table 3. Computation time of the algorithm of the MNN, FGDNI and the proposed algorithm (seconds).

Problem	MNN	FGDNI	Proposed algorithm
1	1.10	3.23	3.08
2	1.15	3.25	3.11
3	1.25	3.34	3.13
4	2.35	6.38	5.34
5	2.36	6.54	5.36
6	2.40	6.57	5.78
7	3.12	7.32	7.12
8	3.21	7.63	7.21
9	3.25	7.65	7.32
10	4.32	11.23	10.21
11	4.56	11.32	10.32
12	4.68	11.58	10.46
13	5.95	13.98	13.21
14	5.98	13.99	13.34
15	6.02	14.21	13.41
16	6.87	16.35	15.56
17	6.88	16.57	15.68
18	6.91	16.68	15.70
19	8.13	19.65	18.56
20	8.32	19.68	18.61
21	8.56	19.85	18.66
22	10.23	26.32	25.61
23	10.32	26.56	25.63
24	10.53	26.89	25.70

Table 4. Experimental results for DIMACS benchmark graphs.

Data	Vertices	Edges	Optimum	Genetic algorithm			Proposed algorithm		
				Best	Av.	Time	Best	Av.	Time
c-fat200-1	200	1534	12	12	12	12.3	12	12	3.23
c-fat200-2	200	3235	24	24	24	16.4	24	24	3.26
c-fat200-5	200	8473	58	58	58	21.1	58	58	3.30
c-fat500-1	500	4459	14	14	14	60.7	14	14	18.30
c-fat500-2	500	9139	26	26	26	72.9	26	26	18.42
c-fat500-5	500	23191	64	64	64	97.7	64	64	18.51
c-fat500-10	500	46627	126	126	126	125.8	126	126	18.55
johnson8-2-4	28	210	4	4	4	0.7	4	4	0.01
johnson8-4-4	70	1855	14	14	14	2.0	14	14	0.02
johnson16-2-4	120	5460	8	8	8	4.5	8	8	0.51
johnson32-2-4	496	107880	16	16	16	63.2	16	16	16.35
keller4	171	9435	11	11	11	9.1	11	11	2.96
hamming6-2	64	1824	32	32	32	1.4	32	32	0.01
hamming6-4	64	704	4	4	4	1.0	4	4	0.01
hamming8-2	256	31616	128	128	128	21.2	128	128	6.89
hamming8-4	256	20864	16	16	16	19.9	16	16	6.78
hamming10-2	1024	518656	512	512	512	351.3	512	512	41.23

efficient in the second DIMACS challenge although it is difficult to judge which was the best⁽³⁴⁾. The proposed method can find the global optimality with 100 percent rate in the reasonable computation time for each benchmark graph in Table 4, which indicates that the proposed algorithm is also comparable with other heuristic or stochastic methods. Unlike the other problem-specific heuristic algorithms, the proposed algorithm is based on the general MNN, therefore it can be easily applied to a lot of other combinational optimization problems, for example, bipartite subgraph problem and module orientation problem⁽¹³⁾⁽¹⁴⁾. Further, the proposed algorithm as a neural network algorithm is essentially a parallel method, and it therefore naturally leads itself to be implemented in parallel hardware. Rapid developments in hardware implementations (both VLSI and FPGA) will soon enable the full exploitation of this potentially powerful technology.

In order to show how the proposed algorithm works, we choose

Fig. 3. The evolution of energy function of the graph with 500 vertices and 6238 edges in MNN algorithm.

Fig. 4. The evolution of energy function and self-feedback connection weight z of the graph with 500 vertices and 6238 edges in the proposed algorithm.

a graph with 500 vertices and 6238 edges from Table 1 as an example. Figure 3 shows the evolution of energy function of the graph in the MNN algorithm. It indicates the MNN is a steepest descent method, and therefore has a tendency to converge to the local minima. Figure 4 shows the evolution of the energy function and self-feedback connection weight z in the proposed algorithm. At the early stage, the energy function of the proposed algorithm changed with larger fluctuations. At this time, the competition or the state transition of neurons is drastic. This mechanism helps the network to escape from a fixed point or a local minimum. With the reduction of the self-feedback connection weight z , the fluctuations gradually become smaller. After about 900 iterations, when z becomes so small that the convergent characteristic dominates the dynamics, finally the neural network finally converges to a stable state or an equilibrium state corresponding to a global or near-global minimum. At the same time, a trend of gradually decreasing energy from the whole course can be observed until the network stabilizes at the globally optimal or near-optimum solution. When the algorithm procedures terminated, the minimum of the energy in the MNN and the proposed algorithm were 3087.5 and 3075, respectively. The corresponding solutions were 68 and 88, respectively.

5. Conclusions

By adding a nonlinear self-feedback to the MNN, we proposed a parallel algorithm that had richer and more flexible nonlinear dynamics and could prevent the network from getting stuck at local minima. After the nonlinear dynamics have vanished, the proposed algorithm then was fundamentally reined by the gradient descent dynamics and usually converged to a stable equilibrium point like the MNN. The proposed algorithm always guaranteed the valid solution and greatly reduced the search space. At the same time, it could avoid being trapped in local minima of the MNN. The proposed algorithm was tested on two types of random graphs and DIMACS benchmark graphs. The simulation results showed that the proposed algorithm had powerful ability to find a solution of the MISP.

(Manuscript received Sep. 19, 2003, revised Aug. 11, 2004)

References

- (1) R.M. Karp : "Reducibility among combinatorial problems", Complexity of Computer Computations, Plenum, New York, pp.85-104 (1972)
- (2) M.R. Garey and D.S.Johon : "Computers and intractability: A guide to the theory of NP-completeness", San Francisco, CA: Free-man (1979)
- (3) F. Gavril : "A algorithm on circular-arc graphs", Networks, Vol.4, pp.357-367 (1974)
- (4) K.J. Supowit : "Finding a maximum planar subset of a set of nets in a channel", IEEE Trans. Computer-Aided Design, Vol.6, pp.93-94 (Jan. 1987)
- (5) W.L. Hsu : "The coloring and maximum independent set problems on planar perfect graphs", J.ACM, Vol.35, No.3, pp.535-563 (July 1988)
- (6) E. Choukhmane and J. Franco : "An approximation algorithm for the maximum independent set problem in cubic planar graphs", Networks, Vol.16 (1986)
- (7) J.E. Burns : "The maximum independent set problem for cubic planar graph", Networks, Vol.19 (1989)
- (8) S. Masuda and K. Nakajima : "An optimal algorithm for finding a maximum independent set of a circular-arc graph", SIAM J. Comput., Vol.17, No.1, pp.41-52 (Feb. 1988)
- (9) Y. Takefuji, L. Chen, K. Lee, and J. Huffman : "Parallel algorithms for finding a near-maximum independent set of a circle graph", IEEE Trans. Neural Networks, Vol.1, No.3, pp.263-267 (Sep. 1990)
- (10) R.M. Karp and A. Wigderson : "A fast parallel algorithm for the maximum independent set problem", J.ACM, Vol.32, No.4, pp.266-272 (Oct. 1985)
- (11) M. Luby : "A simple parallel algorithm for the maximal independent set

- problem", *SIAM J. Comput.*, Vol.15, No.4, pp.1-10 (Nov. 1986)
- (12) M. Goldberg and T. Spencer : "A new parallel algorithm for the maximal independent set problem", *SIAM J. Comput.*, Vol.18, No.2, pp.419-427 (Apr. 1989)
 - (13) Y. Takefuji and J. Wang : "Neural computing for optimization and combinatorics. Singapore", World Scientific, Chapter 3 (1996)
 - (14) 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.*, Vol.67, pp.243-251 (1992)
 - (15) 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*, Vol.E83-A, No.9, pp.1815-1823 (2000)
 - (16) S. Oka, T. Ogawa, T. Oda, and Y. Takefuji : "A new self-organization classification algorithm for remote-sensing image", *IEICE Trans. Inf.&Syst.*, Vol.E81-D, No.1, pp.132-135 (1998)
 - (17) S.-Y. Yuan and S.-Y. Kuo : "A new technique for optimization problems in graph theory", *IEEE Trans. Computers*, Vol.47, No.2, pp.190-196 (1998)
 - (18) A. Sakamoto, X. Liu, and T. Shimamoto : "A genetic algorithm for maximum independent set problems", *IEICE Trans. Fundamentals*, Vol.E80-A, No.3, pp.551-556 (March 1997)
 - (19) K. Aihara, T. Takabe, and M. Toyoda : "Chaotic neural networks", *Phys. Lett. A*, Vol.144, pp.333-340 (1990)
 - (20) M. Inoue and A. Nagayoshi : "A chaos neuro-computer", *Phys. Lett. A*, Vol.158, pp.373-376 (1991)
 - (21) T. Yamada, K. Aihara, and M. Kotani : "Chaotic neural networks and the traveling salesman problem", *Proc. International Joint Conference on Neural Network*, Vol.2, pp.1549-1552 (1993)
 - (22) L. Chen and K. Aihara : "Chaotic simulated annealing by a neural network model with transient chaos, *Neural Networks*", Vol.8, No.6, pp.915-930 (1995)
 - (23) L. Wang and K. Smith : "On chaotic simulated annealing", *IEEE Transact. on Neural Networks*, Vol.9, No.4, pp.716-718 (1998)
 - (24) L. Chen and K. Aihara : "Global searching ability of chaotic neural networks", *IEEE Trans. Circuits Syst.*, Vol.46, No. 8, pp.974-993 (1999)
 - (25) L. Chen and K. Aihara : "Chaos and asymptotical stability in discrete-time neural networks", *Physica*, Vol.104, pp.286-325 (June, 1997)
 - (26) M. Hasegawa, T. Ikeguchi, and K. Aihara : "Combination of chaotic neurodynamics with the 2-opt algorithm to solve traveling salesman problems", *Physical Review Letters*, Vol.79, No.12, pp.2344-2347 (1997)
 - (27) M. Hasegawa, T. Ikeguchi, K. Aihara, and K. Itoh : "A novel chaotic search for quadratic assignment problems", *European Journal of Operational Research*, Vol.139, pp.543-556 (2002)
 - (28) F. Tian and L. Wang : "Solving optimization problems using transiently chaotic neural networks: choices of network parameters", *Proc. Second International Conference on Information, Communications & Signal Processing*, Singapore, pp.7-10 (1999)
 - (29) I.M. Bomze, M. Budinich, P.M. Pardalos, and M. Pelillo : "The maximum clique problem" in: D.Z.Du and P.M.Pardalos (ed.), *Handbook of Combinatorial Optimization*, Kluwer Acad. Publisher, Dordrecht, pp.1-74 (1999)
 - (30) J.J. Hopfield and D.W. Tank : "'neural' computation of decisions in optimization problems", *Bio.Cybern.*, Vol.52, pp.142-152 (1985)
 - (31) P.L. Hammer : "Some network flow problems solved with pseudo-Boolean programming", *Operations Research*, Vol.13, pp.388-399 (1965)
 - (32) F. Barahona, M. Junger, and G. Reinelt : "Experiments in quadratic 0-1 programming", *Mathematical Programming*, Vol.44, pp.127-137 (1986)

- (33) T. Kwok and K.A. Smith : "Experimental analysis of chaotic neural network models for combinatorial optimization under a unifying framework", *Neural Networks*, Vol.13, pp.731-744 (2003)
- (34) D. Johnson and M. Trick : "Cliques, coloring, and satisfiability", *AMS, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, Vol.26 (1996)
- (35) N.J.A. Sloane : "Unsolved problem in graph theory arising from the study of codes", *Graph Theory Notes of New York XVIII* (1989)
- (36) J.C. Lagarias and P.W. Shor : "Keller's cube-tiling conjecture is false in high dimensions", *Bulletin AMS*, Vol.27, No.2, pp.279-283 (1992)

Jiahai Wang

(Non-member) received a B.S. degree from Gannan Teachers College, Jiangxi, China and an M.S. degree from Shandong University, Shandong, China in 1999 and 2001, respectively. Now he is working toward the Ph.D degree at Toyama University, Toyama, 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 and Tsinghua 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.

Xinshun Xu

(Member) received a B.S. degree from Shandong Normal University, Shandong, China and an M.S. degree from Shandong University, Shandong, China in 1998 and 2002, respectively. From 1998 to 2000, she was an Engineer in Shandong Provincial Education Department, Shandong, China. Now he is working toward the Ph.D degree at Toyama University, Toyama, Japan. His main research interests are neural networks, pattern recognition, and optimization problems, data mining and biometrics.