International Conference on Control, Automation and Systems 2008
Oct. 14-17, 2008 in COEX, Seoul, Korea

A Reinforcement Learning System with Chaotic Neural Networks-Based Adaptive Hierarchical Memory Structure for Autonomous Robots

Masanao Obayashi[1], Kenichiro Narita[1], Takashi Kuremoto[1] and Kunikazu Kobayashi[1]

[1] Division of Computer Science & Design Engineering, Yamaguchi University, Ube, Japan
(Tel : +81-836-85-9518; E-mail:{ m.obayas,wu,koba}@yamaguchi-u.ac.jp)

Abstract: Human learns incidents by own actions and reflects them on the subsequent action as own experiences. These experiences are memorized in his brain and recollected if necessary. This research incorporates such an intelligent information processing mechanism, and applies it to an autonomous agent that has three main functions: learning, memorization and associative recollection. In the proposed system, an actor-critic type reinforcement learning method is used for learning. Auto-associative chaotic neural network is also used like mutual associative memory system. Moreover, the memory part has an adaptive hierarchical layered structure of the memory module that consists of chaotic neural networks in consideration of the adjustment to non-MDP (Markov Decision Process) environment. Finally, the effectiveness of this proposed method is verified through the simulation applied to the maze-searching problem.
Keywords: Reinforcement learning ,chaotic neural network, hierarchical memory structure, autonomous robot

1. INTRODUCTION

Reinforcement learning (R.L.) is a framework for an agent to learn the choice of an optimal action based on a reinforcement signal [1]. It has been applied to a variety of problem such as autonomous robot navigation and non-linear control and so on. However, so far, so many systems with R.L. have been made up for only use of the one task. Research for systems making use of memorizing the results of learning of many tasks and applying them to other task without learning has been little done. In this study, we use the associative chaotic neural network (ACNN) proposed by Aihara et.al [2] as a storage mechanism of results of R.L. However, the storage capacity of ACNN is small, it is not suitable for working alone. So, to resolve the problem, we made up the hierarchical memory structure by making use of ACNN: short-term memory for present learning result, long-term memory for many useful learning results. Another characteristic of the proposed system is that it is capable of dealing with non-MDP problem in some degree, because of the chaotic association ability of ACNN. Finally it is verified that our proposed method is useful through the computer simulation for a maze searching problem.

2. PROPOSED SYSTEM STRUCTURE

The proposed system consists of two parts: memory and learning. The memory consists of short-term memory (S.T.M.) and long-term memory (L.T.M.). Fig.1 shows its overall structure.

Learning sector : actor-critic system is adopted. It learns the choice of action to maximize the total predictive rewards obtained over the future considering the environmental information (s) and reward (r) as result of action (a).

S.T.M. sector: it memorizes the learning path of the information (environmental information and action) obtained in Learning part. Unnecessary information is forgotten and useful information is stored.

L.T.M. sector: it memorizes only the enough sophisticated and useful experience in S.T.M..

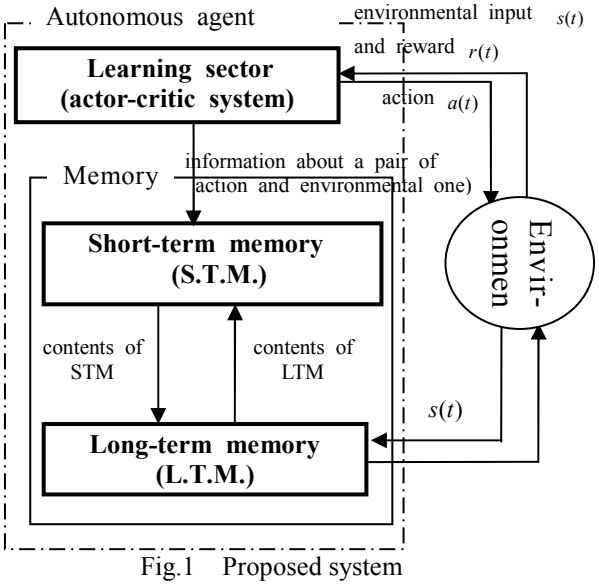


Fig.1 Proposed system

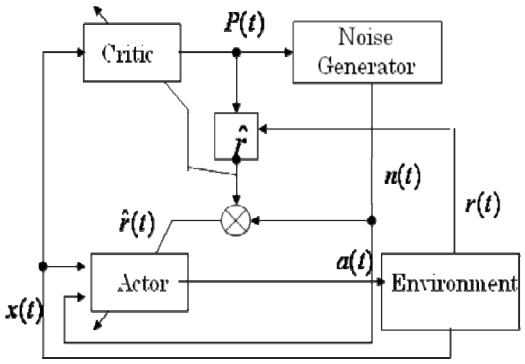


Fig. 2 The construction of actor-critic system

3. ACTOR-CRITIC REINFORCEMENT LEARNING SYSTEM

The actor-critic reinforcement learning system is shown in Fig.2.

978-89-93215-01-4-98560/08/$15

3.1. Structure and learning of critic

3.1.1 Structure

Function of the critic is calculation of P(t) : the prediction value of sum of the discounted rewards that will be gotten over the future and its prediction error. These are shortly explained as follows;

The sum of the discounted rewards that will be gotten over the future is defined as $V(t)$.

$$V(t) \equiv \sum_{n=0}^{\infty} \gamma^n \cdot r(t+n), \qquad (1)$$

where γ ($0 \le \gamma < 1$) is constant called discount rate.
Eq. (1) is rewritten as

$$V(t) = r(t) + \gamma V(t+1). \qquad (2)$$

Here the prediction value of $V(t)$ is defined as $P(t)$.

The prediction error $\hat{r}(t)$ is expressed as follows;

$$\hat{r}(t) = r(t) + \gamma P(t+1) - P(t). \qquad (3)$$

The parameters of the critic are adjusted to reduce this prediction error $\hat{r}(t)$. The prediction value $P(t)$ is calculated as follows;

$$P(t) = \sum_{j=0}^{J} \omega_j y_j(t) \qquad (4)$$

$$y_j(t) = \exp\left[-\sum_{i=1}^{n} (x_i - m_{ij})^2 \Big/ \sigma_{ij}^2 \right]. \qquad (5)$$

Here, ω_j : weight of the jth output, y_j : jth output of the middle layer of the critic, x_i : ith input, m_{ij} , σ_{ij}^2 : center, dispersion for ith input of jth basis function respectively, J : the number of nodes in the middle layer of the critic,. The construction of the critic is also consisted of the RBFN as shown in Fig.3.

3.1.2 Learning

Learning of critic is done by using commonly used Back Propagation method which makes prediction error $\hat{r}(t)$ goes to zero. Updating rule of parameters are as follows:

$$\Delta\omega_i^c = -\eta_c \cdot \frac{\partial \hat{r}_t^2}{\partial \omega_i^c}, \quad (i = 1, \cdots, J). \qquad (6)$$

3.2. Structure and learning of actor

3.2.1 Structure

Fig.4 shows the construction of the actor. The actor is basically consisted of Radial Basis Function Network. The jth basis function of the middle layer node is as follows;

$$y_j = \exp\left[-\sum_{i=1}^{n} (x_i - m_{ij})^2 \Big/ \sigma_{ij}^2 \right], \qquad (7)$$

$$u_k(t) = \sum_{j=1}^{J} \omega_{kj} y_j(t) + n(t), \quad (k = 1, \cdots, K). \qquad (8)$$

Here y_j : jth output of the middle layer of the actor, m_{ij} , σ_{ij}^2 : center, dispersion for ith input of jth basis function respectively, K: the number of the actions, n : additive noise, u_k : representative value of kth action, ω_{kj} : connection weight from jth node of the middle layer to kth output.

3.2.2 Noise generator

Noise generator let the output of the actor have the diversity by adding the noise to it. It comes to realize the learning of the trial and error. Calculation of the noise $n(t)$ is as follows;

$$n(t) = n_t = noise_t \cdot \min(1, \exp(-P(t))), \qquad (9)$$

where $noise_t$ is uniformly random number of $[-1, 1]$. As the $P(t)$ will be bigger, the noise will be smaller. This leads to the stable learning of the actor.

3.2.3 Learning

Parameters of actor, ω_{kj}^a $(j = 1, \cdots, J, k = 1, \cdots, K)$, are adjusted by using output u_1 of actor and noise n.

$$\Delta\omega_{kj}^a = \eta_a \cdot n_t \cdot \hat{r}_t \cdot \frac{\partial u_k}{\partial \omega_{kj}^a}, \qquad (10)$$

$\eta_a (> 0)$ is the learning coefficient. Eq. (10) means that $(-n_t \cdot \delta_t)$ is considered as error, ω_i^a is adjusted opposite to sign of $(-n_t \cdot \delta_t)$.

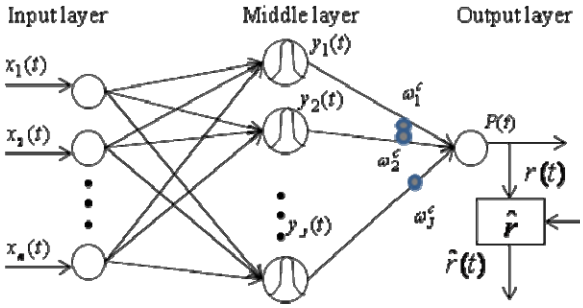


Fig.3 Structure of critic

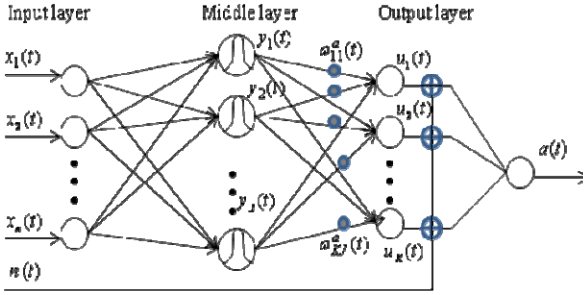


Fig.4 Structure of actor

3.3 Action selection

The action a_b at time t is selected stochastically using Gibbs distribution Eq.(11).

$$P(a_b \mid \mathbf{x}(t)) = \frac{\exp(u_b(t)/T)}{\sum_{k=1}^{K} \exp(u_k(t)/T)} \quad . (11)$$

Here, $P(a_b \mid \mathbf{x}(t))$: selection probability of bth action, a_b , T : positive constant called temperature constant.

4. A HIERARCHICAL MEMORY SYSTEM

4.1 Associative Chaotic Neural Network (ACNN)

CNN is constructed with chaotic neuron models that have refractory and continuous output value. Its useful usage is as associative memory network named ACNN. Here are the dynamics of ACNN.

$$x_i(t+1) = f(y_i(t+1) + z_i(t+1)), \quad (12)$$

$$y_i(t+1) = k_r \cdot y(t) - \alpha \cdot x_i(t) + a_i, \quad (13)$$

$$z_i(t+1) = k_f \cdot z_i(t) + \sum_{j=1}^{n} \varpi_{ij} x_j(t), \quad (14)$$

$$\varpi_{ij} = \frac{1}{P} \sum_{p=1}^{P} (2x_i^p - 1) \cdot (2x_j^p - 1) \quad , \quad (15)$$

$x_i(t)$: output of the ith neuron at time t, $y_i(t)$: internal state respect to refractory of the ith neuron at time t, $z_i(t)$: internal state respect to mutual operation of the ith neuron at time t., $f(\cdot)$: sigmoid function, ϖ_{ij}: connection weight from jth neuron to ith neuron, x_i^p: ith element of pth stored pattern.

4.2 Network control

Here, network control is defined as control which makes transition of network from chaotic state to non-chaotic one and vice versa. The network control algorithm of ACNN is shown in Fig.5. The state of ACNN is calculated by $\Delta x(t)$, total change of internal state $x(t)$ temporally, and when $\Delta x(t)$ is less than a threshold value θ, the chaotic retrieval of ACNN is stopped by changing values of parameters k_r into small one. As a result, network converges to a stored pattern near the present network state.

4.3 Mutual associative type ACNN

4.3.1 Short term memory(S.T.M.)

We make use of ACNN as a mutual associative memory system, namely, auto-associative matrix is constructed with environmental inputs $s(t)$ and their corresponding actions $a(t)$. When $s(t)$ is set as initial state of ACNN, ACNN retrieves $a(t)$ with $s(t)$ (refer to Fig.6). l is a random vector to weaken the correlation between $s(t)$ and $a(t)$. The memory matrix W_S is described as Eq.(16), here, λ_S is a forgetting coefficient, and η_S is a learning coefficient. λ_S is set to small ,because that at initial learning stage $s(t)$ is not corresponding to optimal $a(t)$.

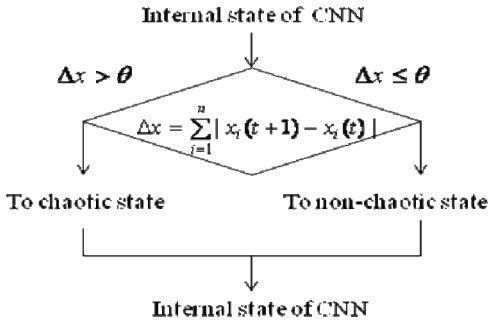
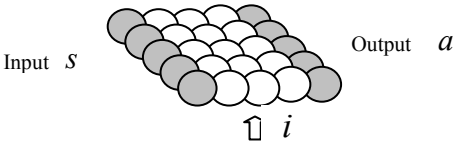


Fig.5 Network control algorithm



l : additional random memory units for weakening correlations between s and a

Fig.6 Memory configuration of ACNN

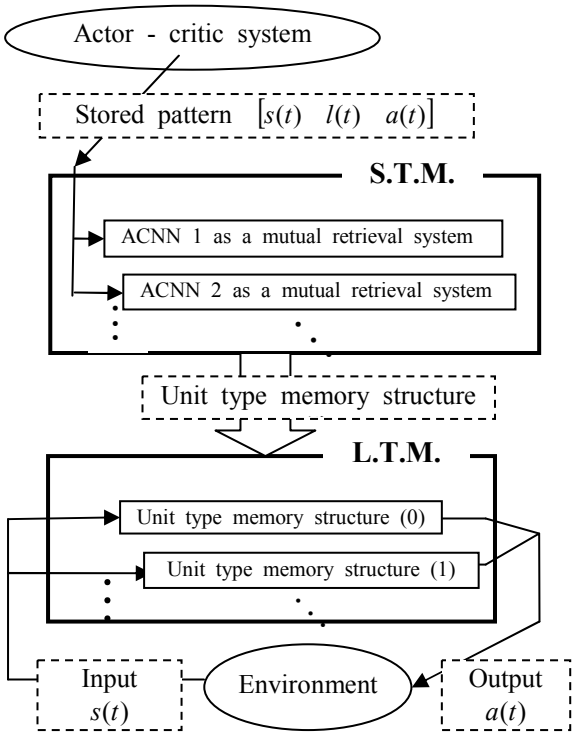


Fig.7 Adaptive hierarchical memory structure

$$W_S^{new} = \lambda_s \cdot W_S^{old} + \eta_S [s \quad l \quad a]^T [s \quad l \quad a] \quad . \quad (16)$$

S.T.M. as one unit consists of plural ACNNs, and one ACNN memorizes information for one environmental input pattern (refer to Fig.7). And S.T.M has path information from start to goal of only one maze searching problem.

4.3.2 Long term memory(L.T.M.)

The L.T.M. consists of plural units. The L.T.M. memorizes enough refined information in the S.T.M. as one unit (refer to Fig.7). Namely, when actor-critic learning has accomplished for a certain maze problem, information in the L.T.M. is updated as follows:
In case that the present maze problem has not been experienced, the stored matrix W_L is set by Eq.(17) ;

$$W_L = W_S . \tag{17}$$

In case that the present maze has been experienced and present learning is additive learning, the stored matrix is updated by Eq.(18) ;

$$W_L^{new} = \lambda_L \cdot W_L^{old} + \eta_L W_S . \tag{18}$$

λ_L is a forgetting coefficient, and η_L is a learning coefficient. λ_L is set to large value as same as one of η_L so as not to forget previous stored patterns.

4.4 Adaptive hierarchical memory structure

Fig.7 shows whole configuration of an adaptive hierarchical memory structure. When an environmental state is inputted to agent, at first it is sent to the L.T.M for confirming if it is the stored information or not. If it is the stored information, the obtained action corresponding to it is executed, otherwise, it is used to learn the actor-critic system. The pair of the enough refined and trained environmental state s and action a in the S.T.M. is sent to the L.T.M. to be stored. If it is similar to the stored pattern, information of the L.T.M. is used to relearn at the actor-critic system in the S.T.M..

5. COMPUTER SIMULATION

5.1 Simulation condition

Agent can perceive whether there is aisle or not at the forward, right-forward, left-forward, right, left as environment s (refer to Fig.8). Agent can move 1 lattice to forward, back, left, and right as action a (refer to Table 1). Therefore in actor-critic, state s of environment consists of 5 inputs (= n). And kinds of action a is 4(= K). The number of hidden nodes of R.B.F. is equal to 32(=J) in Fig. 3 and 4. And the number of units l is equal to 21 in Fig.6. When agent gets the goal, agent is given reward, 1.0. For the case of collision with wall, reward is -1.0, and for each action except collision is - 0.1. Other parameters used in this simulation are shown in Table 2.

5.2 Simulation and results

5.2.1 In the case of a simple maze

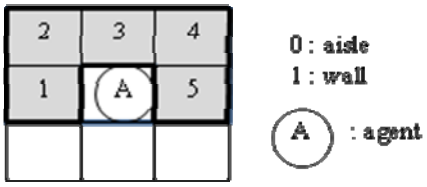


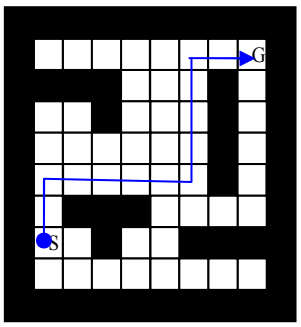
Fig.8 Perceptible area of agent : shaded area

Table 1 Action code of agent

action	forward	backward	right	left
code	0000	0001	0010	0011

Table 2 Parameters of simulation

ACTOR-CRITIC			
σ	0.1	ξ	0.7
η	0.3	γ	0.5
T	0.3	-	-
Forgetting and Learning coefficients			
λ_S	0.89	η_S	1.00
λ_L	1.00	η_L	1.00
Chaos control parameters of ACNN			
	Chaos / Non-chaos		Chaos / Non-chaos
α	10.0/1.00	k_r	0.98/0.10
ε	0.015/0.015	k_f	0.20/020
T	0.3/0.3	-	- -



Number of Layer in the S.T.M.	The number of stored patterns
Layer 0	54062
Layer 1	6073
Layer 2	6332
Layer 3	370
Layer 4	20033
Layer 5	1949
Layer 6	73
Layer 7	9996
Layer 8	13383
Layer 9	3007
total	115278

(a) (b)

Fig.9 Experimental maze 1 and results

At first, there is no data in the L.T.M., agent learns the shortest path of the maze of Fig.9(a) by using the actor-critic system and stores the result of learning in the ACNN of the S.T.M. corresponding to state s in the form of Eq.(16). The final refined result for the mae

is sent to be stored in the first layer (= unit(0))of the L.T.M. After learning, agent restarted from the initial position and got the information from each layer of the L.T.M. and got the goal like the arrow line in Fig.9(a). Fig.9(b) shows that the number of stored patterns concentrates at Layer 0, this is because that when agent goes this maze again, agent uses the information in unit(0) of the L.T.M., but retrieval in ACNN failed on the way, all the information W_L of unit(0) moved to the S.T.M. and additional actor-critic learning was done and learning results were written additively in the form of Eq.(18) and all the information W_L^{new} was sent to unit(1) of the L.T.M. as a new experienced information. Fig.9(b) shows the number of stored patterns by Eq.(16) twice and failure happened at Layer 0, so the number of stored patterns concentrated at Layer 0.

5.2.2 In the case of aliasing

Agent moves keeping to take the posture such that front of the agent is always upside of the paper. In Fig.10, the optimal path at state A is right, however, its path at B is left. Agent perceives state A and B as same state, but its optimal actions are different, this is called aliasing. In our case, both patterns are stored as different patterns. Our method solves this problem by using chaos control of ACNN. Namely, in the case of same sate, same input to ACNN, ACNN outputs either left or right as agent action, consequently agent moves right at A.

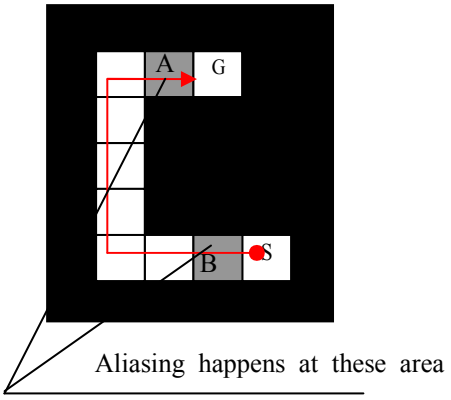
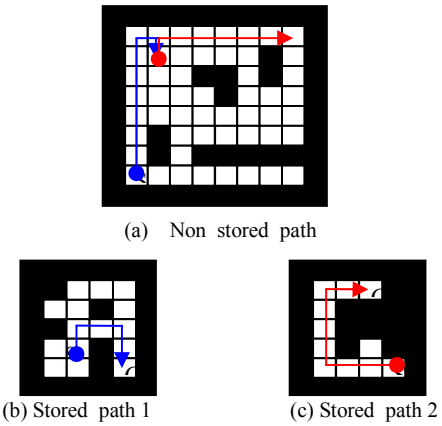


Fig.10 Experimental maze 2

5.2.3 In the case of use of stored path information

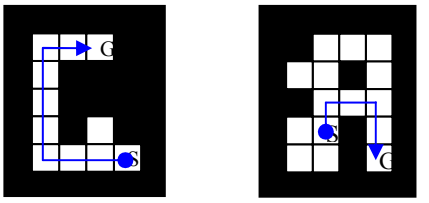
At first, there is no data in the L.T.M., agent learns the shortest path of the maze of Fig.11 (b) by using the actor-critic system and stores the result of learning in the S.T.M. in the form of Eq.(15) for each action. The final refined result is sent to be stored in the first layer (= unit(0)) of the L.T.M. in shown Fig.7. Second, for the maze of Fig.11(c), there is the path information, agent tries to get the action using the ACNN of unit(0) in the L.T.M., but fails because of no information corresponding to this environment. Agent also learns



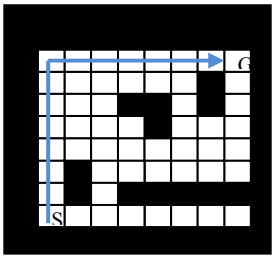
(a) Non stored path

(b) Stored path 1 (c) Stored path 2

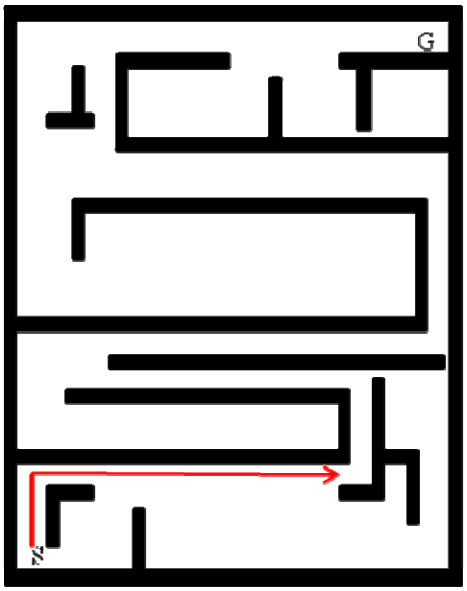
Fig.11 Experimental mazes and results



(a) Stored path 1 (b) Stored path 2



(c) Stored path 3



(d) Non stored path

Fig.12 Experimental large scale maze

and the final refined result is sent to the second layer (= unit(1)) of the L.T.M.. Fig.11(a) shows the results that agent is moving along the optimal path by making use of experiences (memory), that is, (b) and (c). The colored path in Fig.11 (a) corresponds to those of (b) and (c).

5.2.4 In the case of large scale maze

After learning of plural small size mazes, Fig. 12(a) to (c), agent tried to get the goal for the maze in Fig.12 (d). Agents could not associate its action at the top of the arrow in Fig.12(d). To get the goal in such a large scale maze, many experienced mazes are needed.

6. CONCLUSIONS

We proposed a reinforcement learning system with chaotic neural networks-based adaptive hierarchical memory structure for autonomous robots and showed its effectiveness through goal searching problem in plural mazes.

In our future work, we would like to try to expand this method to be used in the case of continuous environment.

Acknowledgements

A part of this study was supported by JSPS-KAKENHI (No.18500230, No.20500277 and No.20500207).

REFERENCES

[1] R.S. Sutton, A.G. Barto : "Reinforcement Learni ng", The MIT Press,1998

[2] M. Adachi, K. Aihara : "Associative Dynamics in a Chaotic Neural Network", *Neural Networks* ,Vol. 10, No. 1, pp.83-98,1997