Basin of Attraction of Associative Memory as it is Evolved by a Genetic Algorithm

Akira Imada
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara,
630-01 Japan
akira-i@is.aist-nara.ac.jp

Keijiro Araki
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara,
630-01 Japan
araki@is.aist-nara.ac.jp

Abstract— **We are applying genetic algorithms to fully connected neural network model of associative memory, We reported elsewhere that random weight matrix evolves to store some number of patterns only by means of a Genetic Algorithm. And we also reported the Genetic Algorithm can enlarge storage capacity of Hebb-rule associative memory. In those two reports, however, we did not mention about the basin of attraction. In this paper, we report concerning the basin of attraction of the networks obtained in those two experiments above.**

I. Introduction

Associative memory is a process in which we can store information by distributing it among neurons. One of the most appealing features is its capability to tolerate noisy and/or partial input in recalling the information, though it has a limited capacity. We are thinking of the possibility to use it as a memory organ of digital organisms of our Artificial Life research in the future. However, as a preliminary step, we are now investigating basic behavior of the function of associative memory under simple evolutionary processes.

In 1982, Hopfield [1] proposed a neural network model as an associative memory system. He used Hebbian rule [2] to make connection matrix store a set of patterns. He estimated the storage capacity to be at most 15% of the number of neurons with computer simulations. Since then many researchers have been trying to enlarge the storage capacity in various ways (see e.g., [3], [4]).

However, the enlargement of capacity spoils the error correcting capability of the network. The size of basin of attraction decreases as the number of storage patterns increases. In fact, Personnaz et al. [5] shows that the basin of attraction of a stored memory falls sharply as the number of patterns approaches to a half of the number of neurons for auto-associative memory. To know more about associative memory, see [6].

Genetic algorithms are a kind of searching techniques ([7], [8]). Since the late 1980's, genetic algorithms have been extensively used to search the optimal set of connection weights of neural networks and/or their optimal architectures (see [9], [10] and references quoted therein). However all of these researches were for layered neural networks, and applications of genetic algorithm for Hopfield networks remain few so far, probably because of the fact that Hopfield network has a fixed architecture. However, sparse connectivity has a possibility to enhance the original associative memory capacity of Hopfield network [3]. Therefore, genetic algorithms are also valid for Hopfield neural networks to explore their architectures as well as to determine their connection weights.

In an earlier papers, we showed that randomly generated connection matrices evolve to store some number of patterns by a genetic algorithm without using any learning rules [11]. We also showed that the genetic algorithm enlarges the capacity of Hebb-rule associative memory by pruning some connections [12]. In these two experiments, we did not refer to the basin of attraction of the associative memory. In this paper, we present some results regarding the basin of attraction derived from these two experiments.

The overall goal for this research is twofold. One is to know if we can use the genetic algorithm as a more effective learning method than those proposed so far, and the other is to clarify the mechanism of this *learning on a population level* of the genetic algorithm.

The following section provides an overview of the implementation of our genetic algorithm. Then results concerning the storage capacity and the size of basin of attraction of obtained networks are described in Section III. Finally some conclusions and ongoing works are given.

II. Implementation of our genetic algorithm

In this section we describe the implementation of our genetic algorithm briefly. As in Figure 1, our genetic algorithm proceeds as follows:

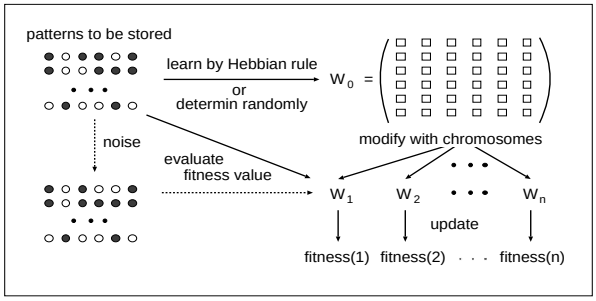

Fig. 1. Schematic representation of our genetic algorithm.

(1) First of all, a weight matrix W_0 is determined either randomly or by Hebbian rule. Each component of the matrix w_{ij} is chosen from $\{-1, 1\}$ in the former case, and in the latter case it is determined by

$$w_{ij} = \sum_{\mu=1}^{p} \xi_i^\mu \xi_{j\neq i}^\mu.$$

We simulate with 49 neurons of Hopfield network, hence the size of W_0 is 49x49 here. This original weight matrix remains unchanged during evolution. **(2)** Then 256 chromosomes are produced randomly. The chromosome has a fixed length of 2401(=49x49) alleles, and these alleles are chosen randomly from $\{-1, 0, 1\}$, where the probability of choosing either -1 or 0 is set to 0.02 in this paper. As will be mentioned in procedure(3) below, each component of the original weight matrix w_{ij} is multiplied by one of these alleles. Allele 0 implies to prune the connection which is multiplied, and -1 to reverse excitatory/inhibitory connection. We denote the *i-th* allele of the *n-th* chromosome as $c^{(n)}(i)$. **(3)** Each chromosome modifies the original weight matrix W_0, which produces 256 weight matrices slightly different from W_0. The modification was made as follows:

$$w_{ij}^{(n)} = w_{ij} \cdot c^{(n)}(49i + j),$$
$$(i, j = 0, 1, 2, \cdots, 48; \quad n = 1, 2, 3, \cdots 256),$$

where $w_{ij}^{(n)}$ denotes i-j component of the *n-th* weight matrix in the population. We must note that the larger the population number, the higher the performances. In these experiments, however, we set it to 256 just because of our computer resources. **(4)** Evaluation of fitness value f for each individual weight matrix is made with a set of randomly pre-determined patterns ξ^μ. First we calculate the total of all the inner products (overlap) of initial states and states at each time of update not more than certain time t_0. And then these are summed up over all sets of initial patterns, i.e.,

$$f = \frac{1}{p \cdot (t_0 - 1) \cdot 49} \sum_{\mu=1}^{p} \sum_{t=2}^{t_0} \sum_{j=1}^{49} \xi_j^\mu s_j^\mu(t),$$

where ξ_i^μ is the *i-th* bit of the μ-*th* pattern which takes the value of either -1 or 1, and $s_i^\mu(t)$ is the state of the *i-th* neuron at time t when the μ-*th* pattern is given to the network. In this paper, t_0 is set to 98 i.e., twice the number of neurons, and we observed that was enough. We must note that fitness 1.000 implies that all the initial patterns are stored as fixed points, while fitness less than 1.000 includes many possible cases. If we evaluated the fitness at one point of updating time instead, some solutions might be limit cycle. We adopted the above time-consuming fitness evaluation in order to avoid these oscillated solutions. **(5)** Two parent chromosomes are chosen uniformly at random from the upper 40% of the population which is ranked by fitness. Then those are recombined. **(6)** Recombinations are made with uniform-crossover. We tested several types of crossover including one- and two-point-crossover, and observed that the uniform-crossover outperformed the others. Furthermore, the offspring from each crossover are each mutated in times, where mutation rotates the value of randomly chosen bit in chromosome $c^{(n)}(i)$ cyclically, i.e.,

$$(1) \rightarrow (-1), \quad (-1) \rightarrow (0), \quad (0) \rightarrow (1).$$

The mutation rate is set to 0.01 in this paper. **(7)** Unless highest fitness value reaches the value of 1.000 nor generation exceeds 12000, the individuals in upper subpopulation (40%) survive to constitute the next generation with their offspring(60%), and the processes from (2) to (7) are repeated.

Both these operations and parameter values were determined mainly on the basis of trial and error.

III. Results and Discussion

In this section, we present the results of simulations of associative memory as it is evolved by genetic algorithm starting with both random weights and Hebbian learned weights. All simulations presented in this paper were carried out on networks with 49 neurons.

Genetic algorithm per se has an ability to evolve a random weight matrix eventually to store certain number of patterns. In fact, we succeeded in obtaining a weight matrix which can store 8 patterns (16% of the number of neurons) as fixed points by evolving randomly determined weight matrix [11].

On the other hand, the same genetic algorithm can also help to enlarge the storage capacity of Hebb-rule associative memory [12]. The associative memory capacity obtained in that experiment was around 33% of the number of neurons, while capacity of the original Hebb-rule matrix is 15% at most.

These learning mechanisms might be referred to

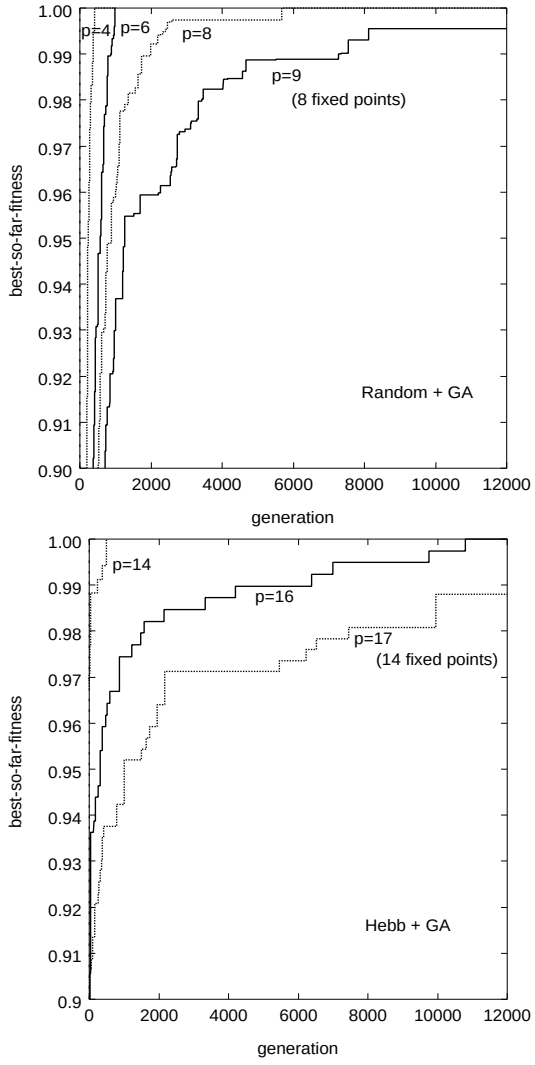

Fig. 2. Best-so-far-fitness vs generation for both the evolution from random weights (upper) and Hebbian weights (lower).

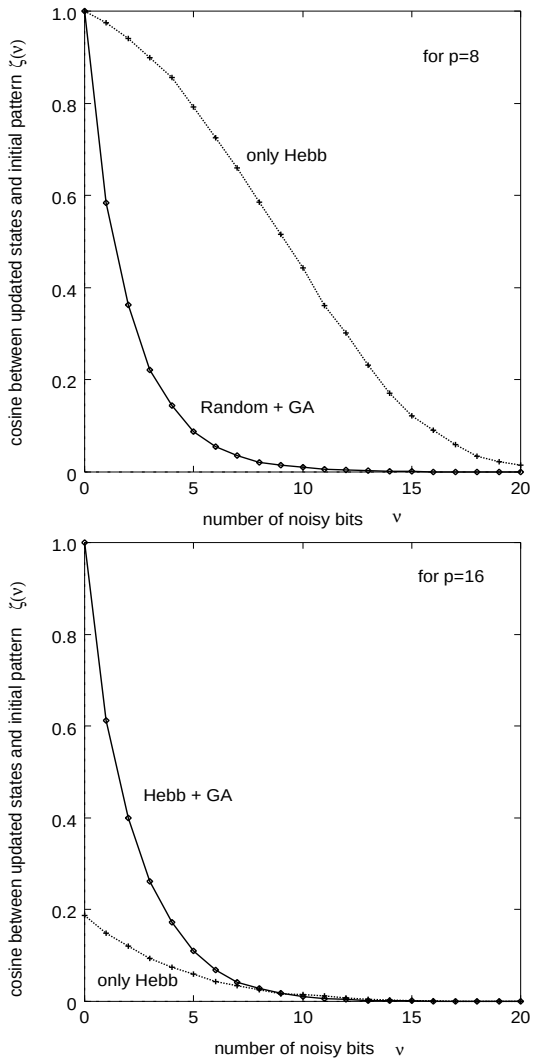

Fig. 3. Similarity of the updated states for the noisy input to its corresponding correct pattern (for the same sample appeared in Figure 2).

as *learning on a population level through evolution.*

The representative examples of best-so-far-fitness against generation of above two experiments are shown in Figure 2. We can see the storage capacities for the evolution from random matrix and Hebbian matrix are 8 and 16, respectively. The number of patterns stored as fixed points for p=9 in the evolution from random matrix is 8, while 14 for p=17 in the evolution from Hebbian learned matrix.

To see the size of basin of attraction, we give noisy inputs of the stored patterns to the networks produced from above two experiments. And the similarity of the updated states for the noisy input to its corresponding correct pattern is observed. To be more specific, all the stored pattern ξ^μ are added ν bits of noise randomly and given to the network to test the tolerance to them. These are repeated 800 times here. Then the similarity $\zeta(\nu)$ is defined by

$$\zeta(\nu) = \frac{1}{800 \cdot p \cdot 49} \sum_{k=1}^{800} \sum_{\mu=1}^{p} \sum_{j=1}^{49} \xi_j^\mu \cdot s_{j\nu k}^\mu(98),$$

where, $s_{j\nu k}^\mu(t)$ is the state of the j-th neuron at time t when the k-th set out of these 800 combinations of ν-bit noisy μ-th pattern is given to the network. In Figure 3, these are depicted as a function of number of noisy bits of the input, both for the network which learn the patterns only by Hebbian rule and for the network produced from the genetic algorithm above. The slower the decay of the curve, the broader the size of basin of attraction it should indicate. In these experiments, however, we observe rather steep decays both in evolutions from random/Hebbian weight matrix. In the case of evolution starting with a random weight matrix (upper figure), we can see big difference between two curves, which shows less capability to tolerate

the noisy input in the network produced by the genetic algorithm. In the case of evolution starting with a Hebbian learned weight matrix (lower figure), although it does not make sense to compare them since 16 patterns to be stored exceed the storage capacity of Hebb-rule associative memory system, we can also see steep decay in the curve.

IV. Summary and Ongoing Works

In this paper, we have described an application of a simple genetic algorithm to fully connected neural network model of associative memory, and we have shown some results of the storage capacity and the basin of attraction.

We obtained the storage capacity of 33% of the number of neurons in the evolution from Hebbian learned matrix, and 16% in the evolution from random matrix.

To obtain larger capacity, we are now exploring the *"learning on an individual level during its life time"* instead of *"learning on a population level through evolution"* as in this paper. One approach is to introduce the Baldwin effect on the evolution [13], and the other is to exploit the Lamarckian inheritance [15]. In the latter case, we implemented the new version of genetic algorithm [14], in which chromosome of each individual is made up of components of its weight matrix directly.

As for the basin size, on the other hand, we do not obtain the satisfactory results in these experiments. The basin sizes are rather small both in evolution from random/Hebbian weight matrix.

We are now using noisy inputs repeatedly as test samples for fitness evaluation in the genetic algorithm (see dotted line in Figure 1) to try to enlarge the basin size for both cases from random/Hebbian weight Matrix with considerably successful results. ([16], [17]).

Acknowledgments

We thank Peter Davis at Advanced Telecommunication Research Institute (ATR). The ideas in this paper were originally derived from discussions with him.

References

[1] J. J. Hopfield, Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences, USA* **79** pp.2554–2558, 1982

[2] D. O. Hebb, *The Organization of Behavior.* Wiley, 1949

[3] H. Yanai, Y. Sawada, and S. Yoshizawa, Dynamics of an Auto-Associative Neural Network Model with Arbitrary Connectivity and Noise in the Threshold. *Network.* **2**(3) pp.295–314, 1991

[4] M. Morita, Associative Memory with Nonmonotone Dynamics. *Neural Networks* **6** pp.115–126, 1993

[5] L. Personnaz, I. Guyon, and G. Dreyfus, Collective Computational Properties of Neural Networks: New Learning Mechanisms. *Physical Review* A**34**(5), pp.4217–4227, 1986

[6] M. H. Hassoun (Ed.), *Associative Neural Memories: Theory and Implementation.* Oxford University Press, 1993

[7] J. Holland, *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, 1975

[8] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989

[9] J. D. Schaffer, D. Whitley, and L. J. Eshelman, Combinations of Genetic Algorithms and Neural Networks: A Survey of the State of the Art. *Proceedings of the Workshop on Combinations of Genetic Algorithms and Neural Networks.* 1–37, 1992

[10] X. Yao, A Review of Evolutionary Artificial Neural Networks. *International Journal of Intelligent Systems.* **8** pp.539–567, 1993

[11] A. Imada, and K. Araki, Mutually Connected Neural Network Can Learn Some Patterns by Means of GA. *Proceedings of the World Congress on Neural Networks.* **1**, pp.803–806, 1995

[12] A. Imada, and K. Araki, Genetic Algorithm Enlarges the Capacity of Associative Memory. *Proceedings of 6th International Conference on Genetic Algorithms.* pp.413–420, 1995

[13] A. Imada, and K. Araki, Baldwin Effect on the Evolution of Associative Memory. (to submit), 1996

[14] A. Imada, and K. Araki, Simple Genetic Algorithm as a Learning Method for Associative Memory. (to submit), 1996

[15] A. Imada, and K. Araki, Lamarckian Evolution of Associative Memory. *IEEE International Conference on Evolutionary Computation* (to appear), 1996

[16] A. Imada, and K. Araki, Basin of Attraction of Hebb-rule Associative Memory as it Evolves. (to submit), 1996

[17] A. Imada, and K. Araki, Basin of Attraction of Associative Memory as it is Evolved from Random Weights. (to submit), 1996