Feed Forward Genetic Image Network: Toward Efficient Automatic Construction of Image Processing Algorithm

Shinichi Shirakawa and Tomoharu Nagao

Graduate School of Environment and Information Sciences, Yokohama National University, 79-7, Tokiwadai, Hodogaya-ku, Yokohama, Kanagawa, 240-8501, Japan
shirakawa@nlab.sogo1.ynu.ac.jp, nagao@ynu.ac.jp

Abstract. A new method for automatic construction of image transformation, Feed Forward Genetic Image Network (FFGIN), is proposed in this paper. FFGIN evolves feed forward network structured image transformation automatically. Therefore, it is possible to straightforward execution of network structured image transformation. The genotype in FFGIN is a fixed length representation and consists of string which encode the image processing filter ID and connections of each node in the network. In order to verify the effectiveness of FFGIN, we apply FFGIN to the problem of automatic construction of image transformation which is "pasta segmentation" and compare with several method. From the experimental results, it is verified that FFGIN automatically constructs image transformation. Additionally, obtained structure by FFGIN is unique, and reuses the transformed images.

1 Introduction

In image processing, it is difficult to select image processing filters to satisfy the transformation from original images to its target images. The system of ACTIT (Automatic Construction of Tree structural Image Transformation)[1–3] have been proposed previously. ACTIT approximates adequate image transformation from original images to their target images by a combination of several known image processing filters. ACTIT constructs tree structured image processing filters using Genetic Programming (GP)[4, 5]. Recently, the extended method of ACTIT, Genetic Image Network (GIN)[6], have been proposed. Instead of tree representation, the representation of GIN is arbitrary network structure. The output images of each node in GIN changes every step. Thus, in GIN the users must decide the parameter of "the number of steps". This parameter is the step of evaluating the output image.

This paper introduces a new method for automatic construction of image transformation. This new method, named Feed Forward Genetic Image Network (FFGIN), uses feed forward network representation. FFGIN evolves the feed forward network structure of image processing filters based on instance based learning in similar ways of the case of ACTIT. The characteristic of FFGIN is its structure of connections of image processing filters (feed forward network structure). In FFGIN, it is possible to straightforward execution of network structured image transformation, and it does not need the parameter of "the number of step". In order to verify the effectiveness of FFGIN, we apply FF-GIN to the problem of automatic construction of image transformation which is "pasta segmentation".

The next section of this paper is an overview of several related works. In section 3, we describe our proposed method, Feed Forward Genetic Image Network (FFGIN). Next, in section 4, we apply the proposed method to the problem of automatic construction of image transformation and show several experimental results. Finally, in section 5, we describe conclusion and future work.

2 Related Works

2.1 Genetic Programming and Graph based Genetic Programming

Genetic Programming (GP)[4, 5] is one of Evolutionary Computation techniques, which was introduced by Koza. GP evolves computer programs, which are usually tree structure, and searches a desired program using Genetic Algorithm (GA). Today, a lot of extensions and improvements of GP are proposed. Parallel Algorithm Discovery and Orchestration (PADO) [7, 8]is one of the graph based GPs instead of the tree structure. PADO was applied to the object recognition problems. Another graph based GP is the Parallel Distributed Genetic Programming (PDGP)[9]. In this approach, the tree is represented as a graph with functions and terminals nodes located over a grid. Cartesian Genetic Programming (CGP)[10, 11] was developed from a representation that was used for the evolution of digital circuits and represents a programs as a graph. In CGP, the genotype is an integer string which denotes a list of node connections and functions. This string is mapped into phenotype of an index graph. Recently, Genetic Network Programming (GNP)[12, 13] which has a directed graph structure is proposed. GNP is applied to make the behavior sequences of agents and shows better performances compared to GP.

By the way, in the literature of genetic approaches for neural networks design or training, many kinds of methods have been proposed[14]. And the evolution of neural networks using Genetic Algorithm (GA) has shown the effectiveness in the various fields. The NeuroEvolution of Augmenting Topologies (NEAT) method for evolving neural networks has been proposed by Stanley[15]. Each genome in NEAT includes a list of connection genes, each of which refers to two node genes being connected.

2.2 Automatic Construction of Tree structural Image Transformation (ACTIT)

ACTIT[1–3] constructs tree structured image processing filters with one-input one-output filters and two-inputs one-output filters by using Genetic Programming (GP) to satisfy the given several image examples. The individual in ACTIT is a tree structured image transformation. The terminal nodes of a tree are the original images and non-terminal nodes are several kinds of image processing filters. A root node means an output image. The users give “Training images”, and ACTIT system constructs appropriate image processing automatically. 3D-ACTIT[2, 3] is extended method which automatically constructs various 3D image processing procedures, and applies to medical image processing.

2.3 Genetic Image Network (GIN)

Genetic Image Network (GIN)[6] is a method for automatically construction of image transformation. Instead of tree representation, the representation of GIN is arbitrary network structure. GIN is composed of several nodes which are well-known image processing filters whose input is one or two. The biggest difference between GIN and ACTIT is the structure of connections of image processing filters. Generally, network structure theoretically includes tree structure (i.e. network structure also represent tree structure). Therefore, the description ability of network representation is higher than that of tree structure. Genotype in GIN consists of a string which indicate image processing filter type and connections. The other work shows that GIN automatically constructs a simple structure for complex image transformation using its network representation[6]. The execution of GIN is as follows. Initially, we set original images to “in” nodes when GIN executes. All nodes synchronously transform each inputted images, and output the transformed image to destination nodes. The output images of each node changes every steps. After predefined iterations (called “the number of steps”), the images of output nodes are evaluated.

3 Feed Forward Genetic Image Network (FFGIN)

3.1 Overview

ACTIT uses Genetic Programming (GP) and has a tendency to create image processing filters with unnecessarily large size. This problem in GP is called *bloat* and increases computational efforts. In GIN, the output images of each node changes every step. Thus, the users must decide the parameter of “the number of step”. This parameter is the step of evaluating the output image. However, the optimum value of “the number of steps” does not find in advance. To overcome these problems, we propose Feed Forward Genetic Image Network (FFGIN) whose representation is feed forward network structure. It is possible to straightforward execution of network structured image transformation, and it does not need the parameter of ”the number of step”. The genotype of GIN is a string which denotes a list of image processing filter ID and connections of each node in the network.

The features of FFGIN are summarized as follows:

– Feed forward network structure of image processing filters.
– Efficient evolution of image processing programs without *bloat* through the genotype of fixed length string.

3.2 Structure of FFGIN

Feed Forward Genetic Image Network (FFGIN) constructs acyclic network structured image transformation automatically. Figure 1 illustrates an example of Phenotype (feed forward network structure) and Genotype (string representing Phenotype) in FFGIN. Each node in FFGIN is well-known image processing filters. In FFGIN, the feedback structure is restricted in genotype level. The nodes take their inputs from either the output of a previous node or from the inputs in a feed forward manner. Therefore, it is possible to straightforward execution of network structured image transformation, and it does not need the parameter of "the number of step". The main benefit of this type of representation is that it allows the implicit reuse of nodes in the network. To adopt evolutionary method, genotype-phenotype mapping is used in FFGIN system. This genotype-phenotype mapping method is similar to Cartesian Genetic Programming (CGP). The feed forward network image processing filters are encoded in the form of a linear string. The genotype in FFGIN is a fixed length representation and consists of string which encode the image processing filter ID and connections of each node in the network. However, the number of nodes in the phenotype can vary but is bounded, as not all of the nodes encoded in the genotype have to be connected. This allows the existence of inactive nodes. The length of the genotype is fixed and equals to $N_{node} * (n_{in} + 1) + N_{out}$, where N_{node} is the number of nodes, n_{in} is the maximum number of inputs of predefined filters and N_{out} is the number of output nodes. FFGIN constructs feed forward network structured image processing filters, thus it is possible to represent plural outputs. FFGIN enables to simultaneously construct plural image transformation using only a single network structure.

3.3 Genetic Operators

To obtain the optimum structure of FFGIN, an evolutionary method is adopted. The genotype of FFGIN is a linear string. Therefore, FFGIN is able to use a usual Genetic Algorithm (GA). In this paper we use *uniform crossover* and *mutation* as the genetic operators. The uniform crossover operator affects two individuals, as follows:

– Select several genes randomly according to the crossover rate P_c for each gene.
– The selected genes are swapped between two parents, and generate offspring.

The mutation operator affects one individual, as follows:

– Select several genes randomly according to the mutation rate P_m for each gene.
– The selected genes are randomly changed.

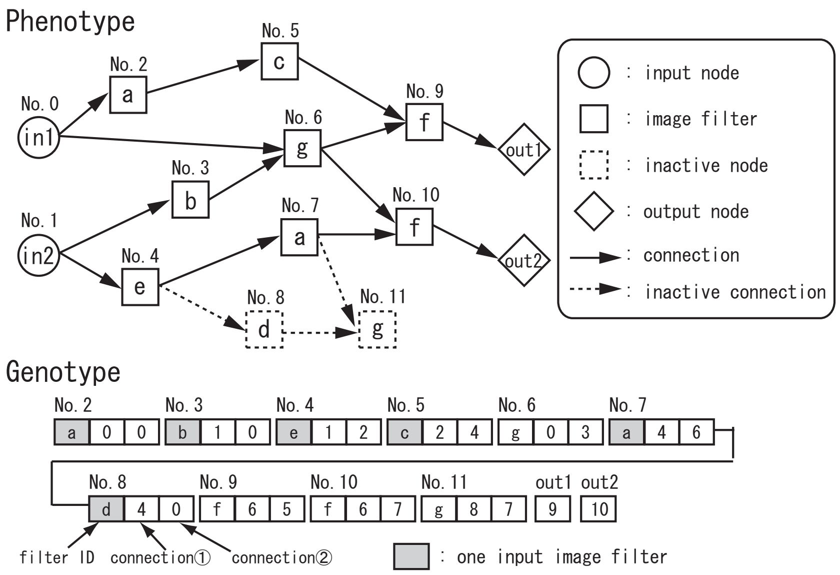

Fig. 1. Structure of FFGIN (phenotype) and the genotype which denotes a list of filter ID and connections.

4 Experiments and Results

In this section, we apply FFGIN to the problem of automatic construction of image transformation. Additionally, the effectiveness of FFGIN, GIN and ACTIT are compared.

4.1 Settings of Experiments

"Training images" we used in the experiments appear in Figure 2 (four training images). "Training images" consist of original images and target images. The number of "Training images" is four. Target images are the images that users require after the image processing (ideal results). This "pasta segmentation problem" is proposed as subject of a competition at Genetic and Evolutionary Computation Conference 2006 (GECCO 2006)[1]. The problem consists in evolving a detection algorithm capable of separating pasta pixels from non-pasta pixels in pictures containing various kinds of (uncooked) pasta randomly placed on textured backgrounds. The problem is made harder by the varying lighting conditions and the presence, in some of the images, of "pasta noise" (i.e., small pieces of pasta representing alphanumeric characters) which must be labeled as background. All images used in the experiments are gray scale images and the size of 128 × 96 pixel.

[1] http://cswww.essex.ac.uk/staff/rpoli/GECCO2006/pasta.htm

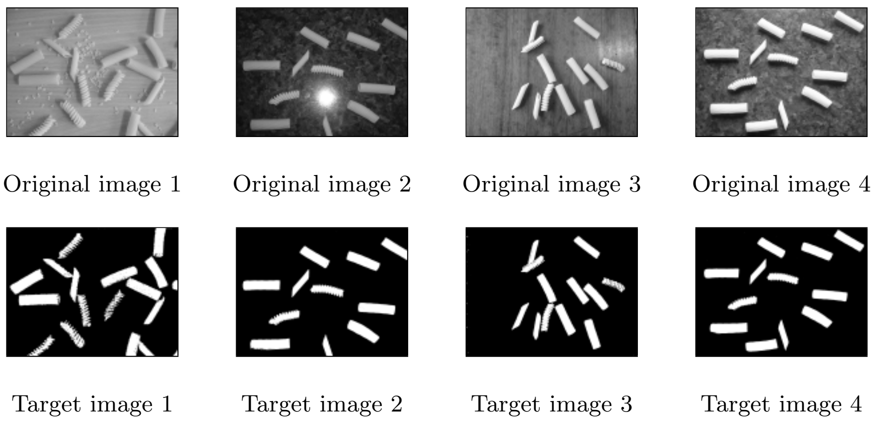

Fig. 2. "Training images" used in the experiments. The number of "Training images" is four.

Table 1. Parameters of each algorithm.

Parameter	FFGIN	GIN	ACTIT
The number of generations	5000	5000	5000
Population size N	150	150	150
Crossover rate P_c	0.9	0.9	N/A
Mutation rate P_m	0.03	0.03	0.9 (for individual)
Generation alternation model	MGG	MGG	MGG
Tournament size	5	5	5
The maximum number of nodes	50	50	50
The number of steps	N/A	10	N/A
The number of independent runs	10	10	10

We use the mean error on the "Training images" as a fitness function. The fitness function used in the experiments is:

$$fitness = \frac{1}{N} \sum_{n=1}^{N} \left\{ 1 - \frac{\sum_{i=1}^{W} \sum_{j=1}^{H} |o_{ij}^n - t_{ij}^n|}{W \cdot H \cdot V_{max}} \right\} \tag{1}$$

where o_n is the transformed image and t_n is its target one. The numbers of pixel in the direction i and j are W, H respectively, and the number of training image set is N. The range of this fitness function is [0.0, 1.0]. The higher the numerical value indicates the better performance.

We use Minimal Generation Gap (MGG) as the generation alternation model. The MGG model [16, 17] is a steady state model proposed by Satoh et al. The

MGG model has a desirable convergence property maintaining the diversity of the population, and shows higher performance than the other conventional models in a wide range of applications. We use the MGG model in the experiments as follows:

1. Set generation counter $t = 0$. Generate N individuals randomly as the initial population $P(t)$.
2. Select a set of two parents M by random sampling from the population $P(t)$.
3. Generate a set of m offspring C by applying the crossover and the mutation operation to M.
4. Select two individuals from set $M + C$. One is the elite individual and the other is the individual by the tournament selection. Then replace M with the two individuals in population $P(t)$ to get population $P(t + 1)$.
5. Stop if a certain specified condition is satisfied, otherwise set $t = t + 1$ and go to step 2.

In the experiments we use $m = 50$.

The parameters used by FFGIN, GIN and ACTIT are shown in Table 1. The common parameters between the three methods are identical. We prepare simple and well-known image processing filters in the experiments (27 one-input one-output filters and 11 two-inputs one-output filters). For instance, Mean filter, Maximum filter, Minimum filter, Sobel filter, Laplacian filter, Gamma correction filter, Binarization, Linear transformation, Difference, Logical sum, Logical prod and so on. FFGIN, GIN and ACTIT construct complex image processing from combination of these filters. Results are given for 10 different runs with the same parameter set.

4.2 Results and Discussion

Figure 3 shows the output images of "Training images" using FFGIN, and the fitness of this image transformation is 0.9676. FFGIN scored very high, and the output images are extremely similar to target images. FFGIN automatically constructs feed forward network structured image transformation.

Figure 4 is obtained structure (feed forward network structured image processing filters) using FFGIN. FFGIN constructs the structure of reusing the transformed images in its network. This structure cannot be constructed using ACTIT. We only give the "Training images", FFGIN constructs the ideal image processing automatically.

Next, we apply the constructed feed forward network structured image processing filters using FFGIN to "Test images". "Test images" are not used in evolutionary process (non-training images which are similar to training images). "Test images" used in the experiments are shown in Figure 5 (four images). The number of "Test images" is four. The results (output images) also appear in Figure 5. From the output images, FFGIN transforms the test images to the ideal images which extracted "pasta" in the varying lighting conditions and the presence. It shows that FFGIN automatically construct general image transformation through learning.

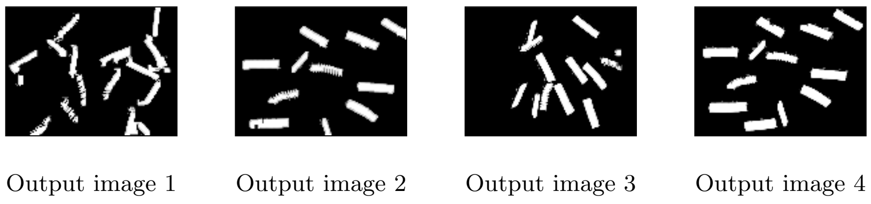

Fig. 3. Output images of “Training images” using FFGIN.

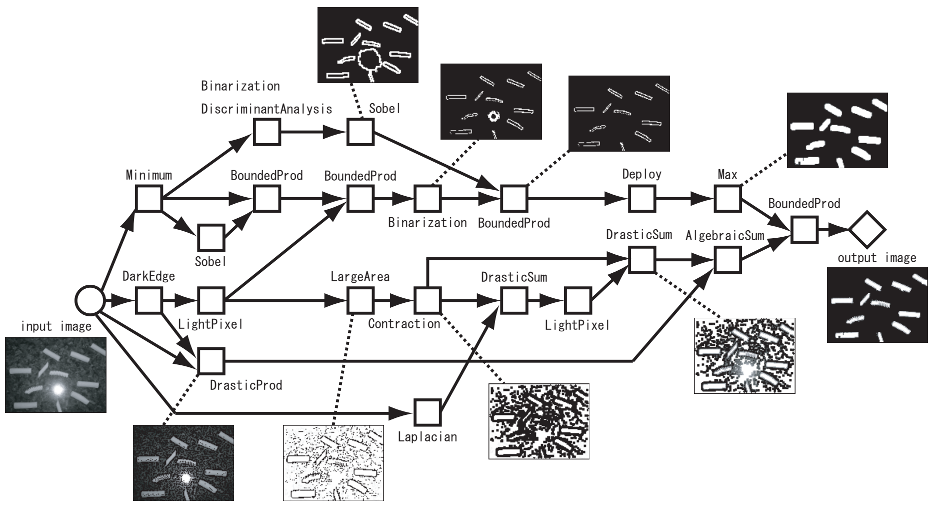

Fig. 4. An example of obtained structure (image processing filters) by FFGIN.

Figure 6 shows transition of average fitness of 10 independent runs. According to the result, we can evaluate all algorithms constructed adequate image processing algorithms. The performance of FFGIN is comparable with GIN and ACTIT.

Finally, we discuss the computational time of the experiments. We generate the results presented in this paper using a Intel Core 2 Duo E6400 processor with 1GB of memory. Table 2 shows the comparison of computational time between FFGIN, GIN and ACTIT. The computational time of FFGIN is about 8 times faster than GIN and about 5 times faster than ACTIT because of the evolution of without *bloat*. FFGIN allows the reuse of nodes, and the constructed structure (image processing filters) tends to compact. Therefore, the computational time decreases.

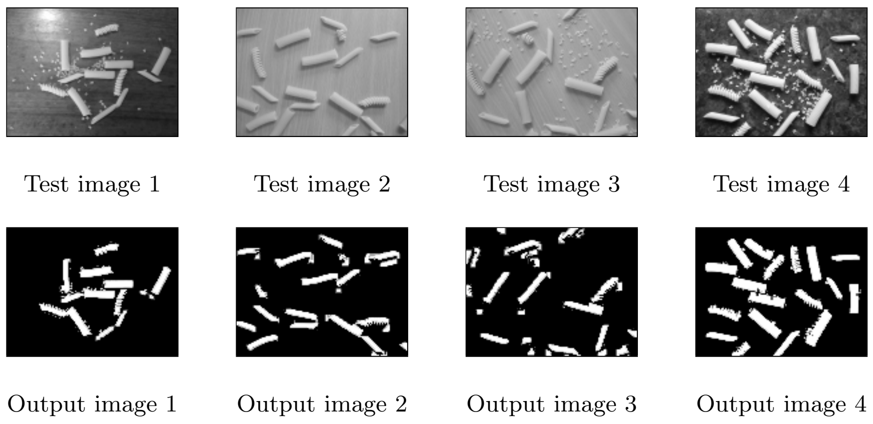

Fig. 5. “Test images” which are not used in evolutionary process (four images), and the output images of the obtained structure by FFGIN.

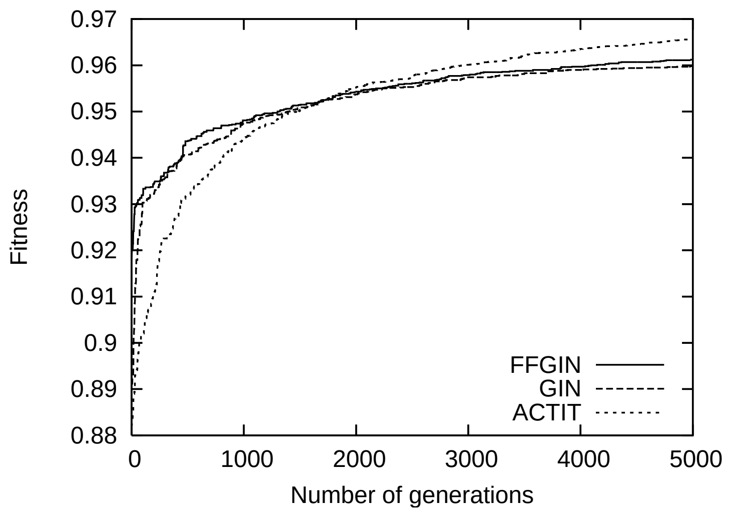

Fig. 6. The transition of fitness of FFGIN, GIN and ACTIT. Each curve is an average of 10 independent runs.

5 Conclusion and Future Work

In this paper, we proposed a new method for automatic construction of image transformation, Feed Forward Genetic Image Network (FFGIN), which evolves feed forward network structured image transformation programs. We applied FFGIN to evolve image processing algorithm of “pasta segmentation” and con-

Table 2. Average computational time in seconds for each algorithm (average of 10 independent runs).

	FFGIN	GIN	ACTIT
Computational time	5427	42620	28130

firmed that the optimum solution in each problem was obtained by the FFGIN system. From the experimental results, the performance of FFGIN is comparable with GIN and ACTIT, and the computational time of FFGIN is about 5 times faster than ACTIT. In future work we will apply FFGIN to other problems of image processing, in particular on larger problems and other types of problems. Moreover, we will introduce to the mechanisms of evolution of numerical parameters simultaneously in FFGIN.

References

1. Aoki, S., Nagao, T.: Automatic construction of tree-structural image transformation using genetic programming. In: Proceedings of the 1999 International Conference on Image Processing (ICIP-99). Volume 1., Kobe, Japan, IEEE (1999) 529–533
2. Nakano, Y., Nagao, T.: 3D medical image processing using 3D-ACTIT; automatic construction of tree-structural image transformation. In: Proceedings of the International Workshop on Advanced Image Technology (IWAIT-2004), Singapore (2004) 529–533
3. Nakano, Y., Nagao, T.: Automatic construction of abnormal signal extraction processing from 3D diffusion weighted image. In: Proceedings of the International Workshop on Advanced Image Technology (IWAIT-2007), Bangkok, Thailand (2007)
4. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA (1992)
5. Koza, J.R.: Genetic Programming II: Automatic Discovery of Reusable Programs. MIT Press, Cambridge, MA, USA (1994)
6. Shirakawa, S., Nagao, T.: Genetic image network (GIN): Automatically construction of image processing algorithm. In: Proceedings of the International Workshop on Advanced Image Technology (IWAIT-2007), Bangkok, Thailand (2007)
7. Teller, A., Veloso, M.: Algorithm evolution for face recognition: What makes a picture difficult. In: International Conference on Evolutionary Computation, Perth, Australia, IEEE Press (1995) 608–613
8. Teller, A., Veloso, M.: PADO: A new learning architecture for object recognition. In Ikeuchi, K., Veloso, M., eds.: Symbolic Visual Learning. Oxford University Press (1996) 81–116
9. Poli, R.: Evolution of graph-like programs with parallel distributed genetic programming. In: Proceedings of the Seventh International Conference on Genetic Algorithms, East Lansing, MI, USA, Morgan Kaufmann (1997) 346–353
10. Miller, J.F., Smith, S.L.: Redundancy and computational efficiency in cartesian genetic programming. IEEE Transactions on Evolutionary Computation **10** (2006) 167–174

11. Miller, J.F., Thomson, P.: Cartesian genetic programming. In: Genetic Programming, Proceedings of EuroGP'2000. Volume 1802 of LNCS., Edinburgh, Springer-Verlag (2000) 121–132
12. Hirasawa, K., Okubo, M., Hu, J., Murata, J.: Comparison between genetic network programming (GNP) and genetic programming (GP). In: Proceedings of the 2001 Congress on Evolutionary Computation (CEC01), Seoul, Korea, IEEE Press (2001) 1276–1282
13. Eguchi, T., Hirasawa, K., Hu, J., Ota, N.: A study of evolutionary multiagent models based on symbiosis. IEEE Transactions on Systems, Man and Cybernetics Part B **36** (2006) 179–193
14. Yao, X.: Evolving artificial neural networks. Proceedings of the IEEE **87** (1999) 1423–1447
15. Stanley, K.O.: Efficient evolution of neural networks through complexification. Technical Report AI-TR-04-314, Ph.D. Thesis; Department of Computer Sciences, The University of Texas at Austin (2004)
16. Satoh, H., Yamamura, M., Kobayashi, S.: Minimal generation gap model for considering both exploration and exploitations. In: Proceedings of the IIZUKA'96. (1996) 494–497
17. Kita, H., Ono, I., Kobayashi, S.: Multi-parental extension of the unimodal normal distribution crossover for real-coded genetic algorithms. In: Proceedings of the 1999 Congress on Evolutionary Computation (CEC99). Volume 2. (1999) 1581–1587