Using Genetic-Fuzzy-Neuro Model to Design Dual-FNNs Controller

Kaijun Xu Yang Xu Jiajun Lai Shuiting Wu

Intelligent Control Development Center, Southwest Jiaotong University, Chengdu 610031, P. R. China

Abstract

During the last decade, there has been increased use of neural networks (NNs), fuzzy logic (FL) and genetic algorithms (GAs) in artificial intelligence (AI). Since these three methods are complementary rather than competitive, a better performance model which has combined GAs, FL and NNs comes into being gradually. This paper presents genetic-fuzzy-neuro (G-F-N) model to design the dual-fuzzy neural-networks (DFNNs) controller. For the convenience of adaptive control, the structure of the two-fuzzy neural-network controller is divided into two parts. Each part is a fuzzy neural-network (FNN). The adaptive controller uses two FNNs. One FNN is used to identify a fuzzy model of controlled object. The other is online-tracking learning the suitable control policy.

Keywords: Dual-FNN, Genetic algorithms, Fuzzy logic, Neural networks, G-F-N model

1. Introduction

This paper presents a novel design method of dual-fuzzy neural networks (DFNNs) controllers using genetic-fuzzy-neuro (G-F-N) model. For the convenience of adaptive control, the structure of the DFNNs controller is divided into two parts. Each part is a fuzzy neural-network (FNN)[1]-[4]. At the same time, one fuzzy neural-network is controller, and its output is control input; the other is on-line tracking to learn the suitable control policy. When time passes, the outside situation has changed, the control policy is no longer suitable. Then the actions of two fuzzy neural-networks can be exchanged and the suitable control policy can be applied to real control system.

The objective of the present research work is to fuse GAs, FL, and NNs[5]-[8] to develop a genetic-fuzzy-neuro model to design the DFNNs controller. When one FNN is the controller, it is using the FL and NNs to control the system, the other is online-tracking learning using GAs and NNs to ensure it can learning the best control policy at any moment[9]-[13].

In this paper, we mainly discuss the DFNNs controller and how to use the genetic-fuzzy-neuro model to control the system and learning the suitable control policy. It is divided into six sections, the first one is introduction, then we will talk about the fuzzy control system. In section 3, DFNNs control system will be discussed. Next, we will advance the genetic-fuzzy-neuro (G-F-N) model to design the DFNNs controller and its adaptation process. Conclusion is the last section.

2. Fuzzy control system

The fuzzy controller controls the output of the controlled object y to follow the command with the manipulated variable u. The FNNs are used for the fuzzy model and the fuzzy controller. Fig. 1 shows the fuzzy control system and it is designed in the following process:

(1) The fuzzy model is identified from the input-output data of the controlled object.

(2) The fuzzy controller is designed with the linguistic fuzzy rules of the fuzzy model. The response of the control system is checked.

(3) The adaptive tuning of the control rules is done using the fuzzy model of the controlled object.

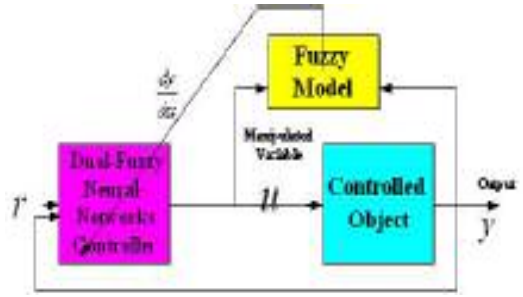

Fig.1: Fuzzy control system.

3. Dual-Fuzzy Neural-Networks (DFNNs) control system

We design a fuzzy neural-network controller that is constructed by two fuzzy neural-networks (Fig. 2). At the same time, one fuzzy neural-network is controller, and its output is control input; the other is on-line tracking learning. At the appropriate time, which is

decided by switching line that can be gotten from the fuzzy control rules table, and function of the two fuzzy neural-networks is exchanged. After exchange, the fuzzy neural-network that was used to control starts on-line tracking learning, and the fuzzy neural-network that was used to on-line tracking learning starts control. The process is continuous. Time series about DFNNs is shown in Fig. 3.

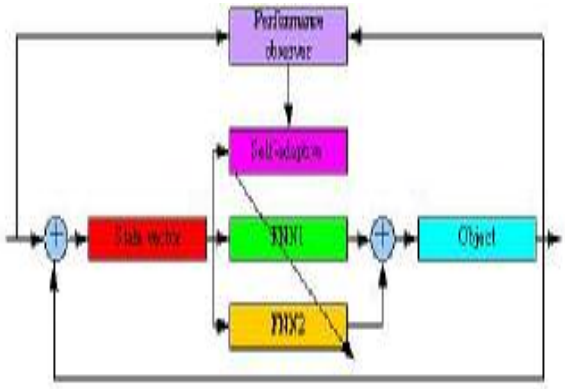

Fig.2: Dual-fuzzy neural-networks control system.

The on-line tracking learning is conventional gradient descent method, and according to Lyapunov stability theorem condition, the on-line tracking learning makes that system stable. Meanwhile, this makes self-adaptive process more effectively, and the control process of the system is not affected by the on-line tracking learning. This is advantageous to real control. Compared with the other studies, the proposed control is simple and effective. It is believed that the proposed control can be applied to many control systems.

We define the dual-fuzzy neural-network law as following:

$$U = \partial(t)U_{FNN1} + (1-\partial(t))U_{FNN2}, \quad (1)$$

Where U_{FNN1} and U_{FNN2} are output of dual-fuzzy neural-network, $\partial(t) \in \{0,1\}$, $\partial(t) = 0$ (or 1)is decided by the switching line (the switching region) which can be gotten from the fuzzy control rules.

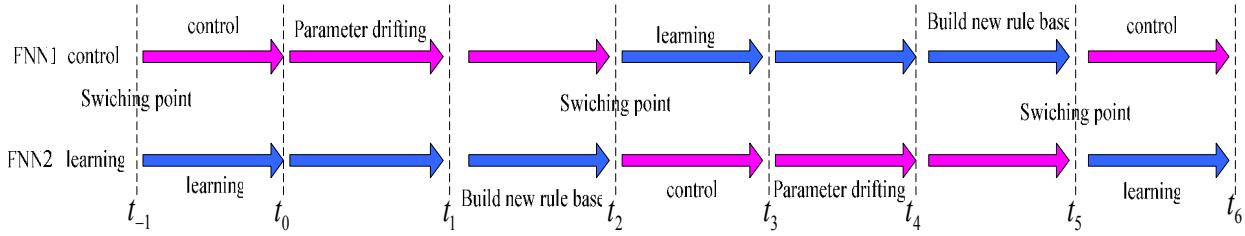

Fig.3: Time series about DFNNs control system.

When $U = U_{FNN1}$ (or $U = U_{FNN2}$), then U_{FNN2} (or U_{FNN1}) is learning, the learning process of FNN is main on-line tracking learning. if U_{FNN2} learning, the learning data come from U_{FNN1} and its learning does not affect the control. When the control error becomes badly, the two fuzzy neural networks are exchanged, and shift the learning from one FNN to the other.

4. Genetic-Fuzzy-Neuro (G-F-N) model architecture

The architecture of DFNNs using G-F-N model is shown in Fig. 4. The proposed G-F-N model is a fusion of GA, FL, and NN paradigms. The hybridization of GAs, FL, and NNs offset conquer the demerits of one paradigm by the merits of another. In the model, GAs are primarily concerned with optimization, FL with imprecision, and NN with fuzzy input–output mapping.

The process by which G-F-N model calculates the fitness of each chromosome to learn the suitable control policy consists of the following steps. In Fig. 4, the rule base and the inference engine in the traditional FLS are replaced by the NN. The NNs architecture and the recall of neural processing are used to represent the functions of the rule base and the inference engine, respectively. The hybridization of the FL and NN is regarded as ‘‘neuro with fuzzy input–output,’’ meaning a NN with both fuzzy inputs and fuzzy outputs [6][10]. In this work, for convenience, the term ‘‘neuro with fuzzy input–output’’ is called FNN which is a general phrase to express fusion of FL and NN[6].

Although FNNs seem to be more reasonable than traditional FL to simulate the characteristics and process of human inference, they have demonstrated difficulties in selecting an appropriate topology as well as appropriate parameters when dealing with different tasks. In addition, the determination of suitable distributions of MFs and defuzzification parameters

for solving disparate problems is a time-consuming process whose difficulties increase with problem complexity. GAs are good at optimization, providing an effective approach to cope with the drawbacks of simultaneously search for all parameters required in a FNN, which provides more chances to derive global FNN. Therefore, G-F-N model employs a GA to simultaneously search for the fittest shapes of MFs, optimal FNN topology, and optimal parameters of FNN. The novelty of this research is to use GAs to optimum solutions for specific problems with less efforts.

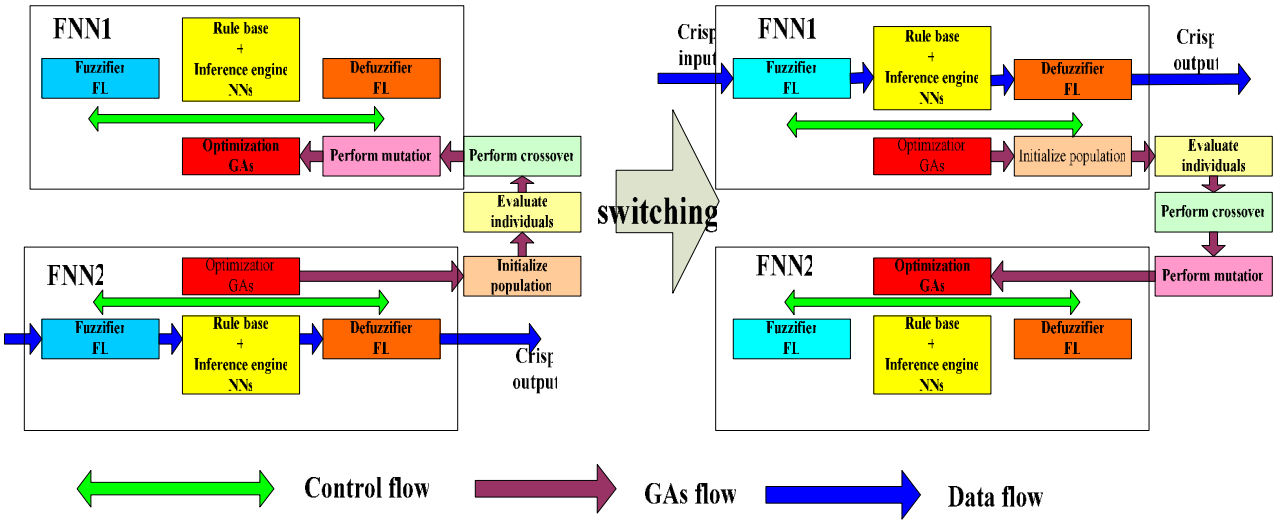

Fig.4: G-F-N model in DFNNs controller.

5. Genetic-Fuzzy-Neuro (G-F-N) model adaptation process

The adaptation process of G-F-N model is shown in Fig. 5, which adapts the evolutionary process of GAs. In the process, $P(t)$ denotes a population of ξ individuals at generation t, $P_0(t)$ denotes an offspring population of s individuals, and $P_M(t)$ denotes a mutation population of t individuals. In the beginning ($t = 0$), a population of ξ individuals is randomly generated. Each solution encodes model variables (such as distributions of MFs, NN topologies, interconnections, synaptic weights, etc.) into a binary string to simulate a natural chromosome. The fitness of chromosomes is then evaluated. The process by which G-F-N model calculates the fitness of each chromosome consists of the following steps. Firstly, the chromosome's genotype is converted to its phenotype. Then, the model evaluates the objective function of the chromosome by the derived phenotype. Finally, the value of objective function is converted into fitness. The crossover repeatedly exchanges high-performance notations in the search for better and better performance. It operates on a pair of chromosomes (parents) at a time and produces two children by exchanging the parents' features, e.g. model variables. The mutation produces spontaneous random changes in various chromosomes. It protects against premature loss of important notations while fine-tuning model variables. Next, the selection process emulates the survival-of-the-fittest mechanism in nature. It selects a new population with respect to the probability distribution based on fitness for survival. The selected new population is then submitted to evaluate their fitness to begin a new generation of evolvement. When time passes, the outside situation has changed, the control policy is no longer suitable. Then the actions of two fuzzy neural-networks can be exchanged and the suitable control policy can be applied to real control system.

5.1. Initialize population

Each solution encodes model variables into a binary string to simulate a natural chromosome. Every string (FNN string) comprises two segments: MF sub-string and NN sub-string. Two methods, SWRM and BRM, are employed to encode MFs and NNs into sub-strings. The SWRM and BRM encode MFs and NN by a fixed and variable sub-string, respectively. Therefore, the EFNIM encodes the problem using variable length gene code. The lengths of the sub-strings depend on the characteristics of the variables including the required variable precision, amount of variables, and variable domains. Then, combining the MF sub-string and the NN sub-string together, the entire chromosome can be acquired. The values of genes are randomly generated

with 0 or 1 to produce random variables for each chromosome.

By using SWRM, the required length of MF sub-string, RLMF, for encoding MFs is carried out as follows:

$$RL^{MF} = \sum_{h=1}^{rn^{cMF}} (n_h^{sm} \times rl^{sm} + n_h^{wd} \times rl^{wd}), \quad (2)$$

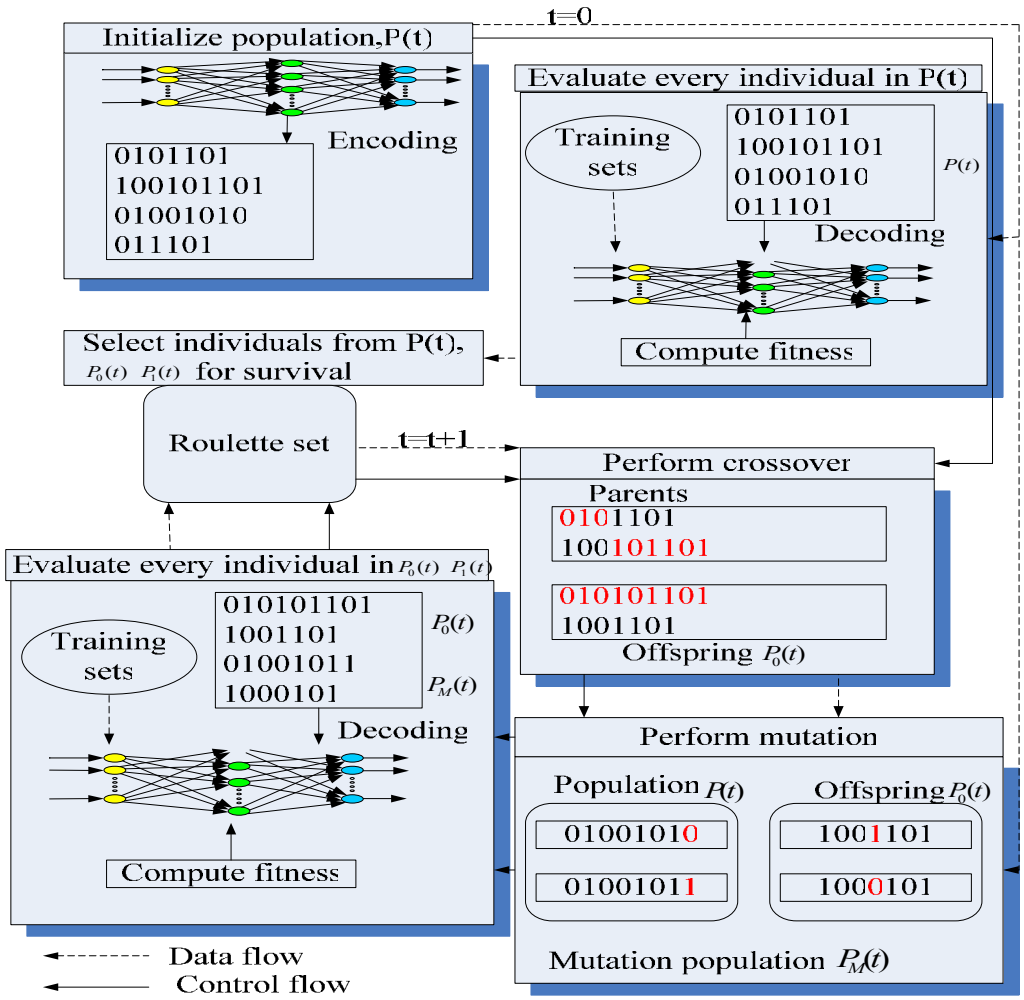

Fig.5: G-F-N model methodology.

Where rn^{cMF} is the required number of complete MF sets, n_h^{sm} is the number of summits in a complete MF set for input variable h, rl^{sm} is the required length for a summit depending on the precision demand, n_h^{wd} is the number of widths in one complete MF set for input variable h, and rl^{wd} is the required length for a width depending on the precision demand.

Considering all input variables use a common complete MF set or each input variable uses its individual complete MF set to fuzzify the crisp input data, rn^{cMF} can be expressed as

$$rn^{cMF} = \begin{cases} 1 \\ n^{iv} \end{cases}, \quad (3)$$

Where n^{iv} is the number of input variables. The crisp input is a value having only single number.

The number of summits in a complete MF set depends on the used MF shape and the number of fuzzy

sets in one complete MF set. So, n_h^{sm} can be defined as follows:

$$n_h^{sm} = \begin{cases} (n_h^{MF} - 2) \times 2 + 3 = 2n_h^{MF} - 1 \\ n_h^{MF} \end{cases}, \quad (4)$$

Where n_h^{MF} is the number of fuzzy sets in one complete MF set for input variable h.

The number of widths in one complete MF set can be calculated from the number of fuzzy sets in one complete MF set. Therefore, n_h^{wd} is calculated in the following form:

$$n_h^{wd} = (n_h^{MF} - 2) \times 2 + 2 = 2(n_h^{MF} - 1), \quad (5)$$

The mapping from a domain $\left[lb^x, ub^x\right]$ to a required length rl^x for variable x is straightforward and can be written as:

$$2^{rl^x - 1} < (ub^x - lb^x) \times 10^{rp} \le 2^{rl^x} - 1, \quad (6)$$

Where rp is the required places after the decimal point and lb^x , ub^x are the lower and upper bound values of the variable x.

Taking log functions on both sides of the right-hand parts of the inequality above yields

$$rl^x = \left\lceil \frac{\log(ub^x - lb^x) \times 10^{rp} + 1}{\log(2)} \right\rceil, \quad (7)$$

5.2. Evaluate individuals

The complete set of genes comprises a genotype of a chromosome, which consists of the MF sub-string and NN sub-string. The resulting organism, MFs and NN, is called a phenotype. In the previous sections, the initial solutions (phenotypes) are encoded into chromosomes (genotypes). In this section, the genotypes of MFs and NNs are decoded into their phenotypes. Here, the binary string is converted into relative real values. The present work encodes the MFs and NNs using SWRM and BRM, respectively. Therefore, the processes of decoding MF and NN sub-strings are related to the encoding procedures, which may be treated as reversion operations of SWRM and BRM.

5.3. Perform crossover

G-F-N model uses one-cut-point crossover and exchanges right parts of parents. The position of the cut point is randomly generated within the union length of parents, as shown in Fig. 5. It means that crossover exchanges the summit positions of MFs, widths of MFs, hidden layers, hidden neurons, interconnections, biases, and activation slopes.

In the crossover procedure, the produced children do not replace parents. The model keeps all produced children at the intermediary population

5.4. Perform mutation

The purpose of mutation is to adjust the value of summits and widths of MFs, interconnections, weights, biases, and activation slopes for better performance. It alters one or more genes with a probability (p^{ge}), which is smaller than or equal to mutation rate (p^{mu}). Mutation operation compares the gene's p^{ge} with p^{mu} bit by bit. If $p^{ge} \le p^{mu}$, then value of gene will be altered.

G-F-N model mutates the genes in population and children population. The mutated chromosomes do not replace the original ones. They are placed in the intermediary population.

5.5. Select individuals

G-F-N model uses the ‘‘roulette set’’ method to select fitter chromosomes on the enlarged intermediary pool. Performing selection on such enlarged sampling space improves GA performance by enlarging the searching space in parallel and by increasing crossover and mutation rates without introducing too much random perturbation. Also, it provides population, children population, and their mutation with the same chance of competing for survival.

6. Conclusions

In this paper, we proposed a new fuzzy control system, the dual-fuzzy neural-networks control system (DFNNs). The two fuzzy neural-networks have different actions in the control process, at the same time, one is controller, and the other is online tracking learning. This paper also presented a G-F-N model for evolving the optimum DFNNs. As a result, the proposed model has a potential to solve various kinds of problems that traditional FNNs can do and further to derive better results. G-F-N model is developed based

on simulating natural evolution process. Further research may construct G-F-N model to mathematically analyze the DFNNs control system adaptation process.

Acknowledgement

This paper is supported by the National Natural Science Foundation of P.R. China (Grant No. 60474022). Also be supported by the specialized Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20060613007).

References

[1] K. S. Narendra and K. Parthasarathy, Identification and Control of Dynamical Systems Using Neural Networks, *IEEE Trans. Neural Networks*, vol. I, pp. 628, 1990.

[2] R. M. Sanner and I. I. E. Slotine, Gaussian networks for direct adaptive control, *IEEE Trans. Neural Networks,* 3:837-863, 1992.

[3] G. A. Rovithakis and M. A. Christodoulou, Neural adaptive regulation of &own nonlinear dynamical systems, *IEEE Trans*. Svst. Man. Cvbem. B. vol. 27,. D.D. 81&822, 1997.

[4] S. Seshagiri and H. K. Khalil, Output feedback control of nonlinear systems using RBF neural network, *IEEE Trans. Neural Networks*, vol. i l , pp. 69-79, 2000.

[5] H. Ghezelayagh and K.Y. Lee, Application of self-organized neuro fuzzy identifier in intelligent predictive control of power plant. *Engineering Intelligent Systems* 13 (2):113–118, 2005.

[6] I. Hayashi, M. Umano, T. Maeda, A. Bastian and L. C. Jain, Acquisition of fuzzy knowledge by NN and GA—a survey of the fusion and union methods proposed in Japan, *Proceedings of IEEE International Conference on Knowledge-Based Intelligent Electronic Systems*, pp. 69–78, 1998.

[7] S.D Mohaghegh, Recent developments in application of artificial intelligence in petroleum engineering, *Journal of Petroleum Technology* 57 (4):86–91, 2005.

[8] S.K. Oh, W. Pedrycz , S.B. Rho and T.C. Ahn, Parameter estimation of fuzzy controller and its application to inverted pendulum. *Engineering Applications of Artificial Intelligence* 17(1):37–60. 2004.

[9] P.K Dash, S. Mishra, S. Dash and A.C. Liew, Genetic optimization of a self organizing fuzzy-neural network for load forecasting, *Proceedings of the IEEE Power Engineering Society Winter Meeting* 2, 1011–1016, 2000.

[10] E. Li, Jia. L and J. Yu, A genetic neural fuzzy system and its application in quality prediction in the injection process, *Chemical Engineering Communications* 191 (3):335–355, 2004.

[11] J.H. Holland, *Adaptation in Neural and Artificial Systems*, The University of Michigan Press, Ann Arbor, Michigan, 1975.

[12] L.A. Zadeh, Fuzzy sets, *Information and Control* 8 (3):338–353, 1965.

[13] S. Haykin , *Neural Networks: A Comprehensive Foundation*, second ed, Prentice-Hall, Upper Saddle River, NJ. 1999.