Emergence of Self-Learning Fuzzy Systems by a New Virus DNA–Based Evolutionary Algorithm

Lihong Ren,[1] Yongsheng Ding,[1,*] Hao Ying,[2] Shihuang Shao[1]
[1]*Department of Automation, College of Information Sciences and Technology, Dong Hua University, Shanghai 200051, China*
[2]*Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202*

In this article, we propose a new approach to the virus DNA–based evolutionary algorithm (VDNA-EA) to implement self-learning of a class of Takagi-Sugeno (T-S) fuzzy controllers. The fuzzy controllers use T-S fuzzy rules with linear consequent, the generalized input fuzzy sets, Zadeh fuzzy logic and operators, and the generalized defuzzifier. The fuzzy controllers are proved to be nonlinear proportional-integral (PI) controllers with variable gains. The fuzzy rules are discovered automatically and the design parameters in the input fuzzy sets and the linear rule consequent are optimized simultaneously by the VDNA-EA. The VDNA-EA uses the VDNA encoding method that stemmed from the structure of the VDNA to encode the design parameters of the fuzzy controllers. We use the frameshift decoding method of the VDNA to decode the DNA chromosome into the design parameters of the fuzzy controllers. In addition, the gene transfer operation and bacterial mutation operation inspired by a microbial evolution phenomenon are introduced into the VDNA-EA. Moreover, frameshift mutation operations based on the DNA genetic operations are used in the VDNA-EA to add and delete adaptively fuzzy rules. Our encoding method can significantly shorten the code length of the DNA chromosomes and improve the encoding efficiency. The length of the chromosome is variable and it is easy to insert and delete parts of the chromosome. It is suitable for complex knowledge representation and is easy for the genetic operations at gene level to be introduced into the VDNA-EA. We show how to implement the new method to self-learn a T-S fuzzy controller in the control of a nonlinear system. The fuzzy controller can be constructed automatically by the VDNA-EA. Computer simulation results indicate that the new method is effective and the designed fuzzy controller is satisfactory. © 2003 Wiley Periodicals, Inc.

1. INTRODUCTION

Fuzzy systems have been used successfully for a variety of applications, especially in control applications and modeling applications.[1] To analyze the

*Author to whom all correspondence should be addressed: e-mail: ysding@dhu.edu.cn.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 18, 339–354 (2003)
 Published online in Wiley InterScience (www.interscience.wiley.com). • DOI 10.1002/int.10097

design analytically fuzzy systems, analytical structures[2] and a universal approximation theory[2–4] for two major fuzzy systems, the Mamdani and Takagi-Sugeno (T-S) fuzzy systems, have been discussed. However, sometimes, expert knowledge cannot be described easily in linguistic language rules. In these cases, the alternative methods are to acquire automatically the fuzzy rules and determine the parameters of a fuzzy system. Many approaches such as artificial neural networks (e.g., Refs. 5, 6) and evolutionary algorithms [EAs; including genetic algorithms (GAs); e.g., See Refs. 7–10] have been proposed to develop fuzzy systems by automatically determining the design parameters in the input fuzzy sets and the rule sequence.

Although the EA (GA) approach possibly provides a way to obtain global optimization solution, it has some limitations. In general, the bit string (0–1's) encoding is the most common method adopted by EA (GA) researchers because of its simplicity and tractability. However, when the size of the population grows and the encoding of the chromosomes is very large, the expense of the evolutionary computation would become almost impractical. The rather long strings after encoding will increase the complexity of the problem. In addition, GAs are not effective for searching the solution space locally because of crossover-based search, and the diversity of the population sometimes decreases rapidly. Moreover, the GAs can neither represent the diverse genetic information by only using 0–1's encoding, nor can it better imitate the regulation action of genes to the genetic processes. As such, some biological operations at the gene level cannot be adopted effectively in the existing GAs. For the applications of GAs, a well-chosen chromosome format can enhance the understanding of the problem formulation and it is also flexible for practical implementation.

As we know, DNA is the major genetic material for life and it encodes plenty of genetic information. A natural question is Could we use the genetic mechanisms of the biological DNA to develop a new DNA-based computation model for optimization problems? Recently, extensive studies have been conducted to explore the possibility of using DNA as computing hardware (e.g., See Refs. 11–14). In addition, research has been directed at the soft computing aspect of the DNA computing,[15,16] e.g., evolutionary computation[17,18] and neural networks.[19,20] Among them, the integration of DNA computing with evolutionary computation is a major topic and the current interests are on two aspects: to explore the relationship between evolutionary computation and DNA-based computation[15,16] and to apply GAs to search for good DNA-encoding methods.[17] From the mechanism of evolutionarily inspired approaches, it can be seen that they seem suitable to be implemented by DNA. As such, to overcome the limitations of the GAs, a few GAs based on the mechanism of the biological DNA such as double-stranded DNA[18] and the DNA-encoding method[16,21–23] have been developed. We have used a DNA-encoding method and developed a DNA-based GA (DNA-GA) to find the fuzzy control rule sets in a Mamdani and a T-S fuzzy system.[16,21] Yoshikawa et al. combined the DNA-encoding method with the pseudobacterial genetic algorithm (PBGA)[23] and developed a DNA PBGA. Besides the algorithms based on DNA models, Chen et al. proposed the laboratory implementation of the DNA-GA for

some simple problems such as the MAX 1s, the royal road, and the cold war problems.[24]

The genetic information is a great source from which to develop variations of the basic GA scheme. For example, during the microbial evolutionary process, bacteria can transfer DNA to recipient cells through bacterial recombination at the bacterial genetic level. Genes can be transferred from a single bacterium to others, accelerating the evolution of the entire population.[25] Inspired by this phenomenon, we could develop a gene transfer operation and a bacterial mutation operation, which directly transfer a gene strand from one cell to other cells.

In this study, we develop a new virus DNA–based EA (VDNA-EA) and use it to design automatically a class of T-S fuzzy controllers. In Section 2, the configuration of the fuzzy controllers is defined. In Section 3, we show how to use the VDNA-EA to discover the effective fuzzy rules of the T-S fuzzy controllers. We incorporate the features of the VDNA and the bacterial evolution into the EA. Some important procedures of the VDNA-EA to design the T-S fuzzy controllers, such as the VDNA-encoding method and DNA-based genetic operators, are also discussed. In addition, we show how to use the VDNA-EA to design automatically the T-S fuzzy controllers by adding or deleting the number of fuzzy rules and tuning the design parameters in the input fuzzy sets and the rule sequence. In Section 4, we provide an example to show the efficiency and effectiveness of VDNA-EA in the design of the T-S fuzzy controllers.

2. CONFIGURATION OF A CLASS OF T-S FUZZY CONTROLLERS

The T-S fuzzy controllers studied in this article use two input variables and one output variable. The input variables are error and rate change of error of system output with respect to output set point. They are denoted as follows:

$$e(nT) = SP(nT) - y(nT)$$

$$r(nT) = (e(nT) - e(nT - T))/T$$

where n is a positive integer, T is sampling period, $SP(nT)$ is the setpoint, and $y(nT)$ is system output. The two input variables are both fuzzified by N input fuzzy sets. The generalized membership function is used for a fuzzy set and is designated as $\mu_i(e)$ [or $\mu_i(r)$], where $i = 1, \ldots, N$. The mathematical representation for the membership functions is

$$\mu_i(x) = GM(\alpha_i, \beta_i, \gamma_i) = e^{-|\alpha_i x + \beta_i|^{\gamma_i}}, \quad x = e \text{ or } r \tag{1}$$

where α_i, β_i, and γ_i are three design parameters in the membership functions. We call Equation 1 the generalized membership functions because with different values of the parameters α_i, β_i, and γ_i, Equation 1 can approximate various membership functions such as widely used triangular, trapezoidal, and gaussian functions.[21] That is to say, the definitions of the input fuzzy sets are very general and contain almost all of the fuzzy sets used in fuzzy systems.[1,2]

The N T-S fuzzy control rules with linear sequence are used. The form of the T-S fuzzy rules is as follows:

$$\text{if } e(n) = GM(\alpha_{ei}, \beta_{ei}, r_{ei}) \quad \text{and} \quad r(n) = GM(\alpha_{ri}, \beta_{ri}, r_{ri})$$

$$\text{then } \Delta u(nT) = p_i e(nT) + q_i r(nT)$$

where $\Delta u(nT)$ is the incremental output contribution of this rule to the fuzzy controller output, and p_i and q_i ($i = 1, \ldots, N$) are design parameters in the rule sequence. For N fuzzy rules, there are $2N$ design parameters in the rule sequence. Zadeh fuzzy logic and operations are used to evaluate the and's in the fuzzy rules, and the combined membership for the rule sequence is

$$\mu_i(\Delta u) = \min(\mu_i(e), \mu_i(r))$$

The generalized defuzzifier is used for defuzzification[26]:

$$\Delta u(nT) = \frac{\sum_{i=1}^{N} \mu_i^{\alpha}(\Delta u) \cdot (p_i e(nT) + q_i r(nT))}{\sum_{i=1}^{N} \mu_i^{\alpha}(\Delta u)} \tag{2}$$

Different defuzzifiers can be obtained by using different α's, where $0 \leq \alpha < \infty$. For instance, the centroid defuzzifier is obtained when $\alpha = 1$.

The new output of the fuzzy controller at $nT + T$ is

$$u(nT + T) = u(nT) + \Delta u(nT)$$

From Equation 2, we have

$$\Delta u(nT) = K_{\mathrm{I}}(e, r)e(nT) + K_{\mathrm{P}}(e, r)r(nT) \tag{3}$$

where

$$K_{\mathrm{P}}(e, r) = \frac{\sum_{i=1}^{N} \mu_i^{\alpha}(\Delta u) \cdot q_i}{\sum_{i=1}^{N} \mu_i^{\alpha}(\Delta u)}$$

$$K_{\mathrm{I}}(e, r) = \frac{\sum_{i=1}^{N} \mu_i^{\alpha}(\Delta u) \cdot p_i}{\sum_{i=1}^{N} \mu_i^{\alpha}(\Delta u)}$$

Recall that the linear PI controller in incremental form is

$$\Delta u(nT) = \bar{K}_{\mathrm{i}} e(nT) + \bar{K}_{\mathrm{p}} r(nT), \tag{4}$$

where $\bar{K}_{\mathrm{p}}$ and $\bar{K}_{\mathrm{i}}$ are proportional gain and integral gain, respectively. Comparing Equation 3 with Equation 4, one sees that the fuzzy controllers are actually nonlinear PI controllers with variable proportional gain $K_{\mathrm{P}}(e, r)$ and variable integral gain $K_{\mathrm{I}}(e, r)$, which are determined by $e(nT)$, $r(nT)$, and the design parameters, α_i, β_i, and γ_i in the input fuzzy sets, and p_i and q_i in the rule function. The foregoing gain relationship between the T-S fuzzy controllers and the linear PI controller can be used to achieve reasonable initial ranges of the design parameters for the fuzzy controllers, as will be shown in detail later.

Table I. Translation from the codons in the DNA chromosome into the amino acids, and then into the parameter values of the T-S fuzzy controllers.

First base	Second base				Third base
	T	C	A	G	
T	Phe (1)	Ser (10)	Tyr (4)	Cys (19)	T
	Phe (1)	Ser (10)	Tyr (4)	Cys (19)	C
	Leu (9)	Ser (10)	Stop (0)	Stop (0)	A
	Leu (9)	Ser (10)	Stop (0)	Trp (20)	G
C	Leu (9)	Pro (7)	His (5)	Arg (11)	T
	Leu (9)	Pro (7)	His (5)	Arg (11)	C
	Leu (9)	Pro (7)	Gln (14)	Arg (11)	A
	Leu (9)	Pro (7)	Gln (14)	Arg (11)	G
A	Ile (2)	Thr (8)	Asn (15)	Ser (10)	T
	Ile (2)	Thr (8)	Asn (15)	Ser (10)	C
	Met (3)	Thr (8)	Lys (16)	Arg (11)	A
	Met (3)	Thr (8)	Lys (16)	Arg (11)	G
G	Val (6)	Ala (12)	Asp (17)	Gly (13)	T
	Val (6)	Ala (12)	Asp (17)	Gly (13)	C
	Val (6)	Ala (12)	Glu (18)	Gly (13)	A
	Val (6)	Ala (12)	Glu (18)	Gly (13)	G

3. SELF-LEARNING OF T-S FUZZY CONTROLLERS VIA THE VDNA-EA

In this section, we develop a new approach to the VDNA-EA and use it to self-learn the T-S fuzzy controllers defined in Section 2. The VDNA-EA uses a new DNA-encoding method inspired from the VDNA. We also discuss how to use the VDNA-encoding method replacing the bit string–encoding method in the GAs to represent the fuzzy control rules. In the VDNA-EA, we use the gene transfer operation to replace the crossover operation, and the bacterial mutation and the frameshift mutation operations replace the point mutation operation in the GAs.

3.1. New VDNA-Encoding Method to Fuzzy Rules

The basic elements of the biological DNA are nucleotides. Because of their different chemical structure, nucleotides can be classified as four bases: adenine (A), guanine (G), cytosine (C), and thymine (T). A triplet code of nucleotide bases specifies the codon, which in turn contains a specific anticodon on transfer RNA (tRNA) and assists subsequent transmission of genetic information in the formation of a specific amino acid. A chromosome consists of combinations of the foregoing four bases and can represent different genes. Although there are 64 possible triplet codes, only 20 amino acids are interpreted by codons. The corresponding relationship between codons and amino acids is as shown in Table I. Note that the same amino acid may be encoded by different codons in the DNA.[25]

From the biological DNA structures, we can exploit an artificial DNA computation model for some practical problems. A single strand of DNA can be likened to a string consisting of a combination of four different symbols: A, G, C,

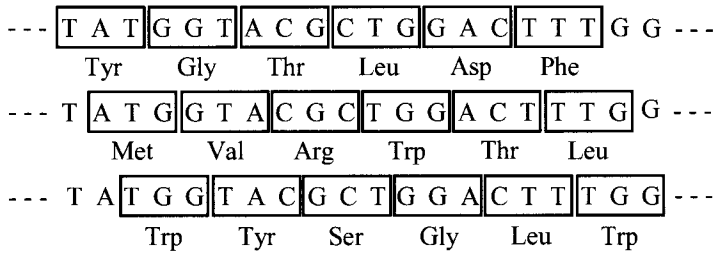

Figure 1. Three genes share a part of the same base sequence in DNA but with a different decoding frame.

and T. This means we have a four-letter alphabet Σ {A, G, C, T} to encode the design parameters of a problem to be solved and to form a DNA chromosome. Based on the DNA model, we can then introduce features of the biological DNA into the EA and develop the new VDNA-EA.

In our new DNA-encoding method, we use gene overlap and frameshift decoding methods of the VDNA. Recent studies show that, in some viruses, one DNA sequence can be encoded into two or three different polynucleotides, and some genes are hidden in other ones. That is to say, two or three different genes can use the same base sequence, even though the encoding frame is different,[25] as shown in Figure 1. Figure 1, shows that three completely different amino acid sequences can be obtained by moving the reading site one base and then another base. Another question is how are these overlapped genes translated into amino acids? The study of biological DNA shows that the translation is completed by using the frameshift decoding method. As such, it can improve the efficiency of DNA encoding. Inspired by the foregoing mechanism, could we use the overlapped genes of DNA and the frameshift decoding frame to encode the design parameters of fuzzy systems, so that the efficiency of DNA encoding could be improved and the population at each generation of the VDNA-EA could be reduced? If yes, the search efficiency for the VDNA-EA to optimization of the fuzzy systems could be improved.

Using the foregoing encoding method, we can obtain the corresponding relationship between a VDNA chromosome and three T-S fuzzy rule sets as shown in Figure 2. In the Figure 2, the DNA chromosome is made of n T-S fuzzy rules, and its length is variable according to the number of the fuzzy rules. A part of DNA codes (totally 24 bases) is corresponding to a fuzzy rule and can be translated into the design parameters in the input fuzzy sets and the rule consequent. The meaning of each amino acid is determined by its position in the sequence of amino acids corresponding to a DNA chromosome. In Figure 2, we also use the frameshift decoding method to translate the codons into the design parameters. Similar to the VDNA, a chromosome has three different decoding frames.[21] Based on Figure 2 and Table I, reading from the top of a chromosome, it can be translated into the design parameters to form a fuzzy rule set. Then, reading from the second base and third base of the chromosome, respectively, we also can obtain another two groups of fuzzy rules. Certainly, to satisfy the foregoing requirements, we should add two

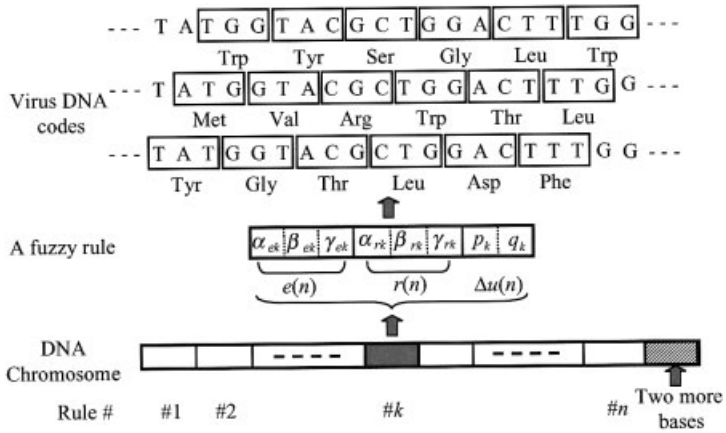

Figure 2. The frameshift decoding method of a DNA chromosome corresponding to the three groups of fuzzy rule sets. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

more bases when we encode a DNA chromosome. In our encoding method with overlapped genes and the frameshift decoding method, a DNA chromosome can be translated into three fuzzy rule sets, so it can significantly reduce the population of each generation.

It should be noted that the VDNA-encoding method to the T-S fuzzy rule sets supplies a high degree of freedom for the VDNA-EA, which can simultaneously define the variables to be used in the rules, the rules themselves, the parameters in the input fuzzy sets and the rule consequent, and the number of the rules in the rule base. However, the encoding method has some drawbacks because of the lack of uniformity of membership functions. Every rule has a different set of membership functions, and, consequently, there is no bonding between the membership functions for a variable. We use such a DNA-encoding method because the DNA mutation operations can be implemented on the DNA chromosome. As such, the adaptive addition or deletion of a fuzzy rule can be implemented easily.

Based on the foregoing DNA-encoding method, we developed the genetic operators in the VDNA-EA.

3.2. Evolutionary Operation

3.2.1. Gene Transfer Operation

The procedures of the gene transfer operation is as follows:

(1) Sort the population and divide it in two halves. The half with higher fitness is called the superior half, and the other half is called the inferior half.
(2) Choose randomly one DNA chromosome from the superior half, called the source DNA chromosome, and another DNA chromosome from the inferior half, called the destination DNA chromosome.

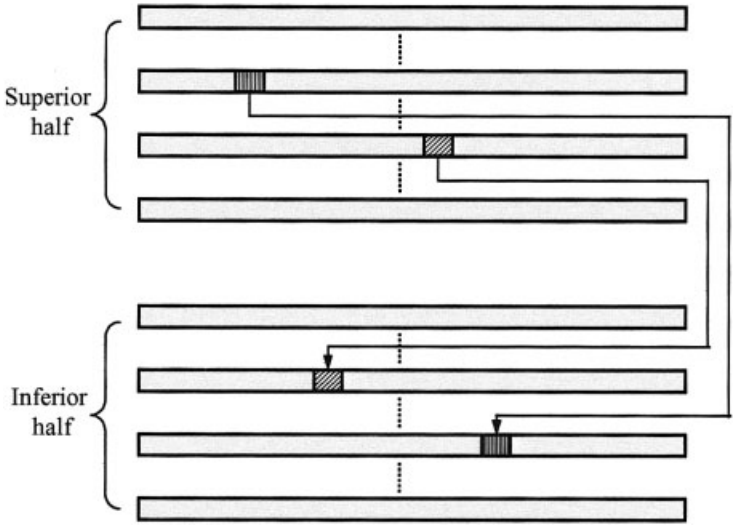

Figure 3. The gene transfer operation. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

(3) According to a given criterion, choose a good part from the source DNA chromosome and transfer it to the destination DNA chromosome. A good part can be a fuzzy rule or a group of rules with a high degree of activation value.
(4) Repeat Steps 1–3 for M times in one generation, where M is the number of infections per generation.

The process for the gene transfer operation is shown in Figure 3. The gene transfer operation is expected to spread rapidly the good parts (corresponding to the good fuzzy rules) of the DNA chromosome with superior fitness to the DNA chromosome with inferior fitness. By doing so, the overall search process should proceed more efficiently, because the operators execute more frequently on better rules, leading to the rapid construction of fuzzy systems that fulfill the system requirements.

3.2.2. Mutation Operation

In VDNA-EA, two mutation methods are used. One is the bacterial mutation and the other is the frameshift mutation.

(1) Bacterial mutation. Suppose there are p parts in a DNA chromosome, each of which is respective to a fuzzy rule. First, the best DNA chromosome is chosen from the m DNA chromosomes. Then, the ith part of the selected DNA chromosome is randomly chosen and transferred to a corresponding part of the rest $m - 1$ DNA chromosome. The foregoing operation is shown in Figure 4. The bacterial mutation operation is always applied to all the m chromosomes in the population.
(2) Frameshift mutation. In the biological DNA chromosome, there are two frameshift mutations. One is the deletion mutation, in which one or more base pairs are lost. Deletion mutation is caused by enzyme operation. The other is insertion mutation, in

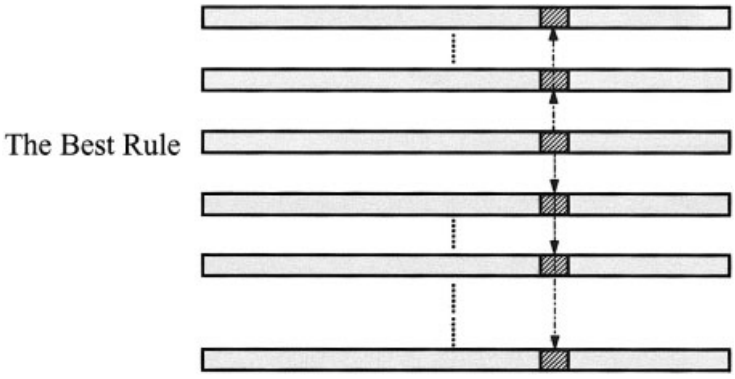

Figure 4. The bacterial mutation operation. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

which one or more base-pairs are inserted into the sequence. Insertion mutation is caused by a virus operation. Accordingly, we have developed two frameshift mutations in VDNA-EA: deletion and insertion. These two frameshift mutations are selectable according to the evolution process of the VDNA-EA. Figure 5 shows an example of deletion operation in which the bases between the start codon TAG and end codon TAT are lost; as such, a fuzzy rule is deleted. Figure 6 shows an example of insertion operation in which a base sequence moves into the chromosome and a fuzzy rule is added into the fuzzy system; i.e., the deletion and the insert mutation operations could be used to add or delete some bases in the DNA chromosome. The operated bases can be one or several fuzzy rules. Generally, if the contributions of two fuzzy rules to the fuzzy controller are near, one fuzzy rule can be deleted by using the deletion operation. If the control performance of the fuzzy system cannot be improved anymore by using the current fuzzy rule sets, we should consider adding a (or several) fuzzy rule(s) by using the insertion operation. As such, the fuzzy rules could be automatically added or deleted to obtain the more proper fuzzy rule sets.

3.3. Performance Evaluation

In biological DNA, a triplet code of nucleotide bases specifies the codon and transmits genetic information in the formation of a specific amino acid. To mimic the biological DNA,[25] we also use the same translation process from the nucleotide bases to amino acids. Because there are 20 amino acids and one stop codon, we relate them to the parameter values [0, 20], as shown in Table I. We then relate the parameter

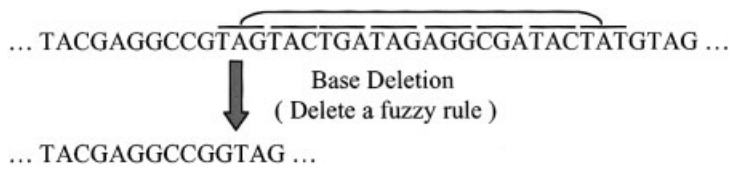

Figure 5. An example of frameshift mutation: deletion operation. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

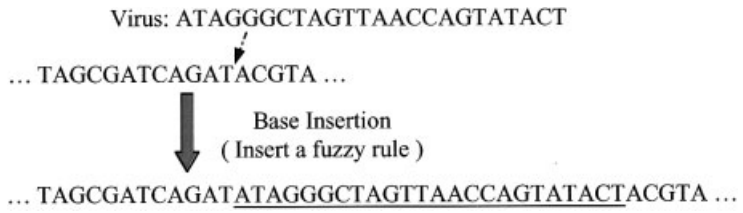

Figure 6. Another example of frameshift mutation: insertion operation. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

values [0, 20] to the ranges of the design parameters of the fuzzy controllers. As such, the transforming relationship between the amino acids and the ranges of the design parameters of the fuzzy controllers can be established. The reason we relate the 64 codons to 20 amino acids and one stop, as such mapped into [0, 20], is that we can use the features of the genetic codes in biological DNA. In biological DNA, the codons with the amino acids of similar features are distributed on the neighbor sites, which have little effect on the property of the potential when the bases mutate.[25] We use this feature of biological DNA to encode the design parameters in our fuzzy systems and hope that the base change in DNA will not affect the design parameters too frequently, to keep the performance of the fuzzy systems relatively stable. From Table I, we know that Leu, Ser, or Arg has six codons; Val, Pro, Thr, Ala, or Gly has four codons; and the other have two codons except for Trp, which only has one codon. As such, the middle value (i.e., 9–11) in [0, 20] corresponds to the Leu, Ser, and Arg with six codons, which means that the design parameters of the T-S fuzzy systems use these values with the highest possibility. On the other hand, the values near both ends (e.g., 1–5 and 14–19) correspond to the amino acids with two codons, which means that the design parameters of the T-S fuzzy systems use these values with lower possibility. In the same way, we can determine the values of the other amino acids, as shown in Table I. In this way, each amino acid can be interpreted as a value in the range of design parameters of the T-S fuzzy controllers.

It should be noted that the translation process in Table I imitates the translation process from DNA to protein. Also, it is a basic framework for translating the codons into the amino acids and then into the design parameters. The range of the design parameters can be adjusted with respect to [0, 20] according to different design problems. For a particular application, one may transfer a value in the range of [0, 20] into the proper range of the design parameters. After the translation, the fuzzy controller with these design parameters can be used and the fitness function can be computed. The choice of the fitness function or the performance index is dependent on the types of responses that are desired for the particular system. Because the central objective of fuzzy control is to minimize the error between the actual system response and the set point, the fitness function we adopt is

$$f_{\text{fit}} = 1 \Big/ \left(C + \sum_{k=1}^{n} (e^2(k) + r^2(k)) \right)$$

where C is a constant.

3.4. Design Procedures

When we design the parameters in linear sequence to the T-S fuzzy systems, we should first determine their ranges. This can be done based on the gain relationships between the PI controller and the T-S fuzzy controller proved in Section 2. We first design a PI controller by using the trial-and-error tuning method and obtain the relative satisfactory gains of the PI controller to control a system. We then choose the proper ranges of the design parameters in rule consequent based on the values of the gains of the PI controller.

Then, we use the VDNA-EA to design automatically the T-S fuzzy controllers. Initially, we use a small number of (e.g., two) input fuzzy sets to fuzzify $e(nT)$ and $r(nT)$. Next, we adopt the gene transfer operation and bacterial mutation operation in the VDNA-EA to evolve the fuzzy systems. After evolution for certain generations, e.g., 10 generations, if the contributions of two fuzzy rules to the fuzzy controller are near, then one fuzzy rule is deleted by using the deletion operation. If the control performance of the fuzzy system cannot be improved anymore by using the current fuzzy rule sets, we consider adding a (or several) fuzzy rule(s) by using the insertion operation. As such, the fuzzy rules could be automatically added or deleted to obtain the more proper fuzzy rule sets. After the convergence of VDNA-EA, the structure of the fuzzy controller can be built, and the design parameters in the input fuzzy sets and rule sequence can be obtained.

3.5. Advantages in the DNA-EA

Some studies have shown that both fuzzy rules and tuning parameters are adjusted by the GAs.[5–7] However, the bit ring (0–1's–encoding method often is adopted by GA researchers. The configuration of the chromosome also is different from the one used in the VDNA-EA. The genetic operations are also different. Previously, we have proven that the DNA-GA are more efficient than the GA in solving some problems.[21] The DNA-GA are superior to the GA in decreasing the complexity of encoding and searching the solution space effectively. In this study, we emphasize the possibility of combining the DNA computing with the EA. We should point out that there are several advantages in the VDNA-EA:

(1) In the VDNA-EA, we use the VDNA-encoding method that is suitable for complex knowledge representation. We have a four-letter alphabet $\Sigma\{A, G, C, T\}$ to encode information, whereas in the GAs only two digits 0 and 1 are used for the same purpose. In addition, we use the frameshift encoding method to encode the fuzzy rules. Because of the new encoding method, the encoding length of the chromosome can be shortened greatly and the efficiency of encoding is improved greatly.

(2) In the DNA-encoded chromosome, we can introduce easily features of the biological DNA into the EAs and develop some new genetic operations. For example, the length of the DNA chromosome is variable and it is easy to insert and delete parts of it by using the new frameshift (i.e., insertion and deletion) mutation operations. They can be used to add (or delete) a (or several) fuzzy rule(s) to (or from) a fuzzy controller. As such, the structure of the fuzzy controller is automatically constructed. Similarly, some other genetic operations at the gene level could be introduced into DNA-based algorithms to enrich the DNA-EAs.

(3) Because of the DNA transduction process, we develop and introduce the gene transfer operation into the VDNA-EA. The gene transfer operation can spread good rules in the population. The good portions of chromosomes with high fitnesses are directly transferred to the individuals with lower fitnesses and, thus, induce the improvement of the overall performance of the population. By doing so, the overall search process should proceed more efficiently, because the operators execute more frequently on better rules, leading to the rapid construction of fuzzy systems that fulfill the requirements. Also, the bacterial mutation operation affected by bacterial genetic process is efficient in the optimization of local portions of chromosomes.

(4) During the translation process from the DNA chromosome to the design parameters of the fuzzy systems, we use the translation process of the biological DNA, i.e., from 64 triplet codes to 20 amino acids. In addition, we adopt the features of DNA, e.g., the codons with the amino acids of similar features are distributed on the neighbor sites. The method could speed up the search process in optimization and quickly find the expected values of the designed parameters.

(5) In the near future, with the development of the DNA computer, DNA-based soft computing[15,16] will have many applications in many scientific and experimental problems. Based on Chen's work,[24] we also may be able to fulfill the laboratory implementation of the VDNA-EA using the biological technology, which will be useful for the DNA computer and the future DNA intelligent computer.

4. SIMULATION STUDY

Although fuzzy controllers are nonlinear controllers known to be capable of regulating nonlinear systems, it is rather difficult to construct the fuzzy rules and tune the design parameters in the input fuzzy sets and the rule sequence because of many design parameters. To examine the effectiveness of the VDNA-EA, we now use it to design automatically the T-S fuzzy controller and use it to control a nonlinear system. The VDNA-EA is used to build the fuzzy controller and optimize the design parameters in the input fuzzy sets and the rule sequence simultaneously. The nonlinear systems is

$$\begin{cases} y(k) = 0.8y(k-1) - 0.6y(k-2) + 0.4x(k-1) + 0.2x(k-2) \\ x(k-1) = u(k-1) + 0.3u^2(k-1) \end{cases} \tag{5}$$

Initially, the T-S fuzzy controller uses two generalized membership function-type input fuzzy sets to fuzzify $e(nT)$ and $r(nT)$, respectively, which means the T-S fuzzy controller only uses two fuzzy rules at the beginning. We use 50 bases in a DNA chromosome, every three of which are interpreted as a design parameter and hence 48 bases for the two fuzzy rules. The remaining two bases are for the frameshift decoding method. The number of population m is 30. The number of infections per generation M is 20.

Before computer simulations, we find a proper initial range for each design parameter in the rule consequent according to the gain relationship between the T-S fuzzy controller and linear PI controller derived in Section 2. First, we design a PI controller by using the trial-and-error tuning method. The good gains of the PI controller are $\bar{K}_P = 0.1$ and $\bar{K}_i = 0.1$. Then, according to the gain relationship between the T-S fuzzy controller and the linear PI controller, we can achieve reasonable initial value ranges of the design parameters for the T-S fuzzy controller. We roughly choose the proper ranges of the design parameters in the rule

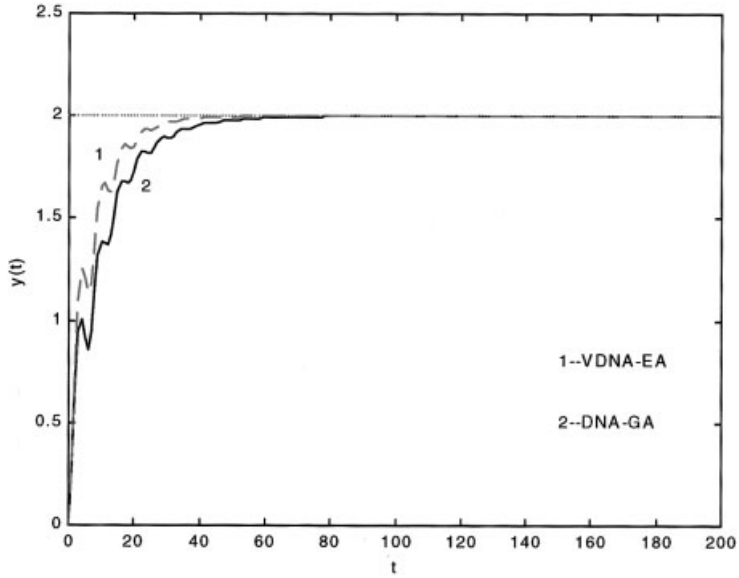

Figure 7. The performance comparison of the T-S fuzzy controllers designed by the VDNA-EA and the DNA-GA in the control of the nonlinear system.[21] [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

consequent based on the values of $\bar{K}_p$ and $\bar{K}_i$, where $[p_i^{min}, p_i^{max}] = [0, 0.2]$ and $[q_i^{min}, q_i^{max}] = [0, 0.2]$. Furthermore, the transforming relationship between the parameter values [0, 20] and $[p_i^{min}, p_i^{max}]$ (or $[q_i^{min}, q_i^{max}]$) can be established.

During the computer simulation, the gene transfer operation and the bacterial mutation operation are used in the VDNA-EA at each generation. After the evolution of a certain generations, e.g., 10 generations, when the satisfactory simulation results cannot be achieved by using the current fuzzy rule sets, we consider adding or deleting one or more fuzzy rules by using the insertion or the deletion operation. Many computer simulations show that after convergence of the VDNA-EA, we can always find a group of parameter values for the input fuzzy sets and the rule consequent that obtain satisfactory control performance. One typical example of the control performance of the T-S fuzzy controller designed by the VDNA-EA is shown in Figure 7. The final number of fuzzy rules is five, and the values of the design parameters of the T-S fuzzy controller are shown in Table II.

To further examine the suitability and effectiveness of the VDNA-EA, another algorithm, the DNA-GA we previously used,[21] is performed in the experiments. The DNA-GA uses the DNA-encoding method with two-point crossover and point mutation genetic operations. The compared performances of the T-S fuzzy controllers optimized by the two algorithms are also given in Figure 7. From Figure 7, we know that the control performances of the T-S fuzzy controllers designed by the two algorithms are comparable in controlling the system.[21] However, the fuzzy controller optimized by the DNA-GA uses 25 fuzzy rules,

Table II. The values of the design parameters of the T-S fuzzy controller designed by the VDNA-EA in the control of the nonlinear system (Equation 5).

Rule no.	$\mu_i(e)$			$\mu_i(r)$			Rule consequent	
	α_{ei}	β_{ei}	γ_{ei}	α_{ri}	β_{ri}	γ_{ri}	p_i	q_i
1	−1.0	−0.15	2.89	−3.50	−1.20	9.56	0.10	0.12
2	−0.5	0	7.33	4.00	−0.15	5.56	0.14	0.02
3	0	0	6.00	2.50	−1.20	2.44	0.18	0
4	−4.0	0.30	8.22	−3.00	1.05	3.33	0.008	0.14
5	−0.5	0.75	5.56	−3.00	−0.75	3.33	0.10	0

whereas the one designed automatically by the VDNA-EA only uses 5 rules. The supposition made to explain this result is that the gene transfer operation in the VDNA-EA is spreading good rules in the population and, thus, inducing the improvement of the overall performance of the population. In addition, the bacterial mutation is efficient in the optimization of local portions of chromosomes.

V. CONCLUSIONS

We proposed a new VDNA-EA that uses the VDNA-encoding method and genetic operators affected by the biological DNA and microbial evolution phenomenon. The VDNA-EA can automatically design the T-S fuzzy controller and optimize the design parameters in the input fuzzy sets and the rule consequent simultaneously. The computer simulation was provided to show the effectiveness of our new method. The VDNA-EA also can be used to optimize the design parameters of other types of fuzzy controllers such as other types of input fuzzy sets, fuzzy logic and operators, and defuzzifiers.

Further development of other DNA-based technologies in soft computing, such as DNA-based immune algorithms and DNA-based neural networks, should be explored. In addition, more operations at the gene level should be introduced into the DNA-based algorithms to enrich the evolutionary methods. The DNA-based learning algorithms have potential advantages in many complex practical problems. We hope this study will be a step toward future DNA computing with application to the intelligent system.

Acknowledgments

The work in this research was supported in part by the National Natural Science Foundation of China (60004006, 69874038) and the Shanghai Shuguang Project (99SG20) from the Shanghai Education Committee, China.

References

1. Yen J, Langari R, Zadeh LA, editors. Industrial applications of fuzzy logic and intelligent systems. Piscataway, NJ: IEEE Press; 1995.
2. Ying H. Fuzzy control and modeling: Analytical foundations and applications. Piscataway, NJ: IEEE Press; 2000.

3. Ding Y-S, Ying H, Shao S-H. Necessary conditions on minimal system configuration for general MISO fuzzy systems as universal approximators. IEEE Trans Syst Man Cybern 2000;30(6):857–864.
4. Ying H, Ding Y-S, Li S-K, Shao S-H. Comparison of necessary conditions for typical Takagi-Sugeno and Mamdani fuzzy systems as universal approximations. IEEE Trans Syst Man Cybern 1999;29(6):508–514.
5. Berenji HR. Learning and tuning fuzzy logic controllers through reinforcements. IEEE Trans Neural Netw 1992;3:724–740.
6. Jang JR. Self-learning fuzzy controllers based on temporal back propagation. IEEE Trans Neural Netw 1992;3:714–721.
7. Lim MH, Rahardja S, Gwee BH. A GA paradigm for learning fuzzy rules. Fuzzy Sets Syst 1996;82:177–186.
8. Matsushita S, Furuhashi T, Tsutsui H, Uchikawa Y. Efficient search for fuzzy models using genetic algorithm. Inform Sci 1998;110:41–50.
9. Shi Y-H, Eberhart R, Chen Y-B. Implementation of evolutionary fuzzy systems. IEEE Trans Fuzzy Syst 1999;7(2):109–119.
10. Wang C-H, Hong T-P, Tseng S-S. Integrating fuzzy knowledge by genetic algorithms. IEEE Trans Evol Comput 1998;2(4):138–149.
11. Adleman LM. Molecular computation of solutions to combinatorial problems. Science 1994;266(5187):1021–1023.
12. Kari L. DNA computing: Arrival of biological mathematics. Math Intell 1997;19(2):9–22.
13. Lipton RJ. DNA solution of hard computational problems. Science 1995;268(5210):542–545.
14. Ren L-H, Ding Y-S, Shao S-H. The study of DNA computing: Current state and future directions. Inform Control 1999;28(4):241–248. (In Chinese)
15. Ren L-H, Ding Y-S, Shao S-H. DNA bio-soft computing and its application to intelligent control systems. J Shanghai Jiaotong University 1999;E-4(2):95–101.
16. Ren, L-H. Artificial DNA computation models-based intelligent systems and their applications. PhD dissertation, Dong Hua University, China, 2000.
17. Deaton R, Murphy RC, Rose JA, Garzon M, Franceschetti DR, Stevens SE Jr. A DNA based implementation of an evolutionary search for good encodings for DNA computation. In: Proc 1997 IEEE Int Conf Evolutionary Computation. Indianapolis, IN, April 13–16, 1997. Piscataway, NJ: IEEE; 1997. pp 267–271.
18. Doi H, Furusawa M. Evolution is promoted by asymmetrical mutations in DNA replication-genetic algorithm with double-stranded DNA. Fujitsu Sci Tech J 1996;32(2)248–255.
19. Farhat NH, Hernandez EDM. Logistic networks with DNA-like encoding and interactions. In: Proc Int Workshop on Artificial Neural Networks. Malaga-Torremolinos, Spain, June 7–9, 1995. Berlin: Springer-Verlag; 1995. pp 215–222.
20. Russo MF, Huff AC, Heckler CE, Evans AC. An improved DNA encoding scheme for neural network modeling. In: 1994 Int Neural Network Society Annual Meeting, San Diego, CA, June 5–9, 1994; pp I/354–I/359.
21. Ding Y-S, Ren L-H. DNA genetic algorithm for design of the generalized membership-type Takagi-Sugeno fuzzy control system. In: Proc 2000 IEEE Int Conf on Systems, Man, and Cybernetics, Nashville, TN, Oct 2000. Piscataway, NJ: IEEE; 2000. pp 3862–3867.
22. Yoshikawa T, Furuhashi T, Uchikawa Y. DNA coding method and a mechanism of development for acquisition of fuzzy control rules. In: Proc 5th IEEE Int Conf on Fuzzy Systems. New Orleans, LA, September 8–11, 1996. Piscataway, NJ: IEEE; 1996. pp 2194–2200.

23. Yoshikawa T, Furuhashi T, Uchikawa Y. The effects of combination of DNA coding method with Pseudo-Bacterial GA. In: Proc 1997 IEEE Int Conf Evolutionary Computation. Indianapolis, IN, April 13–16, 1997. Piscataway, NJ: IEEE; 1997. pp 285–290.
24. Chen J-H, Antipov E, Lemieux B, Cedeno W, Wood DH. DNA computing implementing genetic algorithms. In: DIMACS Workshop on Evolution as Computation. Princeton University, January 11, 1999. Piscataway, NJ: 1999. pp 39–49. http://www.cis.udel.edu/~wood/papers/DIMACS_99.ps
25. Yan L-F, Zhang Y-L. editors. Molecular biology. Beijing: The Press of China Agriculture University, 1997.
26. Filev DP, Yager RR. A generalized defuzzification method via BAD distributions. Int J Intell Syst 1991;6:687–697.