ORIGINAL RESEARCH

Two-Strategy reinforcement group cooperation based symbiotic evolution for TSK-type fuzzy controller design

Sheng-Fuu Lin[1], Jyun-Wei Chang[1], Yu-Bi Hong[1], Yung-Chi Hsu[2]

1. Department of Electrical Engineering, National Chiao Tung University, Hsinchu, Taiwan, R.O.C. 2. 3C Software Engineering Division, Quanta Computer Inc., Taoyuan, Taiwan, R.O.C.

Correspondence: Sheng-Fuu Lin. Address: 1001, Ta Hsueh Road, Hsinchu, Taiwan 300, R.O.C. Telephone: 886-3-571-2121 ext. 54365. Email: sflin@mail.nctu.edu.tw

Received: April 18, 2012 **Accepted:** June 28, 2012 **Published:** September 1, 2012
DOI: 10.5430/air.v1n1p1 **URL:** http://dx.doi.org/10.5430/air.v1n1p1

Abstract

This paper proposes a TSK-type fuzzy controller (TFC) with a two-strategy reinforcement group cooperation based symbiotic evolution (TSR-GCSE) for solving various control problems. The TSR-GCSE proposes the two-strategy reinforcement (TSR) signal designed to improve the performance of the traditional reinforcement signal designed. Moreover, the TSR-GCSE is different from the traditional symbiotic evolution; with each population in the TSR-GCSE method is divided to several groups. Each group represents a set of the chromosomes that belongs to a fuzzy rule and can cooperate with other groups to generation the better chromosomes by using elites-base compensation crossover strategy (ECCS). The illustrative examples show that the proposed method has the better time steps and CPU times than other existing methods.

Key words

Fuzzy controller, Symbiotic evolution, Control, Reinforcement learning

1 Introduction

In recent years, the concept of the fuzzy logic or artificial neural networks for control problems has grown into a popular research area [1-3]. The reason is that classical control theory usually requires a mathematical model for designing controllers. The inaccuracy of mathematical modeling of plants usually degrades the performance of the controllers, especially for nonlinear and complex control problem [4, 5]. Fuzzy logic has the ability to express the ambiguity of human thinking and translate expert knowledge into computable numerical data.

A fuzzy system consists of a set of fuzzy IF-THEN rules that describes the input-output mapping relationship of the networks. Obviously, it is difficult for human experts to examine all the input-output data from a complex system to find proper rules for a fuzzy system. To cope with this difficulty, several approaches that are used to generate the fuzzy IF-THEN rules from numerical data have been proposed [1-3]. These methods were developed for supervised learning; i.e., the correct "target" output values are given for each input pattern to guide the learning of the network. However, most of the supervised learning algorithms for neural fuzzy networks require precise training data to tune the networks for various

applications. For some real world applications, precise training data are usually difficult and expensive, if not impossible, to obtain. For this reason, there has been a growing interest in reinforcement learning algorithms for use in neural controller [1, 3- 9] or fuzzy controller design [7, 10-14]. Lin proposed a temporal difference and genetic algorithm based reinforcement learning method and applies it to the control of a real magnetic bearing system [13]. Berenji used the reinforcements to learning and tuning a fuzzy logic controller from a dynamic system [14]. Lin proposed a reinforcement fuzzy adaptive learning control network. This is carried out by integrating two fuzzy adaptive learning control networks. It reduces the combinatorial demands placed by the standard methods for adaptive linearization of a system [7]. Hinojosa present a novel reinforcement learning approach to combine the universal-function-approximation capability of fuzzy system with consideration of probability distributions over possible consequences of an action, and enhance this algorithm by the introduction of a probability measure into the learning structure [12]. Tzafestas' algorithm been presented for robotic manipulators is based on the principled of sliding-mode control and fuzzy logic. This can ease the noise on the metal surface [35]. Vengerov present an algorithm to overcome some power control problem. And the experiment result shows the algorithm can converge deterministically to a neighborhood of optimal parameter values, as opposed to a noisy stochastic convergence of earlier algorithm [10].

For many real problems, training data are usually difficult to obtain. Hence, reinforcement learning is a better solution than supervised learning. Unlike supervised learning problems, in which the correct "target" output values are given for each input pattern, reinforcement learning problems have very simple "evaluative" or "critical" information, rather than "instructive" information, available for learning. In the extreme case, there is only a single bit of information to indicate whether the output is right or wrong. In the reinforcement learning, the most well-known algorithm is Barto and his colleagues' actor-critic architecture [8], which consists of a control network and a critic network. The proposed new reinforcement architecture [16] has a structure in which the only available feedback is a reinforcement signal that notifies the model only when a failure occurs. However, the Barto's architecture is complicated and is not easy to implement. Hence, several researches proposed the time-step reinforcement architecture to improving Barto's architecture [9, 13]. In time-step reinforcement architecture, the only available feedback is a reinforcement signal that notifies the model only when a failure occurs. An accumulator accumulates the number of time steps before a failure occurs. Although time-step reinforcement architecture becomes more simple and easier to implement than the Barto's architecture, there is still a problem that such reinforcement architecture only knows how long the system works as a "success". It does not know how well the controller controls the system.

In the design of a fuzzy controller, adjusting the required parameters is important. To do this, back-propagation (BP) training was widely used [1, 7]. It is a powerful training technique that can be applied to networks with a forward structure. Since the steepest descent technique is used in BP training to minimize the error function, the algorithms may reach the local minima very fast and never find the global solution. For solving these problems, recently, several evolutionary algorithms, such as genetic algorithm (GA) [15-17], and evolution strategies [18], have been proposed. They are parallel and global search techniques. Because they simultaneously evaluate many points in the search space, they are more likely to converge toward the global solution. For this reason, an evolutionary method using for training the fuzzy model has become an important field.

The evolutionary fuzzy model generates a fuzzy system automatically by incorporating evolutionary learning procedures [13, 19, 20, 22- 26]. One of the most well-known evolutionary learning procedure is the genetic algorithms (GAs). Several genetic fuzzy models have been proposed [13, 19, 20, 22]. Karr [19] applied GAs to the design of the membership functions of a fuzzy controller, with the fuzzy rule set assigned in advance. Since the membership functions and rule sets are co-dependent, simultaneous design of these two approaches will be a more appropriate methodology. Based on this concept, many researchers have applied GAs to optimize both the parameters of the membership functions and the rule sets [27]. Lin and Jou [13] proposed GA-based fuzzy reinforcement learning to control magnetic bearing systems. Juang et al. [15] proposed genetic reinforcement learning in the design of fuzzy controllers. The GA adopted [15] was based upon traditional symbiotic evolution which, when applied to fuzzy controller design, complemented the local mapping property

of a fuzzy rule. Tang [23] proposed a hierarchical genetic algorithm. The hierarchical genetic algorithm enables the optimization of the fuzzy system design for a particular application. J. K. Koza [16] proposed the CQGAF to simultaneously design the number of fuzzy rules and free parameters in a fuzzy system. Lin [25] proposed a sequential-search based dynamic evolution (SSDE) to let better chromosomes will be initially generated while better mutation points will be determined for performing dynamic-mutation. Lin proposed a hybrid evolution learning algorithm (HELA). The HELA combines the compact genetic algorithm (CGA) and the modified variable-length genetic algorithm, performs the structure/parameter learning for constructing the network dynamically. However, these approaches may encounter one or more of the following major problems: 1) all the fuzzy rules are encoded into one chromosome; 2) the population cannot evaluate each fuzzy rule locally.

In this paper, a TSK-type fuzzy controller with a two-strategy reinforcement group cooperation based symbiotic evolution (TSR-GCSE) is proposed for solving the problems about reinforcement signal and evolutionary fuzzy model designed that mention above. The TSR-GCSE consists of the two-strategy reinforcement (TSR) signal design and the group cooperation based symbiotic evolution (GCSE). In the proposed TSR, the two-strategy is defined to design the reinforcement signal. The TSR feedback signal takes the form of an accumulator determined by two different strategies named the judgment and the evaluation strategy. In the judgment strategy, the signal is defined according to how long the experiment is still a "success". In the evaluation strategy, the signal is defined according to measures how well the TFC controls the system in the predefined range. The accumulator is used as the fitness values of the proposed TSR-GCSE method. In the proposed GCSE, each chromosome represents only one fuzzy rule. The TSR-GCSE method promotes both cooperation and specialization, which ensures diversity and prevents a population from converging to suboptimal solutions. *n* chromosomes selected from several groups construct an complete n-rules fuzzy system. In order to make the groups that perform well can be corporate to generate better generation, the elite-based compensatory of crossover strategy (ECCS) is proposed. In the ECCS, each group will be corporate to perform the crossover step. Therefore, the better chromosomes of each group will be selected to perform crossover in the next generation of each group. The advantages of the proposed TSR-GCSE are summarized as follows: 1) The TSR-GCSE uses group-based population to evaluate the fuzzy rule locally. 2) The TSR-GCSE uses the ECCS method to let the better solutions form different groups can cooperate to generate better solutions in the next generation. 3) The TSR-GCSE uses the TSR to improve the traditional reinforcement signal design.

This paper is organized as follows. Section 2 introduces the TSK-type fuzzy controller (TFC). The proposed compensation of group cooperation based symbiotic evolution (GCSE) is described in Section 3. Section 4 introduces the proposed two-strategy reinforcement GCSE using for constructing the TFC model. Section 5 presents the simulation results. The conclusions are summarized in the last section.

2 Review a TSK-Type fuzzy controller

A Takagi-Sugeno-Kang (TSK) type fuzzy controller (TFC) employs different implication and aggregation methods when compared with the standard Mamdani controller. Instead of using fuzzy sets, the conclusion part of a rule is a linear combination of the crisp inputs.

$$\text{IF } x_1 \text{ is } A_{1j}\ (m_{1j},\ \sigma_{1j}\ ) \text{ and } x_2 \text{ is } A_{2j}(m_{2j},\ \sigma_{2j}\ ) \text{ and } \ldots \text{and } x_n \text{ is } A_{nj}\ (m_{nj},\ \sigma_{nj})$$
$$\text{THEN y'}=w_{0j}+w_{1j}x_1+\ldots+w_{nj}x_n \tag{1}$$

Since the consequence of a rule is crisp, the defuzzification step becomes obsolete in the TSK inference scheme. The control output is computed as the weighted average of the crisp rule outputs, which is computationally less expensive then calculating the center of gravity.

In this paper, a TSK-type fuzzy system with a TSR-GCSE is used to solve nonlinear control problems. The structure of TFC is shown in Fig. 1, where *n* and *M* are, respectively, the number of input dimensions and the number of rules. It is a five-layer network structure. The functions of the nodes in each layer are described as follows:

Layer 1 (Input Node):

No function is performed in this layer. The node only transmits input values to layer 2. That is

$$u_i^{(1)} = x_i \tag{2}$$

Layer 2 (Membership Function Node):

Nodes in this layer correspond to one linguistic label of the input variables in layer 1; that is, the membership value specifying the degree to which an input value belongs to a fuzzy set is calculated in this layer. For an external input x_i , the following Gaussian membership function is used:

$$u_{ij}^{(2)} = \exp\left(- \frac{\left[u_i^{(1)} - m_{ij}\right]^2}{\sigma_{ij}^2} \right) \tag{3}$$

where m_{ij} and σ_{ij} are, respectively, the center and the width of the Gaussian membership function of the *j*th term of the *i*th input variable x_i .

Layer 3 (Rule Node):

The output of each node in this layer is determined by the fuzzy AND operation. Here, the product operation is utilized to determine the firing strength of each rule. The function of each rule is

$$u_j^{(3)} = \prod_i u_{ij}^{(2)} \tag{4}$$

Layer 4 (Consequent Node):

Nodes in this layer are called consequent nodes. The input to a node in layer 4 is the output delivered from layer 3, and the other inputs are the input variables from layer 1 as depicted in Figure 1. For this kind of node, we have

$$u_j^{(4)} = u_j^{(3)} (w_{0j} + \sum_{i=1}^{n} w_{ij} x_i) \tag{5}$$

where the summation is over all the inputs and where w_{ij} are the corresponding parameters of the consequent part.

Layer 5 (Output Node):

Each node in this layer corresponds to one output variable. The node integrates all the actions recommended by layers 3 and 4 and acts as a defuzzifier with

 ISSN 1927-6974 E-ISSN 1927-6982

$$y = u^{(5)} = \frac{\sum_{j=1}^{M} u_j^{(4)}}{\sum_{j=1}^{M} u_j^{(3)}} = \frac{\sum_{j=1}^{M} u_j^{(3)} (w_{0j} + \sum_{i=1}^{M} w_{ij} x_i)}{\sum_{j=1}^{M_k} u_j^{(3)}} \tag{6}$$

where M is the number of fuzzy rule.

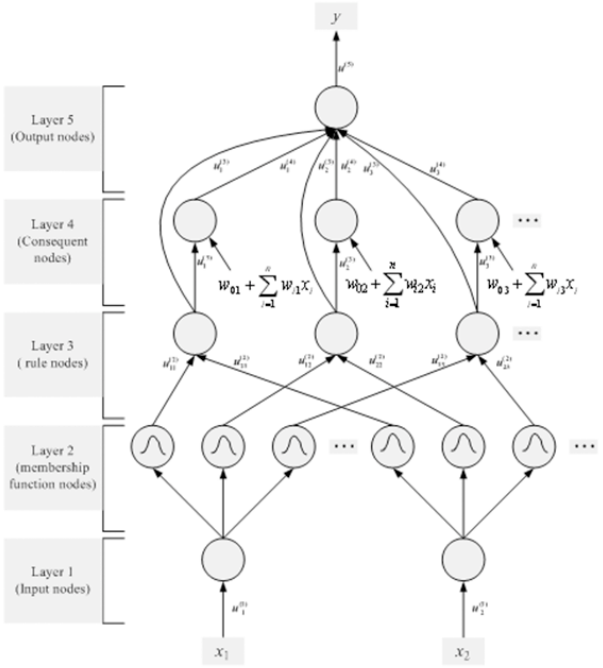

Figure 1. Structure of the proposed TFC

3 A group cooperation based symbiotic evolution

This section will introduce the proposed group cooperation based symbiotic evolution (GCSE) method. Recently, many researches try to enhance the traditional GAs have been made [29, 30-32]. One category of them tries to modify the structure of a population. Examples in this category include the distributed GA [31], the cellular GA [31], and the symbiotic GA [32].

This study proposes the group cooperation based symbiotic evolution (GCSE) to improve the symbiotic GA [31]. In the proposed GCSE, the algorithm is developed from symbiotic evolution. The idea of symbiotic evolution was first proposed in an implicit fitness-sharing algorithm that is used in an immune system mode [33]. The authors developed artificial antibodies to identify artificial antigens. Because each antibody can match only one antigen, a different population of antibodies is required to effectively defend against a variety of antigens. As shown in the research [24, 32], partial solutions can be characterized as specializations. The specialization property ensures diversity, which prevents a population from converging to suboptimal solutions. A single partial solution cannot "take over" a population since there must exists other specializations. Unlike the standard evolutionary approach which always causes a given population to converge, hopefully at the global optimum, but often at a local one, the symbiotic evolution find solutions in different, unconverted populations[22, 32]. The GCSE method divides the population into several groups to enhances its ability of finding global optimum. Each group represents a set of the chromosomes that belong to a fuzzy rule.

In the proposed GCSE, the structure of the population consists of several groups. The structure of the chromosome in the GCSE is shown in Figure 2.

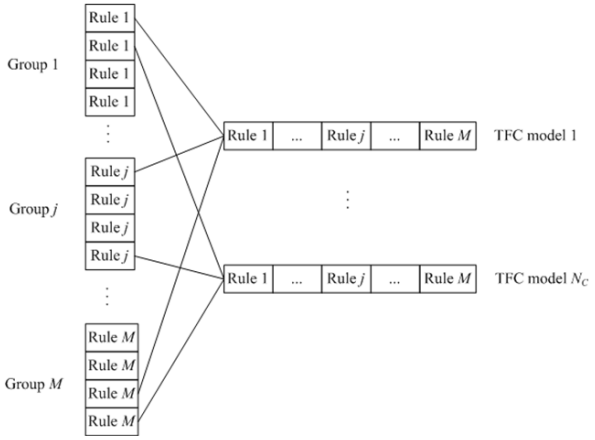

Figure 2. The structure of the chromosome in the GCSE

Figure 3 describes a fuzzy rule that had the form of Equation (1) where m_{ij} and σ_{ij} represent a Gaussian membership function with mean and deviation with *i*th dimension and *j*th rule node.

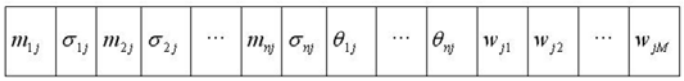

Figure 3. Coding a rule of a TFC into a chromosome in the GCSE

The learning process of the GCSE in each group involves five major operators: initialization, fitness assignment, elite-based reproduction strategy (ERS), elite-based compensatory of crossover strategy (ECCS), and mutation. The whole learning process is described step-by-step as follows:

a. Initialization Step:

Before the GCSE applies, individuals formed by several initial groups should be generated. The initial groups of the GCSE are generated randomly within a fixed range. The following formulations show how to generate the initial chromosomes in each group:

$$\text{Deviation: } Chr_{g,c}[p] = random[\sigma_{\min}, \sigma_{\max}] \quad (7)$$

where p=2, 4, …, 2n; g=1, 2,…, M; c=1, 2, …, N_C;

$$\text{Mean: } Chr_{g,c}[p] = random[m_{\min}, m_{\max}] \quad (8)$$

where p=1, 3, …, 2n-1;

$$\text{Weight: } Chr_{g,c}[p] = random[w_{\min}, w_{\max}] \quad \text{where } p = 2n+1, 2n+2, \ldots, 2n+(1+n) \tag{9}$$

where $Chr_{g,c}$ represents cth chromosome in gth group; M represents total number of groups and N_C is the total number of chromosomes in each group; p represents the pth gene in a $Chr_{g,c}$; and $[\sigma_{\min}, \sigma_{\max}]$, $[m_{\min}, m_{\max}]$, and $[w_{\min}, w_{\max}]$ represent the range that are predefined to generate the chromosomes.

b. Fitness Assignment Step:

As previously stated, for the GCSE method, the fitness value of a rule (an individual) is calculated by summing up the fitness values of all the possible combinations in the chromosomes that are selected randomly from M groups. The details for assigning the fitness value are described step by step as follows:

Step 1: Randomly choose M fuzzy rules from the M groups with size N_C.

Step 2: Evaluate every TFC model that is generated from step1 to obtain a fitness value.

Step 3: Divide the fitness value by M and accumulate the divided fitness value to the selected rules with their fitness value records that were set to zero initially

Step 4: Repeat the above steps until each rule (chromosome) in each group has been selected a sufficient number of times, and record the number of TFC models in which each individual has participated.

Step 5: Divide the accumulated fitness value of each chromosome by the number of times it has been selected. The average fitness value represents the performance of a rule.

c. Elites-based Reproduction Strategy:

Reproduction is a process in which individual strings are copied according to their fitness value. A fitness value is assigned to each chromosome in each group according to a fitness assignment method in which high numbers denote a good fit. The goal of the GCSE method is to maximize the fitness value. For keeping the algorithm stable, this study proposes an elite-based reproduction strategy (ERS) to let the best combination of chromosomes in each group can be kept in the next generation. In the GCSE, the chromosome that has best fitness value may not be the chromosome in the best combination. As a result, in the ERS, every chromosome in the best combination in each group must be kept to perform reproduction step. In the other chromosomes in each group, this study uses the roulette-wheel selection method [27] – a simulated roulette is spun – for the reproduction process. The best performing chromosomes in the top half of each group [22] advance to the next generation and the other half perform crossover operations on chromosomes in the top half of the parent generation. In the reproduction step, the top half of the population for each group must be kept the same number of chromosomes.

d. Elite-based Compensatory of Crossover Strategy:

Although the ERS operation can search for the best existing individuals, it does not create any new individuals. In nature, an offspring has two parents and inherits genes from both. The main operator working on the parents is the crossover operator, the operation of which occurs for a selected pair with a crossover rate. In this paper, for letting groups that can cooperate to generate better solutions, the elite-based compensatory of crossover strategy (ECCS) is proposed to perform the crossover operation. The ECCS mimics the cooperation phenomenon in society, in which individuals become more

suited to the environment as they acquire and share more knowledge of their surroundings. In the ECCS, the elites of each group will be selected to perform crossover operation in the next generation. The best performing individuals in the top half of each group that are called elites are used to select the parents for performing the ECCS. Details of the ECCS are shown below.

Step 1

The first one of the parents that is used to perform the crossover operation is selected from the original group by using the following equations:

$$Fitness\_Ratio_{g,\,t} = \frac{\sum_{u=1}^{t} fitness_{g,\,u}}{\sum_{c=1}^{Nc} fitness_{g,\,c}}, \quad \text{where} \quad t = 1,\ 2,\ \cdots,\ Nc \tag{10}$$

$$Rand\_Value[g] = Random[0,\ 1], \quad \text{where} \quad g = 1,\ 2,\ \cdots,\ M \tag{11}$$

$$\begin{aligned} &Parent\_SiteA[g] = t, \text{if} \\ &Fitness\_Ratio_{g,\,t-1} < Rand\_Value[g] \le Fitness\_Ratio_{g,\,t} \end{aligned} \tag{12}$$

where $Fitness\_Ratio_{g,t}$ is a fitness ratio of the fitness value of tth chromosome in the gth group; $Rand\_Value[g] \in [0,1]$ is the random values of gth group; $Parent\_SiteA[g]$ is the site where the first parent is. According to Equation (12), if the $Rand\_Value[g]$ is greater than the fitness ratio at (t-1)th chromosome in gth group and smaller or equal to the fitness ratio at tth chromosome in gth group, the site of the first parent of gth group is assigned to t.

Step 2

After determining the first parent, the best performing elites every group is used to determine the other parent. In this step, the total fitness ratio of every group is computed according to the following equations:

$$Total\_Fitness_{g} = \sum_{c=1}^{Nc} fitness_{g,\,c}, \quad \text{where} \quad g = 1,\ 2,\ \cdots,\ M, \tag{13}$$

$$Total\_Fitness\_Ratio_{w} = \frac{\sum_{u=1}^{w} Total\_Fitness_{u}}{\sum_{g=1}^{M} Total\_Fitness_{g}}, \quad \text{where} \quad w = 1,\ 2,\ \cdots,\ M, \tag{14}$$

where $Total\_Fitness_g$ represents the summation of the fitness value of every chromosomes in gth group; $Total\_Fitness\_Ratio_w$ is a total fitness ratio of wth group.

Step 3

Determine the group where the chromosome is selected from to be the other parent for performing crossover with the $Parent\_SiteA[g]$ th chromosome in gth group according to the following equations:

$$Group\_Rand\_Value[g] = Random[0,1] \quad \text{where} \quad g = 1,\ 2,\ \cdots,\ M, \tag{15}$$

ISSN 1927-6974 E-ISSN 1927-6982

$$Parent\_Group\_SiteB[g] = w, \quad \text{if}$$
$$Total\_Fitness\_Ratio_{w-1} < Group\_Rand\_Value[g] \le Total\_Fitness\_Ratio_{w}, \tag{16}$$

where $Group\_Rand\_Value[g] \in [0,1]$ is a random values of gth group; $Parent\_Group\_SiteB[g]$ represents the site of the group that the second parent is selected from.

Step 4

After the $Parent\_Group\_SiteB[g]$ th group is selected, the ECCS determines the other present in the selected $Parent\_Group\_SiteB[g]$ th group according to the following equations:

$$Fitness\_Ratio_{Selected\_g,\ t} = \frac{\sum_{u=1}^{t} fitness_{Selected\_g,\ u}}{\sum_{c=1}^{Nc} fitness_{Selected\_g,\ c}}, \tag{17}$$

where $t = 1,\ 2,\ \cdots,\ Nc;\ Selected\_g = Parent\_Group\_SiteB[g],$

$$Rand\_Value[g] = Random[0,\ 1], \quad \text{where} \quad g = 1,\ 2,\ \cdots,\ M, \tag{18}$$

$$Parent\_SiteB[g] = l, \quad \text{if}$$
$$Fitness\_Ratio_{Selected\_g,\ l-1} < Rand\_Value[g] \le Fitness\_Ratio_{Selected\_g,\ l}, \tag{19}$$

where $Fitness\_Ratio_{Selected\_g,\ t}$ is a fitness ratio of the fitness value of tth chromosome in the $Parent\_Group\_SiteB[g]$ th group; and $Parent\_SiteB[g]$ is the site where the second parent is.

After the ECCS selects the presents form the gth group and $Parent\_Group\_SiteB[g]$ th group, the individuals ($Parent\_SiteA[g]$ th chromosome and the $Parent\_SiteB[g]$ th chromosome) are crossed and separated using a two-point crossover in the gth group. The two-point crossover exchanges the site's values between the selected sites of parents' individual create new individuals. After this operation, the individuals with poor performances are replaced by the newly produced offspring.

e. Mutation:

Mutation is an operator that randomly alters the allele of a gene. Mutation can randomly alter the allele of a gene. In this paper, a uniform mutation [27] is adopted, and the mutated gene is drawn randomly from the domain of the corresponding variable.

The aforementioned steps are done repeatedly and stopped when the predetermined condition is achieved.

4 Two-Strategy reinforcement GCSE for a TFC model

In this study, the reinforcement signal is designed [25]. Although time-step reinforcement architecture becomes more simple and easier to implement than the Barto's architecture, there still have a problem that such reinforcement architecture only knows how long the controller successfully control the system. There are no criteria measuring the performance between two control processes with identical control time steps. In this paper, the two-strategy reinforcement

group cooperation based symbiotic evolution (TSR-GCSE) is proposed for solving the reinforcement problem that mentioned above. Figure 4 shows the TSR-GCSE and its training environment. The reinforcement signal is measured by two different strategies (judgment and evaluation strategy). As shown in Figure 4, the proposed TSR-GCSE consists of a TFC model, which acts as the control network to determine a proper action according to the current input vector (environment state). The structure of the proposed TSR-GCSE is different from Barto and his colleagues' actor-critic architecture [8], which consists of a control network and a critic network. The input to the TFC model is the state of the plant, and the output is a control action of the state, denoted by *f*. The feedback takes the form of an accumulator. An accumulator plays a role as a relative performance measurement. The key of the TSR-GCSE is formulating a number of time steps before failure occurs and before the controller does not perfume well and using this formulation as the fitness function of the TSR-GCSE method. It will be observed that the advantage of the proposed TSR-GCSE method is its global optimization capability.

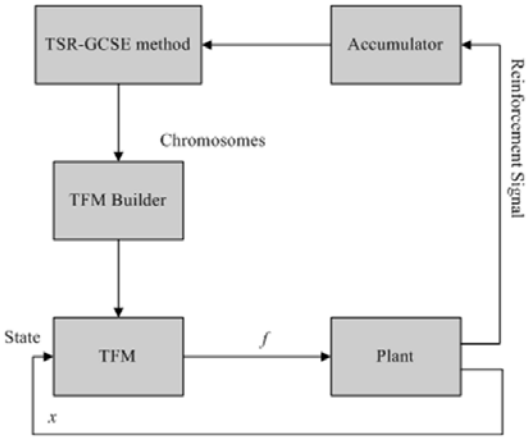

Figure 4. Schematic diagram of the TSR-GCSE for the TFC model

The proposed TSR-GCSE method runs in a feed forward fashion. The fitness function takes the form of an accumulator determined by how long the experiment is a "success" and performs well. In this way, according to a defined fitness function, a fitness value is assigned to each string in the population where high fitness values means good fit. The goal of the TSR-GCSE method is to maximize the fitness value. Details of the TSR are shown below.

Step 1

In the judgment strategy, the signal is defined according to how long the experiment is still a "success" The equation of the judgment strategy is shown as follow:

$$\begin{aligned} &\text{If the controller success controls the system} \\ &\text{Then} \quad Failure\_TimeStep = Failure\_TimeStep + 1, \end{aligned} \tag{20}$$

where the *Failure_TimeStep* represents how long the experiment is still a "success".

Step 2

In the evaluation strategy, the signal is defined according to measures how long the TFC model that still controls the system well in the predefined criterion value. The equation of the evaluation strategy is shown as follow:

$$\begin{aligned} &\text{If } \left(Time\_Value - i \middle/ Desired\_Times \right) \geq Criterion\_Value \\ &\text{Then } \quad NotWell\_TimeStep = NotWell\_TimeStep + 1, \\ &\text{where } \ = 1, 2, \cdots, Desired\_Times, \end{aligned} \tag{21}$$

where *NotWell_TimeStep* represents how long the experiment perform well; *Time_Value*, *Criterion_Value*, and *Desire_Times* are the predefined parameters.

Step 3

The accumulator only accumulates the minimum time steps of the signal in two strategies. That is, the accumulator will indicate the "fitness" of the current TFC model. The equation of an accumulator is shown as follow:

$$Accumulator\_Value = \min[Failure\_TimeStep, NotWell\_TimeStep], \tag{22}$$

where the Equation (22) reflects the fact that long-time steps before failure occurs or performs not well (to keep the desired control goal longer) mean higher fitness of the TSR-GCSE method.

5 Illustrative examples

Two applications are discussed in this section. The first simulation was performed to balance the cart-pole system that was described in [34, 35]. The second simulation was performed to balance the ball and beam system that was described in [33]. For the two examples, the initial parameters are given in Table 1. The initial parameters are determined by practical experimentation.

Example 1: Control of a Cart-Pole Balancing System

In this example, the TSR-GCSE is applied to the classic control problem of the cart-pole balancing. This problem is often used as an example of inherently unstable and dynamic systems to demonstrate both modern and the classic control techniques [30, 31, 32], or the reinforcement learning schemes [28], and is now used as a control benchmark. The cart-pole balancing problem is the problem of learning how to balance an upright pole. The bottom of the pole is hinged to a cart that travels along a finite-length track to its right or left. Both the cart and the pole can move only in the vertical plane; that is, each has only one degree of freedom.

Table 1. The initial parameters before training

Parameters	Value
N_c	10
Crossover Rate	0.5
Mutation Rate	0.3
Time_Value	10100
Desire_Times	10000
[$\sigma_{\min}$, $\sigma_{\max}$]	[0, 2]
[$m_{\min}$, $m_{\max}$]	[0, 2]
[$w_{\min}$, $w_{\max}$]	[-20, 20]

There are four state variables in the system: θ, the angle of the pole from an upright position (in degrees); $\dot{\theta}$, the angular velocity of the pole (in degrees/seconds); x, the horizontal position of the cart's center (in meters); and $\dot{x}$, the velocity of the cart (in meters/seconds). The only control action is f, which is the amount of force (in Newtons) applied to cart to move it toward left or right. The system fails when the pole falls past a certain angle ($\pm$ 12 is used here) or the cart runs into the

bounds of its track (the distance is 2.4m from the center to each bound of the track). The goal of this control problem is to determine a sequence of forces applying to the cart to balance the pole upright. The equations of the cart-pole balancing system can be found [25]. A control strategy is deemed successful if it can balance a pole for 100,000 time steps.

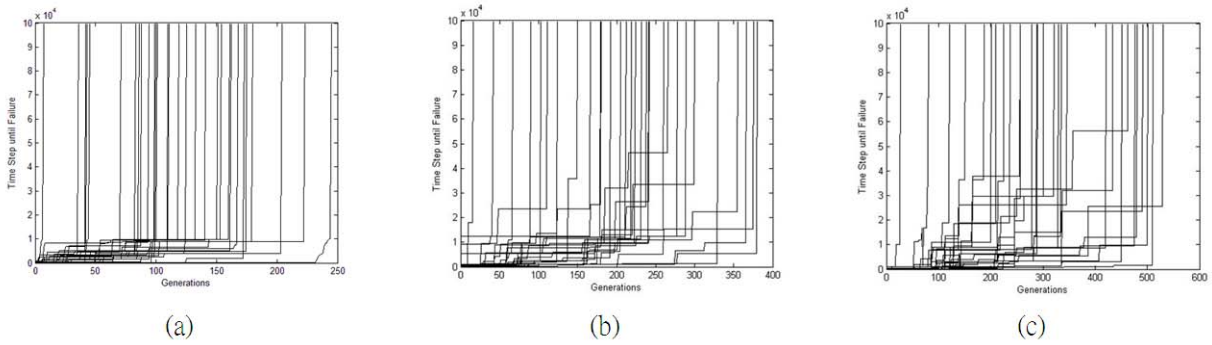

Figure 5. The performance of (a) the TSR-GCSE method, (b) the R-SE method, and (c) the R-GA method on the cart-pole balancing system

The four input variables $(\theta,\ \dot{\theta},\ x,\ \dot{x})$ and the output f_t are normalized between 0 and 1 over the following ranges, θ: [-12, 12], $\dot{\theta}$: [-60, 60], x: [-2.4, 2.4], $\dot{x}$: [-3, 3], f_t: [-10, 10]. The four normalized state variables are used as inputs to the proposed TFC model. The coding of a rule in a chromosome is the form in Figure 3. The values are floating-point numbers assigned using the TSR-GCSE initially. The fitness function in this example is defined in Equation (22) to train the TFC model where Equation (22) represents how long the cart-pole balancing system fails and receives a penalty signal of -1 when the beam deviates beyond a certain angle ($|\theta|>12°$) and the cart runs into the bounds of its track ($|x|>2.4$m) and how long the TFC model controls the system well. In this experiment, the initial values are set to (0, 0, 0, 0) and θ is chosen to be *Criterion_Value* from the four input variables. There are four rules to construct the TFC model. A total of thirty runs were performed. Each run started in the same initial state. Figure 5(a) shows that the TFC model learned on average to balance the pole at the 117th generation. In this figure, each run represents that largest fitness value in the current generation is selected before the cart-pole balancing system fails. When the TSR-GCSE is stopped, the best combination of strings at the final generation are selected and test them on the system.

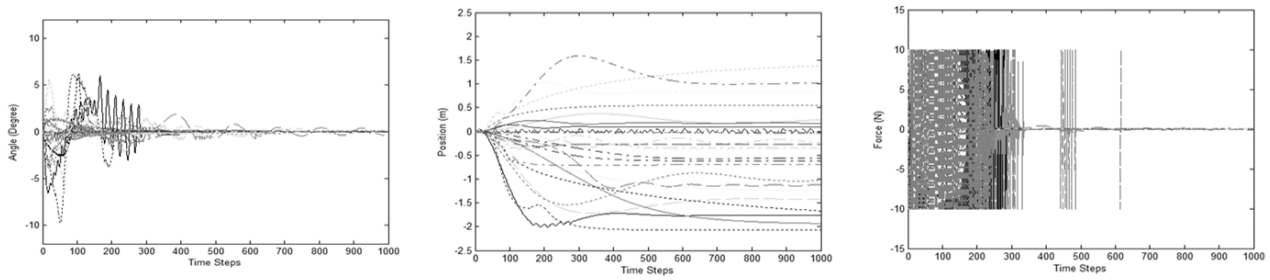

Figure 6. Control results of the cart and pole balancing system using the TSR-GCSE in Example 1. (a) Angle of the pole. (b) Position of the cart. (c) Control force.

The successful results, which consist of the pole angle, cart position and controller output, are shown in Figure 6. Each line in Figure 6 represents each run with a same initial state. The results shown in this figure is the first 1,000 time steps in the 100,000 control time steps. As shown in Figure 6, the TSR-GCSE successfully controlled the cart-pole balancing system

in thirty runs. Moreover, the small angular deviation is obtained by using the TSR method. The average angular deviation was 0.57°.

In this example, in order to show the effectiveness and efficiency of the proposed TSR-GCSE method, reinforcement symbiotic evolution (R-SE) [25], and the reinforcement genetic algorithm (R-GA) [19] were applied to the same problem. Four rules were set for the R-SE and R-GA because the number of rules used from the TSR-GCSE is four. In this study, the parameters are found using the method [37]. Therefore, the population size has the range from 10 to 250 in increments of 10, the crossover rate has the range from 0.25 to 1 in increments of 0.05, and the mutation rate has the range from 0 to 0.3 in exponential increments. Figure 5(b) and (c) shows the R-SE and the R-GA methods learned on average to balance the pole at the 214th and 324th generations. Figure 7(a) and (b) shows the angular deviation of the pole in thirty runs when the cart-pole balancing system was controlled by (R-SE) [25] and (R-GA) [19]. The average angular deviation of methods [8, 12] were 3.270 and 4.630. As shown in Figures 5-7, the control capabilities of the trained TFC model using the TSR-GCSE are better than in the cart-pole balancing system [19, 25]. Moreover, the angular deviation of the proposed TSR-GCSE is smoother than that of [19, 25]. The GENITOR [35], the SANE (Symbiotic Adaptive Neuro-Evolution) [32], and the R-HELA [26] have been applied to the same control problem and the simulation results are listed in Table 2. This experiment uses a Pentium 4 chip with a 1.5GHz CPU, a 512MB memory, and the visual C++ 6.0 simulation software. Table 2 shows the number of pole-balance trials (which reflects the number of training episodes required), CPU times, and angular deviation. As shown in Table 2, the proposed TSR-GCSE is feasible and effective.

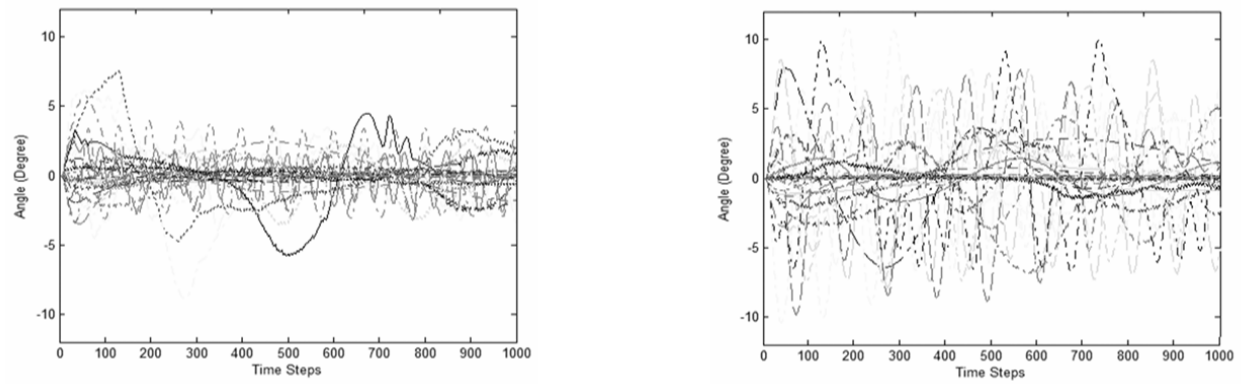

Figure 7. Control results (angle of the pole) of the cart and pole balancing system using (a) the R-SE method and (b) the R-GA method

Table 2. Performance comparison of various existing models in example 1

Method	Time steps			CPU Time (seconds)			Angular deviation
	Mean	Best	Worst	Mean	Best	Worst	
GENITOR[35]	3268	415	18743	70.95	33.34	246.36	5.98
SANE[32]	1984	46	5865	43.56	16.54	156.84	5.32
R-GA[19]	324	26	520	47.59	11.85	102.63	4.63
R-SE[25]	214	15	380	38.85	8.53	90.78	3.27
R-HELA[26]	174	21	287	31.45	7.49	65.74	1.63
TSR-GCSE	**117**	**7**	**215**	**13.87**	**3.06**	**30.04**	**0.57**

Example 2: Control of a Ball and Beam System

The ball and beam system is shown [32]. The beam is made to rotate in vertical plane by applying a torque at the center of rotation and the ball is free to roll along the beam. The goal is that the ball remains in contact with the beam.

The equations of the ball and beam system can be found [26, 27]. A control strategy is deemed successful if it can balance the ball and beam system for 100,000 time steps. The four input variables $(r, \dot{r}, \theta, \dot{\theta})$ and the output $u(k)$ are normalized between 0 and 1 over the following ranges, r:[-5, 5], $\dot{r}$:[-3, 3], θ:[-1, 1], $\dot{\theta}$:[-2, 2], and u:[-70, 70]. The values are floating-point numbers assigned using the TSR-GCSE initially. In the proposed TSR-GCSE method, the fitness function in this example is also defined in Equation (22) to train the TFC model where Equation (22) represents how long the ball and beam system fails and receives a penalty signal of -1 when the beam deviates beyond a certain angle ($|\theta| > 12°$) and the ball reaches the end of the beam ($|r| > 2$m) and how long the TFC model controls the ball and beam system well. In this example, the r is chosen to be *Criterion_Value* from the four input variables. There are five rules to construct the TFC model. A total of thirty runs were performed. Each run started in the same initial state. Figure 8(a) shows the TFC model learned on average to balance the ball at the 139th generation. In this figure, each run represents that largest fitness value in the current generation is selected before the ball and beam system fails. When the learning process is stopped, the best combination of strings each group at the final generation are selected and test it on the ball and beam system.

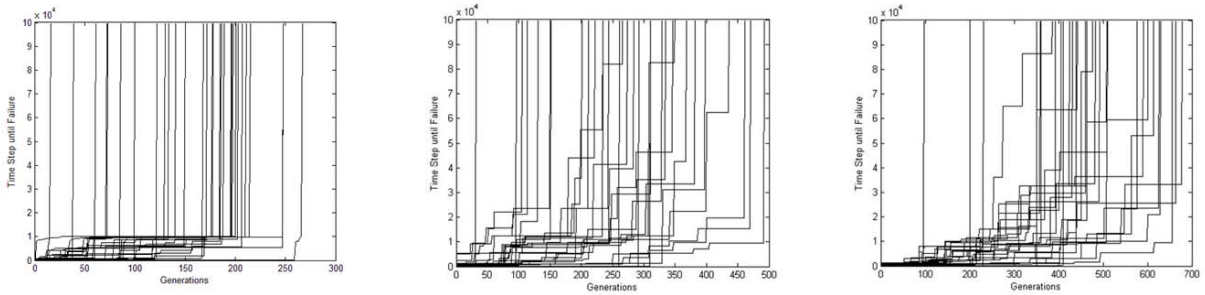

Figure 8. The performance of (a) the TSR-GCSE method, (b) the R-SE method, and (c) the R-GA method on the ball and beam balancing system

The simulation was run thirty times. The successful results, which consist of the beam angle, ball position, and controller output, are shown in Figure 9. Each line in Figure 9 represents each run in the TSR-GCSE. The results shown in this figure is the first 1,000 time steps in the 100,000 control time steps. As shown in Figure 9, the TSR-GCSE in thirty runs successfully controls the ball and beam system with a small deviation in both the position and angle. Moreover, as show in this figure, the position of the ball is decay to zero gradually by using the proposed TSR method. The results show the good control ability of the trained TFC model in the ball and beam balancing system. Moreover, the smooth position of the ball is obtained by using the TSR method.

In this example, as with example 1, the TSR-GCSE is also compared the performance with other methods (the R-SE [25] and R-GA [19]). The parameters [19, 25] are the same with example 1. Figure 8(b) and (c) shows the R-SE [25] method and the R-GA [19] method learned on average to balance the ball at the 274th generation and 466th generation. Figure 10(a) and (b) shows the position deviation of the ball in thirty runs when the ball and beam system was controlled by the R-SE and the R-GA methods starting at the initial state: $r(0) = -1.2$, $\dot{r}(0) = -0.01$, $\theta(0) = 0.58$, and $\dot{\theta}(0) = 0.58$. As shown in Figures 8-10, the control capabilities of the trained TFC model using the TSR-GCSE are also better than [19, 25] in the ball and beam balancing system. Moreover, the position deviation of the proposed TSR-GCSE is smoother than that of [19, 25]. Table 3

shows the performance compared (the number of trials and CPU times) with various existing models [19, 25, 26, 32, 35] in Example 2. From Tables 3, the proposed TSR-GCSE method obtains better performance indices and smaller CPU times than the existing models.

Table 3. Performance comparison of various existing models in Example 2

Method	Time steps			CPU Time (seconds)		
	Mean	**Best**	**Worst**	**Mean**	**Best**	**Worst**
GENITOR[35]	4982	551	19853	113.15	73.34	297.62
SANE[32]	2287	150	6217	70.26	51.54	197.61
R-GA[19]	466	97	678	60.79	46.35	122.93
R-SE[25]	274	32	492	42.25	18.23	101.43
R-HELA[26]	198	32	312	38.38	10.95	68.62
TSR-GCSE	**139**	**12**	**214**	**16.34**	**4.01**	**23.58**

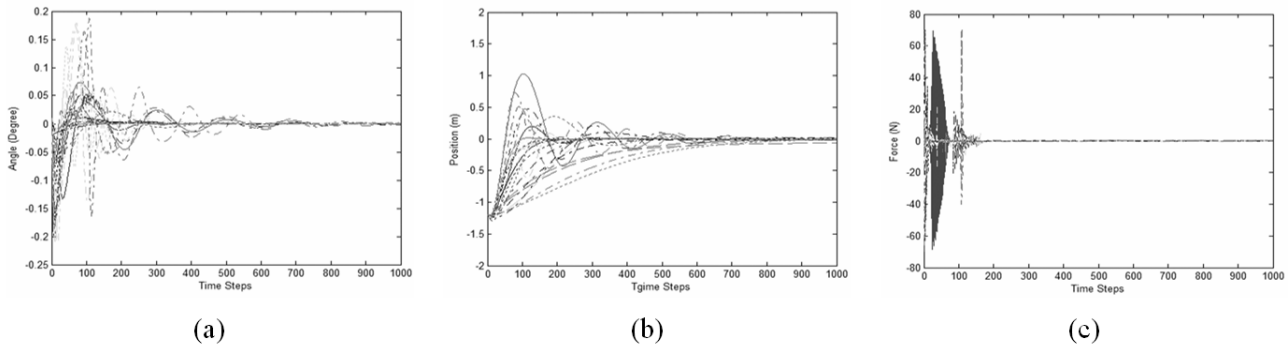

Figure 9. Control results of the ball and beam balancing system using the TSR-GCSE in Example 2. (a) Angle of the beam. (b) Position of the ball. (c) Control force.

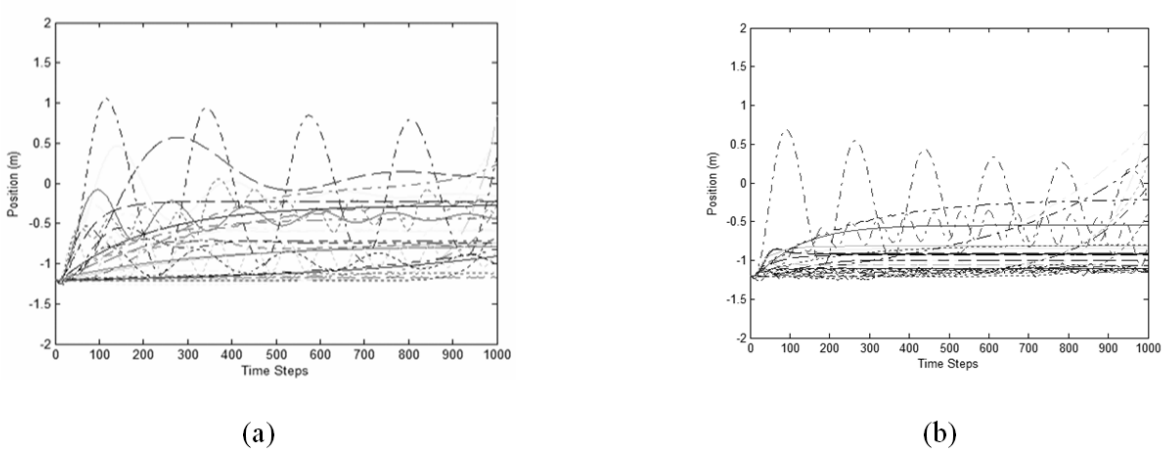

Figure 10. Control results (position of the ball) of the ball and beam balancing system using (a) the R-SE method and (b) the R-GA method.

6 Conclusion

In this paper, a TFC with the two-strategy reinforcement group cooperation based symbiotic evolution method is proposed. The proposed TSR-GCSE method consists of the two-strategy reinforcement signal design and the group cooperation based symbiotic evolution to perform the parameter learning for tuning the TFC model efficiently. The TSR-GCSE method can evaluate the fuzzy rule locally and cooperate with each group to generate the better chromosomes by using elites-base compensation crossover strategy. Moreover, the TSR-GCSE method proposes the TSR that uses two strategies to design the reinforcement signal for improving the performance of the traditional reinforcement signal design. Computer simulations have shown that the proposed method has a better performance than the other compared methods.

Acknowledgment

This work was supported in part by National Science Council, Taiwan, R.O.C., under Contract No. NSC 100-2221-E-009-038.

References

[1] G. G. Towell and J. W. Shavlik, “Extracting refined rules from knowledge-based neural networks,” Machine Learning. 1993; 13: 71-101. http://dx.doi.org/10.1007/BF00993103

[2] C. J. Lin and C. T. Lin, “An ART-based fuzzy adaptive learning control network,” IEEE Trans. Fuzzy systs. 1997; 5(4): 477-496.

[3] C. F. Juang and C. T. Lin, “An on-line self-constructing neural fuzzy inference network and its applications,” IEEE Trans. Fuzzy Systs. 1998; 6(1): 12-31. http://dx.doi.org/10.1109/91.660805

[4] C. J. Lin and C. C. Chin, “Prediction and identification using wavelet-based recurrent fuzzy neural networks,” IEEE Trans. Systs., Man, Cybern. Part B. 2004; 34(5): 2144-2154. http://dx.doi.org/10.1109/TSMCB.2004.833330

[5] K. S. Narendra and K. Parthasarathy, “Identification and control of dynamical systems using neural networks,” IEEE Trans. Neural Networks. 1990; 1(1): 4-27. PMid:18282820 http://dx.doi.org/10.1109/72.80202

[6] V. B. M. Walter, G. Schram, R. Babuska, and H. B. Verbruggen , “Adaptive fuzzy control of satellite attitude by reinforcement learning,” IEEE Trans. Fuzzy Systs. 1998; 6(2): 185-194. http://dx.doi.org/10.1109/91.669012

[7] C. J. Lin, and C. T. Lin, “Reinforcement learning for an ART-based fuzzy adaptive learning control network,” IEEE Trans. Neural Network. 1996; 7(7): 709-731. PMid:18263467

[8] G. Barto, R. S. Sutton, and C. W. Anderson, “Neuron like adaptive elements that can solve difficult learning control problem,” IEEE Trans. Syst., Man, Cybern. 1983; 13(5): 834-847.

[9] J. Lin, “A GA-based neural network with supervised and reinforcement learning,” Journal of the Chinese Institute of Electrical Engineering. 2002; 9(1): 11-25.

[10] Vengerov, N. Bambos, and H. R. Berenji, “A fuzzy reinforcement learning approach to power control in wireless transmitters,” IEEE Trans. Systs., Man, and Cybern.-Part B: Cybern. 2005; 35(4): 768-778. http://dx.doi.org/10.1109/TSMCB.2005.846001

[11] S. G. Tzafestas, and G. G. Rigatos, “Fuzzy reinforcement learning control for compliance tasks of robotic manipulators,” IEEE Trans. Syst., Man, Cybern. - Part B: Cybernetics. 2002; 32(1): 107-113. PMid:18238109 http://dx.doi.org/10.1109/3477.979965

[12] W. M. Hinojosa, and S. Nefti, “Systems control with generalized probabilistic fuzzy-reinforcement learning,” IEEE Trans. Fuzzy Systs. 2001; 19(1): 51-64. http://dx.doi.org/10.1109/TFUZZ.2010.2081994

[13] C. T. Lin and C. P. Jou, “GA-based fuzzy reinforcement learning for control of a magnetic bearing system,” IEEE Trans. Syst. Man Cybern. Part B. 2000; 30(2): 276-289. PMid:18244754 http://dx.doi.org/10.1109/3477.836376

[14] H. R. Berenji and P. Khedkar, “Learning and tuning fuzzy logic controllers through reinforcements,” IEEE Trans. Neural Networks. 1992; 3(5): 724-740. PMid:18276471 http://dx.doi.org/10.1109/72.159061

[15] D. E. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning. Reading, MA: Addison-Wesley, 1989.

[16] J. K. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection. Cambridge, MA: MIT Press, 1992.

[17] L. J. Fogel, “Evolutionary programming in perspective: The top-down view,” in Computational Intelligence: Imitating Life, J. M. Zurada, R. J. Marks II, and C. Goldberg, Eds. Piscataway, NJ: IEEE Press, 1994.

[18] Rechenberg, “Evolution strategy,” in Computational Intelligence: Imitating Life, J. M. Zurada, R. J. Marks II, and C. Goldberg, Eds. Piscataway, NJ: IEEE Press, 1994.

[19] C. L. Karr, “Design of an adaptive fuzzy logic controller using a genetic algorithm,” Proc. The Fourth Int. Conf. Genetic Algorithms. 1991; 450-457.

[20] M. Lee and H. Takagi, “Integrating design stages of fuzzy systems using genetic algorithms,” Proc. 2nd IEEE Int. Conf. Fuzzy Systems, San Francisco, CA. 1993; 612-617.

[21] C. T. Lin and C. P. Jou, “GA-based fuzzy reinforcement learning for control of a magnetic bearing system,” IEEE Trans. Syst., Man, Cybern., Part B. 2000; 30(2): 276-289.

[22] C. F. Juang, J. Y. Lin, and C. T. Lin, “Genetic reinforcement learning through symbiotic evolution for fuzzy controller design,” IEEE Trans. Syst., Man, Cybern., Part B. 2000; 30(2): 290-302.

[23] S. Tang, “Genetic algorithms in modeling and optimization,” Ph.D. dissertation, Dep. Electron. Eng., City Univ. Hong Kong, Hong Kong, 1996.

[24] C. F. Juang, “Combination of online clustering and Q-value based GA for reinforcement fuzzy system design,” IEEE Trans. Fuzzy Systs. 2005; 13(3): 289-302. http://dx.doi.org/10.1109/TFUZZ.2004.841726

[25] C. J. Lin and Y. J. Xu, “Efficient reinforcement learning through dynamical symbiotic evolution for TSK-Type fuzzy controller design,” International Journal of General Systems. 2005; 34(5): 559-578. http://dx.doi.org/10.1080/03081070500132377

[26] C. J. Lin and Y. C. Hsu, “Reinforcement hybrid evolutionary learning for recurrent wavelet-based neurofuzzy systems,” IEEE Trans. Fuzzy Systs. 2007; 15(4): 729-745. http://dx.doi.org/10.1109/TFUZZ.2006.889920

[27] D. W. C. Ho, P. A. Zhang, and J. Xu, “Fuzzy wavelet networks for function learning,” IEEE Trans. Fuzzy Systs. 2001; 9(1): 200-211. http://dx.doi.org/10.1109/91.917126

[28] Z. Michalewicz, Genetic Algorithms+Data Structures=Evolution Programs. New York: Springer-Verlag, 1999.

[29] R. Tanese, “Distributed genetic algorithm,” Proc. Int. Conf. Genetic Algorithms. 1989; 434-439.

[30] Arabas, Z. Michalewicz, and J. Mulawka, “GAVaPS-A genetic algorithm with varying population size,” Proc. IEEE Int. Conf. Evolutionary Computation, Orlando. 1994; 73-78.

[31] D. E. Moriarty and R. Miikkulainen, “Efficient reinforcement learning through symbiotic evolution,” Machine Learning. 1996; 22: 11-32. http://dx.doi.org/10.1007/BF00114722

[32] R. E. Smith, S. Forrest, and A. S. Perelson, “Searching for diverse, cooperative populations with genetic algorithms,” Evol. Comput. 1993; 1(2): 127-149. http://dx.doi.org/10.1162/evco.1993.1.2.127

[33] Cordon, F. Herrera, F. Hoffmann, and L. Magdalena, Genetic fuzzy systems evolutionary tuning and learning of fuzzy knowledge bases. Advances in Fuzzy Systems-Applications and Theory, vol.19, NJ: World Scientific Publishing, 2001.

[34] C. Cheok and N. K. Loh, “A ball-balancing demonstration of optimal and disturbance-accommodating control,” IEEE Contr. Syst. Mag. 1987; 54-57. http://dx.doi.org/10.1109/MCS.1987.1105235

[35] D. Whitley, S. Dominic, R. Das, and C. W. Anderson, “Genetic reinforcement learning for neuro control problems,” Machine Learning. 1993; 13: 259-284. http://dx.doi.org/10.1023/A:1022674030396

[36] J. Hauser, S. Sastry, and P. Kokotovic, “Nonlinear control via approximate input-output linearization: the ball and beam example,” IEEE Trans. Automatic Control. 1992; 37(3): 392-398. http://dx.doi.org/10.1109/9.119645

[37] J. J. Grefenstette, “Optimization of control parameters for genetic algorithms,” IEEE Trans. Syst., Man, Cybern. 1986; 6(1): 122-128. http://dx.doi.org/10.1109/TSMC.1986.289288