Generation of a Fuzzy Logic Controller Using Evolutionary Strategies

Tao-Yuan Huang and Yung-Yaw Chen
Institute pf Electrical Engineering,
National Taiwan University
Taipei, Taiwan, ROC, 106
hubert@ipmc.ee.ntu.edu.tw

Abstract

In this paper, the authors propose a *fuzzy logic controller* (FLC) generation scheme using a modified *(μ+λ) evolution strategy* (ES) with *parental population sizing* [1]. The object variable portion of each ES individual represents a control rule of the FLC to be optimized. Therefore, the *parental population* along constitutes one and only one candidate FLC rulebase improved over the generations. The improvement is attributable to an increase of better-fitted individuals in the parental population. Fitness values of individual rules are calculated symbiotically [2]. At each generation, offspring rules compete with their parents to form a new rulebase, which then strives to replace the original candidate rulebase according to elitist principle. In this aspect, the proposed FLC generation scheme practices (1+1) ES on the rulebase level. Furthermore, size of the candidate rulebase is simultaneously adjusted via parental population sizing to reflect system demands. The resultant scheme not only can construct FLC's with better performance but also provide additional flexibility to their rulebase structures. Simulations on several control examples have been conducted to demonstrate the virtue of this FLC generation scheme.

1. Introduction

Fuzzy logic controllers (FLC's) are applications of *fuzzy logic systems* (FLS's) based on Zadeh's theory of fuzzy sets and fuzzy logic [3][4]. The introduction of fuzzy terms and membership functions to deal with uncertainties and imprecise information enables computerized systems to operate in a more human-like style. Moreover, the combination of fuzzy logic and approximate reasoning makes incorporation of expert knowledge into control rules straightforward. In short, FLC's possess the abilities to exploit ambiguity in operation strategies and handle complexity of controlled systems.

Although FLC's stand the advantages of being intuitive as well as convenient over traditional control systems, its symbolic nature makes controller design difficult. Conventionally, the construction of a FLC relies significantly on design heuristics and domain knowledge. The obtained controller is subsequently fine-tuned through trial-and-error for better performance. Despite its simplicity, the design procedure is too unsystematic to cope with deep knowledge [5][6]. Furthermore, when necessary knowledge is incomplete or difficult to be conveyed linguistically, a satisfactory FLC may be hard to come by.

To redress these serious drawbacks, many approaches have been developed to generate FLC's automatically from numerical data. These include the use of *artificial neural networks* (ANN's) [7][8][9] and *evolutionary algorithms* (EA's) such as *genetic algorithms* (GA's) and *evolution strategies* (ES's) [10]–[21] in FLC construction. Among these alternatives, the employment of GA's to build FLC's receives great attention because they are powerful search techniques well suited to such complex issues as placement of fuzzy term sets and structural arrangement of rules.

In the beginning, GA's are only used to optimize parameters of fuzzy term sets and rules [10]. However, as technology advances, simultaneous adjustment of membership functions and rulebase becomes probable [22]. This integrated design strategy is important and necessary because choices about fuzzy terms and rule sets are heavily interdependent. Nevertheless, the manner how a FLC is represented by a GA individual determines the efficiency and efficacy of the GA-tuning scheme. For example, if each individual of the GA population encodes a candidate FLC, its length could grow exponentially with dimensionality of the problem space, number of fuzzy term sets, and structure of rules. On the other hand, if chromosome length based on predetermined FLC specifications is held fixed during evolution, the pursuit of better performance may become very difficult.

Inspired by such improved EA's with flexible encoding schemes [23][24] and variable-length chromosomes [25], the authors propose a modified FLC generation scheme using a modified (μ+λ) ES with *parental population sizing* [1]. The object

0-7803-7078-3/01/$10.00 (C)2001 IEEE.

variable portion of each ES individual herein represents a control rule of the FLC to be optimized. Therefore, the *parental population* along constitutes only one candidate FLC rulebase to be improved over the generations.

This FLC generation algorithm looks like the ES version of *Symbiotic Evolution for Fuzzy Controller* (SEFC) design methodology proposed in [2]. However, improvements have been made to the selection process and rulebase sizing scheme. At each generation, offspring rules compete with their parents to form a new rulebase, which then strives to replace the original candidate rulebase according to elitist principle. From this aspect, the algorithm practices (1+1) ES on the rulebase level to generate a robust FLC. Furthermore, size of the candidate rulebase is adjusted simultaneously via parental population sizing to reflect actual system demands. The resultant scheme not only constructs FLC's with better performance but also provides additional flexibility to rulebase structures.

This paper is organized as follows. First, a brief review of FLC is given. Second, the modified ES with parental population sizing is introduced. Algorithm of the proposed FLC generation scheme is then described and compared with other EA-tuning algorithms. Simulations examples is also provided to demonstrate the virtue of this FLC generation scheme. Finally, conclusions and future research directions are summarized.

2. Fuzzy Logic Controllers

A FLC builds on fuzzy logic to bind the world of measurements with the domain of knowledge. It performs transformations on crisp numerical data from input variables into fuzzy term sets and membership values so that fuzzy inference can be processed to produce a conclusion. The conclusion in terms of fuzzy sets is then re-transformed back to numerical control commands feeding system plants. Accordingly, a FLC usually consists of the following four major constituents:

- *Fuzzifier*: transforming crisp inputs into fuzzy antecedents;
- *IF-THEN Rulebase*: defining what consequents are to be resolved when antecedents to a rule are fulfilled;
- *Fuzzy Inference Engine*: producing firing strengths of fuzzy rules and consequents by combining fuzzy membership values of fired antecedents through logical connectives; and
- *Defuzzifier*: transforming fuzzy consequents to crisp output values.

In real-world implementations, transformations between crisp numerical data and fuzzy terms are conducted over predefined fuzzy membership functions taking up value [0, 1]. Fuzzy membership value denotes the degree of correspondence between a piece of numerical data and a fuzzy term set. Popular membership functions are triangular, trapezoidal, Gaussian membership functions, and fuzzy singletons.

As mentioned before, the relationship between input and output fuzzy terms is specified by the IF-THEN rules of the form:

Rule n: **IF** x_1 is X_{1n} **AND** x_2 is X_{2n} … ,
THEN y_1 is Y_{1n} **AND** y_2 is Y_{2n} … .

where X_{in}'s and Y_{jn}'s are fuzzy term sets associated respectively with the input variable x_i and output variable y_j in Rule n.

Normally, input spaces and output spaces of the FLC are partitioned into predetermined fuzzy term sets according to expert knowledge or by clustering techniques. These term sets are shared by all rules in the rulebase, which can be constructed by exhaustive enumeration. Nevertheless, the FLC in this paper adopts a variable rulebase structure with fuzzy term sets and number of rules optimized by the ES scheme during evolution.

For the sake of simplicity, the authors choose Gaussian membership functions with input variables and fuzzy singletons with outputs. The Gaussian membership function can be represented by two parameters, i.e. $X_{in} = \mu(m_{in}, \sigma_{in})$, where m_{in} is the center and σ_{in} is the spread of the term set. On the other hand, fuzzy singleton only needs one parameter, $Y_{jn} = \omega_{jn}$, denoting a crisp value.

A rule used in this paper therefore looks like

Rule n: **IF** x_1 is $\mu(m_{1n}, \sigma_{1n})$ **AND** x_2 is $\mu(m_{2n}, \sigma_{2n})$ … ,
THEN y_1 is ω_{1n} **AND** y_2 is ω_{2n} … .

Finally, by using *product*-**AND** and *center-average defuzzification* [26], control commands of the FLC can be computed with the formula:

$$y_j = \frac{\sum_n \left\{ \omega_{jn} \cdot \exp\left[-\sum_i \frac{(x_i - m_{in})^2}{\sigma_{in}^2} \right] \right\}}{\sum_n \exp\left[-\sum_i \frac{(x_i - m_{in})^2}{\sigma_{in}^2} \right]} \quad (1).$$

Obviously, values of these parameters not only define corresponding fuzzy term sets but also influence behavior of the FLC.

3. Evolution Strategies with Parental Population Sizing

Rechenberg [27] and Schwefel [28] originally developed ES's to solve parameter optimization problems. Like GA's, this methodology evolves a population of possible solutions by subjecting these individuals repeatedly into the evolution cycle, which includes such evolution operators as recombination, mutation, and selection. Over the generations, this evolving population of possible solutions embodies a fitness-driven random search after a (global) optimum.

ES's stand out from GA's for their emphasis on mutation as a key evolutionary operator. In fact, self-adaptive mutation mechanism is one major feature of ES's. For example, the j^{th} individual in ES's employing n-dimensional uncorrelated self-adaptive mutation over an n-dimensional optimization problem usually consists of

$$\vec{a}_j(k) = (\vec{X}_j(k),\ \vec{\sigma}_j(k)) \quad (2),$$

where

$$\vec{X}_j(k) \in R^n \quad (3)$$

is the vector of *object variables* in the n-dimensional search space, and

$$\vec{\sigma}_j(k) \in R_+^n \quad (4)$$

is the vector of *strategy parameters.*

By definition, object variable portion of an individual signifies the actual position of the candidate solution in search space. On the other hand, the vector of strategy parameters are concerned with "mutation step sizes" of an individual.

Other features include the employment of truncation selection operator with high selective pressure and the almost exclusive use of real-number representations with search solutions. All of these characteristics contribute to ES's rapid convergence rate over other population-based search techniques.

Since ES's are primarily dedicated to real-number optimization problems, they should also be very efficient at tuning FLC parameters like those in (1). However, since generation of FLC's includes both parameter optimization as well as rulebase formation, which is combinatorial in nature, applications of ES's in this area are far fewer in number than those of GA's.

Nonetheless, if one is to overlook the linguistic requirements of fuzzy term sets for a moment, as having been done in [24], the construction of fuzzy rulebases is no different from assigning appropriate membership functions to each variable in every rule on a case-by-case basis. As a consequence, the FLC generation problem is translated back to a parameter optimization problem to which ES's can be applied.

Basic scheme of the ES's used in this paper is as that described in [29][30]. Each ES trial starts with a random initial population, $P(0) = \{\vec{a}_1(0), \cdots, \vec{a}_\mu(0)\}$, of μ individuals. At the k^{th} generation, these individuals reproduce an intermediate population ($P'(k)$) of λ offspring ($1 \le \mu \le \lambda$) through recombination and self-adaptive mutation. After that, a selection pool $P''(k)$ is formed, where a new population of μ individuals will be chosen as parents for the next generation ($P(k+1)$). The selection pool can be either the entirety of offspring alone ((μ, λ) ES) or the union of offspring and parents together ((μ+λ) ES) [31]. The authors choose the more robust (μ+λ) ES as the backbone evolution scheme for FLC generation because the elitism it practices always conserves the best solutions found.

Moreover, since parental population sizing has been observed in [1] to enhance performance of (μ+λ) ES greatly, the authors therefore try to apply this modification to the backbone FLC generation scheme for better results. To sum up, parental population sizing varies the number of individuals to be selected as new parents for the next generation in response to *fitness concentration* of the present population.

Definition 1. *Best, Average, Worst Population Fitness* (F_B, F_A, F_W):

Given a population $P''(k) = \{\vec{a}_1(k), ..., \vec{a}_{\mu+\lambda}(k)\}$ of (μ+λ) individuals with known fitness values $\{\phi_1(k), \cdots, \phi_{\mu+\lambda}(k)\} = \{\Phi(\vec{X}_1(k)), \cdots, \Phi(\vec{X}_{\mu+\lambda}(k))\}$ at the k^{th} generation, where $\Phi(\cdot)$ is the fitness function, then *Average Population Fitness* is computed by

$$F_A(k) = \frac{1}{\mu+\lambda}\sum_j \phi_j(k) \quad (5);$$

Best and *Worst Population Fitness* are respectively defined as

$$F_B(k) = \min_j(\phi_j(k)) \quad (6),$$

and

$$F_W(k) = \max_j(\phi_j(k)) \quad (7).$$

Definition 2. *Fitness Concentration* (F_C):
With the same population given above, the measure of *Fitness Concentration* is defined to by

$$F_C(k) = \frac{F_W(k) - F_A(k)}{F_W(k) - F_B(k)} \quad (8).$$

However, if $F_W(k)$ and $F_B(k)$ are of such values that singularity may arise, $F_C(k)$ is defined as unity.

Having obtained the value of $F_C(k)$, number of parents to be selected at the k^{th} generation ($\mu(k)$) is given by

$$\mu(k) = \lceil (\mu_{max} - \mu_{min}) \cdot F_C(k) \rceil + \mu_{min} \quad (9).$$

The rationale behind parental population sizing is self-apparent: when fitness values distribution of the population becomes too concentrated, select more individuals from the selection pool to boost diversity. On the other hand, when fitness value distribution of the population exhibit enough diversity, use less parents to accelerate population convergence.

With an ES population of variable size to freely adapt fuzzy membership functions, the authors decide to further deviate from the beaten track where one individual defines one candidate rulebase and inefficiency haunts the evolution process as dimensionality of search space increases.

4. Modified FLC Generation Scheme by ($\mu+\lambda$) ES with Parental Population Sizing

One long-standing issue with FLC construction is the number of rules in the rulebase. Since most system plants have complex mathematical models unknown or unspecified at the time of design, how the fuzzy term sets and fuzzy rules of the FLC are to be assigned becomes a big problem. Moreover, given the fact that these two intertwined properties determine the performance of the controller, to evolve a FLC with variable rulebase structure seems to be the only solution.

Now that the modified ES has a parental population with an adjustable size, analogy can be drawn between the variable FLC and ES population. Following this thread of thought, each ES individual is comparable to a rule in the rulebase, and the whole population constitutes a corresponding FLC rulebase. For example, object variable portion of the individual representing Rule n in FLC (1) should consists of

$$\vec{X}_j(k) = (m_{1n}, \sigma_{1n}, m_{2n}, \sigma_{2n}, \cdots, \omega_{1n}, \omega_{2n}, \cdots) \quad (10).$$

This idea is identical to the SEFC design procedure appeared in [2]. Therefore, the authors attempt to combine the modified ES with the symbiosis procedure to make an even improved FLC generation scheme. Given a system plant G, a *FLC* to be evolve is done through the following steps:

Algorithm *Modified FLC Generation Scheme with Rulebase Sizing*

- $k = 0$;
- *initializing* $P(0)$ of size $\mu(0)$;
- *setting FLC* = object variable portion of $P(0)$;
- *evaluating FLC* over G;

while (termination criteria is not fulfilled) **do**
  - *recombination* and *self-adaptive mutation* to produce an offspring $P'(k)$ of size λ;

  - *evaluation* (**symbiotic** [2])
    for (each individual in $P'(k)$) **do**
      - *forming* a *tentative* FLC_T by the union of that individual and ($\mu(k)$-1) other individuals from $P''(k)=P(k) \cup P'(k)$;
      - *evaluating* FLC_T over G;
      - *assigning* fitness *incrementally* to all participating individuals;

    eof
    - *correcting* fitness of an individual by the number of tentative FLC_T's it has participated in;

  - *sizing* the parental population base on F_C of $P''(k)$ according to (9), getting $\mu_R(k)$;
  - *rule selection* (**elitist**)
    $P_R(k)$ = the best $\mu_R(k)$ individuals in $P''(k)$;
  - *setting* FLC_R = object variable part of $P_R(k)$;
  - *re-evaluating* FLC'';

  - *rulebase replacement* (**elitist**)
    if FLC_R is better than *FLC* **then**
      $P(k+1) = P_R(k)$;
      $FLC = FLC_R$;
      $\mu(k+1) = \mu_R(k)$;
    else
      $P(k+1) = P(k)$;
      $FLC = FLC$;
      $\mu(k+1) = \mu(k)$;
    fi

- $k = k + 1$;

od

Note that the above algorithm differs the most significantly from that in [2] at the additional steps of *re-evaluation* and *rulebase replacement*. As mentioned before, the modified FLC generation scheme practices ($\mu+\lambda$) ES at individual rule level and (1+1) ES at rulebase level. Thus, the candidate *FLC* is always the best controller ever evolved.

5. Simulation Examples

To demonstrate effectiveness of the modified FLC generation scheme, closed-loop step responses of three 2nd-order plants controlled by three sets of FLC's tuned by

(1) *Symbiotic* ($\mu+\lambda$) ES only, similar to SEFC scheme used in [2];
(2) *Symbiotic* ($\mu+\lambda$) ES with *Elitist Rulebase Replacement*;
(3) *Symbiotic* ($\mu+\lambda$) ES with *Elitist Rulebase Replacement* and *Rulebase Sizing*.

are compared.

All of the FLC generation schemes have the same setup including $\mu(0)=9$, $\lambda(0)=45$, and same initial populations. However, μ_{min} and μ_{max} for ES with parental population sizing are 2 and 9 respectively. *Global discrete recombination* and *self-adaptive mutation* are the ES operators. All ES runs stop at the 100^{th} generation.

Transfer functions of the three plants are listed in the following [32][33]:

$$G_A(s) = \frac{2}{s(s+1.4)+2} \qquad (11)$$

$$G_B(s) = \frac{2}{(s+1)(s+2)} \qquad (12)$$

$$G_C(s) = \frac{1}{s(0.1s+1)} \qquad (13)$$

Fitness function used to show performance of these FLC's is the *integral-of-time-multiplied absolute-error* (ITAE) criterion, which is defined as

$$\Phi(t) = \int t \cdot |error(t)| dt \qquad (14).$$

Simulations are discretized with a sampling period of 0.01s over 0s to10s. Inputs to FLC's are *error* and *error change rate*. Maximal absolute value of control output is 5. Simulation results with 10 different initial conditions are listed in Table 1. Closed-loop step responses of G_A are given in Figure 1.

6. Conclusions

From the table, it can be observed that FLC generation ES schemes practicing elitism do perform better than FLC generation scheme with symbiotic evolution only. Averagely speaking, FLC generation scheme with rulebase sizing also performs better than that having fixed number of rules. This again vindicates the notion that optimal size of the rulebase must come hand in hand with the actual arrangement of fuzzy membership parameters as well as mechanics of the control plant.

Finally, it should be noted that parental population sizing performed by the ES does not adjust size of offspring population. Nonetheless, it is obvious that

Plant	FLC Generation Scheme	(Initial) Population Size		Rulebase Fitness (i.e. ITAE)				Number of Rules (i.e. $\mu(100)$)	
		μ	λ	Best	Mean	Worst	STD	Best	Worst
G_A	Symbiotic Only	9	45	153.965	1095.0445	2579.1446	780.407	9	9
	Elitist Symbiotic	9	45	58.924	110.3504	196.4096	43.2317	9	9
	Elitist Symbiotic w. Rulebase Sizing	9	45	68.9464	113.3706	173.9028	28.8986	5	6
G_B	Symbiotic Only	9	45	178.6215	1163.1865	3644.5417	1242.0707	9	9
	Elitist Symbiotic	9	45	75.4386	118.9264	188.9223	35.7968	9	9
	Elitist Symbiotic w. Rulebase Sizing	9	45	33.8329	86.1327	159.972	35.2766	5	7
G_C	Symbiotic Only	9	45	1322.9457	6452.9614	17961.4596	5720.4673	9	9
	Elitist Symbiotic	9	45	15.1472	221.205	777.4147	273.3696	9	9
	Elitist Symbiotic w. Rulebase Sizing	9	45	20.9365	172.722	513.8526	138.2946	5	6

Table 1 Rulebase Fitness and Number of Rules Evolved by Different FLC Generation Schemes

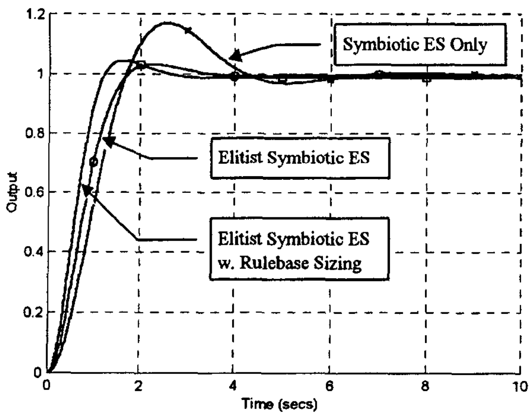

Figure 1 Closed-loop Step Response of G_A

size of the offspring population not only sets an upper bound to size of the parental population (hence size of the rulebase) but also dominates computational load of the algorithm. Therefore, the proposed FLC generation scheme should further be improved if an offspring population sizing mechanism could be devised. This is the main direction the authors are working toward at present.

7. References

[1] T.-Y. Huang and Y.-Y. Chen, "Parental Population Sizing in Evolutionary Strategies," to be appear in Congress on Evolutionary Computation 2001.

[2] 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: Cybern.*, vol. 30, no. 2, pp. 290–302, Apr. 2000.

[3] L. A. Zadeh, "Fuzzy Sets," *Inf. Contr.*, vol. 21, pp. 338–353, 1965.

[4] C. C. Lee, "Fuzzy Logic in Control Systems: Fuzzy Logic Controller—Part II," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 419–435, 1990.

[5] C. C. Lee, "Fuzzy Logic in Control Systems: Fuzzy Logic Controller—Part I," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, pp. 404–418, Mar./Apr. 1990.

[6] C. Elkan, "The Paradoxical Success of Fuzzy Logic," in *Proc. 11th Nat. Conf. Artificial Intelligence*, pp. 698–703, 1993.

[7] B. Kosko, *Neural Networks and Fuzzy Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1992.

[8] R. Jang, "Self-Learning Fuzzy Controllers Based on Temporal Back Propagation," *IEEE Trans. Neural Networks*, vol. 3, no. 5, pp. 714–723, 1992.

[9] R. Jang,, "Fuzzy Controller Design Without Domain Experts," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, pp. 289–296, 1992.

[10] C. Karr, "Genetic Algorithms for Fuzzy Controllers," *AI Expert*, vol. 6, pp. 26–33, Feb. 1991.

[11] D. T. Pham and D. Karaboga, "Optimum Design of Fuzzy Logic Controllers Using Genetic Algorithms," *J. Syst. Eng.*, vol. 1, no. 2, pp. 114–118, 1991.

[12] A. Homaifar and E. McCormick, "Full Design of Fuzzy Controllers Using Genetic Algorithms," in *SPIE Vol. 1766—Neural and Stochastic Methods in Image and Signal Processing*, pp. 393–404, 1992.

[13] M. A. Lee and H. Takagi, "Integrating Design Stages of Fuzzy Systems Using Genetic Algorithms," in *Proc. 2nd IEEE Int. Conf. Fuzzy Syst.*, pp. 612–617, 1993.

[14] K. C. Ng and Y. Li, "Design of Sophisticated Fuzzy Logic Controllers Using Genetic Algorithms," in *Proc. 3rd IEEE Int. Conf. Fuzzy Systems*, pp. 1708–1712, 1994.

[15] M. G. Cooper, "Evolving a Rule-Based Fuzzy Controller," *Simulation*, vol. 65, no. 1, pp. 67–73, 1995.

[16] D. A. Linkens and H. O. Nyonggesa, "Genetic Algorithms for Fuzzy Control—Part 1: Offline System Development and Application," *Proc. Inst. Elect. Eng. Contr. Theory Applicat.*, vol. 142, pp. 161–176, May 1995.

[17] D. A. Linkens and H. O. Nyonggesa, "Genetic Algorithms for Fuzzy Control—Part 2: Online System Development and Application," *Proc. Inst. Elect. Eng. Contr. Theory Applicat.*, vol. 142, pp. 177–185, May 1995.

[18] F. Bolata and A. Nowe, "From Fuzzy Linguistic Specifications to Fuzzy Controllers Using Evolution Strategies," in *Proc. 4th IEEE Int. Conf. Fuzzy Syst.*, vol. 3, pp. 1089–1094, 1995.

[19] O. Cordon and F. Herrera, "Evolutionary Design of TSK Fuzzy Rule-Based Systems Using (μ, λ) – Evolution Strategies," in *Proc. 6th IEEE Int. Conf. Fuzzy Syst.*, vol. 1, pp. 509–514, 1997.

[20] M. Fathi-Torbaghan and L. Hildebrand, "Model-Free Optimization of Fuzzy Rulebased Systems Using Evolution Strategies," *IEEE Trans. Syst., Man, Cybern.—Part B: Cybern.*, vol. 27, no. 2, pp. 270–277, Apr. 1997.

[21] Y. Jin, W. von Seelen, and B. Sendhoff, "On Generating FC Fuzzy Rule Systems from Data Using Evolution Strategies," *IEEE Trans. Syst., Man, Cybern.—Part B: Cybern.*, vol. 29, no. 6, pp. 829–845, Dec. 1999.

[22] A. Homaifar and E. McCormick, "Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 3, pp. 129–139, Apr. 1995.

[23] D. D. Leitch, "A New Genetic Algorithm for the Evolution of Fuzzy Systems," Ph.D. dissertation, Dept. Eng. Sci., Univ. Oxford, Oxford, U.K., 1995.

[24] N. E. Nawa and T. Furuhashi, "Fuzzy System Parameters Discovery by Bacterial Evolutionary Algorithms," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 5, pp.608–616, Oct. 1999.

[25] M. G. Cooper and J. J. Vidal, "Genetic Design of Fuzzy Controllers," in *Proc. 2nd Int. Conf. Fuzzy Theory Technology*, Durham, NC, 1993.

[26] L.-X. Wang, *Adaptive Fuzzy Systems and Control*, Prentice Hall, NJ, 1994.

[27] I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Stuttgart*, Germany: Frommann-Holzboog, 1973.

[28] H.-P. Schwefel, *Evolutionsstrategie und numerische Optimierung*, Dissertation, Technische Universität Berlin, Germany, May 1975.

[29] T. Bäck, U. Hammel, and H.-P. Schwefel, "Evolutionary Computation: Comments on the History and Current State," IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 3-17, Apr. 1997.

[30] D. B. Fogel, "An Introduction to Simulated Evolutionary Optimization," IEEE Trans. on Neural Networks, Vol. 5, No. 1, pp. 3-14, 1994.

[31] Jörg Heitkötter and David Beasley, eds., "The Hitch-Hiker's Guide to Evolutionary Computation: A list of Frequently Asked Questions (FAQ)," USENET: comp.ai.genetic. Available via anonymous FTP from rtfm.mit.edu/pub/usenet/news.answers/ai-faq/genetic/, 1999.

[32] F. Cheong and R. Lai, "Constraining the Optimization of a Fuzzy Logic Controller Using an Enhanced Genetic Algorithm," *IEEE Trans. Syst., Man, Cybern.—Part B: Cybern.*, vol. 30, no. 1, pp. 31–46, Feb. 2000.

[33] T.C. Chin, H.C. Hua, K.W. Yeo, and W.L. Yeo, "Genetic Algorithm-Based Optimal Fuzzy Controller," in *Proc. 2nd Int. Conf. Intelligent Systems*, pp. B135–B140, 1994.