Automatic Extraction of the Fuzzy Control System by a Hierarchical Genetic Algorithm

Jérôme Mendes[a,*], Rui Araújo[a], Tiago Matias[a], Ricardo Seco[a], Carlos Belchior[a]

[a] *Institute of Systems and Robotics (ISR-UC), and Department of Electrical and Computer Engineering (DEEC-UC), University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal*

Abstract

The paper proposes a new method to automatically extract all fuzzy parameters of a Fuzzy Logic Controller (FLC) in order to control nonlinear industrial processes. The main objective of this paper is the extraction of a FLC from data extracted from a given process while it is being manually controlled. The learning of the FLC is performed by a hierarchical genetic algorithm (HGA), from a set of process-controlled input/output data. The algorithm is composed by a five level structure, being the first level responsible for the selection of an adequate set of input variables. The second level considers the encoding of the membership functions. The individual rules are defined on the third level. The set of rules are obtained on the fourth level, and finally, the fifth level selects the elements of the previous levels, as well as, the *t*-norm operator, inference engine and defuzzifier methods which constitute the FLC. To optimize the proposed method, the HGA's initial populations are obtained by an initialization algorithm. This algorithm has the main goal of providing a good initial solution for membership functions and rule based populations, enhancing the GA's tuning. Moreover, the HGA is applied to control the dissolved oxygen in an activated sludge reactor within a wastewater treatment plant. The results are presented, showing that the proposed method extracted all parameters of the fuzzy controller, successfully controlling a nonlinear plant.

Keywords: Fuzzy Control, Hierarchical Genetic Algorithm, Activated sludge process, Fuzzy Knowledge Base

1. Introduction

Fuzzy control systems (FCSs) have been used for a wide variety of industrial systems and consumer products, attracting the attention of many researchers. Fuzzy logic controllers (FLCs) are rule-based systems which are useful in the context of complex ill-defined processes, especially those which can be controlled by a skilled human operator without any mathematical knowledge of the process's underlying dynamics (Herrera et al., 1995). FLCs are based on a set of fuzzy control rules that make use of people's common sense and experience. However, there still exist many difficulties in designing fuzzy systems to solve certain complex nonlinear problems.

In general, it is not easy to determine the most suitable fuzzy rules and membership functions to control the output of a plant, when the only available knowledge concerning the process is the empirical information transmitted by a human operator. Thus, a major challenge in current fuzzy control research is translating human empirical knowledge into FLCs. A possible candidate to meet this challenge is the application of the genetic algorithm (GA) approach to data extracted from a given process while it is being manually controlled.

GA's have been successfully applied to a wide variety of applications over the years. In particular, these algorithms have been applied in many automatic control problems, such as the development and tuning of FLCs. For that matter, they have been previously employed to select adequate sets of membership functions and fuzzy rules. Sharkawy (2010) developed a self-tuning PID control scheme with an application to antilock braking systems (ABSs) via combinations of fuzzy and genetic algorithms. Ali and Ramaswamy (2009) present an optimal fuzzy logic control algorithm for vibration mitigation of buildings using magneto-rheological (MR) dampers. A micro-genetic algorithm (m-GA) and a par-

*Corresponding author

Email addresses: jermendes@isr.uc.pt (Jérôme Mendes), rui@isr.uc.pt (Rui Araújo), tmatias@isr.uc.pt (Tiago Matias), cabelchior@isr.uc.pt (Carlos Belchior)

ticle swarm optimization (PSO) are used to optimize the FLC parameters. Alam and Tokhi (2008) proposed a GA-based hybrid fuzzy logic control strategy for input tracking and vibration reduction at the end point of a single-link flexible manipulator. For that matter, a GA is used to extract and optimize the rule base of the fuzzy logic controller. In Homayouni et al. (2009), a genetic fuzzy logic control methodology is used to develop two production control architectures: genetic distributed fuzzy (GDF) and genetic supervisory fuzzy (GSF) controllers. The GA is used to tune the input variable membership functions for the GSF and GDF controllers. Coban and Can (2010) designed a trajectory tracking genetic fuzzy logic controller for research reactors. Membership function boundaries and fuzzy control rule action weights were optimally determined by GAs.

The cited methods only optimize membership function parameters considering the other components of the fuzzy system fixed, such as implication, aggregation and defuzzifier methods. Other common limitation is the selection of the correct set of input variables. The variable selection process is usually manual and not accompanied with the accurate selection of the right time delays, probably leading to low-accuracy results. A variable with the correct delay may contain more information about the output, than one which does not consider any delay (Souza et al., 2010).

Delgado et al. (2001) introduced a hierarchical genetic algorithm (HGA) to optimize the parameters of Takagi-Sugeno fuzzy systems from available input/output data by means of a coevolutionary genetic algorithm and a neuro-based technique. Moreover, as an improvement of this methodology, Delgado et al. (2009) added a methodology for pre-selection of variables using an auxiliary criterion. However, the variable and delay selection are not jointly performed with the learning of the fuzzy model, which precludes the global optimization of the prediction setting.

The method proposed in this paper is based on the application of the HGA suggested by Delgado et al. (2009), although, applied to controller design. The main advances and differences contemplated in this work are the improvement of the whole hierarchical structure, automatically extracting the control fuzzy rules. First, a new hierarchical level, responsible for the selection of an adequate set of input variables and respective time delays is added; then the T-S fuzzy model approach (for an identification problem) was replaced by a standard fuzzy control system, i.e., the consequent part of the rules is represented by a membership function, rather than the T-S model proposed by Delgado et al. (2009); finally, an initialization algorithm based on (Andersen et al., 1997) is integrated to initialize the populations of the HGA, improving the convergence time.

Moreover, in order to validate and demonstrate the performance and effectiveness of the proposed algorithm, the control of the dissolved oxygen in an activated sludge reactor within a wastewater treatment plant (WWTP) is studied. WWTPs are large and complex non-linear systems subject to large disturbances in influent flow rate and pollutant load, together with uncertainties concerning the composition of the incoming wastewater (Belchior et al., 2012). In Belchior et al. (2012), a standard fuzzy controller is manually developed and applied to control the dissolved oxygen in an activated sludge reactor within a WWTP. Using this standard fuzzy controller, a learning dataset was obtained and then the proposed HGA is applied off-line with the aim of achieving a fuzzy controller with a response similar to the manually designed one. Once the fuzzy controller parameters are determined, the controller is applied to the same plant mentioned before, considering different control references, and comparing the results with the ones previously obtained using the standard FLC. The results show that the proposed method is able to extract all parameters of the fuzzy controller, enabling the successful control of the nonlinear plant.

The paper is organized as follows. Section 2 introduces the fuzzy logic system. The proposed HGA is described in Section 3. The HGA application and respective results are presented and analyzed in Section 4. Finally, remarks and conclusions are made in Section 5.

2. Fuzzy system

This section briefly overviews the main concepts of fuzzy control systems (Wang, 1997). A fuzzy system is a knowledge-based system defined by a group of **IF-THEN** rules, which can be used to implement fuzzy controllers. The following example illustrates such a rule:

$$\begin{aligned}&\text{IF the } \textit{temperature} \text{ is } \textit{cold},\\&\text{THEN turn } \textit{down} \text{ the } \textit{speed} \text{ of the fan,}\end{aligned} \tag{1}$$

where *temperature* and *speed* are input and output variables, respectively. These variables are characterized by the fuzzy sets A, through a mapping defined by $\mu_A(x) = \mathbf{U} \rightarrow [0, 1]$, for which, *cold* and *down*, are referred as linguistic terms. **U** is the universe of discourse of the variable.

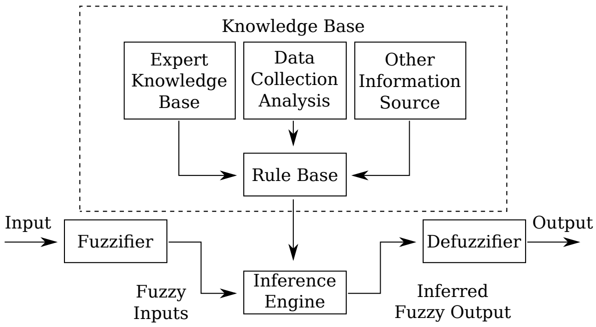

Figure 1: Basic configuration of fuzzy logic systems.

Fuzzy systems are constituted by a group of four main elements: knowledge base, fuzzifier, fuzzy inference engine and defuzzifier, as shown in Figure 1.

The knowledge base is composed by a set of N fuzzy IF-THEN rules R_j in the generic form

$$R_j : \text{IF } x_1 \text{ is } A_1^j, \text{ and} \dots \text{and } x_n \text{ is } A_n^j \text{ THEN } u \text{ is } B_j, \quad (2)$$

where $j = 1, 2, \cdots, N$; x_i $(i = 1, 2, \dots, n)$ are the input variables of the fuzzy system, u is the output variable and A_i^j and B_j are the linguistic terms characterized by the fuzzy membership functions $\mu_{A_i^j}(x)$ and $\mu_{B_j}(u)$, respectively.

The fuzzifier is the fuzzy system element responsible for mapping the real values of the input linguistic variables, x, into corresponding fuzzy sets described by membership functions X. In this paper, the only utilized fuzzifier is the singleton fuzzifier (Wang, 1997).

The next element, the fuzzy inference engine (FIE), uses the collection of fuzzy IF–THEN rules to map the input fuzzy set X into the rule consequent fuzzy sets B_j. The collection of fuzzy outputs of the rules are then combined into an overall inferred fuzzy output U. In this paper, to process the antecedent part of the rule, only propositions connected by the fuzzy AND operator (T-norms) are considered.

Finally, the defuzzifier is responsible for mapping a fuzzy set U into a real-valued output, u^*.

3. Hierarchical genetic algorithm

This section describes the proposed algorithm. The main goal of this work is to develop a FLC that does not require prior explicit expert knowledge. To do so, the work proposes an automatic method based on a HGA for the extraction of all fuzzy parameters of a FLC from a dataset obtained from an existing controller (human or automatic).

Some biologically inspired algorithms, such as genetic algorithms (GAs), ant colony optimization (ACO), particle swarm optimization (PSO), have been proved efficient in optimization problems. GAs are search methods that are inspired on natural evolution, selection, and survival of the fittest in the biological world. PSO is inspired in the social behavior of living organisms such as bird flocking or fish schooling. ACO is a multiagent approach that simulates the foraging behavior of ants. All algorithms could be used to design the T-S fuzzy models. However, because GAs provide a robust search with the ability to find near optimal solutions in complex and large search spaces ((Cordón et al., 2001), (Herrera, 2008)), GAs are a useful soft computing technique to design T-S fuzzy models. Other advantages in the use of GAs in the design of T-S fuzzy models are: GAs are simple to implement, they have the possibility of using different types of solution encoding (e.g. for different parts of the model), and they are adaptive, which means that they have the ability to learn, accumulating relevant knowledge to solve optimization problems (Kasabov, 1996).

A HGA will be used instead of a GA with just one optimization level due to the complexity of the problem. It is well known that computation, search, and optimization problems become more difficult to solve when the dimensionality increases (curse of dimensionality), and, therefore, when more complex design decisions involving a large number of parameters must be made, a global formulation of the problem representing all the parameters in just one optimization level can be inadequate.

3.1. Genetic algorithms

Genetic algorithms are a family of computational models introduced by Holland (1975), that are inspired by the natural processes of evolution. The GA's starts with the initialization of the population. In this step, the potential solutions of the problem are encoded into chromosomes (individuals) to form the population. In the evolutionary phase, the population is evolved using the genetic operators, giving, as in nature, more reproductive opportunities to the best chromosomes.

The main advantages of GAs are the possibility of solving every optimization problem where the solutions can be represented in chromosomes, the possibility to solve problems with multiple solutions and with large number of variables, the absence of requirements regarding the availability of mathematical knowledge of the optimization problem, the resistance to being trapped in local minima, and the possibility of the use of parallel computing. The main disadvantages are that the optimality of the solution is not guaranteed, and that

GAs cannot be used in real-time applications because the convergence time is unknown.

3.2. Hierarchical architecture

The proposed coevolutionary model is constituted by five hierarchical levels (Figure 2). The first level represents the population of the set of input variables and their respective time delays. The population of the second level is given by all fuzzy system membership functions, i.e., the antecedent and consequent membership functions which constitute the fuzzy rules. The individual rule population is defined on the third level and the rules set population is obtained on the fourth level. Finally, the fifth level represents the population with the indices of the selected elements of the previous levels, as well as, the antecedent aggregation method, inference engine and defuzzifier that are used on the fuzzy controller. The detailed description of each level is given below:

Level 1*:* it is formed by the population of the set of input variables, and delays, which will be used to design the fuzzy controller. Its chromosome is represented by binary encoding, where each allele (element of the chromosome located in a specific position) corresponds to each input variable/delay pair (see Figure 2). The length of the chromosome is given by the total number of pairs of system variables and respective delays that are considered as possible candidates to be used as inputs for the fuzzy system.

Level 2*:* contains every membership function defined in the universe of the variables involved. The chromosome is formed by the aggregations of all partition sets associated with each input and output variable. An example of its structure can be seen in detail in Figure 3, where each first allele uses integer encoding to represent the type of membership function. In this paper, three different types of possible membership functions are permitted: trapezoidal ($S_k = 1$), triangular ($S_k = 2$) and Gaussian ($S_k = 3$). Alleles 2-5 use real encoding to represent the parameters of the membership function. Considering the kth membership function, for trapezoidal functions, alleles 2-5 are converted into absolute values, given by (see Figure 3):

$$m_{1k} = m_{2,k-1} + C_{1k}, \quad (3)$$
$$m_{2k} = m_{1k} + C_{2k}, \quad (4)$$
$$b_{1k} = m_{1k} - L_k, \quad (5)$$
$$b_{2k} = m_{2k} + R_k. \quad (6)$$

For triangular membership functions, the center is found by the average between m_{1k} and m_{2k} (see Figure 3). For Gaussian functions, the central value is calculated the same way as in the triangular case, and the dispersion is given by $\sigma_{kj} = \Delta_k/3$ where $\Delta_k = (L_k + R_k)/2$. As can be seen in Figure 2, Level 2 contains all membership functions parameters.

Level 3*:* it is constituted by a population of individual rules. The length of the chromosome is determined by the number of input variables selected by Level 1, plus an additional allele that characterizes the output variable. The chromosome is represented by integer encoding, where each allele contains the index of the corresponding antecedent and consequent membership function. Null values indicate the absence of membership function and are only considered for antecedent indices.

Level 4*:* it is constituted by a set of fuzzy rules, where each allele contains the index of the corresponding individual rule that has been included in the set. The chromosome is represented by integer encoding, where once again, null values indicate that the corresponding allele does not contribute to the inclusion of any rule in the set of fuzzy rules. The length of the chromosome is determined by the maximum number of fuzzy rules.

Level 5*:* it represents a fuzzy system, i.e., all the information required to develop the fuzzy controller is contemplated on this level. The chromosome is represented by integer encoding and is constituted by seven alleles. The first allele represents the t-norm operator, used to implement the fuzzy "and" operations used for aggregation in rule antecedents. For this matter, the GA selects from between three types of t-norms: (1) product, (2) minimum and (3) bounded difference (other aggregation operators can be used, see more in (Wang, 1997)). The second allele indicates the index, k, of the set of rules specified on Level 4. The third allele contains the qth partition set given by Level 2. The fourth allele represents the index, l, of the set of input variables selected on Level 1. The fifth allele specifies the type of implication operator used. For this study the implemented implication methods where the (1) Mamdani product and the (2) Mamdani minimum (Wang, 1997). The sixth allele indicates the type of aggregation operator, where the following operators have been used as possibilities: (1) Maximum, (2) Bounded sum, and (3) Normalized sum. Finally, the seventh allele is responsible for determining the type of defuzzifier. The considered defuzzifiers are: (1) center of gravity (COG), (2) first of maximum (FOM), (3) last of maximum (LOM) and (4) mean of maximum (MeOM). All these operators (t-norm, implication, aggregation and defuzzifier) can be consulted in (Mendes et al., 2011).

An example of the encoding and the hierarchical relations is given in Figure 2. In this example the ith set of Level 5 indicates that the fuzzy system uses the product

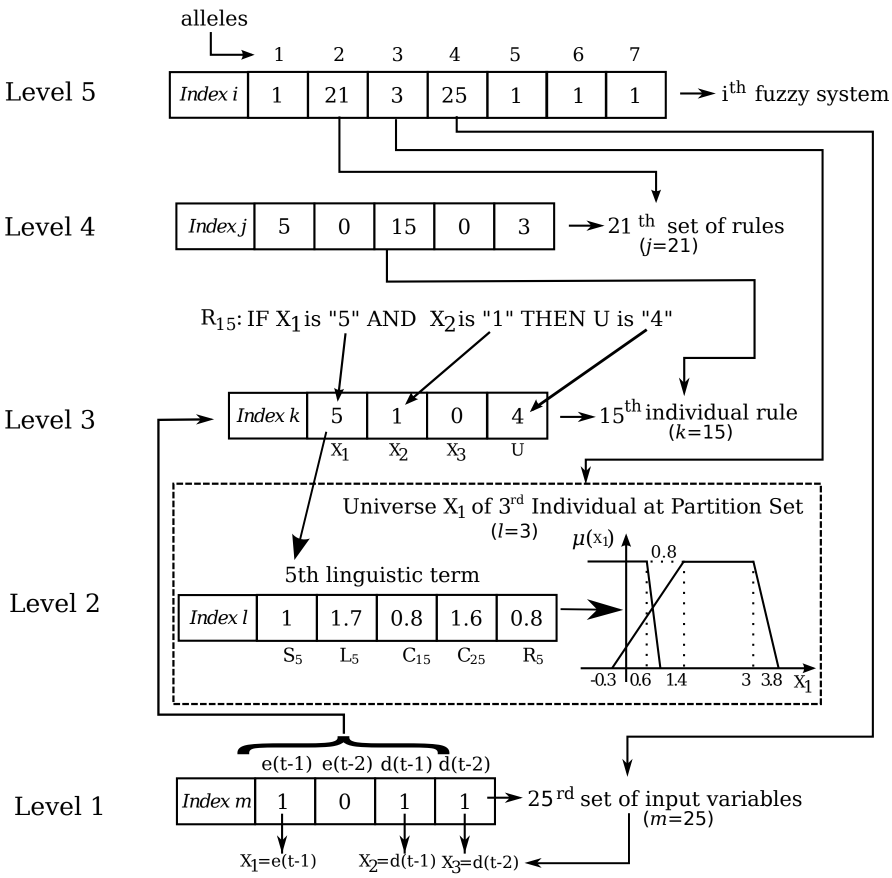

Figure 2: Encoding and hierarchical relations among the individuals of different levels of the genetic hierarchy.

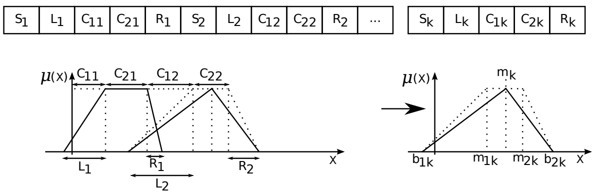

Figure 3: Level 2's membership function encoding.

t-norm operator as the antecedent aggregation method (allele 1), the 21st set of fuzzy rules of Level 4 (allele 2), the 3rd partition set of Level 2 (allele 3), the 25th set of selected input variables and delays in Level 1 (allele 4), product implication (allele 5), maximum aggregation (allele 6), and center of gravity defuzzification (allele 7). The 21st set of fuzzy rules (Level 4) contains the 5th, 15th, and 3rd individual rules, where the 15th individual rule (Level 3) is composed by two input variables with linguistic terms 5 and 1, respectively, and one output variable corresponding to linguistic term 4, i.e.:

$$\begin{aligned} R_{15}: \quad & \text{IF } x_1(t) \text{ is “5” and } x_2(t) \text{ is “1”} \\ & \text{THEN } u(t) \text{ is “4”}. \end{aligned} \tag{7}$$

The linguistic term “5” is defined in the 3rd chromosome of Level 2, and the input variables x_1, x_2, and x_3 are determined on Level 1.

Let i_{max}, j_{max}, k_{max}, l_{max} and m_{max}, be the maximum number of chromosomes at Levels 5, 4, 3, 2, and 1, respectively. The fitness evaluation of each individual of the hierarchical population at each level is defined as follows:

- Fuzzy system (Level 5):

$$J_5^i = \frac{1}{(1 + \frac{2dim}{T})\mathrm{MSE}(\mathbf{u}, \hat{\mathbf{u}}^i)}, \tag{8}$$

where $\mathrm{MSE}(\mathbf{u}, \hat{\mathbf{u}}^i) = \frac{1}{T}\sum_{t=1}^{T}(u(t) - \hat{u}^i(t))^2$ is the mean square error between the target control output $\mathbf{u}$ and estimated control output $\hat{\mathbf{u}}^i$ obtained with individual i, and $(1+\frac{2dim}{T})$ is the Akaike's information criterion (AIC) (Espinosa et al., 2004) which penalizes the more complex individuals to avoid overparameterization. The complexity is measured by *dim* which represents the total number of the parameters of the FLC tuned by the HGA.

- Rule base (Level 4):

$$J_4^j = max(J_5^{a_1}, \cdots, J_5^{a_p}), \tag{9}$$

where $\{a_1, \ldots, a_p\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain rule-base j (set of fuzzy rules);

- Individual rule (Level 3):

$$J_3^k = mean(J_4^{b_1}, \cdots, J_4^{b_q}), \tag{10}$$

where $\{b_1, \ldots, b_q\} \subseteq \{1, \ldots, j_{max}\}$ is the subset of all chromosomes of Level 4 that contain individual rule k;

- Partition set (Level 2):

$$J_2^l = max(J_5^{c_1}, \cdots, J_5^{c_r}), \tag{11}$$

where $\{c_1, \ldots, c_r\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain partition set l;

- Inputs and delays selection (Level 1):

$$J_1^m = max(J_5^{d_1}, \cdots, J_5^{d_s}), \tag{12}$$

where $\{d_1, \ldots, d_s\} \subseteq \{1, \ldots, i_{max}\}$ is the subset of all chromosomes of Level 5 that contain the m-th selection of inputs and delays.

The main steps of the GA algorithm used to learn/improve the fuzzy controller parameters are presented in Algorithm 1. Each level of the genetic hierarchy is evolved separately as an independent genetic algorithm using its own population and its own fitness function. However, since the values of the fitness functions of each level depend on all the other populations, then the evolution of each level is also influenced by the evolution of every other level.

Algorithm 1 Proposed algorithm.

input: Maximum number of chromosomes for each level, i_{max}, j_{max}, k_{max}, l_{max}, m_{max}; Number of generations Gen_{max}.
output: An optimized fuzzy system.

```
procedure
    Initialize populations of all levels;
    Gen ← 1;
    while Gen < Gen_max do
        for all i = 1, ..., i_max do
            Evaluate J_5^i using Eq. (8);
        end for
        for all j = 1, ..., j_max do
            Evaluate J_4^j using Eq. (9);
        end for
        for all k = 1, ..., k_max do
            Evaluate J_3^k using Eq. (10);
        end for
        for all l = 1, ..., l_max do
            Evaluate J_2^l using Eq. (11);
        end for
        for all m = 1, ..., m_max do
            Evaluate J_1^m using Eq. (12);
        end for
        Select two chromosomes on each of the Levels 5, 4, 3, 2, and 1, according to (optimizing) J_5^i, J_4^j, J_3^k, J_2^l, and J_1^m [eqs. (8)-(12)] to be parents;
        Obtain new children for all levels from the selected five pairs of parents;
        Perform mutation in all new children for all levels;
        Replace the current population with the new evolved population;
        Gen ← Gen + 1;
    end while
end procedure
```

3.3. Genetic algorithm operators

The methods used for initialization, selection, crossover, mutation and replacement in the genetic algorithm of the proposed approach are described below.

Initialization: The use of random initialization of the population in a GA may result in a very exhausting optimality search, requiring more iterations to attain convergence. So in order to obtain a initial satisfactory starting point, reducing the computational cost and increasing the algorithm's performance, a initialization algorithm (Algorithm 2) based on the algorithm proposed in (Andersen et al., 1997) was used. The solution obtained by Algorithm 2 was used to initialize the first individual of Level 1 and all the individuals of Levels 2-4. Furthermore, the initial solution was also used to initialize alleles 2 (set of rules), 3 (partition set), and 4 (set of input variables) of the first individual of Level 5. The

Algorithm 2 Initialization algorithm used to initialize the HGA fuzzy rules and membership functions.

procedure

for all sampling instances $t = 1, \ldots, T$ **do**

Apply the input vector $\mathbf{x}(t) = [x_1(t), \ldots, x_m(t)]^T$ to the system to be modeled (in this case the controller) and obtain the data sample $(\mathbf{x}(t), u(t))$, where $u(t)$ is the output scalar;

end for

Using the collected data samples $(\mathbf{x}(t), u(t))$, select a set of variables $\{x_{v_1}(t - d_1), \ldots, x_{v_n}(t - d_n)\}$ which will be used to initialize the HGA; $v_i \in \{1, \ldots, m\}$, and d_i is an integer, for $i = 1, \ldots, n$.

for all all input variables $i = 1, \ldots, n$ **do**

Specify the minimum and maximum values, x_i^- and x_i^+, of input variable x_i (i.e. the universe of discourse limits);

Define the number of individual membership functions K_i for each input variable x_i;

for all $k = 1, \ldots, K_i$ **do**

Determine the initial center, $c_{A_i^k}$, of each antecedent membership function $\mu_{A_i^k}$ (assuming a Gaussian type) by distributing them uniformly over the universe of x_i using an interval width of $\left(x_j^+ - x_j^-\right)/K_i$. In this way, the centers are chosen such that they are distributed evenly over the universe of x_i;

For each antecedent membership function $\mu_{A_i^k}$, determine the associated dispersion, σ_i^k, where σ_i^k is given by $\sigma_i^k = (\alpha_i/K_i) \,.\, \left(x_j^+ - x_j^-\right)$ and α_i is the scaling factor which influences the Gaussian membership function width.

end for

end for

Form the rule-based system containing a different rule for every possible combination of antecedent membership function, where the total number of rules is $N = \prod_{i=1}^{n} K_i$;

for all $j = 1, \ldots, N$ **do**

Obtain the centers, c_{B_j}, of the consequent membership functions by using the heuristic method, $c_{B_j} = \left(\sum_{t=1}^{T} u(t) \prod_{i=1}^{n} \mu_{A_i^j}(x_i(t))\right) / \left(\sum_{t=1}^{T} \prod_{i=1}^{n} \mu_{A_i^j}(x_i(t))\right)$;

end for

Build the output membership functions (assuming for example a triangular shape) where the centers are given by c_{B_j}, and the triangular aperture can be represented by $(u^+ - u^-) / (N.l)$, where u^+ and u^- are the upper and lower limits of the output universe of discourse, respectively, and l is a scaling factor (e.g. $l = 2, 4, 8, \cdots$).

end procedure

remaining alleles (1, 5, 6, and 7) of this individual were initialized with product t-norm, Mamdani product implication, maximum aggregation and center of gravity deffuzification. The remaining individuals of Levels 1 and 5 were randomly chosen with uniform distribution.

Selection: The roulette wheel selection method was used. The principle of roulette selection consists in a linear search of individuals through a roulette wheel, where the wheel slots are weighted in proportion to the individuals fitness values (Sivanandam and Deepa, 2007).

For *Crossover* the Single Point crossover technique was used (Sivanandam and Deepa, 2007). The process consists of taking two parents and producing two offspring solutions (childs) from them. For the first child, the crossover process generates a random point of crossover (uniform distribution is used), R_r, and the child will receive the alleles from 1 to R_r from the first parent and the rest of the alleles are received from the second parent. The second child is constituted by the remaining alleles of the parents.

Mutation is used to maintain the diversity of the population and to prevent the algorithm from being trapped in local minima. In real and integer encoded chromosomes uniform mutation was used, where the value of one randomly selected allele of the chromosome is replaced by a uniform random value selected between the upper and lower bounds defined for that allele. In binary-encoded chromosomes, the flip bit mutation technique is used. In this technique the value of a random allele (uniform distribution is used) is inverted.

Replacement: The weakest individuals replacement technique was used. It consists in replacing the individuals of the old generation with weakest fitness by the new individuals (obtained by the crossover and mutation operators) in order to form the new population. So, the new generation is constituted by the two new individuals and the survivors from the old population.

4. Results

This section addresses the application of the HGA for automatic extraction of the fuzzy parameters of a fuzzy controller in order to control the dissolved oxygen (DO) in an activated sludge reactor within a wastewater treatment plant.

First, a dataset of the plant being controlled by any controller (human or automatic) is obtained with the aim of providing a set of input/output data necessary for the HGA to learn the FLC's parameters. The dataset is obtained by applying the FLC proposed by Belchior et al. (2012) to the process, and recording along time the values of the input and output variables that constitute the dataset. The HGA is then applied to the obtained dataset, with the aim of determining a controller with a response similar to the one that is being replicated. Note that in this simulation, the aim is learn a FLC output, but it could be to any other controller, as for example, in a process controlled by a human operator. The results were obtained by considering that the crossover and mutation probabilities are 80% and 10%, respectively, the number of generations is $Gen_{max} = 1000$, and the number of chromosomes for each level of the architecture are: $i_{max} = 100$, $j_{max} = 80$, $k_{max} = 200$, $l_{max} = 15$, and $m_{max} = 200$. These parameters were tuned by means of experimentation.

4.1. Dissolved oxygen control

Wastewater treatment plants are large and complex nonlinear systems subject to large disturbances in the influent flow rate and pollutant load, together with uncertainties concerning the composition of the incoming wastewater (Belchior et al., 2012). The proposed methodology is applied on the control of the dissolved oxygen (DO) in an activated sludge reactor within a WWTP. The control architecture proposed in this paper is tested in the Benchmark Simulation Model no.1 (BSM1). BSM1 is a platform-independent simulation environment developed under COST Action 682 and 624 that is dedicated to the optimization of performance and cost-effectiveness of wastewater management systems (Jeppsson and Pons, 2004).

A general overview of the BMS1 plant is presented in Figure 4. The biological reactor is distributed over five reactors connected in cascade. Reactors 1 and 2 are non-aerated compartments with a volume equal to 1000 [m3] each. Reactors 3, 4, and 5 are aerated and their volumes are approximately equal to 1333 [m3] each. Reactors 3 and 4 have a fixed oxygen transfer coefficient, and the DO of reactor 5 should be controlled by manipulation of the oxygen transfer process from an aerator to the activated sludge inside the biological reactor, named $KLa5$. The DO concentration is measured on reactor 5 and is controlled by manipulation of the $KLa5$ on the same reactor. For more details about the BMS1 plant, references (Jeppsson and Pons, 2004; Belchior et al., 2012) are recommended. The sampling period is 15 [min], and the simulations have a maximum of 14 [days].

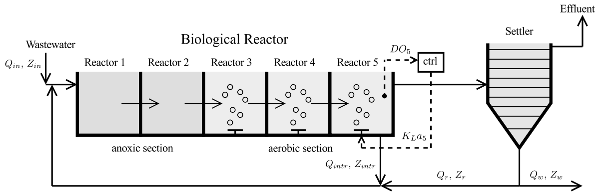

Figure 4: General overview of the BSM1 plant (Belchior et al., 2012).

4.2. HGA application and results analysis

The input/output dataset is obtained by controlling the DO concentration with the FLC described in (Belchior et al., 2012), named as FLC-BSM1. The dataset was obtained, while controlling the BSM1 plant, by extracting the command signal, $KLa5(t)$, the tracking error of the DO concentration, $E(t) = DO_{ref}(t) - DO(t)$, and the derivative of $E(t)$, $\Delta E(t)$, where $DO_{ref}(t)$ is the desired reference for $DO(t)$. The first four delays of $E(t)$ and $\Delta E(t)$, are also included in the learning dataset, i.e. $E(t-1), \cdots, E(t-4)$ and $\Delta E(t-1), \cdots, \Delta E(t-4)$, allowing a better selection of the FLC's input variables. The input variables were divided into two groups: one group for variables $[E(t), E(t-1), E(t-2), E(t-3), E(t-4)]$ and the other for $[\Delta E(t), \Delta E(t-1), \Delta E(t-2), \Delta E(t-3), \Delta E(t-4)]$. Level II of the hierarchical genetic fuzzy system (Sec. 3) was configured so that (i) variables belonging to the same group have the same range of possible values, and (ii) for each individual, all variables within each group were forced to share the same partition set. The reference signal used to obtain the dataset was

$$DO_{ref}(t) = \begin{cases} 1, & 0 < t \leq 10\,[\text{days}], \\ 2, & 10\,[\text{days}] < t \leq 12\,[\text{days}], \\ 3, & 12\,[\text{days}] < t \leq 14\,[\text{days}]. \end{cases} \quad (13)$$

Figure 5 shows the response obtained with the FLC-BSM1 that was used to construct the dataset. Figure

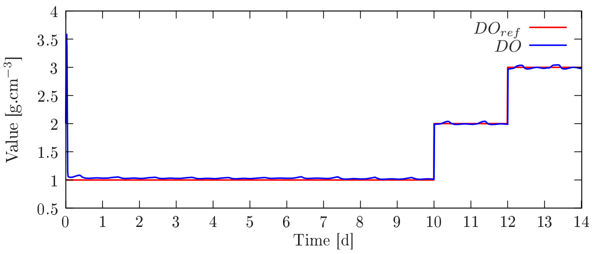

Figure 5: FLC-BSM1 performance used to compile the learning dataset.

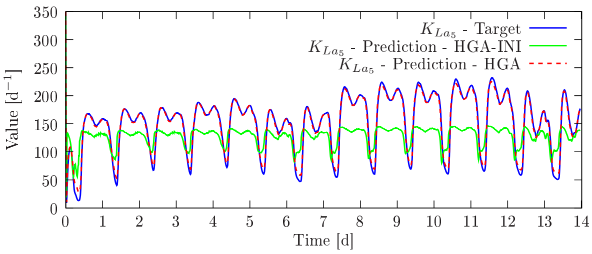

Figure 6: FLC target (FLC-BSM1) and command signals learned by the initialization method (HGA-INI), and by HGA (FLC-HGA).

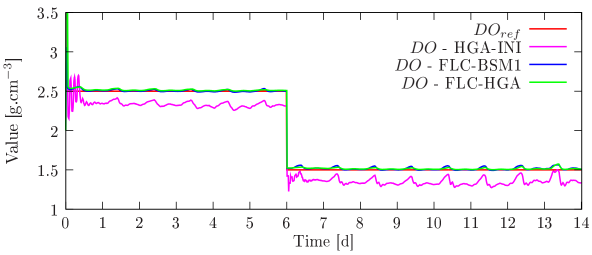

Figure 7: Results obtained by controlling the BSM1 plant for a reference different from the one used for training the FLC-HGA.

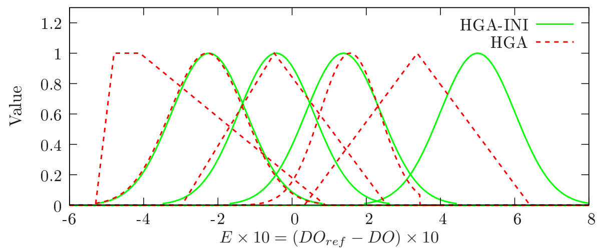

Figure 8: Membership functions of the partition set of E obtained using the initialization method (HGA-INI) and using the proposed methodology (HGA).

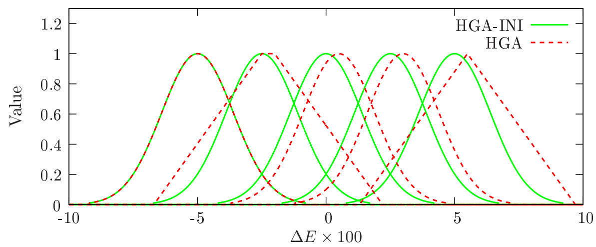

Figure 9: Membership functions of the partition set of ΔE obtained using the initialization method (HGA-INI) and using the proposed methodology (HGA).

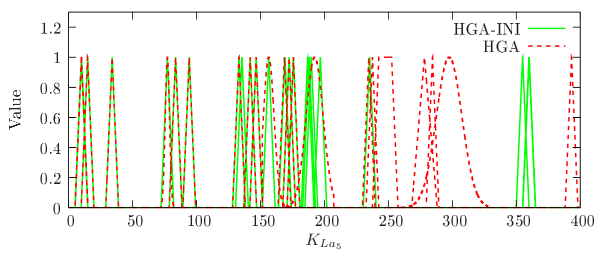

Figure 10: Membership functions of the partition set of $KLa5(t)$ obtained using the initialization method (HGA-INI) and using the proposed methodology (HGA).

6 shows the target response (FLC-BSM1), the response of the initial FLC (HGA-INI) obtained by the initialization method of Algorithm 2, and the command signals learned by the proposed method (FLC-HGA). Figure 7 compares the responses obtained by FLC-BSM1, by HGA-INI, and by FLC evolved by the HGA (FLC-HGA) for a reference signal different from the one used to generate the dataset that was employed for training the FLC-HGA.

The membership functions and rules obtained by the HGA-INI initialization method are shown in Figures 8-10, and Table 1, respectively. In Table 1 the numbers 1 to 25 represent the linguistic terms of the respective variable. Note that the linguistic terms of partition sets of E and ΔE are 1 to 5, and the linguistic terms of partition set of $KLa5$ are 1 to 25. The FLC's membership functions and fuzzy rules obtained from the operation of the proposed methodology (FLC-HGA) are shown in Figures 8-10, and Table 2, respectively. In Table 2 null values indicate the absence of membership function. Finally, Figure 11 demonstrates the fitness function value evolution along the HGA's iterative learning process.

In Figure 6 it can be seen that the performed identification was still sufficient to obtain a controller with a response resembling the one that was intended to be achieved. It can also be verified that the HGA improved the initial FLC (HGA-INI) response given by the initialization method of Algorithm 2.

Table 1: Initialization method and HGA fuzzy rule structure.

$E(t)$	$\Delta E(t)$	$KLa5(t)$
1	1	1
1	2	2
1	3	3
1	4	4
1	5	5
2	1	6
2	2	7
2	3	8
2	4	9
2	5	10
3	1	11
3	2	12
3	3	13
3	4	14
3	5	15
4	1	16
4	2	17
4	3	18
4	4	19
4	5	20
5	1	21
5	2	22
5	3	23
5	4	24
5	5	25

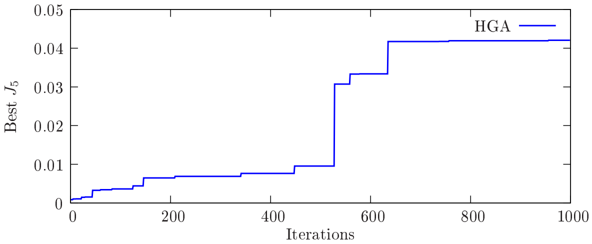

Figure 11: HGA's best individual fitness function value evolution along the generations.

As can be seen in Figure 7 the response of the FLC attained by the HGA (FLC-HGA) is approximate to the desired one. It is also verified that the result obtained with the HGA has a response closer to the reference signal than the result obtained by the initialization method of Algorithm 2. This is also supported by the results presented in Figure 11 which shows the time evolution of the fitness function.

Furthermore, when viewing Table 2 and Figures 8-10, it can also be observed the effort of optimization of the set of fuzzy rules and membership functions by the HGA. Besides choosing the adequate set of rules and membership functions, the HGA also identified the most adequate set of input variables as being $E(t)$,

Table 2: Fuzzy rule structure obtained/learned by the proposed method (FLC-HGA).

Selected FLC inputs			FLC output
$E(t)$	$E(t-1)$	$\Delta E(t)$	$KLa5(t)$
2	0	1	6
5	0	2	22
5	0	1	21
3	3	2	5
2	4	5	6
4	0	4	19
1	0	5	5
2	0	1	6
3	0	3	16
1	0	5	5
3	0	3	16
3	0	5	15
3	0	5	15
2	2	2	4
5	0	2	22
4	0	4	19
1	0	3	1
2	2	2	4
3	3	2	5
3	3	2	5
3	0	3	16
3	0	5	15
2	0	1	6
3	0	1	11
5	0	2	22

$E(t-1)$ and $\Delta E(k)$, and the following parameters of the fuzzy system were identified: product t-norm, Mamdani minimum implication, maximum rule aggregation, and center of gravity defuzzification.

4.3. Sensitivity analysis

In order to discuss the performance of the proposed method, a sensitivity analysis with regard to the effect of the parameters extracted by the HGA is made. All the parameters obtained by optimization in the FLC-HGA, i.e., the antecedent partition set, the consequent partition set, the linguistic terms of the antecedent, the linguistic terms of the consequent, and the t-norm, implication, aggregation, and defuzzification operators are analyzed. In this section all random variables have uniform distribution.

Each part of the obtained FLC-HGA is separately analyzed, i.e. when the sensitivity analyses of one part of the learned FLC-HGA is performed, the other parts are considered fixed. The sensitivity analysis is made by the following steps:

- All the parameters of the antecedent partition set (m_{1k}, m_{2k}, b_{1k}, and b_{2k} (3)-(6)) vary randomly in the following ranges: $[-5, 5]\,[\%]$,

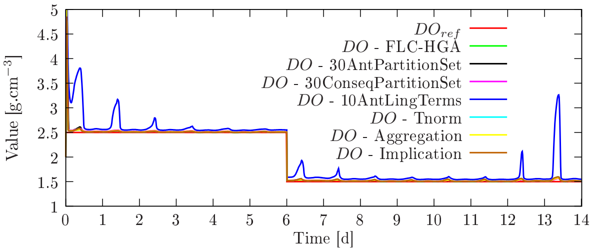

Figure 12: Results obtained by controlling the BSM1 plant for three cases of the sensitivity analysis (see main text). DO-10AntLingTerms (blue) is the only response that significantly diverges from DO_{ref}.

$[-10,-5] \cup [5,10]\,[\%]$, $[-15,-10] \cup [10,15]\,[\%]$, $[-20,-15] \cup [15,20]\,[\%]$, $[-25,-20] \cup [20,25]\,[\%]$, and $[-30,-25] \cup [25,30]\,[\%]$.

- The same procedure applied to the antecedent partition set is applied on the consequent partition set.

- A number of linguistic terms among all the linguistic terms of all the input variables were randomly selected to be randomly changed. The number of linguistic terms randomly selected to be randomly changed is given by the 5%, 10%, 15%, 20%, 25%, and 30% of the total number of linguistic terms of the antecedents available to be used in the learned fuzzy rules. For the fuzzy rules in Table 2 the total number of linguistic terms of the antecedents is 25×3 (Number of rules $\times$ number of input variables). The linguistic terms of the input variables are randomly changed to one of the linguistic terms 0-5, where the null value indicates the absence of membership function.

- An analysis similar to the one performed for the antecedent linguistic terms is also made on the linguistic terms of the consequent. The randomly selected consequent linguistic terms are randomly changed to one of the linguistic terms 1-25.

- The learned t-norm operator is changed randomly to a different t-norm operator. The same procedure is made for the implication operator, aggregation operator, and defuzzification operator.

In Figure 12 it can be seen the results of the control of the DO for the attained worst cases (i.e., lowest $1/MSE$, attained for $[-30,-25] \cup [25,30]\,[\%]$ parameter changes) of the antecedent and consequent partition sets variations (labels "DO-30AntPartitionSet", and "DO-30ConseqPartitionSet", respectively), and the results attained for the cases of including random changes

Table 3: Sensitivity analysis of the parameters extracted by the FLC-HGA when controlling the BSM1 plant. As a baseline for comparison, the FLC-HGA has $1/MSE = 174.587$ in the test control.

Part of the learned FLC-HGA analyzed	Variation	$1/MSE$
Antecedent partition set	$[-5,5]\,[\%]$	124.5716
	$[-10,-5] \cup [5,10]\,[\%]$	91,4354
	$[-15,-10] \cup [10,15]\,[\%]$	90,3751
	$[-20,-15] \cup [15,20]\,[\%]$	87,4639
	$[-25,-20] \cup [20,25]\,[\%]$	52,1271
	$[-30,-25] \cup [25,30]\,[\%]$	38.1754
Consequent partition set	$[-5,5]\,[\%]$	157.9354
	$[-10,-5] \cup [5,10]\,[\%]$	133,0071
	$[-15,-10] \cup [10,15]\,[\%]$	116,2449
	$[-20,-15] \cup [15,20]\,[\%]$	116,8461
	$[-25,-20] \cup [20,25]\,[\%]$	89,2929
	$[-30,-25] \cup [25,30]\,[\%]$	52.1721
Linguistic terms of the antecedents	5%	35.0451
	10%	10.1196
	15%	5.5380
	20%	5.5380
	25%	0.2795
	30%	0.2885
Linguistic terms of the consequent	5%	104.7581
	10%	57.6163
	15%	33.3169
	20%	2.3388
	25%	0.2751
	30%	0.2732
t-norm operator	Randomly changed	73.6559
Implication operator	Randomly changed	73.5319
Aggregation operator	Randomly changed	72.5152
Defuzzification operator	Randomly changed	0.2827

in the t-norm, implication, and aggregation operators (labels "DO-Tnorm", "DO-Implication", and "DO-Aggregation", respectively). These results have a very similar control response, and the differences to the (unperturbed) learned FLC are not significant. Taking into account the values of $1/MSE$ in Table 3, it is seen that the results are more sensitive to changes in the antecedent partition set than in the consequent partition set. Taking into account the values of $1/MSE$ in Table 3, and mainly the responses in Figure 12, it is concluded that the control response of the FLC is not too affected (not very sensitive) to variations on the antecedent and consequent partition sets, and on t-norm, implication, and aggregation operators when compared to the performance of the attained optimal FLC-HGA ($1/MSE = 174.587$). The result of the variation of 10%

of the linguistic terms of the antecedents is shown in Figure 12, with the label "*DO*-10AntLingTerms" - as can be seen the control response is poor in this case. By making the corresponding experiments, it was also concluded that the process is not controllable for variations of 15%, 20%, 25%, and 30% on the linguistic terms of the antecedents, for variations of 20%, 25%, and 30% on the linguistic terms of the consequent, and for variations in the defuzzification operator.

5. Conclusion

This paper proposed a new method to automatically extract all fuzzy parameters and design the structure of a FLC in order to control nonlinear processes. The learning of the FLC is performed by a HGA, using a set of input/output data, previously extracted from a process under control (e.g. by manual control). The method does not require any prior knowledge concerning the fuzzy rule structure, location or shape of membership functions, implication and aggregation operators, defuzzification methods, or selection of adequate input variables and corresponding time delays.

The main purpose of the HGA is to develop a FLC with a response similar to the one used to compile the dataset, or in less successful attempts, to develop a controller which constitutes a starting point for further adjustments. In order to obtain a better control error if necessary, the proposed algorithm could be easily applied to initialize the required fuzzy knowledge-base of adaptive controllers such as the ones presented in (Mendes et al., 2011). Additionally, the method may also be used to understand a process for which there is little or no information available, since it automatically extracts all fuzzy parameters, and it is able to gather a knowledge-base about the process control.

The control of the dissolved oxygen in an activated sludge reactor within a wastewater treatment plant (WWTP) was studied. The results showed that the proposed method extracted all parameters of the FLC. After the learning, the resulting FLC controlled the process with success.

Acknowledgments

This work was supported by Mais Centro Operational Program, financed by European Regional Development Fund (ERDF), and Agência de Inovação (AdI) under Project SCIAD/2011/21531.

Jérôme Mendes has been supported by Fundação para a Ciência e a Tecnologia (FCT) under Grant SFRH/BD/63383/2009.

References

Alam, M., Tokhi, M., 2008. Hybrid fuzzy logic control with genetic optimisation for a single-link flexible manipulator. Engineering Applications of Artificial Intelligence 21, 858–873.

Ali, S.F., Ramaswamy, A., 2009. Optimal fuzzy logic control for mdof structural systems using evolutionary algorithms. Engineering Applications of Artificial Intelligence 22, 407 – 419.

Andersen, H., Lotfi., A., Tsoi, A., 1997. A new approach to adaptive fuzzy control: The controller output error method. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 27, 686–691.

Belchior, C.A.C., Araújo, R.A.M., Landeck, J.A.C., 2012. Dissolved oxygen control of the activated sludge wastewater treatment process using stable adaptive fuzzy control. Computers & Chemical Engineering 37, 152 – 162.

Coban, R., Can, B., 2010. A trajectory tracking genetic fuzzy logic controller for nuclear research reactors. Energy Conversion and Management 51, 587–593.

Cordón, O., Herrera, F., Hoffmann, F., Magdalena, L., 2001. Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases. volume 19 of *Advances in Fuzzy Systems–Applications and Theory*. World Scientific Publishing Co. Pte. Ltd.. 1 edition.

Delgado, M.R., Nagai, E.Y., de Arruda, L.V.R., 2009. A neuro-coevolutionary genetic fuzzy system to design soft sensors. Soft Computing - A Fusion of Foundations, Methodologies and Applications 13, 481–495.

Delgado, M.R., Zuben, F.V., Gomide, F., 2001. Hierarchical genetic fuzzy systems. Information Sciences 136, 29 – 52.

Espinosa, J., Vandewalle, J., Wertz, V., 2004. Fuzzy Logic, Identification and Predictive Control (Advances in Industrial Control). Springer-Verlag New York, Inc, Secaucus, NJ, USA.

Herrera, F., 2008. Genetic fuzzy systems: Taxonomy, current research trends and prospects. Evolutionary Intelligence 1, 27–46.

Herrera, F., Lozano, M., Verdegay, J.L., 1995. Tuning fuzzy logic controllers by genetic algorithms. International Journal of Approximate Reasoning 12, 299–315.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI, USA.

Homayouni, S.M., Hong, T.S., Ismail, N., 2009. Development of genetic fuzzy logic controllers for complex production systems. Computers & Industrial Engineering 57, 1247–1257.

Jeppsson, U., Pons, M., 2004. The COST benchmark simulation model-current state and future perspective. Control Engineering Practice 12, 299 – 304.

Kasabov, N.K., 1996. Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engineering. MIT Press, Cambridge.

Mendes, J., Araújo, R., Sousa, P., Apóstolo, F., Alves, L., 2011. An architecture for adaptive fuzzy control in industrial environments. Computers in Industry 62, 364–373.

Sharkawy, A.B., 2010. Genetic fuzzy self-tuning pid controllers for antilock braking systems. Engineering Applications of Artificial Intelligence 23, 1041 – 1052.

Sivanandam, S.N., Deepa, S.N., 2007. Introduction to Genetic Algorithms. Springer.

Souza, F., Santos, P., Araújo, R., 2010. Variable and delay selection using neural networks and mutual information for data-driven soft sensors, in: IEEE Conference on Emerging Technologies and Factory Automation (ETFA), 2010, pp. 1–8.

Wang, L.X., 1997. A Course in Fuzzy Systems and Control. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.