Mathware & Soft Computing 8 (2001) 179-201

A Multicriteria Genetic Tuning for Fuzzy Logic Controllers *

R. Alcalá[1†], J. Casillas[2], J.L. Castro[2], A. González[2] and F. Herrera[2]
[1]Dept. of Computer Science
University of Jaén, E-23071 – Jaén, Spain
e-mail: alcala@ujaen.es
[2]Dept. of Computer Science and Artificial Intelligence
University of Granada, E-18071 – Granada, Spain
e-mail: {casillas,castro,gonzalez,herrera}@decsai.ugr.es

Abstract

This paper presents the use of genetic algorithms to develop smartly tuned fuzzy logic controllers in multicriteria complex problems. This tuning approach has some specific restrictions that make it very particular and complex because of the large time requirements existing due to the need of considering multiple criteria —which enlarges the solution search space—, and to the long computation time models usually used for fitness assessment. To solve these restrictions, two efficient genetic tuning strategies considering different multicriteria approaches have been developed and tested in a real-world problem for fuzzy control of HVAC Systems.

Keywords: Genetic Tuning, Multiple Criteria, Multiple Objectives, Fuzzy Logic Controllers.

1 Introduction

Fuzzy Logic Controllers (FLCs) [8, 22] are very robust tools which would enable the implementation of control strategies incorporating expert knowledge. These strategies typically define a nonlinear mapping from the system's state space to the control space. Thus, it is possible to consider the output of an FLC as a nonlinear control surface reflecting the process of the operator's prior knowledge.

Tuning approaches are usually based on the availability of a predefined Rule Base (RB) and a preliminary set of membership functions associated to the fuzzy partitions, Data Base (DB). Their main aim is to find a better set of parameters

*This research has been supported by the European Commission under the Genesys Project JOE-CT98-0090.
†Corresponding author.

by only changing the DB components, thus making optimum the FLC behavior. In this way, FLCs could be obtained from human experience or learning methods to subsequently be tuned by the application of automatic tuning techniques.

However, many real-world problems involve two types of difficulties:

- The evaluation is based on multiple criteria. This fact adds complexity to the search because we must obtain the best trade-off among the different criteria.
- The controller accuracy is assessed by means of simulations which usually take a long time. This causes the run time of the algorithms to be extremely long.

In this case, numerous factors have to be considered in order to address these restrictions. It makes the system being controlled very complex and presents a strong non linearity. Therefore, an efficient operation of the automatic tuning techniques is a necessary condition in order to achieve good results.

Genetic Algorithms (GAs) are global search techniques that can represent any type of fuzzy rules, present flexibility to work with different FLC architectures and have a good capability to include expert knowledge [5]. Furthermore, the ability to handle complex problems, involving features such as discontinuities, multimodality, disjoint feasible spaces and noisy function evaluations, reinforces the potential effectiveness of GAs in multicriteria search and optimization. For these reasons, GAs have been recognized to be possibly well-suited to multicriteria optimization [3, 6, 11, 28, 31].

Although there are many genetic tuning approaches [2, 4, 14, 16, 20], neither of them can be used satisfactorily because they do not properly address the said restrictions. Therefore, in order to solve these two problems, efficient genetic tuning approaches considering both restrictions should be developed.

In this work, the use of GAs to develop smartly tuned FLCs in complex multicriteria problems is presented. Two efficient genetic tuning strategies, considering different multicriteria approaches, have been proposed.

To do so, this contribution is arranged in the following way. Section 2 proposes the use of multicriteria techniques together with some approaches that increase the convergence speed of GAs for solving the said restrictions. Section 3 briefly introduces the principles of the GAs and the different multicriteria genetic optimization approaches. Section 4, combines the multicriteria optimization with the efficient genetic tuning approaches proposing and presenting two particular genetic tuning techniques. Finally, experimental results are shown in Section 5, whilst some concluding remarks are pointed out in Section 6.

2 Tuning Restrictions

As we have indicated, the two important restrictions we want to solve are the need of considering multiple criteria (which enlarges the solution search space) and the long computation time models require to assess the accuracy of each individual.

The **first restriction** will be solved by using multicriteria optimization techniques that allow us to work with fitness functions comprised by competitive objectives. In these cases, we could obtain not only an optimal solution, but a possible solution set. Depending on the number of solutions obtained, we can distinguish between two multicriteria approaches:

- Multicriteria aggregation-based methods: All classical methods scalarize the objective vector reducing it to a scalar optimization problem. Probably, the simplest of all these classical techniques is the method of objective weighting. In this case, multiple criteria functions are combined into one overall objective function by means of a vector of weights. This technique has much sensitivity and dependency toward weights. However, when trustworthy weights are available, this approach reduce the search space providing the adequate direction into the solution space and its use is highly recommended. Therefore, the main question to consider this approach is: Have we trusted weights to estimate the importance of each objective?

- Multicriteria non aggregation-based methods or multiobjective methods: In a typical multicriteria optimization problem, there is a set of solutions that are superior to the rest in the search space when all the objectives are considered. These solutions are known as non-dominated solutions (pareto set), whilst the remaining solutions are known as dominated solutions. None of the solutions in the non-dominated set is absolutely better than the other ones.

  Mathematically, the concept of *Pareto-optimality* or *non-dominance* is defined as follows. Let us consider, without loss of generality, a multicriteria function $f(x) = (f_1(x), f_2(x), \ldots, f_n(x))$ to be minimized with m parameters, $x = (x_1, x_2, \ldots, x_m) \in X$, and n objectives. A decision vector $a \in X$ dominates $b \in X$ (noted as $a \succ b$) if, and only if:

$$\forall i \in \{1, 2, \ldots, n\},\ f_i(a) \leq f_i(b) \quad \wedge$$

$$\exists j \in \{1, 2, \ldots, n\} \ \mid \ f_j(a) < f_j(b) \ .$$

  Any vector that is not dominated by any other is said to be pareto-optimal or non-dominated.

In order to solve the **second restriction**, the use of efficient tuning methods is necessary. There are some approaches that increase the convergence speed of GAs:

- An objective weighting technique would reduce the search space providing the adequate direction into the pareto when trustworthy weights are used.

- A steady-state GA [29], that consists of selecting two of the best individuals in the population and combining them to obtain new offspring. This approach improves the convergence and simultaneously decreases the number of evaluations.

- In order to reduce the GAs search space, an integer coding could be used. This one uses discrete parameter domains forcing to take values from a finite value set [13] (see Figure 1). The cardinality of this set must be rich enough in order to allow the tuning process to achieve accurate results, but small enough so that small changes provoke significant variations.

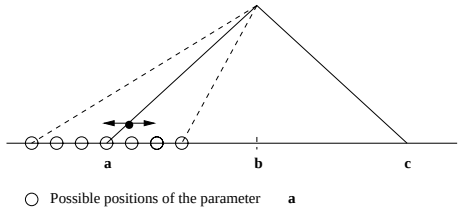


Figure 1: Integer coding to tune fuzzy sets

- Reducing the population size, the number of evaluations is significantly decreased. However, this size must be large enough in order to maintain the diversity in the genetic population.

Several efficient tuning methods with different characteristics will be proposed based on the combination of multicriteria techniques with these different approaches.

3 GAs and Multicriteria Optimization

3.1 Genetic Algorithms

GAs are general-purpose global search algorithms that use principles inspired by natural population genetics to evolve solutions to problems. The basic principles of the GAs were first laid down rigorously by Holland [19] and are well described in many texts such as [23].

The basic idea is to maintain a population of knowledge structures that evolves over time through a process of competition and controlled variation. Each structure in the population represents a candidate solution to the specific problem and has an associated *fitness* to determine which structures are used to form new ones in the process of competition.

Hence, a subset of relatively good solutions are selected for reproduction to give offspring that replace the relatively bad solutions which die. In the traditional **generational approach**, offspring replace their parents for the next generation [23]. These new individuals are created by using genetic operators such as crossover and mutation. The crossover operator combines the information contained into the parents increasing the average quality of the population (explotation), while the mutation operator randomly changes the new individuals helping the algorithm to avoid local optima (exploration).

The **steady-state approach** [29] consists of selecting two of the best individuals in the population and combining them to obtain one or two offspring. Every new individual is included in the population replacing the worst individual (usually) if the former is better adapted than the latter.

An advantage of the steady-state approach with respect to the generational one is that good solutions are used as soon as they are available. Therefore, the convergence is accelerated while the number of evaluations is decreased. However, a premature convergence could provoke the GA to get stuck in a local optima. Thus, while the steady-state approach is quicker obtaining relatively good solutions, the generational one is theoretically more robust and sure, and in most cases preferable when the search speed is not important.

3.2 Multicriteria Genetic Optimization

Generally, multicriteria GAs only differ from the rest of GAs in the fitness function and/or in the selection operator. The evolutionary approaches in multicriteria optimization can be classified in three groups [11]: plain aggregating approaches, population-based non-pareto approaches, and pareto-based approaches.

Plain Aggregating Approaches

As conventional GAs require scalar fitness information to work on, a scalarisation of the objective vectors is always necessary. In most problems, where no global criterion directly emerges from the problem formulation, objectives are often artificially combined, or aggregated, into a scalar function according to some understanding of the problem, and then the GA is applied. Practically, all the classical aggregation approaches can be used with GAs. Some approach of this kind has been reported in the literature: weighted sum [26], distance functions [30], etc.

Optimizing a combination of the objectives has the advantage of producing a single compromise solution, requiring no further interaction with the decision-maker. The problem is that, if the optimal solution can not be accepted, new runs of the optimizer may be required until a suitable solution is found. However, in the case of objective weighting, when trustworthy weights are available this problem disappears.

Population-Based Non-Pareto Approaches

This approach allows to exploit the special characteristic of GAs. The use of an population of individuals offers the possibility to treat non-commensurable objectives separately and to search for multiple non-dominated solutions concurrently in a single GA run.

Now, a non-dominated set of individuals is obtained instead of obtaining only one of them. In order to do this, the selection operator is changed. Generally, the best individuals according to each objective are selected (in many occasions, an order according to its importance is followed) and then these partial results are combined to obtain the new population.

Different proposals based on this idea can be found in [12, 15, 21, 24].

Pareto-Based Approaches

The population-based non-pareto approaches attempt to promote the generation of multiple non-dominated solutions. However, none makes direct use of the actual definition of Pareto-optimality. As the best solution cannot be selected among the non-dominated solutions set, the approaches should assign equal probability of reproduction to all of them.

In GAs based on the concept of pareto-optimality, to calculate the probability of reproduction of each individual, the solutions are compared by means of the dominance relation ($\succ$). Some approach of this kind can be found in [10, 25].

Although the Pareto-based ranking correctly assigns all non-dominated individuals the same fitness, it does not guarantee that the Pareto set be uniformly sampled. When it is presented with multiple equivalent optima, finite populations tend to converge to only one of them, due to stochastic errors in the selection process. This phenomenon is known as *genetic drift* [7].

Since preservation of diversity is crucial in the field of multiobjective optimisation, several multiobjective GAs have incorporated the *niche* and *specie* concepts for the purpose of favouring such behaviour [10, 25].

4 Proposal: Two Multicriteria Genetic Tuning Strategies

Thinking on these three different multicriteria approaches, we must consider an important aspect in the selection of the best techniques and methods to accomplish the tuning process: Have we trusted weights to estimate the importance of each objective? Depending on whether we can obtain them or not, the recommended strategy will be different. Nevertheless, we will be able to use several tuning processes with and without weights if we want to compare the results or if we are not sure of the weight reliability. In this way, combining the multicriteria and the said efficient tuning approaches, two strategies have been developed (Table 1 shows a summary of the two developed methods):

- Taking into account the existence of trusted weights and in order to benefit from them, we propose a simple steady-state GA with the classical real coding [18] and with a fitness function based on objective weighting that considers them. It will be called, Weighted Multi-Criterion Steady-State Genetic Algorithm (WMC-SSGA). Furthermore, the use of fuzzy goals for dynamically adapting the search direction in the space of solutions will be considered. It will make the method robust and more independent from the weight selection for the fitness function.

- If we do not have trustworthy weights, we must search for advanced multiobjective techniques with adequate characteristics to obtain the desired convergence, e.g., the use of the integer coding. In this way, a multiobjective

approach based in the well-known algorithm presented in [10] is proposed. The so called Multi-Objective Steady-State Genetic Algorithm (MO-SSGA) presents likely premature convergence getting a quick search speed at the expense of decreasing the diversity. This algorithm is complemented with the use of a steady-state approach.

Table 1: Summary on Tuning Strategies

Method	**When**	**Multicriteria approach**		**Characteristics**
One	Trusted weights exist	Aggregation function	fitness	SSGA + real coding
Two	Trusted weights not exist	Pareto-based function	fitness	SSGA + integer coding

There is an important aspect that the proposed methods address in the same way, the definition of the variation intervals for each gene. In the following subsection, this common characteristic is introduced. After this, the two proposed methods will be widely explained.

4.1 Dynamic Variation intervals

In order to be meaningful, each chromosome (a complete DB) must maintain their genes (the DB definition points) within their respective variation intervals. These intervals are usually computed and fixed from the initial solution —DB provided by experts—. However, in our case, these intervals are dynamically adapted from the best individual for each GA iteration, avoiding the restrictions of fixing them from the beginning of the GA run. Thus, once these intervals have been calculated, the genes out of range are randomly generated within them.

Let (a_j^i, b_j^i, c_j^i) be the definition points of the j-th membership function label of the i-th variable. In a strong fuzzy partition (those in which the membership degree within the variable domain is kept to 1.0) the vertex of each label (b_j^i) coincides with the nearest extreme points of its neighbor labels, $c_{j-1}^i = b_j^i = a_{j+1}^i$. In this case, only the vertex of the labels has to be considered and the same variation interval can be defined for coincident points. Thus, the variation intervals are usually defined by the middle points between the correspondent vertex and the vertex of the previous and the next label.

In our case, a more flexible approach is considered and the vertex of the labels does not have to coincide with the nearest extreme points of its neighbor labels (see Figure 2). However, considering these three points as a simple set for each label $B_j = \{c_{j-1}^i, b_j^i, a_{j+1}^i\}$ and taking into account that they have the same variation interval, the same approach can be followed. In this way, the middle point between two sets can be computed considering the maximum point of the first set and the minimum point of the second set. Therefore, to calculate the left extreme of the variation interval for a concrete definition point $x \in B_j$, we should consider the

maximum point of B_{j-1} (l_x^1) and the minimum point of the corresponding set B_j (l_x^2). And for the corresponding right extreme, we should consider the maximum point of B_j (r_x^1) and the minimum point of B_{j+1} (r_x^2).

Finally, taking into account that $a_j^i \in B_{j-1}$, $b_j^i \in B_j$ and $c_j^i \in B_{j+1}$, the variation intervals of each definition point of the j-th label membership function of the i-th variable, (a_j^i, b_j^i, c_j^i), are calculated from the initial or best individual as,

$$\{l^1_{a_j^i}, l^2_{a_j^i}\} = \{\max(c_{j-3}^i, b_{j-2}^i, a_{j-1}^i), \min(c_{j-2}^i, b_{j-1}^i, a_j^i)\}$$
$$\{r^1_{a_j^i}, r^2_{a_j^i}\} = \{\max(c_{j-2}^i, b_{j-1}^i, a_j^i), \min(c_{j-1}^i, b_j^i, a_{j+1}^i)\}$$
$$[L_{a_j^i}, R_{a_j^i}] = [l^2_{a_j^i} - \frac{l^2_{a_j^i} - l^1_{a_j^i}}{2}, r^1_{a_j^i} + \frac{r^2_{a_j^i} - r^1_{a_j^i}}{2}] \ ,$$

$$\{l^1_{b_j^i}, l^2_{b_j^i}\} = \{\max(c_{j-2}^i, b_{j-1}^i, a_j^i), \min(c_{j-1}^i, b_j^i, a_{j+1}^i)\}$$
$$\{r^1_{b_j^i}, r^2_{b_j^i}\} = \{\max(c_{j-1}^i, b_j^i, a_{j+1}^i), \min(c_j^i, b_{j+1}^i, a_{j+2}^i)\}$$
$$[L_{b_j^i}, R_{b_j^i}] = [l^2_{b_j^i} - \frac{l^2_{b_j^i} - l^1_{b_j^i}}{2}, r^1_{b_j^i} + \frac{r^2_{b_j^i} - r^1_{b_j^i}}{2}] \ ,$$

$$\{l^1_{c_j^i}, l^2_{c_j^i}\} = \{\max(c_{j-1}^i, b_j^i, a_{j+1}^i), \min(c_j^i, b_{j+1}^i, a_{j+2}^i)\}$$
$$\{r^1_{c_j^i}, r^2_{c_j^i}\} = \{\max(c_j^i, b_{j+1}^i, a_{j+2}^i), \min(c_{j+1}^i, b_{j+2}^i, a_{j+3}^i)\}$$
$$[L_{c_j^i}, R_{c_j^i}] = [l^2_{c_j^i} - \frac{l^2_{c_j^i} - l^1_{c_j^i}}{2}, r^1_{c_j^i} + \frac{r^2_{c_j^i} - r^1_{c_j^i}}{2}] \ ,$$

Notice that the associated variation intervals of the corresponding extreme values, a_j^i and c_j^i, are calculated exactly as the intervals for b_{j-1}^i and b_{j+1}^i, respectively.

Figure 2 graphically depicts the variation intervals for the i-th variable following the proposed approach. We have considered that the vertex of the labels at the edges of the variables' domain must coincide with the extreme points. These labels will be symmetrical with respect to their vertexes.

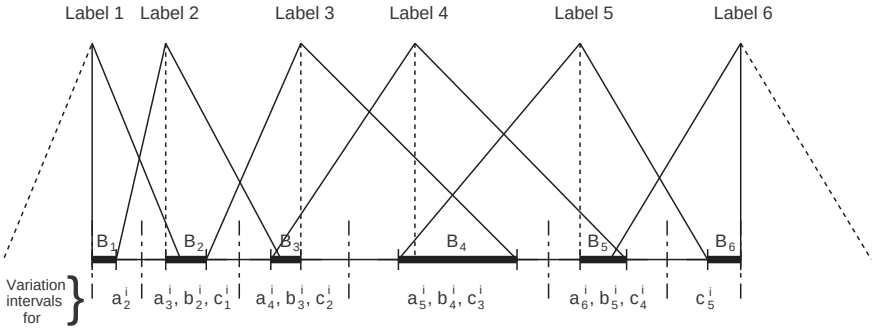


Figure 2: Variation intervals of the i-th variable.

4.2 Weighted Multi-Criterion Steady-State Genetic Algorithm

WMC-SSGA consists of a GA based on the well-known steady-state approach [29]. Its fitness function is based on objective weighting. However, in order to make the method robust and more independent from the weight selection for the fitness function, the use of fuzzy goals for dynamically adapting the search direction in the space of solutions will be considered.

Coding scheme: WMC-SSGA uses a real coding scheme [18]. A solution is directly encoded in a chromosome by joining the definition points (a_j^i, b_j^i, c_j^i) of the l_i labels of each one of the m variables composing the DB. For example:

$$C_i = (a_1^i, b_1^i, c_1^i, \ldots, a_{l_i}^i, b_{l_i}^i, c_{l_i}^i), \quad i = 1, \ldots, m \ ,$$
$$C = C_1 C_2 \ldots C_m \ .$$

Initial gene pool: To make use of the existing knowledge, the DB previously obtained from expert knowledge is included in the population as an initial solution. The remaining individuals are randomly generated maintaining their genes within their respective variation intervals. These intervals are computed from the initial solution (see Section 4.1).

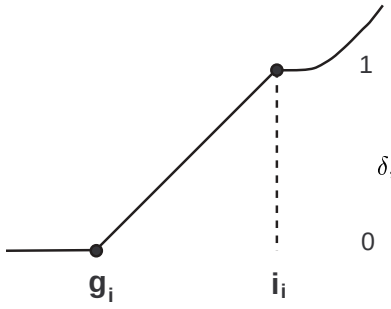
Evaluating the chromosome: The fitness function is based on objective weighting. However, it has been modified in order to consider the use of fuzzy goals for dynamically adapting the search direction in the space of solutions, decreasing the improvement possibility of those objectives which approach their goals in the first place. Thus, a function modifier parameter, $\delta_i(x)$, is used to penalize each objective (taking values over 1.0) whenever its value gets worse with respect to the initial solution or to decrement the importance of each individual fitness value whenever it comes to its respective goal (taking values close to 0.0). Moreover, a penalization rate has been included in $\delta_i(x)$, allowing the user to set up priorities in the objectives. This penalization rate, p_i, for each objective is a real number from 0.7 to practically 1, although the user specifies this penalization from 0 to 1 (less and more priority, respectively), which is more interpretable. Therefore, the global fitness is evaluated as:

$$F = \sum_{i=1}^{n} w_i \cdot \delta_i(C_i) \cdot C_i \ ,$$

with C_i being the considered criteria (objectives) for each specific problem and w_i being the corresponding weighting coefficients.

Two cases can be presented in the corresponding individual according to the value of the goal, g_i, and the value of the initial solution, i_i. Depending on these values, two different δ functions will be applied.

- The first case is when the value of g_i is lesser than the value of i_i, presenting the following behavior (see Figure 3),

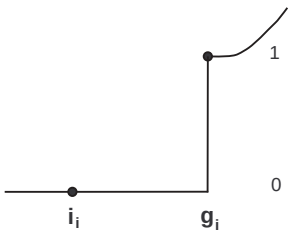


$$\delta_i(x) = \begin{cases} 0, & \text{if } x \leq g_i \\ \dfrac{x - g_i}{i_i - g_i}, & \text{if } g_i < x < i_i \\ \dfrac{x - i_i}{x - x \cdot p_i} + 1, & \text{if } i_i \leq x \end{cases} .$$

Figure 3: $\delta_i(x)$ when $g_i \leq i_i$

In this case, the objective is not considered if the goal is met and penalized if the initial results are worsen.

- The second case happens when the initial value, i_i, is lesser than the goal value, g_i (see Figure 4),



$$\delta_i(x) = \begin{cases} 0, & \text{if } x < g_i \\ \dfrac{x - g_i}{x - x \cdot p_i} + 1, & \text{if } g_i \leq x \end{cases} .$$

Figure 4: $\delta_i(x)$ when $g_i > i_i$

Now, the initial results can be worsen while the goal is met, and it is penalized otherwise.

Notice that the penalization function allows the search to slightly worsen the goal, improving other objectives to subsequently met the goal again.

Genetic operators: The **selection** is based on the Baker's stochastic universal sampling [1] (by only selecting two individuals). WMC-SSGA also follows the interval adapting scheme explained in Section 4.1.

Since WMC-SSGA uses the real coding scheme, the crossover and mutation operators have been selected according to this aspect: the Max-Min-Arithmetical crossover [17] and Michalewicz's non-uniform mutation [23].

Using the max-min-arithmetical **crossover**, if $C_v^t = (c_1, \ldots, c_k, \ldots, c_H)$ and $C_w^t = (c'_1, \ldots, c'_k, \ldots, c'_H)$ are going to be crossed, the next four offspring are obtained:

$$C_1^{t+1} = aC_w^t + (1-a)C_v^t$$
$$C_2^{t+1} = aC_v^t + (1-a)C_w^t$$
$$C_3^{t+1} \text{ with } c_{3k}^{t+1} = \min\{c_k, c'_k\}$$
$$C_4^{t+1} \text{ with } c_{4k}^{t+1} = \max\{c_k, c'_k\} \ ,$$

with a being a constant parameter chosen by experts, and H being the number of genes.

In the case of the Michalewicz's non-uniform **mutation**, a gene c_k, with a variation interval $[L_{c_k}, R_{c_k}]$, can be mutated as $c'_k = c_k + \Delta(t, R_{c_k} - c_k)$ with probability 0.5, or as $c'_k = c_k - \Delta(t, c_k - L_{c_k})$, in other case. With t being the current generation, function $\delta(t, y)$ returns a value in the range $[0, y]$ such that the probability of $\delta(t, y)$ being close to 0 increases as the number of generations increases. This function is formulated as $\delta(t, y) = y(1 - r^{(1-\frac{t}{T})^b})$, with r being a random number in $[0, 1]$, T the total number of generations, and b being selected by the user to determine the dependency with t.

Thus, once the mutation operator is applied over the four offspring obtained from the crossover operator, the resulting descendents are the two best of these four individuals.

Restart approach: Finally, to get away from local optima, this algorithm uses a restart approach [9]. Thus, when the population of solutions converges to very similar results (practically the same fitness value in the population), the entire population but the best individual is randomly generated within the variation intervals. It allows the algorithm to perform a better exploration in the search space and to avoid getting stuck at local optima.

4.3 Multi-Objective Steady-State Genetic Algorithm

MO-SSGA consist on an integer coded steady-state GA for multiobjective optimization. This algorithm presents likely premature convergence getting a quick search speed at the expense of decreasing the diversity. Its fitness function is based on the multiobjective approach presented in [10]. However a new scheme to accommodate goal attainment have been proposed.

Coding scheme: Once again, a solution is encoded in a chromosome by joining the representation (definition points) of the l_i labels of each one of the m variables composing the DB. However, in this case, an integer coding scheme is used to represent the possible solutions. This one uses discrete parameter domains for each one of the m variables, forcing to take values from a finite value set. The cardinality of these domains, G_i, is determined by the global granularity parameter G —chosen by experts— and directly depends on the corresponding number of labels l_i:

$$G_i = G * l_i + 1, \ \ i = 1, \ldots, m.$$

Let L^i and R^i be the left and right extremes of the i-th variable domain. The corresponding possible values for this variable are uniformly distributed from L^i to R^i. Therefore, the distance between a point and the next one will be,

$$d_i = \frac{R^i - L^i}{G_i - 1}.$$

In this way, a mapping between an integer ordered set (from 0 to $G_i - 1$) and the set B_i of the corresponding real discrete values could be established as follows:

$$D : \{0, 1, \ldots, G_i - 1\} \longrightarrow B_i,$$
$$D(x^i) = L^i + x^i * d_i, \ \forall \ x^i \in \{0, 1, \ldots, G_i - 1\}.$$

The integer coding scheme consists on coding the possible solutions by using these integer ordered sets to represent the corresponding real discrete values —which can be easily obtained by using $D(x)$—. This representation eases the application of the genetic operators.

Initial gene pool: The initial pool is comprised of one individual containing the DB previously obtained from expert knowledge and the remaining ones generated at random maintaining their genes within their respective variation intervals. The intervals in which will switch around each gene are established from the initial solution as explained in Section 4.1. However, since this method is based on integer coding, once the variation intervals are computed as in the real coding, the upper and lower limits defining these intervals and the initial solution values must be encoded by rounding to the closer corresponding integer in the coding scheme as follows:

$$C : [L^i, R^i] \longrightarrow \{0, 1, \ldots, G_i - 1\},$$
$$C(x^i) = round\left(\frac{\frac{x^i - L^i}{R^i - L^i}}{\frac{1}{G_i - 1}}\right), \ \forall \ x^i \in [L^i, R^i].$$

After this, the initial gene pool can be generated within these variation intervals following the integer coding.

Evaluating the chromosome: A rank-based fitness assignment method for Multiple Objective GA (MOGA) was developed by Fonseca and Fleming in [10]. With this purpose, a rank is assigned to each individual of the population. These individuals are sorted to be selected according to rank. MO-SSGA is based on this approach. Thus, the position in the individuals' ranking can be given by:

If *non-dominated* **then** $rank(x_i) = 1$
Else $rank(x_i) = 1 + (dominants\ of\ x_i)$

The traditional assignment of fitness according to rank may be extended as follows:

1. Sort population according to rank.

2. Assign fitness to individuals interpolating from the best (rank 1) to the worst (rank $n^* \leq N$) in the usual way, according to some function, usually linear but not necessarily.

3. Average the fitness of individuals with the same rank, so that all of them will be sampled at the same rate. Note that this procedure keeps the global population fitness constant while maintaining appropriate selective pressure, as defined by the function used.

A scheme to accommodate goal attainment was proposed in [10]. It consists on modifying the ranking scheme for selection. In this way, the ranking in MO-SSGA was extended to accommodate goal information by altering the way in which individuals are compared with one another. However, considering the original approach interesting individuals were lost. Therefore, a new scheme has been implemented replacing the *concept of dominance* by the *concept of preferable*. Following this new scheme, individuals that meet more goals are more important than individuals that meet less even if they do not dominate to these latter ones. Thus, $y_a = (y_{a,1}, \ldots, y_{a,q})$ is preferable to $y_b = (y_{b,1}, \ldots, y_{b,q})$:

- If y_a meets more number of goals than y_b, or,

- If y_a meets exactly the same goals than y_b, $k+1, \ldots, q$, and y_a dominates to y_b in $k+1, \ldots, q$ objectives.

Genetic operators: The chromosome **selection** is based in the Baker's stochastic universal sampling [1]. However, in this case only two individuals are selected. These two individuals replace the to worst in the population.

The genetic operators are a two points crossover and a mix mutation. Using the two points **crossover**, two points are randomly generated and the genes between these points are exchanged to obtain two offsprings. After this, the mix **mutation** is applied over these two offspring. It consists on randomly using one of the following mutation operators:

- A random mutation, by randomly selecting an integer value within the variation interval of the corresponding gene.

- The Thrift mutation [27], by changing the gene one level either up or down.

In the same way that WMC-SSGA, this method follows the same interval adapting scheme explained in Section 4.1. However, since the integer coding is considered, the best solution in each iteration must be decoded —by using $D(x)$— to compute the variation intervals as in the real coding scheme. After this, the upper and lower limits defining these intervals must be encoded by using $C(x)$.

5 Experiments and Results for Fuzzy Control of HVAC Systems

5.1 HVAC Systems

In EU countries, primary energy consumption in buildings represents about 40% of total energy consumption, and depending on the countries, more than a half of this energy is used for indoor climate conditions. On a technological point of view, it is estimated that the consideration of specific technologies like Building Energy Management Systems (BEMSs) can save up to 20% of the energy consumption of the building sector, i.e., 8% of the overall Community consumption. BEMSs are generally applied only to the control of active systems, i.e., Heating, Ventilating, and Air Conditioning (HVAC) systems.

HVAC Systems are equipments usually implemented for maintaining satisfactory comfort conditions in buildings. The energy consumption as well as indoor comfort aspects of ventilated and air conditioned buildings are highly dependent on the design, performance and control of their HVAC systems and equipments. Therefore, the use of appropriate automatic control strategies, as FLCs, for HVAC systems control could result in important energy savings when compared to manual control.

However, due to the complexity of the problem —many criteria have to be considered—, a rational operation and improved performance of FLCs is required. The use of smart setting and tuning techniques for these controllers could improve the energy savings and the indoor comfort by fitting the DB parameters of previously obtained KBs provided by experts.

In this way, in order to evaluate the goodness of the proposed tuning techniques, several experiments have been carried out within the framework of the JOULE-THERMIE programme under the GENESYS [1] project. A real test site (building) provided by a French private enterprise —whose name must remain anonymous— was available for the experiments. From now on, this site will be called ATC test site —from Anonymous Test Cell—. The main objective was the energy performance but maintaining the required indoor comfort levels.

5.2 ATC test site

Located in France, this test environment consists of two adjacent twin cells. Around these test cells walls, an artificial climate can be created at any time (winter conditions can be simulated in summer and vice-versa). These test cells are medium weight constructions. The HVAC system tested is a fan coil unit supplied by a reverse-cycle heat pump, and a variable fan speed mechanical extract for ventilation.

To assess the proposed tuning techniques for fitness computation, accurate models of this controlled building (as well as the corresponding initial FLCs) were pro-

[1] GENESYS Project: Fuzzy controllers and smart tuning techniques for energy efficiency and overall performance of HVAC systems in buildings, European Commission, Directorate-General XII for Energy (contract JOE-CT98-0090).

vided by experts for each season, considering fall and spring as the same kind of season. These models require long computation times, which makes more complex the tuning process. The results obtained were very satisfactory, specially for the ATC summer-season model. However, due to the large number of results, we will work only with a cross-section of the models, the ATC summer-season model.

A hierarchical FLC architecture considering the Predicted Mean Vote [2] (PMV), CO_2 concentration, previous HVAC system status and outdoor temperature was proposed for this site. The ATC summer-season FLC architecture and variables are presented in Figure 5. Each module in the figure represents a simple fuzzy controller. With respect to the fuzzy reasoning method used for each simple FLC of this global architecture, we have selected the singleton fuzzification, the *minimum t-norm* playing the role of the implication and conjunctive operators, and the *mean of maxima weighted by the matching degree* as deffuzification strategy.

The initial KB was obtained from BEMS designers for this model. Figures 5 and 6 show the initial RB and DB of the ATC FLC for summer-season. This initial RB is fixed for all the tuning process. As initial DB, we considered symmetrical fuzzy partitions of triangular-shaped membership functions for each one of the m variables. These membership functions were labeled from $L1$ to Ll_i, with l_i being the number of membership functions of the i-th variable. The decision table (initial RB) for each simple FLC (module) is represented in Figure 5 in terms of this labels. Therefore, each cell of the table represents a fuzzy subspace and contains its associated output consequent(s), i.e., the correspondent label(s).

In this case, the objective was to **minimize** the following five criteria:

C_1 Upper thermal comfort limit: $if\ PMV > 0.5, C_1 = C_1 + (PMV - 0.5)$.

C_2 Lower thermal comfort limit: $if\ PMV < -0.5, C_2 = C_2 + (-PMV - 0.5)$.

C_3 Indoor air quality requirement: $if\ CO_2\ conc. > 800ppm, C_3 = C_3 + (CO_2 - 800)$.

C_4 Energy consumption: $C_4 = C_4+$ Power at time t.

C_5 System stability: $C_5 = C_5+$ System change from time t to $(t-1)$.

Therefore, the fitness function for WMC-SSGA was comprised of five criteria. The main problem was then to assign appropriate weights to each criterion. The basic idea in this weight definition was to find financial equivalents for all of them. Such equivalences are difficult to define and there is a lack of confident data on this topic. Whereas, energy consumption cost is easy to set, comfort criteria are more difficult. Recent studies have shown that an 18% improvement in people's satisfaction about indoor climate corresponds to a 3% productivity improvement for office workers. Based on typical salaries and due to the fact that PMV and CO_2 concentrations are related to people's satisfaction, such equivalences can be defined.

[2] The PMV global thermal comfort index selected by international standard ISO 7730 (incorporating relative humidity and mean radiant temperature).

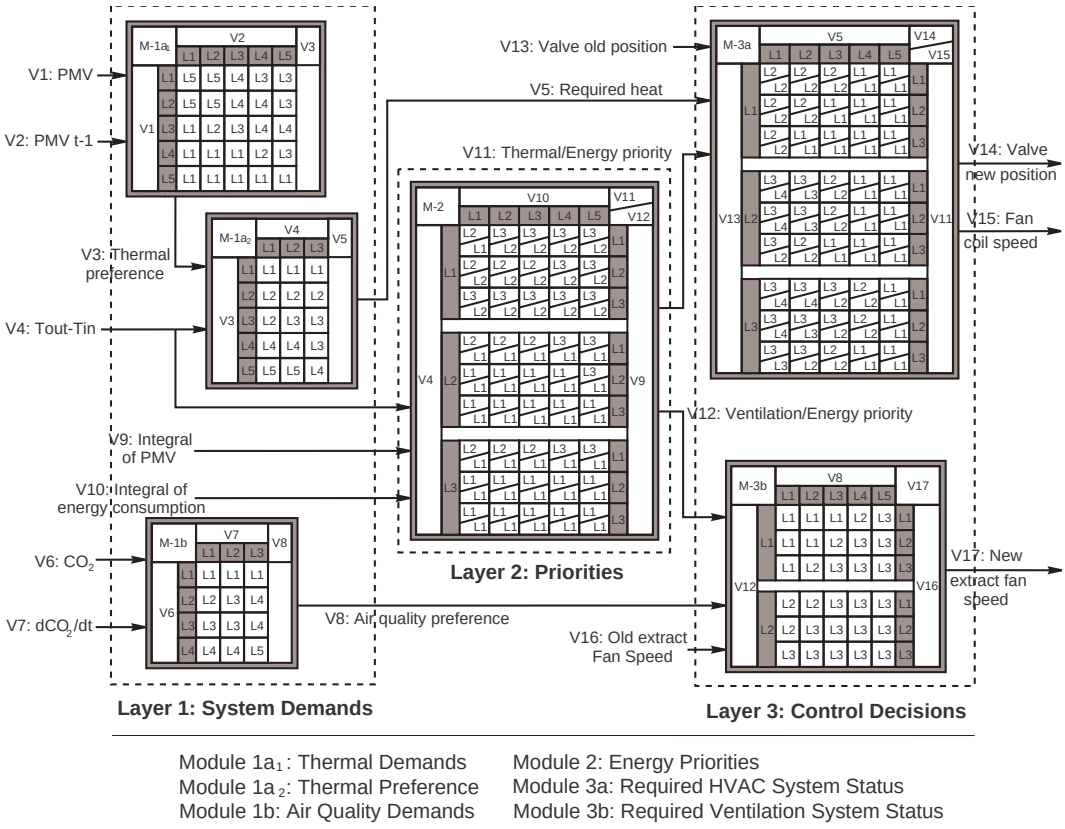


Figure 5: Initial rule base and generic structure of the ATC summer-season fuzzy logic controller

The same strategy can be applied to the system stability criterion, life-cycle of various systems being related to number of operations. Based on this, trusted weights for this test site were obtained. The chosen values were: $w_1 = 0.0041511$, $w_2 = 0.0041511$, $w_3 = 0.00000228333$, $w_4 = 0.0000017832$ and $w_5 = 0.000761667$.

The tuning strategies were assessed with simulations of 10 days with the corresponding climatic conditions. The results obtained by the tuning methods for the ATC summer-season model are presented in the following subsection. These results will be compared to the performance of the initial expert FLC and to a classic control technique, an On-Off controller.

The intention from experts was to try to have 15% energy saving (C_4) together with a global improvement of the system behavior compared to On-Off control. Comfort parameters could be slightly increased if necessary (no more than 1.0 for criteria C_1 and C_2).

5.3 Experiments

The results obtained by the proposed methods are shown in Table 2 together with the goals values (g_i) imposed to them, where % stands for the improvement rate with respect to the On-Off controller for each criterion. The tuned DBs presented in this table for the Summer ATC model correspond to three individuals from the population at iteration 500 with WMC-SSGA. The remaining results were obtained at iteration 500 with MO-SSGA.

Since the time required for each model run was 215 seconds approximately, the estimated run times were, four days for 500 iterations in WMC-SSGA and two days for 500 iterations in MO-SSGA (computed as product of the number of evaluations per generation, the evaluation time and the number of generations).

Following the experts intention, the goal values imposed to WMC-SSGA and MO-SSGA were the following ones:

- WMC-SSGA: $g_1 = 1$, $g_2 = 1$, $g_3 = 7$, $g_4 = 2000000$ and $g_5 = 1000$, with penalization rates of $p_1 = 1$, $p_2 = 1$, $p_3 = 1$, $p_4 = 0.9$, and $p_5 = 0.97$, respectively.
- MO-SSGA: $g_1 = 1$, $g_2 = 1$, $g_3 = 8$, $g_4 = 2700000$ and $g_5 = 1100$, with granularity $G = 40$.

Notice that these goals imposed to the algorithms are higher than the ones initially required since the initial goals were easily met.

Table 2: Results obtained with the ATC Summer-Season model

ATC	PMV>0.5		PMV<-0.5		CO_2		Energy		Stability	
MODEL	C_1	%	C_2	%	C_3	%	C_4	%	C_5	%
ON-OFF	0.0	–	0	–	0	–	3206400	–	1136	–
INIT.	0.0	–	0	–	0	–	2901686	9.50	1505	-32.48
WMC (g_i)	1.0	–	1	–	7	–	2000000	–	1000	–
MO (g_i)	1.0	–	1	–	8	–	2700000	–	1100	–
WMC-1	0.0	–	0	–	0	–	2575949	19.66	1115	1.85
WMC-2	0.0	–	0	–	0	–	2587326	19.31	1077	5.19
WMC-3	0.0	–	0	–	0	–	2596875	19.01	1051	7.48
MO-1	0.1	-10	0	–	0	–	2697449	15.87	1543	-35.83
MO-2	0.5	-51	0	–	0	–	2836160	11.55	1053	7.31

In this case, the expert goal has been easily met by WMC-SSGA. Moreover, the solutions present a desirable diversity that allow us to select different and interesting FLCs.

From the results in Table 2, experts selected the third DB from WMC-SSGA as the most promising one. In this case, the solutions obtained from this method

present improvement ratios of about 20% in energy and 5% in stability. The results obtained with MO-SSGA were clearly worse than the ones obtained with WMC-SSGA. However, in this case acceptable solutions could be quickly obtained.

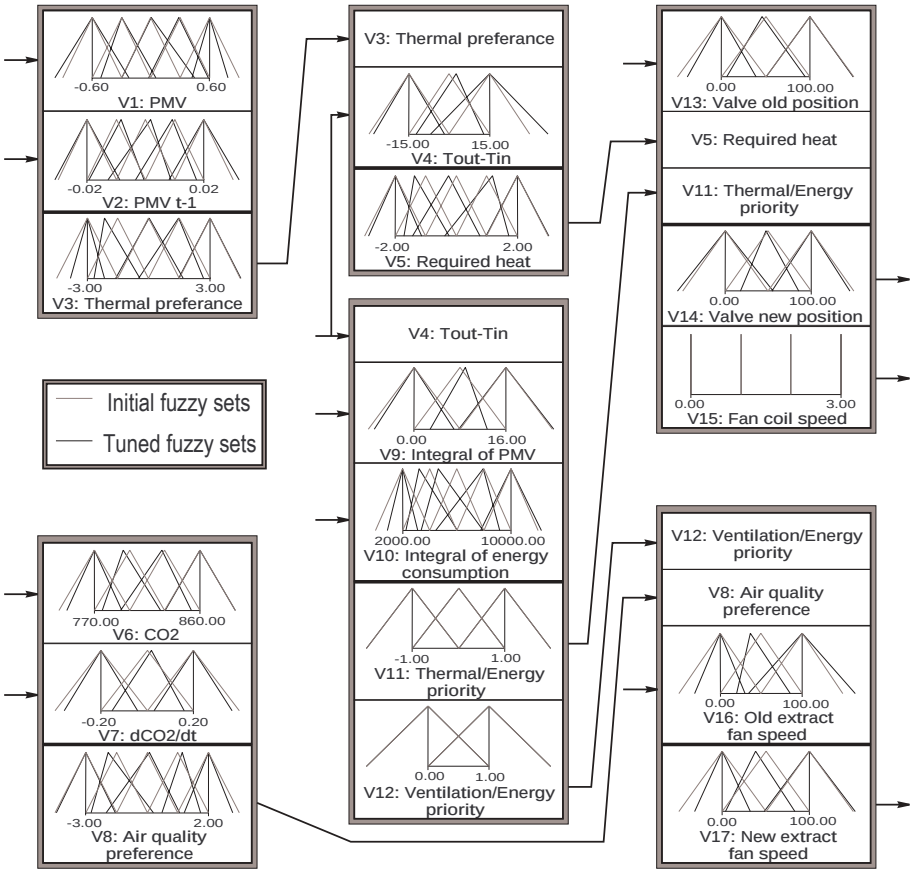


Figure 6: Initial and tuned DB of the ATC summer-season FLC

Figure 6 represents the initial and the final DBs for the ATC FLC taking as final DB the third solution from WMC-SSGA in Table 2. It shows that small variations in the membership function parameters provoke large improvements in the FLC behavior.

5.4 Methods Analysis

All the proposed techniques have yielded much better results than the classical On-Off controller, showing the good behavior FLCs can achieve on these kinds of complex multicriteria problems. In this case, WMC-SSGA presented the best results practically meeting all the requested goals. On the other hand, we also have

seen as MO-SSGA generates acceptable solutions with less computation time than WMC-SSGA. In any case, the computation time for WMC-SSGA is reasonable.

Therefore, from this experimental study, we could say that WMC-SSGA is the best strategy for this problem. However, we must think that it is possible thanks to properly guide the search by using trusted weights, while MO-SSGA perform a robust search. Therefore, in the case in which trusted weights can not be provided probably MO-SSGA would be the best strategy.

Therefore, the good results obtained by WMC-SSGA can be attributed to the use of a method of objective weighting that can directly guide to the best solution, to the use of fuzzy goals for dynamically adapting the search direction in the space of solutions, and to the restart approach getting away from local optima. In Figure 7 the way in which these factors affect to the fitness function can be easily observed.

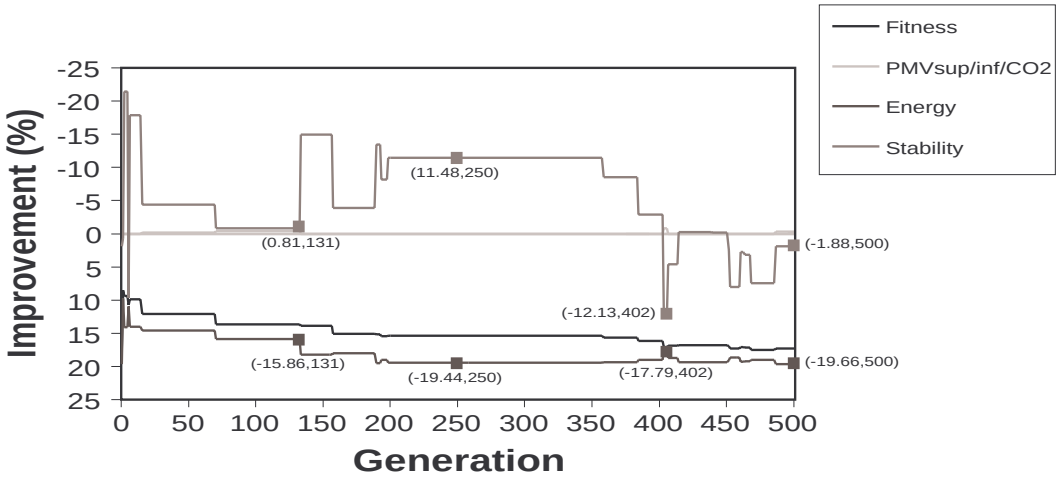


Figure 7: Evolution of the WMC-SSGA in the ATC summer season model

This figure illustrates the evolution chart of the fitness (expression without considering fuzzy goals) and performance values (C_i) obtained by the WMC-SSGA method when tuning the ATC summer model (PMVsup, PMVinf, and CO_2 improvements have been depicted with the same shade since they presents a very similar behavior with values near of 0%). The chart has been generated obtaining the values of the best individual (according to the fitness with fuzzy goals) in each generation. The improvement attained by the tuning process with respect to the On-Off controller solution is represented in vertical axis, where 0% stands for no improvement, negative value for a worsened result, and positive value for an improved result.

Analyzing the chart, we can observe how, after some initial generations where the algorithm is being stabilized, the energy consumption is gradually decreased until the generation 131 where almost 16% of improvement is achieved. Stability

and hence fitness are also improved during this period. After that, a significant improvement of the energy causes a worse stability to be obtained and the algorithm lies in a local optimum where an improvement of 19.4% for energy is obtained at the expense of stability, 11.5% worse than that of the On-Off controller. This is kept until the generation 402 where making the energy slightly worse involves finding a good stability result 12.1% better than the On-Off controller. This fact is derived from the restart action performed some generations before and it allows the algorithm to get away from the local optimum. From this generation to the end of the run, the energy is gradually improved with an acceptable stability that entails decreasing the fitness function value.

The obtained chart leads us to notice the restart influence and the convergence degree of this algorithm, and analyze the tuning process from the efficiency (time-consuming) point of view. From this angle, it is interesting to verify that a good solution where the energy consumption is improved in a 15.9% with the rest of performance values similar to the On-Off controller is obtained in less than 100 generations.

6 Concluding Remarks

In this paper, the use of GAs to develop smartly tuned FLCs dedicated to the control of complex multicriteria systems is presented. Two efficient genetic tuning strategies considering different multicriteria approaches have been proposed. Several FLCs have been produced and tested in laboratory experiments in order to check the adequacy of such tuning techniques.

From this experimental study, we could say that probably the objective weighting method WMC-SSGA is the best option when trusted weights are available while in the case in which trusted weights can not be provided probably a most robust technique as MO-SSGA would be the best option. Moreover, the proposed techniques have yielded much better results than the classical On-Off controller showing the good behavior that FLCs can achieve on these kinds of complex multicriteria problems.

The proposed tuning algorithms have an interesting advantage for industrial application: the consideration of goals to perform the multicriteria optimization. These goals significantly improve the tuning performance and it makes easier the expert's knowledge interpretation since the specification of goals, i.e., when each objective has been properly improved, seems to be easy to give. Furthermore, the use of fuzzy goals together with the penalization factor internally changes the initial proposed weights during the evolution of the WMC-SSGA algorithm, dynamically adapting the search direction in the space of solutions. It makes this method robust and more independent from the weight selection for the fitness function.

Acknowledgements

This work has been carried out within the framework of the JOULE-THERMIE programme under the GENESYS [1] Project sponsored by the European Commission, Directorate-General XII for Energy.

We would like to thank the GENESYS [1] project partners for their careful implementation of the presented models and for their invaluable assistance.

References

[1] J.E. Baker, Reducing bias and inefficiency in the selection algorithm, in: J.J. Grefenstette (Ed.), Proc. of the 2nd International Conference on Genetic Algorithms, Lawrence Erlbaum Associates (Hillsdale, NJ, USA, 1987) 14–21.

[2] P. Bonissone, Y. Chen, P. Khedkar, GA tuning of fuzzy logic controllers: A transportation application, Proc. of the 1996 IEEE Conference on Fuzzy Systems (FUZZ-IEEE'96), (New Orleans, LA, 1996) 674–680.

[3] C.A. Coello, A comprehensive survey of evolutionary-based multiobjective optimization techniques, Knowledge and Information Systems 1:3 (1999) 269–308.

[4] O. Cordón, F. Herrera, A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples, International Journal of Approximate Reasoning 17:4 (1997) 369–407.

[5] O. Cordón, F. Herrera, F. Hoffmann, L. Magdalena, Genetic fuzzy systems: Evolutionary tuning and learning of fuzzy knowledge bases (World Scientific, Singapore, 2001).

[6] K. Deb, Multi-objective optimization using evolutionary algorithms, (John Wiley & Sons, 2001).

[7] K. Deb, D.E. Goldberg, An investigation of niche and species formation in genetic function optimization, Proc. of the 3rd International Conference on Genetic Algorithms (1989) 42–50.

[8] D. Driankov, H. Hellendoorn, M. Reinfrank, An introduction to fuzzy control (Springer-Verlag, 1993).

[9] L.J. Eshelman, The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination, in: G.J.E. Rawlins (Ed.), Foundations of Genetic Algorithms (Morgan Kauffman, San Mateo, CA, 1990) 265–283.

[10] C.M. Fonseca, P.J. Fleming, Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization, in: S. Forrest (Ed.), Proc. of the 5th International Conference on Genetic Algorithms, Morgan Kaufmann (1993) 416–423.

[11] C.M. Fonseca, P.J. Fleming, An overview of evolutionary algorithms in multiobjective optimization, Evolutionary Computation 3 (1995) 1–16.

[12] M.P. Fourman, Compaction of symbolic layout using genetic algorithms, in: J.J. Grefenstette (Ed.), Genetic algorithms and their applications: Proc. of the 1st International Conference on Genetic Algorithms, Lawrence Erlbaum (1985) 141–153.

[13] P.Y. Glorennec, Coordination between autonomous robots, International Journal of Approximate Reasoning 17:4 (1997) 433–446.

[14] H.B. Gürocak, A genetic-algorithm-based method for tuning fuzzy-logic controllers, Fuzzy Sets and Systems 108:1 (1999) 39–47.

[15] P. Hajela, C.-Y. Lin, Genetic search strategies in multicriterion optimal design, Structural Optimization 4 (1992) 99–107.

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

[17] F. Herrera, M. Lozano, J.L. Verdegay, Fuzzy connectives based crossover operators to model genetic algorithms population diversity, Fuzzy Sets and Systems 92:1 (1997) 21–30.

[18] F. Herrera, M. Lozano, J.L. Verdegay, Tackling real-coded genetic algorithms: Operators and tools for the behaviour analysis, Artificial Intelligence Review 12 (1998) 265–319.

[19] J.H. Holland, Adaptation in natural and artificial systems (Ann arbor: The University of Michigan Press, 1975) (The MIT Press, London, 1992).

[20] C. Karr, Genetic algorithms for fuzzy controllers, AI Expert (1991) 26–33.

[21] F. Kursawe, A variant of evolution strategies for vector optimization, in: H.-P. Schwefel, R. Männer (Eds.), Proc. of the 1st workshop on Parallel Problem Solving from Nature, Springer-Verlag (1991) 193–197.

[22] E.H. Mamdani, Applications of fuzzy algorithms for control a simple dynamic plant, Proc. of the IEEE 121:12 (1974) 1585–1588.

[23] Z. Michalewicz, Genetic algorithms + data structures = evolution programs (Springer-Verlag, 1996).

[24] J.D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms, in: J.J. Grefenstette (Ed.), Genetic algorithms and their applications: Proc. of the 1st International Conference on Genetic Algorithms, Lawrence Erlbaum (1985) 93–100.

[25] N. Srinivas, D. Kalyanmoy, Multiobjective optimization using nondominated sorting in genetic algorithms, Evolutionary Computation 2:3 (1994) 221–248.

[26] G. Syswerda, J. Palmucci, The application of genetic algorithms to resource scheduling, in: R.K. Belew, L.B. Booker (Eds.), Proc. of the 4th International Conference on Genetic Algorithms (ICGA'91), Morgan Kaufmann (1991) 502–508.

[27] P. Thrift, Fuzzy logic synthesis with genetic algorithms, in: R.K. Belew, L.B. Booker (Eds.), Proc. of 4th International Conference on Genetic Algorithms (ICGA'91), Morgan Kaufmann (San Mateo, CA, 1991) 509–513.

[28] D.A. Van Veldhuizen, G.B. Lamont, Multiobjective evolutionary algorithms: analyzing the state-of-the-art, Evolutionary Computation 8:2 (2000) 125–147.

[29] D. Whitley, J. Kauth, GENITOR: A different genetic algorithm, Proc. of the Rocky Mountain Conference on Artificial Intelligence, Denver (1988) 118–130.

[30] D. Wienke, C. Lucasius, G. Kateman, Multicriteria target vector optimization of analytical procedures using a genetic algorithm — Part I. Theory, numerical simulations and application to atomic emission spectroscopy, Analytical Chimica Acta 265:2 (1992) 211–225.

[31] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evolutionary Computation 8:2 (2000) 173-195.