3

A General Study on Genetic Fuzzy Systems

OSCAR CORDÓN, FRANCISCO HERRERA

3.1 INTRODUCTION

As it is known, a rule based system (production rule system) has been successfully used to model human problem-solving activity and adaptive behavior, where a classic way to represent the human knowledge is the use of IF/THEN rules. The satisfaction of the rule antecedents gives rise to the execution of the consequent, one action is performed. The conventional approaches to knowledge representation are based on bivalent logic. A serious shortcoming of such approaches is their inability to come to grips with the issue of uncertainty and imprecision. As a consequence, the conventional approaches do not provide an adequate model for modes of reasoning and all commonsense reasoning fall into this category.

Fuzzy Logic (FL) may be viewed as an extension of classical logical systems, provides an effective conceptual framework for dealing with the problem of knowledge representation in an environment of uncertainty and imprecision. FL, as its name suggests, is the logic underlying modes of reasoning which are approximate rather than exact. The importance of FL derives from the fact that most modes of human reasoning -and especially commonsense reasoning- are approximate in nature. FL is concerned in the main with imprecision and approximate reasoning.

The applications of FL to rule based systems have been widely developped. From a very broad point of view a Fuzzy System (FS) is any Fuzzy Logic Based Sytems, where FL can be used either as the basis for the representation of different forms of knowledge systems, or to model the interactions and relationships among the system variables. FS have been shown to be an important tool for modelling complex systems, in which, due to the complexity or the imprecision, classical tools are unsuccessful.

Sample Contributed Book
Editor Jenny Smith

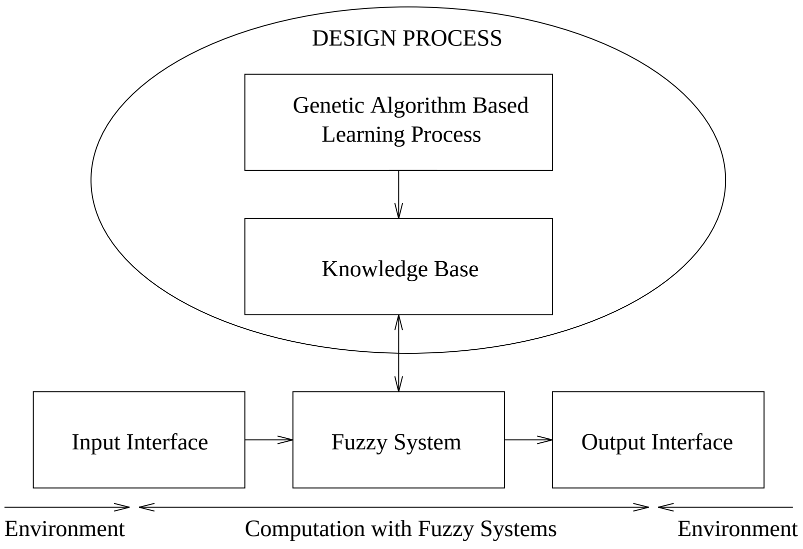

Figure 3.1 Genetic Fuzzy Sysems

Among the most successful applications of this systems has been the area of Fuzzy logic controllers (FLCs). FLCs are rule based systems useful in the context of complex ill-defined processes, especially those which can be controlled by a skilled human operator without knowledge of their underlying dynamics. Recentely fuzzy control techniques have been applied to many industrial processes, FLCs have been widthly used in automation and engineering. The experience of skilled operators and the knowledge of control engineers are expressed qualitetively by a set of fuzzy control rules. In fact, one of the features of the FLCs is that the *IF-THEN* rules are described on the base of the conventional control strategy and the experts' knowledge. Each fuzzy rule has an antecedent, or *IF*, part containing several preconditions, and a consequent, or *THEN*, part which prescribes the value.

Recentely, numerous papers and applications combining fuzzy concepts and genetic algorithms (GAs) have become known, and there is an increasing concern in the integration of these two topics. In particular, there are a great number of publications exploring the use of GAs for developping fuzzy systems, the called genetic fuzzy systems (GFSs). Figure 1 shows this idea.

This paper presents an overview of the GFSs, showing the use of the GAs in the construction of the fuzzy logic controllers knowledge bases comprising the known knowledge about the controlled system.

To achieve that, this paper is divided into 4 sections the first being this introduction. The section 2 introduces the fuzzy systems with a special attention to FLCs, while section 3 presents the GFSs. Some final remarks are made in section 4.

3.2 FUZZY SYSTEMS

Fuzzy logic and fuzzy sets in a wide interpretation of FL (in terms of which fuzzy logic is coextensive with the theory of fuzzy sets, that is, classes of objects in which the transition from membership to nonmembership is gradual rather than abrupt) have placed modeling into a new and broader perspective by providing innovative tools to cope with complex and ill-defined systems. The area of fuzzy sets has emerged following some pioneering works of Zadeh [Zad65, Zad73] where the first fundamentals of fuzzy systems were established.

As we aforesaid, a rule based system has been successfully used to model human problem-solving activity and adaptive behavior. The conventional approaches to knowledge representation, are based on bivalent logic. A serious shortcoming of such approaches is their inability to come to grips with the issue of uncertainty and imprecision. As a consequence, the conventional approaches do not provide an adequate model for modes of reasoning. Unfortunatelly, all commonsense reasoning fall into this category.

The application of FL to rule based systems leads us to the fuzzy systems. The main role of fuzzy sets is representing knowledge about the problem, or to model the interactions and relationships among the system variables. There are two essential advantages for the design of rule-based systems with fuzzy sets and logic:

- the key features of knowledge captured by fuzzy sets involve handling uncertainty, and
- inference methods become more robust and flexible with approximate reasoning methods of fuzzy logic.

Knowledge representation is enhanced with the use of linguistic variables and their linguistic values that are defined by context-dependent fuzzy sets whose meanings are specified by graded membership functions. On other hand, inference methods such as generalized modus ponens, tollens, etc., which are based on fuzzy logic form the bases of approximate reasoning with pattern matching scores of similarity. Fuzzy logic provides an unique computational base for inference in rule based systems. Unlike traditional logical systems, fuzzy logic is aimed at providing modes of reasoning which are approximate and analogical rather than exact.

Abording the fuzzy system modeling issue, it is essentially developed into two different types of system models identified as acquisition of rules and their parameters: i) fuzzy expert system models, and ii) fuzzy logic controllers.

Fuzzy expert system models are designed, developed and implemented with a direct participation of a system's expert who is throughly familiar with the characteristic behaviour of the system under investigation. The knowledge of the expert is extracted from the expert through experimental methods of questionnaires, protocols and interviews which may be conducted by people or by computers for the purpose of identifying the form and the structure of the rules, i.e., structure identification as well as the membership functions of the linguistic values of linguistic variables, i.e., parameter identification.

On the other hand, fuzzy control model design, development and implementation are dependent on the availability of input-output data sets. The system structure identification in terms of rules and specification of membership functions that define

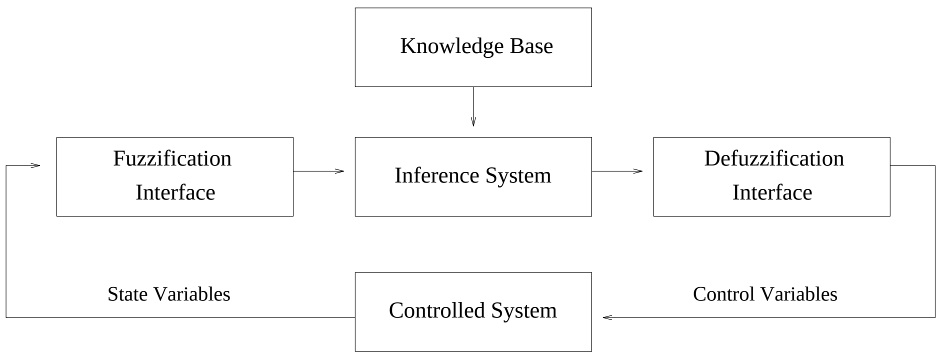

Figure 3.2 Generic structure of a fuzzy logic controller

the meaning representation of linguistic values of linguistic variables are determined by learning techniques. Also there is a third kind that is the combination of the others two that may be called *fuzzy expert-control.*

Here, we will center in the second fuzzy system model, the FLCs, where the GAs have been used for design the structure identification of the system.

FLCs, initiated by Mamdani and Assilian in the work [MA75], are now considered as one of the most important applications of the fuzzy set theory. FLCs are knowledge based controllers that make use of the known knowledge of the process, expressed in form of fuzzy linguistic control rules collected in a knowledge base (KB), to control it. The advantage of this approach with respect to the classical *Control Theory* is that it has not necessity of expressing the relationships existing in the system by means of a mathematical model, what constitutes a very difficult task in many real situations presenting nonlinear characteristics or complex dynamic. In the following two subsections we present an introduction to FLCs and to the FLC KBs.

3.2.1 Fuzzy Logic Controllers

An FLC is composed by a *Knowledge Base*, that comprises the information given by the process operator in form of linguistic control rules, a *Fuzzification Interface*, which has the effect of transforming crisp data into fuzzy sets, an *Inference System*, that uses them joined with the Knowledge Base to make inference by means of a reasoning method, and a *Defuzzification Interface*, that translates the fuzzy control action so obtained to a real control action using a defuzzification method. The generic structure of an FLC is shown in figure 2 [Lee90].

Several factors with a significant influence have to be analyzed in order to design an FLC for a concrete process. Concretely, there are two main decisions to make in order to design a FLC, to derive a KB for the system and to decide the reasoning method to use. As can be viewed, only the first one depends directly on the concrete application altough several reasoning methods will perform better with some kind of systems than others.

The **Knowledge Base** is the FLC component comprising the expert knowledge known about the controlled system. So it is the only component of the FLC depending on the concrete application and it makes the accuracy of the FLC depends directly on its composition. It is composed by a set of fuzzy control rules with the form:

If X_1 is A_{i1} and X_2 is A_{i2} and ... and X_n is A_{in} then Y is B_i

being the X_i and Y linguistic system variables and the A_i and the B_i linguistic labels associated with fuzzy sets specifying their meaning.

The **Fuzzification Interface** defines a mapping from an observed input space to fuzzy sets in certain input universes of discourse, obtaining the membership function associated to each one of the crisp system inputs.

The **Inference System** is based on the application of the Generalized Modus Ponens, extension of the classical logic Modus Ponens proposed by Zadeh. It is done by means of the Compositional Rule of Inference (CRI) given by the following expression:

$$\mu_{B'}(y) = Sup_{x \in X} \left\{ T^{'}(\mu_{A'}(x), I(\mu_{A_i}(x), \mu_B(y))) \right\} \tag{3.1}$$

being T, $T^{'}$ connectives, $\mu_{A_i}(x) = T(\mu_{A_{i1}}(x), \ldots, \mu_{A_{in}}(x))$ and I an implication operator.

Since the input x corresponding to the state variables of the controlled system is crisp, $x = x_0$, the fuzzy set $A^{'}$ is a singleton, that is, $\mu_{A'}(x) = 1$ if $x = x_0$ and $\mu_{A'}(x) = 0$ if $x \neq x_0$. Thus the CRI is reduced to:

$$\mu_{B'}(y) = I(\mu_{A_i}(x_0), \mu_B(y)) \tag{3.2}$$

Since from each rule R_i is obtained a fuzzy set $B_i^{'}$ from the inference process, the **Defuzzification Interface** uses an aggregation operator G which composes them and applies a defuzzification method D to translate the fuzzy sets obtained in this way into values corresponding to the control variables of the system. So, calling S to the FLC, x_0 to the inputs value and y_0 to the crisp value obtained from the defuzzification, we have:

$$\mu_{B'}(y) = G\left\{ \mu_{B_1^{'}}(y), \mu_{B_2^{'}}(y), \ldots, \mu_{B_n^{'}}(y) \right\} \tag{3.3}$$

$$y_0 = S(x_0) = D(\mu_{B'}(y)) \tag{3.4}$$

At present, the commonly used defuzzification methods may be described as the *Max Criterion*, the *Mean of Maximum (MOM)* and the *Center of Area (COA)* [Lee90].

The design tasks that have to be developed in order to decide the FLC reasoning method are the selection of the fuzzy operators I, T and G and the defuzzification operator D [KKS85]. The problem of selection them have been analyzed in several works such us [CCC$^+$94, CCC$^+$95, CHP95b, CHP95a, KKS85].

For more information about FLCs see [CHP95a, DHR93, HMB93, Lee90].

3.2.2 The Fuzzy Logic Controller Knowledge Base

The Knowledge Base is comprised of two components, a *Data Base* (DB), containing the definitions of the fuzzy control rules linguistic labels, that is, the membership functions of the fuzzy sets specifying the meaning of the linguistic terms, and a *Rule Base* (RB), constituted by the collection of fuzzy control rules representing the expert knowledge. We are going to analize more concretely this two components in the following sections.

The Data Base

The concepts associated with a DB are used to characterize fuzzy control rules and fuzzy data manipulation in an FLC [Lee90]. In this way, the main task to be done in order to design an FLC DB is to associate a membership function to every one of the linguistic terms that the system input and output variables can take as possible values. There are two modes of DB definition:

- By means of a quantization or normalization process.
- By means of a tuning or learning process.

Every linguistic variable involved in the FLC KB forms a fuzzy space with respect to a certain universe of discourse and have associated a label set containing the possible linguistic values that it can take. A *fuzzy partition* determines how many terms should exist in the label set. The choice of the term set is equivalent to finding the primary fuzzy sets or linguistic labels (terms). This is a previous task for the first mode and for some of the methods included in the second one.

The first definition mode makes use only of a little part of the a priori known knowledge of the system. In order to define the meaning of the linguistic term set, a discretization of the process input and output variables continuous universes of discourse have to be performed. This process is usually called *quantization* and is done in a number of steps [DHR93, HMB93, Lee90]:

1. The continuous domain is discretized (quantized) into a certain finite number of segments called quantization levels. Each segment is labeled as a generic element and the set of all generic elements forms a discrete universe of discourse.
2. Given a linguistic value from a certain term set, the fuzzy set defining the meaning of this linguistic value is built by assigning a degree of membership to each generic element. This is done for every linguistic value in a term set.

The use of quantized or normalized domains requires a scale transformation which maps the physical values of the process variables into the discretized universe. In both cases, the mapping can be uniform (linear), non-uniform (nonlinear) or both. Either a numerical or functional definition may be used to assign the degrees of membership to the primary fuzzy sets [DHR93, HMB93, Lee90]. This choice is based on the subjective criteria and it should be more convenient that the human processes operators could represent the meaning of their usually employed linguistic terms in form of fuzzy sets. Unfortunatelly, in many real cases it is not possible for him to make that, and it is very common the use of an uniform fuzzy partition as the one proposed by Liaw and Wang [LW91] shown in figure 3.

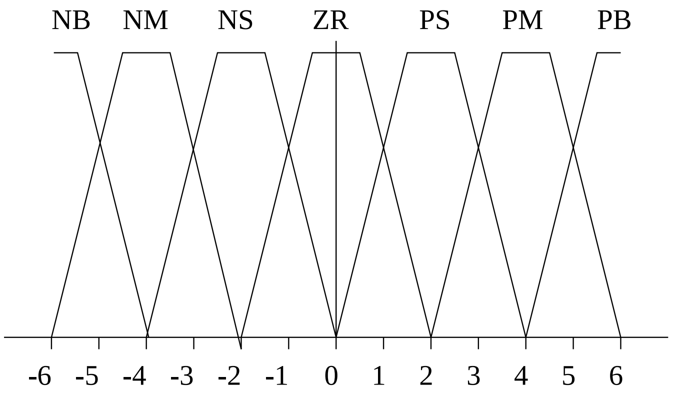

Figure 3.3 Fuzzy partition proposed by Liaw

The second definition mode employs many of the known knowledge and leads to a more application specific DB requiring less intervention of the controller designer and presenting more automatically developed tasks. Belonging to this group, we can distinguish two different subgroups:

- Definition by means of a tuning process
- Definition by means of a learning process

Methods included in the first one make use of a primary DB definition developed by means of a quantization or normalization process and then apply a process that modifies the meaning of the linguistic labels, that is, the definitions of the fuzzy sets associated to them. On the other hand, it would be a learning process when there is not an initial DB definition. Usually, this last process is carried out joined to a Rule Base learning process.

Moreover, an important decision in order to define the FLC is to determine the nature of its DB. There are two different approximations for it depending on the scope in what there is assigned the meaning to the linguistic labels belonging to the different term sets. On one hand, an usual DB definition process in what the fuzzy sets giving meaning to the linguistic labels are uniformly defined for all rules included in the RB constitutes a *descriptive* approach since the linguistic labels represents a real world semantics. On the other hand, it can be considered a KB whose rules present different meaning for the same linguistic terms. The meaning associated to a concrete label will depend on the concrete rule in what this label appears. In this case, the KB and the FLC using it present a different philosofy. The approach is *approximative* and the system is in the line of an *Universal Approximator* [Cas95].

The Rule Base

There are different kinds of fuzzy control rules proposed in the specialized literature regarding to the expression of the consequent. Mamdani and Assilian employ rules in which the consequent is another fuzzy variable [MA75], while Sugeno et al. use rules whose conclusion is a polynomial function of the inputs (TSK rules) [TS85]. Another kind of rules present also the consequent being a function of the input parameters. The following three rules show respectively the generical expressions of the three types commented:

$If\ X_1\ is\ A_1\ and\ ...\ and\ X_n\ is\ A_n\ then\ Y\ is\ B$

$If\ X_1\ is\ A_1\ and\ ...\ and\ X_n\ is\ A_n\ then\ Y = p_0 + p_1 X_1 + ... + p_n X_n$

$If\ X_1\ is\ A_1\ and\ ...\ and\ X_n\ is\ A_n\ then\ Y = f(X_1, ..., X_n)$

being the X_i and Y, linguistic system variables and the A_i and B, linguistic labels associated with fuzzy sets specifying their meaning.

There are four modes of derivation of fuzzy control rules that are not mutually exclusive [HMB93, Lee90]. These modes are the following:

1. Expert Experience and Control Engineering Knowledge.
2. Modeling of the Operator's Control Actions.
3. Based on the Fuzzy Model of a Process.
4. Based on Learning and Self-Organization.

The first method has been widely used. This method is effective when expert human operators can express what they use to control the system in terms of control rules. The rules more usually obtained by means of this process are Mamdani type because they present an adequate form to represent the expert knowledge. The second method directly models the control actions of the process operator. Instead of interviewing the operator, the types of control actions taken by him are modeled. The third approach is based on developing a model of the plant and construct an FLC to control the fuzzy model generating the fuzzy control rules of the RB by means of the fuzzy model of the system. It makes this approach similar to that traditionally used in Control Theory. Hence, structure and parameter identification are needed. Finally, the fourth method is focused on learning. In this case, the ability to create fuzzy control rules and to modify them based on experience in order to improve the controller performance is considered.

3.3 DESIGNING GENETIC FUZZY SYSTEMS

As it has been shown in the above section, there are many tasks that have to be performed in order to design an FLC to control a concrete system. We have commented yet that the derivation of the KB is the only one directly depending on the controlled system and it presents a significative importance in the design process. It is known that the more used method in order to perform this task is based directly on extracting the expert experience from the human processes operators. The problem arises when these are not able to express their knowledge in terms of fuzzy control rules. In order to avoid this drawback, researches have been investigating automatic learning methods for designing FLCs by deriving automatically an appropiate KB for the controlled system without necessity of its human operator.

The genetic algorithms (GAs) have demostrated to be a powerful tool for automating the definition of the KB since adaptative control, learning and self-organization can be considered in a lot of cases as optimization or search processes. The fuzzy systems making use of a GA in their design process are called generically GFSs.

These advantages have extended the use of the GAs in the development of a wide range of approaches for designing FLCs in the last years. Some of these approaches of *genetic FLC design* will be shown in the present chapter. It is possible to distinguish three different groups of *genetic FLC design processes* according to the KB components included in the learning process. These ones are the following:

1. Genetic definition of the Fuzzy Logic Controller Data Base.
2. Genetic derivation of the Fuzzy Logic Controller Rule Base.
3. Genetic learning of the Fuzzy Logic Controller Knowledge Base.

In the following subsections we are going to analyze each one of the approaches for genetic design of FLCs.

3.3.1 Defining the Fuzzy Logic Controller Data Base using Genetic Algorithms

As we have commented already, one of the modes of definition of the FLC DB is based on learning or tuning this FLC component. The difference between these two approaches depends on the existence of a previously primary DB definition. While learning processes do not need this previous definition, tuning processes works over it obtaining a more accurated one.

Several methods have proposed in order to define the FLC DB using GAs [BN95, BMU95, FTH94, HLV95b, HTS93, Kar91b]. All of them are based on the existence of a previously defined RB, usually extracted from the process operator. Each chromosome involved in the evolution process will represent different DB definitions, that is, each one of the chromosomes will contain a coding of the whole membership functions giving meaning to the linguistic terms. The degree of adaptation of an individual is measured using a fitness function that usually is based on the aplication of the FLC to the controlled system, using a KB formed by the RB and the DB encoded by the chromosome.

There are two different approaches for the genetic definition of FLC DBs depending on the scope of the association of membership functions to linguistic labels in the KB, either all fuzzy control rules using the same meaning for the system variables linguistic terms or a different approximation in what each rule presents its own meaning for the labels involved by it. In this subsection we analyze an example of each one of both groups. The method proposed by Karr [Kar91b], belonging to the first one, and the method of Herrera et al. [HLV95b], belonging to the second one.

The DB definition method proposed by Karr

The approach of Karr [Kar91b] is based on the existence of primary fuzzy partitions of the different system variables input and output spaces. The GA is applied for defining the meaning of the different linguistic term sets, that is, for learning the fuzzy sets associated to each one of the linguistic labels belonging to the fuzzy partitions. As it can be viewed, this approach is a descriptive one because all the fuzzy terms appearing in the fuzzy control rules will use the same meaning (that defined by means of the GA learning process).

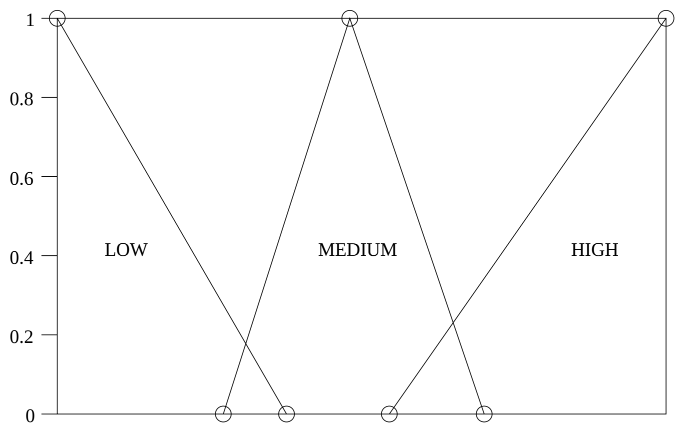

Figure 3.4 An example of fuzzy partition in the Karr's DB definition method

In order to develop this task, Karr uses a Simple GA with binary coding, proportional selection mechanism, simple crossover and random mutation. The binary coding outline will represent all the membership functions associated to the different linguistic labels belonging to each one of the linguistic term sets into a single chromosome.

The membership functions selected by Karr to define the meaning of the linguistic labels are triangular-shaped. He consider only two points, extremes of the fuzzy sets support, for defining every triangle, fixing its central point which presents value 1 in the membership function. In this way, the evolution performed by the GA can make the triangles to be distorted (when the base width is altered) and translated (when these two points are shifted along the x-axis) freely. The process is different for the extreme triangles, which only requires a single point with value of the membership functions equal to 0 to be defined. Their modal point is clearly placed in the correspondent extreme of the concrete system variable universe of discurse. Only one operation can be performed with these triangles, altering their base width in one of the two x-axis directions, that is, making it bigger or smaller. Figure 4 shows an example.

The constraint placed over the membership functions is that the ones associated to the extreme labels must remain right triangles while those associated to the interior terms must remain isosceles triangles. It is clear that this constraint avoid the GA to obtain incorrectlly defined membership functions.

A chromosome coding all the membership functions is built finally by joining all the individual coding of these ones into a single string. As each point is represented by a binary number with a fixed number of bits, the chromosomes are of fixed-length, that is, all individuals in the population will present the same length.

Finally, Karr do not present a concrete fitness function but introduces several considerations in order to define it. His idea for measuring the accuracy of a concrete DB in the optimal control of the system is based on an application-dependent measure, that is, any error or convergence measure (see [CCC$^+$94, CHP95a]).

The DB definition method proposed by Herrera et al.

In [HLV95b] it was presented a DB definition process used to tune the DB parameters. The process is based on the existence of a previous complete KB, that is, an initial DB definition and a RB constituted by m control rules. The chromosomes will encode a complete KB since each one of them contains the RB with a different DB associated.

The GA designed for the process present real coding issue and use the stochastic universal sampling as selection procedure and the Michaelewicz's non-uniform mutation operator. Regarding to the crossover, two different operators are employed: the simple and the Max-Min-Arithmetical crossover. This last operator have been proposed by the authors and makes use of fuzzy operators in order to improve the behavior of the GA crossover operator.

The membership functions selected in order to define the DB are trapezoidal- shaped. They have associated a parametric representation based on a 4-tupla of real values. Let the following rule be the *ith* rule of the previous RB:

$$\text{If } X_1 \text{ is } A_{i1} \text{ and } X_2 \text{ is } A_{i2} \text{ and } \ldots \text{and } X_n \text{ is } A_{in} \text{ then } Y \text{ is } B_i$$

Then the fuzzy sets giving meaning to the linguistic labels A_{ij} associated to the input variables X_i will be represented by the 4-tuple $(c_{ij}, a_{ij}, b_{ij}, d_{ij})$ and the ones associated to the output variable linguistic labels B_i by $(c_i^{'}, a_i^{'}, b_i^{'}, d_i^{'})$. Thus each one of the rules will be encoded in pieces of chromosome C_{ri}, $i = 1, \ldots, m$, in the following way:

$$C_{ri} = (c_{i1}, a_{i1}, b_{i1}, d_{i1}, ..., c_{in}, a_{in}, b_{in}, d_{in}, c_i^{'}, a_i^{'}, b_i^{'}, d_i^{'}) \tag{3.5}$$

Therefore the complete RB with an associated DB is represented by a complete chromosome C_r:

$$C_r = C_{r1}\ C_{r2}\ ...\ C_{rm} \tag{3.6}$$

As it can be viewed, each individual of the population represents a complete KB. More concretelly, all of them encode the derived system RB and the difference between them is the meaning associated to the linguistic variables taking part in the fuzzy control rules, that is, the DB definition. As each rule is coded in a piece of chromosome, the fuzzy set giving meaning to a linguistic term can be changed in one rule and not in the other ones in which it appears or, in a more extreme case, it can be changed in many rules presenting different forms in several ones. The meaning of the linguistic terms will depend then on the rule in which they are involved. The KB so obtained will present an approximative behavior.

The initial gene pool is created from the initial KB. This KB is encoded directly in a chromosome, denoted as C_1. The remaining individuals are generated by associating an interval of performance, $[c_h^l, c_h^r]$ for every gene c_h of C_1, $h = 1 \ldots (n+1) \times m \times 4$. Each interval of performance will be the interval of adjustment for the correspondent variable, $c_h \in [c_h^l, c_h^r]$.

If $(t \bmod 4) = 1$ then c_t is the left value of the support of a fuzzy number. The fuzzy number is defined by the four parameters $(c_t,\ c_{t+1},\ c_{t+2},\ c_{t+3})$ and the intervals of performance are the following:

$$c_t \in [c_t^l, c_t^r] = [c_t - \tfrac{c_{t+1} - c_t}{2}, c_t + \tfrac{c_{t+1} - c_t}{2}]$$

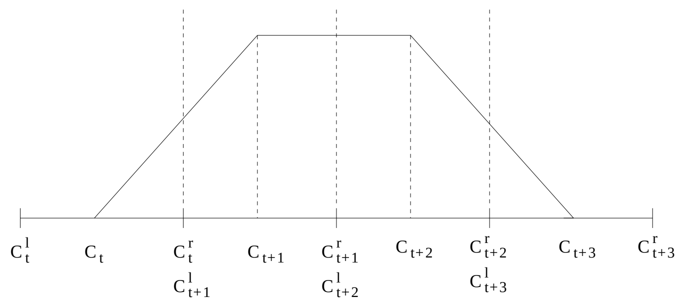

Figure 3.5 Intervals of performance

$$\begin{aligned}
c_{t+1} &\in [c_{t+1}^l, c_{t+1}^r] = [c_{t+1} - \tfrac{c_{t+1}-c_t}{2}, c_{t+1} + \tfrac{c_{t+2}-c_{t+1}}{2}] \\
c_{t+2} &\in [c_{t+2}^l, c_{t+2}^r] = [c_{t+2} - \tfrac{c_{t+2}-c_{t+1}}{2}, c_{t+2} + \tfrac{c_{t+3}-c_{t+2}}{2}] \\
c_{t+3} &\in [c_{t+3}^l, c_{t+3}^r] = [c_{t+3} - \tfrac{c_{t+3}-c_{t+2}}{2}, c_{t+3} + \tfrac{c_{t+3}-c_{t+2}}{2}]
\end{aligned}$$

Figure 5 shows these intervals. Therefore, we create a population of chromosomes containing C_1 as its first individual and the remaining ones initiated randomly, with each gene being in its respective interval of performance.

By using a training input-output data set, E_{TDS}, and a concrete error measure, the square medium error, the fitness function of a chromosome is defined. In this way, the adaptation value associated to an individual is obtained by computing the error between the outputs given by the FLC using the KB encoded in the chromosome and those contained in the training data set. The fitness function is represented by the following expression:

$$E(C) = \frac{1}{2|E_{TDS}|} \sum_{e_k \in E_{TDS}} (ey^k - S(ex^k))^2 \tag{3.7}$$

3.3.2 Deriving the Fuzzy Logic Controller Rule Base using Genetic Algorithms

The great majority of the approaches belonging to this group are based on learning the consequents of the fuzzy control rules included in the FLC RB. In this way, many of these genetic processes encode the complete system decision table in the chromosomes. There are different methods developing this task [Bon93, HT94, Kar91a, KB93, Thr91] but in many cases the only difference existing among them is the cappability of learning the number of rules forming the RB. This characteristic is presented when there exist a possible alelle representing the absence of consequent for a rule with a concrete antecedent, that is, the absence of the correspondent rule in the RB.

All methods belonging to this family suppose the existence of a defined DB. Thus, a primary partition of the fuzzy spaces, that is, a term set for each one of them, and a collection of fuzzy sets giving meaning to these primary labels are considered. Other common characteristic for a majority of them is that they consider the FLC RB represented in form of *decision table* (called too *look-up table*). As it is known, an usual FLC RB constituted by control rules presenting n input variables and a single output variable can be represented using an n-dimensional decision table, each dimension corresponding to each one of the input variables. Every dimension will have associated an array containing the labels of the concrete variable term set and the cells of the decision table will contain the linguistic label that the output variable take for the combination of antecedents represented in this cell. Therefore each table cell represents a fuzzy control rule that can belong to the FLC RB.

This structure is encoded in the individuals forming the GA population. If there not exist a value for the alleles representing the absence of value for the rule consequent, there is not possible to derive a FLC RB with an optimal number of rules but all the possible rules have to be considered belonging to it. This is the case of the methods proposed by Karr [Kar91a], and Kropp and Baitinger [KB93] altough the first of them do not encode the complete decision table as we are going to see in the following. The remaining ones commented are able to learn the number of fuzzy rules.

In this subsection we are going to study three different approaches. The methods selected were proposed by Thrift [Thr91], Karr [Kar91a] and Bonarini [Bon93]. This last one constitutes an original approach for learning FLC RB and differs a lot from the others belonging to this family as we are going to see in the following.

The RB derivation method proposed by Thrift

This method, as many others belonging to these category, is based on encoding all the cells of the complete decision table in the chromosomes. In this way, Thrift [Thr91] establishes a mapping between the label set associated to the system output variable and an increasing integer set (containing one element more and taking 0 as first element) representing the alelle set. An example is shown to clarify the concept. Being $\{NB, NS, ZR, PS, PB\}$ the term set associated to the output variable, it can be noticed the absence of value for the output variable by the symbol -. The complete set formed joining this symbol to the term set is mapped into the set $\{0, 1, 2, 3, 4, 5\}$. Hence the label NB is associated with the value 0, NS with 1, ..., PB with 4 and the blank symbol - with 5.

Therefore the GA emploies an integer coding. Each one of the chromosomes is constituted by joining the partial coding associated to each one of the linguistic labels contained in the decision table cells. A gen presenting the alelle - will represent the absence of the control rule contained in the corresponding cell in the RB being the chromosome phenotype.

The GA proposed emploies an elitist selection scheme and the genetic operators used are of different nature. While the crossover operator is the standard two -point crossover, the mutation operator is specifically designed for the process. When it is applied over an alelle different from the blank symbol, changes it either up or down one level or to the blank code. When the previous gen value is the blank symbol, it selects a new value at random.

Finally, the fitness function is based too on an application specific measure. A Measure of Convergence is considered for this task. The fitness of an individual is determined by applying the FLC employing the RB coded in its genotype to the controlled system with several different starting points and computing the convergence of it to the desired equilibrium point.

The RB derivation method proposed by Karr

Karr proposes a method for deriving FLC RBs [Kar91a] where the complete RB in form of decision table is not coded in each chromosome. The approach of Karr is based in the fact that the number of rules is provided by an expert, together with many complete rules forming the RB and the antecedents for the reamining ones. He presents an GA derivation method for learning the rule consequents of this last type rules.

As it can be viewed, the method needs the existence of a deep knowledge about the controlled system. As Karr comments, his approach is useful when the controller human designer knows all the input states relevant for the problem (in this way, he knows the number of rules needed to control it) and the control action that have to be performed in order to obtain an optimal control from several of them (that is, he knows several complete antecedent-consequent control rules), but the actions that have to be associated to the remaining input states are not obvious for him. Usually, the known rules are those describing the extreme conditions, which are easy to write in many cases.

The coding outline will associate a binary number to each label belonging to the output variable term set in the way that each of them will be mapped to a binary number according to its order in the label set. Thus, considering seven linguistic terms describing the control variable, the possible control actions for one rule will be represented as a three-bit string (000 represents action 1 (NB), 001 represents action 2 (NM), and so on). A chromosome is obtained by joining the partial codings of the rules for what there is not known the control action to apply. Therefore a chromosome codes only a little part of the complete RB. In the commented example, calling n the number of not determinated rules, a string of length $3n$ represents every possible configuration for this partial FLC rule set.

The GA employed is the same used for the Karr's DB definition method that we have commented in the previous subsection, that is, a Simple GA with binary coding, proportional selection mechanism, simple crossover and random mutation. In the same way, the fitness function is defined by using an application specific measure.

Finally, it is important to note that the author proposes to combine their two FLC genetic design methods, the DB definition method and the RB derivation method, in order to design a complete KB. In this way, the RB derivation method is applied in a first step and, once an acceptable RB have been learned by the GA, the selection of high-performance membership functions (an accurated DB) is carried out using the above described tuning process.

The RB derivation method proposed by Bonarini

The method proposed by Bonarini [Bon93], called ELF (Evolutionary Learning of Fuzzy Rules), is quite different to the other approaches developed into this group. The author considers the high computationally cost needed to derive a RB by applying an GA with a population of individuals coding the whole decision table and presents a model based on learning only the control rules that the FLC will use.

In order to develop this task, the GA used by Bonarini maintains a population of rules This new coding will allow to learn the complete structure of the control rules, that is, their antecedent and consequent and the optimal number of rules forming the FLC RB. The author wants to learn with ELF what states occur in the controlled system and which are irrelevant for it, obtaining optimal RBs for the application.

The fitness function judges the state reached at each activation of the rules. Each individual of the population, that is, each rule, will have associated information about several questions: how good it has been judged (its strength), when it has been generated, when it has been triggered the last time and how much it contributed to past control actions performed by the controller. ELF will modify the strength associated to a rule according to the performance of the action it contributed to. This performance is evaluated by the fitness function.

Other important characteristic of this method is that it is designed in order to run in a real enviroment. The process first selects among the rules matching the current state of the controlled system, the rules matching better than a degree given by the FLC designer. The rules belonging to this set will compete between them in order to propose the best control action for the current state.

Several genetic operations are applied over this set of individuals. Some of these rules are considered tested enough since they contributed to past control actions more than a given reference. If some of them have low strength, it means that the control actions proposed by them do not perform well. These rules are substituted by others performing better and the consequents of such rules are modified with a probability proportional to their strength. This last step constitute the GA mutation operator and makes ELF to look for new rules in a neighboorhood of the good ones previously learned.

If much time have passed from the last rule modification, this means that the population of rules matching well the current state have stucked and all of them have almost the same strength. In this case, ELF selects the worst rule and mutates its consequents in order to continue the search looking for a better configuration.

If there exists few rules matching the current state (the parameter representing the enough number of rules is changed dinamically), ELF generates a new rule covering it and proposes a control action at random. This is the only mechanism introducing new antecedents and it is called *cover-detector*. It may introduce with a given probability also "dont care" symbols as values for some variables in the new rule generated. Rules whose antecedents contain these kind of symbols match different states and compete with different groups of rules, one for each of the matching states.

The dual situation is in which there are too many rules matching a state. In this case, the genetic operator applied is called *rule-killer* and it simply eliminate the worst rule matching this state from the population.

One time performed the selection and applied the genetic operators, ELF uses the FLC with the RB coded in the population taking the current state as input. The fitness function will then evaluate the new state obtained, giving it a reward. There exists a sharing process in the distibution of this reward, each individual gets part of the total reward according to the contribution of the rule that it codes gave to the control action applied. This process results in a modification of the strenght associated to the rules coded in the population.

Finally, several questions have to be noted. On one hand, Bonarini proposes application specific performance measures to define the fitness function. On the other hand, ELF gives different RBs as output of the learning process. It is due to each time the performance of the system is higher than a ”satisfactory” value given by the controller designer, it saves the current RB and forces modifications in the current population in order to obtain a better solution.

3.3.3 Learning the Fuzzy Logic Controller Knowledge Base using Genetic Algorithms

This last group is the one to with more contributions have been made in the last years. There exists many approaches for genetic learning complete FLC KBs such as [CV93, HLV94, HLV95a, LT93, LP94, LM94, NL94, NHW92, SK94, VM95] and all of them present different characteristics. Belonging to these great group of works, we find approaches presenting variable cromosomal length [CV93, LP94], others making use of many expert knowledge in order to improve the learning process [HLV95a, LT93] and several working with chromosomes encoding single control rules instead of complete KBs [HLV95a, VM95]. Many of them define the fitness function by means of a single application specific measure, usually a measure of convergence, while others include more objective to optimize for obtaining more robust KBs [HLV95a, LT93]. In the following we are going to study five different approaches belonging to this family.

The KB learning method proposed by Lee and Takagi

In [LT93] Lee and Takagi introduce an GFS design method that allows to learn automatically a complete FLC KB. The method is based on an RB formed by TSK control rules. In this way, the derivation of the rule consequents will consist on learning the weights w_i used for combining linearly the input values in order to obtain the outputs.

For the linguistic terms in the rule antecedents, the authors consider triangular-shaped membership functions although they note that the method can work with any kind of parametrized membership functions, such as Gaussian, bell, trapezoidal or sigmoidal ones. They suppose a fuzzy partition of the input space. Each triangular membership function associated to each one of the primary fuzzy sets (linguistic terms) is represented by three parameters. The first one is the center point of the triangle, that is, its modal point. Only the center of the triangle associated to the first primary fuzzy set is given as an absolute position, while the parameters associated to the other ones represent the distance existing from the center point of the current triangle to the previous one. The other two parameters represent the left and right point of the triangle base respectively and they both present membership function value 0.

The GA used emploies a binary coding. First, the membership functions are encoded by taking the binary values of the three associated parameters and joining them into a binary substring. Eight bits are used to encode each parameter value. Then, the complete DB is encoded by joining the partial codings of the membership functions associated to each one of the input variables primary fuzzy sets one after other. The last part of the chromosome is built by coding the parameters w_i associated to each combination of the input values and joining them into a new binary substring. Eight bits are used again to encode the values of these parameters. In this way, each chromosome represents a complete KB. The number of rules forming the KB will depend on the number of primary fuzzy sets associated to each one of the input variables and it will be equal to the product of them.

The coding used allows to decide the optimal number of control rules forming the RB in the following way. Those linguistic terms in what the center point of their associated triangular membership functions are out of a concrete bound, obtained from the known knowledge about the system, are not considered to belong to any rule in the RB. Thus all rules whose antecedent should be a combination of the valid input values coded in the first part of the chromosome, will belong to the FLC RB. As an example, let us consider a chromosome coding the KB of a system with two input variables having associated m and n valid linguistic labels respectively. The whole number of rules of the RB encoded in it will be $m \times n$.

The fitness function is based on optimizing two different criteria. On one hand, an application specific measure is considered (concretely, in the example proposed it is used a measure of convergence and the FLC is applied from different initial states). On the other hand, chromosomes coding RBs with a large number of rules are penalized in order to obtain others with less rules.

Finally, it is very important to note that the authors propose two different ways of incorporating previous known knowledge about the system to the proposed GFSs design method in order to improve its behavior. They comment that this previous knowledge will make the GA gain significant speedup if the solutions designed by means of it are approximatelly correct. On one hand, it is possible to incorporate knowledge via the initial settings for the FLC KB parameters. This knowledge is used to generate the GA initial population. Thus, the individuals forming it are not all generated at random but several of them are obtained by equally partitioning the input spaces into varying number of linguistic terms. This knowledge can also be used to initially set appropriately the parameters w_i of the rule consequents. On the other hand, the previous knowledege can be incorporated via structural representation of the KBs. For example, in a problem of symmetric nature as the Inverted Pendulum, the FLC can be constrained in order to simetrically partition the input space about the origin. This will reduce the size of the search space because the number of membership functions is reduced by half.

The KB learning method proposed by Ng and Li

The method proposed by Ng and Li [NL94] is very specific since it is designed for applying in two-inputs-one-output systems all whose input and output spaces are partitioned into exactly seven primary fuzzy sets (the term set of the three variables is the same: $\{zero, \pm small, \pm medium, \pm large\}$). On the other hand, it is not able to learn the number of rules that will constitute the optimal RB. As we are going to see in the following, the whole decision table is represented in the part of chromosome corresponding to the RB. Therefore, the way for derivating the RB is the same used by the methods belonging to the second group commented in this work. The look-up table will allways present dimmension 7×7 and all the rules contained in it will belong to the final RB.

The DB considered is designed by using symmetrical exponential membership functions defined by the following expression:

$$\mu_{\pm i}(x) = exp\left(-\frac{|x \pm \alpha_i|^{\beta_i}}{\sigma_i}\right) \tag{3.8}$$

considering that $\mu_{+large} = 1$, if $x > \alpha_{+large}$ and $\mu_{-large} = 1$, if $x < \alpha_{-large}$.

Hence, it is possible a parametric representation of the membership functions by means of the set of parameters (α, β, σ), representing respectively the position, shape and scaling parameters.

The most important characteristic of this approach is that the FLC design space is coded in base-7 chromosomes. As in the great majority of the methods belonging to this group, each chromosome represents a complete KB. Each chromosome is built by joining five substrings coding the different parameters of the problem. The first one of them represents the FLC RB and, as we have commented yet, the process is similar to this developed in the methods deriving only the FLC RB. The fourty nine rule consequents contained in the decision table cells are encoded one after other in a substring presenting a base-7 value for each one of them.

The second substring is constituted by the scaling parameters (σ) associated to the seven linguistic labels of the two input variables. It presents fourteen base-7 values, two for each one of the following parameters: $\sigma_{\pm large}$, $\sigma_{\pm medium}$, $\sigma_{\pm small}$ and σ_{zero}. The next substring of eight base-7 values represent the positions (α) of the fuzzy sets associated to the terms "small" and "medium" (each one requiring two bits), whilst the positions of those giving meaning to "large" and "zero" are fixed ($\alpha_{-large} = -3$, $\alpha_{+large} = 3$ and $\alpha_{zero} = 0$). The fourth substring represents K_1 and K_2 as gains of the two inputs variables, error and change of error, with three base-7 bits associated to each and, finally, the last group of eigth integer characters encodes the shape coefficients β of the fuzzy sets associated to the term set of both variables, requiring a base-7 value for each parameter.

The fitness function employed by the GA is application specific. It is based directly on computing the value obtained by the FLC in a measure of convergence when controlling the system by using the concrete KB encoded in the individual.

The KB learning method proposed by Leitch and Probert

The more important characteristic presented by the method proposed by these authors [LP94] is the specific coding outline that they design in order to represent FLC KBs into chromosomes in an efficient way. The coding proposed will make the genetic variation to be increased, reducing in this way premature convergence and avoiding the need for complex crossover operators.

Leitch and Probert begin studying the previous approaches for designing GFSs and comment that the coding schemes employed by the different GAs developed for this task in the specialized literature represent the KB in a very fixed form. In order to avoid this shortcoming they develop a coding scheme more flexible than the position dependent ones commented. In their *context depending coding* the meaning of a piece of chromosome is not determined by the absolute position that it presents into the genotype, as usual, but it is determined by surrounding genes contained into it. In this way, the chromosomal length is variable.

They consider a DB with spherical fuzzy sets each one determined by its concrete center point presenting height 1, and all of them by a global radius value. All inputs and output are scaled to the interval $[0, 1]$ and the values of the commented parameters are encoded by using binary substrings of variable length. The parameter value is decoded in the following way. Let it be a binary substring of length n whose integer value is $m \leq n$. The real number encoded in it is $\frac{m}{2^n - 1}$.

The method is able to learn the optimal number of rules forming the RB, the input variables involved in each rule and the shape of the membership functions associated to the input and output variables. As it can be viewed, it does not use a previously defined fuzzy partition of the input and output spaces but learn directly the membership functions associated to each variable involved in each concrete rule. Therefore, the KB obtained from the learning process clearly present an approximative behavior.

Each chromosome encodes a complete KB by using the context depending coding commented. The alphabet of the chromosome consists of the integers 0, 1 and the letter E, used to indicate the end of a number (the value E is associated to a surrounding gen). The encoding of a rule is obtained by joining several substrings separated by E symbols. The first one presents a bit for each system input variable, taking value 1 if the concrete variable is involved in the rule and 0 otherwise. The surrounding genes contained in this substring are ignored. Next substring encodes the center of the membership functions associated to the input variables involved in the rule, each one separated from each other by a surrounding gen. An empty value is associated to the variables not taking part in the rule. The third substring presents only a binary number coding the membership functions radius. The value encoded represents a percentage from a maximum radius value given by the controller designer. The last substring encodes the central point of the membership function associated to the output variable.

A chromosome is obtained by joining the partial codings of the rules contained in the KB. The evolutionary process can make the end of a chromosome not coincide with the end of a rule (it rarely does). In this case, the incomplete rule is ignored and the genes coding it (called junk genes) are used to limit the effect of the disruption caused by the genetic operators.

Regarding to these ones, single point crossover is employed with a little modification in its usual form: the crossover point is randomly selected on each of the chromosomes involved. On other hand, classical mutation operator is used. Due to the coding scheme employed by the GA, both operators can not produce illegal chromosomes.

We finish noting several remarks about the method. The initial population formed by chromosomes of random initial length is generated at random. The fitness function emploies a measure of convergence since the authors enunciate that this is the more efficient way to design GFSs. Several noise is introduced in the simulations in order to obtain more robust KBs from the learning process.

The KB learning method proposed by Cooper and Vidal

The underlying idea in the approach proposed by Cooper and Vidal [CV93] is that the excesive length of the chromosomes encoding the KB can make the GA not able to find accurated solutions due to the high complexity of the search space. In this way, they propose a novel encoding scheme which maintains only those rules necessary to control the target system, alowing to obtain an RB with optimal number of rules.

In this case, the membership functions contained in the DB are triangles characterized by the location of its center and the half-length of its base. A single rule, therefore, consists of the concatenation of the one-byte unsigned characters (assuming values from 0 to 255) specifying the centers and half-lengths of the membership functions. The rule descriptions for a single KB are then concatenated into a single bit string where the number of rules is not restricted. Hence, the GA employs a integer coding and the chromosomes present variable-length.

This GA does not use the classical genetic mutation operator. In this approach, this operator include the inversion of the copied bit and the addition or deletion of an entire rule. These latter two mutations permit the size of a FLC KB to evolve. The cycle of evaluation and reproduction continues for a predetermined number of generations or until an acceptable performance level is achieved.

To be meaningful, the genetic paradigm requires that the rules in the two strings should be aligned so that similar rules are combined with each other. Simply by combining the strings in the order they appear it does not preserve much information about either KB encoded and produces nearly random results, rather as a child KB that performs in a manner similar to its parents. For example, this can make that the child chromosome will present repeated rules. Therefore, before reproduction, both strings must be aligned so that the centers of the input variables match as closely as possible. The most closely matching rules are combined first, followed by the next most closely matching rules from those remaining and so on. Any rules forming a longer string that is not matched are added at the end.

The fitness function is designed by using a measure of convergence. The FLC is used to control the system from twenty test cases and its accuracy is measured during sixty seconds each trial.

The KB learning method proposed by Herrera et al.

The KB learning process proposed in [HLV95a] presents several important differences with respect to the other methods belonging to the same group as we are going to see in the following. First, it can be considered the existence of previous control rules derivated directly from the human process operator and include them in the learning process, being able to combine them with other rules automatically learned and to detect incorrect rules into this previous set. Moreover, the approach is based on several steps and not in a single process such in the other methods. The chromosomes of the main GA represent single rules and not complete KBs. On other hand, the fitness function is based not only in an FLC performance measure but it include several criteria in order to obtain optimal KBs. Finally, the approach will obtain a KB with an approximative nature even in the case in which there exists several descriptive fuzzy control rules provided by the expert.

The approach is based on the use of GAs under the following hypotheses:

- There is some linguistic information from the experience of the human controller but linguistic rules alone are usually not enough for designing successfully a control system or could not be available.
- There is some numerical information from sampled input-output (state-control) pairs that are recorded experimentally.
- The combination of these two kinds of information may be sufficient for

a successful design of a FLC KB.
- The possibility of not having any linguistic information and having a complete numerical information is considered.

Taking into account the aforementioned hypothesis a learning process is designed according to the following goals:

- to develop a KB generating process from numerical data pairs; and
- to develop a general approach combining both kinds of information, linguistic information and fuzzy control rules obtained by the generating process, into a common framework using both simultaneously and cooperatively to solve the control design problem.

In order to reach these goals, it is proposed a methodology based on the design of the three following components:

1. a KB generating process of desirable fuzzy control rules able to include the complete knowledge of the set of examples,
2. a combining information and simplifying rules process, which finds the final KB able to approximate the input-output behaviour of a real system,
3. a tuning process of the final KB DB,

all of them developed by means of GAs.

As it is possible to have some linguistic $IF - THEN$ rules given by an expert, it is used a linguistic fuzzy rules structure to represent them. That is, a previously defined DB representing fuzzy partitions with real-world meaning is considered. On other hand, there are sampled input-output pairs and a free fuzzy rules structure to generate the fuzzy rules covering these examples is used (the meaning presented by alinguistic term is different when it belongs to different rules, that is, the example set are used to generate rules with an approximative behavior). The membership functions of the linguistic labels involved in the control rules are trapezoidal-shaped. Then both kind of rules are combined, by using a simplification method based on a GA, and finally a tuning method is applied over the simplified set of rules that will make all of them finally present an approximate behavior.

The generating fuzzy rules process consists of a *generating method* of desirable fuzzy rules from examples using GAs together with a *covering method* of the set of examples.

- The generating method of fuzzy rules is developed by means of a real coded GA (RCGA) where a chromosome represents a fuzzy rule and it is evaluated by means of a frequency method. The RCGA finds the best rule in every running over the set of examples according to the following features which will be included in the fitness function of the GA.
- The covering method is developed as an iterative process. It permits to obtain a set of fuzzy rules covering the set of examples. In each iteration, it runs the generating method choosing the best chromosome (rule), assigns to every example the relative covering value and removes the examples with a covering value greater than a value ϵ provided by the controller designer.

Because we can obtain two similar rules in the generating process or one rule similar to another given by an expert, it is necessary to combine and simplify the complete KB obtained from the previous process for deriving the final KB allowing to control the system. Finally, the tuning method presented in [HLV95b] and commented in a previous subsection is applied over the simplified KB for obtaining a more accurated one.

3.4 FINAL REMARKS

In this chapter we have reviewed some approaches of GFSs in which GAs have been used for designing FLCs KBs. Three different modes to cope this problem have been attached and different methods developing each one of them have been analyzed.

There are many others researchs that have contributed to these three areas as we have cited above. Moreover, there are other researchs that have contributed to various aspects of GFSs, not included in the three GFS designing modes aforementioned, which we have not a change to deal with but are worthy to mention. They are commented in the following.

In [SKG93] it is proposed an automatic design method combining self-organizing feature maps and GAs. In a first step, the fuzzy control rules and linguistic variables are extracted from a referential data set by using a self-organizing process. Then a GA is used to find optimal membership functions (that is, to tune the DB).

George et al. [GSR94] propose a method based on GAs and Neural Networks (NN) for defining FLC DBs by learning the membership functions associated to the primary fuzzy partitions provided by the expert. The combination of the two search process allows to combine the best characteristics of each one of them in order to solve the problem. Other papers combining both GA and NN are [IFSA95, SFH95].

Hoffman and Pfister present two different genetic based methods for designing hierarchical FLCs by learning the RB [HP94, HP95]. While the first of them uses a Simple GA with fixed-length binary coding and usual genetic operators [HP94], the second one employs a Messy GA with variable-length binary coding and the crossover operator replaced by two simple ones: splice and cut.

Several approaches based on fuzzy classifier systems (FCSs) have been developed [BK94, CF94, PB93, VR91b, VR91a]. The FCS is a genetic based machine learning system which integrates a fuzzy RB, the bucket-brigade learning algorithm (the learning algorithm commonlly used by the classifier systems) and a GA [GS92, VR91b, VR91a]. Each classifier represents a fuzzy rule and the FCS employs a GA to evolve adequate rules running over the population of classifiers searching new improved ones.

Finally, to point out that although the application of GA for designing fuzzy systems is recent, it has an increasing concern over the last years that will allow to obtain fruitful researches in the building of fuzzy logic based intelligent systems.

References

[BK94] Bäck T. and Kursawe F. (July 1994) Evolutionary algorithms for fuzzy logic: A brief overview. In *Proc. Fifth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU'94)*, pages 659–664. Paris.

[BMU95] Braunstingl R., Mujika J., and Uribe J. P. (March 1995) A wall following robot with a fuzzy logic controller optimized by a genetic algorithm. In *Proc. Fourth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'95)*, pages 77–82. Yokohama.

[BN95] Bolata F. and Nowé A. (March 1995) From fuzzy linguistic specifications to fuzzy controllers using evolution strategies. In *Proc. Fourth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'95)*, pages 1089–1094. Yokohama.

[Bon93] Bonarini A. (September 1993) Elf: Learning incomplete fuzzy rule sets for an autonomous robot. In *Proc. First European Congress on Fuzzy and Intelligent Technologies (EUFIT'93)*, pages 69–75. Aachen.

[Cas95] Castro J. L. (March 1995) Fuzzy logic controllers are universal approximators. To appear in IEEE Transactions on Systems, Man and Cybernetics.

[CCC$^+$94] Cárdenas E., Castillo J. C., Cordón O., Herrera F., and Peregrín A. (January 1994) Influence of fuzzy implication functions and defuzzification methods in fuzzy control. *BUSEFAL* 57: 69–79.

[CCC$^+$95] Cárdenas E., Castillo J. C., Cordón O., Herrera F., and Peregrín A. (January 1995) Applicability of t-norms in fuzzy control. *BUSEFAL* 61: 28–37.

[CF94] Carse B. and Fogarty T. C. (1994) A fuzzy classifier system using the pittsburgh approach. In Davidor Y., Schwefel H. P., and Mäanner R. (eds) *Parallel Problem Solving from Nature - PPSN III*, pages 260–269. Springer-Verlag, Berlin.

[CHP95a] Cordón O., Herrera F., and Peregrín A. (March 1995) Applicability of the fuzzy operators in the design of fuzzy logic controllers. Technical Report DECSAI-95111, University of Granada, Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain.

[CHP95b] Cordón O., Herrera F., and Peregrín A. (July 1995) T-norms versus implication functions as implication operators in fuzzy control. To appear in Proc. Sixth International Fuzzy Systems Association World Congress (IFSA'95).

[CV93] Cooper M. G. and Vidal J. J. (1993) Genetic design of fuzzy logic controllers. In *Proc. Second International Conference on Fuzzy Theory and Technology (FTT'93)*. Durham.

[DHR93] Driankov D., Hellendoorn H., and Reinfrank M. (1993) *An Introduction to Fuzzy Control*. Springer-Verlag.

[FTH94] Fathi-Torbaghan M. and Hildebrand L. (July 1994) Evolutionary strategies for the optimization of fuzzy rules. In *Proc. Fifth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU'94)*, pages 671–674. Paris.

[GS92] Geyer-Schulz A. (1992) Fuzzy classifier systems. In Lowen R. (ed) *Fuzzy Logic: State of the Art.* Kluwer Academic Publishers, Dordretch.

[GSR94] George S. M., Saxena A., and Rambabu P. (September 1994) Genetic algorithm in the aid of fuzzy rule deduction. In *Proc. Second European Conference on Soft Computing and Intelligent Technologies (EUFIT'94)*, pages 1130–1133. Aachen.

[HLV94] Herrera F., Lozano M., and Verdegay J. L. (July 1994) Generating fuzzy rules from examples using genetic algorithms. In *Proc. Fifth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU'94)*, pages 675–679. Paris.

[HLV95a] Herrera F., Lozano M., and Verdegay J. L. (February 1995) A learning process for fuzzy control rules using genetic algorithms. Technical Report DECSAI-95108, University of Granada, Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain.

[HLV95b] Herrera F., Lozano M., and Verdegay J. L. (1995) Tuning fuzzy logic controllers by genetic algorithms. *International Journal of Approximate Reasoning* 12: 293–315.

[HMB93] Harris C. J., Moore C. G., and Brown M. (1993) *Intelligent Control. Aspects of Fuzzy Logic and Neural Nets.* World Scientific Publishing.

[HP94] Hoffmann F. and Pfister G. (September 1994) Automatic design of hierarchical fuzzy controllers using genetic algorithms. In *Proc. Second European Conference on Soft Computing and Intelligent Technologies (EUFIT'94)*, pages 1516–1522. Aachen.

[HP95] Hoffmann F. and Pfister G. (July 1995) A new learning method for the design of hierarchical fuzzy controllers using messy genetic algorithms. To appear in Proc. Sixth International Fuzzy Systems Association World Congress (IFSA'95).

[HT94] Hwang W. R. and Thompson W. E. (June 1994) Design of fuzzy logic controllers using genetic algorithms. In *Proc. Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*, pages 1383–1388. Orlando.

[HTS93] Hu H.-T., Tai H.-M., and Shenoi S. (1993) Fuzzy controller design using cell mappings and genetic algorithms. In *Proc. Second International Conference on Fuzzy Theory and Technology (FTT'93).* Durham.

[IFSA95] Ishigami H., Fukuda T., Shibata T., and Arai F. (May 1995) Structure optimization of fuzzy neural networks by genetic algorithm. *Fuzzy Sets and Systems* 71(3): 257–264.

[Kar91a] Karr C. (March 1991) Applying genetics. *AI Expert* pages 38–43.

[Kar91b] Karr C. (February 1991) Genetic algorithms for fuzzy controllers. *AI Expert* pages 26–33.

[KB93] Kropp K. and Baitinger U. G. (September 1993) Optimization of fuzzy logic controller inference rules using a genetic algorithm. In *Proc. First European Congress on Fuzzy and Intelligent Technologies (EUFIT'93)*, pages 1090–1096. Aachen.

[KKS85] Kiszka J. B., Kochanska M. E., and Sliwomska D. S. (January 1985) The influence of some fuzzy implication operators on the accuracy of a fuzzy model - parts i and ii. *Fuzzy Sets and Systems* 15: 111–128, 223–240.

[Lee90] Lee C. C. (March 1990) Fuzzy logic in control systems: Fuzzy logic controller - parts i and ii. *IEEE Transactions on Systems, Man and Cybernetics* 20(2): 404–435.

[LM94] Liska J. and Melsheimer S. S. (June 1994) Complete design of fuzzy logic systems using genetic algorithms. In *Proc. Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*, pages 1377–1382. Orlando.

[LP94] Leitch D. and Probert P. (1994) Context depending coding in genetic algorithms for the design of fuzzy systems. In *Proc. IEEE/Nagoya University WWW on Fuzzy Logic and Neural Networks/Genetic Algorithms.* Nagoya.

[LT93] Lee M. A. and Takagi H. (July 1993) Embedding apriori knowledge into an integrated fuzzy system design method based on genetic algorithms. In *Proc. Fifth International Fuzzy Systems Association World Congress (IFSA'93)*, pages 1293–1296. Seoul.

[LW91] Liaw C. M. and Wang J. B. (July 1991) Design and implementation of a fuzzy controller for a high performance induction motor drive. *IEEE Transactions on Systems, Man and Cybernetics* 21(4): 921–929.

[MA75] Mamdani E. H. and Assilian S. (1975) An experiment in linguistic systhesis with a fuzzy logic controller. *International Journal of Man-Machine Studies* 7: 1–13.

[NHW92] Nomura H., Hayashi I., and Wakami N. (1992) A learning method of simplified fuzzy reasoning by genetic algorithm. In *Proc. International Fuzzy Systems and Intelligent Control Conference (FSIC'92)*, pages 236–245. Louisville.

[NL94] Ng K. C. and Lee Y. (June 1994) Design of sophisticated fuzzy logic controllers using genetic algorithms. In *Proc. Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*, pages 1708–1712. Orlando.

[PB93] Parodi A. and Bonelli P. (July 1993) A new approach to fuzzy classifier system. In *Proc. Fifth International Conference on Genetic Algorithms (ICGA'93)*, pages 223–230.

[SFH95] Shijojima K., Fukuda T., and Hasegawa Y. (May 1995) Self-tuning fuzzy modeling with adaptive memebership function, rules, and hierachical structure based on genetic algorithm. *Fuzzy Sets and Systems* 71(3): 295–309.

[SK94] Satyadas A. and Krishnakumar K. (June 1994) Ga-optimized fuzzy controller for spacecraft attitude control. In *Proc. Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'94)*, pages 1979–1984. Orlando.

[SKG93] Surmann H., Kanstein A., and Goser K. (September 1993) Self-organizing and genetic algorithms for an automatic design of fuzzy control and decision systems. In *Proc. First European Congress on Fuzzy and Intelligent Technologies (EUFIT'93)*, pages 1097–1104. Aachen.

[Thr91] Thrift P. (1991) Fuzzy logic synthesis with genetic algorithms. In *Proc. Fourth International Conference on Genetic Algorithms (ICGA'91)*, pages 509–513.

[TS85] Takagi T. and Sugeno M. (January 1985) Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on Systems, Man and Cybernetics* 15(1): 116–132.

[VM95] Velasco J. R. and Magdalena L. (July 1995) Genetic learning applied to fuzzy rules and fuzzy knowledge bases. To appear in Proc. Sixth International Fuzzy Systems Association World Congress (IFSA'95).

[VR91a] Valenzuela-Rendón M. (1991) The fuzzy classifier system: A classifier system for continuously varing variables. In *Proc. Fourth International Conference on Genetic Algorithms (ICGA'91)*, pages 346–353.

[VR91b] Valenzuela-Rendón M. (1991) The fuzzy classifier system: Motivations and first results. In Männer R. and Manderick B. (eds) *Parallel Problem Solving from Nature - PPSN II*, pages 330–334. Springer-Verlag, Berlin.

[Zad65] Zadeh L. A. (1965) Fuzzy sets. *Information and Control* 8: 358–353.

[Zad73] Zadeh L. A. (1973) Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man and Cybernetics* 3: 28–44.