Genetic Algorithms: Optimization, Search and Learning
1 Introduction ................................................................. 3
2 Genetic Algorithms .......................................................... 4
  2.1 Applications of GAs ....................................................... 5
3 Learning with GAs ........................................................... 6
  3.1 Some Remarks ............................................................ 7
4 Concluding Remarks ....................................................... 8

Integrating Genetic Algorithms and Fuzzy Logic
1 Introduction ................................................................. 9
2 Classification Areas ......................................................... 11
3 Concluding Remarks ....................................................... 14

Genetic Fuzzy Rule Based Systems
1 Introduction ................................................................. 15
2 Genetic Fuzzy Rule Based Systems ........................................ 16
  2.1 Obtaining the Knowledge for an FRBS ................................ 16
  2.2 The Keys to the Tuning/Learning Process .......................... 17
  2.3 The Cooperation vs. Competition Problem ......................... 19
3 Concluding Remarks ....................................................... 20

Genetic Tuning of Fuzzy Rule Based Systems: Basic Models
1 Introduction ................................................................. 21
2 Adapting the Context ....................................................... 22
3 Tuning the Membership Functions ....................................... 22
  3.1 Shape of the Membership Functions ................................. 24
  3.2 Scope of the Semantics ................................................. 24
  3.3 The Approximate Genetic Tuning Process ......................... 25
  3.4 The Descriptive Genetic Tuning Process ......................... 26
4 Concluding Remarks ....................................................... 26

Learning with Genetic Fuzzy Systems: An Application
1 Introduction ................................................................. 27
2 The Fuzzy System .......................................................... 28
3 The Optimization of the Classification System ....................... 29
4 Concluding Remarks ....................................................... 33

Learning with Genetic Fuzzy Systems: Pittsburgh Approach
1 Introduction ................................................................. 34
2 Genetic Learning of RB ..................................................... 35
  2.1 Using a Complete RB .................................................... 35
  2.2 Using a Partial RB ....................................................... 35
Genetic Algorithms: Optimization, Search and Learning

Francisco Herrera

Dept. of Computer Science and A.I., E.T.S. Ingeniería Informática
University of Granada, E-18071 Granada, Spain
E-mail: herrera@decsai.ugr.es

Abstract

Genetic algorithms play a significant role, as search techniques for handling complex spaces, in many fields such as artificial intelligence, engineering, robotic, etc. Genetic algorithms are based on the underlying genetic process in biological organisms and on the natural evolution principles of populations. A short description is given in this lecture, introducing their use for machine learning.

Key words: Genetic Algorithms, Evolutionary Computation, Learning.

1 Introduction

Evolutionary Computation (EC) uses computational models of evolutionary processes as key elements in the design and implementation of computer-based problem solving systems. There are a variety of evolutionary computational models that have been proposed and studied which are referred as Evolutionary Algorithms (EAs). Shortly, this paradigm covers several variations, such as Evolutionary Strategies, addressing continuous function optimization [72], Evolutionary Programming, generating finite state automata that describe strategies or behaviors [26], Genetic Algorithms, providing continuous and discrete function optimization and search [32,46] and Genetic Programming, evolving computer programs to approximately solve problems [52].

In this lecture we will give a short introduction to the most widely studied EA, Genetic Algorithms, and the use of them for Machine Learning.
2 Genetic Algorithms

Genetic algorithms (GAs) have had a great measure of success in search and optimization problems. The reason for a great part of their success is their ability to exploit the information accumulated about an initially unknown search space in order to bias subsequent searches into useful subspaces, i.e., their adaptation. This is their key feature, particularly in large, complex, and poorly understood search spaces, where classical search tools (enumerative, heuristic,...) are inappropriate, offering a valid approach to problems requiring efficient and effective search techniques.

GAs are general purpose search algorithms which use principles inspired by natural genetic populations to evolve solutions to problems [46,32]. The basic idea is to maintain a population of chromosomes, which represent candidate solutions to the concrete problem, that evolves over time through a process of competition and controlled variation. Each chromosome in the population has an associated fitness to determine which chromosomes are used to form new ones in the competition process, which is called selection. The new ones are created using genetic operators such as crossover and mutation.

A GA starts off with a population of randomly generated chromosomes, and advances toward better chromosomes by applying genetic operators modeled on the genetic processes occurring in nature. The population undergoes evolution in a form of natural selection. During successive iterations, called generations, chromosomes in the population are rated for their adaptation as solutions, and on the basis of these evaluations, a new population of chromosomes is formed using a selection mechanism and specific genetic operators such as crossover and mutation. An evaluation or fitness function ($f$) must be devised for each problem to be solved. Given a particular chromosome, a possible solution, the fitness function returns a single numerical fitness, which is supposed to be proportional to the utility or adaptation of the solution represented by that chromosome.

Although there are many possible variants of the basic GA, the fundamental underlying mechanism consists of three operations:

(1) evaluation of individual fitness,
(2) formation of a gene pool (intermediate population) through selection mechanism, and
(3) recombination through crossover and mutation operators.

Next procedure shows the structure of a basic GA, where $P(t)$ denotes the population at generation $t$. 

4
Procedure Genetic Algorithm

begin (1)
  \( t = 0; \)
  initialize \( P(t); \)
  evaluate \( P(t); \)
  \textbf{While (Not termination-condition) do}
  \begin{align*}
  & t = t + 1; \\
  & \text{select } P(t) \text{ from } P(t - 1); \\
  & \text{recombine } P(t); \\
  & \text{evaluate } P(t); \\
  \end{align*}
end (2)
end (1)

The basic principles of GAs were first laid down rigorously by Holland ([46]), and are well described in many books, such as [32,60]. It is generally accepted that the application of a GA to solve a problem must take into account the following five components:

1. A genetic representation of solutions to the problem,
2. a way to create an initial population of solutions,
3. an evaluation function which gives the fitness of each chromosome,
4. genetic operators that alter the genetic composition of offspring during reproduction, and
5. values for the parameters that the GA uses (population size, probabilities of applying genetic operators, etc.).

2.1 Applications of GAs

GAs may deal successfully with a wide range of problem areas. The main reasons for this success are: 1) GAs can solve hard problems quickly and reliably, 2) GAs are easy to interface to existing simulations and models, 3) GAs are extendible and 4) GAs are easy to hybridize. All these reasons may be summed up in only one: GAs are robust. GAs are more powerful in difficult environments where the space is usually large, discontinuous, complex and poorly understood. They are not guaranteed to find the global optimum solution to a problem, but they are generally good at finding acceptably good solutions to problems acceptably quickly. These reasons have been behind the fact that, during the last few years, GA applications have grown enormously in many fields.

The following references show monograph books of applications in different areas: engineering and computer science [22,84], machine learning [38,29], pattern recognition [63], neuronal networks [83], robotic [21], investment strategies [5], management applications [6], and fuzzy systems [43,67,70].
3 Learning with GAs

Although GAs are not learning algorithms, they may offer a powerful and domain-independent search method for a variety of learning tasks. In fact, there has been a good deal of interest in using GAs for machine learning problems ([23,38,31]).

Three alternative approaches, in which GAs have been applied to learning processes, have been proposed, the Michigan ([48]), the Pittsburgh ([74]), and the Iterative Rule Learning (IRL) approaches [33,82]. In the first one, the chromosomes correspond to classifier rules which are evolved as a whole, whereas in the Pittsburgh approach, each chromosome encodes a complete set of classifiers. In the IRL approach each chromosome represents only one rule learning, but contrary to the first, only the best individual is considered as the solution, discarding the remaining chromosomes in the population. Below, we will describe them briefly.

**Michigan approach.** The chromosomes are individual rules and a rule set is represented by the entire population. The collection of rules are modified over time via interaction with the environment. This model maintains the population of classifiers with credit assignment, rule discovery and genetic operations applied at the level of the individual rule.

A genetic learning process based on the Michigan approach receives the name of Classifier System. A complete description is to be found in [12].

**Pittsburgh approach.** Each chromosome encodes a whole rule sets. Crossover serves to provide a new combination of rules and mutation provides new rules. In some cases, variable-length rule bases are used, employing modified genetic operators for dealing with these variable-length and position independent genomes.

This model was initially proposed by Smith in 1980 [74]. Recent instances of this approach may be found in [38].

**Iterative Rule Learning approach.** In this latter model, as in the Michigan one, each chromosome in the population represents a single rule, but contrary to the Michigan one, only the best individual is considered to form part of the solution, discarding the remaining chromosomes in the population. Therefore, in the iterative model, the GA provides a partial solution to the problem of learning. In order to obtain a set of rules, which will be a true solution to the problem, the GA has to be placed within an iterative scheme similar to the following:

1. Use a GA to obtain a rule for the system.
2. Incorporate the rule into the final set of rules.
3. Penalize this rule.
4. If the set of rules obtained till now is adequate to be a solution to the problem, the system ends up returning the set of rules as the solution. Otherwise return...
The main difference with respect to the Michigan approach is that the fitness of each chromosome is computed individually, without taking into account cooperation with other ones. This substantially reduces the search space, because in each sequence of iterations only one rule is searched.

A more detailed description of this approach may be found in [33].

3.1 Some Remarks

The Michigan approach will prove to be the most useful in an on-line process. It is more flexible to handle incremental-mode learning (training instances arrive over time) and dynamically changing domains, whereas the Pittsburgh and the IRL approaches seem to be better suited to batch-mode learning, where all training instances are available before learning is initiated, and for static domains.

The major problem in the Michigan approach is that of resolving the conflict between the individual and collective interests of classifiers within the system. The ultimate aim of a learning classifier system is to evolve a set of co-adapted rules which act together in solving some problems. In a Michigan style system, with selection and replacement at the level of the individual rule, rules which cooperate to effect good actions and receive payoff also compete with each other under the action of the GA.

This conflict between individual and collective interests of individual classifiers does not arise with Pittsburgh-style classifier systems, since reproductive competition occurs between complete rule sets rather than individual rules. However, maintenance and evaluation of a population of complete rule-sets in Pittsburgh-style systems can often lead to a much greater computational burden (in terms of both memory and processing time). Therefore, problems with the Pittsburgh approach have proven to be, at least, equally as challenging. Although the approach avoids the problem of explicit competition between classifiers, large amounts of computing resources are required to evaluate a complete population of rule-sets.

When compared with the latter, the advantage of the IRL approach is that, in the first stage space it considerably reduces the search because it looks for only one rule in each sequence of iterations, although this approach also implies a great computational burden.
4 Concluding Remarks

A short introduction of GAs have been presented. Regarding to their use for machine learning, to point out that GAs are also used for refining parameters in other learning approaches, as is done using GAs for determining weights in a neural network.
Integrating Genetic Algorithms and Fuzzy Logic

Francisco Herrera

Dept. of Computer Science and A.I., E.T.S. Ingeniería Informática
University of Granada, E-18071 Granada, Spain
E-mail: herrera@decsai.ugr.es

Abstract

In this lecture, an evaluation of the current situation regarding to the combination of Genetic Algorithms and Fuzzy Logic is given. This is made by means of a classification in areas, giving a short introduction to each one of them.

Key words: Fuzzy Logic, Genetic Algorithms.

1 Introduction

Recently, numerous papers and applications combining Fuzzy Logic (FL) and Genetic Algorithms (GAs) have become known, and there is an increasing interest in the integration of these two topics.

In the following we explore this combination from the bidirectional integration:

- the use of FL based techniques for either improving GA behaviour and modeling GA components, the results obtained have been called fuzzy genetic algorithms (FGAs), and
- the application of GAs in various optimization and search problems involving fuzzy systems.

The present lecture tries to give a short review of the combination of FL and GAs, introducing a classification of the publications in fourteen areas, presenting briefly them.

Before to introduce the aforementioned areas, a few remarks seem to be necessary.
• The first is regarding to the bibliography. It is collected in our technical report O. Cordón, F. Herrera, M. Lozano, "A Classified Review on the Combination Fuzzy Logic-Genetic Algorithms Bibliography", Dept. of Computer Science and A.I., University of Granada, Tech.Report 95129, October 1995 (Last version December 1996). Available at the URL address: http://decsai.ugr.es/~herrera/fl-ga.html. It classifies and lists 562 references. This report classifies the bibliography in 15 sections according to the following table. It contains the keywords and the number of papers on each of them. These keywords covers the application of FL based tools to GAs (with the name of fuzzy genetic algorithms) and the different areas of FL and fuzzy set theory where GAs have been applied. The underlying report is continuously being updated.

<table>
<thead>
<tr>
<th></th>
<th>Classification keywords</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fuzzy genetic algorithms</td>
<td>24</td>
</tr>
<tr>
<td>2</td>
<td>Fuzzy clustering</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>Fuzzy optimization</td>
<td>39</td>
</tr>
<tr>
<td>4</td>
<td>Fuzzy neural networks</td>
<td>34</td>
</tr>
<tr>
<td>5</td>
<td>Fuzzy relational equations</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>Fuzzy expert systems</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>Fuzzy classifier systems</td>
<td>33</td>
</tr>
<tr>
<td>8</td>
<td>Fuzzy information retrieval and database querying</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td>Fuzzy decision making, financial, and economic models</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>Fuzzy regression analysis</td>
<td>6</td>
</tr>
<tr>
<td>11</td>
<td>Fuzzy pattern recognition and image processing</td>
<td>24</td>
</tr>
<tr>
<td>12</td>
<td>Fuzzy classification - Concept Learning</td>
<td>24</td>
</tr>
<tr>
<td>13</td>
<td>Fuzzy logic controllers (Design, Learning, Tuning, Applications)</td>
<td>287</td>
</tr>
<tr>
<td>14</td>
<td>Fuzzy logic - Genetic algorithms framework</td>
<td>13</td>
</tr>
<tr>
<td>15</td>
<td>Fuzzy logic miscellaneous</td>
<td>38</td>
</tr>
</tbody>
</table>

Table 1. Classification keywords


We consider fourteen areas, we join the table areas 7 and 13 in a global area with the name Genetic fuzzy rule-based control systems. In the following we describe the classification areas. The exhaustive bibliography is found in the aforementioned reference.
2 Classification Areas

In this section we introduce a description of every area and describe shortly the application of GAs to them.

**Fuzzy genetic algorithms.** A Fuzzy Genetic Algorithm (FGA) is considered as a GA that uses fuzzy logic based techniques or fuzzy tools to improve the GA behaviour modeling different GA components.

An FGA may be defined as an ordering sequence of instructions in which some of the instructions or algorithm components may be designed with fuzzy logic based tools, such as, fuzzy operators and fuzzy connectives for designing genetic operators with different properties, fuzzy logic control systems for controlling the GA parameters according to some performance measures, fuzzy stop criteria, representation tasks, etc.

**Fuzzy clustering.** Clustering plays a key role in searching for structures in data. Given a finite set of data, $X$, the problem of clustering in $X$ is to find several cluster centers that can properly characterize relevant classes of $X$. In classical cluster analysis, these classes are required to form a partition of $X$ such that the degree of association is strong for data within blocks of the partition and weak for data in different blocks. However, this requirement is too strong in many practical applications, and it is thus desirable to replace it with a weaker requirement. When the requirement of a crisp partition of $X$ is replaced with a weaker requirement of a *fuzzy partition* or a *fuzzy pseudopartition* on $X$, the emerged problem area is referred as *fuzzy clustering*.

GAs are used for a global search of the space of possible data partitions given a choice of the number of clusters or classes in the data, for determining the number of clusters, etc.

**Fuzzy optimization.** Fuzzy optimization deals with how to find a best point under some fuzzy goals and restrictions given as linguistic terms or fuzzy sets.

GAs are used for solving different fuzzy optimization problems. This is the case for instance of fuzzy flowshop scheduling problems, vehicle routing problems with fuzzy due-time, fuzzy mixed integer programming applied to resource distribution, interactive fuzzy satisfying method for multiobjective 0-1, fuzzy optimal reliability design problems, job-shop scheduling problem with fuzzy processing time, fuzzy optimization of distribution networks, etc.

**Fuzzy neural networks.** Neural networks have been recognized as an important tool for constructing membership functions, operations on membership functions, fuzzy inference rules, and other context-dependent entities in fuzzy set theory.
On other hand, attempts have been made to develop alternative neural networks, more attuned to the various procedures of approximate reasoning. These alternative neural networks are usually referred to as fuzzy neural networks. The following features, or some of them, distinguish fuzzy neural networks from their classical counterparts: inputs are fuzzy numbers, outputs are fuzzy numbers, weights are fuzzy numbers, weighted inputs of each neuron are not aggregated by summation, but by some other aggregation operation. A deviation from classical neural networks in any of these features requires a properly modified learning algorithm to be developed.

GAs are used for designing an overall good architecture of fuzzy neural networks and fuzzy neural networks, for determining an optimal set of link weight, for participating in hybrid learning algorithms, etc.

**Fuzzy relational equations.** The notion of fuzzy relational equations is associated with the concept of composition of binary relations. This operation involves exactly the same combinations of matrix entries as in the regular matrix multiplication. However, the multiplications and additions that are applied to these combinations in the matrix multiplication are replaced with other operations. These alternative operations represent, in each given context, the appropriate operations of fuzzy set intersection and union, respectively. Fuzzy relational equations have been intensively exploited in many areas of applications of fuzzy sets.

GAs may be used either for finding approximate solutions to a system of fuzzy relational equations or for learning in relational structures.

**Fuzzy expert systems.** An expert system is a computer-based system that emulate the reasoning process of a human expert within a specific domain of knowledge. In fuzzy expert systems, the knowledge is usually represented by a set of fuzzy production rules, which connect antecedents with consequent, premises with conclusions, or conditions with actions.

GAs can solve two basical problems of the knowledge base, the knowledge base building and the knowledge filtering.

**Fuzzy information retrieval** Information retrieval may be defined as the problem of the selection of documentary information from storage in response to search questions. The motivation of the application of fuzzy set theory to the design of databases and information storage and retrieval systems lies in the need to handle imprecise information. The database that can accommodate imprecise information can store and manipulate not only precise facts, but also subjective expert opinions, judgments, and values that can be specified in linguistic terms.

GAs are used for designing models for optimization of queries in a fuzzy information retrieval system.
Fuzzy decision making, financial, and economic models. Decision making is the study of how decisions are actually made and how they can be made better or more successfully. Fuzzy set theory has been widely used in the field of decision making. For the most part, the application consisted on fuzzifications of the classical theories of decision making. Also it is used for modelling some financial and economic problems.

GAs are used for cooperating in the design and resolution of these models.

Fuzzy regression analysis Regression analysis is an area of statistics that deals with the investigation of the dependence of a variable upon one or more other variables. Two distinct motivations, fuzzy relation seems intuitively more realistic and the nature of data which in some applications are inherently fuzzy, lead to two types of fuzzy regression analysis. One involves fuzzy parameters and crisp data, while the other one involves crisp parameters and fuzzy data.

GAs are used for solving the underlying optimization problems.

Fuzzy pattern recognition and image processing. There are various aspects of image processing and analysis problems where the theory of fuzzy sets has been applied: as generalizations of classical membership-roster methods, generalizations of classical syntactic methods, providing image ambiguity/information measures and quantitative evaluation, computing fuzzy geometrical properties, etc.

In handling uncertainty in pattern analysis, GAs may be helpful in determining the appropriate membership functions, rules and parameter space, and in providing a reasonably suitable solution. For this purpose, a suitable fuzzy fitness function needs to be defined depending on the problem.

Fuzzy classification - Concept learning.

Fuzzy classification systems based on fuzzy logic are capable of dealing with cognitive uncertainties such as the vagueness and ambiguity involved in classification problems. In a fuzzy classification system, a case or an object can be classified by applying (mainly) a set of fuzzy rules based on the linguistic values of its attributes.

GAs are used in a fuzzy classification system for learning fuzzy rules, membership functions, fuzzy partitions, etc.

Genetic fuzzy rule based control systems. Fuzzy rule based systems 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.

GAs have demonstrated to be a powerful tool for automating the definition of the Knowledge Base of a Fuzzy Controller since adaptive control, learning, and self-organization may be considered in a lot of cases as optimization or search pro-
cesses. Their advantages have extended the use of GAs in the development of a wide range of approaches for designing Fuzzy Controllers over the last few years. In particular, the application to the design, learning and tuning of KBs has produced quite promising results. These approaches can receive the general name of Genetic Fuzzy Systems (GFSs). On other hand, we also must understand the GFSs as the application of GAs to any fuzzy system being the fuzzy rule based systems a particular case although the most extended, this is the reason of calling this area as genetic fuzzy rule based control systems.

3 Concluding Remarks

After the short description of areas, to point out that the use of fuzzy logic techniques permits GA behaviour to be improved in different ways, as well as emphasize the potential of GAs in fuzzy environments as a flexible tool for optimization and search.

Finally, to mention six references. Two of them are technical reports that collect bibliography on the combination of GAs and FL [1,19], the third reference is the paper basis for this summary [20], and the last three references, the three edited books [43,67,70], present a collection of papers dealing with the topic.
Genetic Fuzzy Rule Based Systems

Francisco Herrera

Dept. of Computer Science and A.I., E.T.S. Ingeniería Informática
University of Granada, E-18071 Granada, Spain
E-mail: herrera@decsai.ugr.es

Abstract

The search capabilities and ability for incorporating a priori knowledge have extended the use of Genetic Algorithms in the development of a wide range of methods for designing fuzzy systems over the last few years. Systems applying these design approaches have received the general name of Genetic Fuzzy Systems.

In this lecture we focus our presentation on genetic fuzzy rule-based systems.

Key words: Genetic Algorithms, Fuzzy Rule Based Systems, Learning, Tuning.

1 Introduction

In a very broad sense, a Fuzzy System (FS) is any Fuzzy Logic-Based System, where Fuzzy Logic can be used either as the basis for the representation of different forms of system knowledge, or to model the interactions and relationships among the system variables. FSs have proven to be an important tool for modeling complex systems, in which, due to the complexity or the imprecision, classical tools are unsuccessful ([66,85]).

Recently, a great number of publications explore the use of Genetic Algorithms (GAs) for designing fuzzy systems. These approaches receive the general name of Genetic Fuzzy Systems (GFSs).

The automatic definition of an FS can be considered in many cases as an optimization or search process. GAs are the best known and most widely used global search technique with an ability to explore and exploit a given operating space using available performance measures. GAs are known to be capable of finding near optimal solutions in complex search spaces. A priori knowledge may be in the form of linguistic variables, fuzzy membership function parameters, fuzzy rules, number of rules, etc. The generic code structure and independent performance features of
GAs make them suitable candidates for incorporating a priori knowledge. These advantages have extended the use of GAs in the development of a wide range of approaches for designing fuzzy systems over the last few years.

We shall center this lecture on Fuzzy Rule Based Systems (FRBSs), [2], the most extended FS model to which the most successful application of FSs belong, the fuzzy logic controllers (FLCs), which have been and are used in many real-world control problems ([24]). As is well known, the Knowledge Base (KB) of an FRBS is comprised of two components, a Data Base (DB), containing the definitions of the scaling factors and 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 rules. GAs may be applied to adapting/learning the DB and/or the RB of an FRBS. This tutorial will summarize and analyze the GFSs, paying a special attention to FRBSs incorporating tuning/learning through GAs.

This lecture presents some characteristics of genetic fuzzy rule based systems.

2 Genetic Fuzzy Rule Based Systems

The idea of a Genetic FRBS is that of a genetic FRBS design process which incorporates genetic techniques to achieve the automatic generation or modification of its KB (or a part of it). This generation or modification usually involves a tuning/learning process, and consequently this process plays a central role in GFSs. The objective of this tuning/learning process is optimization, i.e., maximizing or minimizing a certain function representing or describing the behavior of the system.

It is possible to define two different groups of optimization problems in FRBSs. The first group contains those problems where optimization only involves the behavior of the FRBS, while the second one refers to those problems where optimization involves the global behavior of the FRBS and an additional system. The first group contains problems such as modeling, classification, prediction and, in general, identification problems. In this case, the optimization process searches for an FRBS able to reproduce the behavior of a certain target system. The most representative problem in the second group is control, where the objective is to add an FRBS to a controlled system in order to obtain a certain overall behavior. Next, we analyze some aspects of the Genetic FRBSs.

2.1 Obtaining the Knowledge for an FRBS

As a first step, it is interesting to distinguish between tuning and learning problems. In tuning problems, a predefined RB is used and the objective is to find a set of parameters defining the DB. In learning problems, a more elaborate process
including the modification of the RB is performed. We can distinguish between three different groups of GFSs depending on the KB components included in the genetic learning process.

For an extensive bibliography see [19] (section 3.13), some approaches may be found in [33].

**Genetic tuning of the DB.** The tuning of the scaling functions and fuzzy membership functions is an important task in the design of fuzzy systems. It is possible to parameterize the scaling functions or the membership functions and adapt them using GAs to deal with their parameters according to a fitness function. As regards to the tuning of membership functions, several methods have been proposed in order to define the DB using GAs. Each chromosome involved in the evolution process represents different DB definitions, i.e., each chromosome contains a coding of the whole set of membership functions giving meaning to the linguistic terms. Two possibilities can be considered depending on whether the fuzzy model nature is descriptive or approximate, either to code the fuzzy partition maintaining a linguistic description of the system, or to code the rule membership functions tuning the parameters of a label locally for every rule, thereby obtaining a fuzzy approximate model.

**Genetic learning of the RB.** All the methods belonging to this family involve the existence of a predefined collection of fuzzy membership functions giving meaning to the linguistic labels contained in the rules, a DB. On this basis GAs are applied to obtain a suitable rule base, using chromosomes that code single rules or complete rule bases.

**Genetic learning of the KB.** There are many approaches for the genetic learning of a complete KB (RB and DB). We may find approaches presenting variable chromosome lengths, others coding a fixed number of rules and their membership functions, several working with chromosomes encoding single control rules instead of a complete KBs, etc.

### 2.2 The Keys to the Tuning/Learning Process

Regardless of the kind of optimization problem, i.e., given a system to be modeled/controlled (hereafter we use this notation), the involved tuning/learning process will be based on evolution. Three points are the keys to an evolutionary based tuning/learning process. These three points are: the population of potential solutions, the set of evolution operators and the performance index.

**The population of potential solutions.** The learning process works on a population of potential solutions to the problem. In this case, the potential solution is an FRBS. From this point of view, the learning process will work on a population of
FRBSs, but considering that all the systems use an identical processing structure, the individuals in the population will be reduced to DB/RB or KBs. In some cases the process starts off with an initial population obtained from available knowledge, while in other cases the initial population is randomly generated.

**The set of evolution operators.** The second question is the definition of a set of evolution operators that search for new and/or better potential solutions (KBs). The search reveals two different aspects: the exploitation of the best solution and the exploration of the search space. The success of evolutionary learning is specifically related to obtaining an adequate balance between exploration and exploitation, that finally depends on the selected set of evolution operators. The new potential solutions are obtained by applying the evolution operators to the members of the population of knowledge bases, each one of these members is referred to as an individual in the population. The evolution operators, that work with a code (called a chromosome) representing the KB, are basically three: selection, crossover and mutation. Since these evolution operators work in a coded representation of the KBs, a certain compatibility between the operators and the structure of the chromosomes is required. This compatibility is stated in two different ways: work with chromosomes coded as binary strings (adapting the problem solutions to binary code) using a set of classical genetic operators, or adapt the operators to obtain compatible evolution operators using chromosomes with a non-binary code. Consequently, the question of defining a set of evolution operators involves defining a compatible couple of evolution operators and chromosome coding.

**The performance index.** Finally, the third question is that of designing an evaluation system capable of generating an appropriate performance index related to each individual in the population, in such a way that a better solution will obtain a higher performance index. This performance index will drive the optimization process.

In identification problems, the performance index will usually be based on error measures that characterize the difference between the desired output and the actual output of the system. In control problems there are two different sources of information to be used when defining the performance index: information describing the desired behavior of the controlled system, or describing the desired behavior of the controller (FRBS) itself. The second situation is closely related to identification problems. The definition of a performance index is usually more complex for the first situation, where the objective is to find a controller that gives the desired behavior in the controlled system.

**The process.** Summarizing the points that characterize a specific learning process, these are: the initial population of solutions (obtained randomly or from some initial knowledge), the coding scheme for KBs (chromosomes), the set of evolution operators and the evaluation function. The initial population and the evaluation function are related to the specific problem while the coding scheme and the evolu-
tion operators could be generic. In addition to these four points, each evolutionary learning process is characterized by a set of parameters such as the dimension of the population (fixed or variable), the parameters regulating the activity of the operators or even theirs effect, and the parameters or conditions defining the end of the process or the time when a qualitative change in the process occurs.

2.3 The Cooperation vs. Competition Problem

A GFS combines the main aspects of the system to be obtained, an FS, and the design technique used to obtain it, a GA, with the aim of improving as far as possible the accuracy of the final FS generated.

One of the most interesting features of an FS is the interpolative reasoning it develops. This characteristic plays a key role in the high performance of FSs and is a consequence of the cooperation between the fuzzy rules composing the KB. As is known, the output obtained from an FS is not usually due to a single fuzzy rule but to the cooperative action of several fuzzy rules that have been fired because they match the input to the system to some degree.

On the other hand, the main feature of a GA is the competition between members of the population representing possible solutions to the problem being solved. In this case, this characteristic is due to the mechanisms of natural selection on which the GA is based.

Therefore, since a GFS combines both aforementioned features, it works by inducing competition to get the best possible cooperation. This seems to be a very interesting way to solve the problem of designing an FS, because the different members of the population compete with one another to provide a final solution presenting the best cooperation between the fuzzy rules composing it. The problem is to obtain the best possible way to put this way of working into effect. This is referred to as cooperation vs. competition problem (CCP) ([10]). The difficulty of solving the introduced problem depends directly on the genetic learning approach followed by the GFS (Michigan, Pittsburgh or IRL approaches). Below we briefly analyze them.

**Michigan approach.** It is difficult to solve the CCP when working with the Michigan approach. In this case, the evolution is performed at the level of fuzzy rules instead of at the level of KBs and it is not easy to obtain an adequate cooperation between fuzzy rules that are competing with one another. To do this, we need a fitness function able to measure both the goodness of a single fuzzy rule and the quality of its cooperation with the other fuzzy rules in the population to give the best action as output. As mentioned in [10], the design of a fitness function of this kind is not an easy task.

**Pittsburgh approach.** This approach is able to solve adequately the CCP. When
using this approach, the GFS evolves populations of KBs and the fitness function associated to each individual is computed taking into account the real action that the FS encoded into the chromosome should give as output when it receives a concrete input. Thus, each time an individual is evaluated, the cooperation between the fuzzy rules composing the KB is measured, so the GFS is able to evolve adequately the population to obtain the FS presenting the best possible cooperation between the fuzzy rules composing its KB. Unfortunately, this approach presents the drawback of having to deal with very large search spaces, which makes it difficult to find optimal solutions. This drawback is usual when designing GFSs belonging to the third family, i.e., when the generation of the whole KB is considered in the genetic learning process. In this case, a large quantity of KB parameters have to be included in the genetic representation, which therefore becomes larger. This fact will be more pronounced if an approximate fuzzy model is considered, the use of different membership function definitions for each rule makes the number of KB parameters increase, and then the search space becomes more complex, making the problem computationally hard.

**IRL approach.** Finally, GFSs based on the IRL approach try to solve the CCP at the same time reducing the search space by encoding a single fuzzy rule in each chromosome. To put this into effect, these processes follow the usual problem partitioning working way and divide the genetic learning process into, at least, two stages. Therefore, the CCP is solved in two steps acting at two different levels, with the competition between fuzzy rules in the first one, the genetic generation stage, and with the cooperation between these generated fuzzy rules in the second one, the post-processing stage.

3 Concluding Remarks

In this lecture we have introduced the GFSs, presenting the basic keys to the tuning/learning processes and the problem of the cooperation vs. competition in the different learning approaches.
Abstract

The tuning of the membership functions is an important task in the design of a fuzzy system. Genetic Algorithms are used for the optimization of membership functions and the scaling functions. This lecture introduces the use of Genetic Algorithms in the tuning of the fuzzy systems.

Key words: Genetic Algorithms, Fuzzy Rule Based Systems, Tuning.

1 Introduction

The tuning of the scaling functions and fuzzy membership functions is an important task in the design of fuzzy systems. It is possible to parameterize the scaling functions or the membership functions and adapt them using Genetic Algorithms to deal with their parameters according to a fitness function.

As regards to the tuning of membership functions, several methods have been proposed in order to define the Data Base (DB) using GAs. Each chromosome involved in the evolution process represents different DB definitions, i.e., each chromosome contains a coding of the whole set of membership functions giving meaning to the linguistic terms. Two possibilities can be considered depending on whether the fuzzy model nature is descriptive or approximate, either to code the fuzzy partition maintaining a linguistic description of the system, or to code the rule membership functions tuning the parameters of a label locally for every rule, thereby obtaining a fuzzy approximate model.

In this lecture we analyze the use of GAs for the tuning of DBs according to the two mentioned areas, the adaptation of contexts using scaling functions and the tuning of membership functions, we shall present briefly them.