Adaptive Genetic Operators Based on Coevolution with Fuzzy Behaviors

Francisco Herrera and Manuel Lozano

***Abstract*—This paper presents a technique for adapting control parameter settings associated with genetic operators. Its principal features are: 1) the adaptation takes place at the individual level by means of fuzzy logic controllers (FLCs) and 2) the fuzzy rule bases used by the FLCs come from a separate genetic algorithm (GA) that coevolves with the GA that applies the genetic operator to be controlled. The goal is to obtain fuzzy rule bases that produce suitable control parameter values for allowing the genetic operator to show an adequate performance on the particular problem to be solved. The empirical study of an instance of the technique has shown that it adapts the parameter settings according to the particularities of the search space allowing significant performance to be achieved for problems with different difficulties.**

***Index Terms*—Adaptive genetic algorithms, coevolution, fuzzy logic controllers.**

I. Introduction

FINDING robust variation operators or control parameter settings is not a trivial task since their interaction with the performance of an evolutionary algorithm is complex and the optimal choices are problem dependent [3]. For the discussion here, we will focus on genetic algorithms (GAs) and genetic operators. Different genetic operators or control parameter values may be necessary during the course of a run for inducing an optimal exploration/exploitation balance. For these reasons, adaptive GAs (AGAs) have been built that dynamically adjust selected control parameters or genetic operators during the course of evolving a solution [1], [28], [31], [40], [48].

One way for building AGAs involves the application of fuzzy logic controllers (FLCs) [10], [17] for adjusting GA control parameters. Although much literature on this adaptive approach has appeared, there are still important challenges that may be considered for applying it. They include the application of FLCs for adapting parameters that control the operation of the genetic operators, taking into account features associated with the parents, and the task of finding good fuzzy rule bases [28].

This paper proposes a technique for adapting genetic operators based on FLCs called coevolution with fuzzy behaviors (FBs), which deals with the two aforementioned challenges.

1) During each genetic operator application event, particular FLCs specify its current genetic control parameter values depending on particular features of the parents. The fuzzy rule bases for an FLC may be coded by means of vectors with the linguistic values of the fuzzy rule consequent. These vectors will be called FBs.
2) The fuzzy rule bases used by the FLCs come from a separate GA that *coevolves* with the GA that uses the genetic operator to be controlled (coevolution). The goal of coevolution with FBs is to obtain fuzzy rule bases that produce suitable control parameter values for allowing the genetic operator to show an adequate performance.

The term *coevolution* is used for describing the fact that two different types of structures evolve in a parallel way with some type of *cooperation* [9]. In our case, FBs coevolve with the chromosomes representing solutions to the particular problem. They induce control parameter values for a genetic operator applied to these chromosomes and evolve according to the efficacy induced on the genetic operator (i.e., whether it generates offspring that are more fit than the parents or introduces high diversity levels, etc.). We should point out that coevolution has been considered as a promising way for producing adaptation [6], [34], [46], [49], [57].

In order to investigate the effectiveness of the model, we propose an instance for the adaptation of a crossover operator that works under real coding: fuzzy recombination (FR) [59]. In particular, we study the instance by dealing with the following issues:

1) performance improvement, i.e., if its results on a given test suite are better than the ones for GAs using a fixed configuration;
2) adaptation itself, i.e., if it adjusts genetic control parameters according to the particularities of the problem to be solved. The distributions of the FBs generated during the runs will be considered for this point;
3) their relation, i.e., if adaptation is responsible for the performance improvement.

The paper is set up as follows: in Section II, the basic idea of the AGAs based on FLCs presented in the literature is explained and the two aforementioned challenges associated with them are described. In Section III, the adaptive model based on the coevolution with FBs is presented. In Section IV, an instance of the model is built for adapting a crossover operator for real-coded GAs (RCGAs) [59]. In Section V, an empirical study of the instance is made and, finally, some concluding remarks are offered in Section VI.

II. Adaptive GAs Based on Fuzzy Logic Controllers

The interaction of GA control parameter settings and GA performance is generally acknowledged as a complex relationship that is not completely understood. Although there are ways to understand this relationship (for instance, in terms of stochastic

Manuscript received April 6, 1998; revised December 7, 1998 and June 28, 1999. This research has been supported by CICY T TIC96-0778.

The authors are with the Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain (e-mail: herrera@decsai.ugr.es; lozano@decsai.ugr.es).

Publisher Item Identifier S 1089-778X(01)04139-X.

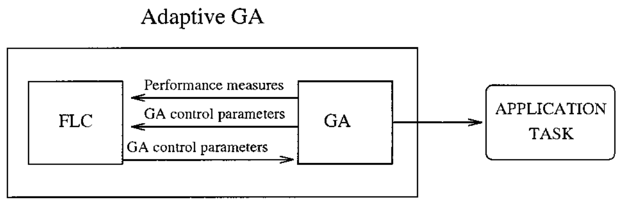


Fig. 1. Structure of the AGA model based on FLCs. GA sends current performance measures and control parameter values to the FLC, which computes the new control parameter values for the GA.

behavior), this kind of understanding does not necessarily result in a normative theory. FLCs are particularly suited to environments that are either ill-defined or are very complex. The adaptation of GA parameters is one such complex problem that may benefit from the use of FLCs [51]. The fuzzy rule bases of FLCs facilitate the capture and representation of a broad range of adaptive strategies for GAs (so, they may be the support for the automatic learning of such strategies). Then, the inference system of FLCs may use this knowledge for carrying out the adaptation of GAs throughout the run.

AGAs based on FLCs are found in [2], [11], [12], [23], [28], [36], [37], [51], [55], [60], [61], [64], and [65]. Their main idea is to use an FLC whose inputs are any combination of GA performance measures or current control parameters and whose outputs are GA control parameters. Current performance measures of the GA are sent to the FLC, which computes new control parameter values that will be used by the GA. Fig. 1 shows this process.

In general, the following steps are needed in order to design an AGA based on FLCs [28].

1) *Defining the inputs and outputs.* Inputs should be robust measures that describe GA behavior and the effects of the genetic setting parameters and genetic operators. In [55], some possible inputs were cited: diversity measures, maximum, average, and minimum fitness, etc. In [36] and [64], it is suggested that current control parameters may also be considered as inputs. Outputs indicate values of control parameters or changes in these parameters [36]. In [55], the following outputs were reported: mutation probability, crossover probability, population size, selective pressure, the time the controller must spend in a target state in order to be considered successful, the degree to which a satisfactory solution has been obtained, etc.
2) *Defining the data base.* Each input and output should have an associated set of linguistic labels. The meaning of these labels is specified through membership functions of fuzzy sets. It is necessary that every input and output have a bounded range of values in order to define these membership functions over it.
3) *Obtaining the fuzzy rule base.* After selecting the inputs and outputs and defining the data base, the fuzzy rules describing the relations between them should be defined. There are different ways to do so:
   a) using the experience and the knowledge of the GA experts;
   b) using an automatic learning technique for those cases where knowledge or expertise are not available. By using an automatic technique, relevant relations and membership functions may be automatically determined and may offer insight for understanding the complex interaction between GA control parameters and GA performance [36].

Next, in Section II-A, we review different approaches of AGAs based on FLCs. Then, in Sections II-B and II-C, we comment on two challenges relating to these AGAs: the adaptation of genetic operators at individual-level based on FLCs and the task of finding good fuzzy rule bases [28].

A. Review on AGAs Based on FLCs

In this section, we survey different AGAs based on FLCs that use fuzzy rule bases built through the expertise, experience, and knowledge on GAs, which have become available as a result of empirical studies conducted over a number of years.

In [64], the use of FLCs to control GAs is considered for solving two problems to which a standard GA may be subjected: very slow search speed and premature convergence. These problems are due to: 1) control parameters not well chosen initially for a given task; 2) parameters always being fixed even though the environment in which the GA operates may be variable; and 3) difficulties resulting from selection of other parameters such as population size and in understanding their influence, both individually and in combination, on the GA performance. FLCs were proposed for controlling GAs in order to: 1) choose control parameters before the GA run; 2) adjust the control parameters online to dynamically adapt to new situations; and 3) assist the user in accessing, designing, implementing, and validating a GA for a given task. An AGA based on FLCs was presented in which the crossover probability and mutation probability were controlled using two FLCs. Both of them had the same inputs: current generation and population size.

In [2], it is claimed that GAs require human supervision during their routine use as practical tools for the following reasons: 1) for detecting the emergence of a solution; 2) for tuning algorithm parameters; and 3) for monitoring the evolution process in order to avoid undesiderable behavior such as premature convergence. It is advised as well that any attempt to develop artificial intelligence tools based on GAs should take these issues into account. The authors proposed FLCs for this task. They called *fuzzy government* the collection of fuzzy rules and routines in charge of controlling the evolution of the GA population. Fuzzy government was applied to the symbolic inference of formulae problem. Genetic programming [35] was used to solve the problem along with different FLCs, which dynamically adjusted the maximum length for genotypes, acted on the mutation probability, detected the emergence of a solution, and stopped the process.

In [28], two FLCs control the use of exploitative and exploratory crossovers and the selective pressure. Two diversity measures are defined for this purpose: genotypical diversity, which measures the (normalized) average distance of the population from the best chromosome, and phenotypical diversity, which measures the ratio between the best fitness and the average fitness. These diversity measures are the inputs to the FLCs. Every five generations, the FLCs evaluate these measures to adjust the probability of using exploitative and exploratory crossovers and the selection pressure (keeping or eliminating diversity in the next generation).

In [60], an AGA based on FLCs is presented for multiobjective optimization problems. In each generation, an FLC decides what transformation of the cost components into an one-dimensional fitness function is taken. In [61], a more complex method called *fuzzy reduction* GA is proposed. It attempts to enable a uniform approximation of the *Pareto optimal* solutions (those that cannot be improved with respect to any cost function without making the value of some other worse). The authors started by explicitly formulating desirable goals for the evolution of the population toward the target Pareto optimal solutions (which could be expressed in vague terms only). Then, they defined deviation measures of a population from these goals, which were the inputs to a FLC. Later, they fixed a set of possible actions that could serve as countermeasures to decrease the deviations. These actions are different selection mechanisms based on classical ones proposed for tackling multiobjective optimization problems. The FLC determines activation rates for the actions. The action that should actually be taken is decided according to the activation rates found.

B. Adaptation at Individual-Level Based on FLCs

In general, there are three levels where the adaptation may take place in an AGA [1], [48].

1) *Population-level* adaptations adjust control parameters that apply for the entire population. An example is found in [15]. An RCGA (a representation found commonly in evolution strategies and evolutionary programming) is proposed, which uses two types of crossover operator and three types of mutation operator. Each operator is given an initial application probability. For each reproduction event, a single operator is selected probabilistically according to the set of operator probabilities. An adaptive process provides for the alteration of operator probabilities in proportion to the fitness of chromosomes created by the operators during the course of a run. Operators that create and cause the generation of better chromosomes are allotted higher probabilities, i.e., they should be used more frequently. On the other hand, operators producing offspring with a fitness which is lower than that of the parents should be used less frequently.
2) *Individual-level* adaptations are centered on the consideration of the individual members of the population rather than the ensemble as a whole. An instance of this type of adaptation is the AGA proposed in [50]. Each chromosome has its own crossover and mutation probabilities. They are varied depending on the convergence state of the population and the fitness value of the chromosome in such a way that high-fitness solutions are protected, while solutions with subaverage fitnesses are totally disrupted. Furthermore, these probabilities are increased when the population tends to get stuck at a local optimum and they are decreased when the population is scattered in the solution space.
3) *Component-level* adaptations dynamically alter how the individual components of each chromosome will be manipulated independently from each other. An approach is the binary-coded AGA presented in [3] and [4], which has the following principal features: a) each position of each chromosome has associated a mutation probability; b) these probabilities are incorporated into the genetic representation of the chromosomes encoded as bitstrings; and c) they are also subject to mutation and selection, i.e., they undergo evolution as well as the chromosomes.

Most AGAs based on FLCs presented in the literature involve population-level adaptation. However, adaptive mechanisms at the individual level based on FLCs may be interesting for adjusting control parameters associated with genetic operators [28], [65]. In this way, the control parameters will be defined on samples instead of on the whole population. Inputs to the FLCs may be central measures and/or measures associated with particular chromosomes or sets of them and outputs may be control parameters associated with genetic operators that are applied to those chromosomes. A justification for this approach is that it allows for the application of different search strategies in different parts of the search space. This is based on the reasonable assumption that, in general, the search space will not be homogeneous and that different strategies will be better suited to different kinds of sublandscapes [48]. For instance, a population member residing currently in a relatively flat region of the search space may be handled more severely than a population member in a more complex portion of the search space [1].

C. Finding Good Fuzzy Rule Bases is Not an Easy Task

The behavior of GAs and the interrelations between the genetic operators are very complex. Although there are many possible inputs and outputs for the FLCs, fuzzy rule bases frequently are not easily available. Finding good fuzzy rule bases is not an easy task. This problem has been recognized by different authors. For example, in [36, p. 78] the following was stated: “Although much literature on the subject of GA control has appeared, our initial attempts at using this information to manually construct a fuzzy system for genetic control were unfruitful.”

In [55, p. 466], a related conclusion was reached: “Statistics and parameters are in part universal to any evolutionary algorithm and in part specific to a particular application. Therefore it is hard to state general fuzzy rules to control the evolutionary process.”

As we have mentioned, automatic learning mechanisms for obtaining fuzzy rule bases may be used for avoiding this problem. In [36] and [37], this kind of automatic technique was proposed for learning fuzzy rule bases along with their data bases. The mechanism is very similar to the meta-GA of Grefenstette [21]. It is based on a high-level GA whose chromosomes code possible fuzzy rule bases together with their corresponding data bases. The fitness function value for a chromosome is calculated using the averaged online performance and averaged offline performance [16] obtained from an AGA based on FLCs that uses the fuzzy rule base coded in such chromosome on the five test functions in [16]. After the high-level GA completed 1000 fitness function evaluations, the best fuzzy rule base reached is returned.

The robustness of the fuzzy rule bases returned by this mechanism depends heavily on the test problem set (which has nothing to do with the particular problem to be solved) and the performance measures used by the high-level GA. Even if the mechanism gets a good fuzzy rule base for some set of problem instances, this might not be the best one for the other instance. Therefore, the fuzzy rule-based definition is still a challenging feature in the fuzzy control of the GAs.

III. Adaptation of Genetic Operators by Coevolution with FBs

In this section, we present a general proposal for a mechanism based on FLCs for the adaptation of genetic operators that use parameters for controlling their operation. Its main features are the following.

1) It incorporates genetic operator adaptation at an individual level based on FLCs. Control parameter values for a genetic operator are computed for each set of parents that undergo it, using an FLC that considers particular features associated with the parents as inputs. In this way, the proposal attempts to deal with the challenge presented in Section II-B.
2) The fuzzy rule bases of the FLCs applied are learnt implicitly throughout the run by means of a separate GA (denoted as FBs-GA) that *coevolves* with the one that applies the genetic operator to be controlled (denoted as main GA). Both GAs have an influence on the other. On the one hand, fuzzy rule bases in FBs-GA induce parameter values for the genetic operator applied to main-GA (FBs-GA $\rightarrow$ main-GA). On the other hand, they evolve according to the performance of the operator on the elements of main-GA (main-GA $\rightarrow$ FBs-GA). The goal of FBs-GA is to obtain the fuzzy rule bases that produce suitable control parameter values for allowing the genetic operator to show an *adequate performance* on the particular problem to be solved (the meaning of this term will be discussed in Section IV-C2). Since the learning technique underlying this approach only takes into account the problem to be solved (in contrast to the approach in [36] and [37], which never considers it), the fuzzy rule bases obtained will specify adaptation strategies particularly appropriate for this problem. With this aim, the proposal attempts to tackle the challenge in Section II-C

FBs-GA does not handle fuzzy rule bases directly. Instead, it uses structures called FBs for representing them, which are more adequate for being treated as chromosomes by a GA. FBs consist of vectors with the linguistic values of the fuzzy rule consequent. They are presented in Section III-A. Then, in Section III-B, we propose the model for the adaptation of genetic operators based on coevolution with FBs, and describe an example for the case of adapting a parameter associated with a pairwise crossover operator. Finally, in Section III-C, we enumerate different ways for applying the technique proposed to genetic operators with control parameters associated.

A. FBs

Let us consider a genetic operator that is applied to sets of n chromosomes (parents), with p being a parameter that controls its operation (either a pairwise crossover operator, a poolwise crossover operator [53], or a mutation operator, etc.). An FLC may be built for the adaptation of p, which receives n features associated with the parents as inputs F_i, $i = 1, \ldots, n$, with L_i being their associated linguistic label sets. The FLC will return a p value for each set of parents that undergoes the genetic operator. The linguistic label set for p is L_p. The fuzzy rule base for the FLC will have a set of fuzzy control rules with the form

$$\text{If } F_1 \text{ is } l_1 \text{ and } F_2 \text{ is } l_2 \text{ and } \cdots \text{ and } F_n \text{ is } l_n \text{ Then } p \text{ is } l_p$$

where $l_i \in L_i$ $i = 1, \ldots, n$ and $l_p \in L_p$.

This fuzzy rule base, constituted by control rules presenting n input variables and a single output variable, may 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 elements of the concrete linguistic label set and the cells of the table will contain the concrete linguistic label that the output variable takes as a value for the combination of antecedents represented by this cell (Table II in Section IV-C1 shows an example). As we have mentioned, all the cells in the complete decision table may be encoded in a single linear vector, called the FB.

B. Coevolution with FBs

In this section, we propose an adaptive mechanism for adjusting the p control parameter based on coevolution with FBs. The idea is to introduce a population of FBs (which represent possible fuzzy rule bases) that *coevolves* with the population of chromosomes (which represent solutions to the particular problem). During the application of the genetic operator to be controlled, a random assignment is established between FBs and sets of parents. Then, the genetic operator is applied to each set using the control parameter value obtained from an FLC that uses the fuzzy rule represented by the corresponding FB.

The population of FBs will undergo evolution through the effects of its own selection process and crossover and mutation operators. The fitness of the FBs will depend on the efficacy induced by them on the genetic operator. Some aspects to be considered may include: whether they generate offspring that are more fit than the parents or introduce high diversity levels, etc. In this way, the adaptive model proposed is based on observing the relative performance of different strategies (represented by the FBs), which appear to be very effective [48].

As an example, Fig. 2 shows a pseudocode algorithm integrating main-GA (which applies a pairwise crossover operator whose operation depends on a p control parameter) and FBs-GA for evolving FBs. $P(t_1)$ denotes the population of chromosomes of main-GA at generation t_1 and $P_{\mathrm{FBs}}(t_2)$ is the population of FBs of FBs-GA at generation t_2. In the following, we explain the algorithm steps briefly.

1) *Main-GA.* Steps 2, 4, 5.2, 5.5, and 5.6 constitute main-GA;
2) *FBs-GA.* Steps 5.8.1 to 5.8.4 form the main loop of FBs-GA. The t_{FBs} parameter (step 5.8) is the gap

```
1. $t_1 = 0$, $t_2 = 0$;
2. Initialize $P(t_1)$;
3. Initialize $P_{FBs}(t_2)$;
4. Evaluate $P(t_1)$;
5. While (not termination-condition ) do
    5.1. $t_1 = t_1 + 1$;
    5.2. Select $P(t_1)$ from $P(t_1 - 1)$;
    5.3. Assign each FB in $P_{FBs}(t_2)$ to $n_{pp}$ pairs of parents in $P(t_1)$, at random;
    5.4. Calculate p for each pair of parents through its FB associated;
    5.5. Perform adaptive crossover (using the $p$ values) and mutation on $P(t_1)$;
    5.6. Evaluate $P(t_1)$;
    5.7. Collect performance measures about the behavior of FBs in $P_{FBs}(t_2)$ on $P(t_1)$.
    5.8. If ($t_1$ is multiple of $t_{FBs}$) then
        5.8.1. Evaluate $P_{FBs}(t_2)$ (using the information collected in 5.7);
        5.8.2. $t_2 = t_2 + 1$;
        5.8.3. Select $P_{FBs}(t_2)$ from $P_{FBs}(t_2 - 1)$;
        5.8.4. Perform crossover and mutation on $P_{FBs}(t_2)$;
```

Fig. 2. Pseudocode algorithm integrating main GA and FBs-GA.

(number of generations performed by main-GA) between two consecutive applications of this main loop of FBs-GA. This parameter determines the synchronization between the run of main-GA and the one of FBs-GA.

3) *Cooperation FBs-GA $\rightarrow$ main-GA.* In steps 5.3 and 5.4, the cooperation from FBs-GA to main-GA is carried out. Throughout these steps, FBs in $P_{\text{FBs}}(t_2)$ induce p values for the operation of the adaptive crossover operator in main-GA. The n_{pp} parameter used in step 5.3 represents the number of pairs of parents to which an FB is assigned.

4) *Cooperation main-GA $\rightarrow$ FBs-GA.* In step 5.7, the cooperation is from main-GA to FBs-GA. If $t_{\text{FBs}} \geq 1$, the same FBs are assigned to pairs of parents that belong to t_{FBs} successive populations of main-GA. Their fitness should be calculated in terms of their average performance over such generations (step 5.8.1), i.e., taking into account their *historical behavior*. In order to do this, step 5.7 collects performance measures describing the behavior of the FBs in $P_{\text{FBs}}(t_2)$ over the t_{FBs} generations of the main-GA in which they are used for generating p values. Later, in step 5.8.1, these measures are used for obtaining an average performance measure for each FB, which will be its fitness.

Finally, we consider some important aspects related with n_{pp} and t_{FBs}.

1) Ratio Between the Population Sizes of Main-GA and FBs-GA: n_{pp} determines the ratio between the population sizes of the two GAs. In particular, there will be as many FBs as pairs of parents that should undergo the crossover operator to be adapted (i.e., $(|P(t)| \cdot p_c)/2$, where p_c is the crossover probability) divided by n_{pp}.

Under a fixed population size for main-GA, the higher n_{pp} is, the lower the population size of FBs-GA will result. If the population size of FBs-GA is too small, it may converge too quickly due to a lack of diversity. On the other hand, if n_{pp} is high, FBs may be evaluated using information about their performance on many different environments (i.e., combinations of values of the features of the parents). In this way, FBs may capture knowledge for inducing adequate p values for parents showing very different feature values. However, this advantage may be achieved as well for low n_{pp} values since the crossover of FBs being found adequate for few combinations of parent features will produce FBs that are adequate for many combinations.

2) Historical knowledge: There are two important factors determining the way in which the use of the historical knowledge about the behavior of FBs $(t_{\text{FBs}} \geq 1)$ may influence positively on the adaptation of the genetic operator: the topology of the particular problem to be solved and the speed of main-GA for visiting different regions of the search space. For example, a high t_{FBs} value may favor a misleading synchronization between the two GAs when dealing with problems showing irregular landscapes, since FBs-GA may select FBs considering their historical effects on a particular region, which shall not be exploited by main-GA in the next generations. This situation probably does not occur on problems with regular landscapes, where the regions considered by the main-GA before and after the application of FBs-GA are likely to show similar features.

With low t_{FBs} values, the synchronization between main-GA and FBs-GA will always be established in a more direct way, since FBs-GA learns FBs, taking into account exclusively their performance on the most recent visited regions. Moreover, a low value for t_{FBs} does not imply sacrificing the existence of FBs with a suitable past in the population of FBs-GA. This GA will keep FBs with a suitable past whenever they have a suitable present (this is due to its own characteristics as GA). In this way, the historical knowledge about FBs is handled adaptively: it is used while useful.

To sum up, all the above considerations seem to recommend the use of low n_{pp} and t_{FBs} values. The effects that these variables have upon the adaptive proposal are investigated empirically in Section V-F.

C. Applications

Different types of parameter settings were associated with genetic operators, which may be adapted by means of coevolution with FBs. They include the following.

1) *Operator probabilities*. There is a type of GAs that do not apply both crossover and mutation to the selected solutions, as in the traditional ones. Instead, a set of operators is available, each with a probability of being used, and one is selected to produce offspring. Many AGAs have been designed starting from this GA approach, which adjust the operator probabilities throughout the run (see [15] and [57]).
2) *Operator parameters*. These parameters determine the way in which genetic operators work. Examples include: a) the step size of mutation operators for RCGAs, which determines the strength in which real genes are mutated [5], [29]; b) parameters associated with crossover operators for RCGAs, such as FR operator [59], BLX-$\alpha-\beta-\gamma$ operator [19], and dynamic FCB-crossover operators [26], [27]; and c) parameters associated with crossover operators for binary-coded GAs, such as n-point crossover [18] and uniform crossover [52].

   The adaptation at individual-level of operator probabilities and operator parameters by coevolution with FBs may be carried out by considering these variables as consequents of the fuzzy rules represented in the FBs. Furthermore, the appropriate features of the parents should be chosen in the basis of which the adjustment of these variable is expressed. On the other hand, hybrid models may be built in such a way that FBs include information for both the adaptation of operator probabilities and operator parameters. In this case, the model will detect the operators that should be applied more frequently along with favorable operator parameter values for them.
3) *Mate selection parameters*. In mate selection mechanisms [43], chromosomes carry out the choice of mate for crossover on the basis of their own preferences (which are formulated in terms of different chromosome characteristics, such as the phenotypical distance between individuals).

   Mate selection strategies may be expressed by means of FBs. In particular, given two chromosomes, an FB may induce a probability of mating depending on their characteristics. This probability determines whether or not they are crossed. Then, the process of coevolution with FBs will discover FBs containing mate selection strategies that encourage recombination between chromosomes that have useful information (characteristics) to exchange.

The use of coevolution for producing adaptation is not new in the AGA literature. In [34], a separate GA is used as well, which controls the strategies that are applied to a main GA, in order to supervise the schemata that are processed by this main GA. In [54], an adaptive model based on coevolution is proposed for the case of the genetic programming [35]. Operator programs representing recombination operators are coevolved with the population of genetic programs and learn to recombine the main population programs better than a random genetic recombination. Another important example involves *self-adaptation* [3], [4], [6], [40], [46], [49], [57]. Generally, control parameters are directly coded onto each member of the population and this allows them to evolve, i.e., they undergo mutation and recombination. Self-adaptation exploits the indirect link between favorable control parameter values and objective function values, with the parameters being capable of adapting implicitly, according to the topology of the objective function [6].

The evolution of the FBs, like self-adaptation, allows valuable hints from evolution to be obtained implicitly and later they may be used to guide the further steps of the GA. However, there is a notable difference between the model presented and self-adaptation; FBs represent general strategies (fuzzy rule bases) instead of particular values. In this way, they may handle different environments represented by the possible combination of values of the features of the chromosomes.

IV. Adaptive FR by Coevolution with FBs

In this section, we implement an instance of the adaptation model proposed using a crossover operator called FR [59], which was presented for working with RCGAs. In RCGAs, chromosomes are vectors of floating point numbers, the size of which is kept the same as the length of the vector, which is the solution to the problem. They have been proven to be more efficient than binary-coded GAs in certain real parameter optimization problems [30], [44].

Next, we explain the reasons of having chosen FR for our instance application. Crossover operators for RCGAs are able to produce exploration or exploitation (at different degrees) depending on the way in which they handle the current diversity of the population. They may either generate additional diversity starting from the current one (and so exploration takes effect) or use this diversity for creating better elements (and so exploitation comes into force). The performance of an RCGA on a particular problem will be determined strongly by the degrees of exploration and exploitation associated with the crossover operator being applied. In the case of FR, these degrees may be easily adjusted by means of varying an associated operator parameter. By adjusting this parameter by coevolution with FBs, FR may be able to adapt its degrees of exploration and exploitation to the particular problem to be solved (which seems a promising way to improve the RCGA performance).

We present FR in Section IV-A, describe the RCGA model used as main-GA in Section IV-B, and in Section IV-C, explain how the adaptation model based on coevolution with FBs may be used for controlling the parameter associated with this operator, resulting an *adaptive* FR.

A. FR

Let us assume that $X = (x_1 \cdots x_n)$ and $Y = (y_1 \cdots y_n)$ $(x_i, y_i \in [a_i, b_i] \subset \mathbb{R}, i = 1 \cdots n)$ are two real-coded chromosomes to be crossed. Then, FR generates an offspring $Z = (z_1 \cdots z_n)$, where z_i is obtained from a distribution $\Phi(z_i) \in$

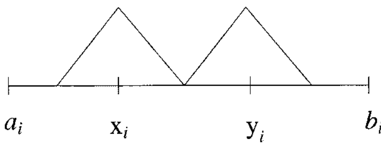


Fig. 3. FR operator.

$\{\phi_{x_i}, \phi_{y_i}\}$ in which ϕ_{x_i} and ϕ_{y_i} are triangular probability distributions having the following features ($x_i \leq y_i$ is assumed):

Prob.Dist.	Min.Val.	ModalVal.	Max.Val.
ϕ_{x_i}	$x_i - d \cdot \lvert y_i - x_i \rvert$	x_i	$x_i + d \cdot \lvert y_i - x_i \rvert$
ϕ_{y_i}	$y_i - d \cdot \lvert y_i - x_i \rvert$	y_i	$y_i + d \cdot \lvert y_i - x_i \rvert$

where $d \geq 0.5$ under its initial formulation. Fig. 3 shows an example of applying this crossover operator for the case of $d = 0.5$. The greater the d value is, the higher the variance (diversity) introduced into the population.

The strategy for applying this crossover operator to the population is the following: for each pair of chromosomes from a total of $p_c \cdot N$ (obtained from a sampling scheme), where p_c is the crossover probability and N is the population size, two offspring are generated, the result of applying the operator to them. They substitute their parents.

B. Main GA

We consider an RCGA model called two-loop RCGA with adaptive control of mutation step sizes (TRAMSS) [29] as main-GA that applies FR. TRAMSS is composed by an *inner* loop and an *outer* loop.

1) Inner Loop: It is designed for processing useful diversity in order to lead the population toward the most promising search areas, producing an effective refinement on them. So, its principal mission is to obtain the best possible *accuracy* levels.

The inner loop performs the selection process and fires the crossover and mutation operators. Furthermore, for achieving its objective, it controls the step size of the mutation operator.

2) Outer Loop: It introduces new population diversity, after the inner loop reaches a stationary point where there are no improvements, that helps the next inner loop to reach better solutions. Therefore, it attempts to induce *reliability* in the search process.

The outer loop iteratively performs the inner one, and later, it applies a *restart operator* that reinitializes the population by mutating all the genes, using a step size that is adapted as well, throughout the runs for this loop.

The Appendix provides a more detailed description of TRAMSS.

In [29], it is shown that TRAMSS manipulates the population diversity adequately for improving both reliability and accuracy with regard to other mechanisms presented for controlling mutation step sizes. This is an important feature for allowing the adaptation of FR to be carried through to a suitable conclusion, such as is suggested in [48, pp. 86]:

"It is the experience of several authors working with adaptive recombination mechanisms that convergence makes the relative assessment of different strategies impossible. Variety within the population is vital as the driving force of selective pressure in all evolutionary algorithms, and will be doubly so in self-adaptive algorithms."

Another reason for using TRAMSS is that it couples suitably with FR. In [29], it was suggested that this occurs because this crossover operator adjusts the intervals for the generation of genes depending on the current population diversity.

C. Adapting the d Parameter by Coevolution with FBs

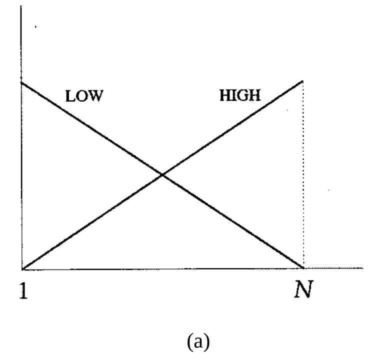
In this section, we use the application of the coevolution with FBs to the adaptation of the d operator parameter used by FR (its range was constrained to the interval $[0, 1]$). We describe first the FB structure chosen (Section IV-C1), then the fitness function for the FBs (Section IV-C2), and finally the design of FBs-GA for evolving FBs (Section IV-C3).

1) FB Structure: We propose using the index of the parents in the population $\text{Index}(X), \text{Index}(Y) \in \{1, \ldots, N\}$ (N is the population size) as the features of the parents to take into account for building FBs. The index of the best chromosome is N and the index of the worst chromosome is one (the fitter elements will have larger indexes). FBs have information for the adaptation of d depending on the goodness of the parents with regards to the chromosomes in the population.

Different mechanisms presented in the GA literature operate considering the importance of the fitness of the chromosomes in the population. An example is the linear ranking selection [7] in which the selection probability of each chromosome is computed according to its rank. Adaptive strategies at the individual level were proposed as well, which use this type of feature. For example, in [50], each chromosome has its own crossover and mutation probabilities, which are varied depending on the importance of its fitness value in the population, with the need to preserve good solutions (lower probability values are assigned for high fitness solutions and higher values for low fitness solutions). In [20], a similar approach is followed for the case of adapting the strength in which real-coded genes are mutated: genes in the fittest individuals undergo mutations with small strengths.

The set of linguistic labels associated with $\text{Index}(X)$ and $\text{Index}(Y)$ is $L = \{\text{Low}, \text{High}\}$. The meanings of these labels are depicted in Fig. 4(a). The set of linguistic labels for d is $L_d = \{\text{Low}, \text{Medium}, \text{High}\}$. Their meanings are shown in Fig. 4(b). Therefore, FBs are vectors with four positions containing labels belonging to L_d. Table I shows the antecedents to which each position corresponds. For example, the FB (High, Low, High, Medium) has associated the fuzzy rule base shown in Table II. Although the L set looks poor, it was chosen with regards to the following aspects.

- Its labels differentiate between two important categories of chromosomes: the best chromosomes and the worst ones (the heuristic rule underlying in the adaptive mechanism proposed in [50] is expressed in terms of these categories as well). FBs may induce d values attending to the level with which the parents match with the two categories. Experiments have demonstrated that this is sufficient for



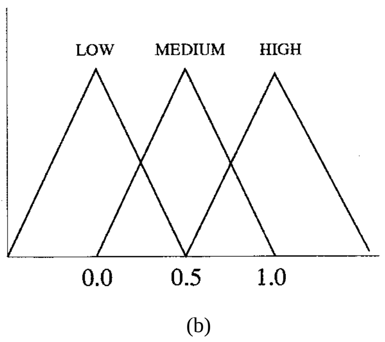


Fig. 4. Meanings for the linguistic labels considered. (a) Chromosome index. (b) d parameter.

TABLE I
ANTECEDENTS FOR EACH POSITION IN AN FB

Position	*Index*(X)	*Index*(Y)
1	High	High
2	High	Low
3	Low	High
4	Low	Low

TABLE II
FUZZY RULE BASE

Index(X)	*Index*(Y)	Consequent
High	High	High
High	Low	Low
Low	High	High
Low	Low	Medium

adapting the d parameter in a suitable way for improving the results of FR with static d values (Section V).

- The size of the space of FBs is $|L_d|^{|L|}$ (it increases with the number of labels in L following an exponential way). In a large space of FBs, FBs-GA may have difficulties for reaching suitable FBs and, therefore, the adaptive mechanism may not be effective. With $|L| = 2$, this space becomes easy to be dealt with.

2) Fitness for the FBs: As we have mentioned, the fitness function associated with the FBs should take into account the performance of the genetic operator when it is applied to the parents with the d value obtained from them. But according to what criterion should we judge this performance? One possibility that has received attention is the ability of an operator to produce children of improved fitness [15], [58], [62]. Clearly, this is necessary for optimization to progress (the aim of a GA is, after all, to uncover new fitter points in the search space). In fact, the overall performance of a GA depends upon it maintaining an acceptable level of productivity throughout the search [57]. However, this is not enough: an efficient crossover operator should introduce the right portion of variance into the offspring population. If the variance is too large, then the GA does not converge at all, whereas if it is too small, then it converges prematurely [59].

Taking into account this two-fold objective, we propose the following fitness function for each FB$_i$ in $P_{\text{FBs}}(t_2)$ (minimization is assumed)

$$\text{Fit}(\text{FB}_i) = \begin{cases} 0, & \text{if } \overline{f}_O < f(X) \le f(Y) \\ 2 - d_i, & \text{if } f(X) \le \overline{f}_O < f(Y) \\ 3, & \text{if } f(X) \le f(Y) \le \overline{f}_O \end{cases}$$

where

$\overline{f}_O$	average of the fitness of the two offspring generated;
$f(X)$ and $f(Y)$	fitness function values of the parents ($f(X) \le f(Y)$ is assumed);
d_i	d value calculated from FB$_i$ and $\text{Index}(X)$ and $\text{Index}(Y)$.

This function induces the aforementioned two-fold objective for reaching crossover operator performance in the following ways:

1) $\text{Fit}(\cdot)$ rewards FBs that produce offspring that are more fit than the parents;
2) $\text{Fit}(\cdot)$ penalizes FBs that produce offspring that are worse than the parents;
3) when the fitness of the offspring is between that of the parents, FBs introducing more diversity (those using greater d values) are preferred. Regardless, the fitness of these FBs will be better than the ones assigned to the FBs in case 2, and worse than the ones in case 1.

We should point out that in the experiments, $\overline{f}_O$ is the average of the fitness of the offspring after they undergo mutation. This is done in order to avoid the waste of fitness function evaluations after the crossover operator application. In doing so, it seems that the adaptation abilities of the mechanism may disappear. However, we think (and results have confirmed this) that the latter does not occur. The adaptation operates considering the joint effect of the crossover and mutation, as if they formed a single operator, but still exploiting its suitable abilities.

D. FBs-GA for Evolving FBs

The evolution of the FBs is carried out by means of a GA with the following properties.

1) Population Size: $|P_{\text{FBs}}| = (p_c \cdot (N/2))/n_{pp}$, i.e., the number of pairs of parents to be crossed in main-GA divided by n_{pp} (Section III-B).

2) Crossover Operator: FBs are crossed by means of the *simple* crossover operator [33], [66]. Given two FBs, FB$_1$ $= (l_1^1 \cdots l_n^1)$ and FB$_2$ $= (l_1^2 \cdots l_n^2)$, the offspring FB$_3$ $= (l_1^1, \ldots, l_i^1, l_{i+1}^2, \ldots, l_n^2)$ and FB$_4$ $= (l_1^2, \ldots, l_i^2, l_{i+1}^1, \ldots, l_n^1)$ are generated, where i is a random number belonging to $\{1, \ldots, n-1\}$. The crossover probability is 0.6.

3) Mutation Operator: The mutation of a gene in an FB is carried out as follows: 1) if the gene is High, produce Medium;

$$f_{sph}$$

$$f_{sph}(\vec{x}) = \sum_{i=1}^{n} x_i^2$$
$$-5.12 \leq x_i \leq 5.12$$
$$f_{sph}(x^*) = 0$$

$$f_{Ros}$$

$$f_{Ros}(\vec{x}) = \sum_{i=1}^{n-1}(100 \cdot (x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$$
$$-5.12 \leq x_i \leq 5.12$$
$$f_{Ros}(x^*) = 0$$

$$f_{Sch}$$

$$f_{Sch}(\vec{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$$
$$-65.536 \leq x_i \leq 65.536$$
$$f_{Sch}(x^*) = 0$$

$$f_{Gri}$$

$$f_{Gri}(\vec{x}) = \frac{1}{d}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$
$$d = 4000$$
$$-600.0 \leq x_i \leq 600.0$$
$$f_{Gri}(x^*) = 0$$

$$f_{Ras}$$

$$f_{Ras}(\vec{x}) = a \cdot n + \sum_{i=1}^{n} x_i^2 - a \cdot \cos(\omega \cdot x_i)$$
$$a = 10,\ \omega = 2\pi$$
$$-5.12 \leq x_i \leq 5.12$$
$$f_{Ras}(x^*) = 0$$

$$ef_{10}$$

$$ef_{10}(\vec{x}) = f_{10}(x_1, x_2) + \ldots + f_{10}(x_{n-1}, x_n) + f_{10}(x_n, x_1)$$
$$f_{10}(x, y) = (x^2 + y^2)^{0.25} \cdot [\sin^2(50 \cdot (x^2 + y^2)^{0.1}) + 1]$$
$$x, y \in (-100, 100]$$
$$ef_{10}(x^*) = 0$$

Fig. 5. Test functions.

2) if it is Medium, produce High or Low at random; and 3) if it is Low, produce Medium. The mutation probability is 0.01.

4) Selection Mechanism: The selection probability calculation follows *linear ranking* [7]. Chromosomes are sorted in order of raw fitness and then the selection probability of each chromosome C_i is computed according to its rank $\text{rank}(C_i)$ (with $\text{rank}(C_{\text{best}}) = 1$) by using the following nonincreasing assignment function

$$p_s(C_i) = \frac{1}{N} \cdot (\eta_{\max} - (\eta_{\max} - \eta_{\min}) \cdot \frac{\text{rank}(C_i) - 1}{N - 1})$$

where N is the population size and $\eta_{\min} \in [0, 1]$ specifies the expected number of copies for the worst chromosome (the best one has $\eta_{\max} = 2 - \eta_{\min}$ expected copies). In the experiments, $\eta_{\min} = 0.75$.

Linear ranking is performed along with *stochastic universal sampling* [8]. This procedure guarantees that the number of copies of any chromosome is bounded by the floor and by the ceiling of its expected number of copies.

5) Restart Mechanism: An important aspect to highlight is that the population of FBs is reinitialized at the same time as the restart mechanism in the outer loop of the main RCGA (Section IV-B) is fired. In the case of multimodal functions, the outer loop applies the restart mechanism after the inner loop has found and refined a local optimum. If the next inner loop run starts from a new population, then it will probably consider a new local optimum. The process for learning FBs should be restarted in order to be able to include the specific properties of this current optimum.

Finally, we should point out that the cooperation strategy between FBs-GA and main-GA follows the structure of the example described in Section III-B.

V. Experiments

Minimization experiments on the test suite described in Section V-A have been carried out in order to study the behavior of the Adaptive FR proposed in Section IV.

The algorithms built in order to do this are described in Section V-B, the results are shown in Section V-C, they are analyzed in Section V-D, a study of the adaptation itself is made in Section V-E, and finally, in Section V-F, the effects of the n_{pp} and t_{FBs} parameters (see Section III-B) upon Adaptive FR performance are analyzed.

A. Test Suite

For the experiments, we have considered six frequently used test functions: 1) sphere model (f_{sph}) [16], [45]; 2) Generalized Rosenbrock's function (f_{Ros}) [16]; 3) Schwefel's Problem 1.2 (f_{Sch}) [45]; 4) Griewangk's function (f_{Gri}) [22]; 5) Generalized Rastrigin's function (f_{Ras}) [4], [56]; and 6) Expansion of f_{10} (ef_{10}) [63]. Fig. 5 shows their formulation. The dimension of the search space is 25.

f_{sph} is a continuous, strictly convex, and unimodal function.

f_{Ros} is a continuous and unimodal function, with the optimum located in a steep parabolic valley with a flat bottom. This feature will probably cause slow progress in many algorithms since they must continually change their search direction to reach the optimum. This function has been considered by some authors to be a real challenge for any continuous function optimization program [47]. A great part of its difficulty lies in the fact that there are nonlinear interactions between the variables, i.e., it is *nonseparable.*

f_{Sch} is a continuous and unimodal function. Its difficulty concerns the fact that searching along the coordinate axes only gives

TABLE III
RULE BASES FOR T-FRB1 AND T-FRB2, RESPECTIVELY

	Index(Y)	
Index(X)	**High**	**Low**
High	High	Medium
Low	Medium	Low

	Index(Y)	
Index(X)	**High**	**Low**
High	Low	Medium
Low	Medium	High

a poor rate of convergence, since the gradient of f_{Sch} is not oriented along the axes. It presents similar difficulties to f_{Ros}, but its valley is much narrower.

f_{Ras} is a scalable, continuous, separable, and multimodal function, which is produced from f_{sph} by modulating it with $a \cdot \cos(\omega \cdot x_i)$.

f_{Gri} is a continuous and multimodal function. This function is difficult to optimize because it is nonseparable and the search algorithm has to climb a hill to reach the next valley. Nevertheless, one undesirable property exhibited is that it becomes easier as the dimensionality is increased [63].

f_{10} is a function that has nonlinear interactions between two variables. Its expanded version ef_{10} is built in such a way that it induces nonlinear interaction across multiple variables. It is nonseparable as well.

B. Algorithms

For the experiments, we implement a version of TRAMSS called T-FBs that uses adaptive FR (Section IV). It considers $n_{pp} = 1$ and $t_{\mathrm{FBs}} = 1$ (Section III-B). The general features of the FLCs are the following: the $\min$ operator is used for conjunction of clauses in the IF part of a rule, the $\min$ operator is used to fire each rule, and the *center of gravity weighted by matching* strategy as the defuzzification operator is considered. This setting was chosen from [13]. This paper studies the combination of inference systems and defuzzification methods using different applications and defining a degree of behavior. The previous combination was the most effective one in the sense that it obtained the best behavior for all the applications. Moreover, other papers have accepted the general good behavior for this combination [41], [42].

We include three TRAMSS instances based on FR with fixed d values. They are called T-0.0, T-0.5, and T-1.0 and use $d = 0.0$, $d = 0.5$, and $d = 1.0$, respectively. We also execute two algorithms where d is adapted at individual level through an FLC with $\mathrm{Index}(X)$ and $\mathrm{Index}(Y)$ as inputs. Each one uses a different fixed fuzzy rule base (Table III), which seemed particularly interesting for us. They are called T-FRB1 and T-FRB2.

All the algorithms were executed 30 times, each one with 10 000 generations. The crossover probability is 0.6, the mutation probability 0.005, and the population size is 60 chromosomes.

C. Results

Table IV shows the results obtained. The performance measures used are the following:

1) A performance: average of the best fitness function found at the end of each run;
2) SD performance: standard deviation. For f_{sph}, some standard deviations have been rounded to 0.0, because the corresponding algorithms have achieved very low results for this function;
3) B performance: best of the fitness values averaged as A performance. If the global optimum has been reached sometimes, this performance will represent the percentage of runs in which this happens.

Moreover, a two-sided t-test (H_o: means of the two groups are equal, H_a: means of the two group are not equal) at 0.05 level of significance was applied in order to ascertain if differences in the A performance for T-FBs are significant when compared against the one for the other algorithms in the respective table. The direction of any significant differences is denoted either by:

1) a plus sign $(+)$ for an improvement in A performance;
2) a minus sign $(-)$ for a reduction; or
3) an approximate sign $(\sim)$ for non significant differences.

The places in Table IV where these signs do not appear correspond with the performance values for T-FBs.

D. Analysis of the Results

First, we deal with the results of the TRAMSS versions with fixed d values, T-0.0, T-0.5, and T-1.0. Then, in order to study the effects of the adaptation of d at individual-level by means of an FLC, we compare the behavior of T-FRB1 and T-FRB2 with the one from the previous algorithms. Finally, in order to study the performance of adaptive FR, we compare the results of T-FBs with the ones from the other algorithms.

1) Analysis for T-0.0, T-0.5, and T-1.0: With regards to the TRAMSS versions with fixed d values, the following may be underlined.

- In general, the best A and B results are reached with $d = 0.5$ (T-0.5). This agrees with [59], where this d value seemed a good choice for a large class of functions. So, with $d = 0.5$, FR introduces a suitable balance between the progress of the average fitness and the standard variance for most functions.
- However, for f_{Ros}, T-0.0 shows better A behavior than T-0.5 and better B performance than T-0.5 and T-1.0. FR is a volume-oriented search. The volume to be searched is a hyperrectangle defined by the parents, which is parallel to the axes. If the minimum of the function is located at the end of a very steep and curved valley (as in the case of f_{Ros}), this operator will have difficulty in locating the minimum since the steep valley is a very small part of the hyperrectangle [59]. When $d = 0$, the hyperrectangle is minimum, allowing the steep valley to be followed in the most profitable way.

2) T-FRB1 and T-FRB2 Versus T-0.0, T-0.5, and T-1.0: T-FRB1 improves the A and B behavior of all TRAMSS versions with fixed d values for all test functions, except for f_{Ras}, and T-FRB2 does the same for f_{Ras}. These facts show that:

TABLE IV
RESULTS OF EXPERIMENTS

	f_{sph}			f_{Ros}			f_{Sch}		
Alg.	**A**	**SD**	**B**	**A**	**SD**	**B**	**A**	**SD**	**B**
T-0.0	3.67e-85 (+)	8.29e-85	6.41e-91	1.38e1 (~)	1.95e1	1.01e-1	8.06e-2 (+)	8.60e-2	6.31e-3
T-0.5	1.02e-184 (+)	0.00e0	2.09e-200	1.47e1 (+)	1.04e1	2.96e-1	9.29e-6 (+)	9.60e-6	5.88e-8
T-1.0	3.08e-129 (+)	1.11e-128	6.69e-135	1.26e1 (+)	1.17e0	9.08e0	1.24e0 (+)	1.37e0	7.27e-3
T-FRB1	1.19e-200 (~)	0.00e0	1.17e-201	1.18e1 (~)	1.26e1	7.97e-4	3.68e-7 (+)	6.82e-7	8.83e-10
T-FRB2	4.05e-170 (+)	0.00e0	2.90e-183	1.04e1 (~)	6.88e0	1.14e-2	1.72e-5 (+)	1.89e-5	2.27e-7
T-FBs	2.69e-200	0.00e0	2.76e-201	1.02e1	4.91e0	8.31e-2	9.25e-9	1.46e-8	5.88e-11

	f_{Ras}			f_{Gri}			ef_{10}		
Alg.	**A**	**SD**	**B**	**A**	**SD**	**B**	**A**	**SD**	**B**
T-0.0	3.32e-2 (~)	1.79e-1	56.6%	1.12e-2 (+)	1.11e-2	26.6%	1.98e-1 (~)	5.70e-1	5.04e-9
T-0.5	6.64e-2 (~)	2.48e-1	90.0%	3.70e-18 (~)	1.99e-17	96.6%	5.82e-4 (~)	2.18e-3	1.15e-49
T-1.0	5.20e0 (+)	2.33e0	9.95e-1	4.11e-4 (~)	2.21e-3	96.6%	3.00e-24 (~)	7.75e-24	2.19e-26
T-FRB1	1.09e0 (+)	1.32e0	40.0%	0.00e0 (~)	0.00e0	100.0%	1.71e-32 (~)	8.95e-32	1.67e-50
T-FRB2	0.00e0 (~)	0.00e0	100.0%	1.23e-3 (+)	3.23e-3	86.6%	1.55e-17 (~)	8.33e-17	8.55e-43
T-FBs	3.32e-2	1.79e-1	96.6%	0.00e0	0.00e0	100.0%	3.55e-25	1.91e-24	1.50e-48

- the adaptation at individual-level of the d parameter by means of an FLC is a suitable way for improving the results of FR;
- the use of inputs representing the importance of the fitness of the parents in the population [$\text{Index}(\cdot)$] has allowed the performance of the FLCs to be effective.

On the other hand, neither T-FRB1 nor T-FRB2 have introduced a good behavior for all test functions. T-FRB1 has achieved good results for most functions; however, it shows a low B measure (40%) for f_{Ras}. T-FRB2 has found the global optimum of f_{Ras} in the 100% of the runs; however, it has returned worse results than T-FRB1 for the remaining functions. Therefore, a different fuzzy rule base may be needed for obtaining the best results for each test function.

3) Analysis for T-FBs: Regarding the results for T-FBs, we may observe that:

- the t-test indicates that T-FBs improves the A performance of the other algorithms for the complex f_{Sch} (it achieves the best B behavior as well);
- its results for the remaining test functions are similar to the ones for the most successful algorithms. Particularly, the t-test results (and the B measure) show that there is no significant difference on performance between T-FBs and T-FRB1 for f_{sph} and f_{Gri} and the same occurs for f_{Ras} with regard to T-FRB2.

These results show that adaptive FR induces a *robust* operation. A robust operation means that the proposal obtains results that are similar to the ones for the most successful algorithms (based on fixed d values or fixed fuzzy rule bases) for each one of the test functions. However, the important point here is that the most successful algorithm for any problem may be different from the one for the other problems (as is claimed above). Figs. 6 and 7 were included in order to observe this fact graphically. They outline the log-scaled best fitness value for each generation in the first run of T-FBs, T-FRB1, and T-FRB2 on the unimodal f_{Sch} and the multimodal f_{Ras}, respectively. There are two notable differences between the plot for f_{Sch} (Fig. 6) and the one for f_{Ras} (Fig. 7).

- The plot for f_{Ras} shows many peaks while the one for f_{Sch} shows none. They are produced by the application of the restart operator, which is called by the outer loop of these algorithms after the inner loop found and refined a local optimum of this function. This does not occur for the case of f_{Sch} because it is unimodal.
- Another difference concerns the opposing behaviors of T-FRB1 and T-FRB2 on these functions. For f_{Ras}, T-FRB2 was able to find the global optimum (around the generation 2000) without requiring restarts (which suggests that the fuzzy rule base used by this algorithm is very adequate for dealing with this function). However, for f_{Sch}, it shows a poor behavior as compared with the one of T-FRB1, which, on the contrary, never found the global optimum of f_{Ras}.

On the other hand, we may see the good behavior of T-FBs for the two functions. For f_{Sch}, it followed the evolution trajectory of T-FRB1, reaching a better result, at the end. For f_{Ras}, it might learn suitable FBs for finding the global optimum after the third restart operator call (this fact will be checked in Section V-E2).

Finally, we should point out that the proposal needs more computational time than the algorithms with fixed settings. However, performance was compared with all algorithms requiring the same number of fitness evaluations, which is usually the principal time resource taken into account.

E. Study of the Adaptation Itself

There are at least two ways to study the operation of an adaptive mechanism for GAs [49]. The first is from the point of view of performance (test functions are commonly used to evaluate performance improvement). The second view is quite different in that it ignores performance and concentrates more on the adaptive mechanism itself, i.e., its ability to adjust the GA configuration according to the particularities of the problem to be solved. Once given these two points of view, it is natural to investigate the way in which adaptive behavior is responsible for the performance improvement.

In Section V-D, we studied adaptive FR from the first point of view. In this section, we consider the point of view of the adap-

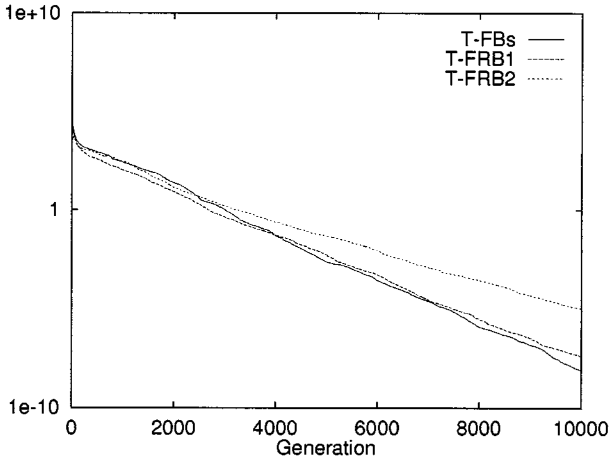


Fig. 6. Log-scaled best fitness value for each generation of T-FBs, T-FRB1, and T-FRB2 on $f_{\rm Sch}$.

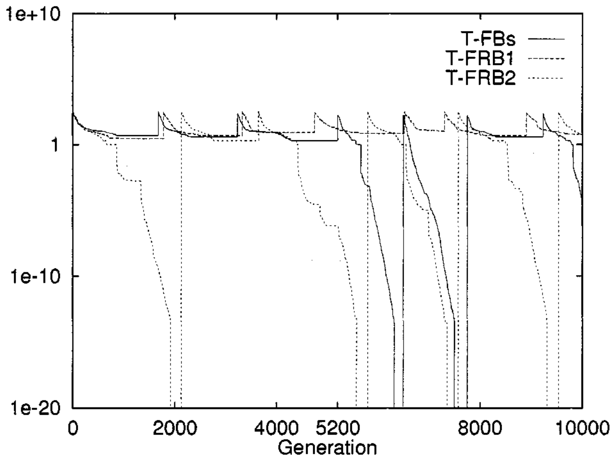


Fig. 7. Log-scaled best fitness value for each generation of T-FBs, T-FRB1, and T-FRB2 on $f_{\rm Ras}$. Peaks in this plot are associated with the application of the restart operator. T-FBs might find the global optimum after the third restart operator call produced at generation 5200.

tation itself. In order to do this, in Section V-E1, we analyze the distributions of FBs appearing during the runs of T-FBs for two test functions with different features. Furthermore, in Section V-E2, we check whether the adaptation behavior induces the performance improvement observed in Section V-D.

1) Distributions of FBs: Fig. 8 outlines the distributions of FBs for $f_{\rm sph}$ and $f_{\rm Ras}$ (which have different features as is indicated in Section V-A). FBs are represented in the coordinate axis following a lexicographical order, starting at *LLLL* (*Low, Low, Low, Low*) and ending at *HHHH* (*High, High, High, High*). The figure shows the number of instances of each FB appeared during the runs of T-FBs divided by the total number of FBs appeared. We may make two important observations.

- The *MMMM* FB and FBs similar to it (such as *MMHM, MHMM,* and *HMMM*) stand out in the two distributions. These FBs induce d values similar to 0.5, which is very suitable as was mentioned above. So, we may point out that Adaptive FR has been able to detect this circumstance.
- There are differences between the distributions. FBs located on the left side of *MMMM* are more fruitful for $f_{\rm Ras}$ than for $f_{\rm sph}$, whereas the opposite effect occurs with regard to the FBs on the right side. In the case of $f_{\rm Ras}$, these effects are more noticeable for FBs similar to *MMMM* having *Low* labels such as *LMMM* and *MLMM* and in the case of $f_{\rm sph}$, for ones that incorporate *High* labels such as *MHMM*, *MHHM*, and *HHMM*.

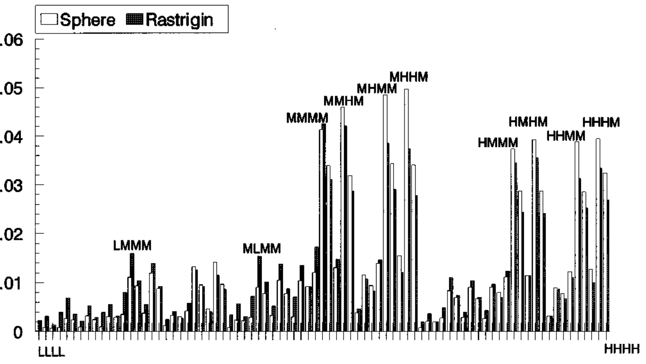


Fig. 8. Distributions of FBs for $f_{\rm sph}$ and $f_{\rm Ras}$.

TABLE V
RESULTS FOR $f_{\rm sph}$ AND $f_{\rm Ras}$ USING DIFFERENT FBs

FBs	f_{sph} A	SD	B
HMMM	1.57e-200	0.00e+00	7.88e-202
MMHM	2.05e-200	0.00e+00	1.35e-201
MHHM	2.24e-198	0.00e+00	1.84e-201
HHMM	9.45e-189	0.00e+00	4.68e-196

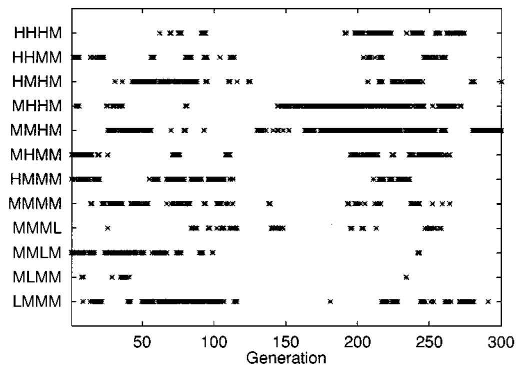
FBs	f_{Ras} A	SD	B
LMMM	0.00e+00	0.00e+00	100.00%
MLMM	0.00e+00	0.00e+00	100.00%
MMMM	3.32e-02	1.78e-01	90.00%
MMHM	3.98e-01	7.53e-01	70.00%

These differences between distributions arise as a sign of the adaptation ability (from the point of view of the adaptation itself) of adaptive FR since they confirm that this operator generates distributions of FBs whose shape depends on the particular problem to be solved.

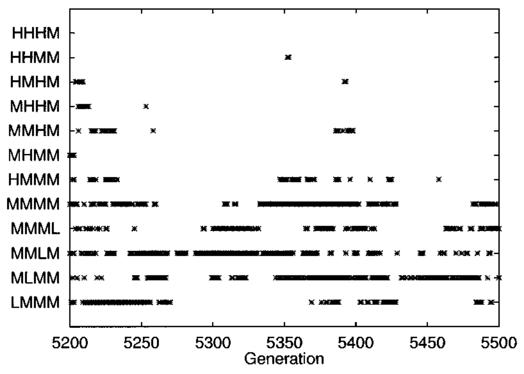
2) Adaptive Behavior and Performance Improvement: Although adaptive FR may show signs of adaptation, we have to check whether this one is the cause of performance improvement. In order to do so, we have executed TRAMSS versions that adapt d through a different FLC, each one based on FBs that represent significant points in the distributions in Fig. 8. Table V has these FBs along with their results.

We observe that FBs similar to *MMMM*, including *High*, labels are very suitable for $f_{\rm sph}$ and ones with *Low* labels are effective for $f_{\rm Ras}$. These results together with the observations of the previous section seem to corroborate that performance improvement is causally related with the adaptation ability since adaptive FR was able to generate different FBs depending on the particular problem (adaptation), which induce a suitable behavior on it (performance improvement).

To see this fact more clearly, we should establish a direct relation between the FBs produced during a particular time interval



(a)



(b)

Fig. 9. FBs generated for f_{Ros} during two runtime intervals. (a) From generation 1 to 300. (b) From generation 5200 to 5500.

and their impact on the behavior of T-FBs during this time interval. Fig. 9 was included for this purpose. Fig. 9(a) shows the appearance of significant FBs (chosen from Fig. 8) during the first 300 generations of the first run of T-FBs on f_{Ras}. Fig. 9(b) illustrates the same information, but during the 300 generations following the restart operator call produced at generation 5200 (see the third peak in the plot for T-FBs in Fig. 7). A mark is printed in the FBs being generated during the generations in these time intervals. A careful observation of the two graphs reveals that the most fruitful FBs are different from one to another.

- FBs similar to *MMMM* having a *Low* label (*LMMM*, *MLMM*, *MMLM*, etc.) predominate in Fig. 9(b) (these FBs have arisen as very suitable for this function as is shown in Table V), whereas the remaining ones are practically absent. Now, observing Fig. 7, we may observe that the effects of these FBs on the performance of T-FBs are very beneficial: T-FBs reaches the global optimum of f_{Ras} during generations following 5200.
- The situation of Fig. 9(a) is different. Here, the proportion of FBs having *High* labels stand out against the one of FBs having *Low* labels (these ones are very slight, in particular *MLMM* and *MMML*). Fig. 7 shows as this distribution of FBs caused T-FBs to be trapped in a local optimum during the initial generations of the run. So, in this case, the FBs learnt were not adequate for finding the global optimum.

We may remark two important conclusions from all these results: 1) Adaptive FR allows adequate distributions of FBs to be generated (thanks to its adaptation ability) for producing a robust operation for problems with different difficulties, and 2) a part of this suitable operation is possible thanks to the use of the TRAMSS model, since its restart operator (along with the restart operator applied to FBs-GA) gives Adaptive FR the chance of learning adequate FBs for different search zones as soon as one of them has been totally exploited.

F. Empirical Study of n_{pp} and t_{FBs}

As explained in Section III-B, the adaptive mechanism proposed requires two new control parameters, n_{pp} and t_{FBs}, which determine the number of pairs of parents to which an FB is assigned and the gap between consecutive applications of FBs-GA, respectively. In the previous experiments, they were fixed to one. Next, we study their effects upon the performance of Adaptive FR in Sections V-F1 and V-F-2.

1) Effects of Varying n_{pp}: The relationship between the performance of the T-FBs algorithm and n_{pp} is investigated in this section. In particular, we set the main population size to 180 and t_{FBs} to one and consider the results of T-FBs with $n_{pp} = 1, 2,$ and 3 (i.e., $|P_{\mathrm{FBs}}(t)| = 54, 27,$ and 18, respectively). When $n_{pp} \geq 1$, each FB is evaluated using the average of the individual fitness values (Section IV-C2) obtained from each one of its pairs of parents assigned. A *t*-test was applied in order to ascertain if differences in the A performance for $n_{pp} = 1$ are significant when compared against the one for $n_{pp} = 2, 3$. The results are shown in Table VI.

In general, we notice no dramatic differences in the performance measures of T-FBs with different n_{pp} values (see *t*-test results). These results agree with the claim made in Section III-B: the effects achieved with $n_{pp} \geq 1$ (to obtain FBs capturing knowledge for inducing adequate p values for parents showing very different feature values) are provided with $n_{pp} = 1$ as well, since successive crossovers of FBs being found adequate for one combination of parent features will produce FBs that are adequate for many combinations. This may explain the similarity in the results obtained.

2) Effects of Varying t_{FBs}: In this section, we examine the performance of T-FBs when using different values of t_{FBs} (1, 5, 10, and 20). The population size of main-GA is 60 and n_{pp} is set to one. When $t_{\mathrm{FBs}} \geq 1$, the evaluation of the FBs is carried out following the same way as in the previous section. A t-test was applied in order to support the possible differences in the A performance for $t_{\mathrm{FBs}} = 1$ compared with the one for the other values. The results are shown in Table VII.

We observe that with $t_{\mathrm{FBs}} = 1$, the higher level of robustness is achieved (see the *t*-test results for f_{sph}, f_{Ros}, and f_{Sch}, and the B performance for f_{Ras} and f_{Gri}). Furthermore, for f_{sph}, f_{Ros}, and f_{Sch}, we see a significant drop in performance with $t_{\mathrm{FBs}} \geq 1$.

- For the case of f_{sph}, these results are not the expected ones. With $t_{\mathrm{FBs}} \geq 1$, FBs may be evaluated with regards to their historical performance, which seems a promising heuristic for this function with a regular landscape. However, since FBs-GA applies its selection mechanism less frequently than with $t_{\mathrm{FBs}} = 1$, some FBs may be kept (and applied) showing low performance. With $t_{\mathrm{FBs}} = 1$, these FBs disappears rapidly, and so results are better.

TABLE VI
RESULTS FOR T-FBs WITH $n_{pp} = 1, 2$, and 3

	f_{sph}			f_{Ros}			f_{Sch}		
n_{pp}	A	SD	B	A	SD	B	A	SD	B
1	2.42e-141	7.79e-141	2.23e-147	4.81e0	6.28e0	4.30e-3	3.35e-6	3.60e-6	4.98e-8
2	8.36e-139(+)	3.29e-138	6.07e-151	7.73e0 (~)	1.28e1	4.22e-3	1.33e-4 (~)	7.03e-4	5.49e-8
3	3.22e-140(+)	1.46e-139	8.64e-149	6.77e0 (~)	6.77e0	1.12e-3	4.81e-6 (~)	6.64e-6	8.81e-9
	f_{Ras}			f_{Gri}			ef_{10}		
n_{pp}	A	SD	B	A	SD	B	A	SD	B
1	0.00e0	0.00e0	100.00%	3.28e-3	6.17e-3	70.00%	5.13e-4	2.76e-3	4.21e-20
2	0.00e0 (~)	0.00e0	100.00%	2.14e-3(~)	4.20e-3	76.67%	1.15e-17(~)	3.75e-17	2.96e-20
3	0.00e0 (~)	0.00e0	100.00%	2.30e-3(~)	4.31e-3	76.67%	1.14e-1 (~)	6.15e-1	4.89e-21

TABLE VII
RESULTS FOR T-FBs WITH $t_{\text{FBs}} = 1, 5, 10$, AND 20

	f_{sph}			f_{Ros}			f_{Sch}		
t_{FBs}	A	SD	B	A	SD	B	A	SD	B
1	2.69e-200	0.00e0	2.76e-201	1.02e1	4.91e0	8.31e-2	9.25e-9	1.46e-8	5.88e-11
5	3.40e-194(+)	0.00e0	2.97e-201	8.38e0 (~)	6.52e0	8.36e-3	2.32e-8 (~)	4.80e-8	1.62e-11
10	8.36e-167(+)	0.00e0	3.03e-201	1.12e1 (+)	1.40e1	1.53e-2	3.69e-6 (+)	7.12e-6	5.67e-12
20	5.46e-168(+)	0.00e0	2.24e-201	1.10e1 (+)	1.35e1	1.24e-4	1.27e-6 (+)	2.20e-6	1.48e-12
	f_{Ras}			f_{Gri}			ef_{10}		
t_{FBs}	A	SD	B	A	SD	B	A	SD	B
1	3.32e-2	1.79e-1	96.67%	0.00e0	0.00e0	100.00%	3.55e-25	1.91e-24	1.50e-48
5	3.32e-2 (~)	1.79e-1	96.67%	4.93e-4(~)	1.84e-3	93.33%	2.14e-29(~)	1.15e-28	4.30e-50
10	0.00e0 (~)	0.00e0	100.00%	0.00e0 (~)	0.00e0	100.00%	3.58e-4 (~)	1.93e-3	3.20e-50
20	0.00e0 (~)	0.00e0	100.00%	0.00e0 (~)	0.00e0	100.00%	1.18e-5 (~)	6.37e-5	2.10e-50

- For the case of f_{Ros} and f_{Sch}, we think that the lower performance of $t_{\text{FBs}} \geq 1$ is due to the location of their global optimum (in a steep parabolic valley with a flat bottom). This feature will probably cause a misleading synchronization between main-GA and FBs-GA (see Section III-B). With $t_{\text{FBs}} = 1$, the synchronization is established in a more direct way, which allowed better results for these functions to be reached. Finally, we should observe an important fact: the B measure for these functions is better as t_{FBs} increases. During some runs the synchronization was suitable, the historical knowledge about the performance of the FBs was useful for leading the search toward better solutions. However, this did not happen during all the runs (as shown by the A measure).

These results along with the ones of the previous section justify the choice of $n_{pp} = 1$ and $t_{\text{FBs}} = 1$ as suitable values, since they allow the higher level of robustness to be kept on the different test functions. This conclusion agrees with the recommendations made in Section III-B.

VI. CONCLUDING REMARKS

This paper presented a general model for the adaptation of genetic operators based on coevolution with FBs. Values for the parameter to control are specified by FLCs during each genetic operator application event depending on particular features of the parents. The fuzzy rule bases of these FLCs are learnt by means of a separate GA that coevolves with the GA that uses the genetic operator to be controlled. In order to study the effectiveness of the model, we have built an instance for the adaptation of the d parameter associated with FR. An empirical study of this instance has been made from two different points of view, one of performance and one of adaptation itself, and the relation between them has been established. The principal conclusions reached are the following.

- The adaptation at the individual level of the d parameter by means of an FLC (whose inputs represent the importance of the fitness of the parents in the population) is a suitable way for improving the results of FR.
- The adaptation ability of adaptive FR allows suitable distributions of FBs to be obtained for producing a robust operation for test functions with different difficulties. Moreover, the integration of this operator in the TRAMSS model has been fundamental in order to demonstrate its good qualities.

These conclusions show promise in the use of the adaptation of genetic operators by coevolution of FBs, for future applications and extensions. They include the following: 1) the adaptation of other genetic operators whose operation is determined by particular control parameters, such as the ones enumerated in Section III-C and ones proposed for genetic programming systems [35]; 2) include into the FBs the meaning of the linguistic labels used, allowing the learning of the whole knowledge base; and 3) the extension to adaptations at component level.

Finally, we should explain the position of the model proposed in relation to the topic of the integration of GAs and *fuzzy logic*. This integration has been accomplished by following two different approaches [25]: 1) the use of fuzzy logic-based techniques for improving GA behavior (such as AGAs based on FLCs) and 2) the application of GAs in optimization and search problems involving fuzzy systems [14], [24], [32], [38], [39]). The adaptation of genetic operators by coevolution with FBs operates by means of the integration of both techniques (GA adaptation based on FLCs and learning of fuzzy rule bases through GAs), each one following one of these approaches.

APPENDIX
DESCRIPTION OF TRAMSS

The mutation operator may be considered to be an important element for solving the premature convergence problem since it serves to create random diversity in the population. In the case of working with real coding, a topic of major importance related with this operator involves the control of the proportion or strength in which real-coded genes are mutated, i.e., the *step size* [5].

In this Appendix, we describe TRAMSS [29]. It adjusts the step size of a mutation operator applied during the inner loop, for producing efficient local tuning. It also controls the step size of a mutation operator used by a restart operator performed in the outer loop, for reinitializing the population in order to ensure that different promising search zones are focused by the inner loop throughout the run.

In Sections A and B, we describe the TRAMSS inner and outer loops, respectively.

A. TRAMSS Inner Loop

The inner loop performs the usual GA process (selection, crossover, and mutation) over a number of generations, G, called the *time interval between observations*. Then, depending on the progress of the population mean fitness found throughout these generations, it adjusts the step size of the mutation operator, and calculates a new value for G.

In short, the objective of the inner loop is to find and refine local optima (or the global one), in an efficient way. Next, we outline this loop (a minimization problem is assumed).

Selection and Mutation (Step 2.2).: Over the time interval between observations G, the following selection mechanism and crossover and mutation operators are applied.

- The selection probability calculation follows *linear ranking* [7], with $\eta_{\min} = 0.25$ and the sampling algorithm is the *stochastic universal sampling* [8]. The *elitist strategy* [16] is considered as well. It involves making sure that the best performing chromosome always survives intact from one generation to the next. This is necessary since it is possible that the best chromosome disappears, due to crossover or mutation.
- FR and adaptive FR were considered as crossover operators.
- The mutation operator used is denoted as *Mutation(δ)*, where δ is the step size ($0 \le \delta \le 1$). This operator is defined as follows: if $x \in [a, b]$ is a gene to be mutated, then the gene resulting from the application of this operator x' will be a random (uniform) number chosen from $[x - \delta \cdot (x - a), x + \delta \cdot (b - x)]$.

```
1. $\delta := \Delta$; $G := G_0$; $yes := 0$; $no := 0$;
2. while ($\delta \ge \delta_{min}$) and (not termination-condition) do
    2.1. $\bar{f}_{Old} := \bar{f}$;
    2.2. perform Selection, Crossover and Mutation($\delta$) over $G$ generations;
    2.3. if ($\bar{f}_{Old} \ge \bar{f}$) then
            yes:=yes+1; no:=0;
            $\delta := \delta \cdot 2^{yes}$; if ($\delta > \Delta$) then $\delta := \Delta$;
         else
            no:=no+1; yes:=0;
            $\delta := \delta / 2^{no}$;
    2.4. $G := G_0 \cdot \delta/\Delta$; if ($G < G_{min}$) then $G := G_{min}$;
```

Fig. 10. TRAMSS inner loop structure.

Adaptive Control of δ (Step 2.3): After G generations, the δ parameter used by the mutation operator is adapted according to the following heuristic: "*increase δ when observing progress on $\overline{f}$ (population mean fitness), decrease it when stuck.*" δ is kept in the interval $[\delta_{\min}, \Delta]$, where Δ is a parameter calculated by the outer loop as described in Section B and $\delta_{\min}$ ($\delta \ge \delta_{\min}$) is the minimum threshold defined by the user (in experiments we assume a value of 1.0e-100).

The inner loop ends when δ reaches the $\delta_{\min}$ value or when a maximum number of generations is reached.

The update rates for δ depend on the number of previous successive observations that were successful or not successful. Two variables, yes and no, are used for recording these occurrences, respectively. If progress is made during many successive previous observations ($\overline{f}_{\text{Old}} \ge \overline{f}$, $\overline{f}_{\text{Old}}$ being the population mean fitness of the previous iteration), then the increasing rate for δ is very high (in particular, δ is multiplied by 2^{yes}), whereas if these observations were not successful, then the decreasing rate is high (δ is divided by 2^{no}).

Time Interval Calculation (Step 2.4): The time interval between observations G is calculated depending on the current values of δ with regard to Δ. If δ is similar to Δ, then the time interval is high ($G_0 = 100$ in the experiments) and, if it is lower, the time interval will become like $G_{\min}$ ($G_{\min} = 5$ in the experiments).

Fig. 10 shows the pseudocode algorithm for the whole TRAMSS inner loop.

B. TRAMSS Outer Loop

The outer loop randomly initializes the population that will be handled throughout the TRAMSS run. It fires the inner loop and when this one returns, it applies a restart operator that reinitializes the population. The restart operator uses a mutation whose step size is controlled adaptively throughout its execution.

The outer loop attempts to introduce adequate diversity levels for allowing the subsequent inner loop processing to be capable of finding better local optima or the global one every time. Now, we outline the main issues related to this loop.

Restart Operator Application (Step 3.4): The outer loop applies a restart operator called *Restart* (Δ) ($0 \le \Delta \le 1$) that applies *Mutation* (Δ) to all the genes in the chromosomes stored in the population. The objective of this operator is to maintain a

```
1. $\Delta := \Delta_0$;
2. run Initialize;
3. while (not Termination-condition) do
    3.1. $f_{OldBest} := f_{Best}$;
    3.2. run Inner Loop;
    3.3. if ($f_{OldBest} \geq f_{Best}$) then
            $\Delta := \Delta/2$; if ($\Delta < \delta_{min}$) then $\Delta := \delta_{min}$;
         else
            $\Delta := \Delta \cdot 2$; if ($\Delta > 1$) then $\Delta := 1$;
    3.4. run Restart($\Delta$);
```

Fig. 11. TRAMSS outer loop structure.

continuous level of exploration of the search space while trying to use the promising zones located as a kind of sketch.

Adaptive Control of Δ (Step 3.3): The outer loop adapts the Δ parameter using information obtained after each inner loop run by means of the following heuristic: "*decrease Δ when observing progress on $f_{\rm Best}$ (fitness of the best element found so far), otherwise increase it.*" This is implemented by dividing the previous Δ value by two or multiplying it by two, respectively. The new Δ value will be the first value for the δ parameter used in the next inner loop. Δ is kept in the interval $[\delta_{\rm min}, 1]$. Its initial value Δ_0 was assigned to 1 in the experiments.

The pseudocode algorithm for the outer loop is depicted in Fig. 11.

Acknowledgment

The authors would like to thank the anonymous referees and D.B. Fogel for their valuable comments which have improved the presentation of the paper.

References

[1] P. J. Angeline, "Adaptive and self-adaptive evolutionary computations," in *Computational Intelligence: A Dynamic Systems Perspective*, M. Palaniswami, Y. Attikiouzel, R. Marks, D. Fogel, and T. Fukuda, Eds. Piscataway, NJ: IEEE Press, 1995, pp. 152–163.

[2] S. Arnone, M. Dell'Orto, and A. Tettamanzi, "Toward a fuzzy government of genetic populations," in *Proc. of the 6th IEEE Conference on Tools with Artificial Intelligence*. Los Alamitos, CA: IEEE Comput. Soc. Press, 1994, pp. 585–591.

[3] T. Bäck, "The interaction of mutation rate, selection, and self-adaptation within genetic algorithm," in *Parallel Problem Solving from Nature 2*, R. Männer and B. Manderick, Eds. Amsterdam, The Netherlands: Elsevier, 1992, pp. 85–94.

[4] ——, "Self-adaptation in genetic algorithms," in *Proc. of the First European Conference on Artificial Life*, F. J. Varela and P. Bourgine, Eds. Cambridge, MA: MIT Press, 1992, pp. 263–271.

[5] T. Bäck, M. Schütz, and S. Khuri, "Evolution strategies: An alternative evolution computation method," in *Artificial Evolution*, J. M. Alliot, E. Lutton, E. Ronald, M. Schoenhauer, and D. Snyers, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 3–20.

[6] T. Bäck and M. Schütz, "Intelligent mutation rate control in canonical genetic algorithms," in *Foundation of Intelligent Systems 9th International Symposium*, Z. W. Ras and M. Michalewicz, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 158–167.

[7] J. E. Baker, "Adaptive selection methods for genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications*, J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1985, pp. 101–111.

[8] J. E. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proceedings of the 2nd International Conference on Genetic Algorithms and Their Applications*, J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 14–21.

[9] H. J. C. Barbosa, "A coevolutionary genetic algorithm for a game approach to structural optimization," in *Proc. of the Seventh Int. Conf. on Genetic Algorithms*, T. Bäck, Ed. San Mateo, CA: Morgan Kaufmann, 1997, pp. 545–552.

[10] A. Bárdossy and L. Duckstein, *Fuzzy Rule-Based Modeling with Applications to Geophysical, Biological and Engineering Systems*. Boca Raton, FL: CRC, 1995.

[11] A. Bastian and I. Hayashi, "A proposal for knowledge-based systems using fuzzy rules and genetic algorithms," *Jpn. J. Fuzzy Theory Syst.*, vol. 8, no. 6, pp. 895–907, 1996.

[12] A. Bergmann, W. Burgard, and A. Hemker, "Adjusting paremeters of genetic algorithms by fuzzy control rules," in *New Computing Techniques in Physics Research III*, K.-H. Becks and D. Perret-Gallix, Eds. Singapore: World Scientific, 1994, pp. 235–240.

[13] O. Cordón, F. Herrera, and A. Peregrín, "Applicability of the fuzzy operators in the design of fuzzy logic controllers," *Fuzzy Sets Syst.*, vol. 86, no. 1, pp. 15–41, 1997.

[14] O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples," *Int. J. Approx. Reason.*, vol. 17, no. 4, pp. 369–407, 1997.

[15] L. Davis, "Adapting operator probabilities in genetic algorithms," in *Proc. of the Third Int. Conf. on Genetic Algorithms*, J. D. Schaffer, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 61–69.

[16] K. A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," Ph.D. dissertation, Univ. of Michigan, Ann Arbor, MI, 1975.

[17] D. Driankow, H. Hellendoorn, and M. Reinfrank, *An Introduction to Fuzzy Control*. Berlin, Germany: Springer-Verlag, 1993.

[18] L. J. Eshelman, A. Caruana, and J. D. Schaffer, "Biases in the crossover landscape," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, J. D. Schaffer, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 86–91.

[19] L. J. Eshelman, K. E. Mathias, and J. D. Schaffer, "Crossover operator biases: Exploiting the population distribution," in *Proceedings of the 7th International Conference on Genetic Algorithms*, T. Bäck, Ed. San Mateo, CA: Morgan Kaufmann, 1997, pp. 354–361.

[20] D. B. Fogel, "An analysis of evolutionary programming," in *Proceedings of the 1st Annual Conference on Evolutionary Programming*, D. B. Fogel and W. Atmar, Eds. La Jolla, CA: Evol. Programm. Soc., 1992, pp. 43–51.

[21] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-16, pp. 122–128, Jan./Feb. 1986.

[22] A. O. Griewangk, "Generalized descent of global optimization," *J. Optim. Theory Applicat.*, vol. 34, pp. 11–39, 1981.

[23] F. Herrera, E. Herrera-Viedma, M. Lozano, and J. L. Verdegay, "Fuzzy tools to improve genetic algorithms," in *Proc. 2nd Eur. Congr. Intelligent Techniques Soft Computing*, Sept. 1994, pp. 1532–1539.

[24] F. Herrera, M. Lozano, and J. L. Verdegay, "Tuning fuzzy logic controllers by genetic algorithms," *Int. J. Approx. Reason.*, vol. 12, no. 3–4, pp. 299–315, Apr./May 1995.

[25] ——, "Tackling fuzzy genetic algorithms," in *Genetic Algorithms in Engineering and Computer Science*, G. Winter, J. Periaux, M. Galan, and P. Cuesta, Eds. New York: Wiley, 1995, pp. 167–189.

[26] ——, "Dynamic and heuristic fuzzy connectives-based crossover operators for controlling the diversity and convengence of real coded genetic algorithms," *Int. J. Intell. Syst.*, vol. 11, no. 12, pp. 1013–1041, Dec. 1996.

[27] F. Herrera and M. Lozano, "Heuristic crossovers for real-coded genetic algorithms based on fuzzy connectives," in *4th International Conference on Parallel Problem Solving From Nature*, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 336–345.

[28] ——, "Adaptation of genetic algorithm parameters based on fuzzy logic Controllers," in *Genetic Algorithms and Soft Computing*, F. Herrera and J. L. Verdegay, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 95–125.

[29] ——, "Two-loop real-coded genetic algorithms with adaptive control of mutation step sizes," *Appl. Intell.*, vol. 13, no. 3, pp. 187–204, Nov. 2000.

[30] F. Herrera, M. Lozano, and J. L. Verdegay, "Tackling real-coded genetic algorithms: Operators and tools for the behavioral analysis," *Artif. Intell. Rev.*, vol. 12, no. 4, pp. 265–319, 1998.

[31] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, pp. 124–141, July 1999.
[32] F. Hoffmann and G. Pfister, "Evolutionary design of a fuzzy knowledge base for a mobile robot," *Int. J. Approx. Reas.*, vol. 17, no. 4, pp. 447–469, 1997.
[33] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[34] Y. Kakazu, H. Sakanashi, and K. Suzuki, "Adaptive search strategy for genetic algorithms with additional genetic algorithms," in *Parallel Problem Solving from Nature 2*, R. Männer and B. Manderick, Eds. Amsterdam, The Netherlands: Elsevier, 1992, pp. 311–320.
[35] J. R. Koza, *Genetic Programing: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.
[36] M. A. Lee and H. Takagi, "Dynamic control of genetic algorithms using fuzzy logic techniques," in *Proceedings of the 5th International Conference on Genetic Algorithms*, S. Forrest, Ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 76–83.
[37] ——, "A framework for studying the effects of dynamic crossover, mutation, and population sizing in genetic algorithms," in *Advances in Fuzzy Logic, Neural Networks and Genetic Algorithms*, T. Furuhashi, Ed. Berlin, Germany: Springer-Verlag, 1994, pp. 111–126.
[38] L. Magdalena and F. Monasterio-Huelin, "A fuzzy logic controller with learning through evolution of its knowledge base," *Int. J. Approx. Reason.*, vol. 16, no. 3-4, pp. 335–358, Apr. 1997.
[39] C. Moraga and E. M. Zu Bexten, "Fuzzy knowledge-based genetic algorithms," *Inf. Sci.*, vol. 103, no. 1-4, pp. 101–114, Dec. 1997.
[40] J. Reed, R. Toombs, and N. A. Barricell, "Simulation of biological evolution and machine learning," *J. Theor. Biol.*, vol. 17, pp. 319–342, 1967.
[41] I. Rojas, O. Valenzuela, M. Anguita, and A. Prieto, "Analysis of the operators involved in the definition of the implication functions and in the fuzzy inference process," *Int. J. Approx. Reason.*, vol. 19, no. 3-4, pp. 367–389, Oct. 1998.
[42] I. Rojas, J. Ortega, F. J. Pelayo, and A. Prieto, "Statistical analysis of the main parameters in the fuzzy inference process," *Fuzzy Sets Syst.*, vol. 102, no. 2, pp. 157–173, Mar. 1999.
[43] E. Ronald, "When selection meets seduction," in *Proceedings of the 5th International Conference on Genetic Algorithms*, S. Forrest, Ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 167–173.
[44] R. Salomon, "The influence of different coding schemes on the computational complexity of genetic algorithms in function optimization," in *4th International Conference on Parallel Problem Solving from Nature*, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Berlin: Springer-Verlag, 1996, pp. 227–235.
[45] H.-P. Schwefel, *Numerical Optimization of Computer Models*. New York: Wiley, 1981.
[46] J. D. Schaffer and A. Morishima, "An adaptive crossover distribution mechanism for genetic algorithms," in *Proceedings of the 2nd International Conference on Genetic Algorithms and Their Applications*, J. J. Grefenstette, Ed. Hillsdale, NJ: Lawrence Erlbaum, 1987, pp. 36–40.
[47] D. Schlierkamp-Voosen and H. Mühlenbein, "Strategy adaptation by competing subpopulations," in *Parallel Problem Solving from Nature PPSN III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. Berlin, Germany: Springer-Verlag, 1994, pp. 199–208.
[48] J. E. Smith and T. C. Fogarty, "Operator and parameter adaptation in genetic algorithms," *Soft Computing*, vol. 1, no. 2, pp. 81–87, 1997.
[49] W. M. Spears, "Adapting crossover in evolutionary algorithms," in *Proc. of the Evolutionary Programming Conference 1995*, L. J. Fogel, P. J. Angenline, and T. Bäck, Eds. Cambrige, MA: MIT Press, 1995, pp. 367–384.
[50] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Trans. Systems, Man, Cybern.*, vol. 24, pp. 656–667, Apr. 1994.
[51] R. Subbu, A. C. Sanderson, and P. P. Bonissone, "Fuzzy logic controlled genetic algorithms versus tuned genetic algorithms: An agile manufacturing application," unpublished.
[52] G. Syswerda, "Uniform crossover in genetic algorithms," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, J. D. Schaffer, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 2–9.
[53] ——, "Simulated crossover in genetic algorithms," in *Foundations of Genetic Algorithms 2*, L. D. Whitley, Ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 239–255.
[54] A. Teller, "Evolving programmers: The co-evolution of intelligent recombination operators," in *Advances in Genetic Programming 2*, P. J. Angeline and K. E. Kinnear Jr., Eds. Cambridge, MA: MIT Press, 1996, pp. 45–68.
[55] A. G. Tettamanzi, "Evolutionary algorithms and fuzzy logic: A two-way integration," in *2nd Joint Conference on Information Sciences*. Durham, NC, 1995, pp. 464–467.
[56] A. Töorn and Ž. Antanas, "Global Optimization," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 1989, vol. 350.
[57] A. L. Tuson and P. Ross, "Adapting operator settings in genetic algorithms," *Evol. Comp.*, vol. 6, no. 2, pp. 161–184, 1998.
[58] ——, "Cost based rate adaptation: An investigation," in *4th International Conference on Parallel Problem Solving From Nature*, H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds. Berlin, Germany: Springer-Verlag, 1996, pp. 461–469.
[59] H.-P. Voigt, H. M. Mühlenbein, and H. Cvetković, "Fuzzy Recombination for the breeder genetic algorithm," in *Proceedings of the 6th International Conference on Genetic Algorithms*, L. Eshelman, Ed. San Francisco, CA: Morgan Kaufmann, 1995, pp. 104–111.
[60] S. Voget, "Multiobjective optimization with genetic algorithms and fuzzy-control," in *Proc. 4th Eur. Congr. Intelligent Techniques and Soft Computing*, Sept. 1996, pp. 391–394.
[61] S. Voget and M. Kolonko, "Multi-criteria optimization with a fuzzy-genetic-algorithm," *J. Heuristics*, vol. 4, no. 3, pp. 221–244, Sept. 1998.
[62] T. White, "Adaptive crossover using automata," in *Proc. of the Parallel Problem Solving from Nature III*, Y. Davidor, H.-P. Schwefel, and R. Männer, Eds. Berlin, Germany: Springer-Verlag, 1994, pp. 228–237.
[63] D. Whitley, R. Beveridge, C. Graves, and K. Mathias, "Test driving three 1995 genetic algorithms: New test functions and geometric matching," *J. Heuristics*, vol. 1, no. 2, pp. 77–104, Fall 1995.
[64] H. Y. Xu, G. Vukovich, Y. Ichikawa, and Y. Ishii, "Fuzzy evolutionary algorithms and automatic robot trajectory generation," in *Proc. 1st IEEE Conf. Evolutionary Computation*, June 1994, pp. 595–600.
[65] X. Zeng and B. Rabenasolo, "A fuzzy logic based design for adaptive genetic algorithms," in *Proc. 2nd Eur. Congr. Intelligent Techniques and Soft Computing*, Sept. 1994, pp. 1532–1539.
[66] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Cambridge, MA: MIT Press, 1992.

Francisco Herrera received the M.S. and Ph.D. degrees in mathematics from the University of Granada, Granada, Spain, in 1998 and 1991, respectively.

He is currently an Associate Professor with the Department of Computer Science and Artificial Intelligence, University of Granada. He has coedited *Genetic Algorithms and Soft Computing* (New York: Springer-Verlag, 1996) and two journal Special Issues, "Genetic Fuzzy Systems for Control and Robotics" (*International Journal of Approximate Reasoning*) and "Genetic Fuzzy Systems" (*International Journal of Intelligent Systems*). His research interests include decision-making problems in fuzzy environment, fuzzy rule-based systems, machine learning, genetic algorithms, genetic fuzzy systems, and combination of fuzzy logic and genetic algorithms.

Manuel Lozano received the M.Sc. and Ph.D. degrees in computer science from the University of Granada, Granada, Spain, in 1992 and 1996, respectively.

He is currently an Associate Professor with the Department of Computer Science and Artificial Intelligence, University of Granada. His research interests include fuzzy rule-based systems, machine learning, genetic algorithms, genetic fuzzy systems, and combination of fuzzy logic and genetic algorithms.