Fuzzy CoCo: A Cooperative-Coevolutionary Approach to Fuzzy Modeling

Carlos Andrés Peña-Reyes and Moshe Sipper

***Abstract*—Coevolutionary algorithms have received increased attention in the past few years within the domain of evolutionary computation. In this paper, we combine the search power of coevolutionary computation with the expressive power of fuzzy systems, introducing a novel algorithm, *Fuzzy CoCo: Fuzzy Cooperative Coevolution* . We demonstrate the efficacy of *Fuzzy CoCo* by applying it to a hard, real-world problem—breast cancer diagnosis—obtaining the best results to date while expending less computational effort than formerly. Analyzing our results, we derive guidelines for setting the algorithm's parameters given a (hard) problem to solve. We hope *Fuzzy CoCo* proves to be a powerful tool in the fuzzy modeler's toolkit.**

***Index Terms*—Cooperative coevolution, fuzzy modeling.**

I. Introduction

A fundamental problem in fuzzy modeling is the simultaneous design of the rules describing qualitatively a behavior and the membership functions linking this description with an observable, real-world behavior. Rules and membership functions, two essential components of fuzzy inference systems, are very different in nature. Rules are symbolic, linguistically interpretable entities, while membership functions map real values into membership values. Rules and membership functions are, however, strongly interdependent and together comprise the major part of a fuzzy inference system. For many applications, one of the major advantages of fuzzy systems is that they favor interpretability, yet finding good fuzzy systems is often an arduous task. Over the past few years evolutionary algorithms, due to their powerful search capabilities, have been employed in some stages of the fuzzy-modeling process.

This paper studies the application of an advanced evolutionary technique—cooperative coevolution—to fuzzy modeling, and is organized as follows: the remaining of this introductory section discusses the use of evolutionary computation to construct fuzzy models. Section II presents the cooperative coevolutionary paradigm. Section III describes the proposed Fuzzy CoCo methodology. Section IV then describes a sample application of Fuzzy CoCo to a hard problem: breast cancer diagnosis. The results obtained are presented in Section V. In Section VI we discuss implementation issues concerning Fuzzy CoCo and the work we envisage to extend the methodology. Finally, Section VII presents concluding remarks.

Manuscript received May 4, 2000; revised September 6, 2000.

The authors are with the Logic Systems Laboratory, Swiss Federal Institute of Technology, Lausanne, Switzerland (e-mail: Carlos.Pena@epfl.ch; Moshe.Sipper@epfl.ch).

Publisher Item Identifier S 1063-6706(01)01354-6.

A. Fuzzy Modeling

The basic structure of a fuzzy inference system includes four main components [30]: 1) A fuzzifier, which translates crisp (real-valued) inputs into fuzzy values; 2) an inference engine that applies a fuzzy reasoning mechanism to obtain a fuzzy output; 3) a defuzzifier, which translates this latter output into a crisp value; and, 4) a knowledge base, which contains both an ensemble of fuzzy rules, known as the rule base, and an ensemble of membership functions known as the database.

Fuzzy modeling is the task of identifying the parameters of a fuzzy inference system so that a desired behavior is attained [45]. With the *direct* approach a fuzzy model is constructed using knowledge from a human expert. This task becomes difficult when the available knowledge is incomplete or when the problem space is very large, thus motivating the use of *automatic* approaches to fuzzy modeling. One of the major problems in fuzzy modeling is the *curse of dimensionality*, meaning that the computation requirements grow exponentially with the number of variables.

The parameters of a fuzzy inference system can be classified into four categories [30, Table I]: logical, structural, connective, and operational. In fuzzy modeling, logical parameters are usually predefined by the designer based on experience and on problem characteristics. Structural, connective, and operational parameters may be either predefined, or obtained by synthesis or search methodologies.

Fuzzy modeling can be considered as an optimization process where part or all of the parameters of a fuzzy system constitute the search space. Generally, the search space, and thus the computational effort, grows exponentially with the number of parameters. Therefore, one can either invest more resources in the chosen search methodology, or infuse more *a priori*, expert knowledge into the system (thereby effectively reducing the search space).

B. Evolutionary Computation

The domain of evolutionary computation involves the study of the foundations and the applications of computational techniques based on the principles of natural evolution. Evolution in nature is responsible for the "design" of all living beings on earth and for the strategies they use to interact with each other. Evolutionary algorithms employ this powerful design philosophy to find solutions to hard problems.

Generally speaking, evolutionary techniques can be viewed either as search methods, or as optimization techniques. As written by Michalewicz [22]: *"Any abstract task to be accomplished can be thought of as solving a problem, which, in turn, can be perceived as a search through a space of potential solutions. Since usually we are after 'the best' solution, we can view this task as an optimization process*."

The first works on the use of evolution-inspired approaches to problem solving date back to the late 1950s [3], [4], [7], [10], [11]. Independent and almost simultaneous research conducted by Rechenberg and Schwefel on *evolution strategies* [35], [36], [38], [39], by Holland on *genetic algorithms* [14], [15], and by Fogel on *evolutionary programming* [8], [9] launched the study and the application of evolutionary techniques.

Evolutionary computation makes use of a metaphor of natural evolution. According to this metaphor, a problem plays the role of an environment wherein lives a population of individuals, each representing a possible solution to the problem. The degree of adaptation of each individual (i.e., candidate solution) to its environment is expressed by an adequacy measure known as the *fitness function*. The phenotype of each individual, i.e., the candidate solution itself, is generally encoded in some manner into its *genome* (genotype). Like evolution in nature, evolutionary algorithms potentially produce progressively better solutions to the problem. This is possible thanks to the constant introduction of new "genetic" material into the population, by applying so-called genetic operators which are the computational analogs of natural evolutionary mechanisms.

There are several types of evolutionary algorithms, among which the best known are *genetic algorithms*, *genetic programming*, *evolution strategies*, and *evolutionary programming*; though different in the specifics they are all based on the same general principles. The archetypal evolutionary algorithm proceeds as follows. An initial population of individuals, $P(0)$, is generated at random or heuristically. Every evolutionary step t, known as a *generation*, the individuals in the current population, $P(t)$, are *decoded* and *evaluated* according to some predefined quality criterion, referred to as the fitness or fitness function. Then, a subset of individuals, $P'(t)$—known as the *mating pool*—is selected to reproduce, with selection of individuals done probabilistically according to their fitness. Thus, high-fitness ("good") individuals stand a better chance of "reproducing," while low-fitness ones are more likely to disappear.

Selection alone cannot introduce any new individuals into the population, i.e., it cannot find new points in the search space. These points are generated by altering the selected population $P'(t)$ via the application of crossover (combining two or more genomes to form novel offspring) and mutation (randomly flipping bits in the genome), so as to produce a new population, $P''(t)$. Crossover tends to enable the evolutionary process to move toward "promising" regions of the search space. Mutation is introduced to prevent premature convergence to local optima, by randomly sampling new points in the search space. Finally, the new individuals $P''(t)$ are introduced into the next-generation population, $P(t+1)$; usually $P''(t)$ simply becomes $P(t+1)$. The termination condition may be specified as some fixed, maximal number of generations or as the attainment of an acceptable fitness level. Fig. 1 presents the structure of a generic evolutionary algorithm in pseudo code format.

As they combine elements of directed and stochastic search, evolutionary techniques exhibit a number of advantages over other search methods. First, they usually need a smaller amount of knowledge and fewer assumptions about the characteristics of the search space. Second, they can more easily avoid getting stuck in local optima. Finally, evolutionary algorithms strike a good balance between *exploitation* of the best solutions and *exploration* of the search space. The strength of evolutionary algorithms relies on their population-based search and on the use of the genetic mechanisms described above. The existence of a population of candidate solutions entails an inherently parallel search with the selection mechanism directing the search to the most promising regions. The crossover operator encourages the exchange of information between these search-space regions, while the mutation operator enables the exploration of new directions.

```
begin EC
   t:=0
   Initialize population P(t)
   while not done do
      Evaluate P(t)
      P'(t) := Select[P(t)]
      P''(t) := ApplyGeneticOperators[P'(t)]
      P(t+1) := Introduce[P''(t),P(t)]
      t:=t+1
   end while
end EC
```

Fig. 1. Pseudocode of an evolutionary algorithm.

The application of an evolutionary algorithm involves a number of important considerations. The first decision to take when applying such an algorithm is how to encode candidate solutions within the genome. The representation must allow for the encoding of all possible solutions while being sufficiently simple to be searched in a reasonable amount of time. Next, an appropriate fitness function must be defined for evaluating the individuals. The (usually scalar) fitness must reflect the criteria to be optimized and their relative importance. Representation and fitness are thus clearly problem-dependent in contrast to selection, crossover, and mutation, which seem *prima facie* more problem-independent. Practice has shown, however, that while standard genetic operators can be used, one often needs to tailor these to the problem at hand as well.

C. Applying Evolution to Fuzzy Modeling

Evolutionary algorithms are used to search large and often complex search spaces. They have proven worthwhile on numerous diverse problems, able to find near-optimal solutions given an adequate performance (fitness) measure. Works investigating the application of evolutionary techniques in the domain of fuzzy modeling first appeared about a decade ago [17], [26]. These focused mainly on the tuning of fuzzy inference systems involved in control tasks (e.g., cart-pole balancing, liquid level system, and spacecraft rendezvous operation). Evolutionary fuzzy modeling has since been applied to an evergrowing number of domains, branching into areas as diverse as chemistry, medicine, telecommunications, biology, and geophysics. For a detailed bibliography on evolutionary fuzzy modeling up to 1996, the reader is referred to [1], [5].

Depending on several criteria—including the available *a priori* knowledge about the system, the size of the parameter set, and the availability and completeness of input/output data, artificial evolution can be applied in different stages of

TABLE I
PARAMETER CLASSIFICATION OF FUZZY INFERENCE SYSTEMS [30]

Class	Parameters
Logical	Reasoning mechanism Fuzzy operators Membership function types Defuzzification method
Structural	Relevant variables Number of membership functions Number of rules
Connective	Antecedents of rules Consequents of rules Rule weights
Operational	Membership function values

the fuzzy-parameters search. Three of the four categories of fuzzy parameters in Table I can be used to define targets for evolutionary fuzzy modeling: structural parameters, connective parameters, and operational parameters [30].

1) *Knowledge Tuning (Operational Parameters)*: The evolutionary algorithm is used to tune the knowledge contained in the fuzzy system by finding membership-function values.
2) *Behavior Learning (Connective Parameters)*: In this approach, one supposes that extant knowledge is sufficient in order to define the membership functions. The evolutionary algorithm is used to find either the rule consequents or an adequate subset of rules to be included in the rule base.
3) *Structure Learning (Structural Parameters)*: In this approach, evolution has to deal with the simultaneous design of rules, membership functions, and structural parameters. Some methods use a fixed-length genome encoding a fixed number of fuzzy rules along with the membership-function values. Other methods use variable-length genomes to allow evolution to discover the optimal size of the rule base.

   Both behavior and structure learning can be viewed as rule-base learning processes with different levels of complexity. They can thus be assimilated within other methods from machine learning, taking advantage of experience gained in this latter domain. In the evolutionary algorithm community there are two major approaches for evolving such rule systems: the Michigan approach and the Pittsburgh approach [22]. A more recent method has been proposed specifically for fuzzy modeling: the iterative rule learning approach [13]. These three approaches are outlined below.
4) *The Michigan Approach*: Each individual represents a *single* rule. The fuzzy inference system is represented by the *entire population*. Since several rules participate in the inference process, the rules are in constant competition for the best action to be proposed and cooperate to form an efficient fuzzy system. The cooperative-competitive nature of this approach renders difficult the decision of which rules are ultimately responsible for good system behavior. It necessitates an effective credit-assignment policy to ascribe fitness values to individual rules.
5) *The Pittsburgh Approach*: The evolutionary algorithm maintains a population of candidate fuzzy systems, each individual representing an *entire* fuzzy system. Selection and genetic operators are used to produce new generations of fuzzy systems. Since evaluation is applied to the entire system, the credit assignment problem is eschewed. This approach allows to include additional optimization criteria in the fitness function, thus affording the implementation of multiobjective optimization. The main shortcoming of this approach is its computational cost, since a population of full-fledged fuzzy systems has to be evaluated each generation.
6) *The Iterative Rule Learning Approach*: As in the Michigan approach, each individual encodes a single rule. An evolutionary algorithm is used to find a single rule, thus providing a partial solution. The evolutionary algorithm is then used iteratively for the discovery of new rules, until an appropriate rule base is built. To prevent the process from finding redundant rules (i.e., rules with similar antecedents), a penalization scheme is applied each time a new rule is added. This approach combines the speed of the Michigan approach with the simplicity of fitness evaluation of the Pittsburgh approach. However, as with other incremental rule-base construction methods, it can lead to a nonoptimal partitioning of the antecedent space.

II. COOPERATIVE COEVOLUTION

Coevolution refers to the simultaneous evolution of two or more species with coupled fitness. Such coupled evolution favors the discovery of complex solutions whenever complex solutions are required [25]. Simplistically speaking, one can say that coevolving species can either compete (e.g., to obtain exclusivity on a limited resource) or cooperate (e.g., to gain access to some hard-to-attain resource). In a competitive coevolutionary algorithm the fitness of an individual is based on direct competition with individuals of other species, which in turn evolve separately in their own populations. Increased fitness of one of the species implies a diminution in the fitness of the other species. This evolutionary pressure tends to produce new strategies in the populations involved so as to maintain their chances of survival. This "arms race" ideally increases the capabilities of each species until they reach an optimum. For further details on competitive coevolution, the reader is referred to [37].

Cooperative (also called symbiotic) coevolutionary algorithms involve a number of independently evolving species which together form complex structures, well suited to solve a problem. The fitness of an individual depends on its ability to collaborate with individuals from other species. In this way, the evolutionary pressure stemming from the difficulty of the problem favors the development of cooperative strategies and individuals.

Single-population evolutionary algorithms often perform poorly—manifesting stagnation, convergence to local optima, and computational costliness—when confronted with problems presenting one or more of the following features: 1) the sought-after solution is complex, 2) the problem or its solution is clearly decomposable, 3) the genome encodes different types of values, 4) strong interdependencies among the components

of the solution, and 5) components-ordering drastically affects fitness. Cooperative coevolution addresses effectively these issues, consequently widening the range of applications of evolutionary computation.

Paredis [25] applied cooperative coevolution to problems which involved finding simultaneously the values of a solution and their adequate order. In his approach, a population of solutions coevolves alongside a population of permutations performed on the genotypes of the solutions. Moriarty [24] used a cooperative coevolutionary approach to evolve neural networks. Each individual in one species corresponds to a single hidden neuron of a neural network and its connections with the input and output layers. This population coevolves alongside a second one whose individuals encode sets of hidden neurons (i.e., individuals from the first population) forming a neural network. Potter and DeJong [33], [34] developed a model in which a number of populations explore different decompositions of the problem. Below we detail this framework as it forms the basis of our own approach.

In Potter's system, each species represents a subcomponent of a potential solution. Complete solutions are obtained by assembling *representative* members of each of the species (populations). The fitness of each individual depends on the quality of (some of) the complete solutions it participated in, thus measuring how well it cooperates to solve the problem. The evolution of each species is controlled by a separate, independent evolutionary algorithm. Fig. 2 shows the general architecture of Potter's cooperative coevolutionary framework and the way each evolutionary algorithm computes the fitness of its individuals by combining them with selected representatives from the other species. The representatives can be selected via a greedy strategy as the fittest individuals from the last generation.

Results presented by Potter and DeJong [34] show that their approach addresses adequately issues like problem decomposition and interdependencies between subcomponents. The cooperative coevolutionary approach performs as good as—and sometimes better than—single-population evolutionary algorithms. Finally, cooperative coevolution usually requires less computation than single-population evolution as the populations involved are smaller, and convergence—in terms of number of generations—is faster.

III. Fuzzy CoCo

Fuzzy CoCo is a Cooperative Coevolutionary approach to fuzzy modeling where two coevolving species are defined: database (membership functions) and rule base. This approach is based primarily on the framework defined by Potter and DeJong ([33] and [34], Sec. II).

A fuzzy modeling process need usually deal with the simultaneous search for operational and connective parameters (Table I). These parameters provide an almost complete definition of the linguistic knowledge describing the behavior of a fuzzy system and the values mapping this symbolic description into a real-valued world (a complete definition also requires structural parameters, such as relevant variables and number of rules). Thus, fuzzy modeling can be thought of as two separate but intertwined search processes: 1) the search for the membership functions (i.e., operational parameters) that define the fuzzy variables and 2) the search for the rules (i.e., connective parameters) used to perform the inference.

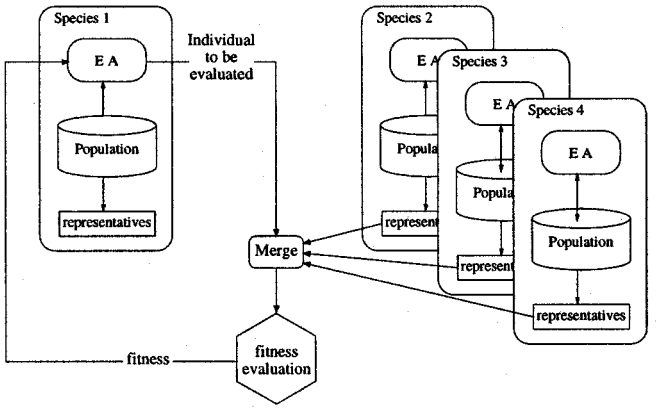

Fig. 2. Potter's cooperative coevolutionary system. The figure shows the evolutionary process from the perspective of Species 1. The individual being evaluated is combined with one or more *representatives* of the other species so as to construct several solutions which are tested on the problem. The individual's fitness depends on the quality of these solutions.

A. Description of Fuzzy CoCo

Fuzzy modeling presents several features discussed in Section II which justify the application of cooperative coevolution.

1) The required solutions can be very complex, since fuzzy systems with a few dozen variables may call for hundreds of parameters to be defined.
2) The proposed solution—a fuzzy inference system—can be decomposed into two distinct components: rules and membership functions.
3) Membership functions are represented by continuous, real values, while rules are represented by discrete, symbolic values.
4) These two components are interdependent because the membership functions defined by the first group of values are indexed by the second group (rules).

Consequently, in Fuzzy CoCo, the fuzzy modeling problem is solved by two coevolving, cooperating species. Individuals of the first species encode values which define completely all the membership functions for all the variables of the system.

> Individuals of the second species define a set of rules of the form:
> **if** (v_1 **is** A_1) **and . . . and** (v_n **is** A_n), **then** (*output* **is** C),
> where the term A_i indicates which one of the linguistic labels of fuzzy variable v is used by the rule. For example, a valid rule could contain the expression:
> **if . . . and** (*Temperature* **is** *Warm*) **and . . . then . . .,**
> which includes the membership function *Warm* whose defining parameters are contained in the first species.

The two evolutionary algorithms used to control the evolution of the two populations are instances of a simple genetic algorithm [44]. Fig. 3 presents the Fuzzy CoCo algorithm in pseudocode format. The genetic algorithms apply fitness-proportionate selection to choose the mating pool (essentially, probabilistic selection according to fitness) and apply an elitist

```
begin Fuzzy CoCo
  g:=0
  for each species S
    Initialize populations P_S(0)
    Evaluate population P_S(0)
  end for
  while not done do
    for each species S
      g:=g+1
      E_S(g) = elite-select P_S(g − 1)
      P'_S(g) = select P_S(g − 1)
      P''_S(g) = crossover P'_S(g)
      P'''_S(g) = mutate P''_S(g)
      P_S(g) = P'''_s(g) + E_S(g)
      Evaluate population P_S(g)
    end for
  end while
end GA
```

Fig. 3. Pseudocode of Fuzzy CoCo. Two species coevolve in Fuzzy CoCo: membership functions and rules. The elitism strategy extracts E_S individuals to be reinserted into the population after evolutionary operators have been applied (i.e., selection, crossover, and mutation). Selection results in a reduced population $P'_S(g)$ (usually, the size of $P'_S(g)$ is $\|P'_S\| = \|P_S\| - \|E_S\|$). The line "Evaluate population $P_S(g)$" is elaborated in Fig. 4.

strategy with an elitism rate Er to allow some of the best individuals to survive into the next generation. Standard crossover and mutation operators are applied [22]. Crossover between two genomes is performed with probability P_c by selecting at random (with uniform probability) a single crossover point and exchanging the subsequent parts to form two new offspring; if no crossover takes place (with probability $1 - P_c$) the two offspring are exact copies of their parents. Mutation involves flipping bits in the genome with probability P_m per bit.

We introduced elitism to avoid the divergent behavior of Fuzzy CoCo, observed in our preliminary trial runs. Nonelitist versions of Fuzzy CoCo often tend to lose the genetic information of good individuals found during evolution, consequently producing populations with mediocre individuals scattered throughout the search space. This is probably due to the relatively small size of the populations which renders difficult the preservation of good solutions while exploring the search space. The introduction of simple elitism produces an undesirable effect on Fuzzy CoCo's performance: populations converge prematurely even with reduced values of the elitism rate E_r. To offset this effect without losing the advantages of elitism, it is necessary to increase the mutation probability P_m by an order of magnitude (Table III) so as to improve the exploration capabilities of the algorithm. As the dispersion effect is less important when Fuzzy CoCo is allowed to manage relatively large populations, the values of both E_r and P_m should be reduced in such case. The condition under which the algorithm terminates is usually satisfied either when a given threshold fitness is attained, or when the maximum number of generations $G_{\max}$ is reached.

A more detailed view of the fitness evaluation process is depicted in Fig. 4. An individual undergoing fitness evaluation establishes cooperation with one or more representatives of the other species, i.e., it is combined with individuals from the other

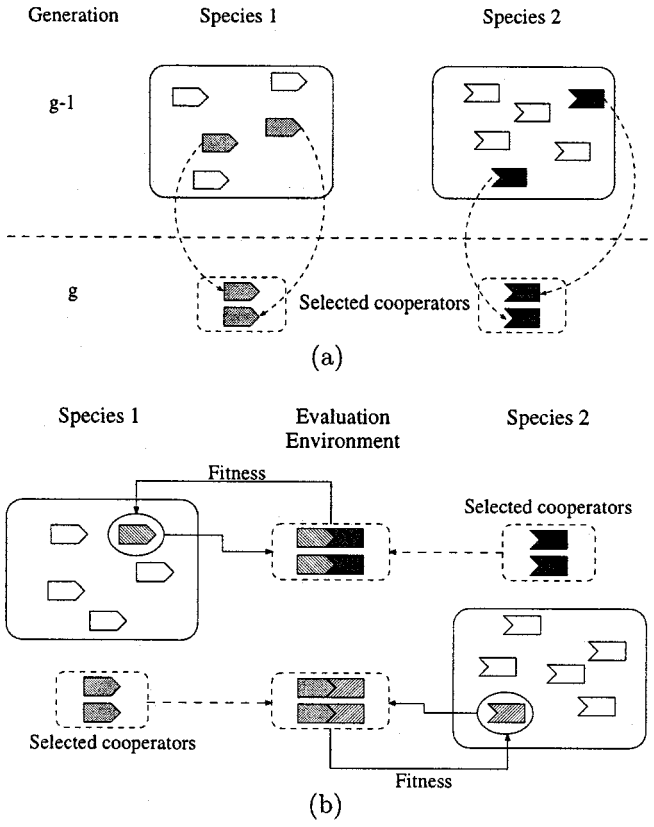

Fig. 4. Fitness evaluation in Fuzzy CoCo. (a) Several individuals from generation $g - 1$ of each species are selected according to their fitness to be the representatives of their species during generation g; these representatives are called "cooperators." (b) During the evaluation stage of generation g (after selection, crossover, and mutation—see Fig. 3), individuals are combined with the selected cooperators of the other species to construct fuzzy systems. These systems are then evaluated on the problem domain and serve as a basis for assigning the final fitness to the individual being evaluated.

species to construct fuzzy systems. The fitness value assigned to the individual depends on the performance of the fuzzy systems it participated in (specifically, either the average or the maximal value).

Representatives, or *cooperators*, are selected both fitness-proportionally and randomly from the last generation in which they were already assigned a fitness value (see Fig. 3). In Fuzzy CoCo, N_{cf} cooperators are selected probabilistically according to their fitness, favoring the fittest individuals, thus boosting the exploitation of known good solutions. The other N_{cr} cooperators are selected randomly from the population to represent the diversity of the species, maintaining in this way exploration of the search space.

B. A Stepwise Guide to Applying Fuzzy CoCo

Applying Fuzzy CoCo requires the definition of parameters of its two main components: 1) the fuzzy system and 2) the cooperative coevolutionary algorithm.

1) Fuzzy-system parameters are defined through the following four-level procedure:
   a) Define the logical parameters. As noted in Section I, logical parameters are predefined by the designer

based on experience and on problem characteristics.

b) Choose the structural parameters.
- A set of relevant variables should be defined. Usually, this set includes *all* the available variables as Fuzzy CoCo can automatically reduce their number.
- Fuzzy CoCo requires predefining the number of membership functions. This issue is discussed in Section VI.
- The number of rules is fixed (by the designer) for a given Fuzzy CoCo run. A discussion about this number is presented in Section VI.

c) Encode the connective parameters into the rules genome. The rules may be either complete (i.e., containing at least one antecedent for each variable) or incomplete [20]. The antecedents in rules may be connected merely by the "and" operator, or may contain also "or" and "not" operators. Fuzzy CoCo thus offers the designer the freedom of choosing any type of rule, given that there exists a proper way to encode it.

d) Encode the operational parameters into the membership-function genome. The membership functions can be of arbitrary form. The usual criteria for choosing the membership functions in fuzzy modeling are also applicable in the context of Fuzzy CoCo. The only condition imposed by Fuzzy CoCo is that all possible lables implied by the rules species should be defined.

Note that Fuzzy CoCo is a methodology for improving the performance and speed of the fuzzy modeling process. It cannot correct on its own wrong decisions made during the definition of the fuzzy-system parameters. Thus, the designer need posses knowledge of the problem or a good evaluation heuristics.

2) Once both genomes, rules and membership-functions are encoded, the coevolutionary parameters presented below must be set according to a number of criteria; the two most important being computational costliness and exploration-exploitation tradeoff. A discussion concerning the qualitative relations among the ranges of the coevolutionary parameters is presented in Section VI.

a) Maximum number of generations $G_{\max}$. Due to the speed gain offered by Fuzzy CoCo, the value $G_{\max}$, related directly to computational costliness, can be up to five times smaller than single-population algorithms.

b) Population size $||P_S||$. Fuzzy CoCo requires smaller populations than the simple genetic algorithm-based approach. Typically, 50 to 80 percent smaller. This markedly reduces the computational cost.

c) Number of "fit" cooperators N_{cf}. Typical values range from 1 to 3 fitness-proportionally selected cooperators. N_{cf} affects directly both the exploitation and the computational cost of the algorithm.

d) Number of random cooperators N_{cr}. Typical values range from 0 to 4 randomly selected cooperators. N_{cr} affects directly both the exploration and the computational cost of the algorithm.

e) Crossover probability P_c. There is no special consideration concerning the value of P_c in Fuzzy CoCo. Standard values—0.5 to 1—are used [23].

f) Mutation probability P_m. As discussed in Section III-A, due to an exploration-exploitation tradeoff with the elitism rate, P_m values in Fuzzy CoCo are usually an order of magnitude higher than in simple genetic algorithms. While the value of P_m proposed by Potter and DeJong ($P_m = 1/length(genome)$) [34] can be applied with relatively large populations, it has to be increased up to 10 times when Fuzzy CoCo is applied with small populations.

g) Elitism Rate E_r. Typical values for E_r are between 0.1 and 0.6, where larger values are required in systems with few rules and small populations. E_r encourages exploitation of solutions found.

IV. Applying Fuzzy CoCo to Breast Cancer Diagnosis

The Wisconsin breast cancer diagnosis (WBCD) problem involves classifying a presented case as to whether it is benign or malignant. It admits a relatively high number of variables and consequently a large search space. The WBCD database [21] consists of nine visually assessed characteristics obtained from fine needle aspirates[1] of breast masses, each of which is ultimately represented as an integer value between 1 and 10. The measured variables are as follows: 1) Clump Thickness (v_1); 2) Uniformity of Cell Size (v_2); 3) Uniformity of Cell Shape (v_3); 4) Marginal Adhesion (v_4); 5) Single Epithelial Cell Size (v_5); 6) Bare Nuclei (v_6); 7) Bland Chromatin (v_7); 8) Normal Nucleoli (v_8); and 9) Mitosis (v_9).

The diagnostics in the WBCD database were furnished by specialists in the field. The database itself consists of 683 cases, with each entry representing the classification for a certain ensemble of measured values:

$case$	v_1	v_2	v_3	$\cdots$	v_9	$diagnostic$
1	5	1	1	$\cdots$	1	$Benign$
2	5	4	4	$\cdots$	1	$Benign$
:	:	:	:		:	:
683	4	8	8	$\cdots$	1	$Malignant$

There are several studies based on this database. Among them, researchers having interpretability of the diagnostic as a prior objective, have applied the method of extracting Boolean rules from neural networks [40], [42], [43]. Our own work on the evolution of fuzzy rules for the WBCD problem has shown that it is possible to obtain diagnostic systems exhibiting high performance, coupled with interpretability and a confidence measure [28]–[30]. In our previous work we used a simple genetic algorithm rather than Fuzzy CoCo.

[1]Fine needle aspiration is an outpatient procedure that involves using a small-gauge needle to extract fluid directly from a breast mass [19].

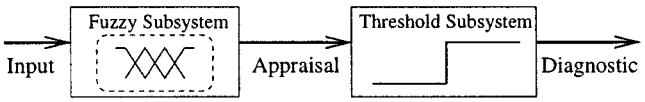

Fig. 5. Proposed diagnosis system.

The solution scheme we propose for the WBCD problem is depicted in Fig. 5. It consists of a fuzzy system and a threshold unit. The fuzzy system computes a continuous appraisal value of the malignancy of a case, based on the input values. The threshold unit then outputs a *benign* or *malignant* diagnostic according to the fuzzy system's output.

Our previous knowledge about the WBCD problem represents valuable information to be used for our choice of fuzzy parameters. When defining our setup we took into consideration the following three results concerning the composition of potential high-performance systems: 1) small number of rules; 2) small number of variables; and 3) monotonicity of the input variables [30]. Some fuzzy models forgo interpretability in the interest of improved performance. Where medical diagnosis is concerned [31], interpretability—also called linguistic integrity—is the major advantage of fuzzy systems. This motivated us to take into account the following five semantic criteria, defining constraints on the fuzzy parameters [30]: 1) distinguishability; 2) justifiable number of elements; 3) coverage; 4) normalization; and 5) orthogonality.

Referring to Table I, and taking into account the above criteria, we delineate below the fuzzy-system setup:

1) Logical parameters: singleton-type fuzzy systems; min-max fuzzy operators; orthogonal, trapezoidal input membership functions; weighted-average defuzzification.
2) Structural parameters: two input membership functions (*Low* and *High*); two output singletons (*benign* and *malignant*); a user-configurable number of rules. The relevant variables are one of Fuzzy CoCo's objectives.
3) Connective parameters: the antecedents and the consequent of the rules are searched by Fuzzy CoCo. The algorithm also searches for the consequent of the default rule which plays the role of an *else* condition. All rules have unitary weight.
4) Operational parameters: the input membership function values are to be found by Fuzzy CoCo. For the output singletons we used the values 2 and 4, which are the values used in the WBCD database for designing *benign* and *malignant*, respectively (note that these two values are represented in the genome by a single bit as presented below).

Fuzzy CoCo is thus used to search for four parameters: 1) input membership function values, 2) relevant input variables, 3) antecedents, and 4) consequents of rules. These search goals are more ambitious than those defined in our previous work [28]–[30], as the consequence of rules are added to the search space. The genomes of the two species are constructed as follows:

Species 1: Membership functions. There are nine variables (v_1–v_9), each with two parameters, P and d, defining the start point and the length of the membership-function edges, respectively (Fig. 6).

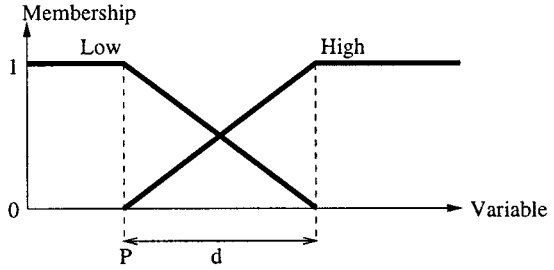

Fig. 6. Fuzzy variables for the WBCD problem. All the variables have two labels: **Low** and **High**, and orthogonal membership functions. P and d define the start point and the length of membership-function edges, respectively.

TABLE II
GENCODING ENCODING OF PARAMETERS FOR BOTH SPECIES. GENOME LENGTH FOR MEMBERSHIP FUNCTIONS IS 54 BITS.GENOME LENGTH FOR RULES IS $19 \times N_r + 1$, WHERE N_R DENOTES THE NUMBER OF RULES. NOTE: IN AN EARLY WORK, PARAMETERS P AND d WERE ALLOWED TO TAKE ON VALUES IN THE RANGE AND $\{1, 2, ..., 10\}$ AND $\{0, 1, ..., 7\}$, RESPECTIVELY [28]. IN LATER WORKS THESE TWO PARAMETERS WERE RESTRICTED TO THE VALUES PRESENTED BELOW.

Species 1: Membership functions

Parameter	Values	Bits	Qty	Total bits
P	{1,2,...,8}	3	9	27
d	{1,2,...,8}	3	9	27
	Total			54

Species 2: Rules

Parameter	Values	Bits	Qty	Total bits
A	{0,1,2,3}	2	$9 \times N_r$	$18 \times N_r$
C	{1,2}	1	$N_r + 1$	$N_r + 1$
	Total			$19 \times N_r + 1$

Species 2: Rules. The ith rule has the form:

if (v_1 **is** A_1^i) **and . . . and** (v_9 **is** A_9^i), **then** (*output* **is** C^i),

A_j^i can take on the values: 1 (*Low*), 2 (*High*), or 0 or 3 (*Other*). C^i bit can take on the values: 0 (*Benign*) or 1 (*Malignant*). Relevant variables are searched for implicitly by letting the algorithm choose nonexistent membership functions (0 or 3) as valid antecedents. In such a case, the respective variable is considered irrelevant.

Table II delineates the parameters encoding for both species' genomes, which together describe an entire fuzzy system. Note that in our previous work both membership functions and rules were encoded in the same genome, i.e., there was only one species.

To evolve the fuzzy inference system, we applied a Fuzzy CoCo algorithm with the same evolutionary parameters for both species. Values and ranges of values used for these parameters were defined according to preliminary tests performed on benchmark problems (mostly function-optimization problems found in Potter [33]). Table III delineates these values (Section VI presents a brief discussion concerning the criteria to define the best combinations of parameter values). The algorithm terminates when the maximum number of generations, $G_{\max}$, is reached (we set $G_{\max} = 1000 + 100 \times N_r$, i.e., dependent on the number of rules used in the run), or when the increase in fitness of the best individual over five successive generations falls below a certain threshold (10^{-4} in our experiments).

TABLE III
FUZZY COCO SET-UP FOR THE WBCD PROBLEM

Parameter	Values
Population size $\|\|Ps\|\|$	30–90
Maximum generations G_{max}	$1000 + 100N_r$
Crossover probability P_c	1
Mutation probability P_m	0.02–0.3
Elitism rate E_r	0.1–0.6
"Fit" cooperators N_{cf}	1
Random cooperators N_{cr}	1–4

Our fitness function combines two criteria: 1) F_c—classification performance, computed as the percentage of cases correctly classified and 2) F_v—the maximum number of variables in the longest rule. The fitness function is given by $F = F_c - \alpha F_v$, where $\alpha = 0.0015$. F_c, the percentage of correctly diagnosed cases, is the most important measure of performance. F_v measures the linguistic integrity (interpretability), penalizing systems with a large number of variables in their rules. The value α was calculated to allow F_v to occasion a fitness difference only among systems exhibiting similar classification performance. The fitness value assigned to an individual is the maximum of the fitness values obtained by the N_c fuzzy systems it participated in (where $N_c = N_{cf} + N_{cr}$).

We stated earlier that cooperative coevolution reduces the computational cost of the search process. In order to measure this cost we calculate the maximum number of fuzzy-system evaluations performed by a single run of Fuzzy CoCo. Each generation, the $||Ps||$ individuals of each population are evaluated N_c times (where $N_c = N_{cf} + N_{cr}$). The total number of fuzzy-system evaluations per run is thus $2 \times G_{\max} \times ||Ps|| \times N_c$. This value ranged from 5.28×10^5 evaluations for a one-rule system search, up to 8.16×10^5 evaluations for a seven-rule system (using typical parameter values: $||Ps|| = 80$, $N_{cf} = 1$, and $N_{cr} = 2$). The number of fuzzy-system evaluations required by our single-population approach was, on the average, 5×10^5 for a one-rule system and 11×10^5 for a seven-rule system [30]. This shows that Fuzzy CoCo produces markedly better results using similar computational resources.

V. RESULTS

A total of 495 evolutionary runs were performed, all of which found systems whose classification performance exceeds 96.7%. In particular, considering the best individual per run (i.e., the evolved system with the highest classification success rate), 241 runs led to a fuzzy system whose performance exceeds 98.0%, and of these, 81 runs found systems whose performance exceeds 98.5%.[2]

Table IV compares our best systems with the top systems obtained in our previous work [30] and with the systems obtained by Setiono's NeuroRule approach [41] (note that the results presented by these two works were the best reported to date for genetic-fuzzy and neuro-Boolean rule systems, respectively, and that they were compared with other previous approaches such as [40], [42], [43]). The evolved fuzzy systems described in this paper can be seen to surpass those obtained by other approaches in terms of performance, while still containing simple, interpretable rules. As shown in Table IV, we obtained higher-performance systems for all rule-base sizes but one, i.e., from two-rule systems all the way up to seven-rule systems, while all our one-rule systems perform as good as the best system reported by Setiono.

We next describe two of our top-performance systems, which serve to exemplify the solutions found by Fuzzy CoCo. The first system, delineated in Fig. 7, presents the highest classification performance evolved to date. It consists of seven rules (note that the `else` condition is not counted as an active rule) with the

[2]Note: In our earlier work, involving a standard genetic algorithm applied to the WBCD problem [30], we used cross-validation methods (basically, we performed three sets of experiments, with different repartitions of the 683-case WBCD database into training cases and test cases. We observed that small fuzzy systems (i.e., with a small number of rules) are relatively little prone to overfitting (at least for this problem). Thus, we decided to eschew a cross-validation study herein, opting to use the entire WBCD database in all experiments.

TABLE IV
COMPARISON OF THE SYSTEMS EVOLVED BY FUZZY COCO WITH THE TOP SYSTEMS OBTAINED USING SINGLE-POPULATION EVOLUTION [30] AND WITH THOSE OBTAINED BY SETIONO'S NEURORULE APPROACH [41]. SHOWN BELOW ARE THE CLASSIFICATION PERFORMANCE VALUES OF THE TOP SYSTEMS OBTAINED BY THESE APPROACHES, ALONG WITH THE NUMBER OF VARIABLES OF THE LONGEST RULE IN PARENTHESES. RESULTS ARE DIVIDED INTO SEVEN CLASSES, IN ACCORDANCE WITH THE NUMBER OF RULES PER SYSTEM, GOING FROM ONE-RULE SYSTEMS TO SEVEN-RULE ONES

Rules per system	Neuro-Rule [41]	Single population GA [30]	Fuzzy CoCo	
	best	best	average	best
1	97.36% (4)	97.07% (4)	97.36% (4)	97.36% (4)
2	–	97.36% (4)	97.73% (3.9)	98.54% (5)
3	98.10% (4)	97.80% (6)	97.91% (4.4)	98.54% (4)
4	–	98.24% (7)	98.12% (4.2)	98.68% (5)
5	98.24% (5)	97.95% (7)	98.18% (4.6)	98.83% (5)
6	–	98.10% (9)	98.18% (4.3)	98.83% (5)
7	–	97.95% (8)	98.25% (4.7)	98.98% (5)

Database

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
P	2	1	1	1	6	1	3	5	2
d	7	8	4	8	1	4	8	4	1

Rule base

Rule 1	**if** (v_1 **is** *Low*) **and** (v_3 **is** *Low*) **then** (*output* **is** *benign*)
Rule 2	**if** (v_4 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** (*output* **is** *benign*)
Rule 3	**if** (v_1 **is** *Low*) **and** (v_3 **is** *High*) **and** (v_5 **is** *High*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** (*output* **is** *benign*)
Rule 4	**if** (v_1 **is** *Low*) **and** (v_2 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_5 **is** *Low*) **and** (v_8 **is** *High*) **then** (*output* **is** *benign*)
Rule 5	**if** (v_2 **is** *High*) **and** (v_4 **is** *High*) **then** (*output* **is** *malignant*)
Rule 6	**if** (v_1 **is** *High*) **and** (v_3 **is** *High*) **and** (v_6 **is** *High*) **and** (v_7 **is** *High*) **then** (*output* **is** *malignant*)
Rule 7	**if** (v_2 **is** *High*) **and** (v_3 **is** *High*) **and** (v_4 **is** *Low*) **and** (v_5 **is** *Low*) **and** (v_7 **is** *High*) **then** (*output* **is** *malignant*)
Default	**else** (*output* **is** *malignant*)

Fig. 7. The best evolved fuzzy diagnostic system with seven rules. It exhibits an overall classification rate of 98.98% and its longest rule includes five variables.

Database

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
P	3		1	3	4	5		7	2
d	8		3	1	2	2		4	1

Rule base

Rule 1	**if** (v_1 **is** *Low*) **and** (v_3 **is** *Low*) **and** (v_5 **is** *Low*) **then** (*output* **is** *benign*)
Rule 2	**if** (v_1 **is** *Low*) **and** (v_4 **is** *Low*) **and** (v_6 **is** *Low*) **and** (v_8 **is** *Low*) **and** (v_9 **is** *Low*) **then** (*output* **is** *benign*)
Default	**else** (*output* **is** *malignant*)

Fig. 8. The best evolved, fuzzy diagnostic system with two rules. It exhibits an overall classification rate of 98.54%, and a maximum of five variables in the longest rule.

longest rule including 5 variables. This system obtains an overall classification rate (i.e., over the entire database) of 98.98%.

In addition to the above seven-rule system, evolution found systems with between two and six rules exhibiting excellent classification performance, i.e., higher than 98.5% (Table IV). Among these systems, we consider as the most interesting the system with the smallest number of conditions (i.e., the total number of variables in the rules). Fig. 8 presents one such two-rule system, containing a total of eight conditions, and which obtains an overall classification rate of 98.54%; its longest rule has five variables.

The improvement attained by Fuzzy CoCo, while seemingly slight (1–2%) is in fact quite significant. A 1% improvement implies seven additional cases which are classified correctly. At the performance rates in question (above 97%) every additional case is hard-won. Indeed, try as we did with the simple genetic algorithm [30]—tuning parameters and tweaking the setup—we arrived at a performance impasse. Fuzzy CoCo, however, readily churned out better-performance systems that were able to classify a significant number of additional cases; moreover, these systems were evolved in less time.

VI. Discussion and Future Work

A. Fuzzy CoCo and Its Application

We propose Fuzzy CoCo as a methodology for modeling fuzzy systems and have conceived it to allow a high degree of freedom in the type of fuzzy systems it can design. Fuzzy CoCo can be used to model Mamdani-type, Takagi–Sugeno–Kang-type, and singleton-type fuzzy models [12]. The rules can contain an arbitrary number of antecedents (i.e., zero, one, or many) for the same variable. The designer is free to choose the type of membership functions used for each variable and the way they are parameterized. The membership functions can be defined either as shared by all fuzzy rules, or per-rule; the former (used by us in Section IV) improves interpretability. If membership functions are defined independently for each rule (thus partitioning the input space in a scattered way [16]), better numeric precision might be attained at the cost of interpretability—a tradeoff at the user's disposal. Fuzzy CoCo is thus highly general and generic.

To better exploit the features of Fuzzy CoCo, the following considerations should be taken into account before applying it to a given problem.

1) Rule-base compactness. The existence of a rules species in Fuzzy CoCo favors the design of systems containing a small subset of all the possible rules. The size of this subset (i.e., the number of rules) is predefined by the user.
2) Known number of membership functions. In order to encode the genomes of the rules and membership functions species, it is necessary to predefine the number of membership functions for each variable. This number can be set to a relatively high value and Fuzzy CoCo will seek automatically an efficient subset of the membership functions.
3) Availability of training data. Since Fuzzy CoCo is a heuristic search method, it needs a sufficient amount of training data in order to measure fitness.

B. Parameter Selection

Fuzzy CoCo requires the designer to define a number of interdependent parameters (i.e., number of rules, population size, number of cooperators, elitism rate, and mutation and crossover probabilities). Although Section III-B presented some criteria to define these parameters, we further discuss this issue below.

1) For a given problem, searching for a compact fuzzy rule base is harder than searching for a slightly bigger system, even if the genome is larger for this latter search. The reason for this apparent contradiction is that a less compact fuzzy system (i.e., with more rules) can cover a larger part of the problem space. However, if evolution seeks too many rules, the fitness landscape becomes too "flat" (intuitively, an abundance of low-performance hills, rather than a few high-performance mountains), thus rendering the search more difficult. For each problem there exists a range of ideal rule-base sizes (for the WBCD problem, it seems to be between four and seven rules).
2) As noted in Section III-B, the number of selected fit and random cooperators (N_{cf} and N_{cr}), are parameters related, respectively, with exploitation and exploration capabilities of Fuzzy CoCo. However, in contrast to elitism and mutation, these two parameters do not affect directly the genetic pool of the next generation (only affecting it indirectly by permitting Fuzzy CoCo to evaluate more accurately individual fitness).
3) As the number of fuzzy-system evaluations per generation depends on the size of the (two) populations $||Ps||$ and on the number of cooperators N_c, these two parameters have to be set according to the available computing resources.
4) Based on the simulations carried out on the WBCD problem, we have derived some qualitative relationships between various parameters of Fuzzy CoCo. These are given in Table V.

C. Present and Future of Fuzzy CoCo

Our promising results have incited us to engage in further investigation of this approach. We are currently pursuing two avenues of research: 1) application of Fuzzy CoCo to more complex diagnosis problems and 2) improving and expanding upon the methodology presented herein. Our goal is to provide an

TABLE V
QUALITATIVE RELATIONSHIPS BETWEEN FUZZY COCO PARAMETERS. DELINEATED BELOW ARE CRITERIA TO GUIDE THE CHOICE OF THE NUMBER OF COOPERATORS N_c, THE MUTATION PROBABILITY P_m (EXPRESSED AS A FUNCTION OF THE GENOME'S LENGTH, L_g), AND THE ELITISM RATE E_r, ALL THREE AS A FUNCTION OF THE DESIRED SIZE OF THE RULE BASE (EXPRESSED IN TERM OF NUMBER OF RULES), AND THE DESIRED POPULATION SIZE (EXPRESSED IN TERM OF PERCENTAGE OF THE TYPICAL POPULATION SIZE OF A SINGLE-POPULATION ALGORITHM; PREVIOUS WORK HAS SHOWN THAT FOR THE WBCD PROBLEM THIS TYPICAL SIZE IS APPROXIMATELY 200). FOR EXAMPLE, IF THE USER WISHES TO EMPLOY A LARGE POPULATION WITH FEW RULES THEN SHE SHOULD SET N_c, P_m, AND E_r TO VALUES WITHIN THE RANGES SPECIFIED IN THE UPPER-RIGHT QUADRANT OF THE TABLE. NOTE THAT THE LINGUISTIC LABELS USED IN THE TABLE CAN BE GIVEN A FUZZY INTERPRETATION, i.e., ONE CAN DEFINE MEMBERSHIP FUNCTIONS FOR LABELS SUCH AS "MANY," "SOME," AND "FEW"

Number of rules		Population size	
		Small (20%–40%)	Large (40%–70%)
Few (2–4)	N_c	many (4–6)	some or many (3–6)
	P_m	large $((8-10)/L_g)$	medium $((4-6)/L_g)$
	E_r	high (0.4–0.6)	medium–high (0.3–0.6)
Many (4–10)	N_c	few or some (2–5)	few (1–3)
	P_m	small or medium $((2-5)/L_g)$	small $((1-2)/L_g)$
	E_r	medium (0.2–0.4)	low (0.1–0.2)

approach for automatically producing high-performance, interpretable fuzzy systems for real-world problems. We have recently applied Fuzzy CoCo to the problem of iris classification [6], [21], obtaining excellent results [27], [].

As regards the application of Fuzzy CoCo, and given that it is a fuzzy modeling technique, possible applications of this approach should be found in domains where the main characteristics of fuzzy systems—linguistic interpretability, processing of imprecise and uncertain information, nonlinear and complex variable mapping—are desired. Moreover, Fuzzy CoCo is particularly well suited for problems where a reduced set of (small) rules is preferred. Thus, some of the possible applications are medical diagnostic and prognostic systems, data mining and knowledge discovery in databases, financial data managing, pattern recognition, and time-series analysis and forecasting.

Concerning the expansion of the methodology, we have two short-term goals:

1) Study the tuning of the genetic-algorithm parameters according to each species characteristics (e.g., encoding schemes, elitism rates, or mutation probabilities).
2) Explore the application of different evolutionary algorithms for each species (e.g., evolution strategies for the evolution of membership functions).

In the long term we plan to test some novel ideas that could improve Fuzzy CoCo:

1) Coevolution of N_r+1 species, one species for each of the N_r rules in addition to the membership-function species.
2) Coexistence of several Fuzzy CoCo instances (each one set to evolve systems with a different number of rules), permitting migration of individuals among them so as to increase the exploration and the diversity of the search process.
3) Apply the strategy of rising and death of species proposed by Potter and DeJong [34] in order to evolve systems with variable numbers of rules and membership functions.

VII. CONCLUDING REMARKS

We presented Fuzzy CoCo, a novel, cooperative coevolutionary approach to fuzzy modeling. We applied Fuzzy CoCo to the WBCD problem, comparing it with other approaches applied to the same problem. Our coevolved systems attained higher classification performance and required less computation to obtain the diagnostic systems. We then discussed the merits of the approach and its application. We hope Fuzzy CoCo proves to be a powerful tool in the fuzzy modeler's toolkit.

REFERENCES

[1] J. T. Alander, "An indexed bibliography of genetic algorithms with fuzzy logic," in *Fuzzy Evolutionary Computation*, P. W. Pedrycz, Ed. Norwell, MA: Kluwer , 1997, pp. 299–318.

[2] C. L. Blake and C. J. Merz, "ftp://ftp.ics.uci.edu/pub/machine-learning-databases/iris/," *UCI Repository Mach. Learning Databases*, 1998.

[3] G. E. P. Box, "Evolutionary operation: A method for increasing industrial productivity," *Appl. Statist.*, vol. 6, no. 2, pp. 81–101, 1957.

[4] G. E. P. Box and J. S. Hunter, "Condensed calculations for evolutionary operation programs," *Technometrics*, vol. 1, pp. 77–95, 1959.

[5] O. Cordón, F. Herrera, and M. Lozano, "On the combination of fuzzy logic and evolutionary computation: A short review and bibliography," in *Fuzzy Evolutionary Computation*, W. Pedrycz, Ed. Norwell, MA: Kluwer , 1997, pp. 33–56.

[6] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.

[7] D. B. Fogel, Ed., *Evolutionary Computation: The Fossil Record*. Piscataway, NJ: IEEE Press, 1998.

[8] L. J. Fogel, "Autonomous automata," *Industrial Research*, vol. 4, pp. 14–19, 1962.

[9] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*. New York: Wiley , 1966.

[10] R. M. Friedberg, "A learning machine: I.," *IBM J. Res. Develop.*, vol. 2, pp. 2–13, 1958.

[11] R. M. Friedberg, B. Dunham, and J. H. North, "A learning machine. II.," *IBM J. Res. Develop.*, vol. 3, pp. 282–287, 1959.

[12] H. Hellendoorn and D. Driankov, Eds., *Fuzzy Model Identification*. Berlin: Springer-Verlag, 1997.

[13] F. Herrera, M. Lozano, and J. L. Verdegay, "Generating fuzzy rules from examples using genetic algorithms," in *Fuzzy Logic and Soft Computing*, B. Bouchon-Meunier, R. R. Yager, and L. A. Zadeh, Eds, Singapore: World Scientific, 1995, pp. 11–20.

[14] J. H. Holland, "Outline for a logical theory of adaptive systems," *J. ACM*, vol. 9, no. 3, pp. 297–314, July 1962.

[15] ——, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. of Michigan Press, 1975.

[16] J.-S. R. Jang and C.-T. Sun, "Neuro-fuzzy modeling and control," *Proc. IEEE*, vol. 83, pp. 378–406, March 1995.

[17] C. L. Karr, "Genetic algorithms for fuzzy controllers," *AI Expert*, vol. 6, no. 2, pp. 26–33, February 1991.

[18] C. L. Karr, L. M. Freeman, and D. L. Meredith, "Improved fuzzy process control of spacecraft terminal rendezvous using a genetic algorithm," in *Proc. Intell. Control Adaptive Syst. Conf.*, G. Rodriguez, Ed: SPIE, Feb. 1990, vol. 1196, pp. 274–288.

[19] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast cancer diagnosis and prognosis via linear programming," Univ. of Wisconsin Press, Madison, WI, Math. Programming Tech. Rep. 94-10, 1994.

[20] J. M. Mendel, "Fuzzy logic systems for engineering: A tutorial," *Proc. IEEE*, vol. 83, pp. 345–377, Mar. 1995.

[21] C. J. Merz and P. M. Murphy, "fpt://ftp.ics.uci.edu/pub/machine-learning-databases/breast-cancer-wisconsin/," *UCI repository mach. learning databases*, 1996.

[22] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, third ed. Heidelberg: Springer-Verlag, 1996.

[23] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[24] D. E. Moriarty, "Symbiotic evolution of neural networks in sequential decision tasks," The University of Texas at Austin, Jan. 1997.

[25] J. Paredis, "Coevolutionary computation," *Artificial Life*, vol. 2, pp. 355–375, 1995.

[26] C. L. Karr, L. M. Freeman, and D. L. Meredith, "The Flowering of Fuzzy CoCo: Evolving fuzzy iris calssifiers," in *Proc. 5th Int. Conf. Artificial Neural Networks and Genetic Algorithms*, V. Kurková, N. C. Steele, R. Neruda, and M Kárný, Eds. New York: Springer-Verlag, pp. 304–307.
[27] ——, "Fuzzy modeling by Fuzzy CoCo,", submitted for publication.
[28] ——, "Evolving fuzzy rules for breast cancer diagnosis," in *Proc.1998 Int. Symp. Nonlinear Theory Applicat. (NOLTA'98)*. Lausanne: Presses Polytechniques et Universitaires Romandes, 1998, vol. 2, pp. 369–372.
[29] ——, "Designing breast cancer diagnostic systems via a hybrid fuzzy-genetic methodology," in *1999 Proc. IEEE Int. Fuzzy Syst. Conf.* : IEEE Neural Network Council, 1999, vol. 1, pp. 135–139.
[30] ——, "A fuzzy-genetic approach to breast cancer diagnosis," *Artif. Intell. Medicine*, vol. 17, no. 2, pp. 131–155, October 1999.
[31] ——, "Evolutionary computation in medicine: An overview," *Artif. Intell. Medicine*, vol. 19, no. 1, pp. 1–23, May 2000.
[32] W. Pedrycz, Ed., *Fuzzy Evolutionary Computation*. Norwell, MA: Kluwer, 1997.
[33] M. A. Potter, "The design and analysis of a computational model of cooperative coevolution," Ph.D. dissertation, George Mason University, Dept. Comput. Sci., 1997.
[34] M. A. Potter and K. A. DeJong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Computation*, vol. 8, no. 1, pp. 1–29, Spring 2000.
[35] I. Rechenberg, "Cybernetic solution path of an experimental problem," in *Farborough Hants: Royal Aircraft Establishment*, August 1965, Library Translation 1122, English Translation of lecture given at the Annual Conference of the WGLR at Berlin in September, 1964.
[36] ——, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart: Frommann-Holzboog, 1973.
[37] C. D. Rosin and R. K. Belew, "New methods for competitive coevolution," *Evolutionary Computation*, vol. 5, no. 1, pp. 1–29, 1997.
[38] H.-P. Schwefel, "Kybernetische evolution als strategie der experimentelen forschung in der stromungstechnik," Master's thesis, Technical University of Berlin, Hermann Föttinger Institute for Hydrodynamics, Mar. 1965.
[39] ——, "Evolutionsstrategie und numerische Optimierung," Ph.D. dissertation, Technische Universitat Berlin, Dept. Process Eng., Berlin, May 1975.
[40] R. Setiono, "Extracting rules from pruned neural networks for breast cancer diagnosis," *Artif. Intell. Medicine*, vol. 8, pp. 37–51, 1996.
[41] ——, "Generating concise and accurate classification rules for breast cancer diagnosis," *Artif. Intell. Medicine*, vol. 18, no. 3, pp. 205–219, 2000.
[42] R. Setiono and H. Liu, "Symbolic representation of neural networks," *IEEE Comput.*, vol. 29, no. 3, pp. 71–77, Mar. 1996.
[43] I. Taha and J. Ghosh, "Evaluation and ordering of rules extracted from feedforward networks," in *Proc. IEEE Int. Conf. Neural Networks*, 1997, pp. 221–226.
[44] M. D. Vose, *The Simple Genetic Algorithm*. Cambridge, MA: MIT Press, Aug. 1999.
[45] R. R. Yager and D. P. Filev, *Essentials of Fuzzy Modeling and Control*. New York: Wiley, 1994.