JOURNAL OF THEORETICAL AND APPLIED MECHANICS
42, 3, pp. 417-444, Warsaw 2004

MULTI-COMBINATIVE STRATEGY TO AVOID PREMATURE CONVERGENCE IN GENETICALLY-GENERATED FUZZY KNOWLEDGE BASES

Sofiane Achiche
Marek Balazinski
Luc Baron

Department of Mechanical Engineering, École Polytechnique de Montréal, Montréal, Canada
e-mail: sofiane.achiche@polymtl.ca; marek.balazinski@polymtl.ca; luc.baron@polymtl.ca

A growing number of industrial fields is concerned by complex and multi-objective problems. For this kind of problems, optimal decision making is critical. Decision support systems using fuzzy logic are often used to deal with complex and large decision making problems. However the main drawback is the need of an expert to manually construct the knowledge base. The use of genetic algorithms proved to be an effective way to solve this problem. Genetic algorithms model the life evolution strategy using the Darwin theory. A main problem in genetic algorithms is the premature convergence, and the last enhancements in order to solve this problem include new multi-combinative reproduction techniques. There are two principal ways to perform multi-combinative reproduction within a genetic algorithm, namely the Multi-parent Recombination, Multiple Crossover on Multiple Parents (MCMP); and the Multiple Crossovers Per Couple (MCPC). Both techniques try to take the most of the genetic information contained in the parents. This paper explores the possibility to decrease premature convergence in a real/binary like coded genetic algorithm (RBCGA) used in automatic generation of fuzzy knowledge bases (FKBs). The RBCGA uses several crossover mechanisms applied to the same couple of parents. The crossovers are also combined in different ways creating a multiple offspring from the same parent genes. The large family concept and the variation of the crossovers should introduce diversity and variation in otherwise prematurely converged populations and hence, keeping the search process active.

Key words: artificial intelligence, fuzzy decision support system, fuzzy knowledge base, learning, premature convergence, genetic algorithm, crossover operators

1. Introduction and problem definition

Fuzzy decision support systems use an approximate reasoning that emulates the human thinking process. The thinking is done through fuzzy knowledge bases (FKB). The behavior of the FKB is easily understandable by a human being, since it is expressed through simple fuzzy sets and a set of linguistic rules. A FKB is generally constructed manually by an expert who translates his own knowledge of the task at hand to propose fuzzy sets – number, shape and repartition – and the appropriate fuzzy rules. The method used by the expert is based on a tedious process of trial and error approach, since fuzzy systems, unlike some other artificial intelligence methods, cannot learn from data. This lack implied several researches to automatically generate the rule base of a FKB from the expert knowledge or a numerical set of data (Castellano *et al.*, 1997; Casillas *et al.*, 2000; Hagras *et al.*, 1999). Other researchers, concentrated their efforts on the automatic generation of fuzzy sets (Nomura *et al.*, 1992; Valesco *et al.*, 1997). Other investigations were focused on simultaneous generation of the fuzzy sets and the fuzzy rules (Achiche *et al.*, 2002; Baron *et al.*, 2001; Xiong and Litz, 2002). To perform automatic generation of FKBs, several optimization algorithms can be used, however with the presence of non-derivable functions in the FKBs (inference engine ... etc), the presence of a high number of possible FKBs for each problem encountered and the number of fuzzy rules increasing exponentially with the number of fuzzy sets, genetic algorithms (GAs) appear to be the most promising learning tool.

The GAs are powerful stochastic optimization methods presented first in the late 1960's and early 1970's by John Holland (Holland, 1973) and his students. In the mid-1980's, these algorithms became more popular and were used in several fields such as machine learning (Goldberg, 1989). GAs are considered here as an optimization tool for the automatic generation of FKBs. The GA used in this paper is a real/binary-like coded genetic algorithm (RBCGA) developed by the authors (Achiche *et al.*, 2003). The RBCGA allows one to solve a contradictory paradigm in terms of FKB precision and simplicity (less fuzzy rules and less fuzzy sets) considering a randomly generated population of potential FKBs. The RBCGA is divided in two principal coding ways. First, a real coded genetic algorithm (RCGA) that maps the fuzzy sets repartition and number into a set of real numbers, and second, a binary like coded genetic algorithm deals with the fuzzy rule base relationships (a set of integers is used). Although, GAs have proved to be effective in the automatic generation of FKBs (Baron *et al.*, 2001), one of their drawbacks is the

premature convergence problem (stagnation of the evolution). A GA is said asymptotically convergent if there is no change in the individuals from a generation to the next. At this state, all the individuals look almost alike. GAs can converge even though the near optimal solution is not yet found, which means that the GA is unable to explore the remaining search space, namely the premature convergence. The premature convergence is generally due to the loss of diversity within the population. This loss can be caused by the selection pressure, the schemata distribution due to crossover operators, and a poor evolution parameters setting (Li *et al.*, 1992; Mahfoud, 1995). Premature convergence is in some cases considered a complete failure of the GAs (Goldberg, 1989; Eshelman and Schaffer, 1991). To avoid the premature convergence, several researches have been performed, such as, dynamic niche sharing to efficiently identify and search multiple niches (peaks) in a multi-modal domain (Brad and Miller, 1996), the partition of the population in several subpopulations have been tried, this inducted the distributed genetic algorithms (Herrera and Lozano, 1997). Multi-combinative techniques have been also newly reported.

There are two principal ways to perform multi-combinative reproduction within a genetic algorithm. The first one being the Multi-parent Recombination, i.e., Multiple Crossover on Multiple Parents (MCMP) (Eiben and Bäck, 1997; Ono and Kobayashi, 1997; Tsutsui, 1998; Tsutsui and Ghosh, 1998). The second one is the Multiple Crossovers Per Couple (MCPC). Both techniques try to take most of the genetic information contained in the parents.

This paper explores the possibility to decrease the premature convergence in the RBCGA used for the automatic generation of FKBs by means of an MCPC strategy. The large family concept and the variation of crossovers should introduce diversity and variation in otherwise prematurely converged populations, and hence, keeping the search process active.

An overview of fuzzy logic based decision system, the FDSS software Fuzzy-Flou, developed at Ecole Polytechnique (Canada) and University of Silesia (Poland) (Balazinski *et al.*, 1993) is first presented. This software is used for the validation tests. Then, it presents a description of the RBCGA used in this work, explaining the specificities of the reproduction and mutation mechanisms and their combination to ensure the MCPC strategy. Validation results are presented along with a comparative study of their performances evaluated through different evolution parameters. Finally, an application on experimental data is discussed.

2. Fuzzy decision support systems

A rule-based approach to the decision making using fuzzy logic techniques may consider an imprecise vague language as a set of rules linking a finite number of conclusions. The knowledge base of such systems consists of two components: a linguistic terms base and a fuzzy rules base (Balazinski *et al.*, 1993). The former is divided into two parts: the fuzzy premises (or inputs) and the fuzzy conclusions (or outputs). In general, both can contain more than one premise and one conclusion. However, we limit ourself in this paper to systems of N multiple inputs and one single output (MISO). Moreover, for the sake of simplicity, we consider only non-symmetric triangular fuzzy sets on the premises and sharp-symmetric triangular fuzzy sets on the conclusion. The representation of such imprecise knowledge by means of fuzzy linguistic terms makes it possible to carry out quantitative processing in the course of inference that is used for handling uncertain (imprecise) knowledge. This is often called approximate reasoning (Zadeh, 1973). Such knowledge can be collected and delivered by a human expert (e.g. decision-maker, designer, process planer, machine operator). This knowledge, expressed by $(k = 1, 2, ..., K)$ finite heuristic fuzzy rules of the type MISO, may be written in the form

$$R^k_{MISO}: \text{ if } x_1 \text{ is } X^k_1 \text{ and } x_2 \text{ is } X^k_2 \text{ and } \ldots \text{ and } x_N \text{ is } X^k_N \text{ then } y \text{ is } Y^k \quad (2.1)$$

where $\{X^k_i\}^N_{i=1}$ denote values of linguistic variables $\{x_i\}^N_{i=1}$ (conditions) defined in the following universe of discourse $\{\boldsymbol{X}_i\}^N_{i=1}$; and Y^k stands for the value of the independent linguistic variable y (conclusion) in the universe of discourse $\boldsymbol{Y}$. The global relation aggregating all rules from $k = 1$ to K is given as

$$R = also^K_{k=1}(R^k_{MISO}) \quad (2.2)$$

where the sentence connective *also* denotes any t- or s-norm (e.g., *min* ($\wedge$) or *max* ($\vee$) operators) or averages. For a given set of fuzzy inputs $\{X'_i\}^N_1$ (or observations), the fuzzy output Y' (or conclusion) may be expressed symbolically as

$$Y' = (X'_1, X'_2, \ldots, X'_N) \circ R \quad (2.3)$$

where $\circ$ denotes a compositional rule of inference (CRI), e.g., the *sup*-$\wedge$ or *sup-prod* (*sup*-$*$). Alternatively, the CRI of Eq. (2.3) is easily computed as

$$Y' = X'_N \circ \ldots \circ (X'_2 \circ (X'_1 \circ R)) \quad (2.4)$$

In FDSS Fuzzy-Flou, there are four variants of CRI: the sentence connective *also* can be either $\vee$ or *sum* ($\sum$); the compositional operator is the *supremum*

(sup) of either $\wedge$ or $*$, denoted $sup\wedge$ and $sup*$; while the sentence connective *and* and the fuzzy relation are always identical to the second part of the latter. For the sake of brevity, all four variants of CRI – i.e.: $\vee$-$sup\wedge$-$\wedge$-$\wedge$; $\vee$-$sup*$-$*$-$*$; $\sum$-$sup\wedge$-$\wedge$-$\wedge$; and $\sum$-$sup*$-$*$-$*$ – are expressed as

$$Y' = \left\{ \begin{matrix} \vee_{k=1}^{K} \\ \sum_{k=1}^{K} \end{matrix} \right\} \sup_{\{x_i \in X_i\}_{i=1}^{N}} *_t\Big(*_t(X'_N, \ldots, X'_2, X'_1),\ *_t(X_1^k, X_2^k, \ldots, X_N^k, Y^k)\Big) \tag{2.5}$$

where $*_t(\cdot)$ denotes the t-norm of $(\cdot)$ defined as either $\wedge$ or $*$. These variants of CRI mechanisms allow us to obtain different conclusions represented as the membership function Y'. In FDSS Fuzzy-Flou, there are three defuzzification methods: the center of gravity (COG); the mean of maxima (MOM); and the height method (HM). All the results presented in this paper are obtained with the $\sum$-$sup*$-$*$-$*$ CRI and COG as defuzzification. Figure 1 shows a screen printout of the premises and conclusion (on the right), the fuzzy rules and settings (on the left) of the FDSS Fuzzy-Flou software.

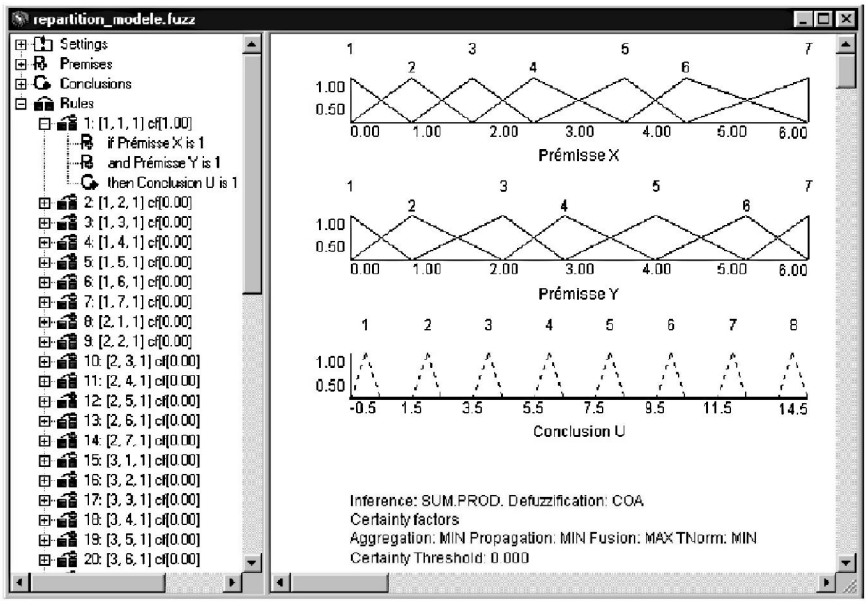

Fig. 1. Screen shot of the FDSS Fuzzy-Flou

3. Real/binary-like coded genetic algorithm

The most important success of the GAs remains in their evolution basis rather than the coding, hence the occurrence of real coded genetic algorithms (RCGA) in recent years. At first, they were mostly used in numerical optimization works (Herrera *et al.*, 1995). The use of RCGAs overcomes some of the weaknesses of Binary Coded Genetic Algorithms (BCGAs), such as the low resolution of the solutions (Baron *et al.*, 2001). Moreover, most optimization works are performed in a continuous mathematical space (real space). However, the FKBs contains two cooperative but distinct parts in terms of:

- premises, conclusions, along with the fuzzy sets on them
- fuzzy rules base.

They are distinct in a way that the first one deals with real numbers while the second one uses integer numbers, since the fuzzy rules are simple pointers to the index of a fuzzy set on the conclusion. For this matter, we used a combination between an RCGA and a BCGA, where the binary part is mapped into a string of integers. The algorithm is therfore called the Real/Binary-Like Coded Genetic Algorithm (RBLGA).

3.1. Coding

The *genotype* corresponds to several independent sets of reals and a set of integers. Here, the *genotype* can be described as follow:

Premises and conclusion – There is as many real number sets as there is premises in the problem and one set for the conclusion. Each set contains a predefined maximum number of real numbers representing the location of the summit of each fuzzy set on each premise and the conclusion. The two summits located at the minimum and maximum limits of each premise and conclusion are not coded. We consider non-symmetrical-overlapping triangular fuzzy sets for the premises and symmetrical triangular fuzzy sets for the conclusion.

Fuzzy rules – The fuzzy rules are coded as a set of integers representing an ordered list of the combination of the premises. Each integer in the set represents a conclusion fuzzy set summit. The initial population of FKBs is composed of P FKBs randomly generated. The *genotype* of each new solution contains all the sets mentioned above, however as we will explain below, the size of the sets can decrease throughout the evolution, but cannot increase.

3.1.1. Reproduction mechanisms of the RBCGA

The reproduction is performed by *crossover* of the *genotype* of the parents to obtain the *genotype* of an offspring (or two offsprings). The reproduction of the FKBs in the RBCGA is preformed through three principal crossover mechanisms, each one has its own purpose, as explained below.

a) Multi crossover

The multi-crossover is a combination of crossovers applied on different parts of the *genotype*.

a.1) Premises/conclusion crossover

For this part of the FKB, three crossover mechanisms are used concurrently as explained later in the paper. For the three reproduction mechanisms, if $A = \{x_1, x_2, \cdots, x_i, \cdots, x_n\}$ and $B = \{y_1, y_2, \cdots, y_i, \cdots, y_n\}$ represents the two selected parents, C the offspring obtained by the crossover of A and B then $C = \{z_1, z_2, \cdots, z_i, \cdots, z_n\}$, where z_i are selected according to the used crossover. The values of the offspring can belong to two main intervals, i.e., the exploitation or the exploration zone. The exploitation means using the interval between the values of the two parents, while the exploration uses an interval outside of these two limits, therefore trying new solutions (see Fig.), knowing that:

- in the exploitation zone, the offspring inherits behaviors close to those of his parent's average, since he was produced between them
- in the exploration zone, the offspring is a result of an exploration, his attributes are away from his parent's average.

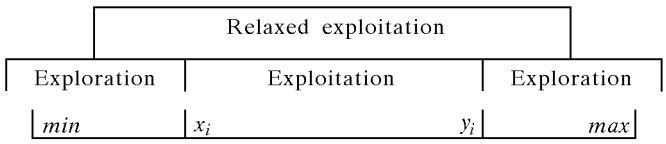

Fig. 2. Interval action of an offspring gene

a.1.1) Blended crossover α

The blended crossover α is denoted as *BLX*-α (Herrera and Lozano, 2000), where α controls the exploitation/exploration level of the offspring obtained from the selected parents. As shown in Fig. 3, the z_i values of the offspring are randomly selected in the interval $[min, max]$ where:

- $max = \text{maximum}\{x_i, y_i\} - \text{I}\alpha$
- $min = \text{minimum}\{x_i, y_i\} + \text{I}\alpha$
- $\text{I} = \text{maximum}\{x_i, y_i\} - \text{minimum}\{x_i, y_i\}$

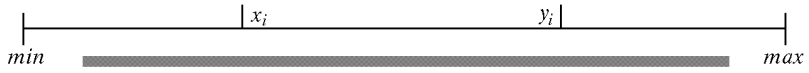

Fig. 3. Blended crossover α – BLX-α

In order to avoid any bias in either direction (exploitation or exploration) the value of α is set to 0.5, which provides offspring in the relaxed exploitation zone.

a.1.2) Non-uniform arithmetical crossover

The non-uniform Arithmetical Crossover (non-uniform arithmetical crossover) (Michalewicz, 1992) is denoted as *NAX*. The z_i values of the offspring are computed as follows:

- $z_i = \lambda x_i + (1-\lambda)y_i$ or $z_i = \lambda y_i + (1-\lambda)x_i$.

The z_i values are set to one of these two values at an equal probability (50% each). The value of λ is randomly selected at each generation in the interval $[0, 1]$, making the non-uniform part of the mechanism. The min, max values are the limits, as shown in Fig. 4.

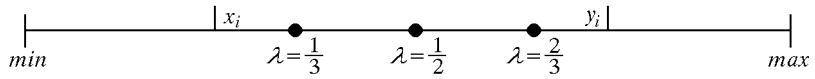

Fig. 4. Arithmetical crossover *NAX*

a.1.3) Extended line crossover

The Extended Line Crossover (Mählenbein and Schlierkamp-Voosen, 1993) is denoted as *ELX*. The z_i values of the offspring are computed as follows:

- $z_i = \beta x_i + \beta(y_i - x_i)$.

The value of β is randomly chosen within the interval $[-0.25, 1.25]$. Again, the min, max values are the limits as shown in Fig. 5.

Fig. 5. Extended line crossover *ELX*

a.2) Fuzzy rules crossover

Since the part of the *genotype* representing the fuzzy rule base is composed of integer numbers, the crossover on this part of the *genotype* is done by a

simple crossover. The use of BLX/NAX/ELX crossovers is not suitable in this case, because of the integer nature of the values. The operation is performed by inverting the end part of the *sets* of the parents at a randomly selected *crossover* site as shown in Fig. 6. These two mechanisms are governed by the initiating probability pr_1.

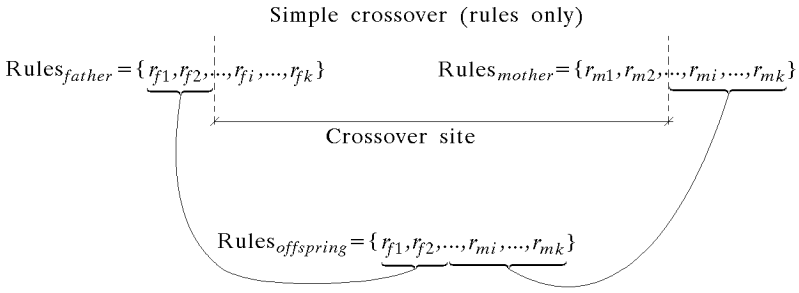

Fig. 6. Simple crossover of the RBLGA

b) Fuzzy set reducer

This mechanism is set to increase the simplicity of the FKBs by selecting on each premise a summit and erasing it from the respective sets representing the premises. The mechanism can be described as follows:

- the set given by

$$premise_i = \{summit_1, summit_2, \cdots, summit_i, \cdots, summit_n\}$$

  is selected
- one of the summits is randomly selected and erased
- the operation is repeated for each premise.

Decreasing the number of fuzzy sets reduces the maximum number of fuzzy rules. This mechanism allows one to obtain different and more simple solutions (FKBs). This mechanism is governed by the initiating probability pr_2.

c) Mutation

Mutation is the creation of a new individual by altering the gene of the existing one. The probability pr_3 governs the occurrence of this mechanism. This paper uses a uniform mutation (Michalewicz, 1992) applied to one randomly selected individual, as follows:

- an individual C is randomly selected, $C = \{z_1, z_2, \cdots, z_i, \cdots, z_n\}$
- a mutation site i is randomly chosen in the interval $[1, n]$

- z_i is changed to $z'_i = \text{random}(a_i, b_i)$, where a_i and b_i are the limits enclosing z_i.

d) Selection mechanism
The selection mechanism used in this paper (to select the parents) is performed as follows:

- a real value R, is randomly generated in the interval $[0, 1]$
- R is multiplied by S, S being the sum of fitness values of the individuals in the population. The value RS is obtained
- beginning from the best individual of the population, the fitness values are summed till the result is higher than RS. The last added individual is considered as a potential parent
- the same mechanism is used to select the second parent.

4. Performance criteria

The performance criteria allow one to compute the rating of each FKB. This performance rating is used by the RBCGA in order to perform the natural selection. Here, the performance criteria are the accuracy level of a FKB (approximation error) in reproducing the outputs of the learning data. The approximation error is measured using the root-mean-square error

$$\Delta_{rms} = \sqrt{\sum_{i=1}^{N} \frac{(RBCGA_{output} - data_{output})^2}{N}} \tag{4.1}$$

where N represents the number of learning data. The fitness value is evaluated as a percentage of the output length $L = z_{max} - z_{min}$ of the conclusion, i.e., $\phi_{rms} = 100(L - \Delta_{rms})/L$.

4.1. Natural selection (elitist approach)

Natural selection is performed on the population by keeping the *most* promising individuals based on their fitness value (elitist approach). This is equivalent to using solutions that are closest to the optimum. For convenience, we keep the size of the population constant.

The first generation begins with P FKBs, and the same number of additional FKBs are generated by reproduction and mutation. Moreover, in the

RBLGA *natural selection* is applied on the $2P$ FKBs by ordering them according to the performance criterion ϕ_{rms} and keeping the P first FKBs. The size P has to be selected depending upon the performances of the computer in use. A high value of P generally ensures a better diversity in the population, which helps to avoid premature convergence. However, it increases the learning time.

5. Evolutionary strategy

To apply the MCPC strategy to the BCGA, the crossover mechanisms applied to the genotype are combined in different ways to create a multiple different offspring from the same parent's genes. A critical point in overcoming premature convergence is the identification of why it occurs and when it occurs, along with a good balance between exploitation and exploration throughout the evolution. It is widely established that a loss of population diversity leads to premature convergence. The MCPC strategy is used on the reproduction mechanisms. Although we can use the recombination strategy on all the parts of the FKB, the quality of the FKB is more sensitive to the quality of the repartition and the number of fuzzy sets (Bonissone *et al.*, 1996), and hence, the evolutionary strategy is only applied to the premises. The *Fuzzy Set Reducer* mechanism is not used in the MCPC strategy.

The MCPC strategy will be applied as follow:

Single application – each crossover mechanism is applied independently (BLX-α, NAX and ELX), the results are used as a comparison basis

Linear combination – a linear combination of the three mechanisms is used

Simultaneous application – all three mechanisms are used simultaneously in the evolution

Exploration balance mechanism – a switch of the reproduction mechanism is set based on some criteria, this mechanism being divided into two main parts.

5.1. Test functions

The learning performances of the RBCGA are investigated using three examples of known behavior in terms of 3D surfaces of the type $z = f(x, y)$, where the nodes are the learning set of sampled data. We have used three

different surfaces of different complexities to have a better idea on the generality of the results. The evolution and selection criteria are set to the following values:

- $pr_1 = 100.0\%$
- $pr_2 = 0.0\%$
- $pr_3 = 5.0\%$
- Maximal complexity: 4 fuzzy sets (including the limits) on each premise and 8 on the conclusion (no limits on the number, it can match the number of fuzzy rules). These numbers were chosen based on performance tests applied on the three surfaces.

The evolution is completely governed by the *multi-crossover reproduction* mechanism, the *fuzzy set reducer* is disabled to allow a more comparable FKBs, otherwise the number of fuzzy rules can vary excessively. However, the number of fuzzy sets can decrease when two summits are overlapping, hence, the reducing of the fuzzy rule base. The theoretical surfaces are the following:

1. Sinusoid surface

The sinusoid surface is defined as

$$z = \sin(x\ y) \qquad \begin{matrix} 0 \leqslant x \leqslant 1.6 \\ 0 \leqslant y \leqslant 1.4 \end{matrix} \tag{5.1}$$

2. Spherical surface

The spherical surface is defined as

$$z = x^2 + y^2 \qquad \begin{matrix} -2.0 \leqslant x \leqslant 2 \\ -2 \leqslant y \leqslant 2 \end{matrix} \tag{5.2}$$

3. Hyper-tangent surface

The hyper-tangent surface is defined as

$$z = \tanh(x(x^2 + y^2)) \qquad \begin{matrix} -0.2 \leqslant x \leqslant 1.4 \\ -0.2 \leqslant y \leqslant 1.4 \end{matrix} \tag{5.3}$$

In order to measure the fitness (accuracy levels) of the RBCGA in generating FKBs, and for the sake of comparison, several runs have been made on the three surfaces. The population size P was set to 20 individuals. The runs were performed three times for 10, 50, 100, 500, 1000, 2500 and 5000 generations. The fitness of the best individual, the fitness of the worst individual and the average fitness of all the individuals are taken into account at the last generation for each surface. The average value of the three different results obtained for each theoretical surface was computed. The runs are performed using different versions of the RBCGAs (different crossover mechanisms).

5.1.1. Test functions: single applications

The multi-crossover of the RBCGA uses the $BLX-0.5$, the NAX and ELX, independently. At the end of the evolution the averages of the results obtained for the three surfaces (each test being performed three times) are computed.

Tables 1 to 7 show the fitness for the best individual, the worst individual in the population, the average fitness of the last population along with the size of the fuzzy rules base corresponding to the crossover mechanism applied.

1. Blended crossover BLX-0.5
As shown in Table 1, the highest value obtained is 91.63% after 5000 generations. However after only 500 generations the fitness is already up to 91.35%. The worst fitness gets closer to the best individual with the improvement of the populations, same goes for the mean value, which shows a lack of diversity in the population. A drawback very difficult to overcome since it is a direct consequence of the elitist philosophy.

Table 1. Average ϕ_{rms} obtained by $BLX-0.5$

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	82.76%	77.88%	79.64%	16
50	90.68%	90.48%	90.54%	12.67
100	91.01%	91.01%	91.01%	12.67
500	91.35%	91.17%	91.18%	12.67
1000	91.35%	91.18%	91.21%	12.67
2500	91.60%	91.36%	91.48%	12.67
5000	91.63%	91.54%	91.59%	12.67

Figure 7 shows the fitness evolution of the best individual, the fitness reaches a plateau around 92.00%. The number of rules is decreased to 13

2. Non-uniform Arithmetical Crossover
In Table 2, the highest value obtained is 92.00% after 5000 generations. The fitness gained around 1.00% from 500 generations to 5000, which is not negligible but still quite low. The worst fitness is very close to the best one, same for the mean value which can be interpreted as a presence of too many alike individuals in the population. This is reached after 50 generations only. The number of rules stays at 16, which means that no simplification occurred during the evolution.

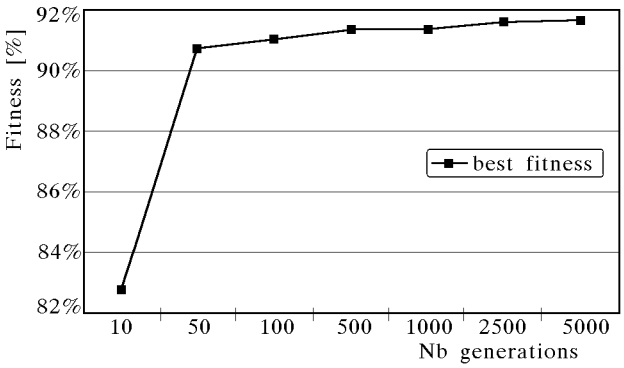

Fig. 7. Accuracy level for the best individuals using $BLX - 0.5$

Table 2. Average ϕ_{rms} obtained by NAX

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	82.85%	79.63%	80.68%	16.00
50	89.17%	89.16%	89.16%	16.00
100	89.45%	89.45%	89.45%	16.00
500	91.09%	91.04%	91.05%	16.00
1000	91.45%	91.45%	91.45%	16.00
2500	91.83%	91.72%	91.73%	16.00
5000	92.00%	91.98%	91.98%	16.00

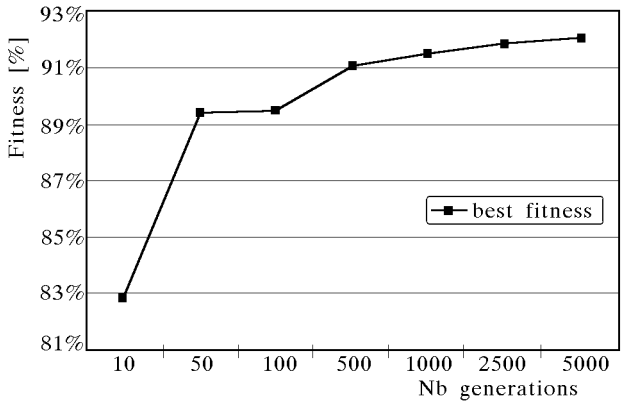

Fig. 8. Accuracy level for the best individual using NAX strategy

Figure 8 shows the fitness evolution of the best individual, the fitness reaches again a plateau around 92.00%. The evolution is slower than for the $BLX - 0.5$, even if the best values are still very comparable (91.63% vs 92.00%).

3. Extended-Line Crossover

From Table 3, the highest fitness value is 92.78% reached after 5000 generations. After 100 generations the best fitness reached is 91.10%. The worst and mean fitness values are very close to the best individual from the $50th$ generation.

Table 3. Average ϕ_{rms} obtained by ELX

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	82.63%	79.39%	80.37%	16.00
50	89.53%	89.46%	89.49%	13.67
100	91.10%	91.10%	91.10%	13.67
500	91.56%	91.56%	91.56%	13.67
1000	91.92%	91.92%	91.92%	13.67
2500	92.55%	91.55%	91.55%	13.67
5000	92.78%	91.77%	91.77%	13.67

The ELX reached the best fitness compared to $BLX - 0.5$ and NAX. From Fig. 9, we can see that the fitness evolution of the best individual is better distributed through the generations, however the slow down at the end is still present, where a plateau seems to be reached.

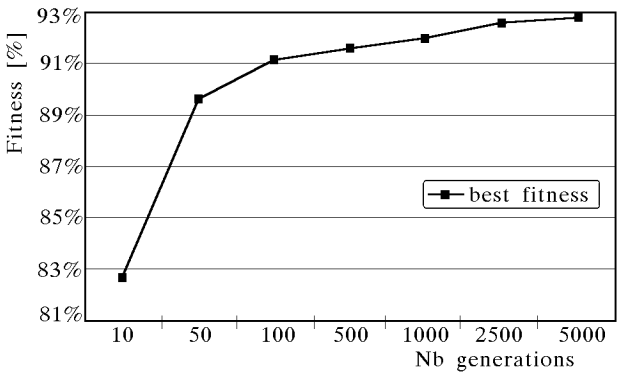

Fig. 9. Fitness the best individual using ELX

Using the single crossover applications, we have noticed three main negative points:

1. Fast convergence (premature): after 50 generations the evolutions slowed down noticeably
2. Lack of diversity: the worst fitness value gets too close to the best fitness value too fast

3. Reaching a plateau: a plateau is reached around the fitness value of 92.00%, a plateau being a period in the learning process in which minimum or no progress takes place.

To try to overcome the three above drawbacks, we applied the strategies presented in Section 5.

5.1.2. Test functions: linear combination

The Linear Combination is denoted as LC. The crossover mechanism is a linear combination of the $BLX-0.5$, the NAX and the ELX. From the existing population P, a new population of the offspring is created, namely P'. However, one third of the offspring is generated using $BLX-0.5$, the other third is generated using NAX and the last third is generated using ELX.

The assumption is that the use of different mechanisms should create more diversity and hence decrease the premature convergence and its consequences. Moreover, it is noteworthy to say that in the LC the three mechanisms do not work in competition, but rather in a symbiotic way, since all the offspring created are part of the evolution. The results obtained are presented in Table 4.

Table 4. Average ϕ_{rms} obtained by LC

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	81.79%	78.92%	79.87%	14.67
50	89.55%	89.47%	89.49%	13.33
100	89.97%	89.47%	89.56%	13.33
500	91.61%	91.61%	91.61%	13.33
1000	91.82%	91.77%	91.77%	13.33
2500	92.54%	91.42%	91.43%	13.33
5000	93.16%	93.05%	93.07%	13.33

We can see that the $\approx 92.00\%$ plateau is exceeded. The LC reached a fitness value of 93.16% after 5000 generations. From Fig. 10, we can see that the speed of convergence decreased (the curve is less abrupt). The plateau, noticed in the single mechanisms, is less acute here. In the last stages of the evolution (1000 generations and more) the improvement is continuing for the best individuals (see Fig. 10). The size of the rule base decreased also, since it went from 16 down to 13 fuzzy rules, which is equivalent to the best result obtained by the single applications.

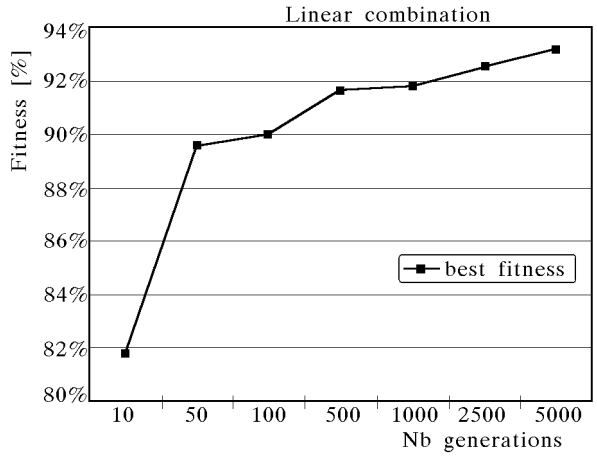

Fig. 10. Accuracy level for the best individual using LC strategy

5.1.3. Test functions: simultaneous application strategy

The Simultaneous Application Strategy is denoted as SA. The crossover mechanism is a simultaneous application of the $BLX - 0.5$, the NAX and the ELX. From the existing population P, a new population of the offspring is created, namely P'. However, for each pair of selected parents, 6 children are generated, two of them with $BLX - 0.5$, two with the NAX and the last two with the ELX. The different mechanisms shall create diversity and hence decrease the premature convergence and its consequences. However, in the SA strategy the crossover mechanisms are used competitively, in a sense that, only the best offsprings are selected to be a part of the evolution. Table 5 shows that the $\approx 92.00\%$ plateau is also exeeded. The SA reached 93.61% after 5000 generations.

Table 5. Average ϕ_{rms} obtained by SA

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	83.13%	78.50%	79.74%	13.67
50	89.75%	89.40%	89.51%	13.67
100	90.84%	90.77%	90.79%	13.67
500	92.93%	92.92%	92.92%	13.67
1000	93.23%	93.23%	93.23%	13.67
2500	93.46%	93.36%	93.40%	13.67
5000	93.61%	93.40%	93.43%	13.67

From Fig. 11, we can see that the speed of convergence decreased if compared to the single evolution strategies (Fig. 7 - Fig. 9). The curve is less abrupt from 50-th generations. The plateau noticed in the single mechanisms disappeared. After the 1000-th generation the improvement is slowing down but still active (see Fig. 11). The SA strategy gave very similar results to the ones obtained by the LC strategy (with a small improvement in the fitness), and the curves are also very alike. The size of the rule base decreased, since it went from the maximum of 16 down to around 13 fuzzy rules, which is again equivalent to the results obtained by the best single application ($\approx$ 13 fuzzy rules) and the LC strategy ($\approx$ 13 fuzzy rules).

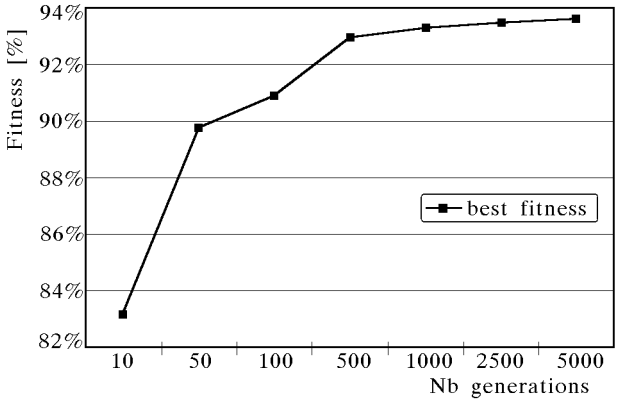

Fig. 11. Accuracy level for the best individual using SA strategy

5.1.4. Test functions: exploration/exploitation balance strategy

The Exploration/Exploitation Balance Strategy is denoted as EEB. As mentioned earlier in the paper, the two main reasons that cause premature convergence are the loss of diversity in the population and the bad balance between the exploration and exploitation throughout the evolution process. In this section, we study the influence of the exploration/exploitation balance on the premature convergence aiming to find which combination is the best as follows:

1. Exploiting the individuals at the early stages of the evolution, applying relaxed exploitation through the main part of the evolution and then exploring the individuals at the late stages
2. Exploring at the early stages of the evolution, applying relaxed exploitation through the main part of the evolution and then exploiting at the late stages of the evolution.

The first combination is called $EEB1$, while the second is called $EEB2$. Both mechanisms use the $BLX - \alpha$, since it is quite easy to control the exploration/exploitation balance through the variation of α. The different values given to α influence the exploration, exploitation or relaxed exploitation levels of the crossover mechanism. The three values are set as follows:

- Exploration: $\alpha = 1.00$ for total exploration
- Relaxed Exploitation: $\alpha = 0.50$
- Exploitation: $\alpha = 0.1$ for close to the maximum exploitation – $\alpha \neq 0$, in order to make a difference between the BLX-α and the uniform mutation.

The question we should ask ourselves is: at which stage of the evolution process these three mechanisms should occur?. The stage of the evolution is defined by the generation number. However, what can be considered at an early stage of the evolution? The 10 first generations? What, if the maximal number generations is set to 10? For this matter we assumed the following, considering the maximum number of generations N:

- The first quarter (1/4) of N is considered being the early stages
- The last quarter of N is considered being the last stages
- The remaining part (from the end of the first quarter to the beginning of the last one, limits non included) is considered being the evolution stage.

In the next sections, we present the results obtained for both EBB_1 and EBB_2 strategies.

1. Exploitation/Relaxed exploitation/Exploration $EEB1$

For EBB_1, α is set to 0.10 at the early stages, then changes to 0.50 for the relaxed exploitation stage and finally switches to 1.00 for the last stages of the evolution. The results obtained by using EBB_1 are reported in Table 6.

Table 6. Average ϕ_{rms} obtained by EEB1

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	84.19%	78.80%	80.81%	16
50	88.71%	87.49%	88.03%	16
100	88.84%	88.47%	88.58%	16
500	90.72%	90.71%	90.72%	16
1000	91.61%	91.28%	91.52%	16
2500	90.97%	90.90%	90.93%	14.67
5000	90.42%	90.42%	90.42%	14.67

The first noticeable change is that the best fitness reached is not for the 5000 generations test but for the 1000 generations one. It is the lowest fitness obtained until now (for the best individual) by the different strategies. From Fig. 12, we can clearly see that unlike for the other tests the fitness level is not proportional to the number of generations. This is due to the fact that the number of exploited and/or explored individuals is different from a test to another. For example, a run using 1000 generations: during 250 generations the new breed is obtained using mostly exploitation, while in a run of 5000 generations, the exploitation is performed during 1 250 generations. A good balance has to be found to obtain the best results with this mechanism, which is not a simple task to do. However, even though the fitness level is not as high as the other tests, it remains that the evolution of the convergence speed is quite interesting, since we do not have a plateau as shown in Fig. 12.

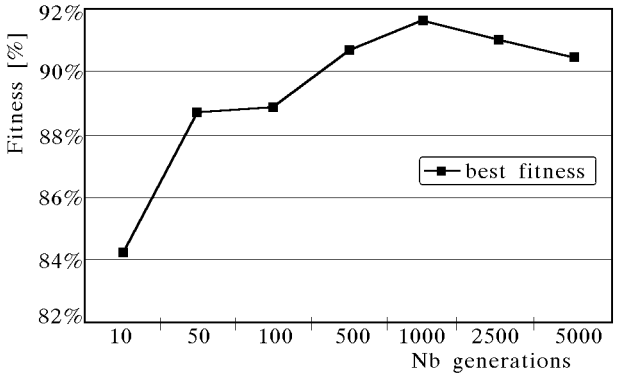

Fig. 12. Accuracy level for the best individual using EEB1 strategy

The number of fuzzy rules is around 15 (see Table 6) which is higher than the one from the previous tests, i.e., the FKBs obtained are less simple.

2. Exploration/Relaxed exploitation/Exploitation $EEB2$

For EBB_2, α is set to 1.00 at the early stages, then changes to 0.50 for the relaxed exploitation stage and finally switches to 0.10 for the last stages of evolution. The results obtained by using EBB_2 are reported in Table 7.

The best fitness level reached is for the 5000 generations test with 93.30%. From the 500 generations test the evolution is stagnating. However, the fitness level achieved exceeded the plateau of the single applications ($\approx$ 92.00%). Figure 13 illustrates the fitness evolution of the best individuals, the fitness reaches a plateau around 93.30%. The evolution seems as fast as for the single applications, even if the best values are still higher ($\approx$ 93.00% vs $\approx$ 92.00%). In the EBB_2, the problem of the balance between the exploration/exploitation

levels and the number of generations did not occur. The EBB_2 seems to be a more natural way to explore solutions since at the early stages it is better to explore the search space, before the diversity starts to drop. The number of fuzzy rules is around 10 (see Table 7) which is the lowest of any other strategies.

Table 7. Average ϕ_{rms} obtained by EEB2

Generation #	Best individual	Worst individual	Mean Fitness	# rules
10	81.92%	79.06%	80.40%	12.33
50	91.59%	91.57%	91.58%	9.00
100	92.89%	92.85%	92.85%	9.67
500	93.29%	93.20%	92.24%	9.67
1000	93.29%	93.23%	93.24%	9.67
2500	93.29%	93.23%	93.25%	9.67
5000	93.30%	93.25%	93.26%	9.67

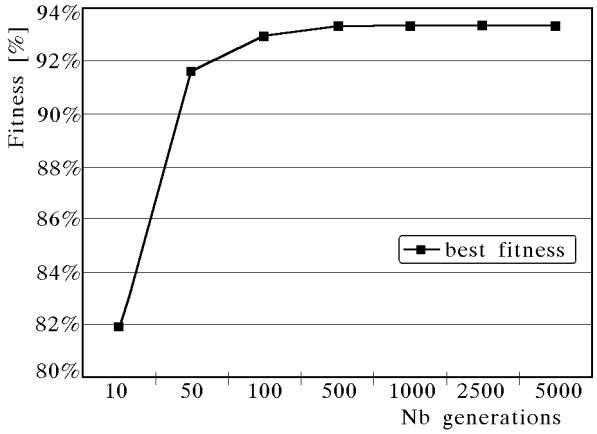

Fig. 13. Accuracy level for the best individual using EEB2 strategy

5.2. Evolutionary strategy: discussion and conclusions

The main goal of the MCPC strategy is to improve (or overcome) the behavior of the RBCGA against premature convergence while generating fuzzy knowledge bases using a small population size. Using three different single crossover applications, we have noticed three negative points:

1. Fast convergence (FC)
2. Lack of diversity (LD)
3. Reaching a plateau (RP).

For the sake of comparison we assume that the FC is overcome if the best fitness value is not reached before 1000 generations (improvement of 0.50% or more), the LD is overcome if the $\approx 92.00\%$ fitness level is exceeded and finally the RP is overcome if the shape of the curve shows no plateau.

The different combinations of multi-combinative evolution strategy achieved various results for the tests made in this paper. Table 8 summarizes the ability of the different strategies, the check mark $\surd$ means that the strategy succeeded in overcoming a negative aspect or achieving a positive one.

Table 8. Performances: single applications vs. MCPC strategies

Strategy	Overcoming FC	Simplification	Improving LD	Overcoming RP
$BLX-0.5$		$\surd$		
NAX	$\surd$			
ELX	$\surd$	$\surd$		
LC	$\surd$	$\surd$	$\surd$	$\surd$
SA	$\surd$	$\surd$	$\surd$	$\surd$
$EEB1$				$\surd$
$EEB2$		$\surd\surd$	$\surd$	$\surd$

From Table 8 it is obvious that the only two strategies achieving all the constraints are the LC and SA ones. Moreover, the $EEB2$, even with the fast convergence, remains very efficient because of the high simplicity level of the FKBs (less fuzzy rules), hence the two $\surd$ marks. The EBB_1 failed almost in every single criteria, which means that the exploitation during the early stages of an evolution is not a good way to perform the optimization process. The best way is to explore the search space while the breed is young and then exploiting at the end for a fine tuning. Among the strategies proposed, the LC, SA and EBB_2 are the best. The SA and LC strategies achieved similar results. The SA achieved a slightly better fitness level at the 5000 generations test. The learning time being a very important point in computer learning, the LC surpasses the SA at this point because it produces three times less children, since for a couple of parents SA produces six children, while the LC generates only two. If a fast convergence (time wise) is needed combined with a high simplification rate (less fuzzy rules), the EBB_2 strategy must be used. The EBB_2 can be very suitable for factory floor applications.

The MCPC strategy overcomes some of the problems faced by the single applications, apart from the EBB_1. In the next section, we apply the multi-

combinative strategy on a set of experimental data and we compare it with the single applications.

6. Application to experimental data

In this section, we use a multi-combinative and single application learning processes on a set of experimental data used to predict tool wear (VB) based on a measure of the feed $feed$, the cutting force F_c, the feed force F_f, the depth of cut ap during turning operations, and the time of turning t (Achiche *et al.*, 2002). In order to simulate factory floor conditions, a typical part is machined on a conventional lathe under six different cutting conditions (Balazinski *et al.*, 202), as shown in Fig. 14.

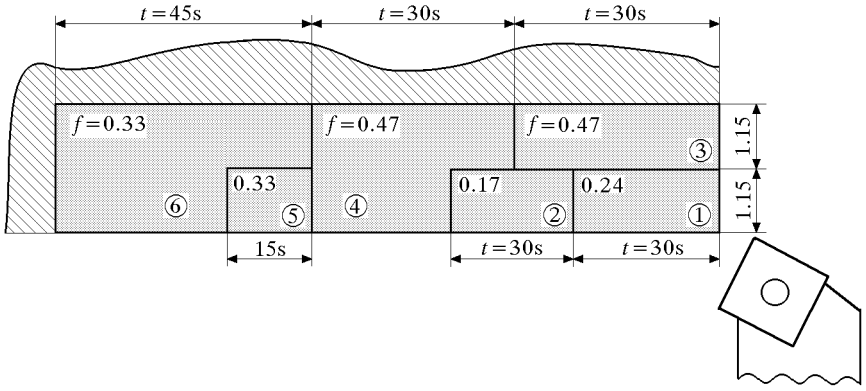

Fig. 14. Sets of cutting conditions

The RBCGA is used to automatically generate the FKB, from the set of experimental data collected during this experiment. The maximum complexity is set to 5 fuzzy sets on each of the 5 input premises and 10 fuzzy sets on the conclusion. Therefore, the maximum number of fuzzy rules is given by $K = 5 \times 5 \times \ldots \times 5 = 5^5 = 3125$.

6.1. Selection of the strategies and the evolution parameters

Since we are dealing with a factory floor modeling problem, we need a strategy that respects the following:

- achieving a reasonable fitness level in a reasonable time (around 5 minutes)

- a simple FKB is preferable, in case it has to be fine tuned manually for further applications.

From these constraints, we can easily pick the EBB_2 application. Because of the simplicity level of its FKBs, the learning will be processed faster by the RBCGA. As for the single application, the $BLX-0.5$ is the one giving the less fuzzy rules, hence it runs faster while achieving comparable results to the other single applications.

The evolution parameters of the RBCGA are:

- Multi crossover application [%]: $pr_1 = 85.00\%$
- Fuzzy set reducer application [%]: $pr_2 = 15.00\%$
- Uniform mutation [%]: $pr_3 = 05.00\%$
- Population size is fixed at 100
- Number of generations is fixed at 350.

The number of generations and the population size has been chosen with respect to the convergence time constraint.

6.2. Application to experimental data: results

As shown in Table 9, the FKB obtained by $BLX-0.5$ is 94.43% accurate, which is a fairly good fitness level. The size of the fuzzy rules base is 32 (reduced from 3125). The highest simplicity level is achieved since each input premise contains the minimum number of fuzzy sets (2) that are obtained by linking the limits.

Table 9. ϕ_{rms} obtained by EEB2 and BLX-0.5 for the experimental data

Crossover mechanism	Best fitness level	# of fuzzy rules
BLX-0.5	94.43%	32
EBB_2	96.11%	72

The EBB_2 outperformed the $BLX-0.5$ accuracy as it reached the 96.13% fitness level. However, the number of fuzzy rules is higher than the one obtained from the $BLX-0.5$, i.e., 72 fuzzy rules. Nevertheless, 72 fuzzy rules remain a very good simplification from the maximal size of 3125. The multi-combinative strategy is very successful in modeling the tool wear monitoring problem. The FKB obtained is accurate but still sufficiently simple for further human tuning.

7. Conclusion

The mutli-combinative strategies used in this paper helped to overcome some aspects of the premature convergence encountered in the automatic generation of fuzzy knowledge bases using genetic algorithms. Different ways to create new breed from the same pair of parents genes is a good way to improve the diversity. However, at the very late stages of the evolution, the presence of too many look-alike individuals is still a problem. We can also conclude that the exploration/exploitation balance influences the results. The rule to use is as follows: exploration at the early stages of the evolution, relaxing in the middle and exploitation at the end. It achieves better results than using the exploitation, exploration or relaxed exploitation only. The multi-combinative strategy also proved to be efficient when applied to experimental data. In order to improve the performances of our RBCGA, a new crossover mechanism that increases the number of fuzzy sets when the minimum of two fuzzy sets is reached on each premise can be added, which is a way to add more diversity to the population.

Acknowledgment

Financial support from the Natural Sciences and Engineering Research Council of Canada under grants of RGPIN-203618, RGPIN-105518 and STPGP-269579 is gratefully acknowledged.

References

1. Achiche S., Balazinski M., Baron L., 2003, Real/binary-like coded genetic algorithm to automatically generate fuzzy knowledge bases, *IEEE Fourth International Conference on Control and Automation*, 799-803

2. Achiche S., Balazinski M., Baron L., Jemielniak K., 2002, Tool wear monitoring using genetically-generated fuzzy knowledge bases, *Engineering Applications of Artificial Intelligence*, **15**, 303-314

3. Balazinski M., Czogala E., Jemielniak K., Leski J., 2002, Tool condition monitoring using artificial intelligence methods, *Engineering Applications of Artificial Intelligence*, **15**, 75-80

4. Balazinski M., Achiche S., Baron L., 2000, Influences of optimization and selection criteria on genetically-generated fuzzy knowledge bases, *International Conference on Advanced manufacturing Technology*, 159-164

5. BALAZINSKI M., BELLEROSE M., CZOGALAE., 1993, Application of fuzzy logic techniques to the selection of cutting parameters in machining processes, *International Journal for Fuzzy Sets and Systems*, **61**, 307-317

6. BARON L., ACHICHE S., BALAZINSKI M., 2001, Fuzzy decisions system knowledge base generation using a genetic algorithm, *International Journal of Approximate Reasoning*, 25-148

7. BONISSONE P.P., KHEDKAR P.S., CHEN Y.T., 1996, Gentic algorithms for automated tunning of fuzzy controllers, a transportation application, *Fifth International Conference on Fuzzy Systems (FUZZ-IEEE'96)*, 674-680

8. BRAD L., MILLER, 1996, Genetic algorithms with dynamic niche sharing for multimodal function optimization, *International Conference on Evolutionary Computation*

9. CASILLAS J., CORDÓN O., HERRERA F., 2000, A methodology to improve ad-hoc data-driven linguistic rule learning methods by inducing cooperation among rules, Technical Report, #DECSAI–000101, University of Granada, Spain

10. CASTELLANO G., ATTOLICO G., DISTANTE A., 1997, Automatic generation of fuzzy rules for reactive robot controllers, *Robitics and Autonomous Systems*, **22**, 133-149

11. EIBEN A.E., BÄCK T., 1997, An empirical investigation of multi-parent recombination operators in evolution strategies, *Evolutionary Computation*, **5**, 3, 347-365

12. ESHELMAN L.J., SCHAFFER J.D., 1991, Preventing premature convergence by preventing incest, *Proceedings of the Fourth International Conference on Genetic Algorithms*, 115-122

13. GOLDBERG D.E., 1989, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley

14. HAGRAS H., CALLAGHAN V., COLLEY M., CARR-WEST M., 1999, A fuzzy-genetic based embedded-agent approach to learning and control in agricultural autonomous vehicles, *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, Michigan, 1005-1010

15. HERRERA F., LAZANO M., 1997, Gradual distributed real-coded genetic algorithms, Technical Report, #DECSAI–97–01–03, Universty of Granada, Spain

16. HERRERA F., LOZANO M., 2000, Gradual distributed real-coded genetic algorithms, *IEEE Transactions on Evolutionary Computation*, **4**, 43-63

17. HERRERA F., LAZANO M., VERDEGAY J.L., 1995, Tunning fuzzy logic controllers by genetic algorithms, *International Journal of Approximate Reasoning*, **12**, 299-315

18. HOLLAND J.H., 1973, Genetic algorithms and the optimal allocation of trials, *SIAM Journal on Computing*, **2**, 2, 88-105

19. LI T.H., LUCASIUS C.B., KATEMAN G., 1992, Optimization of calibration data with the dynamic genetic algorithm, *Analytica Chimica Acta*, 123-134

20. MAHFOUD S.W., 1995, Niching methods for genetic algorithms, Ph.D. Thesis, University of Illinois at Urbana-Champaign

21. MICHALEWICZ Z., 1992, *Genetic Algorithms + Data Structure = Evolution Programs*, New York: Springer

22. MÄHLENBEIN H., SCHLIERKAMP-VOOSEN D., 1993, Predictive models for the breeder genetic algorithm: I. Continuous parameter optimization, *Evolutionary Computation*, **1**, 1, 25-49

23. NOMURA H., HAYASHI I., WAKAMI N., 1992, A self-tuning method of fuzzy reasoning by genetic algorithm, *Proceedings International fuzzy Systems and Intelligent Control Conference*, 236-245.

24. ONO I., KOBAYASHI S., 1997, A real-coded genetic algorithm for function optimization using uni-modal normal distributed crossover, *Proceedings of the 7th International Conference on Genetic Algorithms*, 246-253

25. TSUTSUI S., 1998, Multi-parent recombination in genetic algorithms with search space boundary extension by mirroring, *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature*, 428-437

26. TSUTSUI S., GHOSH A., 1998, A study on the effect of multi-parent recombination in real coded genetic algorithms, *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation*, 119-133

27. VALESCO J.R., LOPEZ S., MAGDALENA L., 1997, Genetic fuzzy clustering for the definition of fuzzy sets, *Proceedings of the Sixth IEEE International Conference On Fuzzy Systems, FUZZ IEEE*, **III**, 1665-1670

28. XIONG N., LITZ L., Reduction of fuzzy control rules by means of premise learning – method and case study, *Fuzzy Sets and Systems*, **132**, 217-231

29. ZADEH L.A., 1973, Outline of new approach to the analysis of complex systems and decisions processes, *IEEE Transactions of Systems, Man and Cybernetics*, **3**, 28-44

Multikombinacyjna strategia unikania przedwczesnej konwergencji w genetycznie generowanych rozmytych bazach wiedzy

Streszczenie

Rosnącej liczbie dziedzin, którymi zainteresowany jest przemysł, towarzyszą złożone zagadnienia wieloobiektowe. Dla takich zagadnień optymalne podejmowanie decyzji jest krytyczne. Często dla wsparcia procesu decyzyjnego w złożonych problemach

stosuje się układy logiki rozmytej. Kłopotem pozostaje jednak potrzeba manualnego wygenerowania bazy wiedzy poprzez eksperta. Okazuje się, że pewnym rozwiązaniem tego problemu może być użycie algorytmów genetycznych. Algorytmy takie modelują zagadnienie ewolucyjne na podstawie teorii Darwina. Głównym problemem w algorytmach genetycznych jest przedwczesna konwergencja, której próby wyeliminowania oparto na strategii multikombinowanych technik reprodukcji. Występują zasadniczo dwie drogi realizacji techniki reprodukcji: *Multiple Crossover on Multiple Parents* (MCMP) oraz *Multiple Crossovers Per Couple* (MCPC). Obydwie metody celują w wykorzystanie jak największej ilości informacji genetycznej od rodziców.

W artykule zajęto się możliwością ograniczania przedwczesnej konwergencji w rzeczywistym/binarnym kodzie genetycznym (RBCGA) używanym w automatycznym generowaniu rozmytych baz wiedzy (FKBs). Algorytm RBCGA stosuje kilka mechanizmów krzyżowania genów w odniesieniu do tej samej pary rodziców. Mechanizmy te przeróżnie kombinowane pozwalają na wielokrotną kreację potomstwa od tej samej pary rodziców. Koncepcja dużej rodziny i różnicowanie krzyżowania powinny wprowadzić dywersyfikację nowogenerowanych pokoleń, które w przeciwnym razie szybko uległyby konwergencji. Zapobieżenie temu zjawisku poprzez strategię multikombinacyjną utrzymuje proces poszukiwania rozwiązania w stanie aktywnym.

Manuscript received March 2, 2004; accepted for print April 13, 2004