© International Journal of Combinatorial Optimization Problems and Informatics, Vol. 8, No. 1, Jan-April 2017, pp. 6-11. ISSN: 2007-1558.

Self-adjusting a Genetic Algorithm Using Fuzzy Logic Techniques

Mario César López-Locés[1], Hector Joaquín Fraire-Huacuja[1], Rodolfo Pazos Rangel[1], Juan J. González Barbosa[1], Jesús David Terán Villanueva[1]
[1]Tecnológico Nacional de Mexico, Instituto Tecnológico de Ciudad Madero
E-mail: mariocesar@lopezloc.es

Abstract. One of the most important tasks in approximately solving an optimisation problem is to adjust the parameters of the metaheuristic used as a solution method. As the metaheuristics are usually general in purpose, it is necessary to make adjustments to them for each optimisation problem to which they are applied to get high-quality solutions. In this paper, we propose the use of a Type 1 Fuzzy Inference System and a Type 2 Fuzzy Logic Inference System to select the operators of a Genetic Algorithm during execution time to solve a set of ten test functions from the literature. The results of computational experiments show that the fuzzy selection of operators improves the performance of the original GA on nine of the ten test functions with practically the same execution time.

Keywords: fuzzy logic, parameter tuning, genetic algorithm

1 Introduction

Exact optimisation methods may take a long time to solve large instances of an NP-hard problem. Metaheuristics are approximation methods that offer good solutions in reasonable times. As these methods are general in purpose, their success in solving a problem depends to a large degree on the configurations of their parameters. Commonly, the parameter configuration task is performed manually, which tends to be laborious and inefficient, but through time some alternatives have been proposed to automatise this task. In this article, we propose the use of Fuzzy Inference Systems (FIS) to perform this configuration.

The use of FIS for tuning the parameter of a metaheuristic has been reported before. A survey presented by Valdez et al. [1] presents a landscape of metaheuristics such as Particle Swarm Optimisation (PSO), Genetic Algorithms (GA) and Ant Colony Optimisation (ACO). Those metaheuristics use FIS to configure the value of their parameters during execution time on tasks such as the control of a flow production system, the diagnosis of diabetes and the optimisation of wireless networks.

In this paper, we propose the use of FIS on a GA, not to adjust the value of some of its parameters, such as mutation probability or population size, but to change the operators themselves during execution time. Reassembling the GA with different constituent parts allowed it to explore different regions of the search space, helping it to escape from local optima, thus increasing the probability of finding better quality solutions.

The organisation of this document is as follows. Section 2 presents the description of the GA and the operators that compose it. Section 3 presents a description of the two FIS methods and how they were incorporated into the GA to control the Mutation and Crossover operators. Section 4 describes the computational experiment made to evaluate the performance of the GA and its fuzzy variants along with the results obtained. Finally, Section 5 describes the conclusions reached in this work.

2 Static Genetic Algorithm

GAs were formalised in the form that is known today by John Holland during the 1970s [2], taking inspiration from the Theory of Evolution by Natural Selection, conceived by Charles Darwin during his famous expedition on the Beagle.

Received Jun 19, 2016 / Accepted Dec 10, 2016

The Theory of Evolution by Natural Selection states that given a population of individuals which are submitted to the pressure of the environment, those individuals who are able to survive for long enough to produce offspring are considered the fittest. Those individuals are the ones that will determine the future of the population, transmitting to their offspring those characteristics that allow them to survive, as long as the environment remains constant.

In the context of computer science, the environment represents a particular optimisation problem that is sought to be solved. Each individual represents a solution for that problem, with an associated fitness value that determines its quality. Taking as a reference the processes that are observed in nature, such as the strategies followed by the members of different species to select potential partners, the different kinds of reproduction, and the mutations that occur naturally in the genetic code of all living things, the GA uses analogous methods to travel across the search space.

Algorithm 1 shows the basic structure of a GA. The Selection operator chooses the individuals from the population that will be used as parents, using strategies that guarantee the diversity of the population and the quality of the new solutions generated. The Crossover operator combines the individuals obtained with the Selection operator to form the offspring. Finally, the Mutation operator makes random changes in the offspring to explore new regions of the search space that are not so easily reached with the Crossover operator.

The GA process consists of applying selection, crossover and mutation operators to the population and evaluating the newly generated individuals to determine if they produce an improvement, in which case, they substitute the individuals with the worst fitness in the population.

Algorithm 1: Genetic Algorithm Structure

Description: Metaheuristic inspired by the evolution by natural selection process.

Output:
Solution: *bestSolution*

Initialise *population* with random candidate solutions.
While *currentEvaluation* < *maxEvaluations*:
    (Selection)
    Choose *parent1* and *parent2* from *population* with *selectionOperator*()
    (Crossover)
    Get *offspring* by applying *crossoverOperator*() to *parent1* and *parent2*
    (Mutation)
    Mutate *offspring* with *mutationOperator*()
    Evaluate *fitness*() of *offspring*
    If the *fitness* of *offspring* is better than the one of *worstSolution* Then
        Remove *worstSolution* from *population*
        Add *offspring* to *population*
    End If
End While
Return *bestSolution* from *population*

Given that the structure of this algorithm is generic to all kinds of problems, the parameters and the methods used as operators of the GA must be adjusted to solve a particular optimisation problem.

3 Fuzzy Control of the Operators of the Genetic Algorithm

We propose the use of FIS for the selection of the operators that comprise a GA. The FIS were originally proposed by Zadeh in 1965 [3]. These systems can take continuous values that represent varying degrees of truth, in contrast to those more commonly used in computer science which use discrete values, (e.g. 1 or 0, true or false).

This characteristic is of great importance because the presence of uncertainty makes it very difficult to assign absolute values to a given variable. For example, a sunny day at 45°C is hot for nearly all people, while the opinion is more divided when the temperature descends to 25°C which can be interpreted by some as hot and by others as perfect. The same situation occurs with a metaheuristic algorithm whose performance depends on subjective values, such as if a number of iterations without improvement indicates that the algorithm is stagnated or if there is still hope for improvement, if the fitness of an individual in the population is good or bad, or if two individuals are similar or different.

As uncertainty prevails to some degree in all kinds of activities and tasks, the applicability of fuzzy logic is almost ubiquitous. Fuzzy logic can be used to determine the quality of water of a farming pool [4]; to filter signals, image and video in real time [5][6]; to control the movement of a robot[7]; to assist in the interpretation of medical records [8]; to maximise the quality of crops while minimising the resources used to grow them[9] and for many other applications.

To determine if the use of a FIS is able to improve the performance of a GA, two different FIS methods were implemented: a Type 1 FIS and a Type 2 FIS. The Type 2 FIS allows diminishing the grade of uncertainty by blurring the frontiers of the variables of the membership functions, compared with the Type 1 FIS that are well defined. Both FIS were used to control the selection of the Crossover and Mutation operators.

Figure 1 shows the membership functions for the Type 1 FIS. The Type 1 FIS takes as input the fitness value mean, the number of iterations without improvement, and the diversity of the individuals in the population to determine if a change to the Crossover (Mutation) operator used by the GA is advisable.

.

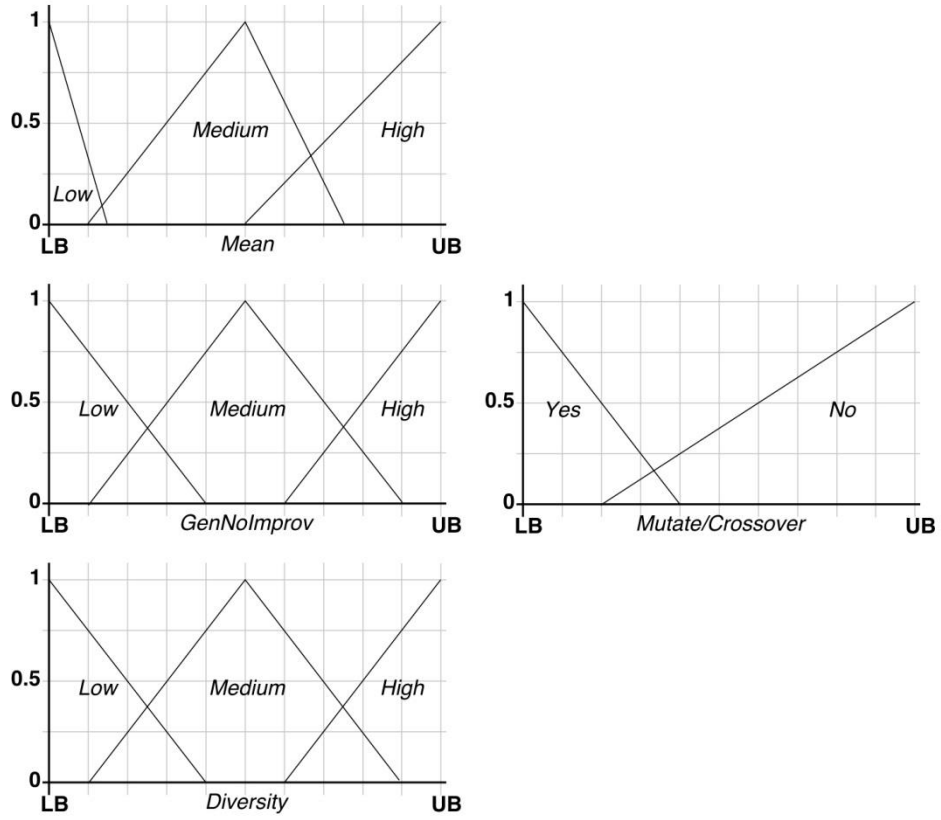

Figure 1. Input and output variables of a Type 1 Fuzzy Inference System

The FIS shown in Figure 2 takes as input the same values as shown in Figure 1, but for a Type 2 FIS. A Type 2 FIS tries to reduce uncertainty by blurring the frontiers of the membership functions.

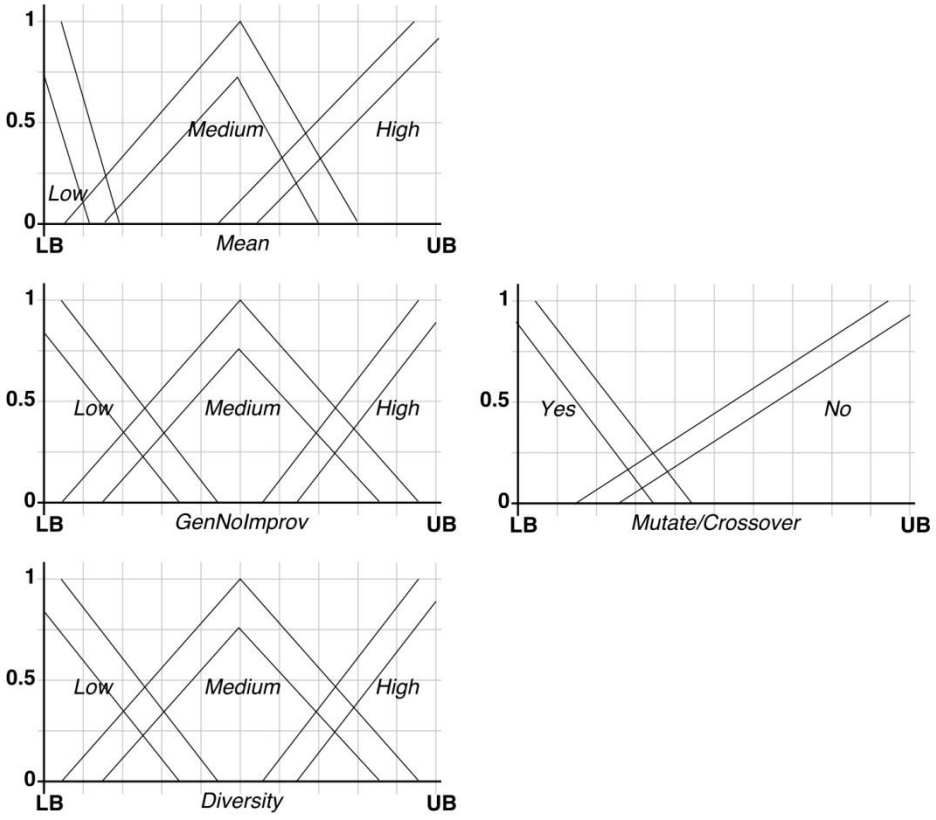

Figure 2. Input and output variables of a Type 2 Fuzzy Inference System

The rules used by both FIS were as follows:

IF Mean is Low THEN Change is Small
IF Mean is Medium THEN Change is Moderate
IF Mean is High THEN Change is Big
IF Mean is NOT Low AND GenNoImprov is High AND Diversity is Low THEN Change is Big
IF Mean is Medium AND GenNoImprov is Medium AND Diversity is Medium THEN Change is Moderate
IF Mean is Low AND GenNoImprov is Low AND Diversity is Medium THEN Change is Small
IF Mean is Low AND GenNoImprov is Low AND Diversity is Low THEN Change is Moderate

The output returned by the FIS is the probability of change of a given operator, Change, which determines whether it's advisable to continue to use the same method of an operator or to change to a different one.

4 Computational Experiments and Results

The hardware specifications of the computers where the experimentation was performed consisted of an Intel i5 processor at 2.0 GHz, 8 GB of LPDDR3 RAM at 1867MHz. and an SSD of 256 GB. The operating system used was Mac OS X 10.12.2.

The GA used for this implementation was taken from the optimisation library from MATLAB 2014, and both FIS were deployed using the fuzzy and it2fuzzy libraries. The stop condition was a fixed number of generations, 200 for the three algorithms.

We used the tests functions shown in the first column of Table 1 to evaluate the performance of the three GA varieties. The selected functions are commonly used to test the performance of algorithms applied to the solution of optimisation problems [10]. The three algorithms were executed 100 times for each test function to guarantee the statistical significance of the results

[11]. We compared the performance of the Static GA against the two GAs that included the Type 1 and Type 2 FIS to determine which one found better quality solutions.

The results obtained during the experimentation are shown in Table 1. The first column shows the test functions evaluated. Columns two, three and four represent the three variants of the GA evaluated: the static GA, the GA with a Type 1 FIS (Fuzzy1GA), and the GA with a Type 2 FIS (Fuzzy2GA). Each one of those columns is divided into two sub-columns: the first one contains the objective value (*OV*) of the solution calculated by each algorithm for the corresponding problem. A cell of this column is the arithmetic mean of this value during the 100 repetitions of the experiment. For each problem, the best *OV* is highlighted in bold. In all cases, a lower *OV* means a better performance. The second sub-column shows the accumulated time in seconds that the algorithm took to solve the 100 repetitions of the experiment. The last column shows the global minimum of the optimal value of each test function problem.

As it can be observed, the two fuzzy GAs outperform in quality the original variant. Of the two proposed methods, Fuzzy1GA obtained the best results, with a better quality and comparatively the same amount of computational time consumed. The original GA outperformed the fuzzy variants only in the test with the Rosenbrock function.

Table 1. Results

	GA		***Fuzzy1GA***		***Fuzzy2GA***		***Global Minimum***
	OV	*Time*	*OV*	*Time*	*OV*	*Time*	
Ackley's function	3.11	19.6	**3.08**	20.9	3.10	20.1	0
Beale's function	0.216	14.7	0.184	14.4	**0.138**	14.9	0
Booth's function	0.145	15.9	**0.134**	16.3	0.158	17.1	0
Bukin function N.6	6.69	11.6	6.28	12.7	**6.11**	12.4	0
Goldstein–Price function	69.0	14.8	**38.0**	14.5	49.2E	15.4	3
Lévi function N.13	0.155	15.5	**0.113**	15.9	0.153	16.0	0
Matyas function	0.01	15.0	0.0112	15.2	**0.00858**	15.0	0
Rastrigin function	1.07	18.5	**1.00**	18.3	1.01	18.0	0
Rosenbrock function	**1.16**	12.6	1.60	13.0	1.93	12.6	0
Sphere function	0.0221	15.0	0.0132	16.2	**0.0106**	17.0	0

Table 2 shows the results of the Wilcoxon–Mann–Whitney hypothesis test that was used to determine if the differences in the results obtained by the best algorithm shown in Table 1 were statistically significantly different from those obtained by the two other algorithms. The Null Hypothesis H0 was that there is no statistically significant difference between two samples. The numbers shown in Table 2 are the *p*-values calculated by this test. A *p*-value lower or equal to the significance value α=0.05 means that H0 is rejected.

The results of Table 2 show that for the Sphere function Fuzzy2GA was statistically different than the GA, hence it can be inferred that the quality of the solution was also significantly better. For the other test functions, even when the two GA with FIS performed empirically better, we cannot claim that the differences are also statistically different with the significance value selected. However, is to be noted that in most cases when one of the fuzzy variants outperformed the original GA, the *p*-value calculated is particularly low.

Table 2. Statistical Significance

	Best Algorithm	***Vs GA***	***Vs Fuzzy1GA***	***Vs Fuzzy2GA***
Ackley's function	*Fuzzy1GA*	0.100751572	1	0.356804258
Beale's function	*Fuzzy2GA*	0.279607081	0.962971901	1
Booth's function	*Fuzzy1GA*	0.348107414	1	0.560061021
Bukin function N.6	*Fuzzy2GA*	0.240372482	0.479340137	1
Goldstein–Price function	*Fuzzy1GA*	0.321175983	1	0.77777467
Lévi function N.13	*Fuzzy1GA*	0.056508556	1	0.116154032
Matyas function	*Fuzzy2GA*	0.478581778	0.55268279	1
Rastrigin function	*Fuzzy1GA*	0.900828921	1	0.94642698
Rosenbrock function	*GA*	1	0.500847465	0.119311968
Sphere function	*Fuzzy2GA*	**0.000131346**	0.191553997	1

5 Conclusions

In this paper, we proposed a GA with FIS that changes its structure dynamically during execution time. The approaches that we found in the literature use FIS to adjust the value of the parameters, varying only in the intensity or amount. Our approach changes the methods that are used as mutation and crossover operators to move to a different neighbourhood which allows it to escape from local optima and explore regions of the search space that could be more promising.

The results of the computational experiments are encouraging because it shows that the fuzzy selection of operators improves the performance of the original GA in nine of the ten test functions used with practically the same execution time.

An additional advantage in the use of FIS in the context of this article is that it frees the developers from most of the burden of performing a manual parameter tuning of the GA, which in turn could be used to adjust other metaheuristics during execution time.

With a fixed number of generations, we showed that Fuzzy algorithms obtained better results than the original version with a negligible consumption in time, which was on average similar for the three methods.

Future works include the dynamic configuration of the rules to improve the performance of the two Fuzzy GA methods, while reducing the computational time consumed, and transferring this technique to other metaheuristics to determine if they yield similar results compared with their original variants on these and other optimisation problems.

Acknowledgements

The authors would like to acknowledge with appreciation and gratitude the CONACyT, TecNM, PRODEP and the Öbex Project. This work has been partially supported by CONACyT Projects 244248 (Mario C. López-Locés) and 177007 (J. David Terán-Villanueva).

References

[1] F. Valdez, P. Melin, and O. Castillo, "A survey on nature-inspired optimization algorithms with fuzzy logic for dynamic parameter adaptation," *Expert Systems with Applications*, vol. 41, no. 14, pp. 6459–6466, Oct. 2014.

[2] J. H. Holland, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*, 1st MIT Press ed. Cambridge, Mass: MIT Press, 1992.

[3] L. A. Zadeh, "Fuzzy Sets," *Information and Control*, vol. 8, pp. 338–353, 1965.

[4] J. J. Carbajal Hernández and L. P. Sánchez Fernández, "Diagnóstico y predicción del hábitat en la camaronicultura," *Computación y Sistemas*, vol. 17, pp. 435–455, 2013.

[5] J. C. García Infante, J. de J. Medel Juárez, and P. Guevara López, "Filtrado Digital Difuso en Tiempo Real," *Computación y Sistemas*, vol. 11, pp. 390–401, 2008.

[6] B. Akay and D. Karaboga, "A survey on the applications of artificial bee colony in signal, image, and video processing," *Signal, Image and Video Processing*, vol. 9, no. 4, pp. 967–990, May 2015.

[7] O. Castillo, H. Neyoy, J. Soria, P. Melin, and F. Valdez, "A new approach for dynamic fuzzy logic parameter tuning in Ant Colony Optimization and its application in fuzzy control of a mobile robot," *Applied Soft Computing*, vol. 28, pp. 150–159, Mar. 2015.

[8] U. Dev, A. Sultana, D. Saha, and N. Mitra, "Application of fuzzy logic in medical data interpretation," *Bangladesh Journal of Scientific and Industrial Research*, vol. 49, no. 3, Feb. 2015.

[9] F. Viani, "Experimental validation of a wireless system for the irrigation management in smart farming applications," *Microwave and Optical Technology Letters*, vol. 58, no. 9, pp. 2186–2189, Sep. 2016.

[10] H. Mühlenbein, M. Schomisch, and J. Born, "The parallel genetic algorithm as function optimizer," *Parallel Computing*, vol. 17, no. 6–7, pp. 619–632, Sep. 1991.

[11] S. M. Ross, *A first course in probability*, 5th ed. Upper Saddle River, N.J: Prentice Hall, 1998.