Evolutionary Algorithms and Fuzzy Logic: A two-way integration

Andrea G. B. Tettamanzi*
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
Via Comelico 39/41, I-20135 Milano, Italy

Summary

Abstract

Two ingredients of soft computing, evolutionary computing and fuzzy logic can be combined in a way that makes them benefit from one another. An evolutionary algorithm can evolve fuzzy systems, while fuzzy logic can be used to control evolution to speed up convergence to a global optimum and escape from local optima. Besides, concepts that are inherent in the workings of evolutionary algorithms, like fitness, can be fuzzified, thus taking advantage of a tolerance for imprecision in order to save computational resources.

These ideas are demonstrated in an evolutionary algorithm for fuzzy controller synthesis and optimization.

1 Introduction

Integration of evolutionary algorithm and fuzzy logic can happen in three complementary forms.

The most obvious one exploits the optimum searching ability of evolutionary algorithms to synthesize and optimize a fuzzy system, such as a fuzzy rule set or a neuro-fuzzy network. This combination has already been investigated by several authors (see for example [4, 12, 6, 7, 3]) and has given quite promising results.

Evolutionary algorithms are relatively easy to implement and in general their performance tends to be rather satisfactory in comparison with the small amount of knowledge about the problem they need in order to work; however, typically they would require some sort of human supervision during their use as practical tools.

The idea is then to use a fuzzy knowledge base (or *fuzzy government* [1]) to detect the emergence of a solution, dynamically tune algorithm parameters, as in [5] or [2], and monitor the evolutionary process to avoid undesired behaviors such as premature convergence.

A third option is to embed fuzziness into the evolutionary algorithm itself. For instance, some precision in the calculation of fitness could be sacrificed to save computational resources by defining a fuzzy fitness, or even fuzzy alleles for genes, as in [8], thus fuzzifying also genetic operators.

*The author was sponsored in part by SGS-Thomson Microelectronics

2 An Application to Fuzzy Control

How this order of ideas can be put to use in practice can be illustrated through an application of evolutionary algorithms to the synthesis and optimization of fuzzy controllers implemented by the author [11].

The algorithm described devises programs for the Weight Associative Rule Processor (W.A.R.P.), a fuzzy processor manufactured by SGS-Thomson Microelectronics and dedicated to control applications, supporting antecedent membership functions with arbitrary shape, up to 256 rules, each comprising up to four antecedent and one consequent clauses [9].

An encoding that directly maps to the machine level of W.A.R.P.'s programming language has been defined, and *ad hoc* mutation and crossover operators have been designed to deal with it.

Everything, from the number and structure of rules to the number and shape of membership functions, is left to be discovered by the evolutionary process. In this sense, the only constraints for the controllers evolved are those imposed by the processor's hardware implementation.

A model for selection based on the concept of *competition* [10] is employed, which circumvents the difficult and often critical task of defining an appropriate fitness function. Instead, very simple and general rules are used to determine the winner of a stochastic tournament between two controllers: Assuming that failure and success conditions of a simulation are defined, an initial condition is randomly generated and the simulations for both competing controllers are carried out in parallel until one of them fails or succeeds; the competition is then won by the controller that succeeds or by the one that does not fail. Despite its extreme simplicity, this selection model is sufficient for evolution to take place.

3 The Fuzzy Fitness

A fuzzy estimate of the fitness of individuals is given based on the recorded outcomes of competitions. This estimate is based on three quantities: The number c of competitions undergone by an individual, the number w of its wins and the number s of successes. The more competitions an individual has gone through, the less fuzzy its fitness will be.

The membership function of fitness, for a given individual, is defined as:

$$\mu_f(x) = N(a,b)x^a(1-x)^b,$$

where

$$N(a,b) = \frac{(a+b)^{a+b}}{a^a b^b}$$

is a normalization factor and $a = w + s$, $b = c - s$. The mode of the membership function, where its value is one, is

$$\arg\max_{x\in[0,1]} \mu_f(x) = \frac{a}{a+b} = \frac{w+s}{w+c}.$$

An interesting predicate defined on pairs of individuals is their *fitness overlap:*

$$f_\wedge(\gamma,\kappa) = \max_{x\in[0,1]} \min\{\mu_{f(\gamma)}(x), \mu_{f(\kappa)}(x)\},$$

i.e. the maximum of the intersection of their fitness. This gives the degree to which γ and κ have the same fitness.

The fuzzy fitness is employed to avoid carrying out useless competitions: When the fitness overlap of the two opponents is below a certain threshold, the one with highest mode is awarded the match.

Besides, fuzzy fitnesses of individuals in the population are aggregated to yield statistics such as maximum, average and minimum fitness for use by the fuzzy government, as explained in the next section.

4 The Fuzzy Government

The algorithm has facilities for writing fuzzy rules involving population statistics in their antecedent parts and algorithm parameter values in their consequent parts. At regular intervals, the statistics are computed and an inference is carried out on the rules defined by the user. The algorithm parameters are then updated accordingly and evolution is resumed.

Statistics that can be used are: Genotypic and phenotypic diversity indices[1], maximum, average and minimum fitness, the fitness range, the tie rate, defined as the fraction of competitions that end in a tie, the success rate, i.e. the fraction of competitions that end because either opponent reached the target, the actual mutation rate and the time-out rate, that is the fraction of competitions that timed out.

The algorithm parameters that can be acted upon are: Mutation and crossover rates, selective pressure, the time a controller must spend in a target state in order to be considered successful, the degree to which a satisfactory solution has been obtained and the degree to which evolution is stuck. The latter two parameters are used to decide whether to stop the algorithm.

Statistics and parameters are in part universal to any evolutionary algorithm and in part specific to a particular application. Therefore it is hard to state general rules to control the evolutionary process. These are to be discovered by the implementer by means of experiments and empirical observations.

[1]A genotypic diversity index is one that takes into account the genes of individuals, whereas a phenotypic diversity index is based on the fitness of individuals or other features that are relevant to the problem to be solved

4.1 Empirical observations

Since the aim of the fuzzy government is to keep the evolutionary process in a dynamical equilibrium, its overall structure will be based on negative feedback as a guiding principle.

For instance, as a general rule, the mappings between population statistics and algorithm parameters defined by a fuzzy government will inversely relate the mutation rate with diversity (the less diverse the population, the higher the mutation rate).

For the application described, it has been observed that convergence to an optimal controller is foreshadowed by an increase of the number of competitions that time out. That is because both opponents are so successful in controlling the systems that they reach the target almost at the same time and neither fails until the whole simulation time has elapsed. Therefore, the time out ratio is in a direct relation with emergence.

From several experiments, it appears that the population maintained by the algorithm improves at the fastest pace when the success rate is not too high. As a consequence, the time a controller must spend in a target state in order to be considered successful should be linked to the success rate: An increase in the success rate will thus trigger an extension of that time, which in turn will lead to a decrement of the success rate. The rules have to be tuned to attain the most advantageous equilibrium point.

5 Conclusions

Three ways in which evolutionary algorithms and fuzzy logic can be combined have been presented and illustrated by means of a sample application. The spirit of this work is in line with the principles of soft computing, i.e. to exploit the tolerance for imprecision to achieve tractability and low solution cost, and to employ different techniques, like evolutionary algorithms and fuzzy logic, in a complementary way.

References

[1] S. Arnone, M. Dell'Orto, and A. Tettamanzi. Toward a fuzzy government of genetic populations. In *Proceedings of the 6th IEEE Conference on Tools with Artificial Intelligence (TAI'94)*, Los Alamitos, CA, 1994. IEEE Computer Society Press.

[2] A. Bergmann, W. Burgard, and A. Hemker. Adjusting parameters of genetic algorithms by fuzzy control rules. In K.-H. Becks and D. Perret-Gallix, editors, *New Computing Techniques in Physics Reserach III*. World Scientific Press, Singapore, 1994.

[3] C. Z. Janikow. A genetic algorithm for learning fuzzy controllers. In *Proceedings of the ACM Symposium on Applied Computing*, New York, 1994. ACM Press.

[4] C. L. Karr. Design of an adaptive fuzzy logic controller using a genetic algorithm. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA, 1991. Morgan Kaufmann.

[5] M. Lee and H. Takagi. Dynamic control of genetic algorithms using fuzzy logic techniques. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo, CA, 1993. Morgan Kaufmann.

[6] M. Lee and H. Takagi. Embedding apriori knowledge into an integrated fuzzy system design method based on genetic algorithms. In *Proceedings of the 5th IFSA World Congress (IFSA'93)*, pages Vol. II, 1293–1296, July 4–9 1993.

[7] M. Lee and H. Takagi. Integrating design stages of fuzzy systems using genetic algorithms. In *Proceedings of the 2nd International Conference on Fuzzy Systems (FUZZ-IEEE'93)*, pages Vol. I, 612–617, 1993.

[8] E. Sanchez and P. Pierre. Fuzzy logic and genetic algorithms in information retrieval. In *3rd International Conference on Fuzzy Logic, Neural Networks and Soft Computing*, 1994.

[9] SGS-Thomson Microelectronics. *W.A.R.P. Software Development Tool User's Guide*, version 1.0 edition, 1994.

[10] A. Tettamanzi. A distributed model for selection in evolutionary algorithms. Technical Report 110-94, Dip. di Scienze dell'Informazione – Università degli Studi di Milano, Milano, Italy, 1994.

[11] A. Tettamanzi. An evolutionary algorithm for fuzzy controller synthesis and optimization. In *To appear in the Proceedings of the 1995 IEEE International Conference on Systems, Man and Cybernetics*, Vancouver, BC, Canada, October 22–25 1995.

[12] P. Thrift. Fuzzy logic synthesis with genetic algorithms. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, San Mateo, CA, 1991. Morgan Kaufmann.