

A Retrospective View of Fuzzy Control of Evolutionary Algorithm Resources

Raj Subbu, Piero P. Bonissone

General Electric Global Research, One Research Circle, Niskayuna, NY 12309, U.S.A.

subbu@research.ge.com, bonissone@research.ge.com

Abstract—A retrospective argument supporting the adaptive control of the resources of an evolutionary algorithm is presented. In this approach, a fuzzy controller monitors the population diversity and maturity of the evolutionary process, and issues control actions to adapt the algorithm's population size and mutation rate. The fuzzy controller's rule base is simple and intuitive, and encodes expert heuristic knowledge on the management of an evolutionary algorithm's resources. Analysis based on experimentation and statistical hypothesis testing reveals that such an adaptive approach significantly reduces the variance in search performance of the evolutionary algorithm. Experiments also reveal that the approach in general improves search performance of the algorithm.

I. INTRODUCTION

EVOLUTIONARY algorithms have proven highly effective for achieving optimal or near-optimal solutions to many complex real-world optimization problems. These algorithms frequently require the specification of a few canonical control parameters such as population size, crossover rate, and mutation rate to achieve good search performance. Selection of an evolutionary algorithm's parameters is often left to the intuition and experience of the user, and it is generally acknowledged that the relationship between parameter values and search performance is complex and not completely understood.

Early work due to De Jong [4] in the area of genetic algorithms popularized the use of a static parameter set, and these suggested parameters were generally accepted for several years by researchers and practitioners alike. Later work by Grefenstette [6] in which he utilized a meta genetic algorithm to search for optimal control parameters for the object-level genetic algorithm supported De Jong's early work. However, recent insight due to the *no free lunch theorem* [15] suggests that any one set of potentially optimal algorithm parameters cannot be expected to result in consistently superior performance over the entire space of optimization problems, as was previously generally accepted. Though this theoretical result could be used to discourage the search for a good general set of algorithm parameters, there is a wide body of experimental evidence that supports the online adaptation of

parameters for improved search performance [5]. For instance, it is widely acknowledged that employing a systematically less aggressive mutation operation as the search matures facilitates a smooth transition from an exploratory search to a more exploitative search. It is also obvious that assuming sufficiently diverse individuals are created, increasing the population size during evolution increases the population diversity, and reducing the population size has the obverse effect. When a population is sufficiently diverse it may be unnecessary to maintain a large population. This is relevant especially for numerous real-world applications where fitness evaluations need to be conserved since they are highly time consuming, expensive, or both. *Intelligent management* of an evolutionary algorithm's mutation rate and population size thus serve as key motivators for this work.

Fuzzy control schemes are particularly suited to environments that are either ill defined or are very complex. When a reasonable amount of expert heuristic knowledge about system behavior is available, fuzzy logic serves as a suitable candidate to efficiently encode and utilize this knowledge. Adaptive control of an evolutionary algorithm's parameters is agreeably a difficult task [14], but one for which a fair amount of expert heuristic knowledge exists. Such a task may indeed benefit from the use of fuzzy logic control. In this paper, we focus on the design and validation of a simple and intuitive fuzzy logic controller that encodes expert heuristic knowledge on evolutionary algorithm behavior. This controller intelligently manages the population resources and mutation rate of the evolutionary algorithm during run-time.

This paper is based on previously unpublished extensive experimentation conducted in September-October 1998, and was motivated by our earlier published findings [12]. The principal finding of our current work based on statistical hypothesis testing is that a fuzzy controlled adaptive evolutionary algorithm has a much tighter variance in search performance, and is a significant improvement over the algorithm based on a static canonical parameter set. Achieving a higher degree of search confidence given a finite number of potentially expensive fitness evaluations is an important benefit.

The rest of this paper is organized as follows. In Section II, we present background information on fuzzy-evolutionary algorithms and the object-level problem domain. In Section III, we describe the architecture of the test bed we have implemented. In Section IV, we present simulation results

The authors acknowledge the support of General Electric during the preparation of this manuscript.

All experiments leading to this work were conducted during 1998 in the Electronics Agile Manufacturing Research Institute (EAMRI) at Rensselaer Polytechnic Institute, Troy, NY. The EAMRI is funded in part by the National Science Foundation.

using a variety of object-level search spaces. We conclude in Section V.

II. BACKGROUND

In this section, we present background information on fuzzy-evolutionary algorithms, and describe the agile manufacturing planning decision problem domain used for generating object-level problems for the evolutionary search.

A. Fuzzy-Evolutionary Algorithms

Numerous researchers have explored the integration of fuzzy reasoning and evolutionary algorithms. Lee and Takagi [10] use a high-level genetic algorithm to identify and tune the rules of a fuzzy logic controller that is used to control the parameters of an object-level genetic algorithm. The object-level algorithm optimizes the five sample problems suggested by De Jong [4]. They use a binary encoding for each three-parameter representation of a triangular membership function. They simultaneously identify the appropriate output rules and shapes of membership distributions. Their fuzzy logic controller is determined by three inputs (*Average Fitness/Best Fitness*, *Worst Fitness/Average Fitness*, and Δ *Best Fitness*), and three outputs (Δ *Population Size*, Δ *Crossover Rate*, and Δ *Mutation Rate*). They constrain the outputs to vary within $\pm 50\%$ of their previous values, and also set overall limits for the outputs to the operational ranges [2, 160], [0.2, 1.0], and [0.0001, 1.0] respectively. They evaluate rule choices and membership function shape performance on the sample object-level problems using as evaluation criteria the averaged *on-line performance*, and averaged *off-line performance* [4].

Subbu et al. [12] present a comparison of the performance of a fuzzy logic controlled genetic algorithm (FLC-GA) and a parameter-tuned genetic algorithm (TGA) for an agile manufacturing application. These strategies are benchmarked using a genetic algorithm (GA) that utilizes a canonical static parameter set. In the FLC-GA, fuzzy logic controllers dynamically schedule the population size, crossover rate, and mutation rate of the object-level GA using as inputs diversity (genotypic, and phenotypic) measures of the population. A fuzzy knowledge base is automatically identified using a meta-GA. In the TGA, a meta-GA is used to determine an optimal static parameter set for the object-level GA. The object-level GA supports a global evolutionary optimization of design, supplier, and manufacturing planning decisions for realizing printed circuit assemblies in an agile environment. The authors demonstrate that high-level control system identification (for the FLC-GA) or tuning (for the TGA) performed with small object-level search spaces can be extended to more elaborate object-level search spaces, without employing additional identification or tuning. The TGA performs superior searches, but incurs large search times. The FLC-GA performs faster searches than a TGA, and is slower than the GA that utilizes a canonical static parameter set. However, search quality measured by the variance in search performance of the FLC-GA is comparable to that of the GA that utilizes a canonical static parameter set. This latter negative result served as the key motivation for investigating

the alternative, less complex approach discussed in this paper. It is our opinion that searching for a knowledge base for a fuzzy controller used to control the parameters of the object-level algorithm via a meta-level algorithm not only adds a second layer of complexity but also generates control surfaces that do not generally correspond with expert knowledge. Evolutionary algorithms on the other hand have been successfully used to tune the performance of an existing fuzzy knowledge base, since this is typically a much smaller problem space.

In [3], the authors present an extensive bibliography of papers that discuss fuzzy-evolutionary algorithms for a variety of applications. Herrera et al. [9] use an evolutionary algorithm to tune each rule used by a fuzzy logic controller. Herrera and Lozano [8] present a detailed summary of a variety of approaches for adapting the parameters of an evolutionary algorithm using fuzzy logic controllers, and recently in [7], the authors present an approach where rule bases for the fuzzy logic controllers simultaneously *coevolve* with the object-level evolutionary algorithm. Tettamanzi and Tomassini [13] (Chapter 7) discuss several combinations of fuzzy logic and evolutionary algorithms that include the development of fuzzy rule-based systems for adaptive control of an evolutionary algorithm.

B. Application Domain

In this subsection, we describe the agile manufacturing environment and the printed circuit assembly manufacturing planning decision problem. For additional details, the reader is referred to [11].

1) Agile Manufacturing

The rapidly changing global marketplace demands innovative products that are more compact, of lower cost, and of higher quality. The necessity that enterprises rapidly respond to these dynamic changes places increased emphasis on achieving highly adaptive manufacturing with closer coupling between design and manufacturing activities. Such adaptive manufacturing practices are broadly characterized as *agile manufacturing*.

2) Problem Description

In the printed circuit board industry, designers face increased complexity in coordinating suppliers and manufacturers of sub-assemblies and components to be competitive in cost, time, and quality. Design, supplier, and manufacturing decisions are made using the experience base in an organization and are often difficult to adapt to changing needs. Systems that support a coupling among design, supplier, and manufacturing decisions, simultaneously reduce supply and manufacturing costs and lead times, and better utilize available manufacturing facilities are critical to improved performance in these industries. A method and architecture for optimal design, supplier, and manufacturing planning for printed circuit assemblies is presented in [11]. The problem formulation from this reference is used as a basis for

generating object-level test problems in this paper. This formulation poses the optimal selection of designs that realize a given functional specification, the selection of parts to realize a design, the selection of suppliers to supply these parts, and the selection of a production facility to manufacture the chosen design, as a global optimization problem where each selection has the potential to affect other selections. The goal is to minimize an aggregate non-linear objective function of total cost and total time, $\min J_{ij} = \mathfrak{F}(Cost_{ij}, Time_{ij})$, of the i^{th} design¹ assigned to the j^{th} manufacturing facility. The total cost and total time for realizing a printed circuit assembly are each coupled non-linear functions dependent on characteristics of a chosen design, parts supply chain characteristics, and characteristics of a chosen manufacturing facility. This decision problem is in the class of *nonlinear discrete assignment problems*, and its characteristics do not support optimization using traditional techniques based on mathematical programming. An evolutionary optimization technique is more easily applied to this problem domain, is robust, and simultaneously searches multiple solutions.

III. SYSTEM ARCHITECTURE

The architecture for adaptive fuzzy control of an evolutionary algorithm's resources appears in Figure 1. During evolutionary search the fuzzy logic controller observes the population diversity and percentage of completed trials, and using the embedded expert knowledge base (KB), specifies changes to the population size and mutation rate.

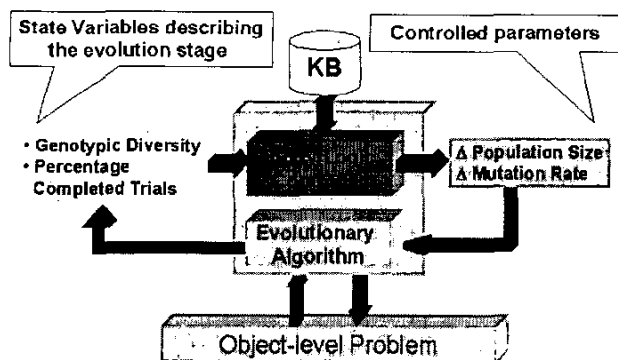


Figure 1: Architecture of the adaptive control system for an evolutionary algorithm.

Genotypic diversity (GD) of a population is the first input, and is a search-space-level measure of the similarity of genes in members of a population. Since the object-level problem space is inherently discrete, the evolutionary algorithm utilizes an integer representation, which maps well to the problem formulation. Given two integer coded genomes of the same length, we compute their *Hamming distance* d . A *Hamming distance* is usually applied to bit strings and computes the number of bit locations that differ in strings of the same length.

¹ Each realizable design requires assignments of several parts, and each assigned part requires an assignment of a supplier who is capable of supplying that part.

So, given two bit strings 00111 and 11011, $d(00111, 11011) = 3$. We extend this idea to integer strings—if the integer alleles for corresponding genes in two genomes are unequal, their distance count is incremented by one. The genotypic diversity of two integer-coded genomes is normalized using the genome length. The genotypic diversity for a population of size n is therefore given by:

$$GD = \frac{2}{n(n-1)} \sum_{i=1, j=i+1}^n \frac{d_{ij}}{\text{Genome Length}} \quad (1)$$

We use a normalization term of $2/n(n-1)$ since given n members only $n(n-1)/2$ comparisons are distinct and non-reflexive. The range for GD is $[0, 1]$, and when GD is close to 0, the diversity is low, indicating convergence of the population, and when it approaches 1, the diversity is very high.

Percentage completed trials (PCT) is the second input, and is a number in the range $[0, 1]$, where 1 signifies exhaustion of all allowed trials, and consequently a maturity of the search given a fixed amount of resources.

In our prior work [12], we used genotypic diversity and phenotypic (fitness-level) diversity as the two inputs to a fuzzy controller. In that work, we did not explicitly introduce time or search maturity as an input, and implicitly used the phenotypic diversity measure to serve this purpose. Since the objectives of the evolutionary search are different in the early and later stages, it is desirable to include time or search maturity as an explicit input. In addition, this leads to a simpler and more intuitive design of the fuzzy knowledge base.

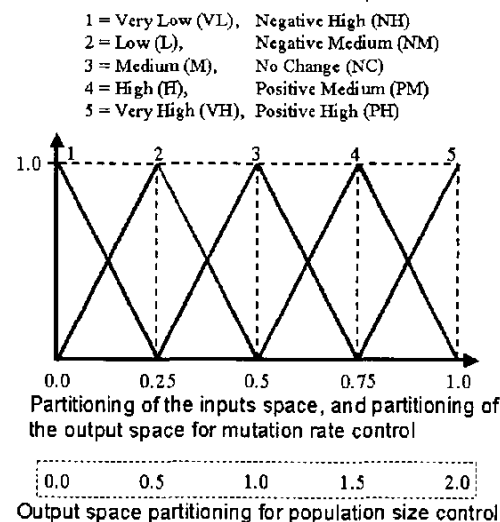


Figure 2: Linguistic terms and membership distributions associated with the fuzzy logic controller.

Fuzzy membership distributions that partition the inputs and outputs spaces are shown in Figure 2. Triangular, proportionally distributed membership functions are used for specification of the knowledge base of the fuzzy logic controller. The knowledge bases for population size control

and mutation rate control are shown in Table 1 and Table 2 respectively.

| | | | | | |
|--------|----|----|----|----|----|
| GD\PCT | VL | L | M | H | VH |
| VL | PH | PH | PH | PM | NC |
| L | PH | PH | PM | NC | NM |
| M | PH | PM | NC | NM | |
| H | PM | NC | NM | | |
| VH | NC | NM | | | |

Table 1: Knowledge base for Δ Population Size as a function of GD and PCT.

| | | | | | |
|--------|----|----|---|---|----|
| GD\PCT | VL | L | M | H | VH |
| VL | VH | VH | H | H | M |
| L | VH | H | H | M | M |
| M | H | H | M | M | M |
| H | H | M | M | M | M |
| VH | M | M | M | M | M |

Table 2: Knowledge base for Δ Mutation Rate as a function of GD and PCT.

Given a population size, the new population size is computed as a product of the current population size and the current Δ population factor. The search is initialized with a population size of 50, and further population sizes are bounded to the range [25, 150] to prevent a search with very small or very large populations. The fuzzy control for population size is fired at each generation. A new mutation rate scaled to the range [0.005, 0.1] is realized as a transient spike (Δ mutation rate) relative to the baseline mutation of 0.005 every 10 generations. The mutation rate is returned to the baseline level at the generation following the one where the control action occurs. Such a mutation is designed to serve as a temporary disruption to introduce additional population diversity, and we return the mutation to the baseline level between control actions in order to exploit the potential benefit due to the disruption.

The *min* operator is used for conjunction of clauses in the *IF* part of a rule, the *min* operator is used to fire each rule, the *max* operator is used to aggregate the outputs of all fired rules, and a *Center of Gravity* method is used for defuzzification.

IV. EXPERIMENTAL RESULTS

In this section, we compare performance of two basic types of evolutionary algorithms to their respective performance when fuzzy control is introduced. The first algorithm called the *Standard Evolutionary Algorithm* (SEA) has a population size of 50, crossover rate of 0.6, mutation rate of 0.005, uses proportional selection, applies uniform crossover to two selected parents to produce two offspring, and completely replaces with elitism the parent population with the offspring population. The second algorithm called the *Steady State Evolutionary Algorithm* (SSEA) is identical to the SEA, except that only 25% of the population is replaced at each

generation. The fuzzy controlled versions of these respective evolutionary algorithms are *Fuzzy Standard Evolutionary Algorithm* (F-SEA), and *Fuzzy Steady State Evolutionary Algorithm* (F-SSEA).

The experimental setup is based on an underlying discrete decision problem space of the order of 6.4×10^7 feasible options. Three different minimization problems are defined over this decision space by introducing various aggregation functions given by:

$$\min J_{ij} = Cost_{ij} \times Time_{ij} \quad (2a)$$

$$\min J_{ij} = Cost_{ij} + Time_{ij}^2 \quad (2b)$$

$$\min J_{ij} = Cost_{ij} \times \exp((Time_{ij} - 10)/3) \quad (2c)$$

These objectives represent various heuristic tradeoffs between total cost and total time objectives in the design, supplier, and manufacturing planning. Experiments are conducted using 3000, 5000, 7000, 9000, and 11000 allowed fitness trials per evolutionary search, and for each experimental setup an algorithm's performance is observed over 20 repeat trials. The resulting experiments space consists of the cross product of 4 algorithm types, 3 versions of objectives, 5 versions of maximum fitness trials, and 20 repeat trials, resulting in a total of 1200 experiments. Algorithm performance measures include a) the mean of the optimum found in 20 repeat trials, and b) the associated standard deviation. The *t-test* for the mean, and *F-test* for variance are used to evaluate statistically significant differences, and we utilize a standard value of $p < 0.05$ for significance determination.

| J | Max Trials | SEA μ | SEA σ | F-SEA μ | F-SEA σ |
|--------------------------|------------|-----------|--------------|-------------|----------------|
| | 3000 | 1789 | 71 | • 1729 | 81 |
| | 5000 | 1768 | 103 | • 1705 | 74 |
| (2a) | 7000 | 1710 | 81 | 1680 | • 41 |
| | 9000 | 1749 | 102 | • 1676 | • 46 |
| | 11000 | 1720 | 88 | • 1668 | • 40 |
| | 3000 | 352 | 21 | • 341 | • 12 |
| | 5000 | 339 | 8 | 338 | 8 |
| (2b) | 7000 | 340 | 8 | 339 | 8 |
| | 9000 | 343 | 15 | • 334 | • 5 |
| | 11000 | 337 | 15 | 332 | • 3 |
| | 3000 | 655 | 90 | 638 | 88 |
| | 5000 | 625 | 95 | 601 | • 48 |
| (2c) | 7000 | 607 | 98 | 566 | • 22 |
| | 9000 | 569 | 30 | 574 | 42 |
| | 11000 | 608 | 129 | 573 | • 36 |
| % Superior | | 7% | 13% | 93% | 73% |
| % Statistically Superior | | | | • 40% | • 60% |

Table 3: Performance comparison of the Standard Evolutionary Algorithm (SEA) and Fuzzy Standard Evolutionary Algorithm (F-SEA). A shaded cell denotes comparatively superior performance. A shaded cell with the symbol • denotes statistically significant superior performance.

Table 3 above shows a rounded-up performance comparison of the SEA and F-SEA. The μ performance of the F-SEA is superior to the μ performance of the SEA 93% of the time, and the σ performance of the F-SEA is superior to the σ performance of the SEA 73% of the time. The μ performance of the F-SEA is *statistically superior* to the μ performance of the SEA 40% of the time, and the σ performance of the F-SEA is *statistically superior* to the σ performance of the SEA 60% of the time. Statistically superior performance occurs when the threshold p for the corresponding *t-test* or *F-test* is smaller than 0.05, and each such occurrence is highlighted and marked with the symbol •.

| J | Max Trials | SSEA μ | SSEA σ | F-SSEA μ | F-SSEA σ |
|--------------------------|------------|------------|---------------|--------------|-----------------|
| | 3000 | 1685 | 63 | 1677 | 58 |
| | 5000 | 1682 | 63 | 1673 | 47 |
| (2a) | 7000 | 1739 | 108 | • 1665 | • 45 |
| | 9000 | 1695 | 82 | 1684 | 70 |
| | 11000 | 1709 | 98 | 1665 | • 45 |
| | | | | | |
| | 3000 | 344 | 23 | • 332 | • 5 |
| | 5000 | 336 | 12 | • 332 | • 4 |
| (2b) | 7000 | 341 | 23 | • 336 | 15 |
| | 9000 | 335 | 15 | 335 | • 6 |
| | 11000 | 337 | 15 | 337 | 15 |
| | | | | | |
| | 3000 | 592 | 53 | • 554 | • 15 |
| | 5000 | 597 | 91 | • 571 | • 25 |
| (2c) | 7000 | 564 | 23 | 566 | 24 |
| | 9000 | • 556 | 18 | 573 | 25 |
| | 11000 | 568 | 24 | 564 | 23 |
| % Superior | | 13% | 13% | 73% | 80% |
| % Statistically Superior | | • 7% | | • 20% | • 47% |

Table 4: Performance comparison of the Steady State Evolutionary Algorithm (SSEA) and Fuzzy Steady State Evolutionary Algorithm (F-SSEA). A shaded cell denotes comparatively superior performance. A shaded cell with the symbol • denotes statistically significant superior performance.

Table 4 above shows a rounded-up performance comparison of the SSEA and F-SSEA. The μ performance of the F-SSEA is superior to the μ performance of the SSEA 73% of the time, and the σ performance of the F-SSEA is superior to the σ performance of the SSEA 80% of the time. The μ performance of the F-SSEA is *statistically superior* to the μ performance of the SSEA 20% of the time, while the μ performance of the F-SSEA is *statistically inferior* to the μ performance of the SSEA 7% of the time, and the σ performance of the F-SSEA is *statistically superior* to the σ performance of the SSEA 47% of the time.

The above results support the statistically based inference that introducing fuzzy control to manage resources of the SEA has more of an impact than when fuzzy control is introduced to

manage resources of an SSEA. Regardless, fuzzy control is able to improve performance of the SSEA as well.

In Table 5, we compare performance of the SEA to the SSEA in an effort to infer relative superiority of the basic approaches. These results support the statistical inference that the performance of the SSEA is significantly better than the performance of the SEA. A potential explanation for this phenomenon is since the SSEA only replaces a portion of its population at each generation it delays the onset of premature convergence, which is detrimental to the evolutionary search process.

| J | Max Trials | SEA μ | SEA σ | SSEA μ | SSEA σ |
|--------------------------|------------|-----------|--------------|------------|---------------|
| | 3000 | 1789 | 71 | • 1685 | 63 |
| | 5000 | 1768 | 103 | • 1682 | • 63 |
| (2a) | 7000 | 1710 | 81 | 1739 | 108 |
| | 9000 | 1749 | 102 | 1695 | 82 |
| | 11000 | 1720 | 88 | 1709 | 98 |
| | | | | | |
| | 3000 | 352 | 21 | 344 | 23 |
| | 5000 | 339 | 8 | 336 | 12 |
| (2b) | 7000 | 340 | • 8 | 341 | 23 |
| | 9000 | 343 | 15 | 335 | 15 |
| | 11000 | 337 | 15 | 337 | 15 |
| | | | | | |
| | 3000 | 655 | 90 | • 592 | • 53 |
| | 5000 | 625 | 95 | 597 | 91 |
| (2c) | 7000 | 607 | 98 | 564 | • 23 |
| | 9000 | 569 | 30 | 556 | • 18 |
| | 11000 | 608 | 129 | 568 | • 24 |
| % Superior | | 13% | 33% | 80% | 53% |
| % Statistically Superior | | | • 7% | • 20% | • 33% |

Table 5: Performance comparison of the Standard Evolutionary Algorithm (SEA) and Steady State Evolutionary Algorithm (SSEA). A shaded cell denotes comparatively superior performance. A shaded cell with the symbol • denotes statistically significant superior performance.

V. CONCLUSIONS

We have presented statistical results based on a large suite of experiments, which support the argument that adaptive control of an evolutionary algorithm's resources via fuzzy logic may be realized in a simple, intuitive, and efficient manner. We have observed that the overhead due to fuzzy control intervention is minimal, in spite of activating the fuzzy controller for population size at each generation. This overhead could increase for some applications that require larger and more complex fuzzy rule bases. In such cases, rules compilation [1] could be used as a novel technique to achieve up to an order of magnitude speedup.

We have demonstrated that expert heuristic knowledge on managing an evolutionary algorithm can be suitably encoded using a fuzzy logic framework to remarkably improve an

evolutionary algorithm's search performance. In this function, the fuzzy logic control scheme serves as an intelligent manager that routinely observes the progress of the search and makes modifications to increase or decrease population diversity as necessary to both improve the search and to control the transition from exploration in the initial stages to exploitation in the later stages.

The principal finding of our current work based on hypothesis testing is that an adaptive evolutionary algorithm generally results in a much tighter variance in search results, and is a significant improvement over evolutionary algorithms based on a static canonical parameter set. Moreover, this approach leads to a higher degree of search confidence as evidenced by a tighter spread in the search. What is also remarkable is that except for one outlier in the space of experiments, the fuzzy controlled evolutionary algorithm approaches are inherently superior to their static parameter versions.

In the approach to adaptive fuzzy control presented, we have emphasized the explicit use of time as a state variable via the introduction of the percentage of completed trials measure. This is motivated by the fact that the nature of the evolution is different in the early, intermediate, and mature stages of search, and striking a balance between diversity and resource constrained convergence is important in solving practical problems. Optimal diversity maintenance without considering time or computational resources would imply postponement of convergence to infinity, which is not a practical option. Considering time or search maturity as an explicit input also leads to a simpler and more intuitive design of the fuzzy knowledge base.

Fitness-based diversity measures do not differentiate dissimilar solutions that may have the same or similar fitness, such as when the solutions correspond to points along a fitness contour or when they lie near different modes with comparable fitness. However, genotypic diversity, such as the measure used in this work is general and would differentiate the solutions in the above examples. It is our opinion that selecting an appropriate diversity measure is an important task and a practitioner must clearly understand the goals of the optimization application in order to choose the most applicable diversity measure. For instance, in multi-modal function optimization, where identification of several modes becomes necessary, one may have to consider clustering-based diversity measures instead. Non-stationary function optimization is another challenging example that requires the selection of a novel diversity measure.

The adaptive fuzzy control approach presented in this paper was discussed in the context of single objective evolutionary optimization, using problems from the domain of discrete design, supplier, and manufacturing planning in an agile environment. In several problem domain instances including this, there is a distinct advantage to treating the multiple objectives independently without using aggregation functions.

The resulting evolutionary search for a *Pareto optimal* solution set [2] may be enhanced as well using the fuzzy control approach outlined in this paper. Preliminary results in this area are encouraging [16], and this topic of research is currently being pursued.

REFERENCES

- [1] P. Bonissone. A Compiler for Fuzzy Logic Controllers. In *Proceedings of the International Fuzzy Engineering Symposium*, 1991.
- [2] S. Bonissone and R. Subbu. Exploring the Pareto Frontier using Multi-Sexual Evolutionary Algorithms: An Application to a Flexible Manufacturing Problem. In *Proceedings of the SPIE Annual Meeting—Program on Algorithms and Architectures*, 2002.
- [3] O. Cordon, F. Herrera, and M. Lozano. A Classified Review on the Combination Fuzzy Logic-Genetic Algorithms Bibliography. *Technical Report DECSAI 95129*, Department of Computer Science and AI, Universidad de Granada, Spain, December 1996.
- [4] K. A. De Jong. An Analysis of the behavior of a class of genetic adaptive systems. PhD thesis, University of Michigan, Ann Arbor, Michigan, 1975.
- [5] A. E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter Control in Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2), 1999.
- [6] J. J. Grefenstette. Optimization of Control Parameters for Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1), 1986.
- [7] F. Herrera and M. Lozano. Adaptive Genetic Operators Based on Coevolution with Fuzzy Behaviors. *IEEE Transactions on Evolutionary Computation*, 5(2), 2001.
- [8] F. Herrera and M. Lozano. Adaptation of genetic algorithm parameters based on fuzzy logic controllers. In F. Herrera and J. L. Verdegay, editors, *Genetic Algorithms and Soft Computing*, pages 95-129. Physica-Verlag, 1996.
- [9] F. Herrera, M. Lozano, and J. L. Verdegay. Tuning Fuzzy Logic Controllers by Genetic Algorithms. *International Journal of Approximate Reasoning*, 12(3/4), 1995.
- [10] M. A. Lee and H. Takagi. Dynamic Control of Genetic Algorithms using Fuzzy Techniques. In *Proceedings of the 5th International Conference on Genetic Algorithms*, 1993.
- [11] R. Subbu, C. Hocaoglu, and A. C. Sanderson. A Virtual Design Environment Using Evolutionary Agents. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1998.
- [12] R. Subbu, A. C. Sanderson, and P. P. Bonissone. Fuzzy Logic Controlled Genetic Algorithms versus Tuned Genetic Algorithms: An Agile Manufacturing Application. In *Proceedings of the ISIC/CIRA/ISAS Conference*, 1998.
- [13] A. Tettamanzi and M. Tomassini. Fuzzy Evolutionary Algorithms. In *Soft Computing: Integrating Evolutionary, Neural, and Fuzzy Systems*. Springer, 2001.
- [14] A. Tuscon and P. Ross. Adapting Operator Settings in Genetic Algorithms. *Evolutionary Computation*, 6(2), 1998.
- [15] D. H. Wolpert and W. G. MacReady. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1), 1997.
- [16] F. Xue. Fuzzy Logic Controlled Multi-Objective Differential Evolution. *Electronics Agile Manufacturing Research Institute (EAMRI) Research Report*, ER03-4. Rensselaer Polytechnic Institute, Troy, NY, 2003.