Evolutionary Algorithms Based Multiobjective Optimization Techniques for Intelligent Systems Design

Michael A. LEE   Henrik ESBENSEN
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720 USA
leem@cs.berkeley.edu esbensen@eecs.berkeley.edu
fax: 510 642 5775

Abstract

In this paper, we present evolutionary algorithms based multiobjective optimization techniques for intelligent systems design. Multiobjective optimization techniques are necessary in situations where the performance of a system is based on multiple, possibly conflicting objective whose aggregation cannot be easily articulated. The evolutionary algorithms approach presented in this paper employs a search mechanism that treats each of the objectives independently, avoiding the objective aggregation step. A key feature of our techniques is that they output a set of solutions rather than a single solution. To demonstrate how our techniques can be used to support system design, we apply them to the task of designing a fuzzy control system. In the final part of the paper, we propose metrics for multiobjective optimization algorithm performance and techniques for employing them in the design an adaptation of evolutionary algorithms based multiobjective optimization.

1 Introduction

In many real world contexts, multiple metrics must be considered simultaneously when evaluating the performance of a system. However, aggregating these metrics in a manner consistent with our intentions can be complicated by our ignorance or our inability to accurately articulate our intentions. For example, when designing a system, an engineer always considers the system complexity and performance trade-off. In this example, the task of the engineer is not only to develop a single solution, but to understand the trade-offs involved with such a solution: what is the best expected performance for a given amount of complexity. The solution in such problems is not a single solution, but rather a set of solutions that represent the set of best alternatives, or *Pareto Optimal* set. The notion of Pareto optimality is based on the concept of dominance. Let $f(x) = (f_1(x), \ldots, f_n(x))$ represent a vector valued objective function and u and v represent two solutions. u dominates v, written $u <_d v$, if and only if $\forall i: f_i(u) \le f_i(v)) \wedge (\exists i: f_i(u) < f_i(v)$. Solutions included in the *Pareto Optimal Set* are those that cannot be improved along any dimension without simultaneously being deteriorated along other dimension(s). In this paper, we present an evolutionary algorithm approach to multiobjective optimization.

2 Multiobjective Evolutionary Algorithms

Evolutionary algorithms are population based stochastic search strategies modeled after natural genetics mechanisms[5]. An individual of a population encodes a solution as a string of parameters, which is then subjected to genetic operations such as mutation and crossover operations. An individual's likelihood to pass information onto the next generation is determined by its ability to thrive in the target environment: evolutionary algorithms implement a *survival of the fittest* policy.

Among the main steps for using evolutionary algorithms are:

- design a solution representation
- design a genetic encoding of the solution
- design an appropriate evaluation function
- choose appropriate algorithm parameter settings

Evolutionary algorithms are well suited for multiobjective optimization problems because of their population based nature. The population at the end of a run represents the solution. The main idea behind using evolutionary algorithms for multiobjective optimization is to search for a good approximation of the true Pareto set. To use evolutionary algorithms in this context requires a modification of the selection strategy.

In practice, virtually all evolutionary algorithms use a single valued function to drive the selection; that is, the performance of an individual is aggregated into a single value that determines its selection probability. For example, the selection probability can be a direct function of a weighted linear combination of different performance metrics. In this case, the weightings determine the relative importance among the objectives. However, as mentioned in the previous section, it can be difficult to combine satisfactorily the objectives into a single value without prior knowledge about the trade-offs.

To address this problem, we compute selection probability based on the dominance concept outlined in Section 1. The basic idea is to first compute the ranking of each solution in the set according the following definition:

$$r(y, X) = |\{x \in X| \ (x <_d y)\}| \tag{1}$$

where y represents a solution and X represents a set of solutions. This ranking mechanisms collapses the multidimensional objective space into a single dimension. The ranking is then passed through a function, i.e. linear or exponential, to compute the selection probability. The parameters of the selection function can be used to balance the exploration and exploration behavior of the algorithm.

3 Experiments and Results

As a demonstration of our technique, we apply a multiobjective evolutionary algorithm to the design a fuzzy system for controlling the cart-pole system.

One of the many variations of this classic problem is to attach a pole, using a hinge, to a cart that slides on a fixed length track. Using this definition, the objective is sometimes stated as *balance the pole and center the cart as fast as possible* by applying a force in either direction parallel to the track. In the literature, the performance of such a system is often formulated in the following way:

$$cart_{error}(T) = \sum_{t=0}^{T} (x(t) - x_d)^2 \tag{2}$$

$$pole_{error}(T) = \sum_{t=0}^{T} (\theta(t) - \theta_d)^2 \tag{3}$$

$$performance(T) = cart_{error}(T) + \beta pole_{error}(T) \tag{4}$$

where x indicates the cart position, θ represents the pole angle measured relative to the line perpendicular to the top of the cart, and T is the amount of time allotted for a trial. In our experiments, the system fails when the pole angle falls below Note that the performance measure, (4), should be minimized and is a linear sum of two components that measure the cart and pole performance independently. Systems optimized according to this performance measure will behave differently depending on the value of β. The problem lies in choosing the proper weighting *a priori*. In reality, there will be different solutions that solve the same problem, but behave differently. Our goal is not to develop a single solution, but a set of solutions that approximates the set of best alternatives. Knowing the set of best alternatives will lead to more informed design decisions.

In our experiments, we demonstrate how the cart and pole performance metrics can be treated independently. The next two subsections describe a fuzzy system architecture and a genetic coding. The representation is based on the notion that the system is allowed a fixed number of rules with which to solve the task at hand. The last subsection presents experimental results.

3.1 Shared-Triangular Fuzzy System Architecture

The *Shared-Triangular* representation uses asymmetrical triangular membership functions and the *min* operator to synthesize multidimensional membership functions[6]. Each triangular membership function is specified by its center, left base width, and right base width.

Rules cover the input space by selecting and combining the one-dimensional membership functions from a globally defined set (all rules have access to the same set of membership functions). There is also a possibility for a rule to have no membership function associated with a particular input dimension, which implicitly forces the rule to cover entirely an input dimension.

Each rule has an additional parameter to modify, or hedge, the rule firing strength. If we define $\mu(\vec{x})$ as the raw rule firing strength, then the modified firing strength of a rule is given by:

$$\mu^p(\vec{x}), \tag{5}$$

where p represents the degree of hedging. The consequent output of each rule has a TSK form with the output of the system computed by taking a weighted sum of the outputs of all firing rules[8].

The total number of system parameters in an n-dimensional system with r rules and a maximum of m membership functions per input dimension is:

$$3mn + r(2n + 2). \tag{6}$$

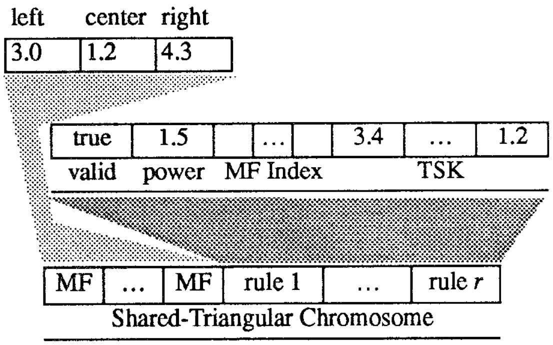

Figure 1 Shared-Triangular genetic representation showing rule validity parameter and rule power parameter. Note that there are $n+1$ TSK parameters.

Each rule requires $n+1$ parameters, because the membership function associated with each dimension must be specified (we implement this via an indexing scheme).

3.2 Genetic Representation

The genetic representation for the Shared-Triangular fuzzy systems is shown in Figure 1. The genetic code is made up of two different macro structures: membership function genes and rule genes. The membership genes specify distance between adjacent triangle centers and the left and right triangle base widths. The rule genes have locations for the membership function set indices for each input dimension as well as a validity bit. There is also a provision for completely covering an input dimension when the index is set to a special value.

3.3 Experimental Results

The design of fuzzy controllers for the cart-pole problem using our technique was simulation based. The performance metric we used was a slightly modified version of (4) (The measure was modified by starting the system from several different initial conditions and summing the performance over all trials.) In addition, to conserve computational resources and discourage failures, in the case of failures, we stopped the trials early and added a value proportional to the time left in the trial.

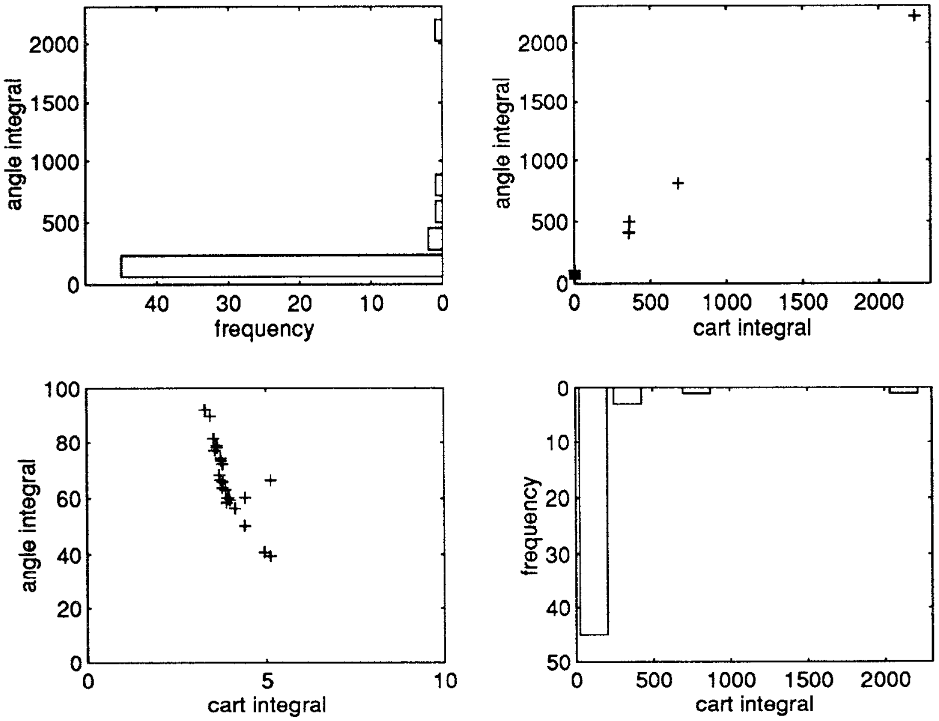

Figure 2 Output set of the evolutionary algorithm multiobjective optimization. The upper left and lower right plots show a histogram of the final solution set along each of the performance axis. The upper right and lower left show the solution set in the objective space.

In Figure 2, we show the output of our results. In the upper left and lower right, histograms of the final solution set as measured against each of the two objectives (the upper left is pole response and the lower right is cart response). The lower left and upper right plot the final solution set in the objective space. The plot in the lower left is a magnification of the objective space near the origin.

The output of this technique is a set of solutions. Solutions contained in the set approximated the set of best alternatives. From this set, we can identify the trade-offs and choose the solution that best satisfies our requirements. To illustrate this point, (3) compares the dynamics of two solutions chosen from the final set: one that has superior cart response relative to the other and one that has superior pole response relative the first.

4 Multiobjective Search Performance Metrics

One of the key features of our proposed approach is that the output of the optimization algorithm is a set of points as opposed to a single point. Using this concept as a guideline, we should consider measuring the performance of our algorithm based on the quality of the output set. Unlike single objective functions, where final solution quality can be measured using a single point, we would like to consider metrics to compare the approximation quality of two sets. For example, when is a cluster of points better than a sparsely distributed set of points. These metrics can be instrumental in designing the search algorithm, and determining and adapting its parameter settings.

As a first attempt to measure set quality, we have developed a metric that models the final selection performed by the user using a parametrized function that specifies the relative importance among individual objectives. We then measure the quality of a set by sampling the space of possible selection functions as given by this functional structure. For illustration purposes, consider a linear weighted combination represents a simple structure, where the linear weightings give the relative importance of each objective. We now define the quality $q(X)$ of a set X as the expected value of the selection function when selecting from X while traversing the class of selection functions. The measure q is introduced and described in detail in [3].

Figure 4 illustrates the set quality measure on eight constructed sets s1 through s8, assuming two optimization dimensions. The corresponding estimated set quality values are given in Table 1. Except for sets s5 and s6, set quality decreases strictly with the set number, as seems intuitively reasonable. However, s5 is not better than s6

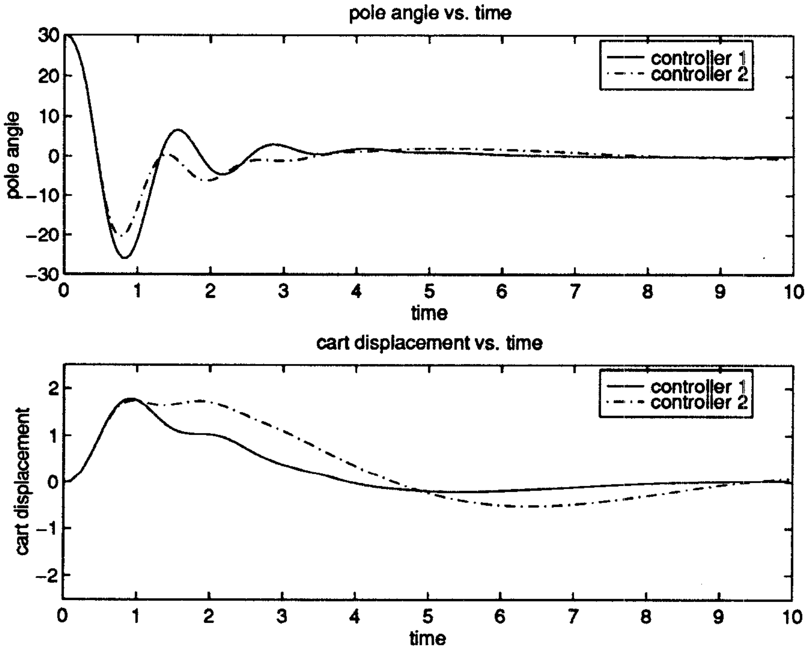

Figure 3 Comparison of the dynamics of two controllers chosen from the final solution set. One exhibits superior cart response while the other exhibits superior pole response.

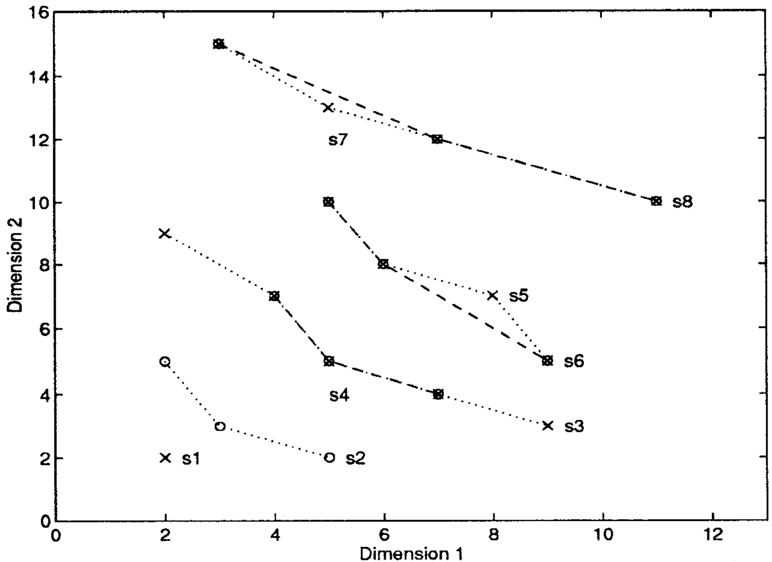

Figure 4 Each set is indicated by either circles connected by a dotted line or crosses connected by a dashed line. s4 is a subset of s3, s6 a subset of s5 and s8 a subset of s7.

because the solution with cost (8,7) is non-convex relative to s5.

Table 1. Estimated set qualities for sets s1 through s8, after sampling 50,000 selection function parameter sets.

set	s1	s2	s3	s4	s5	s6	s7	s8
$\hat{q}$	0.0001	0.0762	0.2127	0.2642	0.4130	0.4130	0.5232	0.5253

5 Summary and Conclusions

In this paper we presented an evolutionary algorithm based multiobjective optimization technique for intelligent systems design. One of the key characteristics of this approach is that the output is a set of solutions rather than a single solution. This is particularly useful in many real world contexts where multiple, possibly competing objectives are involved and the relationship between the objectives is not easily articulated or known. Although we applied our technique on a fuzzy systems design task, the technique will easily generalize to other tasks involving multiple objectives, such as component layout and financial engineering. We have also presented techniques for measuring the performance of multiobjective optimization algorithms. With further refinement, these metrics will provide us with the tools necessary for designing algorithm behavior and performance.

Acknowledgments

This research is supported in part by NASA Grant NCC-2-275, ONR Grant N00014-96-1-0556, and BISC program of UC Berkeley. The authors would also like to thank Prof. David Wessel and the Center for New Music and Audio Technologies at UC Berkeley for use of computing resources. Dr. Esbensen is supported mainly by the Danish Technical Research Council.

References and Related Publications

[1] DeJong, K.A. (1975) An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. Dissertation, University of Michigan, University Microfilms No. 68-7556.

[2] Esbensen, H., Kuh, E.S., "An MCM/IC Timing-Driven Placement Algorithm Featuring Explicit Design Space Exploration," *Proc. of the 1996 IEEE Multi-Chip Module Conference*, pp. 170-175, 1996.

[3] Esbensen H., Kuh, E.S., "Design Space Exploration Using the Genetic Algorithm," *Proc. of the IEEE International Symposium on Circuits and Systems*, 1996 (to appear).

[4] Fonseca, C.M. and Fleming, P.J. (1993) "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion, and Generalization," in *Proc. of the Fifth International Conference on Genetic Algorithms*, ed. S. Forest, San Mateo, CA: Morgan Kaufmann.

[5] Holland, J. H. (1975) *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA.

[6] Lee, M. A. (1995) "On Genetic Representation of High Dimensional Fuzzy Systems," *Proc. of NAFIPS'95*, College Park, MD, IEEE Computer Science Press, pp. 752-757.

[7] Lee, M.A. and Takagi, H. (1993) "Dynamic Control of Genetic Algorithms using Fuzzy Logic Techniques," in *Proc. of the Fifth International Conference on Genetic Algorithms*, ed. S. Forest, San Mateo, CA: Morgan Kaufmann.

[8] Takagi, T. and Sugeno, M. (1985) "Fuzzy Identification of Systems and Its Applications to Modelling and Control," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol.5, No.3, pp. 116-132.