Category: special track on Optimal Design of Engineered Structures

THE MAXIMIN FITNESS FUNCTION FOR MULTI-OBJECTIVE EVOLUTIONARY COMPUTATION: APPLICATION TO CITY PLANNING

R.J.Balling and S.A.Wilson

Department of Civil and Environmental Engineering
Brigham Young University
Provo, UT 84602

balling@byu.edu

801-378-2648

Category: special track on Optimal Design of Engineered Structures

THE MAXIMIN FITNESS FUNCTION FOR MULTI-OBJECTIVE EVOLUTIONARY COMPUTATION: APPLICATION TO CITY PLANNING

ABSTRACT

A new fitness function, known as the maximin fitness function, is presented for multi-objective genetic algorithms. This fitness function directs genetic algorithms towards final generations that are both close to the universal Pareto front and diverse. The performance of a genetic algorithm with the maximin fitness function as well as with the traditional Pareto-ranking fitness function is compared on a real-world test problem from city planning.

INTRODUCTION

Genetic algorithms, unlike other optimization methods, operate on generations of designs. This makes it possible to converge to a final generation containing many Pareto-optimal solutions in a single algorithm run. Fonseca and Fleming (1995) survey the early research work in multi-objective evolutionary optimization. They divided approaches into "non-Pareto approaches" and "Pareto-based approaches". The later approaches made direct use of the definition of Pareto optimality, while the former approaches did not.

The first Pareto-based approach was proposed by Goldberg (1989). This approach introduced the "Pareto-ranking fitness function". To evaluate this function for the designs in a particular generation, the Pareto subset is first identified. The Pareto subset consists of the non-dominated designs in the generation. A design is dominated if there exists another design in the generation that is better or equal in every objective, and better in at least one objective. Figure 1 plots the designs in a particular generation in objective space. The two objectives are being minimized. According to the definition of domination, Design A dominates Design B, and Design C dominates Design A. There is no design that dominates Design C. Therefore, according to the definition of Pareto optimality, Design C is in the Pareto subset of the generation. As shown in Figure 1, all designs in the Pareto subset are assigned a rank of one. This subset is temporarily deleted from the generation, and the Pareto subset of the remaining designs is identified and assigned a rank of two. The approach continues temporarily deleting Pareto subsets, identifying the Pareto subsets of the remaining designs, and assigning increasing ranks until all designs in the generation have been assigned ranks as shown in Figure 1. The value of the Pareto-ranking fitness for each design in the generation is taken as the reciprocal of its rank.

Fonseca and Fleming (1993) proposed a slightly different Pareto-ranking fitness function in which the rank of a design is one plus the number of designs in the generation by which it is dominated. Since designs in the Pareto subset are non-dominated, they have a rank of one. Again, the fitness is the reciprocal of the rank.

The use of the Pareto-ranking fitness function in a genetic algorithm causes the algorithm to advance the Pareto front from generation to generation towards the universal Pareto front. The universal Pareto front contains the designs in the Pareto subset of the universe of all possible designs. However, the Pareto-ranking fitness function does not guarantee that the

advance of the Pareto front from generation to generation will remain uniform. After all, designs on the universal Pareto front would have exactly the same fitness value. When presented with multiple equivalent optima, finite populations tend to cluster at a few or even a single solution of the problem. This phenomenon, known as genetic drift (Goldberg and Segrest, 1987), has been observed in natural as well as in artificial evolution. For this reason, researchers have developed niche induction techniques for use with the Pareto-ranking fitness function to produce final generations which are both close to the universal Pareto front and diverse (Fonseca and Fleming, 1993; Cieniawski, 1993; Srinivas and Deb, 1994).

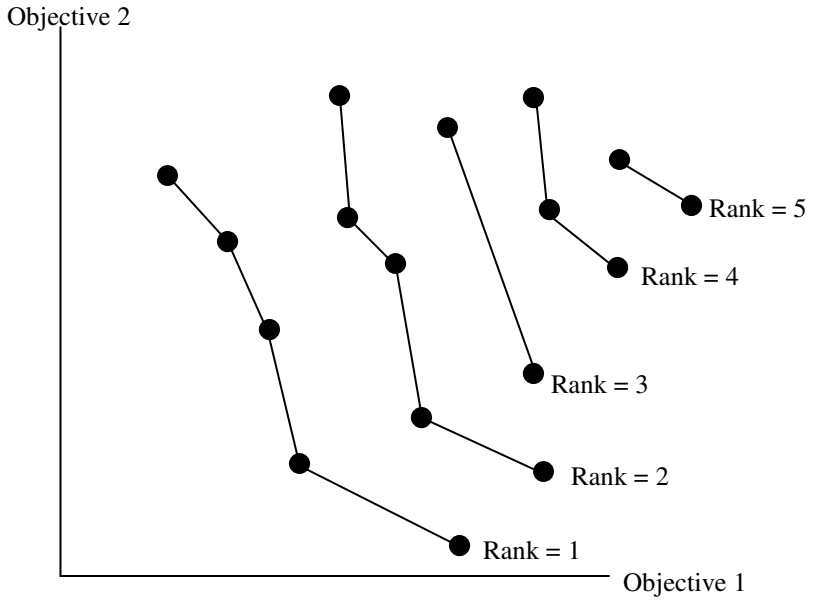


Figure 1: Pareto-Ranking Fitness for a Particular Generation

In this paper, we examine a new fitness function first introduced by Balling (2000) that is based directly on the definition of Pareto optimality. We call this function the "maximin fitness function". This fitness function directs genetic algorithms toward final generations that are both close to the universal Pareto front and diverse. The maximin fitness function will be compared with the Pareto-ranking fitness function on a real-world, multi-objective, city planning problem.

THE MAXIMIN FITNESS FUNCTION

Consider the case of two minimized objectives, f_1 and f_2. Consider design i and design j in a particular generation, and assume that these two designs are distinct in objective space:

$$\text{either } f_{1i} \neq f_{1j} \text{ or } f_{2i} \neq f_{2j} \text{ or both} \tag{1}$$

where f_{1i} and f_{1j} are the values of the first objective for the ith and jth designs, respectively, and f_{2i} and f_{2j} are the values of the second objective for the ith and jth designs, respectively. Since the two designs are distinct, the ith design will be dominated by the jth design if:

$$f_{1i} \geq f_{1j} \text{ and } f_{2i} \geq f_{2j} \tag{2}$$

This is equivalent to:

$$\min (f_{1i} - f_{1j}, f_{2i} - f_{2j}) \geq 0 \tag{3}$$

Thus, the ith design is dominated if:

$$\max_{j \neq i} \left(\min \left(f_{1i} - f_{1j}, f_{2i} - f_{2j} \right) \right) \geq 0 \tag{4}$$

From Equation (4), we construct the maximin fitness function for any number of minimized objectives:

$$\text{fitness}_i = 1 - \max_{j \neq i} \left(\min_k \left(f_{ki} - f_{kj} \right) \right) \tag{5}$$

fitness_i = fitness of ith design in the generation

f_{ki} = kth objective evaluated at ith design

f_{kj} = kth objective evaluated at jth design

In Equation (5), the min is taken over all the objectives, and the max is taken over all designs in the generation whose value for at least one objective is different than the corresponding objective value for the ith design. The maximin fitness will be greater than one for designs in the Pareto subset of the generation, and will be less than or equal to one for dominated designs. If the values of the objectives for designs in the generation are scaled between zero and one prior to evaluating the maximin fitness function, then the maximin fitness values will range from zero to two.

The simplicity of Equation (5) makes implementation of the maximin fitness function both easy and efficient. One constructs three nested loops. The outermost loop is over designs i in the generation, the middle loop is over designs $j \neq i$ in the generation, and the innermost loop is over objectives k. If tournament selection is used, selection pressure can be increased by increasing the tournament size. If roulette-wheel selection is used, selection pressure can be increased by raising the maximin fitness to an exponent greater than one before allocating space on the roulette wheel.

BEHAVIOR OF THE MAXIMIN FITNESS FUNCTION

The behavior of the maximin fitness function can be understood by considering a few simple examples. In Figures 2-6, we give examples of generations plotted in objective space involving two minimized objectives. Each design is labeled with a letter, and the value of the maximin fitness function is listed beside the letter in parentheses.

In Figure 2 we have a generation of four designs. Designs A, B, and C are in the Pareto subset and their maximin fitnesses are greater than one. Design D is dominated, and its maximin fitness is less than one.

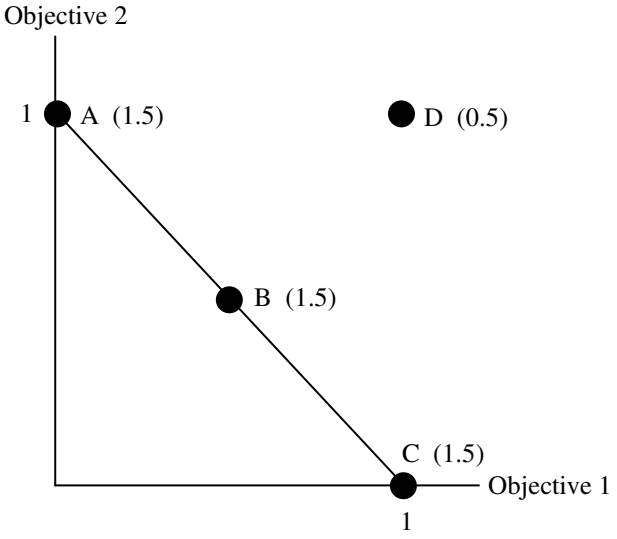


Figure 2: Maximin Fitness for a Particular Generation

The generation in Figure 3 differs from the generation in Figure 2 in that the objective coordinates of Design B have been moved from (0.5, 0.5) to (0.2, 0.2). Again, Designs A, B, and C are in the Pareto subset, and their maximin fitnesses are greater than one. While the Pareto front in Figure 2 is linear, the Pareto front in Figure 3 is convex. The maximin fitness of Design B is greater than the maximin fitnesses of Designs A and C in Figure 3. Thus, the maximin fitness function does not treat all designs in the Pareto subset equally. It appears to favor designs in the center of convex Pareto fronts. Note also that the maximin fitness of Design D drops from 0.5 in Figure 2 to 0.2 in Figure 3. This is because the amount of domination of Design D by Design B increases from Figure 2 to Figure 3. If the Pareto-ranking fitness function were used, the fitness of Designs A, B, C, and D would not change from Figure 2 to Figure 3.

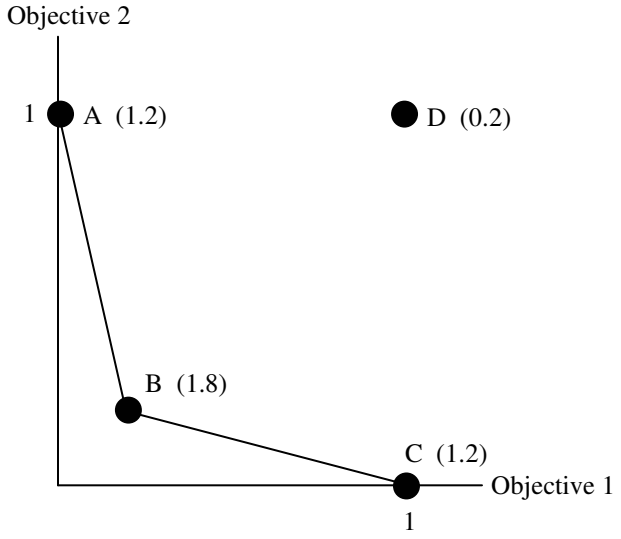


Figure 3: Maximin Fitness for a Particular Generation

Because Design B in Figure 3 has such a high fitness, it is likely that in the next generation, new designs will be created with objective values near those of Design B. An example of this is shown in Figure 4 where the generation is identical to the generation in Figure 3 except that Design E has been added near Design B. Note that the presence of Design E near Design B caused the maximin fitness of Design B to drop from 1.8 in Figure 3 to 1.1 in Figure 4. Thus, when clustering begins to occur, the maximin fitness automatically degrades.

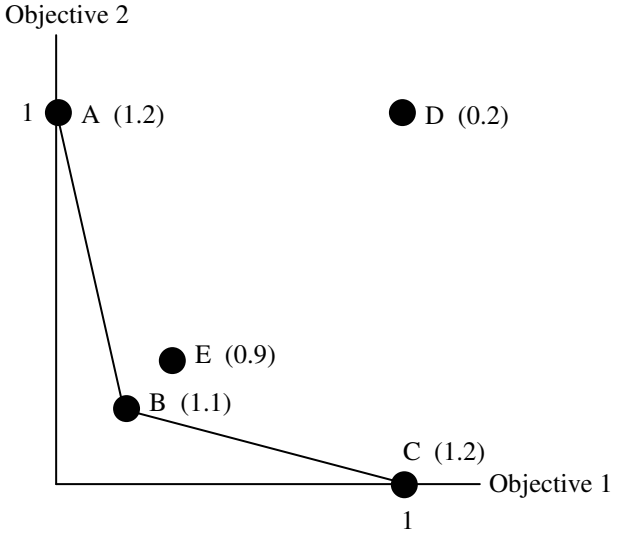


Figure 4: Maximin Fitness for a Particular Generation

Now consider Figure 5. Designs A, B, and C are in the Pareto subset, but this time the Pareto front is concave. The maximin fitness favors Designs A and C over Design B. Thus, the maximin fitness function appears to favor the extremes of concave Pareto fronts.

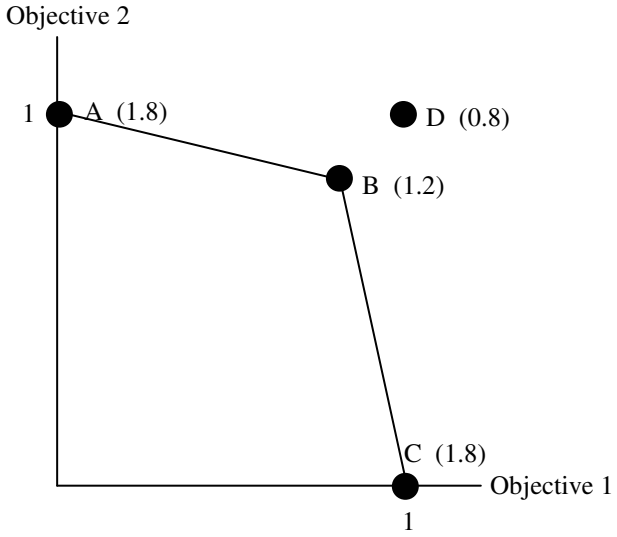


Figure 5: Maximin Fitness for a Particular Generation

Because the fitness of Design A is so high in Figure 5, suppose the next generation adds Design E near Design A as shown in Figure 6. This clustering around Design A causes its fitness to drop from 1.8 in Figure 5 to 1.1 in Figure 6, and suddenly Design C looks much better. The Pareto subset consists of Designs A, B, and C whose maximin fitnesses are greater than one, while Designs D and E are dominated since their maximin fitnesses are less than or equal to one.

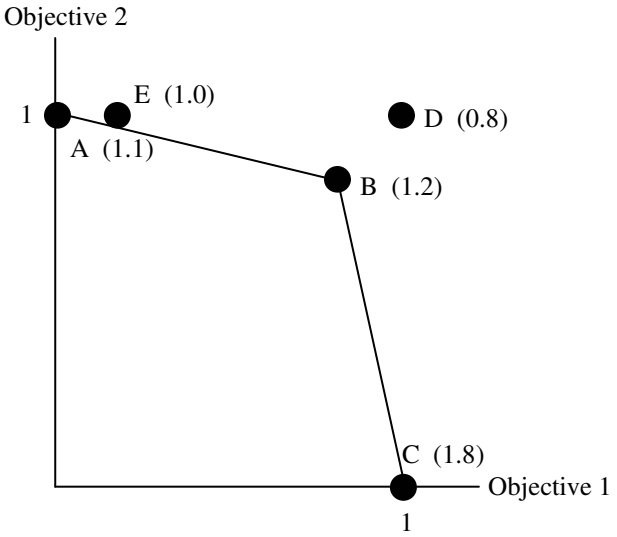


Figure 6: Maximin Fitness for a Particular Generation

These simple examples illustrate the potential advantages of the maximin fitness function over the Pareto-ranking fitness function. First, when clustering on the Pareto front begins to occur, the maximin fitness begins to degrade causing the genetic algorithm to search elsewhere on the Pareto front thereby maintaining diversity. Second, the maximin fitness function quantifies the amount of domination. The Pareto-ranking fitness function does not capture any information about how much worse designs with rank 2 are than designs with rank 1. In a sense, the maximin fitness function quantifies the distance between ranks. Finally, if a generation contains uniformly spaced designs on the Pareto front, the maximin fitness

function favors designs in the center of convex fronts and designs at the extremes of concave fronts. An argument can be made that these are the designs of most interest to designers in these two very different situations.

APPLICATION TO A CITY PLANNING EXAMPLE

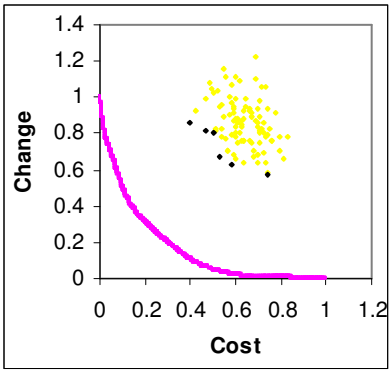
The different fitness functions were tested on a multi-objective real world problem. The problem was one of finding optimum future land-use and transportation plans for two adjacent high-growth cities in the state of Utah in the United States. The land in the cities was divided into 199 zones, and the major streets in the cities were divided into 60 corridors. The problem had one variable for each zone and one variable for each corridor. The possible values of the zone variables were integers from 1 to 11 corresponding to eleven different future residential and commercial land uses. The possible values of the corridor variables were integers from 1 to 10 corresponding to ten different future street types. City planners from both cities were involved to reduce the search space by identifying which land uses would be allowed for each zone and which street types would be allowed for each corridor. The number of possible future plans in the search space was reduced to 10^{108}.

A single constraint was imposed requiring that the future housing capacity of any plan be greater than the projected future population of 327,000 for both cities. Incidentally, the current zoning plan for these cities was infeasible. During execution of the genetic algorithm, infeasible plans would be generated about 1% of the time, and these were immediately deleted from all generations and replaced with feasible plans. Several objectives were identified for this problem, but the results that we will now present involved only two objectives. The first was the minimization of cost minus revenues. Costs included the construction and right-of-way costs of upgrading corridors, and revenues included the property and sales tax revenues from residential and commercial land. The second objective was the minimization of change from the status quo. This objective was quantified by summing the product of property value, area, and a degree-of-change factor over all zones and all land adjacent to corridors. Degree-of-change factors were empirically set to represent the severity of change in a zone or corridor from its current classification to its future planned classification. We will refer to these two minimized objectives as the "cost objective" and the "change objective".

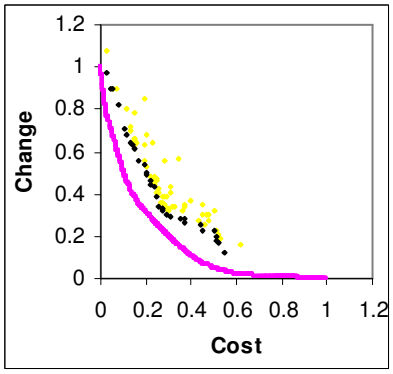
A genetic algorithm was developed to solve this problem. We ran the algorithm for 1000 generations with a generation size of 100 feasible plans. Tournament selection was used with a tournament size of 6. Single-point crossover was used with a crossover probability of 1.0. The mutation probability started at 0.035 in the first generation and was linearly decreased to zero in the last generation. Elitism was implemented by copying two plans from each generation to the next. These plans consisted of the plan with the lowest value of the cost objective, and the plan with the lowest value of the change objective.

The genetic algorithm was executed with the Pareto-ranking fitness function and the maximin fitness function. Generations 1, 200, 400, 600, 800, and 1000 are plotted in scaled objective space in Figures 7 and 8. In each figure a baseline curve approximating the universal Pareto front for this problem is shown for reference. Designs in the Pareto subsets of the generations in these figures are shown with dark dots while dominated designs are shown with light dots. Note that dominated designs may still exist in final generations. This is to be expected because crossover and mutation of designs on the Pareto front of one generation may produce dominated designs in the next generation. Thus, the solution from the genetic algorithm should be taken as the Pareto subset of the final generation.

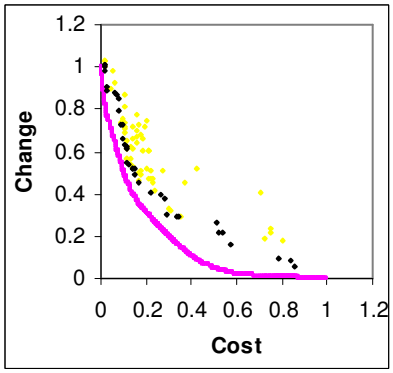
Note that in the final generations, the Pareto-ranking fitness function leads to clustering, while the maximin fitness function causes the final generations to spread out along the Pareto front. The algorithm was executed several times with different random number sequences, and the results were similar each time.



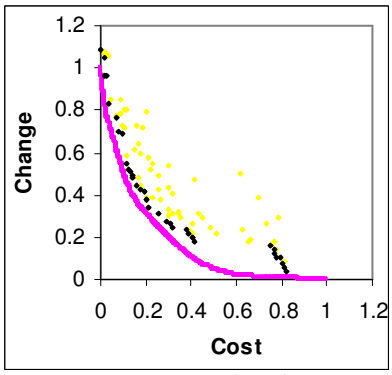
Generation 1



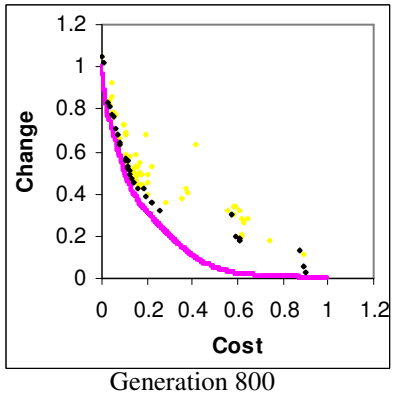
Generation 200



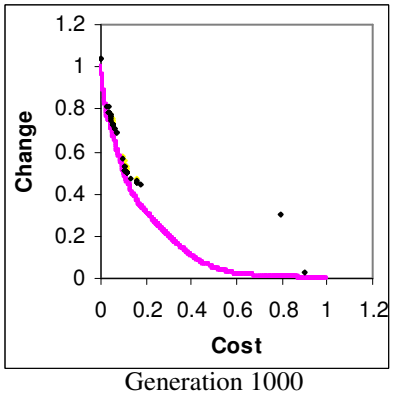
Generation 400



Generation 600

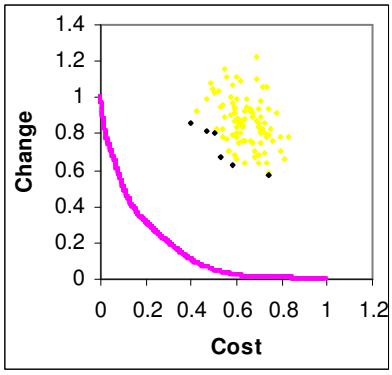


Generation 800

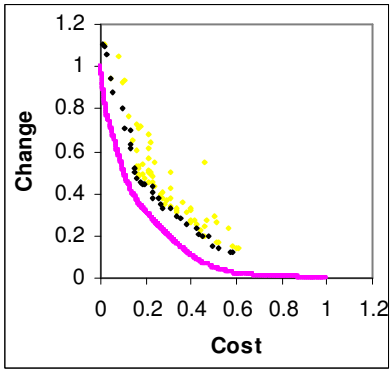


Generation 1000

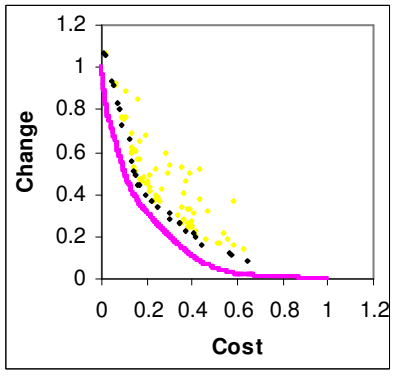
Figure 7: Pareto-Ranking Fitness Function



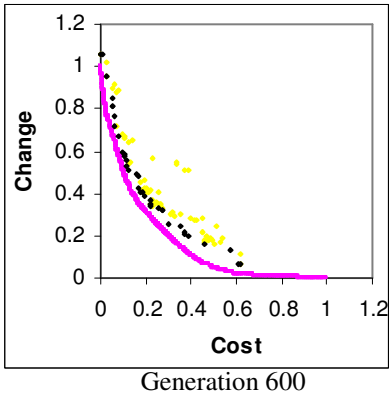
Generation 1



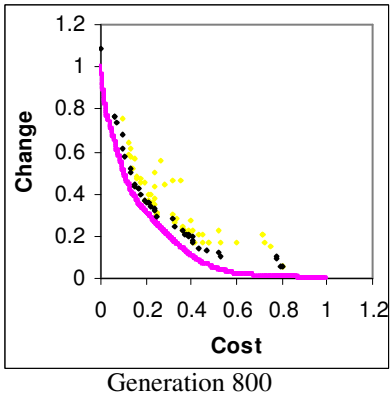
Generation 200



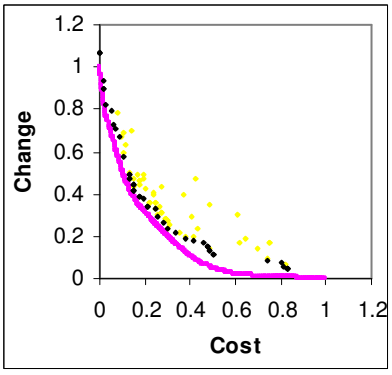
Generation 400



Generation 600



Generation 800



Generation 1000

Figure 8: Maximin Fitness Function

CONCLUSIONS

A new fitness function for multi-objective genetic algorithms has been presented and investigated. Genetic algorithm performance with this fitness functions was compared to the performance with the traditional Pareto-ranking fitness function on a real-world test problem. The Pareto-ranking fitness function led to clustering of designs in the final generations while the new fitness function did not.

The new fitness function is called the maximin fitness function. This function is designed to achieve both diversity and closeness to the universal Pareto front in multi-objective genetic algorithms without having to employ niche induction techniques. The maximin fitness function is derived directly from the definition of Pareto optimality and is naturally extended to problems with N objectives. Its simple form makes implementation easy.

ACKNOWLEDGEMENT

This work was funded by the National Science Foundation under Grant No. CMS-9817690, for which the authors are grateful.

REFERENCES

Balling, R. J. (2000). Pareto sets in decision-based design. *Journal of Engineering Valuation and Cost Analysis*, 3, 189-198.

Cienawski, S. E. (1993). *An investigation of the ability of genetic algorithms to generate the tradeoff curve of a multi-objective groundwater monitoring problem*. Master's thesis, University of Illinois at Urbana-Champaign.

Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In Forrest, S., (Ed.), *Proceedings of the fifth international conference on genetic algorithms* (pp. 416-423). San Mateo, California: Morgan Kauffman.

Fonseca, C. M. and Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3, 1-16.

Goldberg, D. E. (1989). *Genetic algorithms for search, optimization, and machine learning*. Reading, Massachusetts: Addison-Wesley.

Goldberg, D. E. and Segrest, P. (1987). Finite Markov chain analysis of genetic algorithms. In J. J. Grefenstette (Ed.), *Genetic algorithms and their applications: Proceedings of the second international conference on genetic algorithms.* (pp. 58-67). Hillsdale, New Jersey: Lawrence Erlbaum.

Srinivas, N., and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3), 221-248.