Constrained Multi-objective Optimization Using Steady State Genetic Algorithms

Deepti Chafekar, Jiang Xuan, and Khaled Rasheed

Computer Science Department
University of Georgia
Athens, GA 30602 USA
{chafekar, xuan, khaled}@cs.uga.edu

Abstract. In this paper we propose two novel approaches for solving constrained multi-objective optimization problems using steady state GAs. These methods are intended for solving real-world application problems that have many constraints and very small feasible regions. One method called Objective Exchange Genetic Algorithm for Design Optimization (OEGADO) runs several GAs concurrently with each GA optimizing one objective and exchanging information about its objective with the others. The other method called Objective Switching Genetic Algorithm for Design Optimization (OSGADO) runs each objective sequentially with a common population for all objectives. Empirical results in benchmark and engineering design domains are presented. A comparison between our methods and Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) shows that our methods performed better than NSGA-II for difficult problems and found Pareto-optimal solutions in fewer objective evaluations. The results suggest that our methods are better applicable for solving real-world application problems wherein the objective computation time is large.

1 Introduction

This paper concerns the application of steady state Genetic Algorithms (GAs) in realistic engineering design domains which usually involve simultaneous optimization of multiple and conflicting objectives with many constraints. In these problems instead of a single optimum there usually exists a set of trade-off solutions called the non-dominated solutions or Pareto-optimal solutions. For such solutions no improvement in any objective is possible without sacrificing at least one of the other objectives. No other solutions in the search space are superior to these Pareto-optimal solutions when all objectives are considered. The user is then responsible for choosing a particular solution from the Pareto-optimal set later.

Some of the challenges faced in the application of GAs to engineering design domains are:

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2723, pp. 813–824, 2003.

- The search space can be very complex with many constraints and the feasible (physically realizable) region in the search space can be very small.
- Determining the quality (fitness) of each point may involve the use of a simulator or an analysis code which takes a non-negligible amount of time. This simulation time can range from a fraction of a second to several days in some cases. Therefore it is impossible to be cavalier with the number of objective evaluations in an optimization.

For such problems steady state GAs may perform better than generational GAs because they better retain the feasible points found in their populations and may have higher selection pressure which is desirable when evaluations are very expensive. With good diversity maintenance, steady state GAs have done very well in several realistic domains [1]. Significant research has yet to be done in the area of steady state multi-objective GAs. We therefore decided to focus our research on this area.

The area of multi-objective optimization using Evolutionary Algorithms (EAs) has been explored for a long time. The first multi-objective GA implementation called the Vector Evaluated Genetic Algorithm (VEGA) was proposed by Schaffer in 1985 [9]. Since then, many Evolutionary algorithms for solving multi-objective optimization problems have been developed. The most recent ones are the Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) [3], Strength Pareto Evolutionary Algorithm-II (SPEA-II) [16], Pareto Envelope based selection-II (PESA-II) [17]. Most of these approaches propose the use of a generational GA. Deb proposed an Elitist Steady State Multi-objective Evolutionary Algorithm (MOEA) [18] which attempts to maintain spread [15] while attempting to converge to the true Pareto-optimal front. This algorithm requires sorting of the population for every new solution formed thereby increasing its time complexity. Very high time complexity makes the Elitist steady state MOEA impractical for some problems. To the best of our knowledge, apart from Elitist Steady State MOEA, the area of steady state multi-objective GAs has not been widely explored. Also constrained multi-objective optimization which is very important for real-world application problems has not received the deserved exposure. In this paper we propose two methods for solving constrained multi-objective optimization using steady state GAs. These methods are relatively fast and practical. It is also easy to transform a single-objective GA to a multi-objective GA by using these methods.

In the first method called the Objective Exchange Genetic Algorithm for Design Optimization (OEGADO) several single objective GAs run concurrently. Each GA optimizes one of the objectives. At certain intervals these GAs exchange information about their respective objectives with each other. In the second method called the Objective Switching Genetic Algorithm for Design Optimization (OSGADO) a single GA runs multiple objectives in a sequence switching at certain intervals between objectives.

Our methods can be viewed as multi-objective transformations of GADO (Genetic Algorithm for Design Optimization) [1, 2]. GADO is a GA that was designed with the goal of being suitable for the use in engineering design. It uses new operators and search control strategies that target engineering domains. GADO has been applied in a variety of optimization tasks which span many fields. It has demonstrated a great deal of robustness and efficiency relative to competing methods.

In GADO, each individual in the GA population represents a parametric description of an artifact. All parameters have continuous intervals. The fitness of each individual is based on the sum of a proper measure of merit computed by a simulator or some analysis code, and a penalty function if relevant. A steady state model is used, in which several crossover and mutation operators including specific and innovative operators like guided crossover are applied to two parents selected by linear rank based selection. The replacement strategy used is a crowding technique, which takes into consideration both the fitness and the proximity of the points in the GA population. GADO monitors the degree of diversity of the GA population. If at any stage it is discovered that the individuals in the population became very similar to one another, the diversity maintenance module rebuilds the population using previously evaluated points in a way that restores diversity. The diversity maintenance module in GADO also rejects proposed points that are extremely similar to previously evaluated points. The GA stops when either the maximum number of evaluations has been exhausted or the population loses diversity and practically converges to a single point in the search space. Floating point representation is used. GADO also uses some search control strategies [2] such as a screening module which saves time by avoiding the full evaluation of points that are unlikely to correspond to good designs.

We compared the results of our two methods with the state-of-the-art Elitist Non-Dominated Sorting Algorithm-II (NSGA-II) [3]. NSGA-II is a non-dominated sorting based multi-objective evolutionary algorithm with a computational complexity of $O(MN^2)$ (where M is the number of objectives and N is the population size). NSGA-II incorporates an elitist approach, a parameter-less niching approach and a simple constraint handling strategy. Due to NSGA-II's low computational requirements, elitist features and constraint handling capacity, it has been successfully used in many applications. It proved to be better than many other multi-objective optimization GAs [3, 18].

In the remainder of the paper, we provide a brief description of our two proposed methods. We then present results of the comparison of our methods with NSGA-II. Finally, we conclude the paper with a discussion of the results and future work.

2 Methods for Multi-objective Optimization Using Steady State GAs

We propose two methods for solving constrained multi-objective optimization problems using steady state GAs. One is the Objective Exchange Genetic Algorithm for Design Optimization (OEGADO), and other is the Objective Switching Genetic Algorithm for Design Optimization (OSGADO). It should be noted that for multi-objective GAs, maintaining diversity is a key issue. However we did not need to take any extra measures for diversity maintenance as the diversity maintenance module already present in GADO [1, 2] seemed to handle this issue effectively. We focused on the case of two objectives in our experiments for simplicity of implementation and readability of the results, but the methods are applicable for multi-objective optimization problems with more than two objectives.

2.1 Objective Exchange Genetic Algorithm for Design Optimization (OEGADO)

The main idea of OEGADO is to run several single objective GAs concurrently. Each of the GAs optimizes one of the objectives. All the GAs share the same representation and constraints, but have independent populations. They exchange information about their respective objectives every certain number of iterations.

In our implementation, we have used the idea of informed operators (IOs) [4]. The main idea of the IOs is to replace pure randomness in traditional GA operators with decisions that are guided by reduced models formed using the methods presented in [5, 6, 7]. The reduced models are approximations of the fitness function, formed using some approximation techniques, such as least squares approximation [5, 7, 8]. These functional approximations are then used to make the GA operators such as crossover and mutation more informed. These IOs generate multiple children [4], rank them using the approximate fitness obtained from the reduced model and select the best.

Every single objective GA in OEGADO uses least squares to form a reduced model of its own objective. Every GA exchanges its own reduced model with those of the other GAs. In effect, every GA, instead of using its own reduced model, uses other GAs' reduced models to compute the approximate fitness of potential individuals. Therefore each GA is informed about other GAs' objectives. As a result each GA not only focuses on its own objective, but also gets biased towards the objectives which the other GAs are optimizing.

The OEGADO algorithm for two objectives looks as follows:

1. Both the GAs are run concurrently for the same number of iterations, each GA optimizes one of the two objectives while also forming a reduced model of it.
2. At intervals equal to twice the population size, each GA exchanges its reduced model with the other GA.
3. The conventional GA operators such as initialization (only applied in the beginning), mutation and crossover are replaced by informed operators. The IOs generate multiple children and use the reduced model to compute the approximate fitness of these children. The best individual based on this approximate fitness is selected to be the newborn. It should be noted that the approximate fitness function used is of the other objective.
4. The true fitness function is then called to evaluate the actual fitness of the newborn corresponding to the current objective.
5. The individual is then added to the population using the replacement strategy.

6. Steps 2 through 5 are repeated till the maximum number of evaluations is exhausted.

If all objectives have similar computational complexity, the concurrent GAs can be synchronized, so that they exchange the current approximations at the right time. On the other hand, when objectives vary considerably in their time complexity, the GAs can be run asynchronously.

It should be noted that OEGADO is not really a multi-objective GA, but several single objective GAs working concurrently to get the Pareto-optimal region. Each GA finds its own feasible region, by evaluating its own objective. For the feasible points found by a single GA, we need to run the simulator to evaluate the remaining objectives. Thus for OEGADO with two objectives:
Total number of objective evaluations = Sum of objective evaluations of each GA + Sum of the number of feasible points found by each GA

A potential advantage of this method is speed, as the concurrent GAs can run in parallel. Therefore multiple objectives can be evaluated at the same time on different CPUs. Also the asynchronous OEGADO works better for objectives having different time complexities. If some objectives are fast, they are not slowed down by the slower objectives. It should be noted that because of the exchange of reduced models, each GA optimizes its own objective and also gives credit to the other objectives.

2.2 Objective Switching Genetic Algorithm for Design Optimization (OSGADO)

The main idea of OSGADO is to use a single GA that optimizes multiple objectives in a sequential order. Every objective is optimized for a certain number of evaluations, then a switch occurs and the next objective is optimized. The population is not changed when objectives are switched. This continues till the maximum number of evaluations is complete.

We modified GADO [1, 2] to create multi-objective OSGADO. OSGADO is inspired from the Vector Evaluated GA (VEGA) [9]. Schaffer (1985) proposed VEGA for generational GAs. In VEGA the population is divided into m different parts for m diff objectives; part i is filled with individuals that are chosen at random from current population according to objective i. Afterwards the mating pool is shuffled and crossover and mutation are performed as usual. Though VEGA gave encouraging results, it suffered from bias towards the extreme regions of the Pareto-optimal curve.

The OSGADO algorithm looks as follows:

1. The GA is run initially with the first objective as the measure of merit for a certain number of evaluations. The fitness of an individual is calculated based on its measure of merit and the constraint violations. Selection, crossover and mutation take place in the regular manner.

2. After a certain numbers of evaluations, the GA is run for the next objective. When the evaluations for the last objective are complete, the GA switches back to the first objective.
3. Step 2 is repeated till the maximum number of evaluations is reached.

In order to fairly compare the methods, in the experiments we first ran OEGADO and obtained the number of feasible points found by each of the two GAs. We then ran OSGADO for the number of evaluations calculated as follows,
Total number of objective evaluations = Sum of evaluations of each objective in OEGADO + Sum of the number of feasible points found by each objective in OEGADO

OSGADO has certain advantages over VEGA. In VEGA every solution is evaluated for only one of the objectives each time and therefore it can converge to individual objective optima (the extremes of the Pareto-optimal curve) without adequately sampling the middle section of the Pareto-optimal curve. However OSGADO evaluates every solution using each of the objectives at different times. So OSGADO is at less risk of converging at individual objective optima.

3 Experimental Results

In this section, we first describe the test problems used to compare the performance of OEGADO, OSGADO and NSGA-II. We then briefly discuss the parameter settings used. Finally, we discuss the results obtained for various test cases by these three methods.

3.1 Test Problems

The test problems for evaluating the performance of our methods were chosen based on significant past studies. We chose four problems from the benchmark domains commonly used in past multi-objective GA research, and two problems from the engineering domains. The degree of difficulty of these problems varies from fairly simple to difficult.

The problems chosen from the benchmark domains are BNH used by Binh and Korn [10], SRN used by Srinivas, Deb [11], TNK suggested by Tanaka [12] and OSY used by Osyczka, Kundu [13]. The problems chosen from the engineering domains are Two-Bar Truss Design used by Deb [14] and Welded Beam design used by Deb [14]. All these problems are constrained multi-objective problems. Table 1 shows the variable bounds, objective functions and constraints for all these problems.

Table 1. Test problems used in this study, all objective functions are to be minimized

Problem	Variable bounds	Objectives functions $f(\mathrm{x})$ and Constraints $C(\mathrm{x})$
BNH	$x_1 \in [0,5]$ $x_2 \in [0,3]$	$f_1(\mathrm{x}) = 4x_1^2 + 4x_2^2$ $f_2(\mathrm{x}) = (x_1-5)^2 + (x_2-5)^2$ $C_1(\mathrm{x}) \equiv (x_1-5)^2 + x_2^2 \le 25$ $C_2(\mathrm{x}) \equiv (x_1-8)^2 + (x_2+3)^2 \ge 7.7$
SRN	$x_1 \in [-20,20]$ $x_2 \in [-20,20]$	$f_1(\mathrm{x}) = 2 + (x_1-2)^2 + (x_2-2)^2$ $f_2(\mathrm{x}) = 9x_1 - (x_2-1)^2$ $C_1(\mathrm{x}) \equiv x_1^2 + x_2^2 \le 225$ $C_2(\mathrm{x}) \equiv x_1 - 3x_2 + 10 \le 0$
TNK	$x_1 \in [0,\pi]$ $x_2 \in [0,\pi]$	$f_1(\mathrm{x}) = x_1$ $f_2(\mathrm{x}) = x_2$ $C_1(\mathrm{x}) \equiv x_1^2 + x_2^2 - 1 - 0.1\cos(16\arctan\frac{x_1}{x_2}) \ge 0$ $C_2(\mathrm{x}) \equiv (x_1-0.5)^2 + (x_2-0.5)^2 \le 0.5$
OSY	$x_1 \in [0,10]$ $x_2 \in [0,10]$ $x_3 \in [1,5]$ $x_4 \in [0,6]$ $x_5 \in [1,5]$ $x_6 \in [0,10]$	$f_1(\mathrm{x}) = -[25(x_1-2)^2 + (x_2-2)^2 + (x_3-1)^2$ $+ (x_4-4)^2 + (x_5-1)^2]$ $f_2(\mathrm{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$ $C_1(\mathrm{x}) \equiv x_1 + x_2 - 2 \ge 0$ $C_2(\mathrm{x}) \equiv 6 - x_1 - x_2 \ge 0$ $C_3(\mathrm{x}) \equiv 2 - x_2 + x_1 \ge 0$ $C_4(\mathrm{x}) \equiv 2 - x_1 + 3x_2 \ge 0$ $C_5(\mathrm{x}) \equiv 4 - (x_3-3)^2 - x_4 \ge 0$ $C_6(\mathrm{x}) \equiv (x_5-3)^2 + x_6 - 4 \ge 0$
Two-bar Truss Design	$x_1 \in [0,0.01]$ $x_2 \in [0,0.01]$ $x_3 \in [1,3]$	$f_1(\mathrm{x}) = x_1\sqrt{16+x_3^2} + x_2\sqrt{1+x_3^2}$ $f_2(\mathrm{x}) = \max(\sigma_1,\sigma_2)$ $C_1(\mathrm{x}) \equiv \max(\sigma_1,\sigma_2) \le 10^5$ $\sigma_1 = 20\sqrt{16+x_3^2}\,/\,x_1x_3$ $\sigma_2 = 80\sqrt{1+x_3^2}\,/\,x_2x_3$

Welded Beam Design	$h \in [0.125,5]$ $b \in [0.125,5]$ $l \in [0.1,10]$ $t \in [0.1,10]$	$f_1(\mathrm{x}) = 1.10471h^2l + 0.04811tb(14+l)$ $f_2(\mathrm{x}) = 2.1952/t^3b$ $C_1(\mathrm{x}) \equiv 13600 - \tau(\mathrm{x}) \geq 0$ $C_2(\mathrm{x}) \equiv 30000 - \sigma(\mathrm{x}) \geq 0$ $C_3(\mathrm{x}) \equiv b - h \geq 0$ $C_4(\mathrm{x}) \equiv P_c(\mathrm{x}) - 6000 \geq 0$ $\tau = \sqrt{(\tau^{'})^2 + (\tau^{''})^2 + l\tau^{'}\tau^{''}/\sqrt{0.25(l^2+(h+t)^2)}}$ $\tau^{'} = 6000/\sqrt{2}hl$ $\tau^{''} = \frac{6000(14+0.5l)\sqrt{0.25(l^2+(h+t)^2)}}{2\sqrt{2}hl(l^2/12+0.25(h+t)^2)}$ $\sigma = 504000/t^2b$ $P_c = 64746.022(1-0.0282346t)tb^3$

3.2 Parameter Settings

Each optimization run was carried out with similar parameter settings for all the methods. The following are the parameters for the three GAs.
Let *ndim* be equal to the number of dimensions of the problems.

1. Population size: For OEGADO and OSGADO the population size was set to 10**ndim*. For NSGA-II the population size was fixed to 100 as recommended in [19].
2. Number of objective evaluations: Since the three methods work differently the number of objective evaluations is computed differently. The number of objective evaluations for OEGADO and OSGADO according to Section 2.1 and 2.2 is given as *Objective evaluations for OEGADO and OSGADO* = 2*500**ndim* + *sum of feasible points found by each GA in OEGADO model*
   NSGA-II is a generational GA, therefore for a two-objective NSGA-II:
   Total number of objective evaluations =2**population size* * *number of generations*
   Since we did not know exactly how many evaluations would be required by OEGADO before hand, to give fair treatment to NSGA-II, we set the number of generations of NSGA-II to be 10**ndim*. In effect NSGA-II ended up doing significantly more evaluations than OEGADO and OSGADO for some problems. We however did not decrease the number of generations for NSGA-II and repeat the experiments as our methods outperformed it in most domains anyway.

3.3 Results

In the following section, Figures 1-4 present the graphical results of all three methods in the order of OEGADO, OSGADO and NSGA-II for all problems. The outcomes of five runs using different seeds were unified and then the non-dominated solutions were selected and plotted from the union set for each method. We are using graphical representations of the Pareto-optimal curve found by the three methods to compare their performance.

It is worth mentioning that the number of Pareto-optimal solutions obtained by NSGA-II is limited by its population size. Our methods keep track of all the feasible solutions found during the optimization and therefore do not have any restrictions on the number of Pareto-optimal solutions found.

The BNH and the SRN (figures not shown) problems are fairly simple in that the constraints may not introduce additional difficulty in finding the Pareto-optimal solutions. It was observed that all three methods performed equally well within comparable number of objective evaluations (mentioned in Section 3.2), and gave a dense sampling of solutions along the true Pareto-optimal curve.

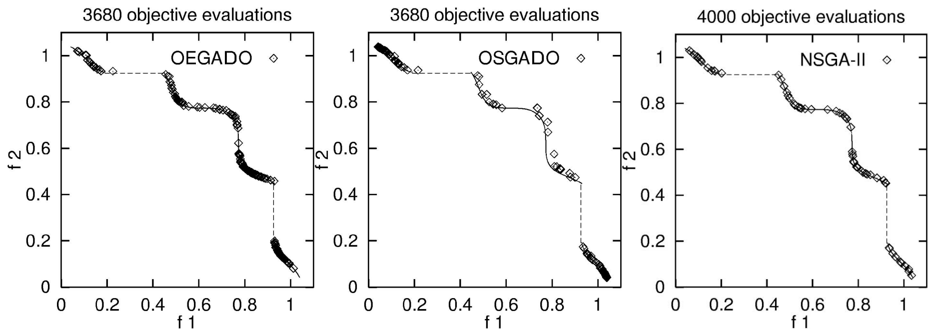

Fig. 1. Results for the benchmark problem TNK

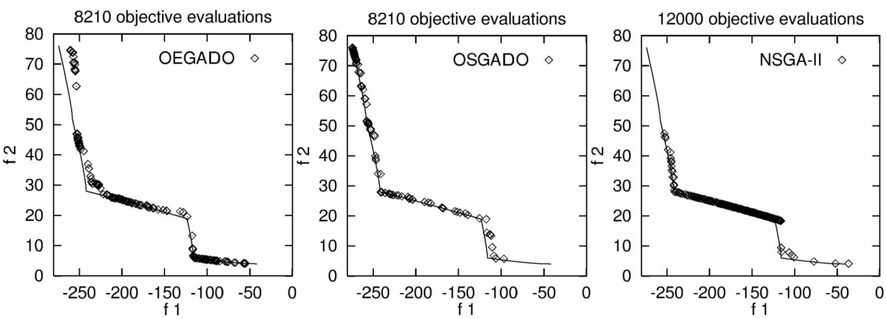

Fig. 2. Results for the benchmark problem OSY

The TNK problem (Fig. 1) and the OSY problem (Fig. 2) are relatively difficult. The constraints in the TNK problem make the Pareto-optimal set discontinuous. The constraints in the OSY problem divide the Pareto-optimal set into five regions that can demand a GA to maintain its population at different intersections of the constraint boundaries. As it can be seen from the above graphs for the TNK problem, within comparable number of fitness evaluations, the OEGADO model and the NSGA-II model performed equally well. They both displayed a better distribution of the Pareto-optimal points than the OSGADO model. OSGADO performed well at the extreme ends, but found very few Pareto points at the mid-section of the curve. For the OSY problem, it can be seen that OEGADO gave a good sampling of points at the mid-section of the curve and also found points at the extreme ends of the curve. OSGADO also performed well, giving better sampling at one of the extreme ends of the curve. NSGA-II however did not give a good sampling of points at the extreme ends of the Pareto-optimal curve and gave a poor distribution of the Pareto-optimal solutions. In this problem OEGADO and OSGADO outperformed NSGA-II while running for fewer objective evaluations.

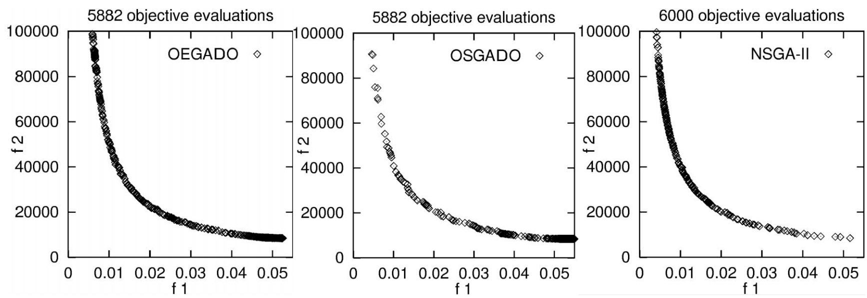

Fig. 3. Results for the Two-bar Truss design problem

For the Two-bar Truss design problem (Fig. 3), within comparable fitness evaluations, NSGA-II performed slightly better than our methods in the first objective. OEGADO showed a uniform distribution of the Pareto-optimal curve. OSGADO however gave a poor distribution at one end of the curve, but it achieved very good solutions at the other end and converged to points that the other two methods failed to reach.

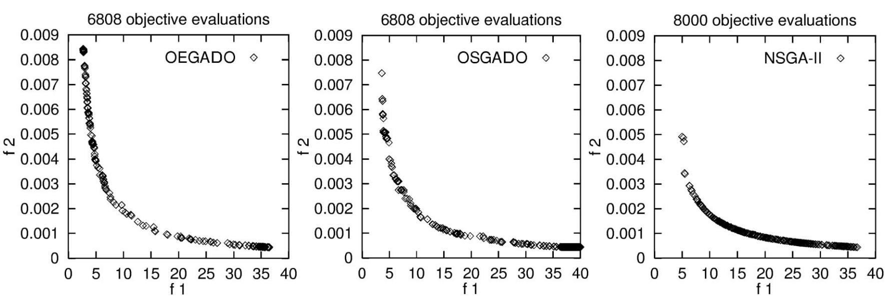

Fig. 4. Results for the Welded Beam design problem

In the Welded Beam design problem (Fig. 4), the non-linear constraints can cause difficulties in finding the Pareto solutions. As shown in Fig. 4, within comparable fitness evaluations, OEGADO outperformed OSGADO and NSGA-II in both distribution and spread [15]. OEGADO found the best minimum solution for f_1 with a value of 2.727 units. OSGADO was able to find points at the other end that the other two methods failed to reach. NSGA-II did not achieve a good distribution of the Pareto solutions at the extreme regions of the curve.

4 Conclusion and Future Work

In this paper we presented two methods for multi-objective optimization using steady state GAs, and compared our methods with a reliable and efficient generational multi-objective GA called NSGA-II. The results show that a steady state GA can be used efficiently for constrained multi-objective optimization. For the simpler problems our methods performed equally well as NSGA-II. For the difficult problems, our methods outperformed NSGA-II in most respects. In general, our methods demonstrated robustness and efficiency in their performance. OEGADO in particular performed consistently well and outperformed the other two methods in most of the domains. Moreover, our methods were able to find the Pareto-optimal solutions for all the problems in fewer objective evaluations than NSGA-II. For real-world problems, the number of objective evaluations performed can be critical as each objective evaluation takes a long time. Based on this study we believe that our methods can outperform multi-objective generational GAs for such problems. However, we need to experiment more and find out whether there are other factors that contribute to the success of our methods other than their steady state nature.

In the future, we would like to experiment with several steady state GAs as the base method. We would also like to improve both of our methods. Currently they do not have any explicit bias towards non-dominated solutions. We therefore intend to enhance them by giving credit to non-dominated solutions. OEGADO has shown promising results and we would like to further improve it, extend its implementation to handle more than two objectives and further explore its capabilities. The current OSGADO implementation can already handle more than two objectives. We would also like to use our methods for more complex real-world applications.

Acknowledgement. This research is sponsored by the US National Science Foundation under grant CTS-0121058. The program managers are Drs. Frederica Darema, C. F. Chen and Michael Plesniak.

References

1. Khaled Rasheed. GADO: A genetic algorithm for continuous design optimization. Technical Report DCS-TR-352, Department of Computer Science, Rutgers, The State University of New Jersey, New Brunswick, NJ, January 1998. Ph.D. Thesis, http://webster.cs.uga.edu/~khaled/thesis.ps.
2. Khaled Rasheed and Haym Hirsh. Learning to be selective in genetic-algorithm-based design optimization. *Artificial Intelligence in Engineering, Design, Analysis and Manufacturing,* 13:157–169, 1999.
3. Deb, K., S. Agrawal, A. Pratap, and T. Meyarivan (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Proceedings of the Parallel Problem Solving from Nature VI*, pp.849–858.

4. Khaled Rasheed and Haym Hirsh. Informed operators: Speeding up genetic-algorithm-based design optimization using reduced models. *In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2000)*, pp. 628–635, 2000.
5. K. Rasheed., S. Vattam, X. Ni. Comparison of Methods for Using Reduced Models to Speed up Design Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2002),* pp. 1180–1187, 2002.
6. Khaled Rasheed. An incremental-approximate-clustering approach for developing dynamic reduced models for design optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC'2002)*, pp. 986–993, 2002.
7. K. Rasheed., S. Vattam, X. Ni. Comparison of methods for developing dynamic reduced models for design optimization. In *Proceedings of the Congress on Evolutionary Computation (CEC'2002)*, pp. 390–395, 2002.
8. William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: the Art of Scientific Computing.* Cambridge University Press, Cambridge [England]; New York, 2nd edition, 1992.
9. J.D.Schaffer .Multi-objective optimization with vector evaluated genetic algorithms. In *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, J.J. Grefenstette, Ed., Pittsburg, PA, July 24–26 1985, pp. 93–100, sponsored by Texas Instruments and U.S. Navy Center for Applied Research in Artificial Intelligence (NCARAI).
10. Binh and Korn. MOBES: A multi-objective Evolution Strategy for constrained optimization Problems. In *Proceedings of the 3rd International Conference on Genetic Algorithm* MENDEL 1997, Brno, Czech Republic, pp.176–182.
11. Srinivas, N. and Deb, K. (1995). Multi-Objective function optimization using non-dominated sorting genetic algorithms. *Evolutionary Computation* (2), 221–248.
12. Tanaka, M. (1995). GA-based decision support system for multi-criteria, optimization. In *Proceedings of the International Conference on Systems, Man and Cybernetics-2*, pp. 1556–1561.
13. Osycza, A. and Kundu, S. (1995). A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural Optimization* (10). 94–99.
14. Deb, K. Pratap, A. and Moitra, S. (2000). Mechanical Component Design for Multiple Objectives Using Elitist Non-Dominated Sorting GA. *KanGAL* Report No. 200002.
15. Ranjithan, S.R., S.K. Chetan, and H.K. Dakshina (2001). Constraint method-based evolutionary algorithm (CMEA) for multi-objective optimization. In E.Z. et al. (Ed.), *Evolutionary Multi-Criteria Optimization 2001, Lecture Notes in Computer Science* 1993, pp. 299–313. Springer-Verlag.
16. Zitzler, E., Laumanns, M., and Thiele, L. (2001). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, *Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH)* Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland.
17. Crone, D. W., Knowles, J.D., and Oates, M.J. (2000). The Pareto Envelope-based Selection Algorithm for Multi-objective Optimization. In Schoenauer, M., Deb, K., Rudolph, g., Yao, X., Luton, E., Merelo, J.J., and Schewfel, H.-P., editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference,* pp.839–848, Paris, France. Springer. Lecture Notes in Computer Science No. 1917.
18. K. Deb. Multi-objective optimization using evolutionary algorithms. Chichester, UK: John Wiley, 2001.
19. K. Deb. S. Gulati (2001). Design of truss-structures for minimum weight using genetic algorithms, In *Journal of Finite Elements in Analysis and Design,* pp.447–465, 2001.