MULTI-OBJECTIVE OPTIMIZATION USING STEADY STATE GENETIC ALGORITHMS

by

DEEPTI RAMESH CHAFEKAR

(Under the Direction of KHALED RASHEED)

ABSTRACT

There are many interesting problems in the real world that require multiple objectives to be satisfied at the same time. For many real world design problems, the number of objective evaluations performed is a critical factor as a single objective evaluation can be quite expensive. The aim of our research is to reduce the number of objective evaluations needed to find a well-distributed sampling of the Pareto-optimal region for real world design problems that have many constraints and small feasible regions. One method called OEGADO runs several GAs concurrently with each GA optimizing one objective and exchanging information about its objective with the others. The other method called OSGADO switches attention between objectives periodically. Empirical results in several engineering and benchmark domains comparing our methods with other contemporary GAs suggest that our methods are well suited for solving real-world application problems.

INDEX WORDS: Genetic Algorithms, Multi-Objective optimization, Pareto optimal solutions, Reduced models
MULTI-OBJECTIVE OPTIMIZATION USING STEADY STATE GENETIC ALGORITHMS

by

DEEPTI RAMESH CHAFEKAR

B.E., Fr. C. Rodrigues College of Engineering, University of Mumbai, India, 2000

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment
of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2004
MULTI-OBJECTIVE OPTIMIZATION USING STEADY STATE GENETIC ALGORITHMS

by

DEEPTI RAMESH CHAFEKAR

Major Professor: Khaled Rasheed
Committee: Walter D. Potter
Liming Cai

Electronic Version Approved:
Maureen Grasso
Dean of the Graduate School
The University of Georgia
December 2004
DEDICATION

I would like to dedicate my thesis to my parents, Anjor Chafekar and Ramesh Chafekar. Thank you for all your support and encouragement.
ACKNOWLEDGEMENTS

I thank Dr. Khaled Rasheed for giving me a chance to work with him on this project. Dr. Rasheed has been my mentor and a constant source of inspiration. I greatly value his advice, suggestions, critique and encouragement. I hope he would serve to be my mentor in my future endeavors. I thank Dr. Don Potter for being on my thesis committee, and for his comments and suggestions. I would also like to thank Dr. Liming Cai for being on my thesis committee, and for his suggestions.

I would like to thank my team members, Liang Shi and Jiang Xuan. We were a great team and I really enjoyed working with my team members on this project.

This research was sponsored by the US National Science Foundation under grant CTS-0121058.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGEMENTS</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>viii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>ix</td>
</tr>
<tr>
<td><strong>CHAPTER</strong></td>
<td></td>
</tr>
<tr>
<td>1  INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Purpose of the Study</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Contribution</td>
<td>3</td>
</tr>
<tr>
<td>2  LITERATURE REVIEW AND MOTIVATION</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Vector Evaluated Genetic Algorithm (VEGA)</td>
<td>5</td>
</tr>
<tr>
<td>2.3 Strength Pareto Evolutionary Algorithm II (SPEA II)</td>
<td>6</td>
</tr>
<tr>
<td>2.4 Non-Dominated Sorting Genetic Algorithm II (NSGA II)</td>
<td>7</td>
</tr>
<tr>
<td>2.5 $\varepsilon$-MOEA</td>
<td>8</td>
</tr>
<tr>
<td>2.6 Motivation</td>
<td>9</td>
</tr>
<tr>
<td>3  CONSTRAINED MULTI-OBJECTIVE OPTIMIZATION USING STEADY STATE GENETIC ALGORITHMS</td>
<td>10</td>
</tr>
<tr>
<td>3.1 Abstract</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Introduction</td>
<td>11</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 3.1: Test Problems ...............................................................................................................20
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Results for benchmark problem TNK</td>
<td>22</td>
</tr>
<tr>
<td>3.2</td>
<td>Results for benchmark problem OSY</td>
<td>23</td>
</tr>
<tr>
<td>3.3</td>
<td>Results for Two-Bar Truss design problem</td>
<td>24</td>
</tr>
<tr>
<td>3.4</td>
<td>Results for Welded Beam problem</td>
<td>24</td>
</tr>
<tr>
<td>4.1</td>
<td>Results of TNK problem and OSY problem</td>
<td>44</td>
</tr>
<tr>
<td>4.2</td>
<td>Results of Engineering design problems</td>
<td>45</td>
</tr>
<tr>
<td>4.3</td>
<td>Results for three-objective problems</td>
<td>46</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

1.1 Purpose of the Study

There are many interesting optimization problems that have multiple criteria or design objectives to be satisfied at the same time. Such problems are known as multi-objective (multi-criteria) optimization problems. Multi-objective optimization problems are present in a variety of applications ranging from engineering design problems to operations research problems. If the objectives are conflicting and the problems have constraints then the task of finding a best possible solution that satisfies all the objectives and constraints becomes extremely difficult. The challenge in solving such problems is to be able to come up with effective compromise solutions rather than solutions satisfying single objectives. Such solutions are known as the Pareto-optimal solutions. Pareto-optimal (non-dominated) solutions are solutions in which improvement in any one of the objectives can only be achieved by degrading least one of the other objectives. The Pareto-optimal solutions can be found by applying the concept of domination [2]. A solution $X_1$ dominates $X_2$, if both the following conditions are true

1. The solution $X_1$ is not worse than $X_2$ in every objective, and

2. The solution $X_1$ is strictly better than $X_2$ in at least one objective.
A Pareto-optimal set therefore consists of all the solutions that are not dominated by any other solutions in the search space. The user can then select one or more solutions from the Pareto-optimal set depending on his preference.
There are typically many approaches for finding Pareto-optimal solutions for multi-objective optimization problems. Traditional methods include the weighted sum approach [2] and ideal-point approach [2]. The main drawback of these methods is their heavy reliance on weights. The user has to manually assign weights and has to be well versed with the objectives involved. As the priorities of the objectives change, the weights need to change accordingly and the optimization needs to be performed again. This can be a cumbersome process. A more robust approach would be to have a set of different possible solutions for the user to choose from. Recently, application of evolutionary methods for solving multi-objective optimization problems has gained a great deal of interest and focus. Population-based evolutionary algorithms [2,3,4,7,14] work simultaneously with a population of design points. These evolutionary algorithms can capture Pareto-optimal solutions as and when they are found. Due to this characteristic, evolutionary algorithms have a great potential in finding multiple optima in a single optimization.

The focus of this work is to develop methods for multi-objective optimization using evolutionary techniques (genetic algorithms). Our methods are aimed to perform the following:

1. Find a well-distributed Pareto front,
2. Converge close to the true Pareto front, and
3. Perform fewer objective evaluations.
1.2 Contribution

Genetic Algorithms have been successfully used for solving complex single-objective optimization problems. Recently, significant research has been directed towards the application of GAs for multi-objective optimization problems. For real world application problems, the task of the GA becomes even more challenging [7]. For one, the search space for these problems can be very complex and the feasible (i.e. physically realizable) region in the search space can be very small. Secondly, determining the quality (fitness) of each point may involve the use of a simulator or an analysis code that 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.

In order to tackle these challenges we felt that steady state GAs are better equipped than generational GAs. Steady state GAs have a better capacity to retain the feasible points than generational GAs. Also steady state GAs may have higher selection pressure than generational GAs [7,3]. Significant research has been carried out in the area of generational GAs [1,3,14] and considerable research has yet to be done in the area of steady state GAs. This work concerns the application of steady state GAs to multi-objective optimization problems. Also, the concept of incorporating reduced models for finding approximate fitness values has been used in the past in the context of single-objective GAs [8, 9]. This work exploits the potential of the reduced models and incorporates them in a multi-objective GA. This approach is fairly unique and to the best of our knowledge has not been adopted in reference to multi-objective GAs.

Our work proposes two novel methods for solving constrained multi-objective optimization problems using steady state GAs. These methods are relatively fast and easy to implement. It is also very easy to transform a single-objective GA to multi-objective GA using
either of our 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. Each GA incorporates reduced models [8, 9] using informed operators [9] and least square approximation techniques [6, 11, 12]. The exchange of information is mainly carried out by the exchange of the reduced models. In the second method called the Objective Switching Genetic Algorithm for Design Optimization (OSGADO) a single GA runs multiple objectives in a sequential manner. The GA periodically switches between objectives; the population however remains the same.

The rest of the thesis is organized as follows: Chapter 2 provides an overview of the related work in the area of multi-objective optimization; Chapter 3 is a conference article named "Constrained Multi-objective Optimization Using Steady State Genetic Algorithms" that appeared in The Genetic and Evolutionary Computation Conference (GECCO'2003). This article describes OEGADO and OSGADO and performs a comparison with a contemporary GA called NSGA-II. Chapter 4 is a journal article named “Constrained Multi-objective GA Optimization Using Reduced Models”. This article is to appear in IEEE Transactions on Systems, Man and Cybernetics Part C. This article describes in detail the construction and features of OEGADO and performs a comparative study of OEGADO and two other state-of-the-art GAs. Chapter 5 provides the conclusion along with a brief discussion of the future work.
CHAPTER 2
LITERATURE REVIEW AND MOTIVATION

2.1 Introduction

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 [13]. Since then, many Evolutionary algorithms for solving multi-objective optimization problems have been developed. The most recent ones are ε-Multi-Objective Evolutionary Algorithm (ε-MOEA) [4], Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) [3] and Strength Pareto Evolutionary Algorithm-II (SPEA-II) [14]. Most of these approaches propose the use of generational GAs. ε-MOEA proposed by Deb is a steady state MOEA based on the ε-dominance concept [5]. This chapter provides a brief overview of each of these methods.

2.2 Vector Evaluated Genetic Algorithm (VEGA):

A simple population-based approach for multi-objective optimization was proposed by Schaffer [13, 2] in 1985. He developed an extension of the single objective GA to a multi-objective GA that he referred to as Vector Evaluated Genetic Algorithm (VEGA). VEGA is basically a generational GA and it uses a proportional selection method. At each generation, a number of sub-populations are generated randomly. Thus for a problem with \( k \) objectives, \( k \) sub-populations would be created each of size \( N/k \) (assuming \( N \) is the size of the total population). Each sub-population is selected using proportional selection based on one of the objectives. The
entire population is then thoroughly shuffled to apply crossover and mutation operators. This algorithm is less efficient to find solutions near the mid-region of the Pareto curve and eventually converges to individual optima. Since individuals are selected based on their fitness in individual objectives, the solutions tend to be found more near the endpoints of the Pareto curve and less near the belly of the curve. This result is undesirable as the solutions near the mid-region of the Pareto-optimal curve are more of the compromised solutions. Even though VEGA had certain drawbacks, it served as a reference model for the future Multi-objective GAs.

2.3 **Strength Pareto Evolutionary Algorithm II (SPEA II):**

SPEA II is an improved version of its predecessor SPEA. SPEA II has an improved fitness assignment scheme than SPEA and it also has an effective guided search technique. SPEA however forms the basis of SPEA II. SPEA uses a regular population and an archive (external set). Initially the archive population is empty. As the algorithm proceeds the non-dominated population members found are copied to the archive. During the evolutionary process, there is a chance that the non-dominated individuals found so far could be dominated by other new individuals found. Therefore updates are made to check the archive population. Any dominated individuals or duplicates are removed from the archive during this update operation. If the size of the updated archive exceeds a predefined limit, further archive members are deleted by a clustering technique that preserves the characteristics of the non-dominated front. Afterwards, fitness values are assigned to both archive and population members. SPEA2 uses a fine-grained fitness assignment strategy that incorporates density information. In SPEA II the archive size is fixed, i.e., whenever the number of non-dominated individuals is less than the predefined archive size, dominated individuals fill up the archive; with SPEA, the archive size may vary over time. In addition SPEA II has an alternative truncation technique that preserves
boundary solutions. SPEA II promises to provide good performance in terms of convergence and diversity. The performance behavior of SPEA II indicates that it outperformed its predecessor SPEA and gave comparable performance with NSGA II. However in some cases NSGA II did seem to outperform SPEA II. The main drawback of SPEA II is that if the book keeping (maintaining the archive population and regular population) is not implemented properly the complexity of the algorithm can be very large. However the implementation of SPEA II shows progress in the area of multi-objective optimization and suggests the introduction of elitism that was later used by $\varepsilon$-MOEA.

2.4 Non-Dominated Sorting Genetic Algorithm II (NSGA-II):

The NSGA-II [3] algorithm uses the concept of non-dominated sorting effectively for finding Pareto optimal solutions. The idea of non-dominated sorting is to sort the population based on the dominance of each solution over the other. The non-dominated solutions are ranked higher, and the dominated solutions are ranked lower, this way a high preference is given to the non-dominated solutions and they can be preserved in the next generations. In order to find which solutions are non-dominated, each solution has to be compared with other solutions for each objective and at each generation. NSGA-II sorts a population of size $N$ according to the level of non-domination. Every solution from the population $P$ is checked against a partially filled population $P'$ for domination. If a solution $p$ from population $P$ dominates any member $q$ of $P'$, then solution $q$ is removed from $P'$, else if a solution $p$ is dominated by any member $q$ of $P'$ then $p$ is ignored. A solution $p$ enters $P'$ only if $p$ is not dominated by any member of $P'$. $P'$ thus contains all the non-dominated solutions. For every subsequent generation the members of $P'$ are discounted from $P$ and the same procedure is reapplied. At the end of the optimization $P'$ contains all the non-dominated solutions.
In addition to the fast non-dominated sorting, diversity maintenance in NSGA-II is carried out by the use of the crowded comparison operator. This operator guides the selection process at various stages by ranking the non-dominated solutions based on their crowding distances.

NSGA-II has low computational requirements, elitist features and constraint handling capacity. Due to NSGA-II’s unique features it has been proven to be better than many other GAs such as SPEA-II [14] and PESA-II [1]. NSGA-II has started out as a robust multi-objective GA and has been successfully used in many applications.

2.5 \( \varepsilon \)-MOEA:

\( \varepsilon \)-MOEA is a steady state GA based on the concept of \( \varepsilon \)-dominance [5]. \( \varepsilon \)-dominance does not allow two solutions with a difference less than \( \varepsilon_i \) in the \( i^{th} \) objective to be non-dominated to each other, thereby maintaining good diversity in the population. The user can select the \( \varepsilon_i \) for each of the \( i \) objectives.

\( \varepsilon \)-MOEA maintains two co-evolving populations, namely the EA population and the archive population. The archive population is based on the \( \varepsilon \)-dominance concept and the EA population is maintained using the usual dominance concept. In each iteration, two solutions (one from each population) are chosen for mating and two offspring solutions are created. Each of the offspring solutions is compared with the archive and the EA population for possible inclusion. Each offspring is compared with each member in the archive for \( \varepsilon \)-dominance and with the EA for regular dominance. If the offspring dominates any population member then the offspring replaces one of the members it dominates. Otherwise it is not accepted. As the optimization further continues, the final archive members are reported to be the obtained solutions. \( \varepsilon \)-MOEA uses an elitist approach. It maintains convergence by emphasizing non-
dominated solutions. In addition, it maintains diversity in the archive by ensuring that no two solutions are within a $\varepsilon$ from each other. It is also a computationally fast procedure. $\varepsilon$-MOEA was compared with other GAs such as SPEA-II and NSGA-II for two and three objective problems. $\varepsilon$-MOEA showed better performance than the other two algorithms. $\varepsilon$-MOEA therefore seems to be one step closer to the pragmatic implementation of Multi-objective optimization algorithms.

### 2.6 Motivation

As the techniques for Multi-objective optimization are evolving various issues in these techniques are being raised. One such issue is the trade-off between well-converged solutions and well-distributed solutions. Another issue is computational speed and time complexity of the algorithm. We found these issues challenging and developing algorithms that would tackle all these issues was the main motivation behind our work. We explored the idea of using reduced models in order to decrease the number of objective evaluations. We ran our GAs independently so that they could find individual objective solutions. We also biased our GAs to other objectives so that they could find solutions in the mid-section of the Pareto region. These strategies ensured us well-distributed Pareto-optimal solutions. Our methods used Genetic Algorithm for Design optimization (GADO) [7] as the base GA. The diversity maintenance module in GADO [7] is very efficient; our methods inherited the same module from GADO in order to maintain diversity. These were some of the design choices we made in order to realize our objectives.
CHAPTER 3

CONSTRAINED MULTI-OBJECTIVE OPTIMIZATION USING STEADY STATE GENETIC ALGORITHMS


Reprinted with the permission of the publisher Springer.

Springer Copyright notice: This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights if translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.
CHAPTER 3

3.1 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.

3.2 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:

- 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.

3.3 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.

3.3.1 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:

   \[
   \text{Total number of objective evaluations} = \text{Sum of objective evaluations of each GA} + \text{Sum of the number of feasible points found by each GA}
   \]

   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.

3.3.2 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,

\[
\text{Total number of objective evaluations} = \text{Sum of evaluations of each objective in OEGADO} + \text{Sum of the number of feasible points found by each objective in OEGADO}
\]

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.

3.4 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.

3.4.1 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 3.1 shows the variable bounds, objective functions and constraints for all these problems.
Table 3.1: Test problems used in this study, all objective functions are to be minimized

<table>
<thead>
<tr>
<th>Problem</th>
<th>Variable bounds</th>
<th>Objectives functions $f(x)$ and Constraints $C(x)$</th>
</tr>
</thead>
</table>
| BNH     | $x_1 \in [0.5]$, $x_2 \in [0.3]$ | $f_1(x) = 4x_1^2 + 4x_2^2$  
          |                  | $f_2(x) = (x_1 - 5)^2 + (x_2 - 5)^2$  
          |                  | $C_1(x) = (x_1 - 5)^2 + x_2^2 \leq 25$  
          |                  | $C_2(x) = (x_1 - 8)^2 + (x_2 + 3)^2 \geq 7.7$ |
| SRN     | $x_1 \in [-20,20]$, $x_2 \in [-20,20]$ | $f_1(x) = 2 + (x_1 - 2)^2 + (x_2 - 2)^2$  
          |                  | $f_2(x) = 9x_1 - (x_2 - 1)^2$  
          |                  | $C_1(x) = x_1^3 + x_2^3 \leq 225$  
          |                  | $C_2(x) = x_1 - 3x_2 + 10 \leq 0$ |
| TNK     | $x_1 \in [0, \pi]$, $x_2 \in [0, \pi]$ | $f_1(x) = x_1$  
          |                  | $f_2(x) = x_2$  
          |                  | $C_1(x) = x_1^2 + x_2^2 - 1 - 0.1\cos(16\arctan \frac{x_1}{x_2}) \geq 0$  
          |                  | $C_2(x) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5$ |
| OSY     | $x_1 \in [0,0.10]$, $x_2 \in [0,0.10]$, $x_3 \in [0.15]$, $x_4 \in [0.6]$, $x_5 \in [0.15]$, $x_6 \in [0.10]$ | $f_1(x) = -[25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2$  
          |                  | $+ (x_4 - 4)^2 + (x_5 - 1)^2]$  
          |                  | $f_2(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 + x_6^2$  
          |                  | $C_1(x) = x_1 + x_2 - 2 \geq 0$  
          |                  | $C_2(x) = 6 - x_1 - x_2 \geq 0$  
          |                  | $C_3(x) = 2 - x_1 + x_2 \geq 0$  
          |                  | $C_4(x) = 2 - x_1 + 3x_2 \geq 0$  
          |                  | $C_5(x) = 4 - (x_3 - 3)^2 - x_6 \geq 0$  
          |                  | $C_6(x) = (x_2 - 3)^2 + x_6 - 4 \geq 0$ |
| Two-bar Truss Design | $x_1 \in [0,0.01]$, $x_2 \in [0,0.01]$, $x_3 \in [1,3]$ | $f_1(x) = x_1 \sqrt{16 + x_1^2} + x_2 \sqrt{1 + x_1^2}$  
          |                  | $f_2(x) = \max(\sigma_1, \sigma_2)$  
          |                  | $C_1(x) = \max(\sigma_1, \sigma_2) \leq 10^6$  
          |                  | $\sigma_1 = 20\sqrt{16 + x_1^2} / x_1x_3$  
          |                  | $\sigma_2 = 80\sqrt{1 + x_1^2} / x_1x_3$ |
| Welded Beam Design | $h \in [0.125,5]$, $b \in [0.125,5]$, $l \in [0.1,10]$, $t \in [0.1,10]$ | $f_1(x) = 1.1047lh/l + 0.04811lb(14 + l)$  
          |                  | $f_2(x) = 2.1952/tb$  
          |                  | $C_1(x) = 13600 - \tau(x) \geq 0$  
          |                  | $C_2(x) = 30000 - \sigma(x) \geq 0$  
          |                  | $C_3(x) = b - h \geq 0$  
          |                  | $C_4(x) = P_c(x) - 6000 \geq 0$  
          |                  | $\tau = \sqrt{(\tau')^2 + (\tau')^2 + \tau \tau' / \sqrt{0.25(t^2 + (h + t)^2)}}$  
          |                  | $\tau' = 6000 / \sqrt{2hl}$  
          |                  | $\tau'' = 6000(14 + 0.5l) \sqrt{0.25(t^2 + (h + t)^2)} / 2\sqrt{2hl(l^2/12 + 0.25(h + t)^2)}$  
          |                  | $\sigma = 504000 / t^2b$  
          |                  | $P_c = 64746.022(1 - 0.0282346)tb^3$ |
3.4.2 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 3.3.1 and 3.3.2 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.

3.4.3 Results

In the following section, Figures 3.1-3.4 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 3.4.2), and gave a dense sampling of solutions along the true Pareto-optimal curve.

![Graphs](image)

Fig. 3.1 Results for the benchmark problem TNK
The TNK problem (Fig. 3.1) and the OSY problem (Fig. 3.2) 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.
For the Two-bar Truss design problem (Fig. 3.3), 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. 3.4 Results for the Welded Beam design problem
In the Welded Beam design problem (Fig. 3.4), the non-linear constraints can cause difficulties in finding the Pareto solutions. As shown in Fig. 3.4, 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.

3.5 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.

3.6 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.

3.7 References


CHAPTER 4

MULTI-OBJECTIVE GA OPTIMIZATION USING REDUCED MODELS

2 Deepti Chafekar, Liang Shi, Khaled Rasheed and Jiang Xuan. Accepted by IEEE Transactions on Systems, Man and Cybernetics Part C.

© 2004 IEEE. Reprinted here with permission of publisher IEEE
CHAPTER 4

4.1 Abstract

In this paper we propose a novel method for solving multi-objective optimization problems using reduced models. Our method called Objective Exchange Genetic Algorithm for Design Optimization (OEGADO) is intended for solving real-world application problems. For such problems the number of objective evaluations performed is a critical factor as a single objective evaluation can be quite expensive. The aim of our research is to reduce the number of objective evaluations needed to find a well-distributed sampling of the Pareto-optimal region by applying reduced models to steady state multi-objective GAs. OEGADO runs several GAs concurrently with each GA optimizing one objective and forming a reduced model of its objective. At regular intervals, each GA exchanges its reduced model with the others. The GAs use these reduced models to bias their search towards compromise solutions. Empirical results in several engineering and benchmark domains comparing OEGADO with two state-of-the-art Multi-Objective Evolutionary Algorithms show that OEGADO outperformed them for difficult problems.

Index Terms—Genetic Algorithms, Multi-Objective optimization, Reduced models.

4.2 Introduction

This paper concerns the application of reduced models for constrained multi-objective Genetic Algorithm (GA) optimization. The GA presented in this paper is mainly aimed at solving problems from realistic engineering design domains that usually involve simultaneous optimization of multiple and conflicting objectives with many constraints. In these problems
instead of a single optimum there is usually a set of trade-off solutions called the non-dominated solutions or Pareto-optimal solutions (also called Pareto front). 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.

Many challenges are faced in the application of GAs to engineering design domains. A large number of objective evaluations may be required in order to obtain trade-off Pareto-optimal solutions. Moreover, the search space can be complex with many constraints and a small feasible (physically realizable) region. However, determining the quality (fitness) of each point may involve the use of a simulator or an analysis code that takes a long time. Therefore it is impossible to be cavalier with the number of objective evaluations in an optimization.

For such problems multi-objective evolutionary algorithms are preferable as they can find the Pareto front in one run. Many Evolutionary algorithms for solving multi-objective optimization problems have been developed. The most recent ones are the ε-Multi-Objective Evolutionary Algorithm (ε-MOEA) [5], Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) [3], Strength Pareto Evolutionary Algorithm-II (SPEA-II) [16], and Pareto Envelope based selection-II (PESA-II) [2]. Most of these approaches propose the use of a generational GA. The ε-MOEA proposed by Deb is a steady state MOEA based on the ε-dominance concept. The main aim of these methods is obtaining a well-converged and well-distributed Pareto front. There usually exists a trade-off in these methods between obtaining a well-distributed Pareto front and the number of objective evaluations performed. Many real world application problems are computationally complex and performing a large number of objective evaluations on these problems may be very difficult. The ε-MOEA method proposed by Deb is a fast multi-objective
evolutionary algorithm in terms of computational time. The goal of our research however is the
development of a method that: (i) converges close to the true Pareto front (ii) finds a well
distributed Pareto front and (iii) Performs fewer objective evaluations.

In this paper we propose a novel method for multi-objective optimization based on the use
of a steady state GA and reduced models. This method is relatively fast and practical. It is also
easy to transform a single-objective GA to a multi-objective GA by using our method.

Our method can be viewed as a multi-objective transformation of GADO (Genetic
Algorithm for Design Optimization) [9, 12], a GA that was designed with the goal of being
suitable for use in engineering design. It uses new operators and search control strategies that
target engineering domains [12]. GADO has been successfully applied to a variety of
optimization tasks, which span many fields. It demonstrated a great deal of robustness and
efficiency relative to competing methods [9].

In GADO, each individual in the GA population represents a parametric description of an
artifact. 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 are applied to two parents
selected by linear rank based selection. One offspring point is produced, and then an existing
point in the population is replaced by the newly generated point. The replacement strategy 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. Floating point representation is used. GADO also uses
some search control strategies [12] such as a screening module that saves time by avoiding the full evaluation of points that are unlikely to correspond to good designs.

In the remainder of the paper, we provide a brief description of our proposed method along with a background study. We then briefly describe two existing competitive methods for multi-objective optimization and present results of the comparison of our method with them in several benchmark domains. Finally, we conclude the paper with a discussion of the results and future work.

4.3 Proposed Method

We propose OEGADO, a novel method for solving multi-objective optimization problems using reduced models. This section briefly describes the background behind OEGADO followed by its implementation details.

4.3.1 Background

As mentioned earlier, OEGADO can be viewed as a multi-objective transformation of GADO. It has therefore inherited many of the key features of GADO. We have previously extended GADO to incorporate reduced models [10] using informed operators [11] and a least squares approximation technique. We used these features as the basis for performing multi-objective optimizations. In this section we provide a brief description of informed operators and reduced model formation in GADO.

Informed Operators

Reduced models are fast but usually less accurate approximations of the actual fitness evaluation function. Reduced models can be physical, such as models relying on simpler physical equations, or numerical approximations such as response surfaces induced using some input-output pairs evaluated by the original expensive model. Reduced models can be used to
speed up the GA optimization or other uses (such as multi-objective optimization as we propose in this paper). The use of reduced models to save time in evolutionary optimization has been extensively researched. A recent survey about fitness approximation in evolutionary computation can be found in [7]. Informed operators (IOs) [11] offer a very convenient way to use reduced models. The main idea behind informed operators is to replace pure randomness in traditional genetic operators with decisions informed by the reduced models.

The types of informed operators used in GADO include:

**Informed initialization**: For generating an individual in the initial population we generate a number of uniformly distributed random individuals in the design space and take the best according to the reduced model.

**Informed mutation**: To do mutation, several random mutations are generated of the base point. Each random mutation is generated according to the regular method used in GADO. The mutation that appears best according to the reduced model is returned as the result of the mutation.

**Informed crossover**: To do crossover two parents are selected at random according to the usual selection strategy in GADO. Several crossovers are conducted by randomly selecting a crossover method, randomly selecting its internal parameters and applying it to the two parents to generate a potential child. Informed mutation is applied to every potential child, and the best among the mutations is the outcome of the informed crossover.

**Reduced model formation**

Our reduced model formation method is based on maintaining a large sample of the points encountered during the optimization. When the sample reaches its maximum size, new points replace existing points at random.
We keep the sample divided into clusters. Starting with one cluster, we introduce one more cluster every specific number of iterations. The reason we introduce the clusters incrementally rather than from the beginning is that this way results in more uniform sized clusters. Every new point entering the sample either becomes a new cluster (if it is time to introduce a cluster) or joins one of the existing clusters. A point belongs to the cluster whose center is closest in Euclidean distance to the point at the time in which the point joined the sample. We use clustering because it makes it possible to fit discontinuous and complicated surfaces with simpler surfaces such as quadratic approximations.

We distinguish between the approximation functions for the measure of merit and those for the sum of constraints (for a detailed discussion please refer to [10]). We use quadratic approximation functions of the form:

\[ \hat{F}(\bar{X}) = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{i=1, j=i}^{n,n} a_{ij} x_i x_j \]  

(1)

Where \( n \) is the dimension of the search space and \( x_i \) is design variable number \( i \).

4.3.2 Objective Exchange Genetic Algorithm for Design Optimization

The main idea of OEGADO is to run several single objective GAs concurrently. Each GA 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. Each single objective GA in OEGADO uses least squares approximation to form a reduced model of its own objective. Every GA exchanges its own reduced model with 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:

7. 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.

8. At intervals equal to twice the population size, each GA exchanges its reduced model with the other GA.

9. The conventional GA operators such as initialization, mutation and crossover are replaced by informed operators (IOs). As described above, 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.

10. The true fitness function is then called to evaluate the actual fitness of the newborn corresponding to the current objective.

11. The individual is then added to the population using the regular replacement strategy.

12. Steps 2 through 5 are repeated until the maximum number of objective evaluations is exhausted.

OEGADO can be extended for the general case of $n$ objectives. We implemented (and used in some of the experiments described below) a three-objective version of OEGADO using a round-robin approach in which the exchange of the reduced models takes place as follows:

- Each GA forms its own reduced model as explained earlier.

- After a given interval of evaluations each GA offers its reduced model to one of the other two GAs and obtains one of their reduced models to be used by its informed operators.
-After the second interval each GA exchanges the reduced model with the other remaining GA.

-This process continues and the GAs continue to exchange their reduced models in a round-robin fashion.

It should be noted that OEGADO is not really a multi-objective GA, but several single objective GAs working concurrently to get the Pareto front. 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 other codes to evaluate the remaining objectives. Thus for OEGADO with two objectives:

\[ \text{Total number of objective evaluations} = \text{Sum of objective evaluations of each GA} + \text{Sum of the number of feasible points found by each GA} \]

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 GAs can run asynchronously which is better for objectives having different time complexities. If some objectives are fast, they are not slowed down by the slower objectives. A limitation of our method is that it is impractical when the evaluation of one objective is computationally comparable to the evaluation of all of them. However, we are targeting truly multi-disciplinary domains such as engineering design where each objective is computed by a different simulator or analysis code.

Usually 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 seemed to handle this issue effectively.
4.4 Experimental Results

In this section, we first describe the competing methods used for comparison with OEGADO, namely ε-MOEA [5] and NSGA-II [3]. Finally, we discuss the results obtained for various test cases by these three methods.

4.4.1 Competing methods for comparison

We decided to compare our approach with two state-of-the-art methods. We describe the two methods briefly below. More detailed descriptions can be found in [5] and [3].

1) ε-MOEA

ε-MOEA is a steady-state MOEA developed for the purpose of achieving well-distributed Pareto-optimal solutions in a relatively short computational time. It is based on the ε-dominance concept that does not allow two solutions with a difference less than ε to be considered non-dominated solutions. This concept is the key feature in maintaining population diversity. ε-MOEA maintains two co-evolving populations, namely the EA population and the archive population. The archive population is based on the ε-dominance concept and the EA population is maintained using the usual dominance concept. In each iteration, two solutions (one from each population) are chosen for mating and two offspring solutions are created. Each of the offspring solutions is compared with the archive and the EA population for possible inclusion. Each offspring is compared with each member in the archive for ε-dominance and with the EA for regular dominance. If the offspring dominates any population member then the offspring replaces one of the members it dominates. Otherwise it is not accepted. As the optimization further continues, the final archive members are reported to be the obtained solutions. Thus the ε-MOEA demonstrates an elitist approach with good diversity maintenance.
2) **NSGA-II**

NSGA-II is a fast non-dominated sorting based multi-objective evolutionary algorithm. It can be viewed as an improved version of NSGA. It proposes a fast non-dominated sorting approach by incorporating a better book-keeping strategy that reduces the complexity involved in the non-dominated sorting procedure in every generation.

NSGA-II sorts a population of size $N$ according to the level of non-domination. Every solution from the population $P$ is checked against a partially filled population $P'$ for domination. If a solution $p$ from population $P$ dominates any member $q$ of $P'$, then solution $q$ is removed from $P'$, else if a solution $p$ is dominated by any member $q$ of $P'$ then $p$ is ignored. A solution $p$ enters $P'$ only if $p$ is not dominated by any member of $P'$. $P'$ thus contains all the non-dominated solutions. For every subsequent generation the members of $P'$ are discounted from $P$ and the same procedure is reapplied. At the end of the optimization $P'$ contains all the non-dominated solutions.

In addition to the fast non-dominated sorting, diversity maintenance in NSGA-II is carried out by the use of the crowded comparison operator. This operator guides the selection process at various stages by ranking the non-dominated solutions based on their crowding distances.

Due to its low computational requirements, elitist features and constraint handling capacity, NSGA-II has been successfully used in many applications. It proved to be better than many other multi-objective optimization GAs.

**4.4.2 Test Problems**

The test problems for evaluating the performance of our method were chosen based on significant past studies. We compared the performance of OEGADO, $\varepsilon$-MOEA and NSGA-II on
a number of test problems having constraints with varying degrees of difficulty and having two or three objectives. We chose two two-objective and two three-objective problems from the benchmark domains commonly used in multi-objective GA research and two two-objective problems from the engineering domains for a total of six problems.

The problems chosen from the benchmark domains are TNK suggested by Tanaka [14], OSY used by Osyczka and Kundu [8], Rémy [13] used by Coello and Cortés [1] and DTLZ8 proposed by Deb [5]. The problems chosen from the engineering domains are Two-Bar Truss Design used by Deb [6] and Welded Beam design used by Deb [6]. All these problems are constrained multi-objective problems.

For OEGADO, the population size was set to the default value recommended in [9] which is ten times the dimension of the problem except for the DTLZ8 problem where the population size was set to half that value. For e-MOEA the EA population size was set to 10 and the archive population was set to 100 as recommended in [5]. For NSGA-II the population size was fixed to 100 as recommended in [4] except for the DTLZ8 problem where the population size was set to 150.

Since we did not know exactly how many evaluations would be required by OEGADO before hand (as it depends on the number of feasible points found), to give fair treatment to NSGA-II, we set the number of generations of NSGA-II liberally. In effect NSGA-II ended up doing significantly more evaluations than OEGADO. We however did not decrease the number of generations for NSGA-II and repeat the experiments as our method outperformed it in most domains anyway. We ran the e-MOEA experiments last and gave it slightly more evaluations than OEGADO.
No attempt was made to control or measure the actual CPU time of the different methods because in real-world applications the number of evaluations usually dominates any other bookkeeping overhead [9, 12].

4.4.3 Results

Figures 4.1-4.3 present the graphical results of all three methods for all problems. Following the experimental methodology proposed in [15], the outcomes of five runs using different seeds were combined and then the non-dominated solutions were selected and plotted from the union set for each method. We used graphical representations of the Pareto fronts found by the three methods to compare their performance. We also indicated the average number of objective evaluations on each graph.

The TNK problem (Fig. 4.1a) and the OSY problem (Fig. 4.1b) 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 a comparable number of fitness evaluations, the three methods performed well. OEGADO and $\varepsilon$-MOEA however gave a denser sampling near the Pareto-region than NSGA-II.

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. The $\varepsilon$-MOEA gave a very good sampling at the mid-section of the curve but gave a very poor distribution at the extreme ends of the curve. NSGA-II did not give a good sampling of points at the extreme ends of the Pareto front and gave a poor distribution of the Pareto optimal solutions. In this problem OEGADO outperformed $\varepsilon$-MOEA and NSGA-II while running fewer objective evaluations.
For the Two-bar Truss design problem (Fig. 4.2a), $\varepsilon$-MOEA performed very well on the mid-section of the curve but not on the extremes. OEGADO however gave a good distribution of points on the whole curve especially on the extremes. NSGA-II performed well on one side of the curve but not the other.

In the Welded Beam design problem (Fig. 4.2b), the non-linear constraints can cause difficulties in finding the Pareto solutions. The figure shows that $\varepsilon$-MOEA performed better than the other methods at the mid section of the Pareto curve. $\varepsilon$-MOEA also found a good enough distribution of points along the Pareto front. OEGADO gave a good distribution of points at the Pareto region but did not give a very dense sampling of points. NSGA-II performed poorly with respect to finding a good distribution of points and only performed well on the second objective.

For the Rémy problem (Fig. 4.3a) which is a three-objective problem, it can be seen that OEGADO not only performed well at the middle region of the Pareto-optimal curve in three dimensions but also performed well at the extreme regions. $\varepsilon$-MOEA performed very well only at the middle region. NSGA-II however performed poorly in both the middle and the extreme region. In this problem OEGADO outperformed $\varepsilon$-MOEA and NSGA-II.

The DTLZ8 problem (Fig. 4.3b) is a three-objective, 30-variable problem. The overall Pareto front here is a combination of a straight line segment and a triangle in two different planes. It is a difficult task for any multi-objective evolutionary method to find solutions in both regions and also maintain a good distribution of solutions on the hyper-plane. As seen from Figure 4.3b, OEGADO found a good distribution of solutions near both the Pareto-optimal line and plane, while $\varepsilon$-MOEA did not give any distribution along the line and gave only a fair distribution on the plane. NSGA-II gave a very poor distribution of solutions on both the line and the plane.
Fig. 4.1 Results of the TNK problem and the OSY problem
Fig. 4.2 Results for the Engineering design problems
Fig. 4.3 Results for the three-objective problems
4.5 Conclusion and Future Work

In this paper we presented a novel method for multi-objective GA optimization. We introduced the idea of using reduced models for multi-objective GA optimization wherein each GA runs concurrently and exchanges its reduced model with the other GAs as opposed to maintaining non-dominated solutions. The exchanged reduced models bias each GA towards the other objectives thus directing the optimization towards the Pareto front. Also since each GA runs independently, the method can easily find solutions in the extreme regions as well. In this way OEGADO is able to find a well-distributed Pareto front with fewer objective evaluations. OEGADO is based on concurrent execution of all the GAs which makes it fast and efficient in multi-processing environments. Thus it is a promising method for multi-objective optimization in real world application domains wherein the number of objective evaluations is a critical factor.

We presented a comparison of our method with two reliable, efficient and top of the line methods namely ε-MOEA and NSGA-II. The results show that for the simpler two-objective problems OEGADO performed as well as ε-MOEA and NSGA-II. For the difficult two-objective problems, OEGADO performed better than or as good as ε-MOEA and outperformed NSGA-II in most respects. For the three-objective problems OEGADO performed well in many respects. OEGADO produced a good approximation of the true Pareto front for the difficult three-objective problems, which the other two methods failed to produce. In general, OEGADO demonstrated robustness and efficiency in its performance. Moreover, OEGADO was able to find the Pareto-optimal solutions for all the problems in fewer objective evaluations than ε-MOEA and NSGA-II. For real-world problems, the number of objective evaluations performed can be critical as each objective evaluation takes a long time.
In the future, we would like to extend the implementation of OEGADO to handle more than three objectives by using a weighted sum approach rather than the round-robin approach. In the weighted sum approach we take a weighted sum of the reduced models of the other objectives (i.e. other than the objective being directly optimized by the respective GA). We would further like to explore the capabilities of OEGADO by challenging it with more complex problems with many variables and many objectives from real world application domains.

4.6 References


Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks
Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35,
CH-8092 Zurich, Switzerland.
CHAPTER 5
CONCLUSION AND FUTURE WORK

This work presents the design philosophy, decision choices and implementation details of two novel algorithms for solving multi-objective optimization problems for real world design problems. The two algorithms, OEGADO and OSGADO are both steady state GAs and use different approaches to find the Pareto-optimal solutions. OSGADO is a more traditional GA that optimizes different objectives sequentially and switches between them. OEGADO on the other hand optimizes each objective concurrently and occasionally each optimization exchanges its reduced model with the other optimizations. These reduced models bias each GA towards other objectives thus directing the optimization towards the Pareto front. Also since each GA runs independently, the method can easily find solutions in the extreme regions. In this way OEGADO is able to find a well-distributed Pareto front with fewer objective evaluations. OEGADO is based on concurrent execution of all the GAs which makes it fast and efficient in multi-processing environments.

The performance of these two algorithms compared with the state-of-the-art contemporary GAs (NSGA-II and ε-MOEA), suggests that our methods performed equally well and also outperformed other GAs in many cases. A comparison between OEGADO, OSGADO and NSGA-II suggests that for simpler problems all three methods performed equally well. For the difficult problems OEGADO and OSGADO outperformed NSGA-II in most aspects. OEGADO however performed better than OSGADO and NSGA-II in many respects.
A comparison between OEGADO, NSGA-II and \( \varepsilon \)-MOEA suggests that for the simpler two-objective problems OEGADO performed as well as \( \varepsilon \)-MOEA and NSGA-II. For the difficult two-objective problems, OEGADO performed better than or as good as \( \varepsilon \)-MOEA and outperformed NSGA-II in most respects. For the three-objective problems OEGADO performed well in many respects. OEGADO produced a good approximation of the true Pareto front for the difficult three-objective problems, which the other two methods failed to produce.

Moreover, OEGADO was able to find the Pareto-optimal solutions for all the problems in fewer objective evaluations than \( \varepsilon \)-MOEA and NSGA-II. This indicates that the presence of reduced models can improve the performance of the GA and can reduce the number of objective evaluations. For real-world problems, the number of objective evaluations performed can be critical as each objective evaluation can take a long time. We therefore conclude that our methods can be efficiently deployed for solving constrained multi-objective optimization problems in the real world domains.

OEGADO appears to be a promising GA; in the future we would like to make OEGADO more robust. We would like to extend the implementation of OEGADO to handle more than three objectives by using a weighted sum approach rather than the round-robin approach. In the weighted sum approach we take a weighted sum of the reduced models of the other objectives (i.e. other than the objective being directly optimized by the respective GA. We would also like to experiment with the idea of using reduced models in OSGADO. We wish to experiment with other faster techniques for forming reduced models. We would further like to explore the capabilities of OEGADO and OSGADO by challenging it with more complex problems with many variables and many objectives from real world application domains.
REFERENCES


