Multi-Objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems

Kalyanmoy Deb*
Kanpur Genetic Algorithms Laboratory (KanGAL)
Department of Mechanical Engineering
Indian Institute of Technology Kanpur
Kanpur, PIN 208 016, India
E-mail: deb@iitk.ac.in

Abstract

In this paper, we study the problem features that may cause a multi-objective genetic algorithm (GA) difficulty to converge to the true Pareto-optimal front. Identification of such features helps us develop difficult test problems for multi-objective optimization. Multi-objective test problems are constructed from single-objective optimization problems, thereby allowing known difficult features of single-objective problems (such as multi-modality or deception) to be directly transferred to the corresponding multi-objective problem. In addition, test problems having features specific to multi-objective optimization are also constructed. The construction methodology allows a simpler way to develop test problems having other difficult and interesting problem features. More importantly, these difficult test problems will enable researchers to test their algorithms for specific aspects of multi-objective optimization in the coming years.

1 Introduction

After about a decade since the pioneering work by Schaffer (1984; 1985), a number of studies on multi-objective genetic algorithms (GAs) have been pursued since the year 1994, although most of these studies took a hint from Goldberg (1989). The primary reason for these studies is a unique feature of GAs—population approach—that make them highly suitable to be used in multi-objective optimization. Since GAs work with a population of solutions, multiple Pareto-optimal solutions can be captured in a GA population in a single simulation run. During the year 1993-94, a number of independent GA implementations (Fonseca and Fleming, 1993; Horn, Nafploitis, and Goldberg, 1994; Srinivas and Deb, 1994) emerged. Later, a number of other researchers have used these implementations in various multi-objective optimization applications with success (Cunha, Oliviera, and Covas, 1997; Eheart, Cieniawski, and Ranjithan, 1993; Mitra, Deb, and Gupta, 1998; Parks and Miller, 1998; Weile, Michelsson, and Goldberg, 1996). A number of studies have also concentrated in developing new and improved GA implementations (Fonseca and Fleming, 1998; Leung et al., 1998; Kursawe, 1990; Laumanns, Rudolph, and Schwefel, 1998; Zitzler and Thiele, 1998a). Fonseca and Fleming (1995) and Horn (1997) have presented overviews of different multi-objective GA implementations. Recently, van Veldhuizen and Lamont (1998) have made a survey of test problems that exist in the literature.

Despite all these interests, there seems to be a lack of studies discussing problem features that may cause multi-objective GAs difficulty. The literature also lacks a set of test problems with known and controlled difficulty measure, an aspect of test problems that allows an optimization algorithm to be tested

*Currently visiting the Computer Science Department/LS11, University of Dortmund, Germany (deb@ls11.informatik.uni-dortmund.de) systematically. On the face of it, studies on seeking problem features causing difficulty to an algorithm may seem a pessimist's job, but we feel that true efficiency of an algorithm reveals when it is applied to difficult and challenging test problems, and not to easy problems. Such studies in single-objective GAs (studies on deceptive test problems, NK 'rugged' landscapes, and others) have all enabled researchers to compare GAs with other search and optimization methods and establish the superiority of GAs in solving difficult optimization problems to their traditional counterparts. Moreover, those studies have also helped us understand the working principle of GAs much better and paved ways to develop new and improved GAs (such as messy GAs (Goldberg, Korb, and Deb, 1990), Gene expression messy GA (Kargupta, 1996), CHC (Eshelman, 1990), Genitor (Whitley, 1989)), Linkage learning GAs (Harik, 1997), and others.

In this paper, we attempt to highlight a number of problem features that may cause a multi-objective GA difficulty. Keeping these properties in mind, we then show procedures of constructing multi-objective test problems with controlled difficulty. Specifically, there exists some difficulties that both a multi-objective GA and a single-objective GA share in common. Our construction of multi-objective problems from single-objective problems allow such difficulties (well studied in single-objective GA literature) to be directly transferred to an equivalent multi-objective GA. Besides, multi-objective GAs have their own specific difficulties, some of which are also discussed. In most cases, test problems are constructed to study an individual problem feature that may cause a multi-objective GA difficulty. In some cases, simulation results using a non-dominated sorting GA (NSGA) (Srinivas and Deb, 1994) are also presented to support our arguments.

In the remainder of the paper, we discuss and define local and global Pareto-optimal solutions, followed by a number of difficulties that a multi-objective GA may face. We show the construction of a simple two-variable two-objective problem from single-variable, single-objective problems and show how multi-modal and deceptive multi-objective problems may cause a multi-objective GA difficulty. Thereafter, we present a generic two-objective problem of varying complexity constructed from generic single-objective optimization problems. Specifically, systematic construction of multi-objective problems having convex, non-convex, and discontinuous Pareto-optimal fronts is demonstrated. We then discuss the issue of using parameter-space versus function-space based niching and suggest which to use when. The construction methodology used here is simple. Various aspects of problem difficulties are functionally decomposed so that each aspect can be controlled by using a separate function. The construction procedure allows many other aspects of single-objective test functions that exist in the literature to be easily incorporated to have a test problem with a similar difficulty for multi-objective optimization. Finally, a number of future challenges in the area of multi-objective optimization are discussed.

2 Pareto-optimal Solutions

Pareto-optimal solutions are optimal in some sense. Therefore, like single-objective optimization problems, there exist possibilities of having both *local* and *global* Pareto-optimal solutions. Before we define both these types of solutions, we first discuss *dominated* and *non-dominated* solutions.

For a problem having more than one objective function (say, f_j, $j = 1, \ldots, M$ and $M > 1$), any two solutions $x^{(1)}$ and $x^{(2)}$ can have one of two possibilities—one dominates the other or none dominates the other. A solution $x^{(1)}$ is said to dominate the other solution $x^{(2)}$, if both the following conditions are true:

1. The solution $x^{(1)}$ is no worse (say the operator $\prec$ denotes worse and $\succ$ denotes better) than $x^{(2)}$ in all objectives, or $f_j(x^{(1)}) \not\prec f_j(x^{(2)})$ for all $j = 1, 2, \ldots, M$ objectives.

2. The solution $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective, or $f_{\bar{j}}(x^{(1)}) \succ f_{\bar{j}}(x^{(2)})$ for at least one $\bar{j} \in \{1, 2, \ldots, M\}$.

If any of the above condition is violated, the solution $x^{(1)}$ does not dominate the solution $x^{(2)}$. If $x^{(1)}$ dominates the solution $x^{(2)}$, it is also customary to write $x^{(2)}$ is dominated by $x^{(1)}$, or $x^{(1)}$ is non-dominated by $x^{(2)}$, or, simply, among the two solutions, $x^{(1)}$ is the non-dominated solution.

The above concept can also be extended to find a non-dominated set of solutions in a set (or population) of solutions. Consider a set of N solutions, each having M (> 1) objective function values. The following procedure can be used to find the non-dominated set of solutions:

Step 0: Begin with $i = 1$.

Step 1: For all $j \neq i$, compare solutions $x^{(i)}$ and $x^{(j)}$ for domination using the above two conditions for all M objectives.

Step 2: If for any j, $x^{(i)}$ is dominated by $x^{(j)}$, mark $x^{(i)}$ as 'dominated'.

Step 3: If all solutions (that is, when $i = N$ is reached) in the set are considered, Go to Step 4, else increment i by one and Go to Step 1.

Step 4: All solutions that are not marked 'dominated' are non-dominated solutions.

A population of solutions can be classified into groups of different non-domination levels (Goldberg, 1989). When the above procedure is applied for the first time in a population, the resulting set is the non-dominated set of first (or, best) level. In order to have further classifications, these non-dominated solutions can be temporarily counted out from the original set and the above procedure can be applied once more. What results is a set of non-dominated solutions of second (or, next-best) level. These new set of non-dominated solutions can be counted out and the procedure may be applied again to find the third-level non-dominated solutions. This procedure can be continued till all members are classified into a non-dominated level. It is important to realize that the number of non-domination levels in a set of N solutions is bound to lie within $[1, N]$. The minimum case of one non-domination level occurs when no solution dominates any other solution in the set, thereby classifying all solutions of the original population into one non-dominated level. The maximum case of N non-domination levels occurs, when there is hierarchy of domination of each solution by exactly one other solution in the set.

In a set of N solutions, the first-level non-dominated solutions are candidates for possible Pareto-optimal solutions. However, they need not be Pareto-optimal solutions. The following definitions ensure them whether they are local or global Pareto-optimal solutions:

Local Pareto-optimal Set: If for every member x in a set $\underline{P}$, there exist no solution y satisfying $\|y - x\|_\infty \leq \epsilon$, where ϵ is a small positive number (in priciple, y is obtained by perturbing x in a small neighborhood), which dominates any member in the set $\underline{P}$, then the solutions belonging to the set $\underline{P}$ constitute a local Pareto-optimal set.

Global Pareto-optimal Set: If there exists no solution in the search space which dominates any member in the set $\bar{P}$, then the solutions belonging to the set $\bar{P}$ constitute a global Pareto-optimal set.

The size and shape of Pareto-optimal fronts usually depend on the number of objective functions and interactions among the individual objective functions. If the objectives are 'conflicting' to each other, the resulting Pareto-optimal front may span larger than if the objectives are more 'cooperating'[1]. However, in most interesting multi-objective optimization problems, the objectives are 'conflicting' to each other and usually the resulting Pareto-optimal front (whether local or global) contains many solutions, which must be found using a multi-objective optimization algorithm.

[1]The terms 'conflicting' and 'cooperating' are used loosely here. If two objectives have similar individual optimum solutions and with similar individual function values, they are 'cooperating', as opposed to a 'conflicting' situation where both objectives have drastically different individual optimum solutions and function values.

3 Principles of Multi-Objective Optimization

It is clear from the above discussion that a multi-objective optimization problem usually results in a set of Pareto-optimal solutions, istead of one single optimal solution[2]. Thus, the objective in a multi-objective optimization is different from that in a single-objective optimization. In multi-objective optimization the goal is to find as many different Pareto-optimal solutions as possible. Since classical optimization methods work with a single solution in each iteration (Deb, 1995; Reklaitis, Ravindran and Ragsdell, 1983), in order to find multiple Pareto-optimal solutions they are required to be applied more than once, hopefully finding one distinct Pareto-optimal solution each time. Since GAs work with a population of solutions, a number of Pareto-optimal solutions can be captured in one single run of a multi-objective GA with appropriate adjustments to its operators. This aspect of GAs makes them naturally suited to solve multi-objective optimization problems for finding multiple Pareto-optimal solutions. Thus, this is no surprise that a number of different multi-objective GA implementations exist in the literature (Fonseca and Fleming, 1995; Horn, Nafploitis, and Goldberg, 1994; Srinivas and Deb, 1994; Zitzler and Thiele, 1998b).

Before we discuss the problem features that may cause multi-objective GAs difficulty, let us mention a couple of matters[3] that are not addressed in the paper. First, in discussions in this paper, we consider all objectives to be of minimization type. It is worth mentioning here that identical properties as discussed here may also exist in problems with mixed optimization types (some are minimization and some are maximization). The use of non-dominated solutions in multi-objective GAs allows an elegant way to suffice the discussion to have only for one type of problems. The meaning of 'worse' or 'better' discussed in Section 2 takes care of other cases. Second, although we refer to multi-objective optimization throughout the paper, we only restrict ourselves to two objectives. This is because we believe that the two-objective optimization brings out the essential features of multi-objective optimization, although scalability of an optimization method to solve more than two objectives is an issue which needs attention. Moreover, to understand the interactions among multiple objectives, it is an usual practice to investigate pair-wise interactions among objectives (Covas, Cunha, and Oliveira, in press). Thus, we believe that we need to understand the mechanics behind what cause GAs may or may not work in a two-objective optimization problem better, before we tackle more than two objectives.

Primarily, there are two tasks that a multi-objective GA should do well in solving multi-objective optimization problems:

1. Guide the search towards the global Pareto-optimal region, and
2. Maintain population diversity in the current non-dominated front.

We discuss the above two tasks in the following subsections and highlight when a GA would have difficulty in achieving each of the above tasks.

3.1 Difficulties in converging to Pareto-optimal front

The first task ensures that, instead of converging to any set, multi-objective GAs proceed towards the global Pareto-optimal front. Convergence to the true Pareto-optimal front may not happen because of various reasons:

1. Multimodality,
2. Deception,
3. Isolated optimum, and

[2]In multi-modal function optimization, there may exist more than one optimal solution, but usually the interest there is to find global optimal solutions having identical objective function value.

[3]A number of other matters which need immediate attention are also outlined in Section 7.

4. Collateral noise.

All the above features are known to cause difficulty in single-objective GAs (Deb, Horn, and Goldberg, 1992) and when present in a multi-objective problem may also cause difficulty to a multi-objective GA.

In tackling a multi-objective problem having multiple Pareto-optimal fronts, a GA, like many other search and optimization methods, may get stuck to a local Pareto-optimal front. Later, we create a multi-modal multi-objective problem and show that a multi-objective GA can get stuck at a local Pareto-optimal front, if appropriate GA parameters are not used.

Deception is a well-known phenomenon in the studies of genetic algorithms (Deb and Goldberg, 1993; Goldberg 1989; Whitley, 1990). Deceptive functions cause GAs to get misled towards deceptive attractors. There is a difference between the difficulties caused by multi-modality and by deception. For deception to take place, it is necessary to have at least two optima in the search space (a true attractor and a deceptive attractor), but almost the entire search space *favors* the deceptive (non-global) optimum, whereas multi-modality may cause difficulty to a GA, merely because of the sheer number of different optima where a GA can get stuck to. There even exists a study where both multi-modality and deception coexist in a function (Deb, Horn, and Goldberg, 1993), thereby making these so-called massively multi-modal deceptive problems even harder to solve using GAs. We shall show how the concepts of single-objective deceptive functions can be used to create multi-objective deceptive problems, which are also difficult to solve using multi-objective GAs.

There may exist some problems where most of the search space may be fairly flat, giving rise to virtually no information of the location of the optimum. In such problems, the optimum is placed isolated from the rest of the search space. Since there is no useful information that most of the search space can provide, no optimization algorithm will perform better than an exhaustive search method. Multi-objective optimization methods are also no exception to face difficulty in solving a problem where the true Pareto-optimal front is isolated in the search space. Even though the true Pareto-optimal front may not be totally isolated from the rest of the search space, reasonable difficulty may come if the density of solutions near the Pareto-optimal front is significantly small compared to other regions in the search space.

Collateral noise comes from the improper evaluation of low-order building blocks (partial solutions which may lead towards the true optimum) due to the excessive noise that may come from other part of the solution vector. These problems are usually 'rugged' with relatively large variation in the function landscapes. However, if adequate population size (adequate to discover signal from the noise) is considered, such problems can be solved using GAs (Goldberg, Deb, and Clark, 1992). Multi-objective problems having such 'rugged' functions may also cause difficulties to multi-objective GAs, if adequate population size is not used.

3.2 Difficulties in maintaining diverse Pareto-optimal solutions

As it is important for a multi-objective GA to find solutions in the true Pareto-optimal front, it is also necessary to find solutions as diverse as possible in the Pareto-optimal front. If only a small fraction of the true Pareto-optimal front is found, the purpose of multi-objective optimization is not served. This is because, in such cases, many interesting solutions with large trade-offs among the objectives may not have been discovered.

In most multi-objective GA implementations, a specific diversity-maintaining operator, such as a niching technique (Deb and Goldberg, 1989; Goldberg and Richardson, 1987), is used to find diverse Pareto-optimal solutions. However, the following features of a multi-objective optimization problem may cause multi-objective GAs to have difficulty in maintaining diverse Pareto-optimal solutions:

1. Convexity or non-convexity in the Pareto-optimal front,
2. Discontinuity in the Pareto-optimal front,
3. Non-uniform distribution of solutions in the Pareto-optimal front.

There exist multi-objective problems where the resulting Pareto-optimal front is non-convex. Although it may not be apparent but in tackling such problems, a GA's success to maintain diverse Pareto-optimal solutions largely depends on fitness assignment procedure. In some GA implementations, the fitness of a solution is assigned proportional to the number of solutions it dominates (Leung et al., 1998; Zitzler and Thiele, 1998b). Figure 1 shows how such a fitness assignment favors intermediate solutions, in the case of problems with convex Pareto-optimal front (the left figure). With respect to an individual champion

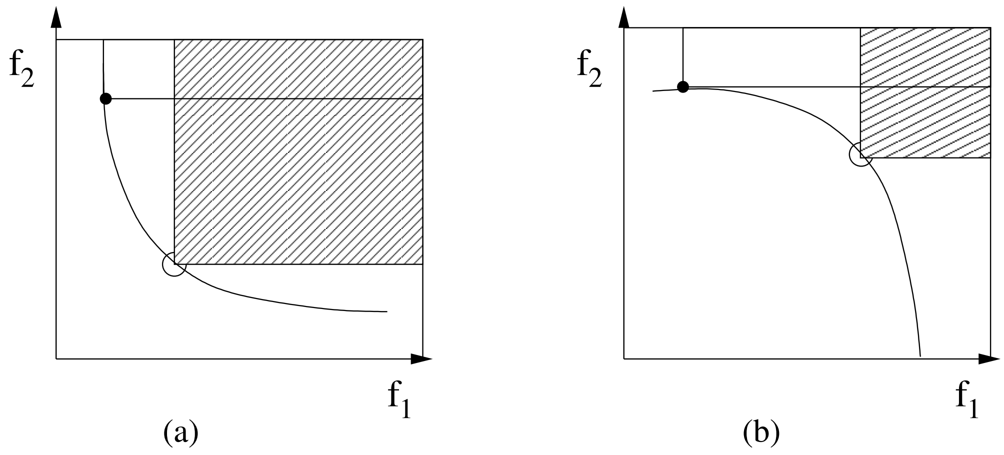

Figure 1: The fitness assignment proportional to the number of dominated solutions (the shaded area) favors intermediate solutions in convex Pareto-optimal front (a), compared to that in non-convex Pareto-optimal front (b).

solution (marked with a solid bullet in the figures), the proportion of dominated region covered by an intermediate solution is more in Figure 1(a) than in 1(b). Using such a GA (with GAs favoring solutions having more dominated solutions), there is a natural tendency to find more intermediate solutions than solutions with individual champions, thereby causing an artificial bias towards some portion of the Pareto-optimal region.

In some multi-objective optimization problems, the Pareto-optimal front may not be continuous, instead it is a set of discretely spaced continuous sub-regions (Poloni et al., in press; Schaffer, 1984). In such problems, although solutions within each sub-region may be found, competition among these solutions may lead to extinction of some sub-regions.

It is also likely that the Pareto-optimal front is not uniformly represented by feasible solutions. Some regions in the front may be represented by a higher density of solutions than other regions. We show one such two-objective problem later in this study. In such cases, there is a natural tendency for GAs to find a biased distribution in the Pareto-optimal region. The performance of multi-objective GAs in these problems would then depend on the principle of niching method used. As appears in the literature, there are two ways to implement niching—parameter-space based (Srinivas and Deb, 1994) and function-space based (Fonseca and Fleming, 1995) niching. Although both can maintain diversity in the Pareto-optimal front, each method means diversity in its own sense. Later, we shall show that the diversity in Pareto-optimal solution vectors is not guaranteed when function-space niching is used, in some complex multi-objective optimization problems.

3.3 Constraints

In addition to above difficulties, the presence of 'hard' constraints in a multi-objective problem may cause further difficulties. Constraints may cause difficulties in both aspects discussed earlier. That is, they may cause hindrance for GAs to converge to the true Pareto-optimal region and they may also cause difficulty in maintaining a diverse set of Pareto-optimal solutions. The success of a multi-objective GA in tackling both these problems will largely depend on the constraint-handling technique used. Typically, a simple penalty-function based method is used to penalize each objective function (Deb and Kumar, 1995; Srinivas and Deb, 1994; Weile, Michelsson and Goldberg, 1996). Although successful applications are reported in the literature, penalty function methods demand an appropriate choice of a penalty parameter for each constraint. Usually the objective functions may have different ranges of function values (such as cost function varying in thousands of dollars, whereas reliability values varying in the range zero to one). In order to maintain equal importance to objective functions and constraints, different penalty parameters must have to be used with different objective functions. Recently, a couple of efficient constraint-handling techniques are developed for single-objective GAs (Deb, in press; Koziel and Michalewicz, 1998), which may also be implemented in a multi-objective GA, instead of the simple penalty function approach. In this paper, we realize that the presence of constraints makes the job of any optimizer difficult, but we defer a consideration of constraints in multi-objective optimization to a later study.

In addition to the above problem features, there may exist other difficulties (such as the search space being discontinuous, rather than continuous). There may also exist problems having a combination of above difficulties. In the following sections, we demonstrate the problem difficulties mentioned above by creating simple to complex test problems. A feature of these test problems is that each type of problem difficulty mentioned above can be controlled using an independent function used in the construction process. Since most of the above difficulties are also common to GAs in solving single-objective optimization problems, we use a simple construction methodology for creating multi-objective test problems from single-objective optimization problems. The problem difficulty associated with the chosen single-objective problem is then transferred to the corresponding multi-objective optimization problem. Avoiding to present the most general case first (which may be confusing at first), we shall present a simple two-variable, two-objective optimization problem, which can be constructed from a single-variable, single-objective optimization problem.

In some instances, one implementation of a multi-objective binary GA (non-dominated sorting GA (NSGA) (Srinivas and Deb, 1994)) is applied on test problems to investigate the difficulties which a multi-objective GA may face.

4 A Special Two-Objective Optimization Problem

Let us begin our discussion with a simple two-objective optimization problem with two problem variables x_1 (> 0) and x_2:

$$\begin{aligned}
\text{Minimize} \quad f_1(x_1, x_2) &= x_1, && (1)\\
\text{Minimize} \quad f_2(x_1, x_2) &= \frac{g(x_2)}{x_1}, && (2)
\end{aligned}$$

where $g(x_2)$ (> 0) is a function of x_2 only. Thus, the first objective function f_1 is a function of x_1 only[4] and the function f_2 is a function of both x_1 and x_2. In the function space (that is, a space with (f_1, f_2) values), the above two functions obey the following relationship:

$$f_1(x_1, x_2) \cdot f_2(x_1, x_2) = g(x_2). \tag{3}$$

For a fixed value of $g(x_2) = c$, a f_1-f_2 plot becomes a hyperbola ($f_1 f_2 = c$). Figure 2 shows three hyperbolic lines with different values of c such that $c_1 < c_2 < c_3$. There exists a number of interesting properties of the above two-objective problem:

[4]With this function, it is necessary to have f_1 and g function values to be strictly positive.

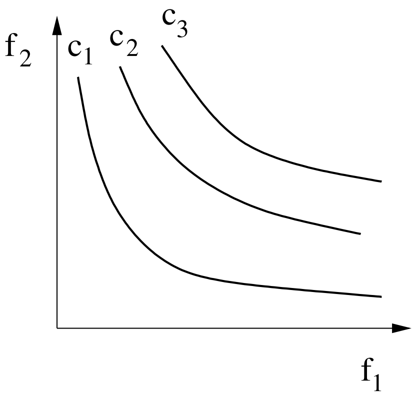

Figure 2: Three hyperbolic lines ($f_1 f_2 = c$) with $c_1 < c_2 < c_3$ are shown.

LEMMA 1 If for any two solutions, the second variable x_2 (or more specifically $g(x_2)$) are the same, both solutions are not dominated by each other.

Proof: Consider two solutions $x^{(1)} = (x_1^{(1)}, x_2^{(1)})$ and $x^{(2)} = (x_1^{(2)}, x_2^{(2)})$. Since x_2 values are same for both solutions (which also means corresponding $g(x_2)$ values are the same), the functions are related as $f_1 = x_1$ and $f_2 = c/x_1$, where $c = g(x_2^{(1)})$. Thus, if $x_1^{(1)} < x_1^{(2)}$ then $f_1(x^{(1)}) < f_1(x^{(2)})$ and $f_2(x^{(1)}) > f_2(x^{(2)})$. That is, the solution $x^{(1)}$ better than solution $x^{(2)}$ in function f_1, but worse in function f_2. Similarly, if $x_1^{(1)} > x_1^{(2)}$, conflicting behavior can be proved. However, when $x_1^{(1)} = x_1^{(2)}$, both the function values are same. Hence, by definition of domination, these two solutions are not dominated by each other. $\square$

LEMMA 2 If for any two solutions, the first variable x_1 are the same, the solution corresponding to the minimum $g(x_2)$ value dominates the other solution.

Proof: Since $x_1^{(1)} = x_1^{(2)}$, the first objective function values are also same. So, the solution having smaller $g(x_2)$ value (meaning better f_2 value) dominates the other solution. $\square$

LEMMA 3 For any two arbitrary solutions $x^{(1)}$ and $x^{(2)}$, where $x_i^{(1)} \neq x_i^{(2)}$ for $i = 1, 2$, and $g(x_2^{(1)}) < g(x_2^{(2)})$, there exists a solution $x^{(3)} = (x_1^{(2)}, x_2^{(1)})$ which dominates the solution $x^{(2)}$.

Proof: Since the solutions $x^{(3)}$ and $x^{(2)}$ have the same x_1 value and since $g(x^{(1)}) < g(x^{(2)})$, $x^{(3)}$ dominates $x^{(2)}$, according to Theorem 2. $\square$

COROLLARY 1 The solutions $x^{(1)}$ and $x^{(3)}$ have the same x_2 values and hence they are non-dominated to each other according to Theorem 1.

Based on the above discussions, we can present the following theorem:

THEOREM 1 The two-objective problem described in equations 1 and 2 has local or global Pareto-optimal solutions (x_1, x_2^*), where x_2^* is the locally or globally minimum solution of $g(x_2)$, respectively, and x_1 can take any value.

Proof: Since the solutions with a minimum $g(x_2)$ has the smallest possible $g(x_2)$ value (in the neighborhood sense in the case of local minimum and in the whole search space in the case of global minimum), according to Theorem 3, all such solutions dominate any other solution in the neighborhood in the case of local Pareto-optimal solutions or in the entire search space in the case of global Pareto-optimal solutions. Since these solutions are also non-dominated to each other, all these solutions are Pareto-optimal solutions, in the appropriate sense. $\square$

Although obvious, we shall present a final lemma about the relationship between a non-dominated set of solutions and Pareto-optimal solutions.

LEMMA 4 Although some members in a non-dominated set are members of the Pareto-optimal front, not all members are necessarily members of the Pareto-optimal front.

Proof: Say, there are only two distinct members in a set, of which $x^{(1)}$ is a member of Pareto-optimal front and $x^{(2)}$ is not. We shall show that both these solutions still can be non-dominated to each other. The solution $x^{(2)}$ can be chosen in such a way that $x_1^{(2)} < x_1^{(1)}$. This makes $f_1(x^{(2)}) < f_1(x^{(1)})$. Since $g(x_2^{(2)}) > g(x_2^{(1)})$, it follows that $f_2(x^{(2)}) > f_2(x^{(1)})$. Thus, $x^{(1)}$ and $x^{(2)}$ are non-dominated solutions. □

This lemma establishes a negative argument about multi-objective optimization methods which work with the concept of non-domination. Since these methods seek to find the Pareto-optimal front by finding the best non-dominated set of solutions, it is important to realize that the best non-dominated set of solutions obtained by an optimizer may not necessarily be the set of Pareto-optimal solutions. More could be true. Even if some members of the obtained non-dominated front are members of Pareto-optimal front, rest all members need not necessarily be members of the Pareto-optimal front. Nevertheless, seeking the best set of non-dominated solutions is the best method that exists in the literature and should be perused in absence of better approaches. But post-optimal testing (by locally perturbing each member of obtained non-dominated set or by other means) may be performed to establish Pareto-optimality of all members in an non-dominated set.

It is interesting to note that if both functions f_1 and f_2 are to be maximized (instead of minimized), the resulting Pareto-optimal front will correspond to the maxima of the g function. However, the construction of problems having mixed minimization and maximization is not possible with the above functional forms. A different function for f_2 function is needed in those cases. However, for the purpose of generating test problems, one particular type is adequate and we concentrate on generating problems where all objective functions are to be minimized.

The above two-objective problem and the associated lemmas and the theorem allow us to construct different types of multi-objective problems from single-objective optimization problems (defined in the function g). The optimality and complexity of function g is then directly transferred into the corresponding multi-objective problem. In the following subsections, we construct a multi-modal and a deceptive multi-objective problem.

4.1 Multi-modal multi-objective problem

According to Theorem 4, if the function $g(x_2)$ is multi-modal with local $\underline{x}_2$ and global $\bar{x}_2$ minimum solutions, the corresponding two-objective problem also has local and global Pareto-optimal solutions corresponding to solutions $(x_1, \underline{x}_2)$ and $(x_1, \bar{x}_2)$, respectively. The Pareto-optimal solutions vary in x_1 values.

We create a bimodal, two-objective optimization problem by choosing a bimodal $g(x_2)$ function:

$$g(x_2) = 2.0 - \exp\left\{-\left(\frac{x_2 - 0.2}{0.004}\right)^2\right\} - 0.8\exp\left\{-\left(\frac{x_2 - 0.6}{0.4}\right)^2\right\}. \tag{4}$$

Figure 3 shows the above function for $0 \leq x_2 \leq 1$ with $x_2 \approx 0.2$ as the global minimum and $x_2 \approx 0.6$ as the local minimum solutions. Figure 4 shows the f_1-f_2 plot with local and global Pareto-optimal solutions corresponding to the two-objective optimization problem. The local Pareto-optimal solutions occur at $x_2 \approx 0.6$ and the global Pareto-optimal solutions occur at $x_2 \approx 0.2$. The corresponding values for g function values are $g(0.6) \approx 1.2$ and $g(0.2) \approx 0.7057$, respectively. The density of the points marked on the plot shows that most solutions lead towards the local Pareto-optimal front and only a few solutions lead towards the global Pareto-optimal front[5].

[5]Although in this bimodal function, most of the search space leads to the local optimal solution, we would like to differentiate this function from a deceptive function. We have chosen this function with only two optima for clarity, but multi-modal functions usually cause difficulty to any search algorithm by introducing many false optima (often, in millions, see table 2), whereas deceptive functions cause difficulty to a search algorithm in constructing the true solution from partial solutions.

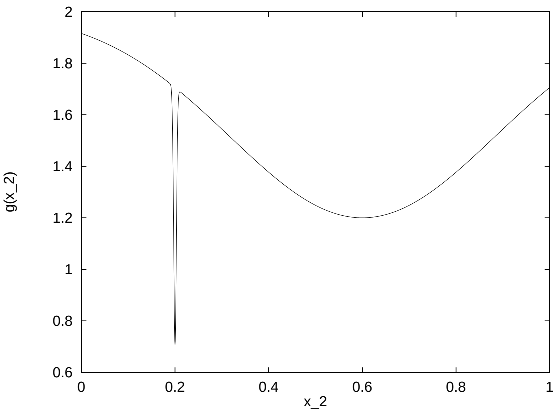

Figure 3: The function $g(x_2)$ has a global and a local minimum solution.

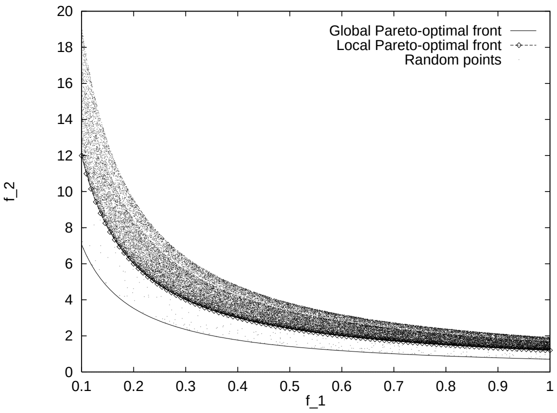

Figure 4: A random set of 50,000 solutions are shown on a f_1-f_2 plot.

To investigate how a multi-objective GA would perform in this problem, the non-dominated sorting GA (NSGA) is used. Variables are coded in 20-bit binary strings each, in the ranges $0.1 \leq x_1 \leq 1.0$ and $0 \leq x_2 \leq 1.0$. A population of size 60 is used[6]. Single-point crossover with $p_c = 1$ is chosen. No mutation is used to investigate the effect of non-dominated sorting concept alone. The niching parameter $\sigma_{\text{share}} = 0.158$ is calculated based on normalized parameter values and assuming to form about 10 niches in the Pareto-optimal front (Deb and Goldberg, 1989). Figure 5 shows a run of NSGA, which, even at generation 100, gets trapped at the local Pareto-optimal solutions (marked with a '+'). When NSGA is

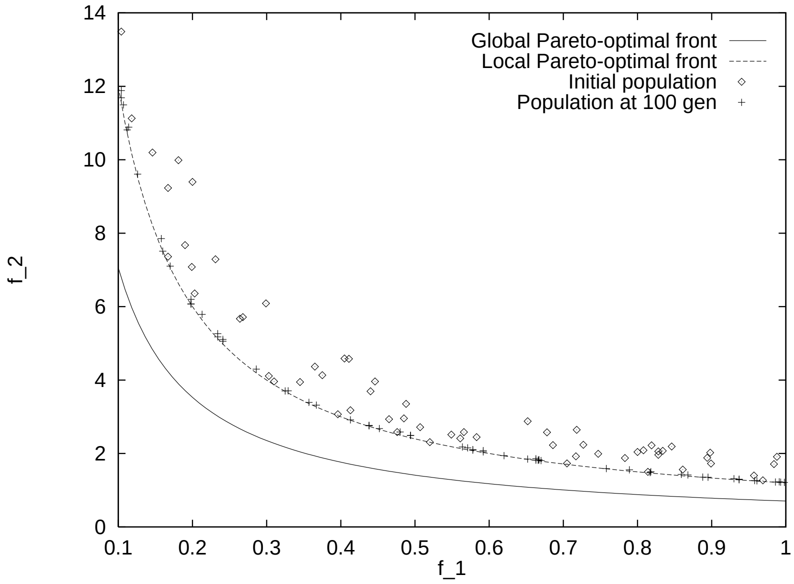

Figure 5: A NSGA run gets trapped at the local Pareto-optimal solution.

tried with 100 different initial populations, it gets trapped into the local Pareto-optimal front in 59 out of 100 runs, whereas in other 41 runs NSGA can find the global Pareto-optimal front. We also observe that in 25 runs there exist at least one solution in the global basin of function g in the initial population and still NSGAs cannot converge to the global Pareto-optimal front. Instead, they get attracted to the local

[6]This population size is determined to have, on an average, one solution in the global basin of function g in a random initial population.

Pareto-optimal front. Among 100 runs, in 21 cases the initial population has at least one solution in the global basin and they are better than the locally optimal ($g < 1.2$) solution. Out of these 21 runs, 20 runs converged to the global Pareto-optimal front, whereas one run gets misled and converge to the local Pareto-optimal front. In 5 out of 100 runs, initial population does not have any solution in the global basin and still NSGAs are able to converge to the global Pareto-optimal front. These results show that a multi-objective GA can even have difficulty from such a simple bimodal problem. However, a more difficult test problem can be constructed using a standard single-objective multi-modal test problems, such as Rastrigin's function, Schwefel's function (Gordon and Whitley, 1993), and others.

4.2 Deceptive multi-objective optimization problem

Next, we shall create a deceptive multi-objective optimization problem, from a deceptive g function. This function is defined over binary alphabets, thereby making the search space discontinuous. Let us say that the following multi-objective function is defined over ℓ bits, which is a concatenation of N substrings of variable size ℓ_i such that $\sum_{i=1}^{N} \ell_i = \ell$:

$$\begin{aligned} \text{Minimize} \quad & f_1 = 1 + u(\ell_1), \\ \text{Minimize} \quad & f_2 = \frac{\sum_{i=2}^{N} g(u(\ell_i))}{1+u(\ell_1)}, \end{aligned} \tag{5}$$

where $u(\ell_1)$ is the unitation[7] of the first substring of length ℓ_1.

The first function f_1 is a simple one-min problem, where the absolute minimum solution is to have all `0`s in the first substring. A one is added to make all function values strictly positive.

The function $g(\ell_i)$ is defined[8] in the following:

$$g(u(\ell_i)) = \begin{cases} 2 + u(\ell_i), & \text{if } u(\ell_i) < \ell_i, \\ 1, & \text{if } u(\ell_i) = \ell_i. \end{cases} \tag{6}$$

This makes the true attractor (with all `1`s in the substring) to have worst neighbors and with a function value $g(\ell_i) = 1$ and the deceptive attractor (with all `0`s in the substring) to have good neighbors and with a function value $g(0) = 2$. Since, most of the substrings lead towards the deceptive attractor, GAs may find difficulty to converge to the true attractor (all `1`s).

Once again, the global Pareto-optimal front corresponds to the solution for which the summation of g function values is absolutely minimum. Since, each minimum g function value is one, the global Pareto-optimal solutions have a summation of g equal to $(N-1)$. Since each g function has two minima (one true and another deceptive), there are a total of 2^{N-1} local minima, of which one is global. Corresponding to each of these local minima, there exist a local Pareto-optimal front (some of them are identical since the functions are defined over unitation), where a multi-objective GA may get stuck to.

In the experimental set up, we have used $\ell_1 = 10$, $\ell_2 = 5$, $\ell_3 = 5$, $\ell_4 = 5$, such that $\ell = 25$. With three deceptive subfunctions, there are a total of 2^3 or 8 Pareto-optimal fronts, of which one is global Pareto-optimal front. Since the functions are defined with unitation values, we have used genotypic niching with Hamming distance as the distance measure between two solutions (Deb and Goldberg, 1989). Since we expect 11 different function values in f_1 (all integers from 1 to 11), we use guidelines suggested in that study to calculate the σ_{share} value. For 11 niches to form, the corresponding $\sigma_{\text{share}} = 9$ is obtained for $\ell = 25$. Figure 6 shows that when a population size of 80 is used, an NSGA is able to find the global Pareto-optimal front from the initial population shown (solutions marked with a '+'). The initial population is expected to be binomially distributed over unitation (with more representative solutions for unitation values around $\ell_1/2$). Although, this introduces a bias against finding solutions for small and large

[7]Unitation is the number of `1` in the substring. Note that minimum and maximum values of unitation of a substring of length ℓ_i is zero and ℓ_i, respectively.

[8]It can be shown that an equivalent dual maximization function $G = \ell_i + 1 - g(u(\ell_i))$ is deceptive according to conditions outlined elsewhere (Deb and Goldberg, 1993). Thus, the above minimization problem is also deceptive.

values of f_1, NSGA with genotypic sharing is able to find a wide distribution of solutions in the global Pareto-optimal front. It is interesting to note that all Pareto-optimal fronts (whether global or non-global) are expected to vary in f_1, which takes integer values from 1 to 11. The figure shows that NSGA has found all but $f_1 = 11$ solution. The global Pareto-optimal solutions also correspond to all 1s in the last 15 bits (in all ℓ_2 to ℓ_4 bits). In the global Pareto-optimal solutions, the summation of g function values equal to 3.

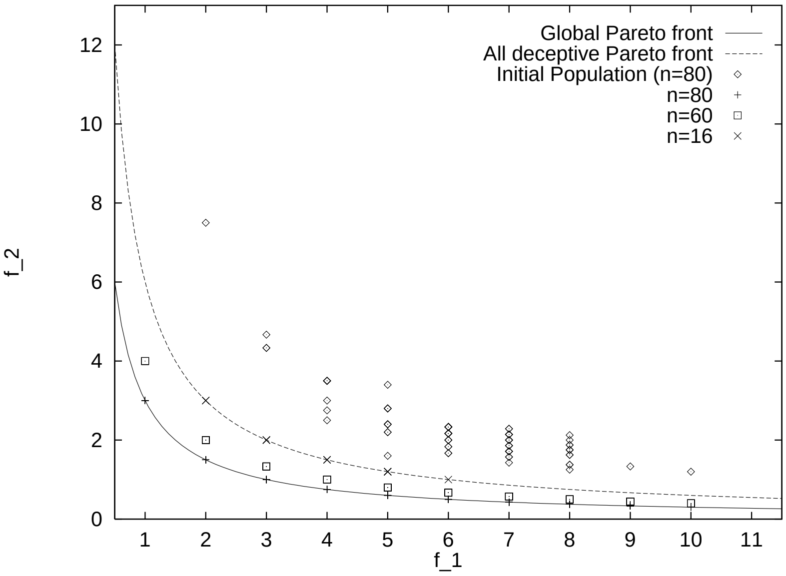

Figure 6: Performance of a single run of NSGA is shown on the deceptive multi-objective function. Fronts are shown by joining the discontinuous Pareto-optimal points with a line.

When a smaller population size ($n = 60$) is used, the NSGA cannot find the true substring in all three deceptive subproblems, instead it converges to the deceptive substring in one subproblem and to the true substring in two other subproblems. This makes a summation of g values equal to 4. When a sufficiently small population ($n = 16$) is used, the NSGA converges to the deceptive attractor in all three subproblems. In these solutions, the summation of g function values is equal to 6. The corresponding local Pareto-optimal front is shown in Figure 6 with a dashed line. Other two distinct (but 6 in total) local Pareto-optimal solutions lie in between the dashed and the solid lines.

In order to investigate further the difficulties that a deceptive multi-objective function may cause to a multi-objective GA, we construct a 30-bit function with $\ell_1 = 10$ and $\ell_i = 5$ for $i = 2, \ldots, 5$ and use $\sigma_{\rm share} = 11$. For each population size, 50 GA runs are started from different initial populations and the proportion of *successful* runs is plotted in Figure 7. A run is considered successful if all four deceptive subproblems are solved correctly (that is, the true optimal solution for the function g is found). The figure shows that NSGAs with small population sizes could not be successful in many runs. Moreover, the performance improves as the population size is increased. To show that this difficulty is due to deception in subproblems alone, we use a linear function for $g = u+1$, instead of the deceptive function used earlier, for comparison. The figure shows that multi-objective GAs with a reasonable population size have worked in much more occasions with this easy problem than with the deceptive problem.

The above two problems show that by using a simple construction methodology (by choosing a suitable g function), any problem feature that may cause single-objective GAs difficulty can also be introduced in a multi-objective GA. Based on the above construction methodology, we now present a generic two-objective optimization problem which may have additional difficulties pertaining to multi-objective optimization.

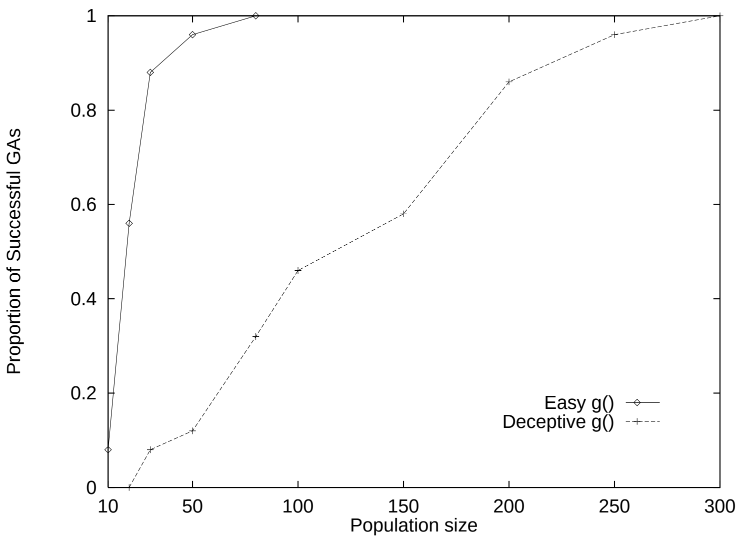

Figure 7: Proportion of successful GA runs (out of 50 runs) versus population size with easy and deceptive multi-objective problems.

5 Generic Two-objective Optimization Problems

In this section, we present a more generic two-objective optimization problem which is constructed from single-objective optimization problems. Let us consider the following N-variable two-objective problem:

$$\begin{array}{ll} \text{Minimize} & f_1(\vec{x}) = f_1(x_1, x_2, \ldots, x_m), \\ \text{Minimize} & f_2(\vec{x}) = g(x_{m+1}, \ldots, x_N) h(f_1(x_1, \ldots, x_m), g(x_{m+1}, \ldots, x_N)). \end{array} \tag{7}$$

The function f_1 is a function of m ($< N$) variables ($\vec{x}_I = (x_1, \ldots, x_m)$) and the function f_2 is a function of all N variables. The function g is a function of $(N - m)$ variables ($\vec{x}_{II} = (x_{m+1}, \ldots, x_N)$), which do not appear in the function f_1. The function h is a function of f_1 and g function values directly. We avoid complications by choosing f_1 and g functions which only take positive values (or $f_1 > 0$ and $g > 0$) in the search space.

By choosing appropriate functions for f_1, g, and h, multi-objective problems having specific features can be created. Some properties that may cause a multi-objective GA difficulty have been described earlier. Specifically, the above construction allows a controlled way to introduce such difficulties in test problems:

1. Convexity or discontinuity in the Pareto-optimal front can be affected by choosing an appropriate h function.

2. Convergence to the true Pareto-optimal front can be affected by using a difficult (multi-modal, deceptive, or others) g function, as already demonstrated in the previous section.

3. Diversity in the Pareto-optimal front can be affected by choosing an appropriate (non-linear or multi-dimensional) f_1 function.

We describe each of the above issues in the following subsections.

5.1 Convexity or discontinuity in Pareto-optimal front

By choosing an appropriate h function (which is a function of f_1 and g), multi-objective optimization problems with convex, non-convex or discontinuous Pareto-optimal fronts can be created. Specifically, if the following two properties of h are satisfied, the global Pareto-optimal set will correspond to the global minimum of the function g and to all values of the function f_1[9]:

1. The function h is a monotonically non-decreasing function in g for a fixed value of f_1.
2. The function h is a monotonically decreasing function of f_1 for a fixed value of g.

The first condition ensures that the global Pareto-optimal front occurs for the global minimum value for g function. The second condition ensures that there is a continuous 'conflicting' Pareto-front. However, we realize that when we violate this condition (the second condition), we shall no more create problems having continuous Pareto-optimal front. Later, we shall use an h function which is oscillatory with respect to f_1, in order to construct a problem having a discontinuous Pareto-optimal front. However, if the first condition is met alone, for every local minimum of g, there will exist one local Pareto-optimal set (corresponding value of g and all possible values of f_1) of the multi-objective optimization problem.

Although many different functions may exist, we present two such functions—one leading to a convex Pareto-optimal front and the other leading to a more generic problem having a control parameter which decides the convexity or the non-convexity of the Pareto-optimal fronts.

5.1.1 Convex Pareto-optimal front

We choose the following function for h:

$$h(f_1, g) = \frac{1}{f_1}. \tag{8}$$

With this function, we only allow $f_1 > 0$. The resulting Pareto-optimal set is $(\vec{x}_I^*, \vec{x}_{II}) = \{(\vec{x}_I, \vec{x}_{II}) : \nabla g(\vec{x}_I) = 0\}$. We have used this example before (see Section 4) and have seen that the resulting Pareto-optimal set is convex. This is one of the simple functions that can be chosen to achieve a convex Pareto-optimal set. In the following, we present another function which can be used to create convex and non-convex Pareto-optimal set by just tuning a parameter.

5.1.2 Non-convex Pareto-optimal front

We choose the following function for h:

$$h(f_1, g) = \begin{cases} 1 - \left(\frac{f_1}{\beta g}\right)^{\alpha}, & \text{if } f_1 \leq \beta g, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

With this function, we may allow $f_1 \geq 0$, but $g > 0$. The global Pareto-optimal set corresponds to the global minimum of g function. The parameter β is a normalization factor to adjust the range of values of functions f_1 and g. To have a significant Pareto-optimal region, β may be chosen as $\beta \geq f_{1,\max}/g_{\min}$, where $f_{1,\max}$ and $g_{\min}$ are the maximum value of the function f_1 and the minimum (or global optimal) value of the function g, respectively. It is interesting to note that when $\alpha > 1$, the resulting Pareto-optimal front is non-convex. We show one such problem a little later. It is important to note that when $\alpha > 1$ is used, the classical weighted-sum method cannot find any intermediate Pareto-optimal solution by using any weight vector. Although there exist other methods (such as ϵ-perturbation method or goal programming method (Steuer, 1986)), they require problem knowledge and, moreover, require multiple application of the single-objective optimizer.

The above function can also be used to create multi-objective problems having convex Pareto-optimal set by setting $\alpha \leq 1$. Other interesting functions for the function h may also be chosen with properties mentioned in Section 5.1.

[9]Although, for other h functions, the condition for Pareto-optimality of multi-objective problems can also be established, here, we state the sufficient conditions for the functional relationships of h with g and f_1. Note that this allows us to directly relate the optimality of g function with the Pareto-optimality of the resulting multi-objective problem.

Test problems having local and global Pareto-optimal fronts being of mixed type (some are of convex and some are of non-convex shape) can also be created by making the parameter α a function of g. Since the value of the function g decides the location of local and global Pareto-optimal solutions, problems with mixed type of fronts can be easily created. These problems would be more difficult to solve, simply because the search algorithm needs to adopt to a different kind of front while moving from local to global Pareto-optimal front. Multi-objective optimization algorithms that work by exploiting the shape of the Pareto-optimal front will have difficulty in solving such problems. Here, we illustrate one such problem, where the local Pareto-optimal front is non-convex, whereas the global Pareto-optimal front is convex. Consider the following functions ($x_1, x_2 \in [0, 1]$):

$$
\begin{aligned}
g(x_2) &= \begin{cases} 4 - 3\exp\left(-\frac{x_2-0.2}{0.02}\right)^2, & \text{if } 0 \leq x_2 \leq 0.4, \\ 4 - 2\exp\left(-\frac{x_2-0.7}{0.2}\right)^2, & \text{if } 0.4 < x_2 \leq 1, \end{cases} && (10) \\
f_1(x_1) &= 4x_1, && (11) \\
\alpha &= 0.25 + 3.75\frac{g(x_2) - g^{**}}{g^* - g^{**}}, && (12)
\end{aligned}
$$

where g^* and g^{**} are the weakest locally optimal and the globally optimal function value of g, respectively. Equation 12 is set to have non-convex weakest local Pareto-optimal front at $\alpha = 4.0$ and convex global Pareto-optimal front at $\alpha = 0.25$.

The function h is given in equation 9 with $\beta = 1$. The function $g(x_2)$ has a local minimum at $x_2 = 0.7$ and a global minimum at $x_2 = 0.2$. The corresponding function values are $g = 2.0$ and 1.0, respectively. Equation 12 suggests that for $g \leq 1.2$, the f_1-f_2 plot for constant x_2 is convex. Since the local minimum for $g(x_2)$ occurs at $g = 2.0$ (which is larger than 1.2), the local Pareto-optimal front is non-convex ($\alpha = 4.0$), as shown in Figure 8. But, at the global minimum, $g = 1.0$ and the corresponding global Pareto-optimal front is convex ($\alpha = 0.25$). A random set of 40,000 solutions ($x_1, x_2 \in [0.0, 1.0]$) are generated and

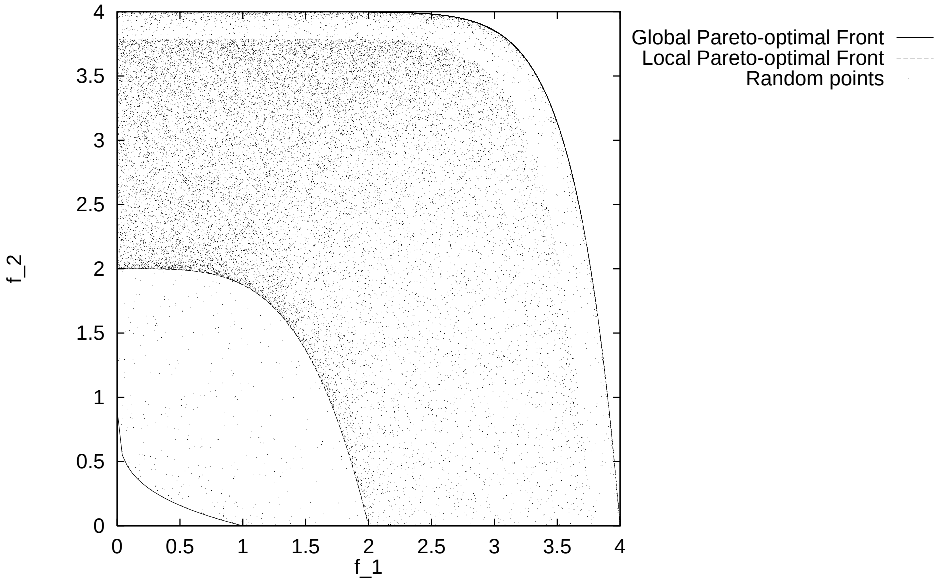

Figure 8: A two-objective function with non-convex local Pareto-optimal front and a convex global Pareto-optimal front. 40,000 random solutions are shown.

the corresponding solutions in the f_1-f_2 space are also shown. The figures clearly show the nature of the global and local Pareto-optimal fronts. Notice that only a small portion of the search space leads to the global Pareto-optimal front. An apparent front at the top of the figure is due to the discontinuity in the $g(x_2)$ function at $x_2 = 0.4$.

Another simple way to create a non-convex Pareto-optimal front is to use equation 8, but maximize both functions f_1 and f_2. The Pareto-optimal front corresponds to the maximum value of g function and the resulting Pareto-optimal front is non-convex in the sense of the corresponding multi-objective optimization problem. The restrictions on f_1 (> 0) and g (> 0) functions apply as before.

5.1.3 Discontinuous Pareto-optimal front

As mentioned earlier, we have to relax the condition for h being a monotonically decreasing function of f_1 to construct multi-objective problems with a discontinuous Pareto-optimal front. In the following, we show one such construction where the function h is a periodic function of f_1:

$$h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{\alpha} - \frac{f_1}{g}\sin(2\pi q f_1). \tag{13}$$

The parameter q is the number of discontinuous regions in an unit interval of f_1. By choosing the following functions

$$\begin{aligned} f_1(x_1) &= x_1, \\ g(x_2) &= 1 + 10x_2, \end{aligned}$$

and allowing variables x_1 and x_2 to lie in the interval [0,1], we have a two-objective optimization problem which has a discontinuous Pareto-optimal front. Since the h (and hence f_2) function is periodic to x_1 (and hence to f_1), we generate discontinuous Pareto-optimal regions.

Figure 9 shows the 50,000 random solutions in f_1-f_2 space. Here, we use $q = 4$ and $\alpha = 2$. When NSGAs (population size of 200, σ_{share} of 0.1, crossover probability of 1, and no mutation) are applied to this problem, the resulting population at generation 300 is shown in Figure 10. The plot shows that if reasonable GA parameter values are chosen, NSGAs can find solutions in all four discontinuous Pareto-optimal regions. A population size of 200 is used to have a wide distribution of solutions in all discontinuous regions. Since a linear function for g is used, the NSGA soon makes most of its population members converged to the optimum solution for $x_2^* = 0$. When this happens, the entire population is almost classified into one non-domination class and niching helps to maintain diversity among Pareto-optimal solution. However, it is interesting to note how NSGAs avoid creating the non-Pareto-optimal solutions, although the corresponding x_2 value may be zero. In general, discontinuity in the Pareto-optimal front may cause difficulty to multi-objective GAs which do not have an efficient mechanism of implementing diversity among discontinuous regions.

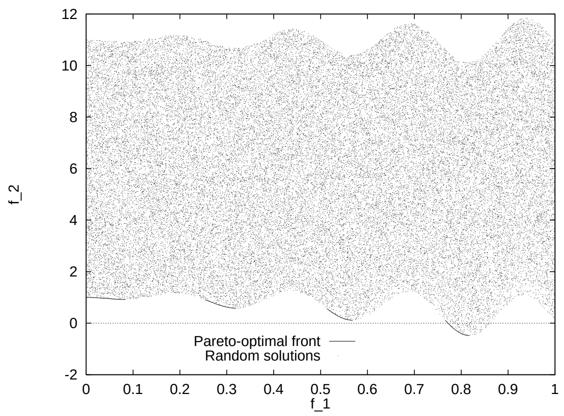

Figure 9: 50,000 random solutions are shown on a f_1-f_2 plot of a multi-objective problem having discrete Pareto-optimal front.

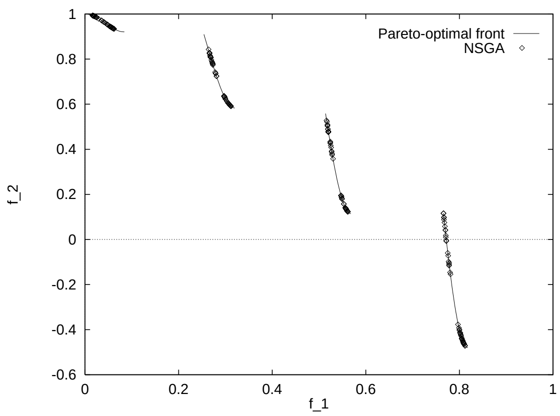

Figure 10: The population at generation 300 for a NSGA run is shown to have found solutions in all four discontinuous Pareto-optimal regions.

5.2 Hindrance to reach true Pareto-optimal front

It is shown earlier that by choosing a difficult function for g alone, a difficult multi-objective optimization problem can be created. Specifically, some instances of multi-modal and deceptive multi-objective optimization have been created earlier. Test problems with standard multi-modal functions used in single-objective GA studies, such as Rastrigin's functions, NK landscapes, and others can all be chosen for the g function.

5.2.1 Biased search space

The function g makes a major role in introducing difficulty to a multi-objective problem. Even though the function g is not chosen to be a multi-modal function nor to be a deceptive function, with a simple monotonic g function the search space can have adverse density of solutions towards the Pareto-optimal region. Consider the following function for g:

$$g(x_{m+1}, \ldots, x_N) = g_{\min} + (g_{\max} - g_{\min}) \left(\frac{\sum_{i=m+1}^{N} x_i - \sum_{i=m+1}^{N} x_i^{\min}}{\sum_{i=m+1}^{N} x_i^{\max} - \sum_{i=m+1}^{N} x_i^{\min}} \right)^{\gamma}, \tag{14}$$

where $g_{\min}$ and $g_{\max}$ are minimum and maximum function values that the function g can take. The values $x_i^{\min}$ and $x_i^{\max}$ are minimum and maximum values of the variable x_i. It is important to note that the Pareto-optimal region occurs when g takes the value $g_{\min}$. The parameter γ controls the biasness in the search space. If $\gamma < 1$, the density of solutions away from the Pareto-optimal front is more. We show this on a simple problem with $m = 1$, $N = 2$, and with following functions:

$$\begin{aligned} f_1(x_1) &= x_1, \\ h(f_1, g) &= 1 - \left(\frac{f_1}{g}\right)^2. \end{aligned}$$

We also use $g_{\min} = 1$ and $g_{\max} = 2$. Figures 11 and 12 show 50,000 random solutions each with γ equal to 1.0 and 0.25, respectively. It is clear that for $\gamma = 0.25$, not even one solution is found in the Pareto-optimal

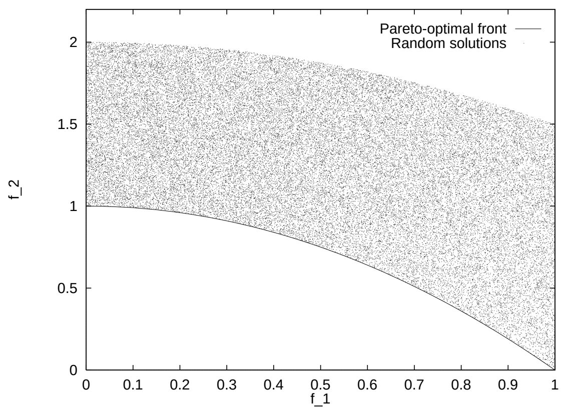

Figure 11: 50,000 random solutions are shown for $\gamma = 1.0$.

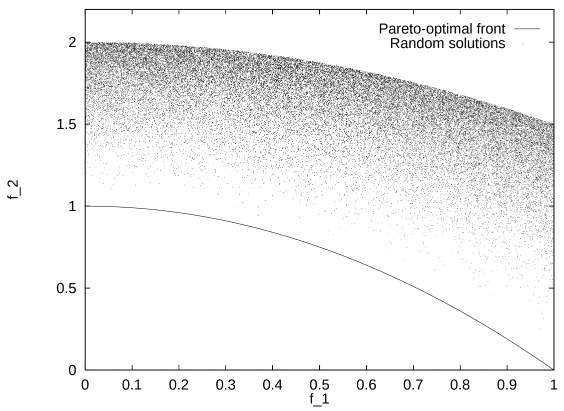

Figure 12: 50,000 random solutions are shown for $\gamma = 0.25$.

front, whereas for $\gamma = 1.0$, many Pareto-optimal solutions exist in the set of 50,000 random solutions. Random search methods are likely to face difficulty in finding the Pareto-optimal front in the case with γ close to zero, mainly due to the low density of solutions towards the Pareto-optimal region. Although multi-objective GAs, in general, will progress towards the Pareto-optimal front, a different scenario may emerge. Although for values of γ greater than one, the search space is biased towards the Pareto-optimal region, the search in a multi-objective GA with proportionate selection and without mutation or without elitism is likely to slow down near the Pareto-optimal front. In such cases, the multi-objective GAs may prematurely converge to a front near the true Pareto-optimal front. This is because the rate of improvement in g value near the optimum ($x_2 \approx 0$) is small with $\gamma \ll 1$. Nevertheless, a simple change in the function g with a change in γ suggested above will change the landscape drastically and multi-objective optimization algorithms may face difficulty in converging to the true Pareto-optimal front.

5.2.2 Parameter interactions

The difficulty in converging to the true Pareto-optimal front may also arise because of parameter dependence to each other. It is discussed before that the Pareto-optimal set in the two-objective optimization problem described in equation 7 corresponds to all solutions of different f_1 values. Since the purpose in a multi-objective GA is to find as many Pareto-optimal solutions as possible and since in equation 7 the variables defining f_1 are different from variables defining g, a GA may work in two stages. In one stage, all variables $\vec{x}_I$ may be found and in the other stage optimal $\vec{x}_{II}$ may be found. This rather simple mode of working of a GA in two stages can face difficulty if the above variables are mapped to another set of variables. If M is a random orthonormal matrix of size $N \times N$, the true variables $\vec{y}$ can first be mapped to derived variables $\vec{x}$ using

$$\vec{x} = M\vec{y}. \tag{15}$$

Thereafter, objective functions defined in equation 7 can be computed using the variable vector $\vec{x}$. Since the components of $\vec{x}$ can now be negative, care must be taken in defining f_1 and g functions so as to satisfy restrictions suggested on them in previous subsections. A translation of these functions by adding a suitable large positive value may have to be used to force these functions to take non-negative values. Since, the GA will be operating on the variable vector $\vec{y}$ and all variables $\vec{y}$ are now related to each other at the Pareto-optimal front, any change in one variable must be accompanying by related changes in other variables in order to remain on the Pareto-optimal front. This makes this mapped version of problem more difficult to solve than the unmapped version. We discuss more about mapped functions in the following section.

5.3 Non-uniformly represented Pareto-optimal front

In all the above test functions constructed above (except the deceptive problem), we have used a linear, single-variable function for f_1. This helped us to create a problem with uniform distribution of solutions in f_1. Unless the underlying problem has discretely spaced Pareto-optimal regions (like in Section 5.1.3), there is no bias for the Pareto-optimal solutions to get spread over the entire range of f_1 values. However, a bias for some portions of range of values for f_1 may also be created by choosing any of the following f_1 functions:

1. The function f_1 is non-linear, and

2. The function f_1 is a function of more than one variable.

It is clear that if a non-linear f_1 function (whether single or multi-variable) is chosen, the resulting Pareto-optimal region (or, for that matter, the entire search region) will have bias towards some values of f_1. The non-uniformity in distribution of the Pareto-optimal region can also be created by simply choosing a multi-variable function (whether linear or non-linear). Consider, for simplicity, a two-variable function for f_1 ($m = 2$):

$$f_1(x_1, x_2) = x_1 + x_2.$$

If each variable is varied between $[0, 1]$, the maximum number of solutions ((x_1, x_2)) have the function value $f_1 = 1$. The number solutions having any other function value $f_1 \in [0, 2]$ reduces, as f_1 reduces or increases from $f_1 = 1$, thereby causing an artificial bias for solutions to cluster around $f_1 = 1$ values. Multi-objective optimization algorithms, which are not good at maintaining diversity among solutions (or function values), will produce a biased Pareto-optimal front in such problems. Thus, the non-linearity in function f_1 or dimension of f_1 measures how well an algorithm is able to maintain distributed non-dominated solutions in a population.

Consider the single-variable, multi-modal function f_1:

$$f_1(x_1) = 1 - \exp(-4x_1)\sin^4(5\pi x_1), \quad 0 \le x_1 \le 1. \tag{16}$$

The above function has five minima for different values of x_1, as shown in Figure 13. The figure also shows the corresponding non-convex Pareto-optimal front in a f_1-f_2 plot with h function defined in equation 9 having $\beta = 1$ and $\alpha = 4$ (since $\alpha > 1$, the Pareto-optimal front is non-convex) . The right figure is

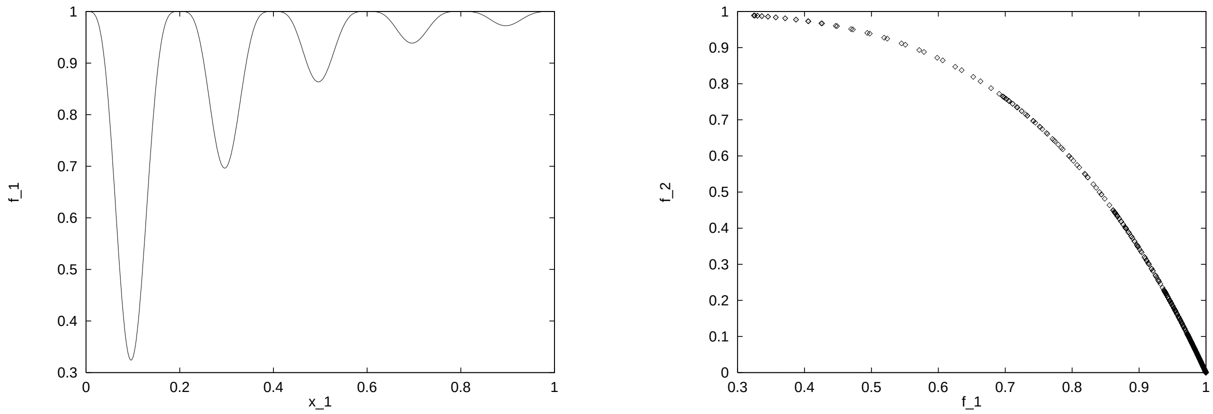

Figure 13: A multi-modal f_1 function and corresponding non-uniformly distributed non-convex Pareto-optimal region are shown. In the right plot, Pareto-optimal solutions derived from 500 uniformly-spaced x_1 points are shown.

generated from 500 uniformly-spaced points in x_1. The value of x_2 is fixed so that the minimum value of $g^*(x_2)$ is equal to 1. The figure shows that the Pareto-optimal region is biased for solutions for which f_1 is near one.

The working of a multi-objective GA on this function provides insights into an interesting debate about fitness-space niching (Fonseca and Fleming, 1995) versus parameter-space niching (Srinivas and Deb, 1994). It is clear that when function-space niching is performed, a uniform distribution in the function space (right plot in Figure 13) is anticipated. There are at least two difficulties with this approach. First, the obtained distribution would depend on the shape of the Pareto-optimal region. Since GAs operate on the solution vector, instead of the function values directly, in such complex problems it is difficult to realize what function-space niching means to the solutions. Secondly, notice that the function f_1 has five distinct minima in x_1 with increasing function value. Since the objective of niching is to maintain diversity among the Pareto-optimal solutions, the fitness-space niching may not maintain diversity in solution vectors, instead may maintain diversity among the objective vectors.

We compare the performance of NSGAs with two different niching—parameter-space niching and function-space niching. NSGAs with a reasonable parameter settings (population size of 100, 15-bit coding for each variable, σ_{share} of 0.2236 (assuming 5 niches), crossover probability of 1, and no mutation) are run for 500 generations. A typical run for both niching methods are shown in Figure 14. Identical σ_{share} value is used in both cases. This is because in both cases the ranges of values of x_i or f_i are the same. Although it seems that both niching methods are able to maintain diversity in function space (with a better distribution in f_1-f_2 space with function-space niching), the left plot in Figure 15 shows that the NSGA with parameter-space niching has truly found diverse solutions, whereas the NSGA with function-space niching (right plot) converges to about 50% of the entire region of the Pareto-optimal solutions. Since the

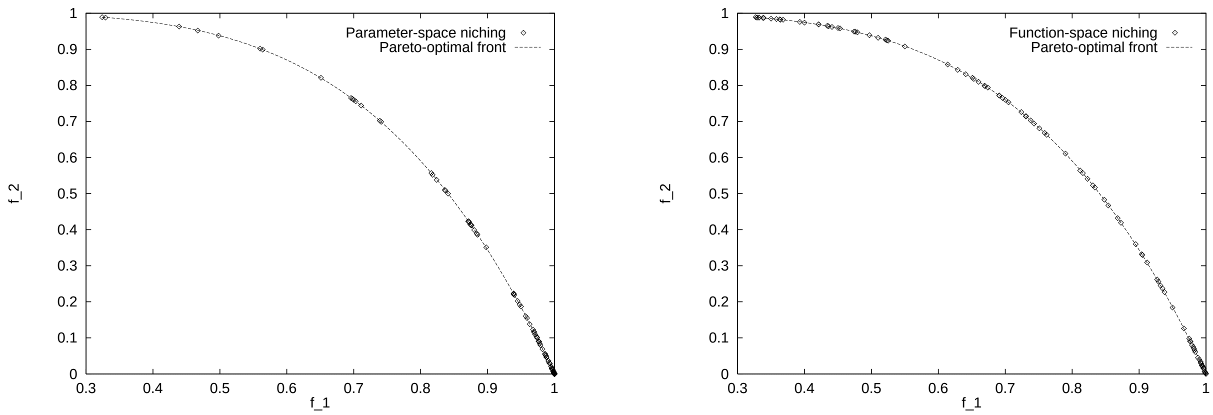

Figure 14: The left plot is with parameter-space niching and right is with function-space niching. The figures show that both methods find solutions with diversity in the f_1-f_2 space. But do each plot suggest adequate diversity in the solution space? Refer to next figure for an answer.

first minimum and its basin of attraction spans the complete space for the function f_1, the function-space niching does not have the motivation to find other important solutions (which are in some sense in the shadow of the first minimum). Thus, in problems like this, function-space niching may hide information about important Pareto-optimal solutions in the search space.

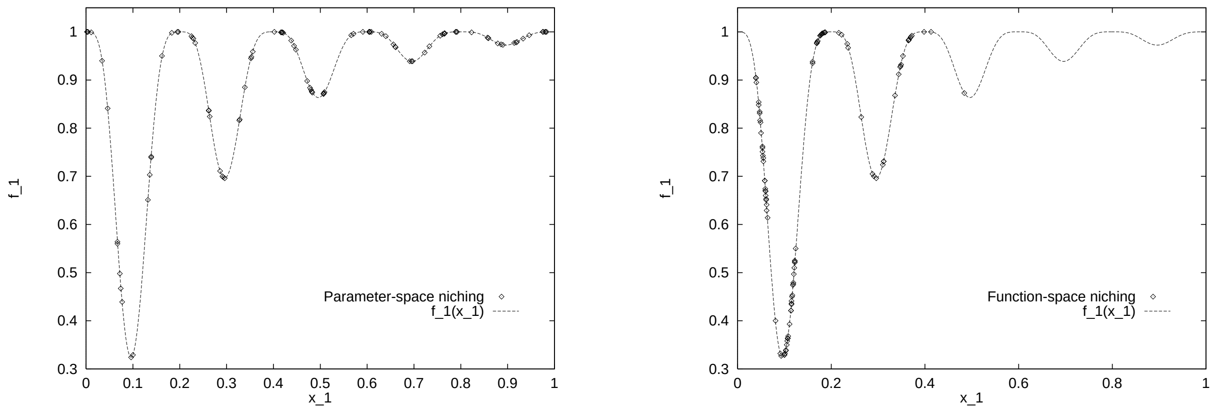

Figure 15: The left plot is with parameter-space niching and right is with function-space niching. Clearly, parameter-space niching is able to find more diverse solutions than function-space niching. All 100 solutions at generation 500 are shown in each case.

It is important to understand that the choice between parameter-space or function-space niching entirely depends on what is desired in a set of Pareto-optimal solutions in the underlying problem. In some problems, it may be important to have solutions with trade-off in objective function values, without much care of how similar or diverse the actual solutions (x vectors or strings) are. In such cases, function-space niching will, in general, provide solutions with more trade-off in objective function values. Since there is no induced pressure for the actual solutions to differ from each other, the Pareto-optimal solutions may not be very different, unless the underlying objective functions demand them to be so. On the other hand, in some problems the emphasis could be on finding more diverse solutions and with clear trade-off among objective functions. In such cases, parameter-space niching would be better. This is because, in some sense, categorizing a population using non-domination and emphasizing all such non-dominated solutions through the selection operator help to maintain some diversity among objective function values. Whereas, if explicit niching in the parameter space is not used, it is not expected to create Pareto-optimal solutions with diversity in parameter values. However, a multi-objective optimization algorithm which explicitly uses both niching (either in each generation or temporally switching from one type of niching to another with generation (L. Thiele and E. Zitzler, personal communication, October, 1998)) would ensure solutions with both kinds of diversity.

We return to the original problem and investigate how interactions among variables effect the performance of NSGAs having both types of niching. When the parameter interaction is introduced by mapping variables to another set of variables (by multiplying the original variable vector with a random normalized matrix (suggested in equation 15) and translated to make the function values non-negative), the distinction between parameter-space and function-space niching is even more clear (Figure 16). GA parameter values same as that in the unmapped case above are used here. Now the f_1-x_1 plot is rotated and the Pareto-optimal front now occurs, not simply for a fixed value of just one variable x_2, but for a fixed value of weighted sum of x_1 and x_2, dictated by the chosen random matrix. This makes the task of finding the Pareto-optimal front harder, as discussed in Section 5.2. The left plot shows that parameter-space niching

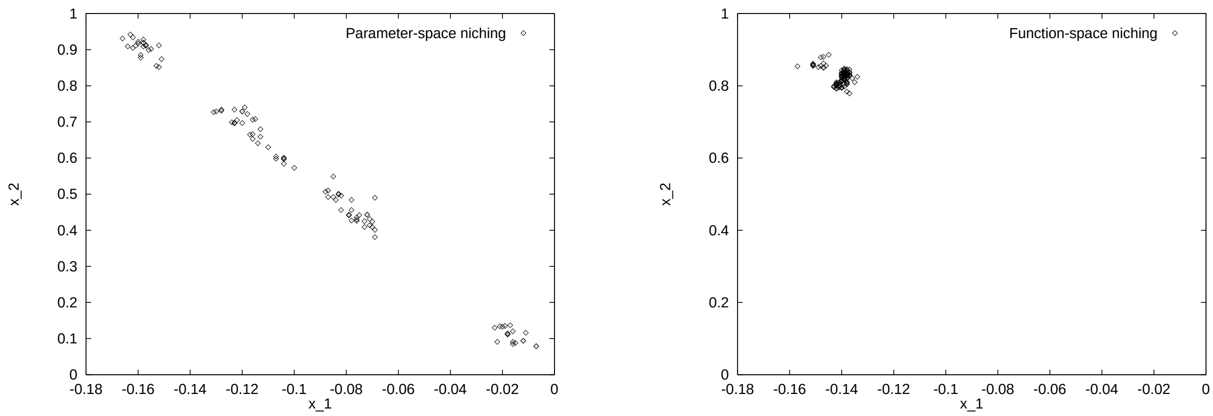

Figure 16: Plots for a rotated function are shown. The left plot is with parameter-space niching and right is with function-space niching. Clearly, parameter-space niching is able to find more diverse solutions than function-space niching. The plots are made with all 100 solutions at generation 500.

is able to find solutions across the entire range, whereas the right plot shows that function-space niching is able to find solutions in one minimum (the first minimum). However, an usual f_1-f_2 plot reveals that the function-space niching is also able to find diverse solutions. But a plot like in Figure 16 truly reveals the diversity achieved in the solutions.

A more complicated search space can be created by introducing bias in both lateral to (on f_2) and along (on f_1) the Pareto-optimal region by using techniques presented in Sections 5.2.1 and 5.3. Using non-linear functions for g and f_1, such bias can be easily created in multi-objective optimization test problems.

6 Summary of Test Problems

The two-objective optimization problem discussed above requires three functional—f_1, g, and h—which can be set to various complexity levels to create complex two-objective optimization test problems. By varying the complexity of one function and by fixing other two functions at their simplest form, we can create multi-objective test problems with known and desired features. However, more complicated test problems can also be created by simultaneously varying the complexity of more than one functions at a time.

In the following, we summarize the properties of a two-objective optimization problem due to each of above functions:

1. The function f_1 tests a multi-objective GA's ability to find diverse Pareto-optimal solutions. The function f_1 can used to create multi-objective problem having nonuniform representation of solu-

tions in the Pareto-optimal region. Thus, this function tests an algorithm's ability to handle difficulties *along* the Pareto-optimal front.

2. The function g tests a multi-objective GA's ability to converge to the true (or global) Pareto-optimal front. The function g can be used to create multi-modal, deceptive, isolated, or other complex multi-objective optimization problems. Thus, this function tests an algorithm's ability to handle difficulties *lateral* to the Pareto-optimal front.

3. The function h tests a multi-objective GA's ability to tackle multi-objective problems having convex, non-convex, or discontinuous Pareto-optimal fronts. The function h can be used to create problems with convex, non-convex, and discontinuous multi-objective optimization problems. Thus, this function tests an algorithm's ability to handle different types of the Pareto-optimal front.

In the light of the above discussion, we summarize and suggest in Tables 1 to 3 a few test functions for the above three functionals, which may be used in combination to each other. Unless specified, all variables x_i mentioned in the tables take real values in the range [0,1].

Table 1: Effect of function f_1 on the test problem.

Function $f_1(x_1, \ldots, x_m) \quad (> 0)$ Controls search space along the Pareto-optimal front			
	Type	Example	Effect
F1-I	Single-variable ($m = 1$) and linear	$\delta_f + c_1 x_1 \quad (\delta_f, c_1 > 0)$	Uniform representation of solutions in the Pareto-optimal front. Most of the Pareto-optimal region is likely to be found.
F1-II	Multi-variable ($m > 1$) and linear	$\delta_f + \sum_{i=1}^{m} c_i x_i \quad (\delta_f, c_i > 0)$	Non-uniform representation of Pareto-optimal front. Some Pareto-optimal regions are not likely to be found.
F1-III	Non-linear (any m)	Eqn 16 for $m = 1$ or, $1 - \exp(-4r)\sin^4(5\pi r)$ where $r = \sqrt{\sum_{i=1}^{m} x_i^2}$	Same as above.
F1-IV	Multi-modal	Eqn 4 with $g(x_2)$ replaced by $f_1(x_1)$ or other standard multi-modal test problems (such as Rastrigin's function, see Table 2)	Same as above. Solutions at global optimum of f_1 and corresponding function values are difficult to find.
F1-V	Deceptive	$f_1 = \sum_{i=1}^{m} f(\ell_i)$, where f is same as g defined in Eqn 6	Same as above. Solutions at true optimum of f_1 are difficult to find.

The functions mentioned in the third column in each table are representative functions which will produce the desired effect mentioned in the respective fourth column. However, other functions of similar type (mentioned in the second column) can also be chosen in each case. While testing an algorithm for its ability to overcome a particular feature of a test problem, we suggest varying the complexity of the corresponding function (f_1, g, or h) and fixing the other two functions at its easiest complexity level. For example, while testing an algorithm for its ability to find the global Pareto-optimal front in a multi-modal multi-objective problem, we suggest choosing a multi-modal g function (G-III) and fixing f_1 as in F1-I

Table 2: Effect of function g on the test problem.

Function $g(x_{m+1}, \ldots, x_N)$ (> 0), say $n = N - m$ Controls search space lateral to the Pareto-optimal front			
	Type	Example	Effect
G-I	Uni-modal, single-variable ($n = 1$), and linear	$\delta_g + c_2 x_2$ ($\delta_g, c_2 > 0$), or Eqn 14 with $\gamma = 1$	No bias for any region in the search space.
G-II	Uni-modal and non-linear	Eqn 14 with $\gamma \neq 1$	With $\gamma > 1$, bias towards the Pareto-optimal region and with $\gamma < 1$, bias against the Pareto-optimal region.
G-III	Multi-modal	Rastrigin: $1 + 10n + \sum_{i=m+1}^{N} x_i^2 - 10\cos(2\pi x_i)$ $x_i \in [-30, 30]$	Many $(61^n - 1)$ local and one global Pareto-optimal fronts
		Schwefel: $1 + (6.5\pi)^2 n - \sum_{i=m+1}^{N} x_i \sin(\sqrt{\lvert x_i \rvert})$ $x_i \in [-512, 511]$	Many $(8^n - 1)$ local and one global Pareto-optimal fronts
		Griewangk: $2 + \sum_{i=m+1}^{N} x_i^2/4000 - \Pi_{i=m+1}^{N} \cos(x_i/\sqrt{i})$ $x_i \in [-512, 511]$	Many $(163^n - 1)$ local and one global Pareto-optimal fronts
G-IV	Deceptive	Eqn 6	Many $(2^n - 1)$ deceptive attractors and one global attractor
G-V	Multi-modal, deceptive	$g(u(\ell_i)) = \begin{cases} 2 + e, & \text{if } e < \ell_i/2, \\ 1, & \text{if } e = \ell_i/2. \end{cases}$ where $e = \lvert u(\ell_i) - \ell_i/2 \rvert$	Many $(\Pi_{i=m+1}^{N} \left[\binom{\ell_i}{\ell_i/2} + 2\right] - 2^n)$ deceptive attractors and 2^n global attractors

Table 3: Effect of function h on the test problem.

Function $h(f_1, g)$ (> 0) Controls shape of the Pareto-optimal front			
	Type	Example	Effect
H-I	Monotonically non-decreasing in g and convex on f_1	Eqn 8 or Eqn 9 with $\alpha \leq 1$	Convex Pareto-optimal front
H-II	Monotonically non-decreasing in g and non-convex on f_1	Eqn 9 with $\alpha > 1$	Non-convex Pareto-optimal front
H-III	Convexity in f_1 as a function of g	Eqn 9 along with Eqn 12	Mixed convex and non-convex shapes for local and global Pareto-optimal fronts
H-IV	Non-monotonic periodic in f_1	Eqn 13	Discontinuous Pareto-optimal front

and h as in H-I. Similarly, using g function as G-I, h function as H-I, and by first choosing f_1 function as F1-I test a multi-objective optimizer's capability to distribute solutions along the Pareto-optimal front. By only changing the f_1 function to F1-III (even with $m = 1$), the same optimizer can be tested for its ability to find distributed solutions in the Pareto-optimal front. This is because with a nonlinear function for f_1 function, there is a bias against finding some subregions in the Pareto-optimal front. It will then be a test for a multi-objective optimizer's ability to find those adverse regions in the Pareto-optimal front.

Some interesting combinations of these three functions will also produce significantly difficult test problems. For example, if a deceptive f_1 (F1-V) and a deceptive g function (G-IV) are used (E. Zitzler, personal communication, October, 1998), it is likely that a multi-objective GA with a small population size will get attracted to deceptive attractors of both functions. In such a case, that GA will not find the global Pareto-optimal front. On the other hand, since not all function values of f_1 are likely to be found, some region in the Pareto-optimal front will be undiscovered. The G-V function for g has a massively multi-modal landscape along with deception (Goldberg, Deb, and Horn, 1992). This function introduces a number of different solutions having the same global optimal g function value. Corresponding to each of these globally optimal solutions for g function, there is one global Pareto-optimal front. In fact, in f_1-f_2 space, all global Pareto-optimal fronts are the same, but solutions differ drastically. The sheer number of local Pareto-optimal fronts and deception in such a problem should cause enough difficulty to any multi-objective GA to converge to one of the global Pareto-optimal fronts. An interesting challenge in this problem would be to find all (or as many as possible) different globally optimal solutions for the g function.

Along with any such combination of three functionals, parameter interactions can be introduced to create even more difficult problems. Using a transformation of the coordinate system, as suggested in section 5.2.2, all the above-mentioned properties can be tested in a space where simultaneous adjustment of all parameter values are desired for finding an improved solution.

7 Future Directions for Research

This study suggests a number of immediate areas of research for developing better multi-objective GAs. A list of them are outlined and discussed in the following:

1. Comparison of existing multi-objective GA implementations
2. Understand dynamics of GA populations with generations
3. Scalability of multi-objective GAs with number of objectives
4. Develop constrained test problems for multi-objective optimization
5. Convergence to Pareto-optimal front
6. Define appropriate multi-objective GA parameters (such as elitism)
7. Metrics for comparing two populations
8. Hybrid multi-objective GAs
9. Real-world applications
10. Multi-objective scheduling and other optimization problems

As mentioned earlier, there exists a number of different multi-objective GA implementations primarily varying in the way non-dominated solutions are emphasized and in the way the diversity in solutions are maintained. Although some studies have compared different GA implementations (Zitzler and Thiele, 1998), they all have done on a specific problem without much knowledge about the complexity of the test problems. With the ability to construct test functions having controlled complexity, as illustrated in this paper, an immediate task would be to compare the existing multi-objective GAs and to establish the power of each algorithm in tackling different types of multi-objective optimization problems. Such a study will not only make a comparative evaluation of the existing algorithms, the knowledge gained from the study can also be used to develop new and improved multi-objective GAs. Currently, we are undertaking such a study, the outcome of which will be reported at a later date.

The test functions suggested here provide various degrees of complexity. The construction of all these test problems has been done without much knowledge of how multi-objective GAs work. If we know more about how such GAs based on non-domination principle actually work, problems can be created to test more specific aspects of multi-objective GAs. In this regard, an interesting study would be to investigate how an initial random population of solutions move from one generation to the next. In an initial random population, it is expected to have solutions belonging to many non-domination levels. One hypothesis about the working of a multi-objective GA would be that most population members soon collapse to a single non-dominated front and each generation thereafter proceeds by improving this large non-dominated front. Let us call this mode of working as ‘level-wise’ progress. On the other hand, GAs may also thought to work by maintaining a number of non-domination levels at each generation (say, ‘multi-level’ progress). Both these modes of working should provide enough diversity for the GAs to find new and improved solutions and are thus likely candidates, although the actual mode of working may depend on the problem at hand. Nevertheless, whether a GA follows one of these two modes of working alone or in combination may also depend on the exact implementation of niching and non-domination principles. Thus, it will be worthwhile to investigate how existing multi-objective GA implementations work in the context of different test problems.

In this paper, we have not considered more than two objectives, although extensions of these test problems for more than two objectives can also be done. It is intuitive that as the number of objectives increase, the Pareto-optimal region is represented by multi-dimensional surfaces. With more objectives, multi-objective GAs must have to maintain more diverse solutions in the non-dominated front in each iteration. Whether GAs are able to find and maintain diverse solutions, as demanded by the search space of the problem with many objectives would be a matter of interesting study. Whether population size alone can solve this scalability issue or a major structural change (implementing a better niching method) is imminent would be the outcome of such a study.

We also have not considered constraints in this paper. Constraints can introduce additional complexity in the search space by inducing infeasible regions in the search space, thereby obstructing the progress of an algorithm towards the global Pareto-optimal front. Thus, creation of constrained test problems is interesting area which should get emphasis in the near future. With the development of such complex test problems, there is also a need to develop efficient constraint handling techniques that would be able to help GAs to overcome hurdles caused by constraints. Some such methods are in progress in the context single-objective GAs (Deb, in press; Koziel and Michalewicz, 1998) and with proper implementations they should also work in multi-objective GAs.

Most multi-objective GAs that exist to date work with the non-domination principle. Ironically, we have shown in Section 4 that all solutions in a non-dominated set need not be members of the true Pareto-optimal front, although some of them could be. This means that all non-dominated solutions found by a multi-objective optimization algorithm may not necessarily be Pareto-optimal solutions. Thus, while working with such algorithms, it is wise to check the Pareto-optimality of each of such solutions (by perturbing the solution locally or by using weighted-sum single-objective methods originating from these solutions). In this regard, it would be interesting to introduce special features (such as elitism, mutation, or other diversity-preserving operators), the presence of which may help us to prove convergence of a GA population to the global Pareto-optimal front. Attempts to some such proofs exist for single-objective GAs (Suzuki, 1993; Rudolph, 1998) and a similar proof may also be attempted for multi-objective GAs.

Elitism is an useful and popular mechanism used in single-objective GAs. Elitism ensures that the best solutions in each generation will not be lost. They are directly carried over from one generation to the next and what is important is that these good solutions get a chance to participate in recombination with other solutions in the hope of creating better solutions. In the context of single-objective optimization, there is only one best solution in a population. But in multi-objective optimization, all non-dominated solutions of the first level are the best solutions in the population. There is no way to distinguish one solution from the other in the non-dominated set. Then if we like to introduce elitism in multi-objective GAs, should we carry over all solutions in the first non-dominated set to the next generation! This may mean copying many good solutions from one generation to the next, a process which may lead to premature convergence to non-Pareto-optimal solutions. How elitism should be defined in this context is an interesting research topic, but one way to do this would be to carry over only those solutions from the previous non-dominated set of the first level that are not dominated by any member in the current population.

In this context, an issue related to comparison of two populations also raises some interesting questions. As mentioned earlier, there are two goals in a multi-objective optimization—convergence to the true Pareto-optimal front and maintenance of diversity among Pareto-optimal solutions. A multi-objective GA may have found a population which has many Pareto-optimal solutions, but with less diversity among them. How would such a population be compared with respect to another which has fewer number of Pareto-optimal solutions but with wide diversity? The practitioners of multi-objective GAs must have to settle for an answer for these questions before they would be able to compare different GA implementations or before they would be able to mimic operators used in single-objective GAs, such as CHC (Eshelman, 1990) or steady-state GAs (Syswerda, 1989).

As it is often suggested and used in single-objective GAs, a hybrid strategy of either implementing problem-specific knowledge in GA operators or using a two-stage optimization process of first finding good solutions with GAs and then improving these good solutions with a domain-specific algorithm would make multi-objective optimization much faster than GAs alone.

Test functions test an algorithm's capability to overcome a specific aspect that a real-world problem may have. In this respect, an algorithm which can overcome more aspects of problem difficulty is naturally a better algorithm. This is precisely the reason why so much effort is spent on doing research in test function development. As it is important to develop better algorithms by applying them on test problems with known complexity, it is also equally important that the algorithms are tested in real-world problems with unknown complexity. Fortunately, most interesting engineering design problems are naturally posed as finding trade-offs among a number of objectives. Among them, cost and reliability are two objectives which are often the priorities of designers. This is because, often in a design, a solution which is less costly is likely to be less reliable and vice versa. In handling such real-world applications using single-objective GAs, often, an artificial scenario is created. Only one objective is retained and all other objectives are used as constraints. For example, if cost is retained as an objective, then an extra constraint restricting the reliability to be greater than 0.9 (or some other value) is used. With the availability of efficient multi-objective GAs, there is no need to have such artificial constraints (which are, in some sense, user-dependent). Moreover, a single run of a multi-objective GA may provide a number of Pareto-optimal solutions, each of which is optimal in one objective with a constrained upper limit on other objectives (such as optimal in cost for a particular upper bound on reliability). Thus, the advantages of using a multi-objective GA in real-world problems are many and there is need for some interesting application case studies which would clearly show the advantages and flexibilities in using a multi-objective GA, as opposed to a single-objective GA.

With the advent of efficient multi-objective GAs for function optimization, the concept of multi-objective optimization can also be applied to other search and optimization problems, such as multi-objective scheduling and other multi-objective combinatorial optimization problems. Since in tackling these problems using permutation GAs, the main differences from binary GAs are in the way the solutions are represented and in the construction of GA operators, identical non-domination principle along with the same niching concept can still be used in solving such problems having multiple objectives. In this context, similar concepts can also be incorporated in developing other population-based multi-objective EAs, such as multi-objective evolution strategies, multi-objective genetic programming or multi-objective evolutionary programming, to better solve specific multi-objective problems which are ideally suited for the respective evolutionary method.

8 Conclusions

For the past few years, there has been a growing interest in the studies of multi-objective optimization using genetic algorithms (GAs). Although, there exist a number of multi-objective GA implementations and GA applications to interesting multi-objective optimization problems, there exists no systematic study to speculate what problem features may cause a multi-objective GA to face difficulties. In this paper, a number of such features are identified and a simple methodology is suggested to construct test problems from single-objective optimization problems. The construction method requires the choice of three functionals, each of which controls a particular aspect of difficulty that a multi-objective GA may have. One functional (f_1) tests an algorithm's ability to handle difficulties along the Pareto-optimal region, other functional (g) tests an algorithm's ability to handle difficulties lateral to the Pareto-optimal region, and the third functional (h) tests an algorithm's ability to handle difficulties arising because of different shapes of the Pareto-optimal region. This allows a multi-objective GA to be tested in a controlled manner on various aspects of problem difficulties. Specifically, multi-modal multi-objective problems, deceptive multi-objective problems, multi-objective problems having convex, non-convex, and discontinuous Pareto-optima fronts, and non-uniformly represented Pareto-optimal fronts are presented. In this regard, definitions of local and global Pareto-optimal fronts are introduced in this paper.

This paper has made a modest attempt to reveal and test some interesting aspects of multi-objective optimization. A number of other salient and related studies are suggested for future research. We believe that more such studies are needed to understand better the working principles of a multi-objective GA. An obvious outcome of such studies would be the development of new and improved multi-objective GAs.

The flip side of this study is also not less important. Since this paper shows a methodology to create a multi-objective optimization problem from single-objective optimization problems and since all properties (mode of difficulties) of the chosen single-objective optimization problem are retained in the resulting multi-objective problem (with some additional complexities related to multi-objective optimization), most theoretical or experimental studies on problem difficulties or on test function development in single-objective GAs are directly of importance to multi-objective optimization.

Acknowledgments

The author acknowledges the support provided by Alexander von Humboldt Foundation, Germany during the course of this study.

References

Cunha, A. G., Oliveira, P., and Covas, J. A. (1997). Use of genetic algorithms in multicriteria optimization to solve industrial problems. *Proceedings of the Seventh International Conference on Genetic Algorithms*. 682–688.

Covas, J. A., Cunha, A. G., and Oliveira, P. (in press). Optimization of single screw extrusion: Theoretical and experimental results. *International Journal of Forming Processes*.

Deb, K. (1995). *Optimization for engineering design: Algorithms and examples.* New Delhi: Prentice-Hall.

Deb, K. (in press). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*.

Deb, K. and Goldberg, D. E. (1989). An investigation of niche and species formation in genetic function optimization. *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 42–50).

Deb, K. and Goldberg, D. E. (1994). Sufficient conditions for arbitrary binary functions. *Annals of Mathematics and Artificial Intelligence, 10*, 385–408.

Deb, K., Horn, J., and Goldberg, D. E. (1993). Multi-Modal deceptive functions. *Complex Systems, 7*, 131–153.

Deb, K. and Kumar, A. (1995). Real-coded genetic algorithms with simulated binary crossover: Studies on multi-modal and multi-objective problems. *Complex Systems, 9*(6), 431–454.

Eheart, J. W., Cieniawski, S. E., and Ranjithan, S. (1993). Genetic-algorithm-based design of groundwater quality monitoring system. *WRC Research Report No. 218*. Urbana: Department of Civil Engineering, The University of Illinois at Urbana-Champaign.

Eshelman, L. J. (1990). The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. *Foundations of Genetic Algorithms*. 265–283.

Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multi-objective optimization: Formulaton, discussion and generalization. *Proceedings of the Fifth International Conference on Genetic Algorithms*. 416–423.

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

Fonseca , C. M. and Fleming, P. J. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms – Part II: Application example. *IEEE Transactions on Systems, Man, and Cybernetics– Part A: Systems and Humans, 28*(1). 38–47.

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

Goldberg, D. E., Deb, K., and Clark, J. H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex Systems, 6*, 333–362.

Goldberg, D. E., Deb, K., and Horn, J. (1992). Massive multimodality, deception, and genetic algorithms. *Parallel Problem Solving from Nature II*, 37–46.

Goldberg, D. E., Korb, B., and Deb, K. (1989). Messy genetic algorithms: Motivation, analysis, and first results, *Complex Systems, 3*, 93–530.

Goldberg, D. E. and Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. *Proceedings of the First International Conference on Genetic Algorithms and Their Applications*. 41–49.

Gordon, V. S. and Whitley, D. (1993). Serial and parallel genetic algorithms as function optimizers. *Proceedings of the Fifth International Conference on Genetic Algorithms*. 177–183.

Harik, G. (1997). Learning gene linkage to efficiently solve problems of bounded difficulty using genetic algorithms (IlliGAL Report No. 97005). Urbana: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Horn, J. (1997). Multicriterion decision making. In Eds. (T. Bäck et al.) *Handbook of Evolutionary Computation*.

Horn, J. and Nafploitis, N., and Goldberg, D. E. (1994). A niched Pareto genetic algorithm for multi-objective optimization. *Proceedings of the First IEEE Conference on Evolutionary Computation*. 82–87.

Kargupta, H. (1996). The gene expression messy genetic algorithm. *Proceedings of the IEEE International Conference on Evolutionary Computation*. 814–819.

Koziel, S. and Michalewicz, Z. (1998). A decoder-based evolutionary algorithm for constrained parameter optimization problems. *Proceedings of the Parallel Problem Solving from Nature, V*, 231–240.

Kurusawe, F. (1990). A variant of evolution strategies for vector optimization. *Parallel Problem Solving from Nature, I*, 193–197.

Laumanns, M., Rudolph, G., and Schwefel, H.-P. (1998). A spatial predator-prey approach to multi-objective optimization: A preliminary study. *Proceedings of the Parallel Problem Solving from Nature, V*. 241–249.

Leung, K.-S, Zhu, Z.-Y, Xu, Z.-B., and Leung, Y. (1998). Multiobjective optimization using nondominated sorting annealing genetic algorithms. (Unpublished document).

Mitra, K., Deb, K., and Gupta, S. K. (1998). Multiobjective dynamic optimization of an industrial Nylon 6 semibatch reactor using genetic algorithms. *Journal of Applied Polymer Science, 69*(1), 69–87.

Parks, G. T. and Miller, I. (1998). Selective breeding in a multi-objective genetic algorithm. *Proceedings of the Parallel Problem Solving from Nature, V*, 250–259.

Poloni, C., Giurgevich, A., Onesti, L., and Pediroda, V. (in press). Hybridisation of multiobjective genetic algorithm, neural networks and classical optimiser for complex design problems in fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*.

Rudolph, G. (1994). Convergence properties of canonical genetic algorithms. *IEEE Transactions on Neural Networks, NN-5*, 96–101.

Rudolph, G. (1998). Evolutionary search for minimal elements in partially ordered finite sets. *Evolutionary Programming VII*. 345–353.

Reklaitis, G. V., Ravindran, A. and Ragsdell, K. M. (1983). *Engineering optimization methods and applications*. New York: Wiley.

Schaffer, J. D. (1984). Some experiments in machine learning using vector evaluated genetic algorithms. (Doctoral Dissertation). Nashville, TN: Vanderbilt University.

Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. *Proceedings of the First International Conference on Genetic Algorithms*, 93–100.

Srinivas, N. and Deb, K. (1994). Multi-Objective function optimization using non-dominated sorting genetic algorithms, *Evolutionary Computation, 2*(3), 221–248.

Steuer, R. E. (1986). *Multiple criteria optimization: Theory, computation, and application*. New York: Wiley.

Suzuki, J. (1993). A markov chain analysis on a genetic algorithm. *Proceedings of the Fifth International Conference on Genetic Algorithms*, 146–153.

Syswerda, G. (1989). Uniform crossover in genetic algorithms. *Proceedings of the Third International Conference on Genetic Algorithms*, 2–9.

van Veldhuizen, D. and Lamont, G. B. (1998). Multiobjective evolutionary algorithm research: A history and analysis. *Report Number TR-98-03*. Wright-Patterson AFB, Ohio: Department of Electrical and Computer Engineering, Air Force Institute of Technology.

Weile, D. S., Michielssen, E., and Goldberg, D. E. (1996). Genetic algorithm design of Pareto-optimal broad band microwave absorbers. *IEEE Transactions on Electromagnetic Compatibility, 38*(4).

Whitley, D. (1989). The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. *Proceedings of the Third International Conference on Genetic Algorithms*, 116–121.

Whitley, D. (1990). Fundamental principles of deception in genetic search. *Foundations of Genetic Algorithms*. 221–241.

Zitzler, E. and Thiele, L. (1998a). Multiobjective optimization using evolutionary algorithms—A comparative case study. *Parallel Problem Solving from Nature, V*, 292–301.

Zitzler, E. and Thiele, L. (1998b). An evolutionary algorithm for multiobjective optimization: The strength Pareto approach. (Technical Report No. 43). Zurich: Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology.