Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms

N. Srinivas
Department of Mechanical Engineering
Indian Institute of Technology
Kanpur, UP 208 016, India

Kalyanmoy Deb
Department of Mechanical Engineering
Indian Institute of Technology
Kanpur, UP 208 016, India
deb@iitk.ernet.in

Abstract

In trying to solve multiobjective optimization problems, many traditional methods scalarize the objective vector into a single objective. In those cases, the obtained solution is highly sensitive to the weight vector used in the scalarization process and demands that the user have knowledge about the underlying problem. Moreover, in solving multiobjective problems, designers may be interested in a set of Pareto-optimal points, instead of a single point. Since genetic algorithms (GAs) work with a population of points, it seems natural to use GAs in multiobjective optimization problems to capture a number of solutions simultaneously. Although a vector evaluated GA (VEGA) has been implemented by Schaffer and has been tried to solve a number of multiobjective problems, the algorithm seems to have bias toward some regions. In this paper, we investigate Goldberg's notion of nondominated sorting in GAs along with a niche and speciation method to find multiple Pareto-optimal points simultaneously. The proof-of-principle results obtained on three problems used by Schaffer and others suggest that the proposed method can be extended to higher dimensional and more difficult multiobjective problems. A number of suggestions for extension and application of the algorithm are also discussed.

Keywords

Multiobjective optimization, nondominated sorting, ranking selection, and phenotypic sharing.

1. Introduction

Many real-world design or decision-making problems involve simultaneous optimization of multiple objectives. In principle, multiobjective optimization is very different from single-objective optimization. In single-objective optimization, one attempts to obtain the *best* design or decision, which is usually the global minimum or the global maximum, depending on whether the optimization problem is one of minimization or maximization. In the case of multiple objectives, there may not exist one solution that is best (global minimum or maximum) with respect to all objectives. In a typical multiobjective optimization problem, there exists a set of solutions that are superior to the rest of the solutions in the search space when all objectives are considered but are inferior to other solutions in the space in one or more objectives. These solutions are known as *Pareto-optimal* solutions or *nondominated* solutions (Chankong & Haimes, 1983; Hans, 1988). The rest of the solutions are known as dominated solutions. Since none of the solutions in the nondominated set is *absolutely* better than any other, any one of them is an acceptable solution. The choice of one solution over the other requires problem knowledge and a number of problem-related factors. Thus, one

© 1995 The Massachusetts Institute of Technology

solution chosen by a designer may not be acceptable to another designer or in a changed environment. Therefore, in multiobjective optimization problems, it may be useful to have knowledge about alternative Pareto-optimal solutions.

One way to solve multiobjective problems is to scalarize the vector of objectives into one objective by averaging the objectives with a weight vector. This process allows a simpler optimization algorithm to be used, but the obtained solution largely depends on the weight vector used in the scalarization process. Moreover, if available, a decision maker may be interested in knowing alternate solutions. Because genetic algorithms (GAs) work with a population of points, a number of Pareto-optimal solutions may be captured using GAs. An early GA application on multiobjective optimization by Schaffer (1984) opened a new avenue of research in this field. Though his algorithm, VEGA, gave encouraging results, it suffered from bias toward some Pareto-optimal solutions. A new algorithm, Nondominated Sorting Genetic Algorithm (NSGA), is presented in this paper based on Goldberg's suggestion (Goldberg 1989). This algorithm eliminates the bias in VEGA and thereby distributes the population over the entire Pareto-optimal regions. Although there exist two other implementations (Fonesca & Fleming, 1993; Horn, Nafpliotis, & Goldberg, 1994) based on this idea, NSGA is different from their working principles and is closer to Goldberg's suggestion.

In the remainder of the paper, we briefly describe the difficulties of using three common classical methods to solve multiobjective optimization problems. A brief introduction to Schaffer's VEGA and its problems are outlined. Thereafter, the nondominated sorting GA is described and applied to three two-objective test problems. Simulation results show that NSGA performs better than VEGA on these problems. A number of extensions to this work are also suggested.

2. Multiobjective Optimization Problem

A general multiobjective optimization problem consists of a number of objectives and is associated with a number of inequality and equality constraints. Mathematically, the problem can be written as follows (Rao, 1991):

$$\begin{aligned} &\text{Minimize/Maximize} && f_i(\mathbf{x}) && i = 1, 2, \ldots, N \\ &\text{Subject to} \\ & && g_j(\mathbf{x}) \leq 0 && j = 1, 2, \ldots, \mathcal{J} \\ & && h_k(\mathbf{x}) = 0 && k = 1, 2, \ldots, K \end{aligned} \tag{1}$$

The parameter $\mathbf{x}$ is a p dimensional vector having p design or decision variables. Solutions to a multiobjective optimization problem are mathematically expressed in terms of nondominated or superior points. In a minimization problem, a vector $\mathbf{x}^{(1)}$ is partially less than another vector $\mathbf{x}^{(2)}$, ($\mathbf{x}^{(1)} \prec \mathbf{x}^{(2)}$), when no value of $\mathbf{x}^{(2)}$ is less than $\mathbf{x}^{(1)}$ and at least one value of $\mathbf{x}^{(2)}$ is strictly greater than $\mathbf{x}^{(1)}$. If $\mathbf{x}^{(1)}$ is partially less than $\mathbf{x}^{(2)}$, we say that the solution $\mathbf{x}^{(1)}$ *dominates* $\mathbf{x}^{(2)}$ or the solution $\mathbf{x}^{(2)}$ is *inferior* to $\mathbf{x}^{(1)}$ (Tamura & Miura, 1979). Any member of such vectors that is not dominated by any other member is said to be *nondominated* or *noninferior*. Similarly, if the objective is to maximize a function we define a dominated point if the corresponding component is not greater than that of a nondominated point. The optimal solutions to a multiobjective optimization problem are nondominated solutions. They are also known as *Pareto-optimal* solutions. The concept of Pareto-optimality is further illustrated in the next section by presenting an example problem. Mathematically, an optimization algorithm should be terminated if any one of the Pareto-optimal solutions is obtained. But in practice, since there could be a number of Pareto-optimal solutions and the

suitability of one solution depends on a number of factors including designer's choice and problem environment, finding the entire set of Pareto-optimal solutions may be desired. In the following section, we describe a number of classical approaches to the solution of multiobjective optimization problems and discuss their difficulties by illustrating a simple two-objective optimization problem.

3. Classical Methods

A common difficulty with multiobjective optimization is the appearance of an *objective conflict* (Hans, 1988)—none of the feasible solutions allow simultaneous optimal solutions for all objectives. In other words, individual optimal solutions of each objective are usually different. Thus, a mathematically most favorable Pareto-optimum is that solution which offers the least objective conflict. Such solutions can be viewed as points in the search space that are optimally placed from the individual optimum of each objective. But such solutions may not satisfy a decision maker because he or she may want a solution that satisfies some associated priorities of the objectives. To find such points all classical methods scalarize the objective vector into one objective. Many classical algorithms for nonlinear vector optimization techniques define a substitute problem, reducing the vector optimization to a scalar optimization problem. Using such a substitute, a compromise solution is found subjected to specified constraints.

In the following subsections, three commonly used methods—method of objective weighting, method of distance functions, and method of min-max formulation—are discussed.

3.1 Method of Objective Weighting

This is probably the simplest of all classical techniques. Multiple objective functions are combined into one overall objective function, Z, as follows:

$$\begin{aligned} Z &= \sum_{i=1}^{\mathbf{N}} w_i f_i(\mathbf{x}), \\ \text{where } \mathbf{x} &\in \mathbf{X}, \quad \mathbf{X} \text{ represents the feasible region;} \end{aligned} \tag{2}$$

the weights w_i are fractional numbers ($0 \leq w_i \leq 1$), and all weights are summed up to 1, or $\sum_{i=1}^{\mathbf{N}} w_i = 1$. In this method, the optimal solution is controlled by the weight vector $\mathbf{w}$. It is clear from Equation 2 that the preference of an objective can be changed by modifying the corresponding weight. Mathematically, a solution obtained with equal weights to all objectives may offer least objective conflict, but as a real-world situation demands a satisfying solution, priority must be induced in the formulation. In most cases, each objective is first optimized and all objective function values are computed at each individual optimum solution. Thereafter, depending on the importance of objectives a suitable weight vector is chosen and the single-objective problem given in Equation 2 is used to find the desired solution. The only advantage of using this technique is that the emphasis of one objective over the other can be controlled, and the obtained solution is a usually a Pareto-optimum solution.

3.2 Method of Distance Functions

In this method, the scalarization is achieved by using a demand-level vector $\bar{\mathbf{y}}$ which has to be specified by the decision maker. The single objective function derived from multiple

objectives is as follows:

$$Z = \left[\sum_{i=1}^{\mathbf{N}} |f_i(\mathbf{x}) - \bar{y}_i|^r\right]^{1/r}, \qquad 1 \leq r < \infty, \tag{3}$$

where $\mathbf{x} \in \mathbf{X}$ (the feasible region). Usually a Euclidean metric $r = 2$ is chosen, with $\bar{\mathbf{y}}$ as individual optima of objectives (Hans, 1988). It is important to note that the solution obtained by solving Equation 3 depends on the chosen demand-level vector. Arbitrary selection of a demand level may be highly undesirable; a wrong demand level will lead to a non-Pareto-optimal solution. As the solution is not guaranteed, the decision maker must have a thorough knowledge of individual optima of each objective prior to the selection of demand level. In a way this method works as a goal programming technique imposing a goal vector, $\bar{\mathbf{y}}$ (demand level), on the given objectives.

This method is similar to the method of objective weighting. The only difference is that in this method the goal for each objective function is required to be known whereas in the previous method the relative importance of each objective is required.

3.3 Min-Max Formulation

This method is different in principle from the above two methods. This method attempts to minimize the relative deviations of the single objective functions from the individual optimum. That is, it tries to minimize the objective conflict. For a minimization problem, the corresponding min-max problem is formulated as follows:

$$\text{minimize } \mathcal{F}(\mathbf{x}) = \max\left[Z_j(\mathbf{x})\right], \qquad j = 1, 2, \ldots, \mathbf{N}, \tag{4}$$

where $\mathbf{x} \in \mathbf{X}$ (the feasible region) and $Z_j(\mathbf{x})$ is calculated for nonnegative target optimal value $\bar{f}_j > 0$ as follows:

$$Z_j(\mathbf{x}) = \frac{f_j - \bar{f}_j}{\bar{f}_j}, \quad j = 1, 2, \ldots, \mathbf{N}. \tag{5}$$

This method can yield the best possible compromise solution when objectives with equal priority are required to be optimized. However, the priority of each objective can be varied by introducing dimensionless weights in the formulation. This can also be modified as a goal-programming technique by introducing a demand-level vector in the formulation.

3.4 Drawbacks of Classical Methods

In all the above methods, multiple objectives are combined to form one objective by using some knowledge of the problem being solved. The optimization of the single objective may guarantee a Pareto-optimal solution but results in a single-point solution. In real-world situations decision makers often need different alternatives in decision making. Moreover, if some of the objectives are noisy or have discontinuous variable space these methods may not work effectively. Some of these methods are also expensive as they require knowledge of the individual optimum prior to vector optimization. The most profound drawback of these algorithms is their sensitivity toward weights or demand levels. The decision maker must have a thorough knowledge of the priority of each objective before forming the single objective from a set of objectives. The solutions obtained largely depend on the underlying weight vector or demand level. Thus, for different situations, different weight vectors need to be used and the same problem needs to be solved a number of times. We illustrate this aspect by considering a simple example.

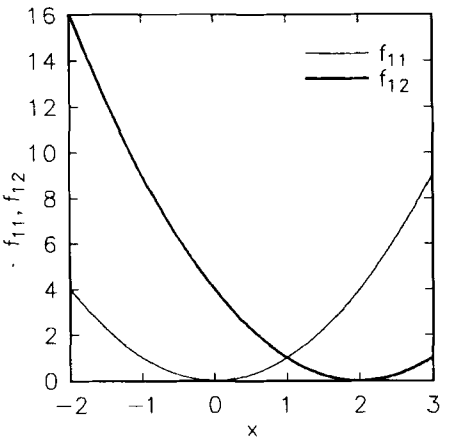


Figure 1. Functions f_{11} and f_{12} are plotted versus x.

A simple two-objective problem $F1$ of one variable is considered to illustrate the concept of multiple Pareto-optimality. This problem was used for the same purpose by Vincent and Grantham (1981) and subsequently by Schaffer (1984). The problem has two objectives and is shown in Figure 1 and Figure 2:

$$\begin{aligned} &\text{Minimize } f_{11} = x^2, \\ &\text{Minimize } f_{12} = (x-2)^2. \end{aligned} \tag{6}$$

From the plot showing the performance space, it is clear that the Pareto-optimal solutions constitute all x values varying from 0 to 2. The solution $x = 0$ is optimum with respect to f_{11} but not so good with respect to f_{12}, and the solution $x = 2$ is optimum with respect to function f_{12} and not so good with respect to f_{11}. Any other point in between is a compromise or trade-off to the above two functions and is a Pareto-optimum point. But the solution $x = 3$, for example, is not a Pareto-optimum point since this point is not better than the solution $x = 2$ with respect to either objective.

Among the possible Pareto-optimal points, the decision maker may want to prefer one point over the other depending on the situation; before taking any decision, he or she may want to know the other possible Pareto-optimal solutions. The traditional methods cannot find multiple Pareto-optimal solutions simultaneously. For example, with all the above methods and with equal priority to both functions having a weight vector (0.5, 0.5), and demand levels as individual optima, the obtained solution is $x^* = 1$. A weight vector (1, 0) results in a scalarized objective as f_{11}. The solution obtained in this case is $x^* = 0$, which is optimum in f_{11} but not so good in f_{12}. Similarly the weight vector (0, 1) produces the solution $x^* = 2$, which is the minimum point of f_{12}. Any point in the range $0 \leq x \leq 2$ may be a valid compromise and can be obtained with a particular choice of a weight vector. Thus, in order to obtain a particular solution, the decision maker has to know the corresponding weight vector, which is a difficult problem in itself. Another problem with using classical methods is that often some objectives may involve uncertainties. If the objective functions are not deterministic, the fixation of a weight vector or a demand level may become even more difficult. This discussion suggests that the classical methods of handling multiobjective

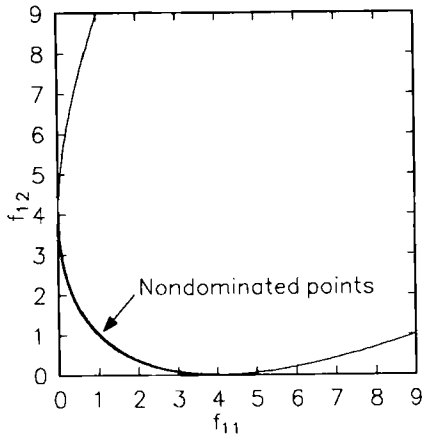


Figure 2. The performance space of problem $F1$ is shown.

optimization problems are inadequate and inconvenient to use. A more realistic method would be one that can find multiple Pareto-optimal solutions simultaneously so that decision makers may be able to choose the most appropriate solution for the current situation. The knowledge of many Pareto-optimal solutions is also useful for later use, particularly when the current situation has changed and a new solution is required to be implemented. Since genetic algorithms deal with a population of points instead of one point, multiple Pareto-optimal solutions can be captured in the population in a single run. In the following section, we describe previous studies and current implementation of GAs to solve multiobjective optimization problems.

4. GA Implementation

As early as 1967, Rosenberg suggested, but did not simulate, a genetic search to the simulation of the genetics and the chemistry of a population of single-celled organisms with multiple properties or objectives (Rosenberg, 1967). The first practical algorithm, called Vector Evaluated Genetic Algorithm (VEGA), was developed by Schaffer in 1984 (Schaffer, 1984). One of the problems with VEGA, as realized by Schaffer himself, is its bias toward some Pareto-optimal solutions. Later, Goldberg suggested a nondominated sorting procedure to overcome this weakness of VEGA (Goldberg, 1989). Our algorithm, Nondominated Sorting Genetic Algorithm (NSGA), is developed based on this idea. There exist at least two other studies, different from our algorithm, based on Goldberg's idea. In the rest of this section, we discuss the merits and drawbacks of VEGA and NSGA, and the differences between NSGA and the two other recent implementations.

4.1 Schaffer's VEGA

Schaffer modified the simple tripartite genetic algorithm by performing independent selection cycles according to each objective. He modified Grefenstette's GENESIS program (Schaffer, 1984) by creating a loop around the traditional selection procedure so that the selection method is repeated for each individual objective to fill up a portion of the mating

pool. Then the entire population is thoroughly shuffled to apply crossover and mutation operators. This is performed to achieve the mating of individuals of different subpopulation groups.

The algorithm worked efficiently for some generations, but in some cases it suffered from its bias toward some individuals or regions. The independent selection of specialists resulted in speciation in the population. The outcome of this effect is the convergence of the entire population toward the individual optimum regions after a large number of generations. Being a decision maker, we may not like to have any bias toward such middling individuals; rather, we may want to find as many nondominated points as possible. Schaffer tried to minimize this speciation by developing two heuristics—the nondominated selection heuristic (a wealth redistribution scheme) and the mate selection heuristic (a cross-breeding scheme) (Schaffer, 1984). In the nondominated selection heuristic, dominated individuals are penalized by subtracting a small fixed penalty from their expected number of copies during selection. Then the total penalty for dominated individuals was divided among the nondominated individuals and was added to their expected number of copies during selection. But this algorithm failed when the population had very few nondominated individuals, resulting in a large fitness value for those few nondominated points, eventually leading to a high selection pressure. The mate selection heuristic was intended to promote the cross-breeding of specialists from different subgroups. This was implemented by selecting an individual, as a mate to a randomly selected individual, which has the maximum Euclidean distance in the performance space from its mate. But it failed too to prevent the participation of poorer individuals in the mate selection because of random selection of the first mate and the possibility of a large Euclidean distance between a champion and a mediocre. Schaffer concluded that the random mate selection is far superior to this heuristic.

One method to minimize speciation is through a nondominated sorting procedure in conjunction with a sharing technique, as suggested by Goldberg (1989). Recently Fonesca and Fleming (1993) and Horn, Nafpliotis, and Goldberg (1994) implemented that suggestion and successfully applied it to some problems. These methods are briefly discussed later. First, we discuss our algorithm NSGA, which is also based on Goldberg's suggestions.

4.2 Nondominated Sorting

The idea behind the nondominated sorting procedure is that a ranking selection method is used to emphasize good points and a niche method is used to maintain stable subpopulations of good points. Our algorithm is developed based on this concept. Since the algorithm is based on a nondominated sorting procedure, we call this algorithm the Nondominated Sorting Genetic Algorithm, NSGA.

4.2.1 Nondominated Sorting Genetic Algorithm (NSGA) NSGA differs from a simple genetic algorithm only in the way the selection operator works. The crossover and mutation operators remain as usual. Before the selection is performed, the population is ranked on the basis of an individual's nondomination, described in Section 2. The nondominated individuals present in the population are first identified from the current population. Then all these individuals are assumed to constitute the first nondominated front in the population and assigned a large dummy fitness value. The same fitness value is assigned to give an equal reproductive potential to all these nondominated individuals. To maintain diversity in the population, these classified individuals are then *shared* with their dummy fitness values. Sharing methods are discussed elsewhere (Goldberg & Richardson, 1987; Deb, 1989; Deb & Goldberg, 1991). Sharing is achieved by performing selection operation

using degraded fitness values that are obtained by dividing the original fitness value of an individual by a quantity proportional to the number of individuals around it. This causes multiple optimal points to co-exist in the population. After sharing, these nondominated individuals are ignored temporarily to process the rest of the population in the same way to identify individuals for the second nondominated front. These nondominated points are then assigned a new dummy fitness value that is kept smaller than the minimum shared dummy fitness of the previous front. This process is continued until the entire population is classified into several fronts.

The population is then reproduced according to the dummy fitness values. A stochastic remainder proportionate selection is used in this study. Since individuals in the first front have the maximum fitness value, they always get more copies than the rest of the population. This was intended to search for nondominated regions or Pareto-optimal fronts. This results in quick convergence of the population toward nondominated regions, and sharing helps to distribute it over this region. By emphasizing nondominated points, NSGA favors the schemata representing Pareto-optimal regions. The efficiency of NSGA lies in the way multiple objectives are reduced to a dummy fitness function using a nondominated sorting procedure. Another aspect of our method is that practically any number of objectives can be solved. Both minimization and maximization problems can also be handled by this algorithm. The only place a change is required for the above two cases is the way the nondominated points are identified, as discussed in Section 2.

Figure 3 shows a flow chart of this algorithm. The algorithm is similar to a simple GA except for the classification of nondominated fronts and the sharing operation. The sharing in each front is achieved by calculating a sharing function value between two individuals in the same front as follows:

$$Sh(d_{ij}) = \begin{cases} 1 - \left(\frac{d_{ij}}{\sigma_{\text{share}}}\right)^2, & \text{if } d_{ij} < \sigma_{\text{share}}; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

In the above equation, the parameter d_{ij} is the phenotypic distance between two individuals i and j in the current front, and σ_{share} is the maximum phenotypic distance allowed between any two individuals to become members of a niche. Some guidelines to set these parameters appear elsewhere (Deb, 1989). A parameter *niche count* is calculated by adding the above sharing function values for all individuals in the current front. Finally, the shared fitness value of each individual is calculated by dividing its dummy fitness value by its niche count.

Fonesca and Fleming (1993) implemented Goldberg's suggestion in different way. In this study, the multiobjective optimization GA (MOGA) uses a similar sorting procedure presented in this paper. In MOGA, the whole population is checked and all nondominated individuals are assigned rank 1. Other individuals are ranked by checking the nondominance of them with respect to the rest of the population in the following way. For an individual point, the number of points that strictly dominate the point in the population is first found. Thereafter, the rank of that individual is assigned to be one more than that number. Therefore, at the end of this ranking procedure, there could be a number of points having the same rank. The selection procedure then uses these ranks to select or delete blocks of points to form the mating pool. As discussed elsewhere (Goldberg & Deb, 1991), this type of blocked fitness assignment is likely to produce a large selection pressure that might cause premature convergence. MOGA also uses a niche-formation method to distribute the population over the Pareto-optimal region. But instead of performing sharing on the parameter values, they have used sharing on objective function values. Even though this maintains diversity in the objective function values, this may not maintain diversity in the parameter set, a matter of

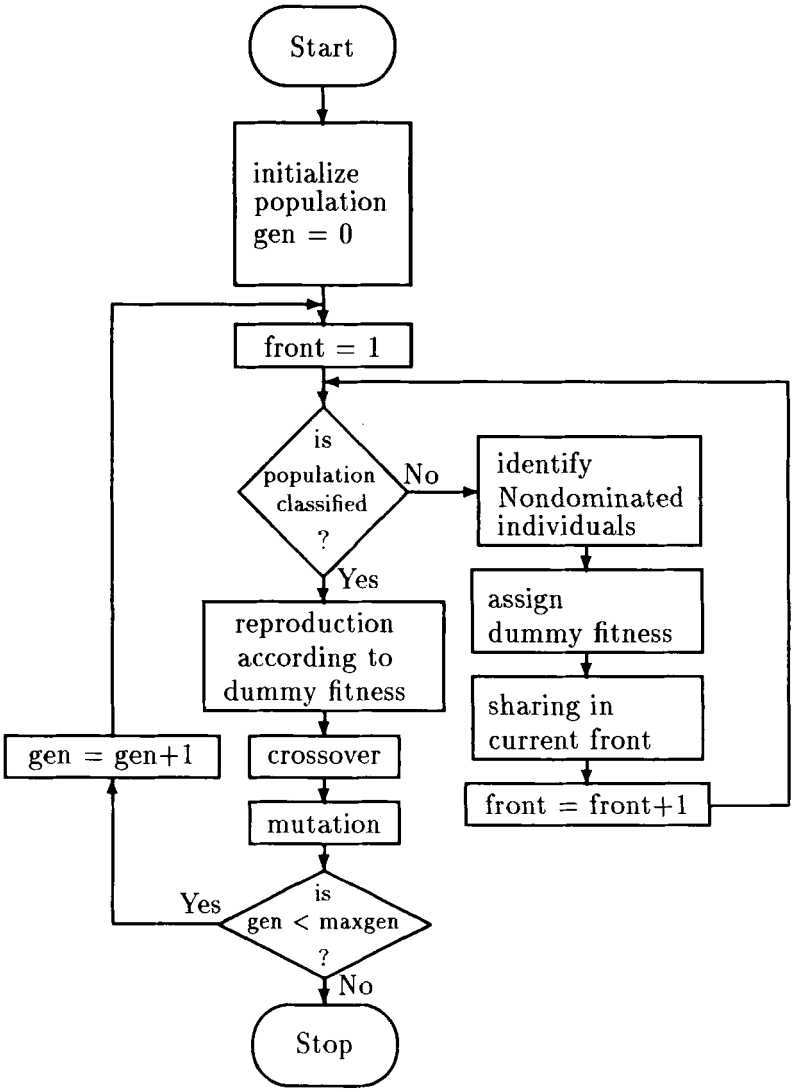


Figure 3. Flow chart of NSGA.

importance for a decision maker. Moreover, MOGA may not be able to find multiple solutions in problems where different Pareto-optimal points correspond to the same objective function value (Srinivas, 1994). However, the ranking of the individuals according to their nondominance in the population is an important aspect of this work.

Horn, Nafpliotis, and Goldberg (1994) used *Pareto domination tournaments* instead of the nondominated sorting and ranking selection method in solving multiobjective optimization problems. In this method, a *comparison set* comprising a specific number (t_{dom}) of individuals is picked at random from the population at the beginning of each selection process. Two random individuals are picked from the population for selecting a winner in a tournament selection according to the following procedure. Both individuals are compared with the

members of the comparison set for domination with respect to the objective functions. If one of them is nondominated and the other is dominated, then the nondominated point is selected. On the other hand, if both are either nondominated or dominated, a niche count is found for each individual in the entire population. The niche count is calculated by simply counting the number of points in the population within a certain distance (σ_{share}) from an individual. The individual with least niche count is selected. The effect of multiple objectives is taken into the nondominance calculation. Since this nondominance is computed by comparing an individual with a randomly chosen population set of size t_{dom}, the success of this algorithm highly depends on the parameter t_{dom}. If a proper size is not chosen, true nondominated (Pareto-optimal) points may not be found. If a small t_{dom} is chosen, this may result in a few nondominated points in the population. Instead, if a large t_{dom} is chosen, premature convergence may result. This aspect is also observed by the authors. They have presented some empirical results with various t_{dom} values. Nevertheless, the concept of niche formation among the nondominated points is an important aspect of this work.

NSGA implements both aspects of Goldberg's suggestion in a better way. The ranking classification is performed according to the nondominance of the individuals in the population, and a distribution of the nondominated points is maintained using a niche formation technique. Both aspects cause the distinct nondominated points to be found in the population.

5. Simulation Results

In this section, we apply NSGA and VEGA on three test problems, two of which were used by Schaffer (1984) and another one was solved by Chankong and Haimes (1983). In all simulations, the GA parameters used in the experiments are as follows:

Maximum generation	:	500
Population size	:	100
String length (binary code)	:	32
Probability of crossover	:	1.0
Probability of mutation	:	0.0

Mutation probability is kept zero in order to observe the effectiveness of NSGA alone. The parameters are held constant across all runs. Unbiased initial population is generated randomly, spreading over the entire variable space in consideration. To make the comparison fair, exactly the same initial population has been used in VEGA and NSGA. To confirm and recheck the solutions, each experiment is repeated five times with different initial populations, and the average performance is presented in each case.

5.1 Problem *F*1

This problem is defined in Section 3. Initial range for the design variable used in simulations is $(-10, 10)$, but the nondominated region is only $(0, 2)$. A comparison of population drift with NSGA and VEGA is shown in Figures 4 through 11. These figures are drawn in performance space, and the σ_{share} parameter used is 0.1. As mentioned earlier, the initial population (at generation 0) is exactly the same for both VEGA and NSGA. At generation 10, both methods show the convergence of the population toward the nondominated region. At generation 100, the difference in distribution is clear, and at generation 500, VEGA's

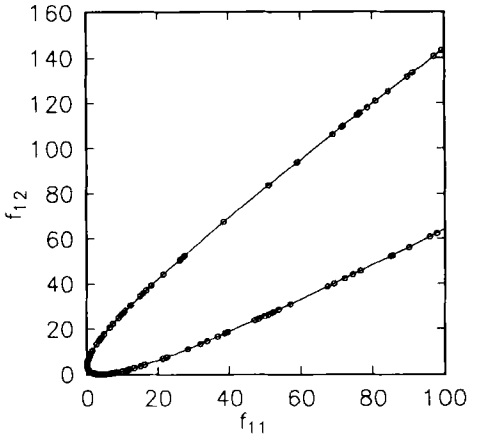


Figure 4. Population at generation 0 obtained using NSGA for problem $F1$ is shown.

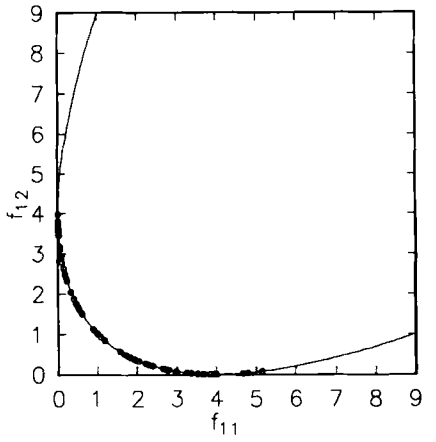


Figure 5. Population at generation 10 obtained using NSGA for problem $F1$ is shown.

population converged to only three subregions that are close to individual optima. Figures 4 through 7 also show the ability of NSGA in distributing the population uniformly and maintaining it until generation 500.

To study the distribution pattern better, the nondominated search space (0, 2) is divided into 10 equal subregions. Since the population size is 100, we expect to have about 10 individuals in each subregion. Figures 12 and 13 show plots drawn with the number of individuals in each subregion and generation number. In the case of VEGA (Figure 12), after some generations, some of the subregions do not have any representation at all. These subregions represent middling individuals. Observations based on a number of simulation results reveal that at most three subregions are populated by VEGA. These are the points around individual optima. In the case of NSGA (Figure 13), the number of individuals in each

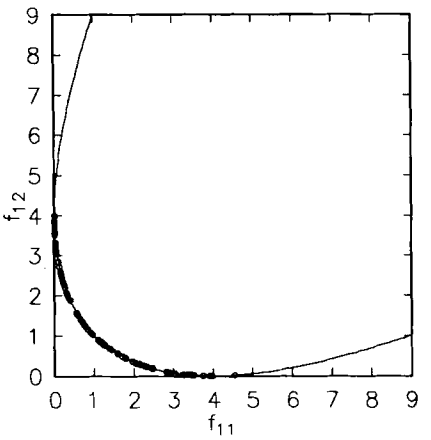


Figure 6. Population at generation 100 obtained using NSGA for problem $F1$ is shown.

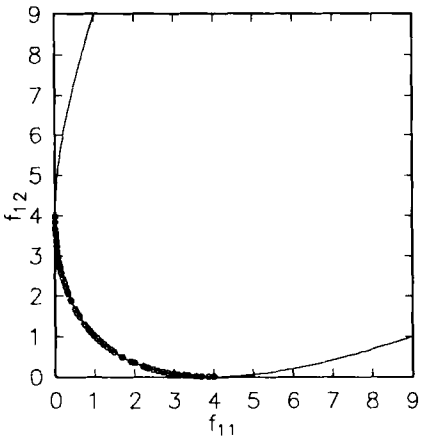


Figure 7. Population at generation 500 obtained using NSGA for problem $F1$ is shown.

subregion fluctuated around a value of 10, which is exactly the expected number of points in each subregion. It is important to note that none of the subregions have zero individuals. Experiments are also carried out by considering a large search space, $-1000 \leq x \leq 1000$, compared to the nondominated region, $0 \leq x \leq 2$, so that the initial random population of size 100 has no Pareto-optimal points. Figure 14 shows that at generation 20 or so, the population is almost filled with nondominated points. The figure also shows that NSGA maintains a large proportion of the population at the nondominated region. A similar trend is also observed till generation 500. The distribution of points in the nondominated region is similar to that in the earlier experiments.

To quantify this distribution capability of population over nondominated regions, we calculate a performance measure, which is discussed in the following subsection.

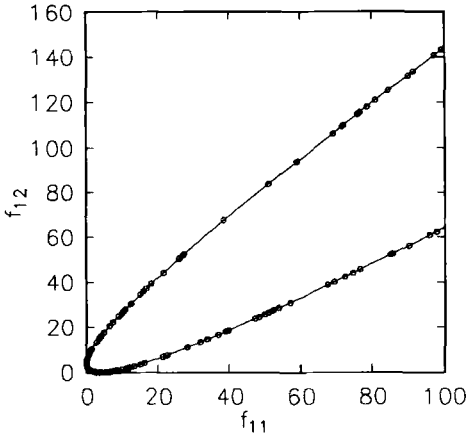


Figure 8. Population at generation 0 obtained using VEGA for problem $F1$ is shown.

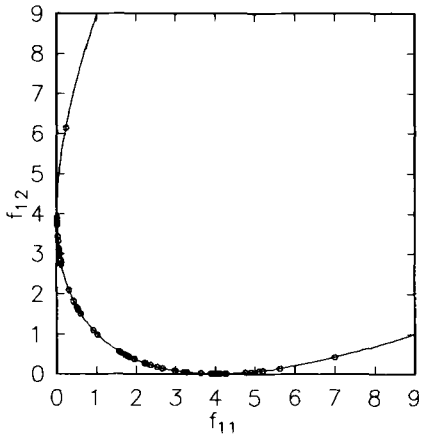


Figure 9. Population at generation 10 obtained using VEGA for problem $F1$ is shown.

5.1.1 Performance Measure In order to investigate how well NSGA and VEGA have distributed individuals over the nondominated region, we use the chi-square-like deviation form distribution measure used elsewhere (Deb, 1989).

$$\text{Performance measure, } \iota = \sqrt{\sum_{i=1}^{q+1}\left(\frac{n_i - \bar{n}_i}{\sigma_i}\right)^2}, \tag{8}$$

where q is the number of desired optimal points and the $(q+1)$-th subregion is the dominated region, n_i is actual number of individuals serving i-th subregion (niche) of the nondominated region, $\bar{n}_i$ is expected number of individuals serving i-th subregion of the nondominated region, and σ_i^2 is the variance of individuals serving i-th subregion of the nondominated

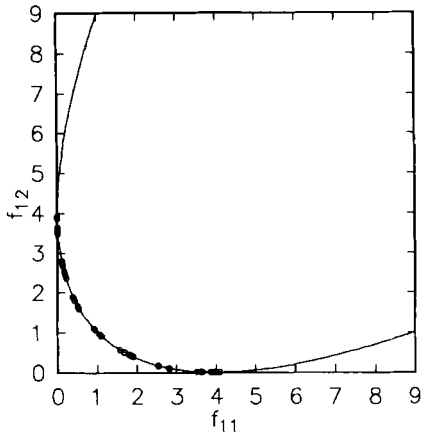


Figure 10. Population at generation 100 obtained using VEGA for problem $F1$ is shown.

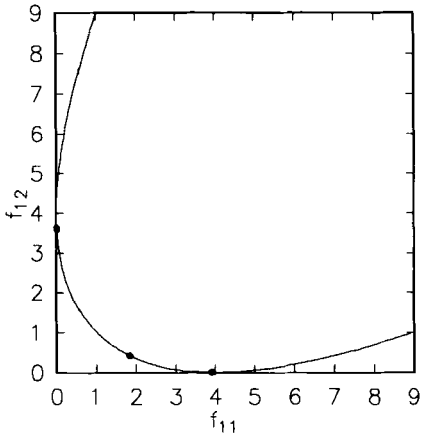


Figure 11. Population at generation 500 obtained using VEGA for problem $F1$ is shown.

region. Using probability theory it was estimated elsewhere (Deb, 1989) that

$$\sigma_i^2 = \bar{n}_i \left(1 - \frac{\bar{n}_i}{\mathrm{P}}\right), \qquad i = 1, 2, \ldots, q,$$

where P is the population size. Since it is not desirable to have any individual in the dominated region ($(q+1)$-th subregion), $\bar{n}_{q+1} = 0$. That study also showed that $\sigma_{q+1}^2 = \sum_{i=1}^{q} \sigma_i^2$. If the distribution of points is ideal with $\bar{n}_i$ number of points in i-th subregion, the performance measure $\iota = 0$. Therefore, an algorithm with a good distributing capability is characterized by a low deviation measure.

To analyze the distribution using this measure, the nondominated region is divided into the same 10 equal subregions (each having a length of 0.2 units in the variable space). Since a

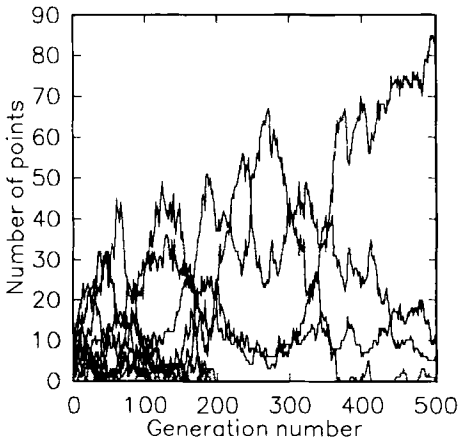


Figure 12. Number of individuals in each subregion versus generation for $F1$ using VEGA is shown.

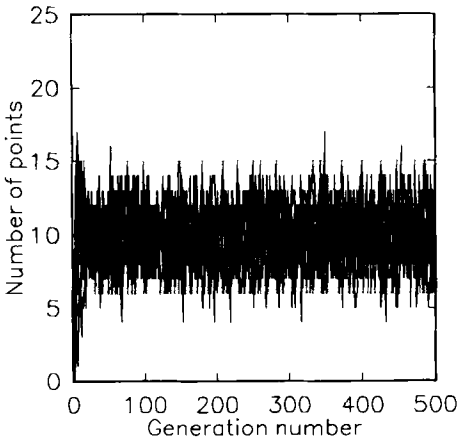


Figure 13. Number of individuals in each subregion versus generation for $F1$ using NSGA is shown.

population of 100 individuals is used, the expected number of points per subregion ($\bar{n}_i$) is 10 with a variance $\sigma_i^2 = 9$. Therefore, the expected variance of dominated individuals $\sigma_{11}^2 = 90$. The actual number of individuals in each subregion is counted and the deviation measure is calculated using Equation 8. Figure 15 shows the deviation measure versus generation number for VEGA and NSGA applied on $F1$. Figure 15 shows the average performance of five runs with different initial populations while taking the same initial population for VEGA and NSGA. Initially both methods start with a high-performance measure because the initial population is spread over the entire variable space with fewer individuals in the nondominated region. VEGA's increasing measure with generation indicates its poor distributing ability. The initial descent is due to the convergence of population toward the nondominated region. At the same time, NSGA with $\sigma_{\text{share}} = 0.1$ (induced number of niches in nondominated region

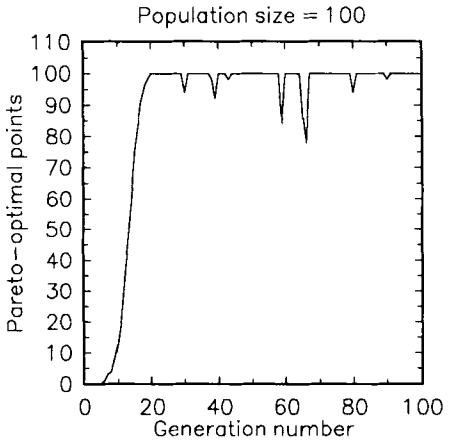


Figure 14. The number of Pareto-optimal points at each generation is plotted for problem $F1$. The initial population did not have any Pareto-optimal points.

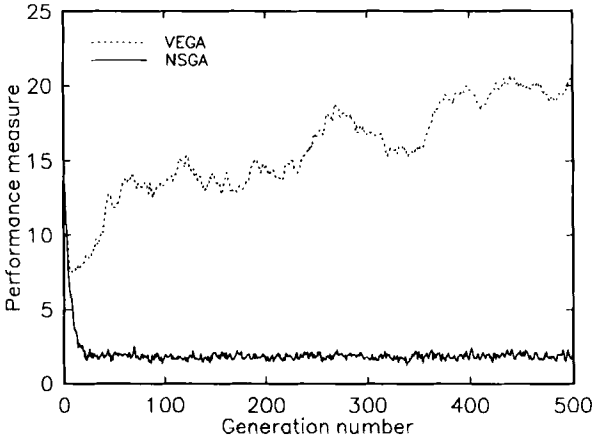


Figure 15. Performance measure ι for NSGA and VEGA on problem $F1$ is plotted versus generation number. An average of five runs is plotted.

is 10), fluctuated at a low deviation measure. This is continued until generation 500, which is long enough to justify the stability of the population distribution in 10 subregions. This shows the ability of NSGA in distributing population over the nondominated region.

To investigate how sensitive the NSGA results are on σ_{share} values, a number of σ_{share} values are tried. Figure 16 shows performance of NSGA with different σ_{share} values. To make a fair comparison among these results, the initial population is taken to be the same in all cases. There is not much difference in performance with $\sigma_{\text{share}} = 0.1$ and $\sigma_{\text{share}} = 0.2$. This shows that both the values resulted in successful distribution of population. But with a considerably high sharing parameter, $\sigma_{\text{share}} = 1.0$, NSGA's performance is poor. A similar

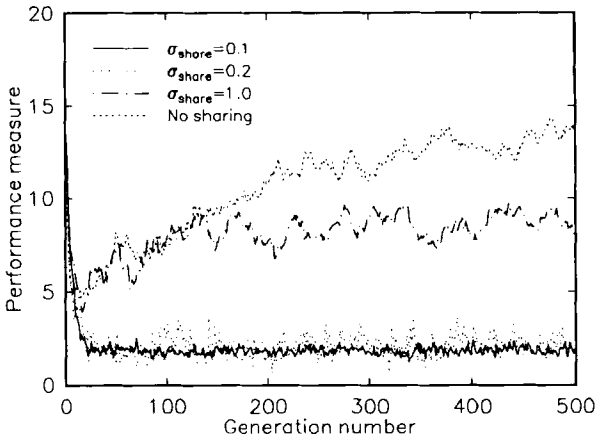


Figure 16. Effect of varying σ_{share} values is shown for problem $F1$.

observation can be made in the case of negligible σ_{share} or without sharing. It is important here to note that although these two cases exhibit increasing deviation measure they are not as poor as that in VEGA. This is due to the fact that equal reproductive potential (dummy fitness) is maintained for all nondominated individuals, thereby minimizing the bias against middling points.

These results suggest that NSGA is effective in finding multiple Pareto-optimal solutions and is better than VEGA in that respect. To consolidate our results better, we try using NSGA on another multiobjective optimization problem used by Schaffer.

5.2 Problem $F2$

The second problem is given in the following (Schaffer, 1984):

$$\begin{aligned}
\text{Minimize } f_{21} &= -x && \text{if } x \leq 1 \\
&= -2 + x && \text{if } 1 < x \leq 3 \\
&= 4 - x && \text{if } 3 < x \leq 4 \\
&= -4 + x && \text{if } x > 4 \\
\text{Minimize } f_{22} &= (x-5)^2 && \qquad (9)
\end{aligned}$$

Functions f_{21} and f_{22} are shown in Figures 17 and 18. The specialty of this problem is its disjointed nondominated regions. These can be seen in Figure 18 as regions $1 \leq x \leq 2$ and $4 \leq x \leq 5$. Here the net length of this region is 2 units in the variable space.

The population evolution is shown in Figures 19 through 26. Both algorithms successfully identified disjointed nondominated regions. But the difference in distribution is clearly visible at 100-th and 500-th generations. This result reiterates the ability of NSGA in distributing the population. The nondominated region is divided into 10 subregions to analyze the distribution of population. Figures 27 and 28 show the number of individuals in each subregion versus generation. In this problem also VEGA failed to sustain some of the subregions, whereas NSGA successfully distributed individuals over both disjointed Pareto-optimal fronts. The deviation measure for these algorithms was similar in pattern to that of problem $F1$.

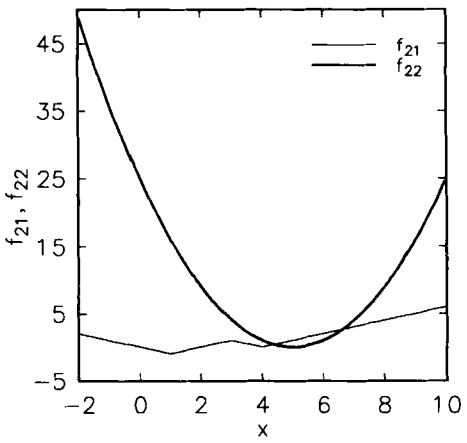


Figure 17. Problem *F2* is plotted between f_{21}, f_{22}, and x.

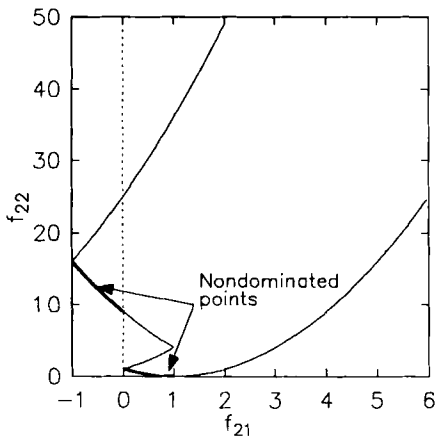


Figure 18. Problem *F2* is plotted between f_{21} and f_{22}.

5.3 Problem *F3*

This problem is used to test NSGA's ability in optimizing multiparameter, multiobjective problems as well as handling constrained search spaces:

$$\begin{aligned} &\text{Minimize} && f_{31} = (x_1 - 2)^2 + (x_2 - 1)^2 + 2, \quad \text{and} \\ &\text{Minimize} && f_{32} = 9x_1 - (x_2 - 1)^2, \\ &\text{Subject to} \\ &&& x_1^2 + x_2^2 - 225 \leq 0, \\ &&& x_1 - 3x_2 + 10 \leq 0. \end{aligned} \tag{10}$$

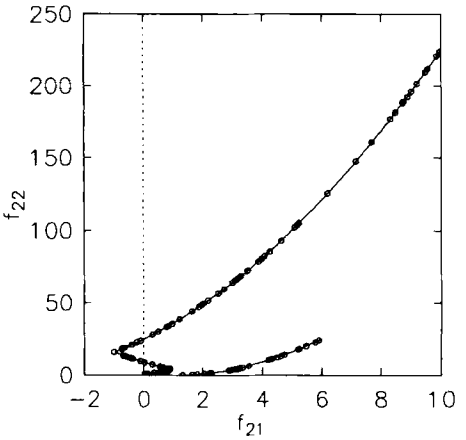


Figure 19. Population at generation 0 obtained using NSGA for problem *F2* is shown.

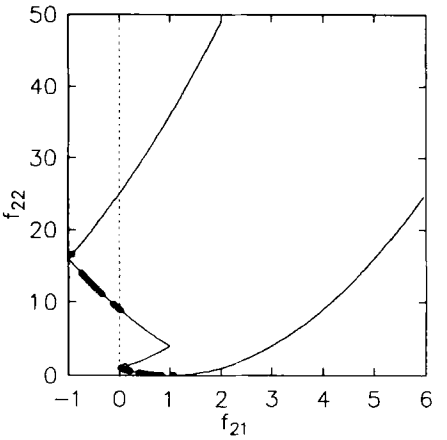


Figure 20. Population at generation 10 obtained using NSGA for problem *F2* is shown.

The unconstrained version of the above problem was solved by Chankong and Haimes (1983) using a goal vector and weights for objectives. But here we make the problem more difficult by constraining the search space.

The first objective function, f_{31}, is a smooth unimodal function that has a minimum at point $x_{31}^* = (2, 1)^T$. The second objective function, f_{32}, decreases monotonically with decreasing x_1 or with increasing absolute value of x_2. A contour plot of these two functions is shown in Figure 29. The contours of the first function are concentric circles with the center at $(2, 1)^T$. This function value increases with increasing diameter of the circle. The second function (parallel parabolas) constantly decreases along the line $x_2 = 1$ toward decreasing x_1. Careful observation reveals that the *tangential points* of circles and parabolas dominate all other points; this occurs because any such tangential point is better in the second objective

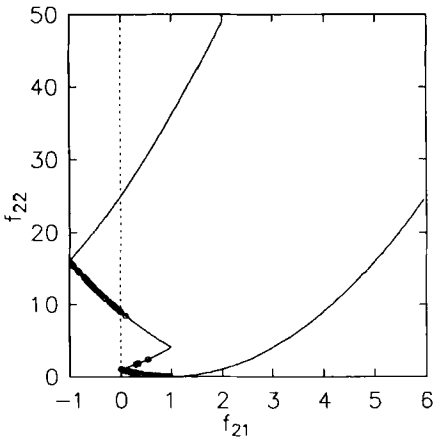


Figure 21. Population at generation 100 obtained using NSGA for problem *F2* is shown.

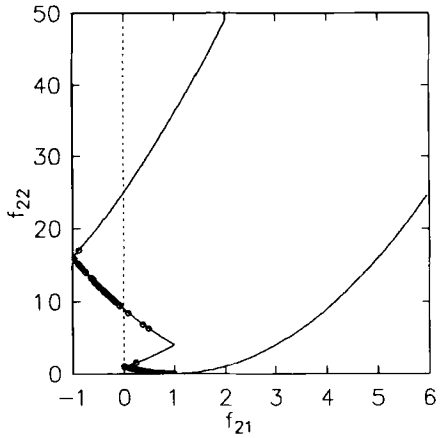


Figure 22. Population at generation 500 obtained using NSGA for problem *F2* is shown.

than all other points belonging to the same circle (same f_{31} value). These tangential points are Pareto-optimal points. Therefore, Pareto-optimal points may be found by equating the slopes (first differentials) of contour curves at common points:

$$\left(\frac{dx_2}{dx_1}\right)_{\text{from } f_{31}} = \left(\frac{dx_2}{dx_1}\right)_{\text{from } f_{32}}$$

or,

$$-\frac{(x_1 - 2)}{(x_2 - 1)} = \frac{9}{2(x_2 - 1)}.$$

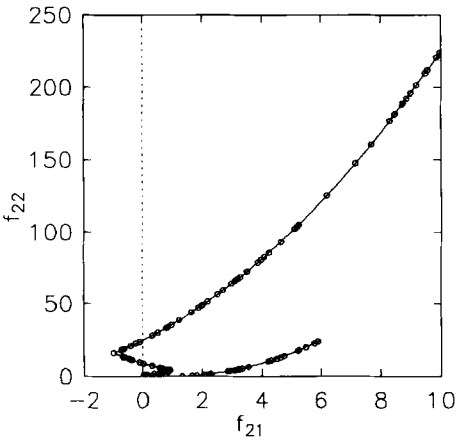


Figure 23. Population at generation 0 obtained using VEGA for problem *F2* is shown.

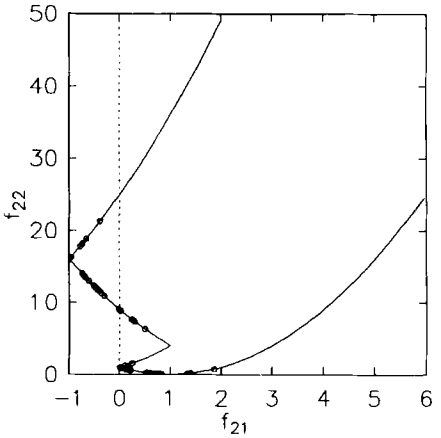


Figure 24. Population at generation 10 obtained using VEGA for problem *F2* is shown.

Assuming $x_2 \neq 1$, and solving yields,

$$x_1 = -2.5.$$

Since the second objective, f_{32}, is monotonically decreasing, the unconstrained Pareto-optimal region, represented by the straight line $x_1 = -2.5$, is unbounded. Because of the constraints, the above unconstrained Pareto-optimal region is now shortened. The constraints make the upper half of the circular region feasible, thereby making the Pareto-optimal region to lie in the region $x_1 = -2.5$, $2.5 \leq x_2 \leq 14.79$, as shown in Figure 29.

NSGA is applied to this problem considering the variable space $-20 \leq x_1, x_2 \leq 20$. A string length of 30 (15-bit string for each variable) is considered. The other parameters such as population size, crossover probability, etc. are kept the same as in the previous experiments.

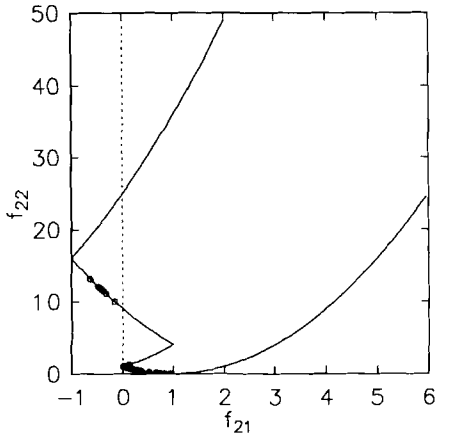


Figure 25. Population at generation 100 obtained using VEGA for problem $F2$ is shown.

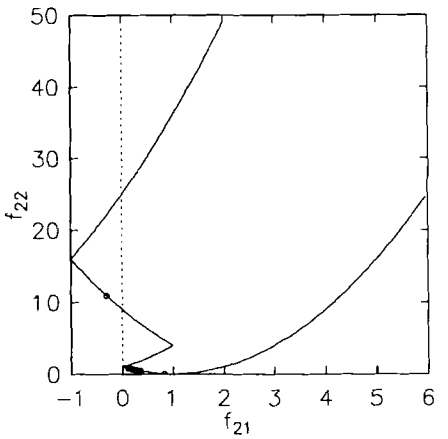


Figure 26. Population at generation 500 obtained using VEGA for problem $F2$ is shown.

This problem is transformed into an unconstrained optimization problem using an exterior penalty function (Rao, 1991). Both the objectives, f_{31} and f_{32}, were penalized if any point lies in the infeasible region. The population movement, in the case of NSGA with $\sigma_{\text{share}} = 9.0$ (for inducing 10 niches in the variable space, calculated using suggestions given elsewhere (Deb, 1989)), is shown in Figures 30 through 33.

It can be observed that at generation zero less than 20% of the population was feasible and by generation 10 the entire population converges to the feasible region. At generation 20, the population starts to move toward the feasible Pareto-optimal line $x_1 = -2.5$. Later on, sharing helps to distribute these nondominated points over the entire Pareto-optimal region. This can be observed at generation 100. Our experiments reveal that this distribution is maintained even at higher generations, observed up to 500.

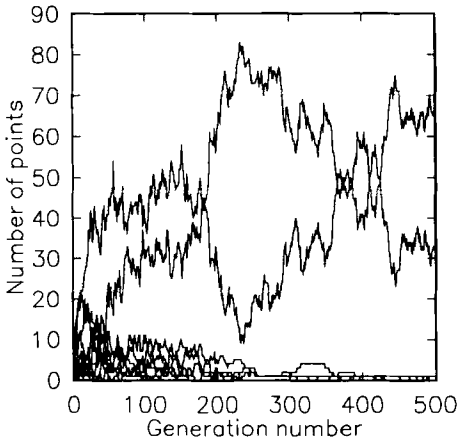


Figure 27. Number of individuals in each subregion versus generation for $F2$ using VEGA is shown.

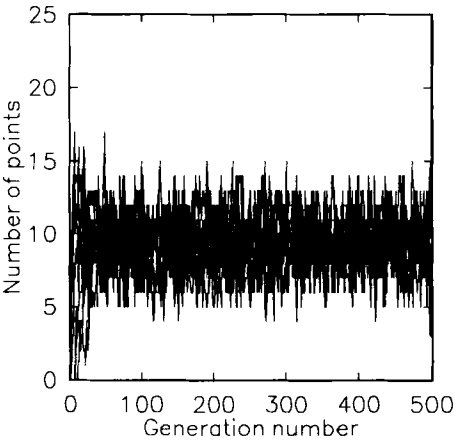


Figure 28. Number of individuals in each subregion versus generation for $F2$ using NSGA is shown.

Figure 34 shows the population distribution at generation 100 obtained using VEGA for problem $F3$. The speciation in the population can be seen in this figure. This experiment has been carried out with the same initial population (at generation 0) as in the case of NSGA (Figure 30). The population movement is similar to that of NSGA during the initial generations up to generation number 20. At generation 100, as shown in Figure 34, the population drifted toward three subregions. Our experiments, observed up to generation 500, revealed that the entire population converged toward two subregions that are nearer to the individual optima of the objectives. This experiment reiterates the distributive ability of NSGA in handling complicated multiobjective problems.

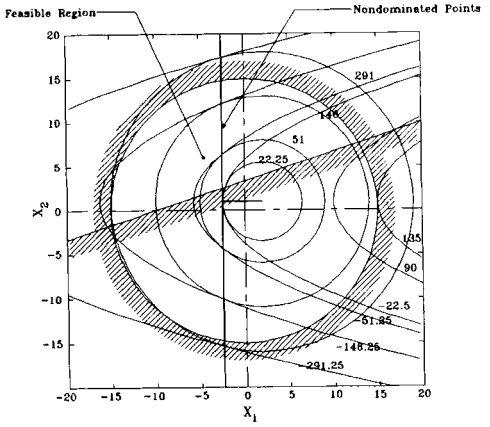


Figure 29. Contour map of problem *F*3.

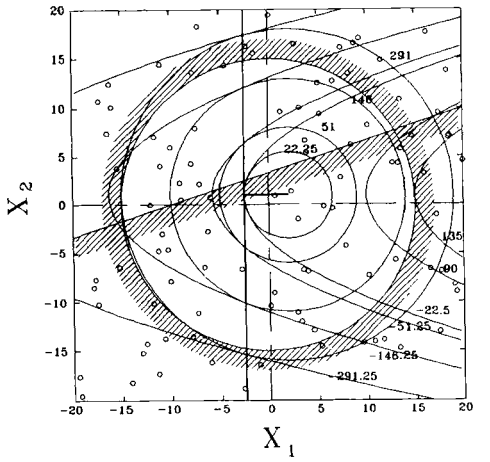


Figure 30. Population at generation 0 obtained using NSGA for problem *F*3 is shown.

6. Extensions

A number of extensions of this study can be pursued:

1. Even though two objectives are used in problems presented in this paper, more objectives can be handled with NSGA. Moreover, the objectives need not be all of the *minimization* type; some of them could be of maximization type. In both situations, the definition of nondominated points will change, but the NSGA algorithm can still be used.

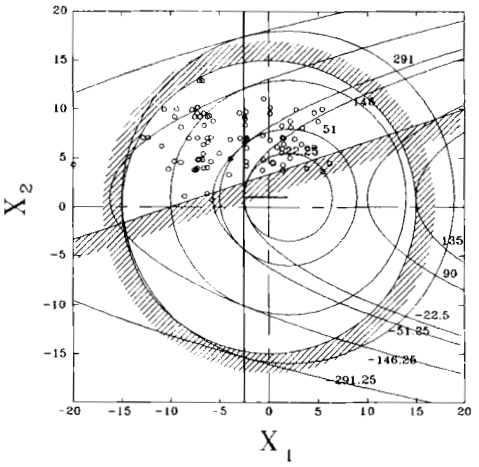


Figure 31. Population at generation 10 obtained using NSGA for problem $F3$ is shown.

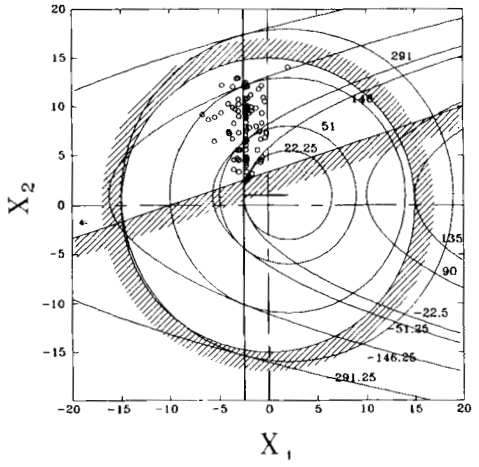


Figure 32. Population at generation 20 obtained using NSGA for problem $F3$ is shown.

2. The other two studies (Fonesca & Fleming, 1993; Horn et al., 1994) stress the importance of sufficient population size and suitable σ_{share} value to yield proper distribution of population. Although the problems presented in this paper used a population size of 100, experiments with a smaller population size of 50 on problems $F1$ and $F2$ have also shown similar performance. The population size requirement may be more for a greater number of objectives, but how this size requirement would increase is a matter for interesting future research.
3. It has been found elsewhere (Goldberg & Deb, 1991) that *tournament* selection puts a more controlled selection pressure and has a faster convergence characteristic than the

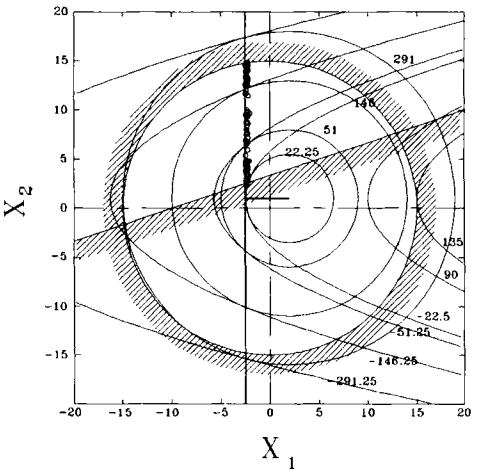


Figure 33. Population at generation 100 obtained using NSGA for problem $F3$ is shown.

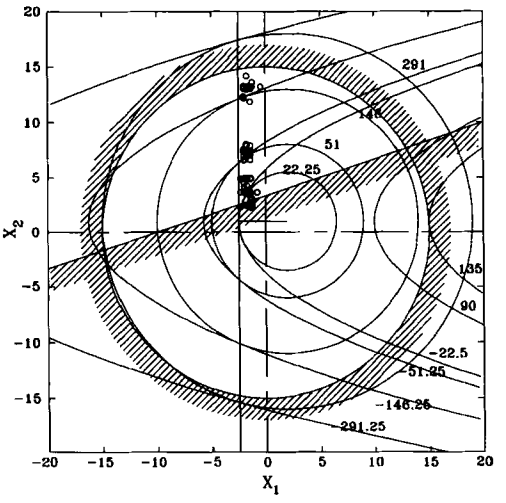


Figure 34. Population at generation 100 obtained using VEGA for problem $F3$ is shown.

proportionate selection method used in this study. The niching technique suggested by Oei, Goldberg, and Chang (1992) can be tried with tournament selection to replace sharing and proportionate selection in NSGA for more controlled and, hopefully, faster solutions.

7. Conclusions

Even though there exists a number of classical multiobjective optimization techniques, they require some a priori problem information. Because genetic algorithms use a population of points, they may be able to find multiple Pareto-optimal solutions simultaneously. Schaffer's Vector Evaluated Genetic Algorithm (VEGA) was one effort in this direction. In this paper, a nondominated sorting genetic algorithm, suggested by Goldberg, is described and used to solve three multiobjective optimization problems. The proof-of-principle simulation results have shown that this algorithm (called NSGA) can maintain stable and uniform reproductive potential across nondominated individuals, which is a serious drawback of VEGA. The results suggest that NSGA can be successfully used to find multiple Pareto-optimal solutions, the knowledge of which could be very useful to designers or decision makers. A number of suggestions for immediate extension and application of NSGA to several multiobjective optimization problems have also been discussed.

References

Chankong, V., & Haimes, Y. Y. (1983). *Multiobjective decision making theory and methodology*. New York: North-Holland.

Deb, K. (1989). *Genetic algorithms in multimodal function optimization* (TCGA Report No. 89002). Tuscaloosa, AL: University of Alabama, The Clearinghouse for Genetic Algorithms.

Deb, K., & Goldberg, D. E. (1991). An investigation of niches and species formation in genetic function optimization. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 42–50). San Mateo, CA: Morgan Kaufmann.

Fonesca, C. M., & Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. In S. Forrest (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms* (pp. 416–423). San Mateo, CA: Morgan Kaufmann.

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

Goldberg, D. E., & Deb, K. (1991). A comparison of selection schemes used in genetic algorithms. In G. J. E. Rawlins (Ed.), *Foundations of Genetic Algorithms* (pp. 69–93). San Mateo, CA: Morgan Kaufmann.

Goldberg, D. E., & Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In J. J. Grefenstette (Ed.), *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms* (pp. 41–49). San Mateo, CA: Morgan Kaufmann.

Hans, A. E. (1988). Multicriteria optimization for highly accurate systems. In W. Stadler (Ed.), *Multicriteria Optimization in Engineering and Sciences*, Mathematical concepts and methods in science and engineering, **19**, 309–352. New York: Plenum Press.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). The niched Pareto genetic algorithm for multiobjective optimization. Unpublished manuscript.

Oei, C. K., Goldberg, D. E., & Chang, S. J. (1992). *Tournament selection, niching, and the preservation of diversity*. IlliGAL Report No. 91011. University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana.

Rao, S. S. (1991). *Optimization theory and application*. New Delhi: Wiley Eastern Limited.

Rosenberg, R. S. (1967). *Simulation of genetic populations with biochemical properties*. Ph.D. dissertation, University of Michigan.

Schaffer, J. D. (1984). *Some experiments in machine learning using vector evaluated genetic algorithms* (TCGA file No. 00314). Ph.D. dissertation, Vanderbilt University, Nashville, TN.

Srinivas, N. (1994). *Multiobjective optimization using nondominated sorting in genetic algorithms.* Unpublished master's thesis, Indian Institute of Technology, Kanpur. India.

Tamura, K., & Miura, S. (1979). Necessary and sufficient conditions for local and global nondominated solutions in decision problems with multiobjectives. *Journal of Optimization Theory and Applications, 27,* 509-523.

Vincent, T. L., & Grantham, G. J. (1981). *Optimality in parametric systems.* New York: John Wiley and Sons.

This article has been cited by:

1. Maoguo Gong, Licheng Jiao, Haifeng Du, Liefeng Bo. 2008. Multiobjective Immune Algorithm with Nondominated Neighbor-Based Selection. *Evolutionary Computation* **16**:2, 225-255. [Abstract] [PDF] [PDF Plus]
2. Elizabeth F. Wanner, Frederico G. Guimarães, Ricardo H. C. Takahashi, Peter J. Fleming. 2008. Local Search with Quadratic Approximations into Memetic Algorithms for Optimization with Multiple Criteria. *Evolutionary Computation* **16**:2, 185-224. [Abstract] [PDF] [PDF Plus]
3. Alfredo G. Hernández-Díaz, Luis V. Santana-Quintero, Carlos A. Coello Coello, Julián Molina. 2007. Pareto-adaptive ε-dominance. *Evolutionary Computation* **15**:4, 493-517. [Abstract] [PDF] [PDF Plus]
4. Edwin D. de Jong. 2007. A Monotonic Archive for Pareto-Coevolution. *Evolutionary Computation* **15**:1, 61-93. [Abstract] [PDF] [PDF Plus]
5. Kalyanmoy Deb , Manikanth Mohan , Shikhar Mishra . 2005. Evaluating the ε-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. *Evolutionary Computation* **13**:4, 501-525. [Abstract] [PDF] [PDF Plus]
6. Edwin D. de Jong , Jordan B. Pollack . 2004. Ideal Evaluation from Coevolution. *Evolutionary Computation* **12**:2, 159-192. [Abstract] [PDF] [PDF Plus]
7. Sanyou Y. Zeng , Lishan S. Kang , Lixin X. Ding . 2004. An Orthogonal Multi-objective Evolutionary Algorithm for Multi-objective Optimization Problems with Constraints. *Evolutionary Computation* **12**:1, 77-98. [Abstract] [PDF] [PDF Plus]
8. Lino Costa , Pedro Oliveira . 2003. An Adaptive Sharing Elitist Evolution Strategy for Multiobjective Optimization. *Evolutionary Computation* **11**:4, 417-438. [Abstract] [PDF] [PDF Plus]
9. Hussein A. Abbass . 2003. Speeding Up Backpropagation Using Multiobjective Evolutionary Algorithms. *Neural Computation* **15**:11, 2705-2726. [Abstract] [PDF] [PDF Plus]
10. Xavier Llorà , David E. Goldberg . 2003. Bounding the Effect of Noise in Multiobjective Learning Classifier Systems. *Evolutionary Computation* **11**:3, 279-298. [Abstract] [PDF] [PDF Plus]
11. Andrea Toffolo , Ernesto Benini . 2003. Genetic Diversity as an Objective in Multi-Objective Evolutionary Algorithms. *Evolutionary Computation* **11**:2, 151-167. [Abstract] [PDF] [PDF Plus]
12. Marco Laumanns , Lothar Thiele , Kalyanmoy Deb , Eckart Zitzler . 2002. Combining Convergence and Diversity in Evolutionary Multiobjective Optimization. *Evolutionary Computation* **10**:3, 263-282. [Abstract] [PDF] [PDF Plus]
13. Rajeev Kumar , Peter Rockett . 2002. Improved Sampling of the Pareto-Front in Multiobjective Genetic Optimizations by Steady-State Evolution: A Pareto Converging Genetic Algorithm. *Evolutionary Computation* **10**:3, 283-314. [Abstract] [PDF] [PDF Plus]
14. Eckart Zitzler , Kalyanmoy Deb , Lothar Thiele . 2000. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* **8**:2, 173-195. [Abstract] [PDF] [PDF Plus]
15. Ian C. Parmee , Dragan Cvetković , Andrew H. Watson , Christopher R. Bonham . 2000. Multiobjective Satisfaction within an Interactive Evolutionary Design Environment. *Evolutionary Computation* **8**:2, 197-222. [Abstract] [PDF] [PDF Plus]
16. David A. Van Veldhuizen , Gary B. Lamont . 2000. Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation* **8**:2, 125-147. [Abstract] [PDF] [PDF Plus]
17. Joshua D. Knowles , David W. Corne . 2000. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation* **8**:2, 149-172. [Abstract] [PDF] [PDF Plus]

18. Carlos M. Fonseca, Peter J. Fleming. 1995. An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation* **3**:1, 1-16. [Abstract] [PDF] [PDF Plus]