Complex Systems **9** (1995) 431–454

Real-coded Genetic Algorithms with Simulated Binary Crossover: Studies on Multimodal and Multiobjective Problems

Kalyanmoy Deb[*]
Amarendra Kumar
Department of Mechanical Engineering,
Indian Institute of Technology,
Kanpur, UP 208 016, India

Abstract. Real-coded genetic algorithms (GAs) do not use any coding of the problem variables, instead they work directly with the variables. The main difference in the implementation of real-coded GAs and binary-coded GAs is in their recombination operators. Although a number of real-coded crossover implementations were suggested, most of them were developed with intuition and without much analysis. Recently, a real-coded crossover operator has been developed based on the search characteristics of the single-point crossover operator used in binary-coded GAs. This simulated binary crossover (SBX) operator has been found to work well in many test problems having continuous search space when compared to existing real-coded crossover implementations. In this paper the performance of the real-coded GA with SBX in solving multimodal and multiobjective problems is further investigated. Sharing function approach and nondominated sorting implementations are included in the real-coded GA with SBX to solve multimodal and multiobjective problems, respectively. It is observed that the real-coded GAs perform equally well or better than binary-coded GAs in solving a number of test problems. One advantage of the SBX operator is that it can restrict children solutions to any arbitrary closeness to the parent solutions, thereby not requiring any separate mating restriction scheme for better performance. Finally, real-coded GAs with SBX have been successfully used to find multiple Pareto-optimal solutions in solving a welded beam design problem. These simulation results are encouraging and suggest the application of real-coded GAs with SBX operator to real-world optimization problems at large.

[*]Electronic mail address: `deb@iitk.ernet.in`.

1. Introduction

With the success of binary-coded genetic algorithms (GAs) in various optimization problems, real-coded GAs are finding some attention primarily in solving continuous search space problems. Real-coded GAs differ from the binary-coded GAs in the coding of the problem variables. Since the problem variables are used directly in real-coded GAs, there lies a need for developing new, yet efficient, crossover and mutation operators. Although there exist a number of studies of real-coded GAs with different crossover and mutation operators [1, 2], recently a crossover and a mutation operator have been developed by simulating the working of their binary counterparts [3]. In that study, the main motivation was to develop real-coded genetic operators having similar *search power* as that in the binary genetic operators. The simulated binary crossover (SBX) operator used in [3] had search power similar to that of a single-point binary-coded crossover operator. The search power was defined as the ability to create any arbitrary child solution from two parent solutions. Based on a derived probability distribution of creating a child solution in the single-point crossover operator, a similar probability distribution was used directly to choose a child solution in SBX. In that study, the performance of the proposed real-coded GAs with SBX was compared with some of the earlier real-coded GAs and the binary-coded GA. Motivated by the success of the proposed real-coded GAs in [3], we extend its application to multimodal and multiobjective function optimization problems.

In this paper we briefly describe the SBX operator and extend the concept of *sharing* in the real-coded GAs to solve a number of multimodal functions for multiple solutions simultaneously. Since children solutions arbitrarily close to the parent solutions can be created using the SBX operator, there is no need to use a separate mating restriction scheme. Thereafter, we extend the principle of nondominated sorting in real-coded GAs to solve a number of multiobjective optimization problems for Pareto-optimal solutions. To show the efficacy of the proposed techniques, we also show simulation results of the presented sharing and nondominated sorting techniques to optimize the engineering design of a welded beam. The successful working of these techniques suggests that the real-coded GA using the SBX operator performs similar to the binary-coded GA and can be used efficiently in solving continuous search space problems.

2. Simulated binary crossover

When problem variables are directly used in a GA, binary-coded crossover operators can no longer be applied. A number of real-coded crossover operators have been developed that create two children solutions from two parent solutions. In most cases, a probability distribution centering the parent solutions is assumed and two children solutions are created based on that probability distribution. Creating children solutions using a fixed probability distribution, which does not depend on the location of the parent solutions, makes the search adaptive. For example, if the parent solutions are close to each other, the children solutions are expected to lie in the neighborhood of the parent solutions. On the other hand, if the parent solutions are far away from each other, children solutions far away from the parent solutions are expected. In early generations of a GA simulation, parent solutions are expected to be away from each other and almost any solution can be created as a child solution. But when the search converges towards a solution, parent solutions become similar and children solutions also become closer to the parent solutions. This adaptiveness in the search power of the real-coded crossover operator is similar in principle to the search power of the binary-coded single-point crossover operator. However, one main difference is that for the binary-coded crossover operator, no explicit probability distribution is used to create a child solution. But there is an implicit probability distribution that depends on the string length used to code the variable. In a real-coded crossover operator, a probability distribution is explicitly used to create a child solution. Since an explicit probability distribution is used, the performance of real-coded GAs depend on that distribution. In earlier real-coded GA implementations, the choice of the probability distribution was somewhat arbitrary and based on intuition. Recently in [3], a SBX operator has been suggested with a probability distribution similar to that in the single-point crossover operator used in binary-coded GAs. In the following, we briefly describe that crossover operator.

It has been observed that in the binary single-point crossover operator, both children solutions lie either inside (contracting crossover) or outside (expanding crossover) the region bounded by the parent solutions. Moreover, the distance of one child solution from one parent is exactly the same as that of the other child from the other parent solution. Further, there is no apparent bias for either contracting or expanding crossover, on expectation. Thus, on an average, the overall probabilities of contracting and expanding crossovers are the same. The SBX operator for real-coded GAs has been developed with the above properties. In order to implement this crossover operator for any two parent solutions p_1 and p_2, a nondimensionalized spread factor β has been defined as the ratio of the spread of created children solutions c_1 and c_2 to that of the parent solutions as follows:

$$\beta = \left| \frac{c_1 - c_2}{p_1 - p_2} \right| . \tag{1}$$

It can be shown that $\beta \leq 1$ corresponds to a contracting crossover and $\beta \geq 1$ corresponds to an expanding crossover. Writing the decoded values of two arbitrary parent strings in terms of their allele values (0,1) and writing the children strings created from the parent strings as a function of the cross-site k along the string length, the spread factor β has been written in terms of k. From this distribution, it has also been possible to calculate the probability of creating a pair of children solutions having a certain β. That probability distribution has been approximated by a polynomial probability distribution

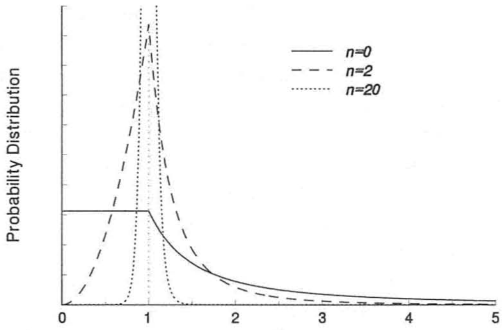

Figure 1: Probability distributions used in the simulated binary crossover operator are shown for different values of the distribution index n.

as follows:

$$\mathcal{P}(\beta) = \begin{cases} 0.5(n+1)\beta^n, & \text{if } \beta \leq 1; \\ 0.5(n+1)\frac{1}{\beta^{n+2}}, & \text{otherwise.} \end{cases} \tag{2}$$

In equations (1) and (2), the distribution index n is any nonnegative real number. A large value of n gives a higher probability for creating solutions near to the parent and a small value of n allows distant points to be selected as children solutions. Figure 1 shows the probability distribution as a function of β for different values of the distribution index n. The advantage of using a probability distribution as a function of β is that the created children solutions are relative to the parent solutions. If the parent solutions are distant, children solutions far away from the parent solutions can be created. On the other hand, if the parent solutions are close to each other, the children solutions, in general, cannot be far away from the parent solutions.

In order to create two children solutions c_1 and c_2 from the parent solutions p_1 and p_2 using the above probability distribution, the following procedure is used.

- Create a random number u between 0 and 1.
- Find a β' for which the cumulative probability

$$\int_0^{\beta'} \mathcal{P}(\beta) d\beta = u. \tag{3}$$

- Knowing the value of β', the children points are calculated as

$$\begin{aligned} c_1 &= 0.5\left[(p_1+p_2) - \beta'|p_2 - p_1|\right], \\ c_2 &= 0.5\left[(p_1+p_2) + \beta'|p_2 - p_1|\right]. \end{aligned}$$

The preceding SBX operator is allowed to create any solution in the entire real space $[-\infty, \infty]$, however, it can also be modified for variables with known lower and upper bounds ($x_i^{(L)} \leq x_i \leq x_i^{(U)}$). This can be achieved by modifying the probability distribution (equation (2)) so that the probability for a solution outside the above bounds is zero. A simple way to achieve this is to first calculate the cumulative probabilities

$$\mathcal{P}'_1 = \int_0^{\beta^{(L)}} \mathcal{P}(\beta) d\beta, \quad (4)$$

$$\mathcal{P}'_2 = \int_0^{\beta^{(U)}} \mathcal{P}(\beta) d\beta. \quad (5)$$

In equations (4) and (5), the parameters $\beta^{(L)}$ and $\beta^{(U)}$ are the spread factors for the lower and upper bounds of the problem variable, respectively:

$$\beta^L = \frac{p_1 + p_2 - 2x_i^{(L)}}{|p_2 - p_1|}, \qquad \beta^{(U)} = \frac{2x_i^{(U)} - p_1 - p_2}{|p_2 - p_1|}.$$

Thereafter, two spread factors β'_1 and β'_2 are calculated using modified probability distributions $\mathcal{P}(\beta)/\mathcal{P}'_1$ and $\mathcal{P}(\beta)/\mathcal{P}'_2$, respectively, (by using equation (3)) to create two children solutions as follows:

$$c_1 = 0.5\left[(p_1 + p_2) - \beta'_1|p_2 - p_1|\right],$$
$$c_2 = 0.5\left[(p_1 + p_2) + \beta'_2|p_2 - p_1|\right].$$

Thus, the modified probability distributions do not create any solution outside the given bounds, instead they scale up the probability for solutions inside the bounds, as shown by the solid line in Figure 2. In the figure, the parent solutions are p_1 and p_2. The probability distributions for the unbounded case ($x \in [-\infty, \infty]$) and the bounded case ($x \in (x^{(L)}, x^{(U)})$) are shown in dashed and solid lines, respectively. It can be observed in the fig-

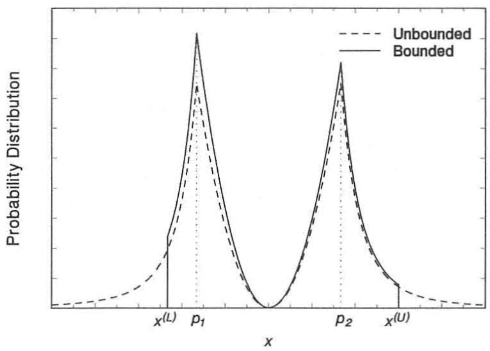

Figure 2: Probability distributions are shown for bounded and unbounded cases.

ure that a solution outside the lower and upper bounds has zero probability to become a child solution, thereby restricting the search within the given bounds.

The focus of this paper is to study the performance of the real-coded GA with SBX on multimodal and multiobjective problems. The application of binary-coded GAs for the solution of these problems is finding increasing attention. This is because the solution of such problems requires multiple solutions to be found simultaneously and GAs are particularly suitable for these problems because of their inherent population approach. In the following, we first investigate the application of real-coded GAs to multimodal problems and then study the multiobjective problems.

3. Multimodal problems using sharing genetic algorithms

A multimodal problem contains multiple optimal solutions in its search space. The objective of a multimodal function optimization procedure is to find multiple optimal solutions having either equal or unequal objective function values. The knowledge of multiple optimal solutions is particularly useful to design engineers for choosing an alternative optimal solution, as and when required. In solving for multiple optimal solutions, traditional optimization algorithms need to be applied as many times as the number of optimal solutions. This is because most of those algorithms are point-by-point search methods and can only find one optimal solution at a time. Since GAs create a population of solutions instead of one solution in each iteration, it may be possible to capture multiple optimal solutions in the population, thereby allowing GAs to be applied only once to find multiple solutions. After the pioneering work in [4], researchers have also developed different GAs to find and maintain a stable subpopulation of optimal solutions in the population (e.g., [5, 6]).

In solving multimodal problems using GAs, the study in [4] was motivated by a "gedanken" experiment based on a modified two-armed bandit problem. It was observed that if the reproduction phase is performed with a modified fitness function obtained by degrading the original fitness value of a solution by a cumulative measure of the proximity between that solution and the rest of the population, a stable subpopulation can be maintained in the population. They defined a proximity measure (d_{ij}) between any two solutions i and j either phenotypically (with the problem variables directly) or genotypically (with corresponding strings). A sharing function $Sh(d_{ij})$ is used to define the sharing effect of one solution to another based on the proximity measure of two solutions as follows:

$$Sh(d_{ij}) = \begin{cases} 1 - \frac{d_{ij}}{\sigma}, & \text{if } d_{ij} \geq \sigma; \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

The parameter σ is the maximum distance between two solutions that can belong to one optimal solution. Thereafter, the cumulative proximity (m'_i) of the ith solution in the entire population (of size N) is calculated by summing the individual sharing function values as follows:

$$m_i' = \sum_{j=1}^{N} Sh(d_{ij}). \tag{7}$$

The reproduction is then performed with the shared fitness $f_i' = f_i/m_i$ instead of the original fitness f_i. This sharing technique has been successfully applied to many multimodal problems [5, 6].

We implement the stated sharing function concept in real-coded GAs. It is important to note that the implementation of sharing is only in the reproduction phase; the crossover and mutation operators need not be changed. The primary differences in the implementation of binary-coded and real-coded GAs are in the coding and in the recombination operators. Thus, the concept of sharing can also be used in real-coded GAs. Since, in real-coded GAs the problem variables are used directly, phenotypic sharing is a natural choice. To calculate the proximity measure d_{ij} and the parameter σ, the following guidelines were suggested in [5] and are used here:

$$d_{ij} = \sqrt{\sum_{k=1}^{p} \left(x_k^{(i)} - x_k^{(j)}\right)^2}, \tag{8}$$

$$\sigma = \frac{1}{2\sqrt[p]{q}}\sqrt{\sum_{k=1}^{p} \left(x_k^{(U)} - x_k^{(L)}\right)^2}, \tag{9}$$

where p is the number of variables and q is the desired number of optimal solutions.

3.1 Simulation results

The real-coded GA with SBX operator and the shared reproduction scheme have been implemented to solve a number of test problems obtained from the literature and also to a number of random bimodal functions. The performance of real-coded GAs has been compared with binary-coded GAs on all test functions.

In solving multimodal problems using GAs, the primary objective has been not only to find all optimal solutions but also to distribute the population members well among multiple optimal solutions [5]. To compare the performance of different GA implementations, a deviation measure similar to chi-square was suggested in [5]. This measure calculates the deviation of the distribution of the population from an ideal distribution, which can be obtained using the modified two-armed bandit gedanken experiment from [4]. The chi-square deviation measure is given as

$$\Psi = \sqrt{\sum_{k=1}^{q+1} \left(\frac{r_k - \bar{r}_k}{\hat{r}_k}\right)^2}, \tag{10}$$

where $\bar{r}_k$ and $\hat{r}_k$ are the expected value and standard deviation of the number of ideal solutions in the kth optimal solution in the population, and r_k is the actual number of the solutions in the kth optimal solution in the population. In all our simulations, all solutions having a fitness greater than or equal to 70 percent of the globally best fitness are counted as the number of solutions (r_k) near the corresponding optimum. However, the suggested values of the parameters $\bar{r}_k$ and $\hat{r}_k$ of the ideal distribution for q optimal solutions having function values f_k, $k = 1, 2, \ldots, q$ are as follows [5]:

$$\begin{aligned}\bar{r}_k &= \frac{f_k}{\sum_{k=1}^{q} f_k} N,\\ \hat{r}_k &= \sqrt{\bar{r}_k\left(1 - \frac{\bar{r}_k}{N}\right)}.\end{aligned}$$

For nonpeak solutions, $\bar{r}_{q+1} = 0$ and $\hat{r}_{q+1} = \sqrt{\sum_{k=1}^{q} \hat{r}_k^2}$. Equation (10) suggests that smaller deviation measures Ψ yield better distribution of the population on the optimal solutions.

3.1.1 Five test functions

Five test functions, which were used in [5], are solved using real-coded GAs. The functions, their domain, and the description of the optimal solutions are given as follows.

MM1:	$\sin^6(5\pi x)$	$0 \le x \le 1$
	Five equispaced maxima with equal function values.	
MM2:	$\exp\left(-2\ln 2\left(\frac{x-0.1}{0.8}\right)^2\right)\sin^6(5\pi x)$	$0 \le x \le 1$
	Five equispaced maxima with unequal function values.	
MM3:	$\sin^6(5\pi(x^{0.75} - 0.05))$	$0 \le x \le 1$
	Five unequispaced maxima with equal function values.	
MM4:	$\exp\left(-2\ln 2\left(\frac{x-0.1}{0.8}\right)^2\right)\sin^6(5\pi(x^{0.75} - 0.05))$	$0 \le x \le 1$
	Five unequispaced maxima with unequal function values.	
MM5:	$\left[1 - \left((x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2\right)/2186\right]$	$-6 \le x_1, x_2 \le 6$
	Four maxima with equal function values.	

The simulation results are compared with binary-coded GAs having a single-point crossover operator. The following GA parameters are used in all simulations.

Population size N:	100	
String length ℓ:	30	(for binary-coded GAs)
Crossover probability:	0.9	
Mutation probability:	0	
Distribution index n:	0–500	(for real-coded GAs).

Figure 3 shows the deviation measure Ψ versus the distribution index n on the function MM1. In order to investigate the effect of n on the performance of the real-coded GAs, we have varied n from 0 to 500. Average deviation measures for generations 101 to 200 are plotted to demonstrate that the algorithms not only distribute their populations well among all the optimal solutions, but also maintain the distribution for a large number of generations. Figure 3 reveals that for small values of n, the SBX operator is unable to maintain stable subpopulations on all five optimal solutions. The horizontal dashed line is the deviation measure obtained for the binary-coded GAs with single-point crossover on this function. Figure 3 also shows that real-coded GAs perform better than binary-coded GAs when n is somewhat larger than 30. The distribution of population members is further illustrated in Figure 4, where the deviation measure is plotted with generation number. Figure 4 shows that the real-coded GAs with $n = 35$ and binary-coded GAs perform more or less the same and that the real-coded GAs with $n = 200$ perform much better than the binary-coded GAs. The reason a comparatively larger value of n is required to adequately solve the problem can be explained as follows. The deviation measure described in equation (10) becomes better if the population is well distributed among the optimal solutions and the number of lethal individuals (solutions not belonging to any optimal solution) is low. A little thought will reveal that if two individuals belonging to two different optimal solutions are allowed to cross over, lethal solutions may

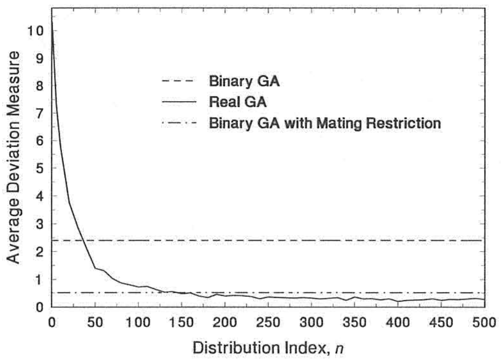

Figure 3: Average deviation measure (for generations 101 to 200) is plotted with distribution index n for function MM1.

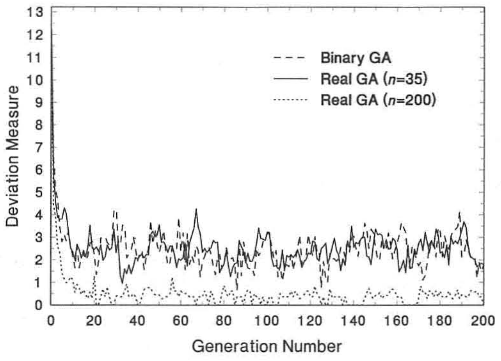

Figure 4: Deviation measure is plotted with generation number for function MM1.

be created. Under SBX, this possibility can be minimized by suitably choosing a distribution index, n. In function MM1, the two extreme peaks are at $x = 0.1$ and $x = 0.9$, respectively. Since solutions having a function value greater than 0.7 are considered to belong to the niche of an optimal solution, the difference between a child solution from the closest parent solution must be at most equal to 0.022. Thus, for contracting crossovers, the minimum spread factor must be equal to $\beta = 0.945$. If we want to succeed in 99 percent of crossovers, the corresponding n can be approximately calculated by equating the cumulative probability of success to 0.99, as follows:

$$1 - \beta^{n+1} = 0.99.$$

For the two extreme optimal solutions, the above equation demands $n \approx 80$. For the two nearest optimal solutions ($x = 0.1$ and $x = 0.3$) the required distribution index is $n \approx 18$. The value of $n = 35$ observed in Figure 3 is a compromise between these two values of n.

In order to improve the performance of binary-coded GAs with sharing, oftentimes, a mating restriction scheme [5] is used. In a mating restriction scheme, similar solutions are only allowed to mate with each other, thereby disallowing the creation of lethal solutions. The SBX operator, on the other hand, can restrict children solutions to lie in the vicinity of the parent solutions by using a large value of the distribution index n. Thus, the mating restriction scheme may not be necessary with the SBX operator to improve performance (either on-line performance as defined in [7] or our deviation measure Ψ) of GAs. In Figure 3, the dashed-dot line shows the deviation measure of the binary-coded GAs with sharing and mating restriction. It can be seen in the figure that the real-coded GAs with n higher than about 150 can achieve similar or better performance than binary-coded GAs with sharing and mating restriction schemes together.

Similar experiments are performed using the other four test functions and similar results are obtained. Figures similar to Figures 3 and 4 are presented in the Appendix for the functions MM2 through MM5.

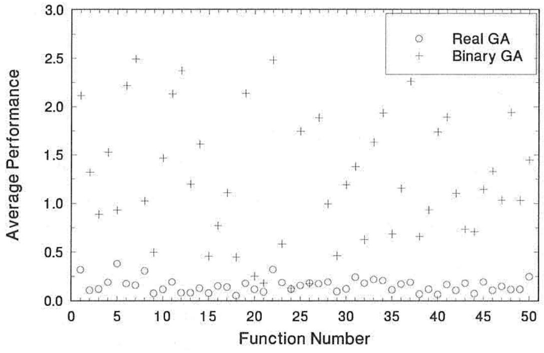

Figure 5: Average deviation measure (from generations 101 to 200) of real-coded GAs and binary-coded GAs are plotted for 50 different bimodal functions.

3.1.2 Random bimodal function

To further compare the performance of real-coded GAs with SBX and binary-coded GAs, we have created 50 functions having two peaks located at any two random points in the range (0,1). The functional form of the two-peaked function (with optimal solutions close to x_1 and x_2) is as follows:

$$\text{MM6:} \quad \frac{\left[\exp\left(-\frac{(x-x_1)^2}{2b^2}\right) + \exp\left(-\frac{(x-x_2)^2}{2b^2}\right)\right]}{\left[1 + \exp\left(-\frac{(x_1-x_2)^2}{2b^2}\right)\right]}, \qquad 0 \le x \le 1.$$

The parameter b acts like the spread of the optimal solutions. We use MM6 with 50 different random combinations of x_1, $x_2 \in (0,1)$, and $b \in (0, 0.05)$ to create 50 bimodal functions.

Binary-coded and real-coded GAs are used to find both optimal solutions in the same 50 random functions with GA parameters as mentioned earlier. The average of the deviation measure Ψ of the simulation runs from generations 101 to 200 is calculated in each case and shown plotted versus the function number in Figure 5. The figure shows that the performance of real-coded GAs with SBX ($n = 200$) is consistently better than that of the binary-coded GAs. This is expected because in real-coded GAs the search power can be controlled using the distribution index, n. With a large value of n, the real-coded GA with SBX behaves like a binary-coded GA with sharing and mating restriction. With large values of n, the search is always confined in the vicinity of the parent solutions. Thus, the solutions improve marginally in each generation and the combined effect of shared reproduction and recombination operators guide the search parallely towards each optimal solution. Since, in the above simulation of binary-coded GAs the mating restriction scheme is not used, there is no restriction for the solutions from different optima to mate with each other. These crossovers sometimes create some nonoptimal solutions, which deteriorate the performance of binary-coded GAs.

3.2 Fractional sharing

One criticism against the implementation of sharing strategy in GAs is that for every individual in the population the sharing function needs to be calculated for every other individual [4, 8]. This requires N^2 evaluations of sharing functions at each generation. However, if proper book-keeping is maintained, this complexity can be reduced to half because the sharing function is commutative (i.e., $Sh(d_{ij}) = Sh(d_{ji})$). Although it is suggested in [4], and later in [16], to use a smaller subpopulation (of $O(N)$) for calculating the cumulative proximity, no simulation results have been reported to the best of our knowledge. In this section, we investigate the effect of using a fractional population for calculating the cumulative proximity measure.

For each individual in the population, a small subpopulation P_η of size η (share size) is chosen at random and sharing function values are calculated for each individual in this subpopulation. Sharing strategy remains the same as before except equation (7) is now replaced by the following:

$$m'_i = \sum_{\substack{j=1 \\ j \in P_\eta}}^{\eta} Sh(d_{ij}). \tag{11}$$

It is obvious that if η is equal to the population size, equation (11) is identical to equation (7).

In order to study the effect of η in the performance of sharing GAs, we apply both binary-coded and real-coded GAs to test functions MM1 through MM5 with different values of η. The distribution index n is kept the same as that used in earlier simulations (i.e., with $\eta = N$). Figure 6 shows the variation of the average deviation measure Ψ of GAs in generations 401 to 500 with η (which is also the percentage measure of share size to the population

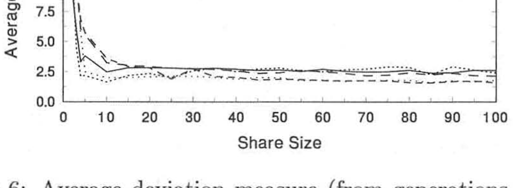

Figure 6: Average deviation measure (from generations 401 to 500) of real-coded GAs is plotted versus share size η.

size). Figure 6 reveals that beyond a critical value of η (much smaller than the complete population size), the performance of GAs are similar to that of $\eta = N$. For example, this critical value is only five percent of the population size in the case of function MM1, 15 percent in the case of function MM2, and ten percent for MM3, MM4, and MM5.

Similar performance is observed with binary-coded GAs in [17]. These experimental results show that only a small fraction of the population size is sufficient to calculate the cumulative proximity for properly distributing the population to all optimal solutions.

4. Multiobjective problems using nondominated sorting genetic algorithms

In a multiobjective optimization problem, there is more than one objective function, all of which are to be optimized simultaneously. Traditionally, the practice is to convert multiple objectives into one objective function (usually a weighted average of the objectives is used) and then treat the problem as a single objective optimization problem. Unfortunately, this technique is subjective to the user, with the optimal solution being dependent on the chosen weight vector. In fact, the solutions of the multiobjective optimization problem can be thought of as a collection of optimal solutions obtained by solving different single objective functions that are formed by using different weight vectors. These solutions are known as Pareto-optimal solutions.

In order to find a number of Pareto-optimal solutions, different extensions of binary-coded GAs have been tried (e.g., [9, 10, 11]). Because of their population approach, GAs are idle candidates to solve this problem. In one implementation of binary-coded GAs, the concept of nondominated sorting of population members is successfully used to solve some test problems [11] and a number of truss-structure optimization problems [12]. We briefly describe that method and show how the same concept can be used in real-coded GAs with the SBX operator.

GAs require only one fitness value for an individual solution in the population. Thus, an artificial fitness value must be assigned to each solution in the population depending on the comparative values of each objective function. In order to assign a fitness measure to each solution, in [11] the idea of nondomination among population members from [13] is used. In a population, the nondominated solutions are defined as those solutions which are better in at least one objective than any other solution in the population. In order to implement such a nondominated sorting concept, the following procedure is adopted.

- The population is sorted to find the nondomination set of solutions. All individuals in this subpopulation are assigned a large artificial fitness value.
- Since the objective is to find a number of Pareto-optimal solutions, sharing is performed among these nondominated solutions and a new shared fitness is calculated for each.

- These solutions are temporarily counted out of the population and the next nondominated set is found. These solutions are assigned an artificial fitness value marginally smaller than the least shared fitness value in the previous nondominated set. This is done to impose a higher preference for solutions in the previous set than for the current set.
- Sharing is performed again among the new nondominated set and this process continues until all population members are ranked in descending order of the nondominated sets.
- Thereafter, the reproduction operation is performed using these artificial fitness values.
- Crossover and mutation operators are applied as usual.

The preceding procedure is implemented with real-coded GAs and sharing is performed phenotypically. The performance of real-coded GAs is compared with binary-coded GAs in solving three test functions used in an earlier study [11].

4.1 Simulation results

All of the test functions have two objective functions to be minimized, although the preceding procedure can be used for more than two objectives. Moreover, the nondominated sorting GAs do not restrict the objectives to be of a minimization type only. A combination of minimization and maximization objective functions can also be handled equally efficiently. This feature of nondominated sorting GAs makes them attractive in solving multiobjective problems. In each of the following three test problems, the objective is to find multiple Pareto-optimal solutions.

$$\text{MO1} \quad \begin{cases} \text{minimize} & x^2 \\ \text{minimize} & (x-2)^2 \end{cases} \qquad \begin{array}{l} \text{Pareto solution:} \\ 0 \le x \le 2. \end{array}$$

$$\text{MO2} \quad \begin{cases} \text{minimize} & \begin{cases} -x & \text{if } x \le 1 \\ -2+x & \text{if } 1 < x \le 3 \\ 4-x & \text{if } 3 < x \le 4 \\ -4+x & \text{if } x > 4 \end{cases} \\ \text{minimize} & (x-5)^2 \end{cases} \qquad \begin{array}{l} \text{Pareto solution:} \\ 1 \le x \le 2 \text{ and } 4 \le x \le 5. \end{array}$$

$$\text{MO3} \quad \begin{cases} \text{minimize} & 2+(x_1-2)^2+(x_2-1)^2 \\ \text{minimize} & 9x_1-(x_2-1)^2 \end{cases} \qquad \begin{array}{l} \text{Pareto solution:} \\ x_1 = -2.5. \end{array}$$

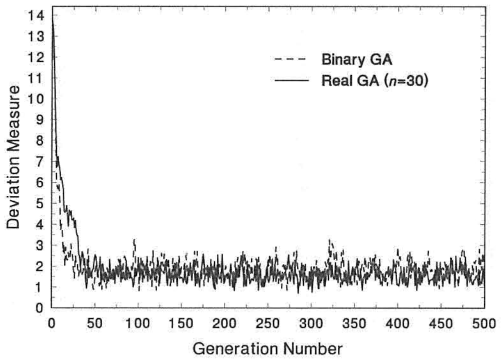

Figure 7: The deviation measures of real-coded GAs and binary-coded GAs are plotted versus generation number for multiobjective problem MO1.

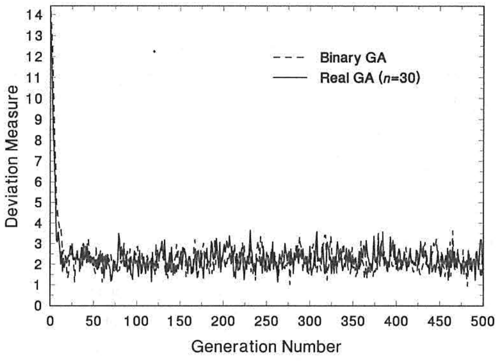

Figure 8: The deviation measures of real-coded GAs and binary-coded GAs are plotted versus generation number for the multiobjective problem MO2.

The same GA parameters used in the multimodal study are chosen here. The chi-square like deviation measure defined in equation (10) is used to judge the working of the algorithms. For the test functions MO1 and MO2, the deviation measures are computed for ten equal intervals ($q = 10$) in the Pareto-optimal set. The performance of both real-coded GAs with SBX and binary-coded GAs on problems MO1 and MO2 are shown in Figures 7 and 8, respectively. From both figures it can be observed that the performance of real-coded GAs is similar to that of binary-coded GAs.

The real-coded GA has also been able to find a number of multiple Pareto-optimal solutions in MO3. As shown in [11], the Pareto-optimal set for this problem is the line $x_1 = -2.5$. Figure 9 shows that all population members, even in generation 500, are distributed along the Pareto-optimal line. Similar results were found for binary-coded GAs in [11].

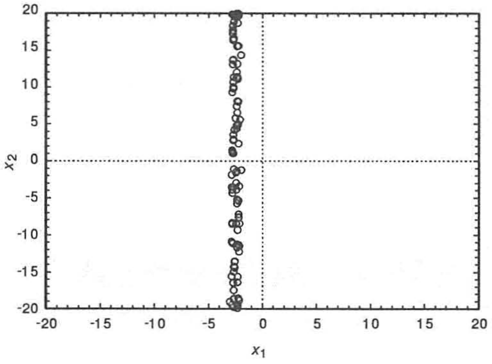

Figure 9: The population of solutions at generation 500 obtained using the real-coded GAs is shown to have found the Pareto-optimal line in problem MO3.

These simulation results suggest that, similar to the nondominated sorting binary-coded GAs, the nondominated sorting principle can be used in real-coded GAs equally efficiently. In order to further test the efficacy of the above technique, the real-coded GAs with sharing and nondominated sorting are applied to the engineering design of a welded beam.

5. Welded beam problem

A beam of rectangular cross-section is welded to a member to carry a certain load (Figure 10). The welded beam problem is a popular engineering design problem where the objective is to find a set of four variables (h, ℓ, t, and b) such that the cost of fabrication of the welded beam is minimum, subject to satisfying a number of constraints [14]. In the original welded beam problem there are five constraints.

1. The bending stress anywhere in the welded beam is limited to the allowable strength of the beam material.
2. The shear stress in the weld is limited to the allowable strength of the weld.
3. Maximum buckling load that can be carried by the plate is caused only by the applied load.
4. The deflection of the end of the beam is limited to a maximum permissible deflection.
5. The weld size must be smaller than the thickness of the beam.

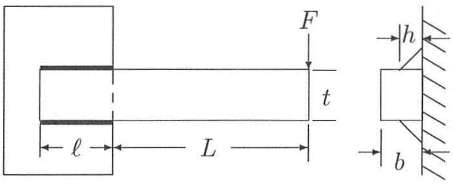

Figure 10: The welded beam problem.

The single-objective, nonlinear optimization problem can be written as follows:

$$\begin{aligned} \text{Minimize cost subject to: } \quad g_1(x) &= S_{yt} - \sigma(x) \geq 0, \\ g_2(x) &= S_{sy} - \tau(x) \geq 0, \\ g_3(x) &= P_c(x) - F \geq 0, \\ g_4(x) &= \delta_{\max} - \delta(x) \geq 0, \\ g_5(x) &= b - h \geq 0. \end{aligned} \tag{12}$$

For brevity, the expressions for the preceding constraints are not given here. They can be found elsewhere [14, 15].

5.1 Multiobjective welded beam design

We convert constraint $g_4(x)$ to an additional objective function $\delta(x)$. Thus, the welded beam problem now has the two objectives of both minimizing cost and end-deflection of the beam. There are now four constraints (g_1, g_2, g_3, and g_5). We use the nondominated sorting GAs discussed in section 4 to solve this problem.

At first, the variables are all assumed to take continuous values. Thus, we use nondominated sorting real-coded GAs. A population size of 100, a crossover probability of 1.0, and a mutation probability of 0.0 are used. A stochastic remainder roulette-wheel selection scheme is used. A distribution index n of 30 is used. The Pareto-optimal solutions found after 500 generations are presented in Figure 11. Although the Pareto-optimal solutions are found within the first 50 generations, the simulation is prolonged for 500 generations to determine whether the GA can maintain a stable set of solutions long after they are initially discovered. Figure 11 shows that the GA has been able to find many Pareto-optimal solutions in one simulation run. The solution set contains a design with a low cost of about \$3.90, having a deflection of about 0.005 inches. The corresponding design variables are $h = 0.423$, $\ell = 2.457$, $t = 9.982$, and $b = 0.433$ inches. The solution set also contains a costly design with a cost of about \$40, having a low deflection of only 0.0005 inches. The corresponding design variables are $h = 0.426$, $\ell = 2.466$, $t = 9.981$, and $b = 4.921$ inches. The other solutions in the solution set reveal that the Pareto-optimal solutions vary only in the value of the variable b. However, neither of the two solutions mentioned above (or any

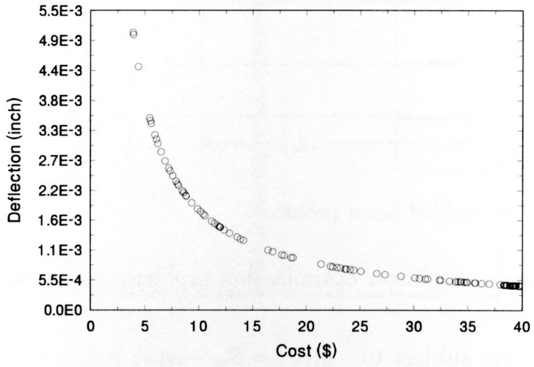

Figure 11: The population of solutions (Pareto-optimal solutions) at generation 500 obtained using the real-coded GAs for the welded beam problem.

other Pareto-optimal solutions found in the solution set) can be considered as the absolute best solution in terms of both objectives. In some applications, cost could be the major factor and the former solution may seem to be better, whereas in applications where rigidity is the major concern, the latter solution is better. In an engineering design, either the cost or the deflection alone may not be the only desired criteria, a combination of both is sometimes a compromised solution. Finding a number of such optimal solutions simultaneously provides the designer flexibility in choosing a suitable solution and also helps provide more insight to the complex interaction of conflicting objectives governing the design. Traditional methods are handicapped in this aspect, since they are expected to find only one solution in one simulation run.

We further applied the nondominated sorting binary-coded GAs to solve the same problem with identical GA parameters using 10-bit string coding for each of the four variables. Figure 12 shows the Pareto-optimal solutions found at the end of 500 generations. It is clear that this GA has also been able to find multiple Pareto-optimal solutions. However, the range of solutions obtained using this method is not as wide as that obtained using real-coded GAs. Although a number of other Pareto-optimal solutions were found early in the simulation, this GA could not maintain some of the high-cost, low-deflection solutions. The concept of sharing with real-coded representation seems to be able to spread solutions better on different optimal solutions than that with binary string-coded representation.

6. Conclusion

A simulated binary crossover (SBX) operator was developed for continuous search space problems in an earlier study [3]. The SBX operator in [3] was applied directly to problem variables, constituting a search similar to a single-

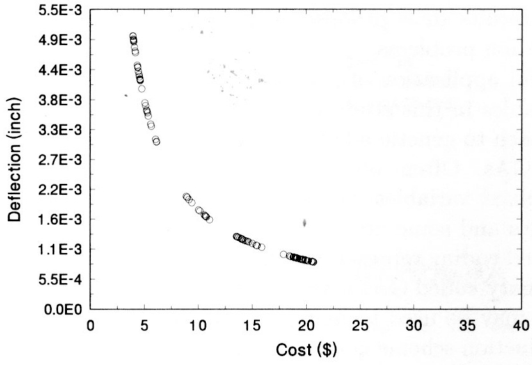

Figure 12: The population of solutions (Pareto-optimal solutions) at generation 500 obtained using the binary-coded GAs for the welded beam problem.

point crossover when applied to binary-coded strings representing problem variables. In this paper, the real-coded GA with SBX operator has been extended to solve multimodal and multiobjective optimization problems.

In the case of multimodal problems, sharing functions have been used to maintain stable subpopulations around multiple optimum solutions. In a number of test functions, the real-coded GA with sharing has performed similar to or better than the binary-coded GA with sharing. It has also been observed that a mating restriction scheme, often used in binary-coded GAs for better performance, may not be used in real-coded GAs; yet a similar performance can be achieved by using a larger distribution index, n. By solving 50 different random bimodal problems, it has been observed that the real-coded GA with SBX can distribute the population members on both optima better than the binary-coded GA. Moreover, the age-old criticism about the full population size requirement of the sharing function method has been disregarded. It has been observed that a small sample of the population members (as large as 15 percent of the population size) can be used to calculate the niche count and still maintain stable subpopulations across the multiple optima.

In the case of multiobjective problems, the concept of nondominated sorting has been implemented in the real-coded GA with SBX operator. Simulation results on three test problems have shown that the real-coded GA can also find and maintain multiple Pareto-optimal solutions in the population. This technique has been further tested to design a welded beam problem. Multiple Pareto-optimal solutions corresponding to minimization of both the cost and deflection of the beam are found using the real-coded GA with SBX operator.

This paper has demonstrated that in solving two different kinds of optimization problems having continuous search space, the real-coded GA with SBX operator applied to direct problem variables can be used efficiently.

The simulation results show promise in the use of these techniques to other similar optimization problems.

The successful application of real-coded GAs with the SBX operator on continuous variables in this study and in [3] suggests the development of a combined approach to genetic adaptive search using both binary-coded GAs and real-coded GAs. Often, optimization problems in engineering and sciences involve mixed variables where some are discrete variables including zero-one variables and some are continuous variables. In the combined GA approach, a mixed coding representing discrete and continuous variables may be used. The binary-coded GAs may be used to handle discrete variables and real-coded GAs may be used to handle continuous variables. Although any standard reproduction scheme can be used, the recombination operators will depend on the underlying variable. The binary single-point crossover or the simulated binary crossover may be used depending on whether the variable being crossed is discrete or continuous, respectively. A similar consideration can be made with the mutation operator. Such a GA will allow only the feasible values of the variables to be searched, thereby reducing the search effort required to find the optimal solution. This GA will constitute a robust and flexible search technique that can be used to solve mixed-variable optimization problems effectively [3].

Acknowledgment

This work is funded by the Department of Science and Technology, New Delhi under grant SR/SY/E-06/93. The authors acknowledge the programming help of Ram Bhushan Agrawal.

Appendix

Deviation measure plots for test functions MM2 through MM5

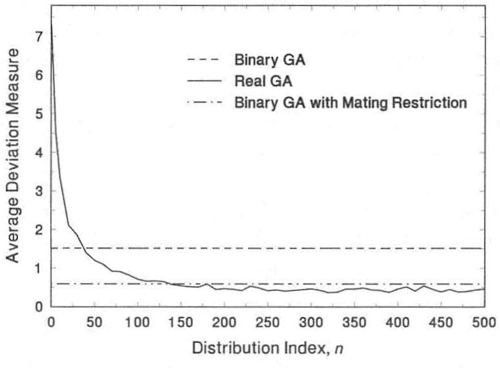

Figure 13: Average deviation measure (for generations 101 to 200) is plotted with distribution index n for function MM2.

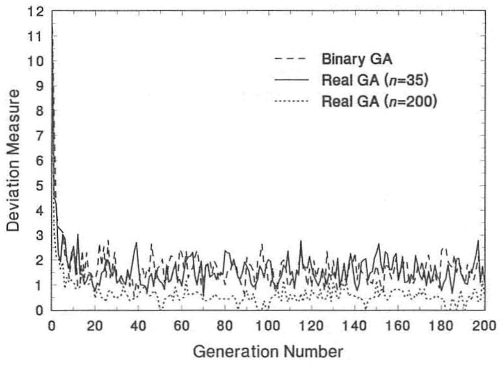

Figure 14: Deviation measure is plotted with generation number for function MM2.

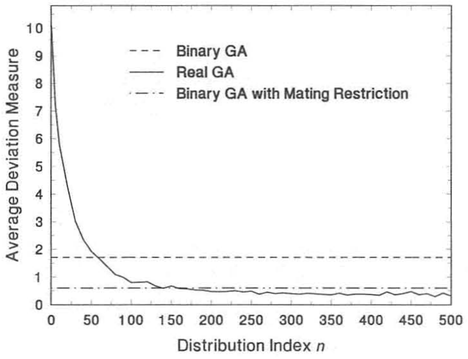

Figure 15: Average deviation measure (for generations 101 to 200) is plotted with distribution index n for function MM3.

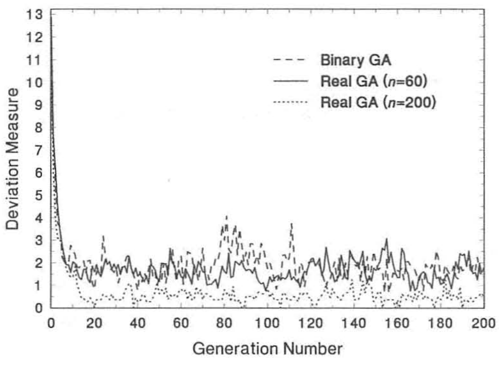

Figure 16: Deviation measure is plotted with generation number for function MM3.

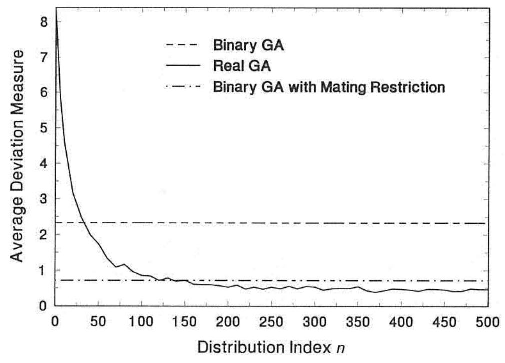

Figure 17: Average deviation measure (for generations 101 to 200) is plotted with distribution index n for function MM4.

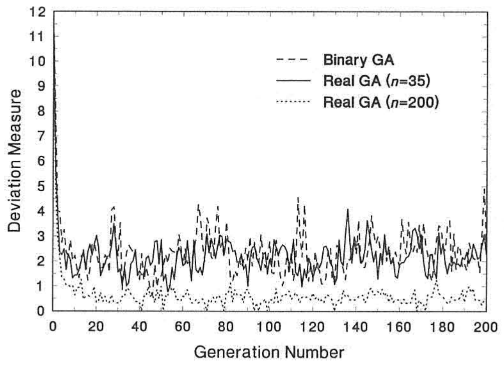

Figure 18: Deviation measure is plotted with generation number for function MM4.

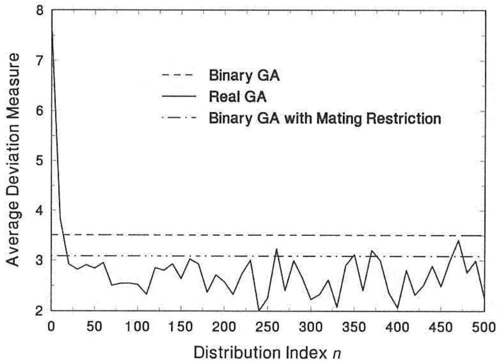

Figure 19: Average deviation measure (for generations 101 to 200) is plotted with distribution index n for function MM5.

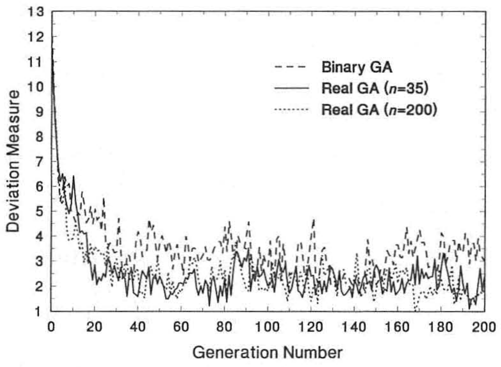

Figure 20: Deviation measure is plotted with generation number for function MM5.

References

[1] Eshelman, L. J. and Schaffer, J. D., "Real-coded Genetic Algorithms and Interval Schemata," in *Foundations of Genetic Algorithms, II*, edited by D. Whitley (Morgan Kaufmann, San Mateo, CA, 1993).

[2] Wright, A., "Genetic Algorithms for Real Parameter Optimization," in *Foundations of Genetic Algorithms*, edited by G. J. E. Rawlins (Morgan Kaufmann, San Mateo, CA, 1991).

[3] Deb, K. and Agrawal, R., "Simulated Binary Crossover for Continuous Search Space," *Complex Systems*, **9** (1995) 115–148.

[4] Goldberg, D. E., and Richardson, J., "Genetic Algorithms with Sharing for Multimodal Function Optimization," in *Proceedings of the Second International Conference on Genetic Algorithms*, edited by J. J. Grefenstette (Lawrence Erlbaum Associates, Hillsdale, NJ, 1987).

[5] Deb, K., "Genetic Algorithms in Multimodal Function Optimization," (University of Alabama, Tuscaloosa, TCGA Report Number 89002).

[6] Goldberg, D. E., Deb, K., and Horn, J., "Massive Multimodality, Deception, and Genetic Algorithms," in *Parallel Problem Solving from Nature II*, edited by R. Manner and B. Manderick (North-Holland, Amsterdam, 1992).

[7] De Jong, K. A., "An Analysis of the Behavior of a Class of Genetic Adaptive Systems," (doctoral dissertation, University of Michigan, Ann Arbor). *Dissertation Abstracts International*, **36** (1975) 5140B (University Microfilms Number 76-9381).

[8] Smith, R. E., Forrest, S., and Perelson, A. S., "Searching for Diverse, Cooperative Populations with Genetic Algorithms," (University of Alabama, Tuscaloosa, TCGA Report Number 92002, 1992).

[9] Fonseca, C. M. and Fleming P. J., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion, and Generalization," in *Proceedings of the Fifth International Conference on Genetic Algorithms*, edited by S. Forrest (Morgan Kaufmann, San Mateo, CA, 1993).

[10] Horn, J. and Nafpliotis, N., "Multiobjective Optimization using Niched Pareto Genetic Algorithms," (University of Illinois at Urbana-Champaign, IlliGAL Report Number 93005, 1993).

[11] Srinivas, N. and Deb, K., "Multiobjective Function Optimization using Nondominated Sorting Genetic Algorithms," *Evolutionary Computation*, **2** (1995) 221–248.

[12] Srinivas, N., *Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms* (master's thesis, Indian Institute of Technology, Kanpur, 1994).

[13] Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning* (Addison-Wesley, Reading, 1989).

[14] Reklaitis, G. V., Ravindran, A., and Ragsdell, K. M., *Engineering Optimization—Methods and Applications* (Wiley, New York, 1983).

[15] Deb, K., "Optimal Design of a Welded Beam Structure via Genetic Algorithms," *AIAA Journal*, **29** (1991) 2013–2015.

[16] Oei, C. K., Goldberg, D. E., and Chang, S-J., "Tournament Selection, Niching, and the Preservation of Diversity," (University of Illinois at Urbana-Champaign, IlliGAL Report Number 91011, 1991).

[17] Kumar, A., "Multimodal and Multiobjective Optimization using Real-coded Genetic Algorithms," (master's thesis, Indian Institute of Technology, 1996).

[18] Deb, K., "GeneAS: A Robust Optimal Design Technique for Mechanical Component Design," in *Evolutionary Algorithms in Engineering Applications*, edited by D. Dasgupta and Z. Michalewicz, (Springer-Verlag, New York, in press).