Top (2011) 19:67–92
DOI 10.1007/s11750-009-0082-7

ORIGINAL PAPER

A crossover operator that uses Pareto optimality in its definition

I. Alberto · P.M. Mateo

Received: 15 February 2008 / Accepted: 21 March 2009 / Published online: 8 April 2009

Abstract Evolutionary Algorithms are search and optimisation methods based on the principles of natural evolution and genetics that attempt to approximate the optimal solution of a problem. Instead of only one, they evolve a population of potential solutions to the problem, using operators like mutation, crossover and selection.

In this work, we present a new crossover operator, in the context of Multiobjective Evolutionary Algorithms, which makes use of the concept of Pareto optimality. After that it is compared to four common crossover operators. The results obtained are very promising.

Keywords Multiobjective decision making · Metaheuristics · Evolutionary algorithms

Mathematics Subject Classification (2000) 90C29 · 90C59 · 68T20

1 Introduction

In order to model problems in a more realistic way, complex formulations of real systems have to be established. A great amount of these formulations constitute multiobjective problems in which there are more than one objective function for comparing the quality of solutions. Often, these complex models are unapproachable with

I. Alberto
Department of Statistical Methods, Technical School of Industrial Engineers, University of Zaragoza, C/María de Luna 3, 50018 Zaragoza, Spain
e-mail: isolina@unizar.es

P.M. Mateo (✉)
Department of Statistical Methods, Faculty of Sciences (Mathematics), University of Zaragoza, C/Pedro Cerbuna 12, 50009 Zaragoza, Spain
e-mail: mateo@unizar.es

Fig. 1 General schema of an evolutionary algorithm

```
generate initial population $\mathcal{P}_0$
evaluate $\mathcal{P}_0$
$t = 0$
while (no termination condition)
    variate $\mathcal{P}_t$
    evaluate $\mathcal{P}_t$
    select $\mathcal{P}_{t+1}$
    $t = t + 1$
end while
```

classical techniques, but the use of heuristic methods allows the researchers to deal with them.

Evolutionary Algorithms, EAs, can be considered the most adequate methods for solving complex Multiobjective Optimisation Problems (MOOPs). The general schema of an EA is shown in Fig. 1. For a general vision in this field, the reader is referred to these classical books: Holland (1975), Goldberg (1989) and Michalewicz (1996), and the more recent book with a complete description of the different techniques by Eiben and Smith (2007).

Regarding the multiobjective optimisation field, since the Vector Evaluated Genetic Algorithm (VEGA) was proposed by Schaffer in the mid-1980s Schaffer (1984, 1985), a large amount of different EA implementations have been proposed and a great quantity of papers on this matter have appeared. The web site by Coello (2008) and the books by Coello et al. (2007) and Deb (2001) provide a good and complete introduction as well as a broad list of references about Evolutionary Multiobjective Optimisation.

Researchers are aware of the importance of efficiency in these kinds of algorithms in the sense of obtaining different elements whose improvement provide us with "better" solutions in a "faster" way. These elements can be: solution representation, management of solutions, design of operators (mutation, crossover, selection), techniques for improving the coverage of the Pareto front, and so on.

The crossover operator is an important element in the design of EAs. Together with the mutation operator, both are responsible for the exploration of the space of solutions. A great amount of crossover operators can be found in the literature, for example, in Deb (2001) and in Coello (2005) some of the most widely used are presented.

In this work, a new crossover operator is introduced and compared to the following operators (Deb 2001): naïve crossover, linear crossover, blend crossover, and simulated binary crossover. As it will be shown, the new crossover operator outperforms the other ones.

The paper is organised as follows: In the next section, we briefly present the definitions and notations on multiobjective problems that we will need later. In Sect. 3, we present the four crossover operators we will use to compare them to the new crossover operator, which is explained in Sect. 4. Next, we explain the experiment carried out for accomplishing the comparison. Finally, the conclusions are presented in Sect. 6.

2 Multiobjective optimisation problems and Pareto optimality

The aim of Multiobjective Optimisation is to optimise a set of objective functions which, in general, may be of a conflicting nature. Hence, the term "optimise" means to find a solution satisfying the constraints, which would give reasonable values of all objective functions to the decision maker. More formally, MOOPs can be defined in the following way:

$$\begin{aligned}&\min\ \mathbf{f}(\mathbf{x}) = \big(f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\big)\\ &\text{s.t.}\quad \mathbf{x} = (x_1, \ldots, x_n) \in D \subset \mathbb{R}^n.\end{aligned}$$

Contrary to single objective optimisation, in multiobjective optimisation it is usually impossible to find one optimal solution. Instead, algorithms for optimising multiobjective problems try to find a family of points known as the *Pareto optimal set*. These points verify that there is no different feasible solution which strictly improves one component of the objective function vector without worsening at least one of the remaining ones.

A more formal definition of Pareto optimality or Pareto efficiency is the following:

Definition 1 If given a solution $\mathbf{y}$, there exists another solution $\mathbf{x}$ such that $\forall j = 1, \ldots, m,\ f_j(\mathbf{x}) \leq f_j(\mathbf{y})$ and $\exists j \in \{1, \ldots, m\}$ such that $f_j(\mathbf{x}) < f_j(\mathbf{y})$, then we will say that solution $\mathbf{x}$ *dominates* solution $\mathbf{y}$ (denoted by $\mathbf{x} \prec \mathbf{y}$), and, obviously, solution $\mathbf{y}$ will never be sensibly selected as the solution to the problem.

Definition 2 A solution $\mathbf{x} \in D$ is said to be *Pareto optimal* or *efficient* if and only if $\nexists \mathbf{y} \in D$ such that $\mathbf{y} \prec \mathbf{x}$.

Definition 3 The real Pareto optimal set will be denoted with P^{true}. The image of P^{true} in the objective function space is called *Pareto front* and it will be denoted by PF^{true}.

The crossover operator we propose takes into account the quality of the solution in terms of its efficiency. In this sense, we will need the solutions to be divided into two groups: the first one contains the efficient solutions and the second the remaining ones. The way in which the operator works depends on whether the selected individuals are efficient or not.

3 Crossover operators

Among the crossover operators that can be found in the literature, we have considered four of them. These operators and their characteristics are the following:

– Naïve crossover or standard one-point crossover (C_1) (Holland 1975): Given $\mathbf{x}^{(1,t)}$ and $\mathbf{x}^{(2,t)}$ two solutions of the current population in iteration t of an EA, the naïve crossover operator selects a random crossover point j (between 2 and n) and then the components j to n in both solutions are swapped. In Fig. 2, the

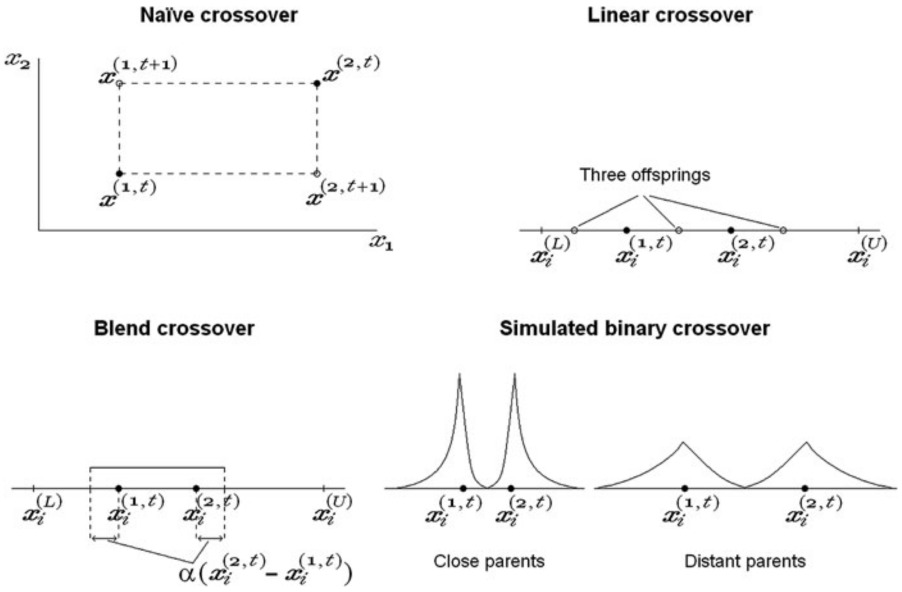

Fig. 2 Crossover operators considered. The values $x_i^{(L)}$ and $x_i^{(U)}$ represent the lower and upper bounds of component x_i, respectively. Parents are marked by *filled circles*. This is a reprint of Figs. 57, 58, 59, 61, and 62 of Deb (2001). Copyright John Wiley & Sons Limited, 2001. Reproduced with permission

solutions $\mathbf{x}^{(1,t)} = (x_1^{(1,t)}, x_2^{(1,t)})$ and $\mathbf{x}^{(2,t)} = (x_1^{(2,t)}, x_2^{(2,t)})$ are recombined taking as the crossover point the second component, obtaining the offsprings $\mathbf{x}^{(1,t+1)} = (x_1^{(1,t)}, x_2^{(2,t)})$ and $\mathbf{x}^{(2,t+1)} = (x_1^{(2,t)}, x_2^{(1,t)})$.

– Linear crossover (C_2) (Wright 1991): This operator creates three solutions from two parents. The value of the ith component of the offsprings are obtained according to:

$$0.5\big(x_i^{(1,t)} + x_i^{(2,t)}\big), \quad 1.5x_i^{(1,t)} - 0.5x_i^{(2,t)}, \quad -0.5x_i^{(1,t)} + 1.5x_i^{(2,t)}.$$

Unlike the original version of this operator, we have used a slight modification which consists of keeping the three offsprings instead of selecting the two best ones. In Fig. 2, the offsprings are marked with empty circles.

– Blend crossover (C_3) (Eshelman and Schaffer 1993): This operator, also denoted BLX-α, generates individuals according to the following formula (assuming $x_i^{(1,t)} < x_i^{(2,t)}$):

$$(1-\gamma_i)x_i^{(1,t)} + \gamma_i x_i^{(2,t)},$$

where $\gamma_i = (1+2\alpha)u_i - \alpha$ and $u_i \in U(0,1)$. The authors of this crossover as well as, for example, of Herrera et al. (1998) reported the value $\alpha = 0.5$ to behave better than others, probably because this value provides a balanced relationship between exploitation and exploration. An important property of this operator is the following: If the parents are close to each other then the offsprings are close to the parents, too. This constitutes an adaptive device: In the early stages when the parents are usually not close to each other, the operator tends to search through the whole variable space. And when the algorithm iterates and the solutions tend to be closer, the operator carries out a focused search. In Fig. 2, marked with a line is the range where the offsprings are generated using a uniform distribution.

- Simulated binary crossover (C_4) (Deb and Agrawal 1995): This operator simulates the behaviour of the single-point crossover operator on binary strings in the sense that common interval schemata between the parents are kept in the offspring. It works generating the components of the offsprings as follows:

$$x_i^{(1,t+1)} = 0.5\big[(1+\beta_i)x_i^{(1,t)} + (1-\beta_i)x_i^{(2,t)}\big],$$
$$x_i^{(2,t+1)} = 0.5\big[(1-\beta_i)x_i^{(1,t)} + (1+\beta_i)x_i^{(2,t)}\big],$$

where

$$\beta_i = \begin{cases} (2u)^{\frac{1}{\eta+1}} & \text{if } u \le 0.5, \\ (\frac{1}{2(1-u)})^{\frac{1}{\eta+1}} & \text{otherwise.} \end{cases}$$

For parameter η, we have considered the value 20 as used in Deb et al. (2002) and Zitzler et al. (2001). In Fig. 2, the line represents the probability density function for generating offsprings in the corresponding interval: The higher the line, the greater the probability of the offspring to be generated there.

4 The new Pareto based crossover operator

The new Pareto based crossover operator, PBC, is a uniform crossover operator which has a different maximum range of variation depending on the quality of the solution. The main idea of the operator is to use good parents (efficient ones) to improve the quality of the offspring (exploitation) and to use not so good parents (non-efficient ones) to explore the whole space (exploration).

More specifically, when pairs of non-efficient solutions are considered for crossover, they are recombined with a BLX-α with α initially equal to 0.5. But α is slightly modified depending on the distance between the solutions in such a way that, if the solutions are very close, the parameter α is increased, and, if the solutions are distant, α is decreased, with the objective of keeping the capacity of exploration more or less stable.

In any other case, two efficient solutions, or one efficient and the other non-efficient, are recombined generating points in intervals centred around them, with the amplitude of the interval depending on both the solutions and the iteration. If both solutions are efficient, the amplitude of the interval depends on the distance between them and on the iteration t. If only one of the solutions is efficient, the efficient one behaves as in the previous case, and the non-efficient one has an amplitude of the interval that only depends on the distance between the solutions. Also, in both cases, with probability equal to 0.25, the centres of the intervals are interchanged with the objective of increasing the capacity of sharing information between the solutions.

Let $\mathcal{P}_t$ be the current population in iteration t, POS_t its Pareto optimal set, and $p_t = |\text{POS}_t|$. In order to decide whether or not the parents are close to each other, a fact that will be used in the design of the operator, we use the following process:

Step 1: Let dist($\mathbf{x}, \mathbf{y}$) be the Euclidean distance between $\mathbf{x}$ and $\mathbf{y}$. Calculate $d^{(\text{eff},t)}$, the mean distance between all pairs of solutions in POS_t, and $\sigma^{(\text{eff},t)}$, the standard deviation, according to:

$$d^{(\text{eff},t)} = \frac{2}{p_t^2 - p_t} \sum_{\substack{\mathbf{x}^{(i,t)}, \mathbf{x}^{(j,t)} \in \text{POS}_t \\ i<j}} \text{dist}\big(\mathbf{x}^{(i,t)}, \mathbf{x}^{(j,t)}\big),$$

$$\sigma^{(\text{eff},t)} = \sqrt{\frac{2}{p_t^2 - p_t} \sum_{\substack{\mathbf{x}^{(i,t)}, \mathbf{x}^{(j,t)} \in \text{POS}_t \\ i<j}} \big(\text{dist}\big(\mathbf{x}^{(i,t)}, \mathbf{x}^{(j,t)}\big) - d^{(\text{eff},t)}\big)^2}.$$

Step 2: Let f^{Neff} and $f^{(\text{eff},t)}$ be the amplitude factors associated to the non-efficient and efficient solutions, respectively, calculated according to

$$f^{\text{Neff}} = 1,$$
$$f^{(\text{eff},t)} = 1 - \lfloor t/\text{stepsIter} \rfloor \cdot 0.11,$$

where stepsIter corresponds to the maximum number of iterations performed by the algorithm divided by 10 and $\lfloor \cdot \rfloor$ represents the floor function, i.e. the greatest integer less than or equal to its argument. Observe that $f^{(\text{eff},t)}$ is a decreasing step function that takes values in [0.01,1].

The amplitude factors $f^{(\text{eff},t)}$ and f^{Neff} will act in such a way that, if the individuals are close, the factors extend the default amplitude of the interval, $\Delta(i) = \frac{3}{4}\,|x_i^{(1,t)} - x_i^{(2,t)}|$, $i = 1, \dots, n$, in which the components of the descendants will take values. Otherwise, they will reduce the amplitude.

After the above establishments, let $\mathbf{x}^{(1,t)}$ and $\mathbf{x}^{(2,t)}$ be the parent solutions selected for crossover. At that moment, and depending on the distance between $\mathbf{x}^{(1,t)}$ and $\mathbf{x}^{(2,t)}$ and on the values of $d^{(\text{eff},t)}$ and $\sigma^{(\text{eff},t)}$, the factors $f^{(\text{eff},t)}$ and f^{Neff} are both multiplied by one of the following quantities:

$$\begin{array}{lll}
\text{(i)} & 1 & \text{if } d^{(\text{eff},t)} = 0 \text{ or } \sigma^{(\text{eff},t)} = 0, \\
\text{(ii)} & \frac{4}{3} & \text{if dist}(\mathbf{x}^{(1,t)}, \mathbf{x}^{(2,t)}) < d^{(\text{eff},t)} - 3\sigma^{(\text{eff},t)}, \\
\text{(iii)} & \frac{2}{3} & \text{if dist}(\mathbf{x}^{(1,t)}, \mathbf{x}^{(2,t)}) > d^{(\text{eff},t)} + 3\sigma^{(\text{eff},t)}, \\
\text{(iv)} & (1 + \frac{d\ 0.11}{\sigma^{(\text{eff},t)}}) & \text{if } d^{(\text{eff},t)} - 3\sigma^{(\text{eff},t)} \le \text{dist}(\mathbf{x}^{(1,t)}, \mathbf{x}^{(2,t)}) < d^{(\text{eff},t)}, \\
\text{(v)} & (1 - \frac{d\ 0.11}{\sigma^{(\text{eff},t)}}) & \text{if } d^{(\text{eff},t)} + 3\sigma^{(\text{eff},t)} \ge \text{dist}(\mathbf{x}^{(1,t)}, \mathbf{x}^{(2,t)}) \ge d^{(\text{eff},t)},
\end{array}$$

where $d = |d^{(\text{eff},t)} - \text{dist}(\mathbf{x}^{(1,t)}, \mathbf{x}^{(2,t)})|$.

It can be observed that in cases (ii) and (iii) the factors are increased and decreased by a factor of 33%, respectively. In cases (iv) and (v), a continuous correction is made but the maximum modification is equal to 33%.

Finally, the last correction in $f^{(\text{eff},t)}$ is accomplished by

$$f^{(\text{eff},t)} = \min\big\{1, f^{(\text{eff},t)}\big\}.$$

After these calculations and in order to carry out the crossover, three cases are considered: neither solution is efficient, both solutions are efficient, and only one is efficient.

- **Case 1**: Neither of the parents is efficient. In this situation, the new components $x_i^{(1,t+1)}$ and $x_i^{(2,t+1)}$, will be generated taking two random values in the interval

$$\left[\frac{x_i^{(1,t)}+x_i^{(2,t)}}{2}-\frac{4}{3}\,\Delta(i)\,f^{\text{Neff}},\frac{x_i^{(1,t)}+x_i^{(2,t)}}{2}+\frac{4}{3}\,\Delta(i)\,f^{\text{Neff}}\right].$$

  Note that this corresponds to a BLX-α operator, where $\frac{1}{6}\leq\alpha\leq\frac{5}{6}$, depending on the value f^{Neff}.
- **Case 2**: Both parents are efficient solutions. Then the two values, $x_i^{(h,t+1)}$, $h=1,2$, for the new solutions are randomly created in the intervals

$$\left[x_i^h-\Delta(i)\,f^{(\text{eff},t)},x_i^h+\Delta(i)\,f^{(\text{eff},t)}\right],\quad h=1,2,$$

  where, when we are obtaining $x_i^{(h,t+1)}$, x_i^h is equal to $x_i^{(h,t)}$ with probability 0.75 or equal to $x_i^{(3-h,t)}$ with probability 0.25, $h=1,2$.
- **Case 3**: Only one of the parents is efficient. This situation is similar to Case 2. Assuming that $\mathbf{x}^{(1,t)}$ is the efficient solution (the other case is analogous), then one value is obtained in the interval

$$\left[x_i^1-\Delta(i)\,f^{(\text{eff},t)},x_i^1+\Delta(i)\,f^{(\text{eff},t)}\right],$$

  where x_i^1 is equal to $x_i^{(1,t)}$ with probability 0.75 or equal to $x_i^{(2,t)}$ with probability 0.25. And the other value is taken in the interval

$$\left[x_i^2-\Delta(i)\,f^{\text{Neff}},x_i^2+\Delta(i)\,f^{\text{Neff}}\right],$$

  where x_i^2 is equal to $x_i^{(2,t)}$ with probability 0.75 or equal to $x_i^{(1,t)}$ with probability 0.25.

In all these situations, the solutions are randomly generated in the corresponding intervals using the uniform distribution (i.e. using a constant probability density function in the interval). In all the cases, if the obtained solution is not feasible, another one is generated in the same way.

From the former cases, the following key ideas can be observed: All the values of the new components are obtained using the uniform distribution in an interval centred in $(x_i^{(1,t)}+x_i^{(2,t)})/2$ or x_i^h, and with a default amplitude equal to $\frac{4}{3}\Delta(i)$ or $\Delta(i)$ depending on whether Case 1, 2 or 3 holds. Our aim is to use non-efficient solutions to broadly explore the space and the efficient ones to maintain the capacity of exploitation. Then, in the case of non-efficient solutions, their default amplitude is increased or reduced by a maximal amount of 33%, maintaining an important capacity of exploration. On the other hand, in the case of efficient solutions, their default amplitude is also multiplied by a factor that tends to zero when the number of iterations is advanced in such a way that initially it has also an exploratory capacity which is subsequently substituted, when iterations proceed, by a higher capacity of exploitation.

5 Computational experiment

In this section, the computational experiment we have carried out to test the performance of PBC is explained. We should remark that the aim of the paper is to compare the PBC with some of the existing crossover operators and not to create a new Evolutionary Algorithm. Therefore, the comparison is accomplished using a naïve algorithm as the one shown in Fig. 1, and in which the 'variate' step only involves one of the crossover operators considered and no mutation operator is used.

All the calculations were carried out on a PC Pentium 2.66 Ghz with 512 MB of RAM under Linux Mandriva 2006, and all programmes were coded in C language and compiled with a GNU GCC compiler. A precision of 10^{-8} has been taken for all arithmetical calculations, one unit of accuracy more than the one used in the Pareto Fronts supplied by CEC (2007). All the codes used in the paper were written by the authors, except the code for the hypervolume calculation that was obtained from Fonseca et al. (2006b) based on Fonseca et al. (2006a), and all the codes for the function evaluations that were extracted from the available codes provided in CEC (2007).

This section contains four subsections. The first focuses on showing the performance measures used to accomplish the comparison of the results, and the second shows the test problems that we have considered. In the third subsection, we describe how the experiment is carried out and, finally, in the last part we report the obtained results.

5.1 Measures for comparing the operators

In order to compare our crossover operator with the other operators pointed out before, we have considered three measures: hypervolume difference to a reference set (CEC 2007; Huang et al. 2007), generational distance (Coello et al. 2007), and set coverage (Zitzler 1989).

The outline of these measures is the following: Given P^*, the set of non-dominated solutions resulting from the execution of an optimisation algorithm, and PF*, its image in the objective space,

– *Hypervolume difference to a reference set* (HD): The original hypervolume measure (Zitzler and Thiele 1999) calculates the volume covered by the hypercube formed by the vectors of PF* (considering that all the objectives are to be minimised) and a reference point. The reference point is placed in such a way so as to be at least weakly dominated by every member in P^*, and can be found, for example, by constructing a vector of worst objective function values in PF*. Since this metric is not free from arbitrary scaling of objectives, we will normalise the objective values first. In order to obtain the value of the hypervolume difference, HD, we calculate the difference between the hypervolume of a reference set contained in PF^{true} and the hypervolume of PF*. This reference set consists of 500 points for two objective problems and of 5000 points for three objective problems, they have all been obtained from CEC (2007). For this measure, the smaller the value, the better the population, in contrast to the original hypervolume measure.

- *Generational distance* (*GD*): This measure, which also requires a reference set contained in $\mathrm{PF}^{\mathrm{true}}$ to be known, reports how far, on average, PF^* is from $\mathrm{PF}^{\mathrm{true}}$. Mathematically, it is defined as:

$$\mathrm{GD} = \frac{(\sum_{i=1}^{n} d_i^2)^{1/2}}{|\mathrm{PF}^*|},$$

where $|\mathrm{PF}^*|$ is the number of vectors in PF^* and d_i is the Euclidean phenotypic distance between each member, i, of PF^* and the closest member in $\mathrm{PF}^{\mathrm{true}}$ to that member i. For this measure, the smaller the value, the better the population.
- *Set coverage* (C): Let P_1^* and P_2^* the non-dominated sets resulting from the execution of two different algorithms. This measure calculates the proportion of solutions in set P_2^* which are weakly dominated ($\preceq$ means $\prec$ or $=$) by solutions in set P_1^*:

$$C(P_1^*, P_2^*) = \frac{|\{b \in P_2^* | \exists a \in P_1^*, a \preceq b\}|}{|P_2^*|}.$$

Since this operator is not symmetrical, both $C(P_1^*, P_2^*)$ and $C(P_2^*, P_1^*)$ have to be calculated. In the best situation, P_1^* is better than P_2^* if $C(P_1^*, P_2^*)$ is close to one and $C(P_2^*, P_1^*)$ is close to zero. In general, P_1^* is better than P_2^* if $C(P_1^*, P_2^*)$ is notably greater than $C(P_2^*, P_1^*)$.

5.2 The test problems

The use of a set of test problems helps to guarantee that the proposed operator will confront efficient solution spaces of different characteristics. The test suites ZDT (Zitzler et al. 2000) and DTLZ (Deb et al. 2002) have been extensively used. However, these suites have some drawbacks since the test problems share some characteristics and often have some design flaws. Then, we have used the extended and rotated or shifted version of some of these problems proposed by Huang et al. (2007). A short description of their characteristics is shown in Table 1.

For problems 1 to 3 and 5, the number of objective functions is 2 and the number of decision variables is 30; for problems 6 and 8, these numbers are equal to 3 and 10, respectively; and finally, problems 4 and 7 have 10 decision variables and 2 and 3 objective functions, respectively. In all of these test problems, the Pareto optimal set is known. For a review on multiobjective test problems the reader is addressed to Huband et al. (2007).

5.3 Selection operators

After applying the crossover operator, we conform a temporary population consisting of the current one together with the new individuals obtained in order to apply the selection process. We have used three different selection operators: two of them of the roulette wheel selection type (surviving and death selection) and the other of the elitist type. The operators use a ranking of the individuals of the population. To establish this ranking, we have used the one proposed by Goldberg (1989), but any other can be used, for example, the one introduced by Fonseca and Fleming (1993)

Table 1 Properties of the test functions. S: Separable; NS: Nonseparable; U: Unimodal; M: Multimodal

Test problem	Objective	Number of variables	Separability	Modality	Geometry
1. S-ZDT1	f_1	1	S	U	Convex
	f_2	>1	S	U	
2. S-ZDT2	f_1	1	S	U	Concave
	f_2	>1	S	U	
3. S-ZDT4	f_1	1	S	U	Convex
	f_2	>1	S	M	
4. R-ZDT4	$f_{1:2}$	>1	NS	M	Convex
5. S-ZDT6	f_1	1	S	M	Concave
	f_2	>1	S	M	
6. S-DTLZ2	$f_{1:3}$	>1	S	U	Concave
7. R-DTLZ2	$f_{1:3}$	>1	NS	M	Concave
8. R-DTLZ3	$f_{1:3}$	>1	S	M	Concave

or even the raw fitness defined in Zitzler et al. (2001) for the algorithm SPEA2. The ranking proposed by Goldberg assigns a value equal to 1 to the efficient solutions, then these solutions are removed from the population. The efficient solutions of this new set are assigned ranking 2, and the process continues until there are no solutions left.

The selection operators considered are the following:

- *Surviving selection* (S_1) (Michalewicz 1996) (pp. 34): This operator assigns a higher probability to those individuals with smaller ranking. In iteration t, the selection probability $p_i(t)$ is defined for each individual $\mathbf{x}^{(i,t)} \in \mathcal{P}_t$ according to:

$$p_i(t) = \frac{r_{\max} + 1 - r_i}{\sum_{\mathbf{x}^{(j,t)} \in \mathcal{P}_t} (r_{\max} + 1 - r_j)},$$

  where r_j is the ranking value of individual $\mathbf{x}^{(j,t)}$ in $\mathcal{P}_t$ and $r_{\max} = \max_j\{r_j\}$. Then, using these probabilities, a roulette wheel selection is performed in order to fill the whole new population.
- *Death selection* (S_2) (Michalewicz 1996) (pp. 62): In this case, each individual receives a probability of not surviving (a higher probability to those individuals with greater ranking) defined according to:

$$p_i(t) = \frac{r_i}{\sum_{\mathbf{x}^{(j,t)} \in \mathcal{P}_t} r_j}.$$

  Then, a roulette wheel selection with these probabilities is accomplished in order to eliminate the spare individuals until reaching the fixed population size. With this mechanism the population does not receive two copies of the same individual, thus helping to avoid crowding the population with the same individuals.

– *Elitist selection* (S_3): Taking into account the size of the population, the spare individuals with higher ranking, i.e. the worst solutions, are removed from the population without random selection. The process is similar to the selection operator used in the algorithm NSGA-II (Deb et al. 2002): It uses a crowding distance to keep a diverse front by making sure each member stays a crowding distance apart. This keeps the population diverse and helps the algorithm to explore the fitness landscape.

5.4 Implementation and execution

For implementing the experiment we have considered a usual fixed population size equal to 100 individuals as in Zitzler et al. (2001) and Deb et al. (2002). The comparison of our PBC with C_1, C_2, C_3, and C_4 is carried out taking into account different scenarios. A scenario is defined by any combination of the levels of the factors: number of iterations and selection operator. The number of iterations takes three different values (levels of this factor): 100, 500 and 1000 times which represent 10^4, 5×10^4 and 10^5 function evaluations since, for each individual $\mathbf{x} \in \mathcal{P}_t$, the probability of crossover is 1, which means that in every generation all the individuals are recombined. Also, when two individuals are selected for crossover, all their components are recombined. The factor selection operator has three different levels which correspond to the three different selection operators of the previous subsection. Since both factors have 3 levels, the number of different scenarios is equal to 9.

For each problem, 50 initial populations are generated. We apply the algorithm using a different crossover operator each time, considering each of the 9 different scenarios. By doing so, we obtain 50 final populations for each combination of crossover operator and scenario. In each of these populations, we obtain the set of efficient solutions, P^*, and we calculate the measures of Sect. 5.1. For calculating HD, for each problem and number of iterations, the reference point is placed by obtaining the worst objective function value among all the P^* populations obtained with the different crossover and selection operators. That is to say, for each problem the same reference point is used for all the P^* once the number of iterations has been fixed.

Figures 3 to 11 summarise the results for the three measures considered: Figs. 3, 4 and 5 for a number of iterations equal to 100; Figs. 6, 7 and 8 for a number of iterations equal to 500, and finally, Figs. 9, 10 and 11 for a number of iterations equal to 1000. The figures show the mean value of the 50 executions in each scenario. To be able to slightly distinguish the results in Figs. 4, 7 and 10 (those corresponding to the measure GD) the scale of edge y is limited to 500. In this measure and for Problem 8 (R-DTLZ3), values around 2500 are reached in some scenarios. Some relevant results for Problem 8 and the measure GD are the following: In the case of 100 iterations, the best value of GD is reached by C_1 for all the selection operators and the second best by PBC. For 500 iterations, the best is C_1 for selections S_1 and S_2, and PBC for selection S_3. The second best is PBC for selections S_1 and S_2, and C_1 for selection S_3. And for 1000 iterations, the best crossover operator is always PBC followed by C_1.

Since only the mean values themselves are not enough to decide whether an operator is better for a given measure than other or not, we have considered carrying

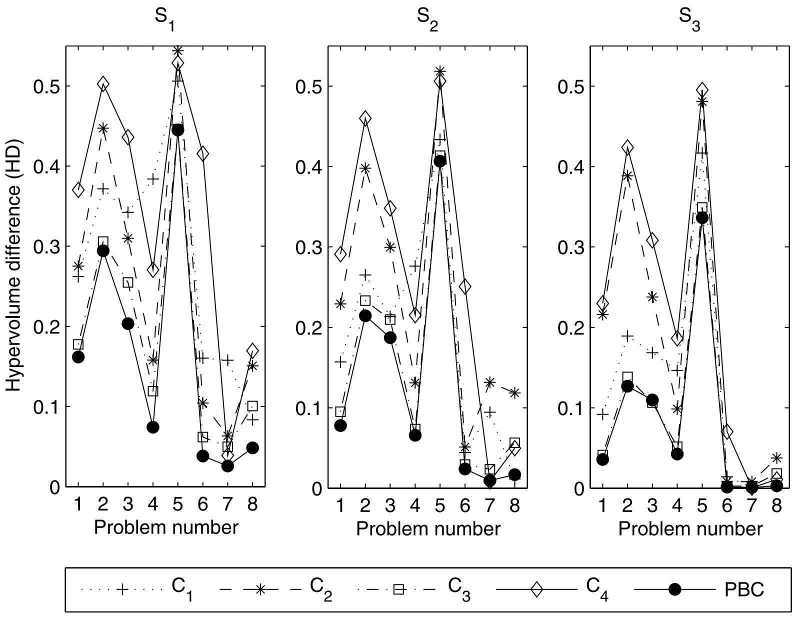

Fig. 3 Mean values of hypervolume difference (HD) for each scenario and crossover operator considered. Results for 100 iterations

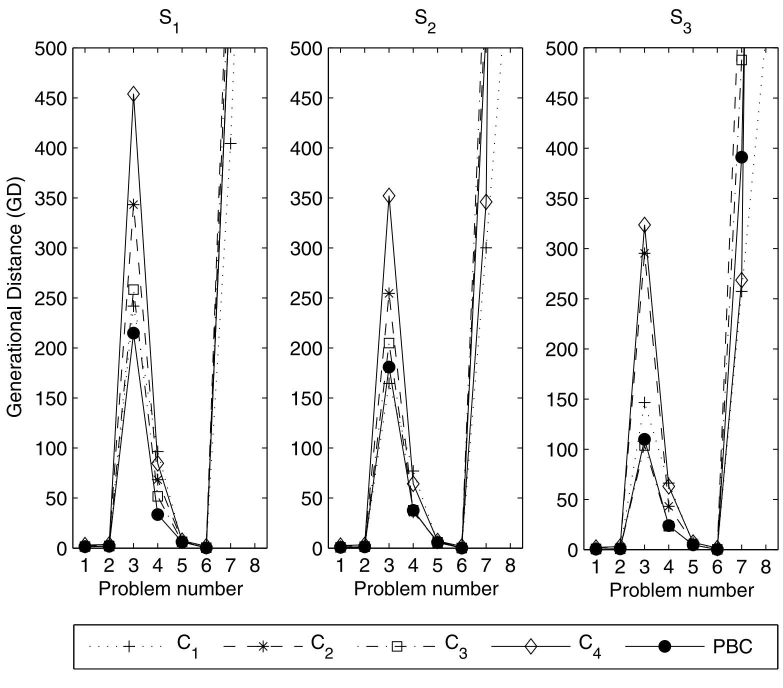

Fig. 4 Mean values of generational distance (GD) for each scenario and crossover operator considered. Results for 100 iterations

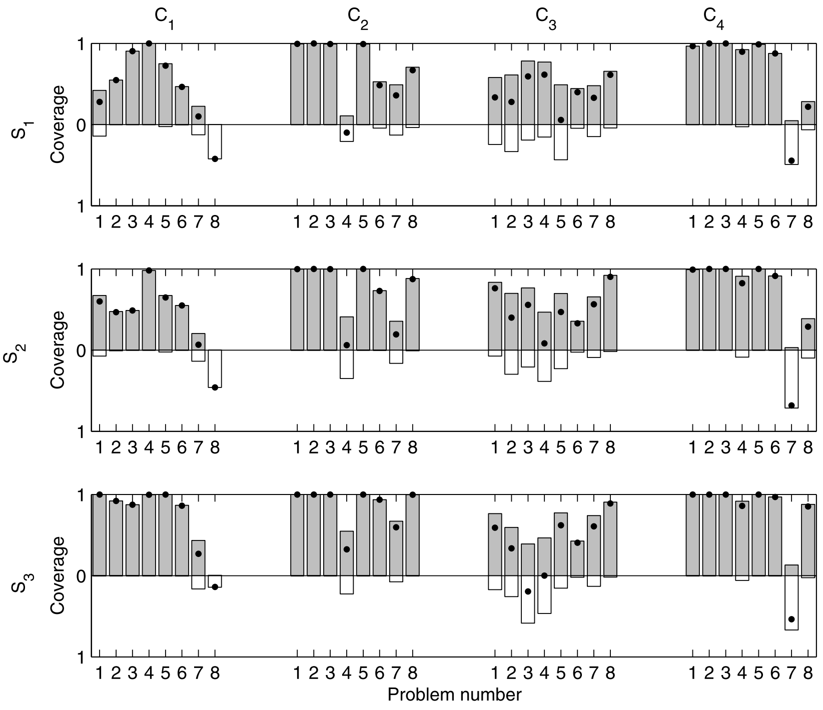

Fig. 5 Mean values of coverage for each scenario; $C(\text{PBC}, C_i)$ in *grey* and $C(C_i, \text{PBC})$ in *white*. The *black dot* represents the difference $C(\text{PBC}, C_i) - C(C_i, \text{PBC})$. Results for 100 iterations

out a statistical test to determine this fact (Casella and Berger 2002). To perform statistical hypothesis tests for comparing PBC with C_i, $i = 1, 2, 3, 4$, and with the objective of determining the effect of the crossover when using them in the algorithm starting from identical initial conditions, the data are paired in the following way: For each scenario, the measure obtained in the resulting population using crossover C_i is paired with the measure obtained in the resulting population using crossover PBC.

Two different hypothesis tests are performed. For each measure, the first hypothesis test is H_0: PBC performs worse than or equal to C_i versus H_1: PBC performs strictly better than C_i, $i = 1, 2, 3, 4$. Note that this hypothesis test is the one that least favours our operator PBC, since the fact of PBC being strictly better than C_i is placed in the alternative hypothesis. The second hypothesis test, applied when the former null hypothesis H_0 is accepted, is H_0: PBC performs equally to C_i versus H_1: PBC is different from C_i, $i = 1, 2, 3, 4$. For both tests, the significance level is equal to 0.05.

In Table 2, the results of these tests are shown. A '+' sign appears when the first null hypothesis is rejected (and so, PBC is strictly better than C_i). If the first null hypothesis is accepted, an '=' sign appears when the second null hypothesis H_0: PBC is equal to C_i is accepted (then, both operators PBC and C_i have an equivalent behaviour); and a '−' sign otherwise (which means that PBC cannot be considered equal to or better than C_i, $i = 1, 2, 3, 4$). Then, in this table, the fewer '−' signs

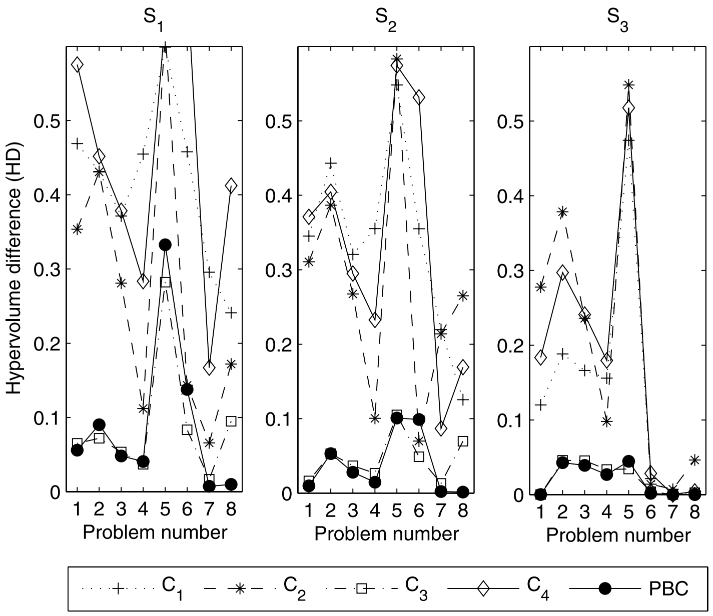

Fig. 6 Mean values of hypervolume difference (HD) for each scenario and crossover operator considered. Results for 500 iterations

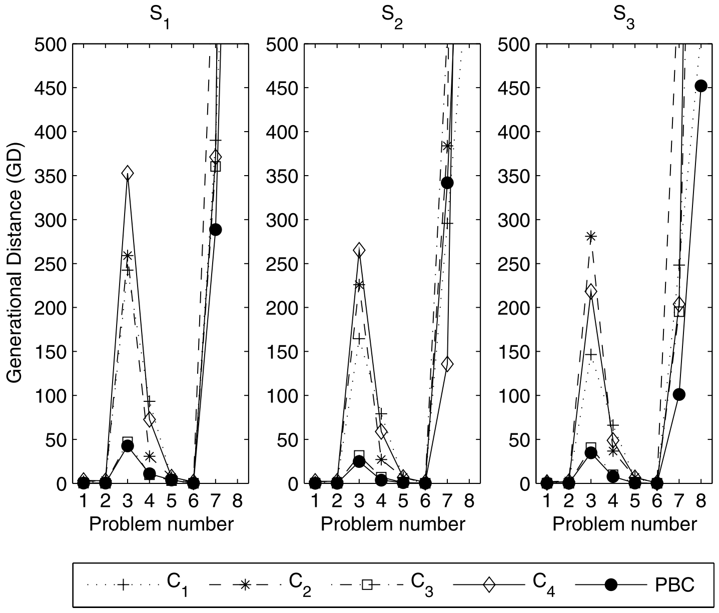

Fig. 7 Mean values of generational distance (GD) for each scenario and crossover operator considered. Results for 500 iterations

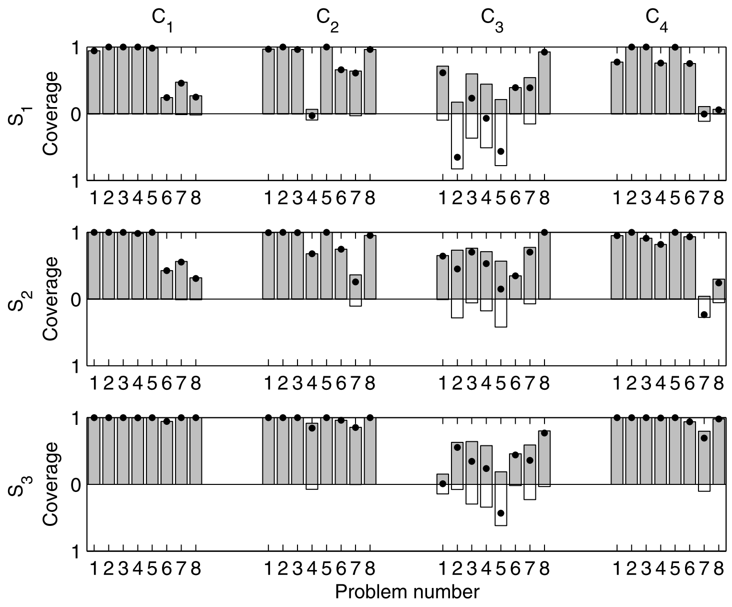

Fig. 8 Mean values of coverage for each scenario; $C(\text{PBC}, C_i)$ in *grey* and $C(C_i, \text{PBC})$ in *white*. The *black dot* represents the difference $C(\text{PBC}, C_i) - C(C_i, \text{PBC})$. Results for 500 iterations

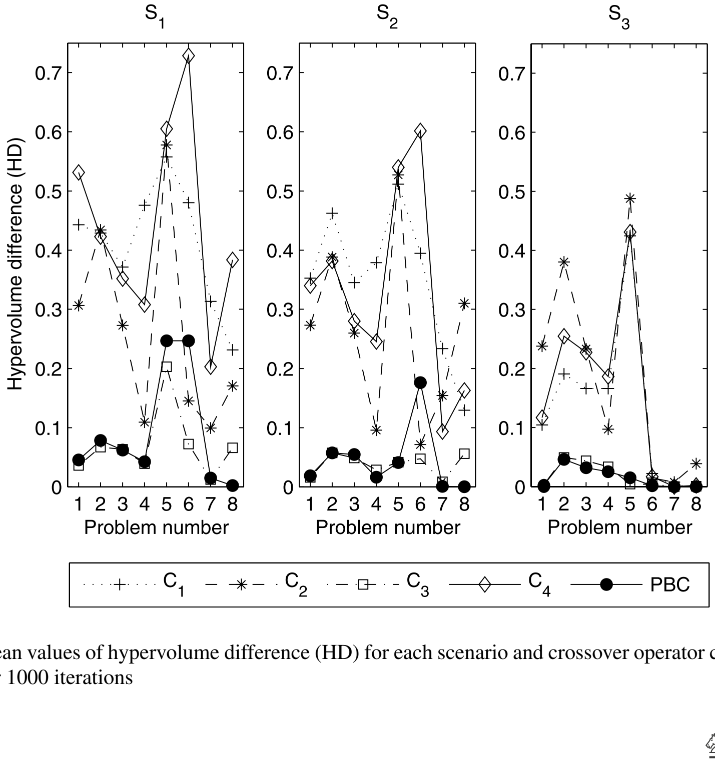

Fig. 9 Mean values of hypervolume difference (HD) for each scenario and crossover operator considered. Results for 1000 iterations

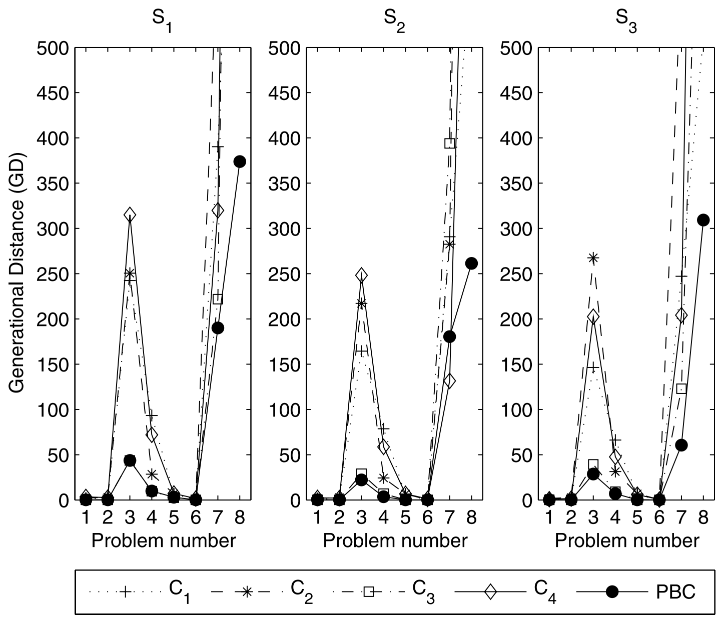

Fig. 10 Mean values of generational distance (GD) for each scenario and crossover operator considered. Results for 1000 iterations

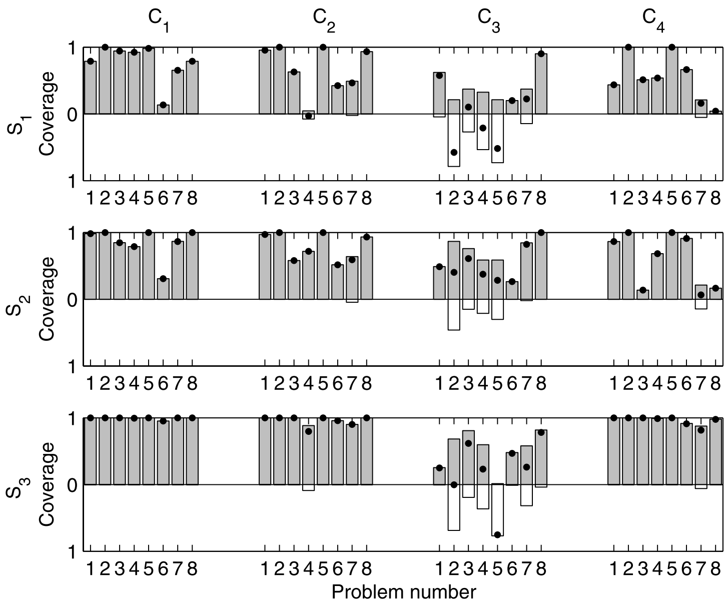

Fig. 11 Mean values of coverage for each scenario; $C(\text{PBC}, C_i)$ in *grey* and $C(C_i, \text{PBC})$ in *white*. The black dot represents the difference $C(\text{PBC}, C_i) - C(C_i, \text{PBC})$. Results for 1000 iterations

Table 2 Results of the hypothesis tests performed considering all problems simultaneously. A '+' sign means that the hypothesis "PBC performs better than C_i" is accepted. An '=' means that the hypothesis "PBC performs equally to C_i" is accepted. And a '−' represents that the hypothesis "PBC performs worse than C_i" is accepted. The numbers represent the 95% confidence interval for the mean of the paired difference of measures $M_{C_i} - M_{\mathrm{PBC}}$, by means of the mean value and the semiamplitude

Sel.	Mea.	PBC versus					
		C_1			C_2		
		No. iterations			No. iterations		
		100	500	1000	100	500	1000
S_1	HD	$+: 0.122 \pm 0.014$	$+: 0.325 \pm 0.014$	$+: 0.320 \pm 0.015$	$+: 0.095 \pm 0.005$	$+: 0.182 \pm 0.013$	$+: 0.172 \pm 0.015$
	GD	$-: -164.2 \pm 41.62$	$+: 32.19 \pm 16.03$	$+: 142.6 \pm 22.96$	$+: 87.81 \pm 15.14$	$+: 271.4 \pm 50.36$	$+: 374.4 \pm 74.66$
	C	$+: 0.450 \pm 0.051$	$+: 0.734 \pm 0.043$	$+: 0.776 \pm 0.043$	$+: 0.674 \pm 0.042$	$+: 0.766 \pm 0.039$	$+: 0.672 \pm 0.039$
S_2	HD	$+: 0.062 \pm 0.010$	$+: 0.301 \pm 0.016$	$+: 0.306 \pm 0.017$	$+: 0.109 \pm 0.009$	$+: 0.236 \pm 0.016$	$+: 0.214 \pm 0.018$
	GD	$-: -202.9 \pm 46.63$	$-: -17.43 \pm 15.19$	$+: 84.05 \pm 13.14$	$+: 79.26 \pm 19.51$	$+: 215.5 \pm 48.50$	$+: 270.4 \pm 60.50$
	C	$+: 0.418 \pm 0.054$	$+: 0.783 \pm 0.039$	$+: 0.849 \pm 0.038$	$+: 0.732 \pm 0.048$	$+: 0.827 \pm 0.033$	$+: 0.789 \pm 0.033$
S_3	HD	$+: 0.047 \pm 0.004$	$+: 0.122 \pm 0.013$	$+: 0.120 \pm 0.013$	$+: 0.102 \pm 0.009$	$+: 0.181 \pm 0.017$	$+: 0.172 \pm 0.016$
	GD	$-: -130.9 \pm 33.06$	$+: 48.13 \pm 7.500$	$+: 71.79 \pm 9.764$	$+: 182.1 \pm 33.24$	$+: 355.5 \pm 66.16$	$+: 376.7 \pm 70.74$
	C	$+: 0.730 \pm 0.042$	$+: 0.992 \pm 0.004$	$+: 0.993 \pm 0.004$	$+: 0.856 \pm 0.034$	$+: 0.957 \pm 0.016$	$+: 0.957 \pm 0.016$

Table 2 (*Continued*)

Sel.	Mea.	PBC versus					
		C_3			C_4		
		No. iterations			No. iterations		
		100	500	1000	100	500	1000
S_1	HD	$+: 0.028 \pm 0.004$	$=: -0.002 \pm 0.006$	$-: -0.022 \pm 0.008$	$+: 0.180 \pm 0.013$	$+: 0.358 \pm 0.015$	$+: 0.350 \pm 0.013$
	GD	$+: 63.09 \pm 14.26$	$+: 155.1 \pm 39.91$	$+: 211.6 \pm 55.78$	$+: 149.0 \pm 33.16$	$+: 209.5 \pm 43.90$	$+: 284.4 \pm 61.36$
	C	$+: 0.452 \pm 0.062$	$+: 0.157 \pm 0.082$	$+: 0.087 \pm 0.082$	$+: 0.688 \pm 0.050$	$+: 0.667 \pm 0.045$	$+: 0.543 \pm 0.045$
S_2	HD	$+: 0.017 \pm 0.002$	$+: 0.008 \pm 0.004$	$-: -0.008 \pm 0.006$	$+: 0.141 \pm 0.010$	$+: 0.295 \pm 0.015$	$+: 0.285 \pm 0.015$
	GD	$+: 78.02 \pm 16.98$	$+: 220.4 \pm 51.50$	$+: 299.3 \pm 70.25$	$+: 76.15 \pm 22.94$	$+: 90.07 \pm 28.40$	$+: 127.3 \pm 27.63$
	C	$+: 0.509 \pm 0.058$	$+: 0.565 \pm 0.061$	$+: 0.532 \pm 0.061$	$+: 0.667 \pm 0.057$	$+: 0.702 \pm 0.047$	$+: 0.603 \pm 0.047$
S_3	HD	$+: 0.007 \pm 0.002$	$=: 0.001 \pm 0.002$	$+: 0.002 \pm 0.002$	$+: 0.134 \pm 0.010$	$+: 0.162 \pm 0.015$	$+: 0.140 \pm 0.014$
	GD	$+: 102.8 \pm 24.88$	$+: 115.1 \pm 29.99$	$+: 84.12 \pm 23.11$	$+: 107.5 \pm 26.60$	$+: 203.7 \pm 41.60$	$+: 215.8 \pm 43.53$
	C	$+: 0.407 \pm 0.065$	$+: 0.285 \pm 0.066$	$+: 0.231 \pm 0.066$	$+: 0.768 \pm 0.050$	$+: 0.950 \pm 0.013$	$+: 0.962 \pm 0.013$

appear, the better the operator PBC is. With the same codification, in Tables 3 to 6 the results for all the problems appear separately.

Table 2 also shows, for every combination of crossover C_i, $i = 1, 2, 3, 4$, and scenario, the confidence interval for the mean of the paired difference of measures $M_{C_i} - M_{\text{PBC}}$, where M_{C_i} is the hypervolume difference or the generational distance calculated in the final population using crossover C_i in a specific scenario and M_{PBC} is the measure calculated in the final population using PBC in the same scenario. Since for both measures the smaller is the better, when the interval lays above 0, it means that $M_{C_i} > M_{\text{PBC}}$, PBC performing then better than C_i for that measure in that scenario. In the case of coverage, for each scenario, we have calculated the interval for the mean of the difference $C(\text{PBC}, C_i) - C(C_i, \text{PBC})$. In this case, when the interval lays above 0, we can also say that PBC performs better than C_i in this measure.

5.5 Results from the experiment

First and foremost, we have to point out that, in the great majority of the scenarios and for both hypothesis tests, the decision to accept or reject the corresponding null hypothesis is taken with p-values higher than 30% (accepting H_0), or of the order of 10^{-5} or smaller (rejecting H_0). That is to say, there are not many indecisive situations for accepting or rejecting H_0 with p-values close to the chosen significance level.

Before starting the analysis of the results obtained, note that, since there are three measures for judging the behaviour of the crossover operators to be compared, the decision of which crossover operator behaves better is itself a multiobjective problem. Then, we have decided to make the following compromise decision: For commenting on the remaining tables we have organised the results in such a way that if the difference between the number of '$+$' and '$-$' signs in a scenario is positive, we will say that PBC "performs better than" C_i in that scenario; if that difference is negative, we will say that C_i "performs better than" PBC in that scenario; C_i and PBC performing in an equivalent way otherwise.

In Table 2, it can be observed that there is no combination of scenario and crossover for which PBC performs worse than C_i in the three measures simultaneously. It can be observed that for C_2 and C_4, under the 9 scenarios considered, PBC always obtains a behaviour strictly superior in the three measures considered. With respect to C_1, the only measure in which C_1 is superior to PBC is in the generational distance with 100 iterations with all the selection operators, and with 500 iterations with selection operator S_2; this fact is due mainly to the values obtained in this measure for problem R-DTLZ3. With respect to hypervolume and coverage, the values obtained with PBC are better than those obtained with C_1 in all the scenarios. In relation to C_3, this operator presents a better behaviour than PBC only in hypervolume in two of the nine scenarios considered, PBC being better than C_3 in the other measures in most cases and in two cases equal to C_3. Also, as it was commented at the beginning of the subsection, the confidence intervals corroborate that there are no indecisive situations in which the p-value of the test is close to the significance level.

Then, globally, we can conclude that the new operator PBC has clearly a better behaviour than C_i, $i = 1, 2, 3, 4$, when considering all the measures together, according to the compromise solution established.

Table 3 Results of the hypothesis tests performed for S-ZDT1 (left) and S-ZDT2 (right). A '+' sign means that the hypothesis "PBC performs better than C_i" is accepted. An '=' means that the hypothesis "PBC performs equally to C_i" is accepted. And a '−' represents that the hypothesis "PBC performs worse than C_i" is accepted

Sel.	Measure	S-ZDT1. PBC versus						S-ZDT2. PBC versus					
		C_1			C_2			C_1			C_2		
		No. iterations			No. iterations			No. iterations			No. iterations		
		100	500	1000	100	500	1000	100	500	1000	100	500	1000
S_1	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	=	+	+	+	+	+	=	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
S_2	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	=	+	+	+	+	+	=	+	+	+	+	+
	C	+	=	+	+	+	+	+	+	+	+	+	+
S_3	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	+	+	+	+	+	+	+	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
		C_3			C_4			C_3			C_4		
S_1	HD	+	+	=	+	+	+	+	−	−	+	+	+
	GD	+	+	+	+	+	+	+	−	−	+	+	+
	C	+	+	+	+	+	+	+	−	−	+	+	+
S_2	HD	+	+	=	+	+	+	+	+	=	+	+	+
	GD	+	+	+	+	+	+	+	+	=	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
S_3	HD	+	−	=	+	+	+	+	=	=	+	+	+
	GD	+	+	+	+	+	+	+	=	=	+	+	+
	C	+	=	+	+	+	+	+	+	=	+	+	+

However, when considering the problems separately, there exist some scenarios for some problems in which the behaviour of PBC is not the best, and then, we comment on the previous results separately for each problem. If we observe Figs. 3 to 11 and Tables 3 to 6, we can conclude the following:

S-ZDT1 If we observe Figs. 3, 6 and 9 which correspond to HD and with regard to this problem, our operator PBC obtains better values than C_1, C_2 and C_4. For C_3, the values are too close to appreciate the difference in the figures. This also happens in Figs. 4, 7 and 10 due to the high values of GD obtained for problems S-ZDT4, R-DTLZ2 and R-DTLZ3. With respect to the coverage measure, Figs. 5, 8 and 11 clearly show that PBC presents better behaviour than the other operators. In a more rigorous way, if we look at the results of the hypothesis tests performed in Table 3, we can conclude that for this problem PBC clearly outperforms C_i, $i = 1, 2, 3, 4$. There is only one scenario in which a '−' sign appears, which corresponds to C_3,

Table 4 Results of the hypothesis tests performed for S-ZDT4 (left) and R-ZDT4 (right). A '+' sign means that the hypothesis "PBC performs better than C_i" is accepted. An '=' means that the hypothesis "PBC performs equally to C_i" is accepted. And a '−' represents that the hypothesis "PBC performs worse than C_i" is accepted

Sel.	Measure	S-ZDT4. PBC versus						R-ZDT4. PBC versus					
		C_1			C_2			C_1			C_2		
		No. iterations			No. iterations			No. iterations			No. iterations		
		100	500	1000	100	500	1000	100	500	1000	100	500	1000
S_1	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	+	+	+	+	+	+	+	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	=	=	=
S_2	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	−	+	+	+	+	+	+	+	+	=	+	+
	C	+	+	+	+	+	+	+	+	+	=	+	+
S_3	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	+	+	+	+	+	+	+	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
		C_3			C_4			C_3			C_4		
S_1	HD	+	+	=	+	+	+	+	=	=	+	+	+
	GD	+	+	=	+	+	+	+	=	=	+	+	+
	C	+	+	=	+	+	+	+	=	=	+	+	+
S_2	HD	+	+	=	+	+	+	=	+	+	+	+	+
	GD	+	+	+	+	+	+	=	+	+	+	+	+
	C	+	+	+	+	+	+	=	+	+	+	+	+
S_3	HD	=	+	+	+	+	+	+	+	+	+	+	+
	GD	=	+	+	+	+	+	=	+	+	+	+	+
	C	=	+	+	+	+	+	=	+	+	+	+	+

but the other two measures have a '+' and an '=' sign, and so PBC and C_3 can be considered equivalent in this scenario. In the other scenarios, the number of '+' signs is 2 or 3.

S-ZDT2 Figures 3, 6 and 9 for HD and Figs. 5, 8 and 11 for coverage show that PBC only alternates the first position with C_3, and only in two scenarios (S_1 with 500 and 1000 iterations). When considering the hypothesis tests performed, as Table 3 shows, for this problem PBC clearly outperforms C_i, $i = 1, 2, 4$. C_3 outperforms PBC in scenarios (S_1, 500 and 1000 iterations) for all the measures. In the other seven scenarios, PBC clearly outperforms C_3 in 4 cases, PBC performs better than C_3 in two cases, and both are equivalent in one case.

S-ZDT4 If we observe the values of HD corresponding to this problem in Figs. 3, 6 and 9, it can be noticed that PBC obtains better values than C_1, C_2 and C_4. With respect to C_3, it presents values which are close but, in general, slightly worse than those obtained with PBC. Something similar happens with respect to GD (Figs. 4,

Table 5 Results of the hypothesis tests performed for S-ZDT6 (left) and S-DTLZ2 (right). A '$+$' sign means that the hypothesis "PBC performs better than C_i" is accepted. An '$=$' means that the hypothesis "PBC performs equally to C_i" is accepted. And a '$-$' represents that the hypothesis "PBC performs worse than C_i" is accepted

Sel.	Measure	S-ZDT6. PBC versus						S-DTLZ2. PBC versus					
		C_1			C_2			C_1			C_2		
		No. iterations			No. iterations			No. iterations			No. iterations		
		100	500	1000	100	500	1000	100	500	1000	100	500	1000
S_1	HD	+	+	+	+	+	+	+	+	+	+	=	−
	GD	+	+	+	+	+	+	+	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
S_2	HD	+	+	+	+	+	+	+	+	+	+	−	−
	GD	+	+	+	+	+	+	+	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
S_3	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	+	+	+	+	+	+	+	+	+	+	+	+
	C	+	+	+	+	+	+	+	+	+	+	+	+
		C_3			C_4			C_3			C_4		
S_1	HD	=	−	−	+	+	+	+	−	−	+	+	+
	GD	=	−	−	+	+	+	+	+	+	+	+	+
	C	=	−	−	+	+	+	+	+	+	+	+	+
S_2	HD	+	=	=	+	+	+	+	−	−	+	+	+
	GD	+	=	=	+	+	+	+	+	+	+	+	+
	C	+	=	+	+	+	+	+	+	+	+	+	+
S_3	HD	+	−	−	+	+	+	+	+	+	+	+	+
	GD	+	−	−	+	+	+	+	+	+	+	+	+
	C	+	−	−	+	+	+	+	+	+	+	+	+

7 and 10) and coverage (Figs. 5, 8 and 11). On the other hand, the hypothesis tests presented in Table 4 show that PBC clearly performs better than C_i, $i = 1, 2, 3, 4$ for all the scenarios, except when compared to C_3 in scenarios (S_1, 1000 iterations) and (S_3, 1000 iterations) where they can be considered equivalent.

R-ZDT4 It can be observed in Figs. 3, 4, 6, 7, 9, and 10 that the only operator that presents values of these measures close but, in general, slightly worse than those obtained with PBC is C_3. With respect to coverage, in Figs. 5, 8 and 11, the only operators and scenarios that present values better than those obtained with PBC are C_2 (S_1, 100 iterations) and C_3 (S_1 500 and 1000 iterations). Also, attending to the results of Table 4, we can say that PBC clearly performs better than C_i, $i = 1, 2, 4$ in all the scenarios. With respect to C_3, PBC clearly outperforms C_3 in five scenarios, in one scenario PBC performs better than C_3 and in the remaining three scenarios PBC and C_3 are equivalent.

Table 6 Results of the hypothesis tests performed for R-DTLZ2 (left) and R-DTLZ3 (right). A '+' sign means that the hypothesis "PBC performs better than C_i" is accepted. An '=' means that the hypothesis "PBC performs equally to C_i" is accepted. And a '−' represents that the hypothesis "PBC performs worse than C_i" is accepted

Sel.	Measure	R-DTLZ2. PBC versus						R-DTLZ3. PBC versus					
		C_1			C_2			C_1			C_2		
		No. iterations			No. iterations			No. iterations			No. iterations		
		100	500	1000	100	500	1000	100	500	1000	100	500	1000
S_1	HD	+	+	+	+	+	+	+	+	+	+	+	+
	GD	−	+	+	+	+	+	−	−	+	+	+	+
	C	+	+	+	+	+	+	−	+	+	+	+	+
S_2	HD	+	+	+	+	+	+	−	+	+	+	+	+
	GD	−	−	+	=	=	+	−	−	+	+	+	+
	C	=	+	+	+	+	+	−	+	+	+	+	+
S_3	HD	+	+	+	+	+	+	−	+	+	+	+	+
	GD	-	+	+	+	+	+	−	+	+	+	+	+
	C	+	+	+	+	+	+	−	+	+	+	+	+
		C_3			C_4			C_3			C_4		
S_1	HD	+	+	=	+	+	+	+	+	+	+	+	+
	GD	+	+	=	=	+	+	+	+	+	+	+	+
	C	+	+	+	−	=	+	+	+	+	+	+	=
S_2	HD	+	+	+	=	+	+	+	+	+	+	+	+
	GD	+	+	+	−	−	−	+	+	+	+	+	+
	C	+	+	+	−	−	=	+	+	+	+	+	+
S_3	HD	+	+	+	−	+	+	+	+	+	+	+	+
	GD	+	+	+	−	+	+	+	+	+	+	+	+
	C	+	+	+	−	+	+	+	+	+	+	+	+

S-ZDT6 After examining Figs. 3 to 11 and Table 5, we observe that when comparing PBC with C_i, $i = 1, 2, 4$, PBC clearly outperforms the other operators since all the signs are '+'. When comparing PBC with C_3, C_3 outperforms PBC in four scenarios, PBC outperforms C_3 clearly in two scenarios. So, globally, in this problem, C_3 outperforms PBC.

S-DTLZ2 In Figs. 3, 6 and 9, it can be noticed that the operator C_2 obtains better values in the measure HD than PBC in three out of nine scenarios, and C_3 obtains better values than PBC in four out of nine scenarios. However, with respect to coverage, in Figs. 5, 8 and 11, it can be observed that the values obtained with PBC are better than those obtained with all the other operators. These facts are also reflected in the hypothesis tests of Table 5. For this problem, PBC performs better than C_i, $i = 1, 2, 3, 4$ in all the scenarios, with only some '−' signs appearing in the measure HD.

R-DTLZ2 With respect to HD it is not possible to distinguish the values in Figs. 3, 6 and 9. For GD, PBC shows bad results with few iterations which are improved when the number of iterations increases. For 100 iterations, C_1 and C_4 obtain better results with all the selection operators considered; for 500 iterations, C_1 and C_4 are better only with S_2; and when 1000 iterations are considered, only C_4 with S_2 remains better than PBC. With respect to coverage, Figs. 5, 8 and 11 show that C_4 is the only operator that obtains, in four cases, better values than PBC (S_1, S_2 and S_3 with 100 iterations, and S_2 with 500 iterations). From Table 6 we obtain that in four scenarios the operator PBC outperforms C_4. In two scenarios, C_4 outperforms PBC and in one it performs better that PBC. In the remaining scenarios, PBC and C_4 are equivalent. With respect to C_i, $i = 1, 2, 3$, PBC outperforms all of them.

R-DTLZ3 It can be observed in Figs. 3, 6 and 9 that the values obtained for HD with PBC are better than those obtained with C_i, $i = 1, 2, 3, 4$, except when selection S_3 is used. In this case, all the values of HD are very close to each other. With respect to coverage, in Figs. 5, 8 and 11 it can be observed that the only operator that obtains better values than PBC is C_1 with 100 iterations. Looking at Table 6, PBC clearly outperforms C_i, $i = 2, 3, 4$. With respect to C_1, this operator clearly outperforms PBC in all scenarios with 100 iterations. But with 500 and 1000 iterations, PBC outperforms C_1.

In the light of these reports, none of the crossover operators considered behaves better than PBC for every problem. Furthermore, C_1 always performs worse than PBC except in three scenarios for problem R-DTLZ3 and both operators can be considered equivalent in one scenario for problem R-DTLZ2. PBC performs always better than C_2. C_3 does not outperform PBC in problems S-ZDT1, S-ZDT4, R-ZDT4, S-DTLZ2, R-DTLZ2, and R-DTLZ3, PBC being superior to C_3 in all of them. Only in problems S-ZDT2 and S-ZDT6, C_3 gets to outperform PBC in some scenarios (two of them in problem S-ZDT2 and four in problem S-ZDT6), and then, only problem S-ZDT6 is the one in which PBC has been outperformed by C_3. And finally, PBC performs better than C_4 except in two scenarios in which they can be considered equivalent and three scenarios in which C_4 performs better than PBC, all these cases in problem R-DTLZ2.

Therefore, globally, we can conclude that the new crossover operator, PBC, presented in this paper performs better than all the crossover operators considered.

6 Conclusions and further research

In this work, we have proposed a new crossover operator that takes into account the quality of the solution (efficient or not) and the distance between parents in relation to the mean distance among efficient solutions, and we have compared it with four standard crossover operators. From the experiments carried out, it follows that our new operator behaves better than the usual ones for the selected test problems. Except for some scenarios, the operator PBC obtains better values of the measures considered (hypervolume difference, generational distance and set coverage) than the other operators.

Considering the mutation operator proposed by the authors in Alberto and Mateo (2008) and this new crossover operator, the next step is to carry out a study of some of the well-established algorithms in the literature (NSGA-II Deb et al. 2002, SPEA2 Zitzler et al. 2001, etc.) by substituting the operators proposed by their authors with our operators and some of the currently most promising algorithms as those based on Differential Evolution for Multiobjective Optimisation. This approach was first introduced for uniobjective optimisation problems by Storn and Price (1995) and subsequently adapted for multiobjective optimisation problems. An updated survey can be found in Mezura-Montes et al. (2008).

Acknowledgements This research work has been supported by the Spanish Ministry of Education and Science under grant MTM2007-66893.

The authors are also grateful to both the editor and referees for their comments, which greatly improved the presentation of this paper.

References

Alberto I, Mateo PM (2008) Using Pareto optimality for defining the mutation operator step size. Technical report. In: Prepublicaciones del seminario matemático García Galdeano, University of Zaragoza, 8, pp 1–19

Casella G, Berger RL (2002) Statistical inference, 2nd edn. Duxbury advanced series. Duxbury, Pacific Grove

CEC (2007) Special session and competition on performance assessment of multi-objective optimization algorithms. Available in web page http://www3.ntu.edu.sg/home/epnsugan/. Accessed August 2008

Coello CA (2005) Recent trends in evolutionary multiobjective optimization. In: Abraham A, Jain L, Goldberg R (eds) Evolutionary multiobjective optimization: theoretical advances and applications. Springer, London, pp 7–32

Coello CA (2008) EMOO repository (online). Available in http://delta.cs.cinvestav.mx/~ccoello/EMOO/. Accessed September 2008

Coello CA, Lamont GB, Van Veldhuizen DA (2007) Evolutionary algorithms for solving multi-objective problems, 2nd edn. Springer, Berlin

Deb K (2001) Multi-objective optimization using evolutionary algorithms, 2nd edn. Wiley, Chichester

Deb K, Agrawal S (1995) Simulated binary crossover for continuous search space. Complex Syst 9(2):115–148

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Deb K, Thiele L, Laumanns M, Zitzler E (2002) Scalable multi-objective optimization test problems. In: Congress on evolutionary computation CEC'2002, vol 1. IEEE Press, Piscataway, pp 825–830

Eiben AE, Smith JE (2007) Introduction to evolutionary computing, 2nd edn. Springer, Berlin

Eshelman LJK, Schaffer JD (1993) Real-coded genetic algorithms and interval-schemata. In: Whitley LD (ed) Foundations of genetic algorithms II. Morgan Kaufmann, Los Altos, pp 187–202

Fonseca CM, Fleming PJ (1993) Genetic algorithms for multi-objective optimization: formulation, discussion and generalization. In: Forrest S (ed) Genetic algorithms: proceedings of the fifth international conference. Morgan Kaufmann, San Mateo, pp 416–423

Fonseca CM, Paquete L, Lopez-Ibanez M (2006a) An improved dimension-sweep algorithm for the hypervolume indicator. In: Congress on evolutionary computation CEC'2006. IEEE Press, Vancouver, pp 1157–1163

Fonseca CM, Paquete L, Lopez-Ibanez M (2006b) Computation of the hypervolume indicator, codes available in web page http://sbe.napier.ac.uk/~manuel/hypervolume. Accessed August 2008

Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison–Wesley, Reading

Herrera F, Lozano M, Verdegay JL (1998) Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. Artif Intell Rev 12(4):265–319

Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor

Huang VL, Qin AK, Deb K, Zitzler E, Suganthan PN, Liang JJ, Preuss M, Huband S (2007) Problem definitions for performance assessment on multiobjective optimization algorithms. Technical report CEC-TR2007, Nanyang Technological University, Singapore

Huband S, Hingston P, Barone L, While L (2007) A review of multiobjective test problems and a scalable test problem toolkit. IEEE Trans Evol Comput 10(5):477–506

Mezura-Montes E, Reyes-Sierra M, Coello CA (2008) Multi-objective optimization using differential evolution: a survey of the state-of-the-art. In: Chakraborty UK (ed) Advances in differential evolution. Springer, Berlin, pp 173–196

Michalewicz Z (1996) Genetic algorithms + data structures = evolution programs, 3rd edn. Springer, Berlin

Schaffer JD (1984) Multiple objective optimization with vector evaluated genetic algorithms. Ph.D. thesis, Vanderbilt University, Nashville, Tennessee

Schaffer JD (1985) Multiple objective optimization with vector evaluated genetic algorithms. In: Grefenstette JJ (ed) Genetic algorithms and their applications, proceedings of the first international conference on genetic. Lawrence Erlbaum, Hillsdale, pp 93–100

Storn R, Price K (1995) Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical report TR-95-12, International Computer Science, Berkeley, California, pp 1–12

Wright AH (1991) Genetic algorithms for real parameter optimization. In: Rawlins GJE (ed) Foundations of genetic algorithms. Morgan Kaufmann, San Mateo, pp 205–218

Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: methods and applications. Ph.D. thesis, dissertation ETH no 13398, Swiss Federal Institute of Technology (ETH), Zürich, Switzerland

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans Evol Comput 3(4):257–271

Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195

Zitzler E, Laumanns M, Thiele L (2001) SPEA2: improving the strength Pareto evolutionary algorithm. Technical report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institue of Technology (ETH), Zurich, Switzerland, pp 1–21