# The Roles of Crossover and Mutation in Real-Coded Genetic Algorithms

Yourim Yoon[1] and Yong-Hyuk Kim[2]*

[1]*School of Computer Science and Engineering, Seoul National University, Seoul*
[2]*Department of Computer Science and Engineering, Kwangwoon University, Seoul*
*Republic of Korea*

## 1. Introduction

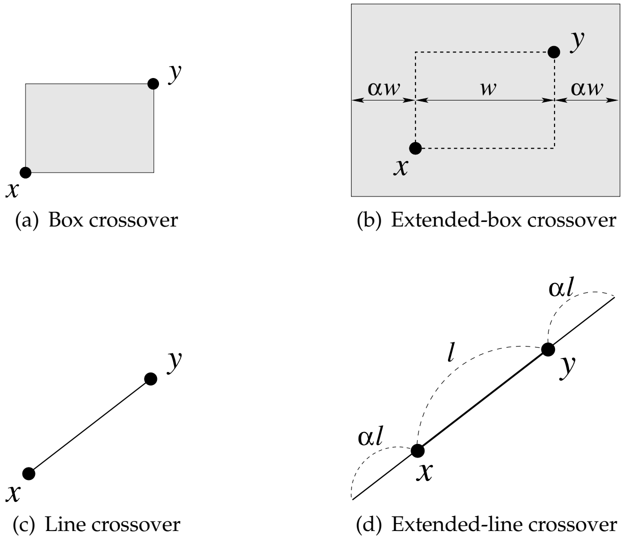
We recognized that the roles of crossover and mutation in real encoding are quite different from those in binary encoding during performing previous work with real-coded genetic algorithms (Yoon et al., 2012). In this study, we are to argue the distinct roles of genetic operators in real encodings.

Recently many studies on evolutionary algorithms using real encoding have been done. They include ant colony optimization (Socha & Dorigo, 2008), artificial bee colony algorithm (Akay & Karaboga, 2010; Kang et al., 2011), evolution strategies (ES) (Beyer, 2001), differential evolution (Das & Suganthan, 2011; Dasgupta et al., 2009; Kukkonen & Lampinen, 2004; 2005; Mezura-Montes et al., 2010; Noman & Iba, 2005; Rönkkönen et al., 2005; Storn & Price, 1997; Zhang et al., 2008), particle swarm optimization (Chen et al., 2007; Huang et al., 2010; Juang et al., 2011; Krohling & Coelho, 2006; l. Sun et al., 2011), and so on. In particular, in the field of ES, we can find many studies based on self-adaptive techniques (Beyer & Deb, 2001; Hansen & Ostermeier, 2001; Igel et al., 2007; 2006; Jägersküpper, 2007; Kita, 2001; Kramer, 2008a;b; Kramer et al., 2007; Meyer-Nieberg & Beyer, 2007; Wei et al., 2011).

Many researchers have also concentrated on using real-valued genes in genetic algorithms (GAs), as in (Ripon et al., 2007). It is reported that, for some problems, real-coded representation and associated techniques outperform conventional binary representation (Eshelman & Schaffer, 1993; Herrera et al., 1998; Janikow & Michalewicz, 1991; Lozano et al., 2004; Ono et al., 1999; Ono & Kobayashi, 1997; Surry & Radcliffe, 1996; Wright, 1991). Several theoretical studies of real-coded GAs have also been performed (Goldberg, 1991; Higuchi et al., 2000; Kita et al., 1998; Qi & Palmieri, 1994a;b). However, the role and behavior of genetic operators in real-coded GAs are fundamentally different from those in binary encodings although motivation of the operators and the framework of GAs are similar.

In this chapter, we try to verify different properties of crossover and mutation in real encodings from those in binary encodings through various experiments. We especially concentrate on the effect of genetic operators (the bias and functions of crossover and mutation) when they are used in real-coded GAs.

*Corresponding author: Yong-Hyuk Kim



(a) Box crossover

(b) Extended-box crossover

(c) Line crossover

(d) Extended-line crossover

Fig. 1. The range of possible offspring in two-dimensional bounded real space

The remainder of this chapter is organized as follows. Traditional and recent genetic operators in real encoding are introduced in Section 2. Previous genetic operators are presented in Section 2.1 and ones we used in real encoding in this study are described in Section 2.2. In Section 3, we describe the concept of bias of genetic operators and analyze that in the case of crossover and mutation for GAs. In Section 4, experimental results for various combinations of crossover and mutation are provided and analyzed. Finally, we make conclusions in Section 5.

## 2. Genetic operators in real encoding

### 2.1 Previous operators

The roles of crossover and mutation may change according to the selection of the operators. We reviewed the most frequently used crossover and mutation operators for real-code representation. We are to analyze how the roles of crossover and mutation can change by studying various combinations of crossover and mutation operators.

In literature many crossover operators for real-code representation are found. Traditional crossover operators for the real-code representation are described in (Bäck et al., 2000). The two main families of traditional crossover operators (Mühlenbein & Schlierkamp-Voosen, 1993) are discrete crossovers[1] (Reed et al., 1967) and blend crossovers (Michalewicz, 1996). Blend crossover operators can be distinguished into line crossovers and box crossovers. Important variations of the last two crossover operators are the extended-line crossover and the extended-box crossover (Mühlenbein, 1994).

The discrete recombination family is the straightforward extension to real vectors of the family of mask-based crossover operators for binary strings including $n$-point and uniform crossover.

[1] It is also called *dominant crossover*.

```
box-crossover$(x, y)$
{
        for $i \leftarrow 1$ to $n$
            $z_i \leftarrow$ a random real number in $[\min(x_i, y_i), \max(x_i, y_i)]$;
        return $z = (z_1, z_2, \ldots, z_n)$;
}
```

Fig. 2. Pseudo-code of box crossover

```
extended-box-crossover$(x, y)$
{
        for $i \leftarrow 1$ to $n$
            $m \leftarrow \min(x_i, y_i), M \leftarrow \max(x_i, y_i)$;
            $em \leftarrow m - \alpha(M - m), eM \leftarrow M + \alpha(M - m)$;
            $z_i \leftarrow$ a random real number in $[\min(em, l_i), \max(eM, u_i)]$;
        return $z = (z_1, z_2, \ldots, z_n)$;
}
// $\alpha$ is extension rate.
```

Fig. 3. Pseudo-code of extended-box crossover

The mask is still a binary vector dictating for each position of the offspring vector from which parent the (real) value for that position is taken.

The blend recombination family does not exchange values between parents like discrete recombinations but it averages or blends them. Line recombination returns offspring on the (Euclidean) line segment connecting the two parents. Box recombination returns offspring in the box (hyper-rectangle) whose diagonally opposite corners are the parents. Extended-line recombination picks offspring on an extended segment passing through the parent vectors but extending beyond them and not only in the section between them. Analogously extended-box recombination picks offspring on an extended box whose main diagonal passes through the parents but extends beyond them.

Recently several new crossovers for the real-coded representation have been designed. Several non-traditional crossover operators for real-coded representation are found in the recent literature. They include SBX (simulated binary crossover) (Ballester & Carter, 2003; 2004b; Deb & Agrawal, 1995; Deb & Beyer, 1999; Deb & Kumar, 1995; Deb et al., 2007), UNDX (unimodal normal distribution crossover) (Kita et al., 1998; 1999; Ono et al., 1999; Ono & Kobayashi, 1997), SPX (simplex crossover) (Higuchi et al., 2000; Tsutsui & Goldberg, 2002; Tsutsui et al., 2001; 1999), PCX (parent-centric crossover) (Ballester & Carter, 2004a; Deb et al., 2002), etc (Herrera et al., 2003; 2005; Takahashi & Kita, 2001). Most of them are complex and based on the specific probability distribution of the offspring (SBX, UNDX, and PCX), self-adaptivity (SBX and UNDX), or multiple parents (UNDX and SPX). Some of them, e.g., include the function of mutation operators. In this study, we focus on traditional crossover that does not consider the specific probability distribution of the offspring but only what offspring can be generated with a probability greater than zero, given the two parents.

```
line-crossover$(x, y)$
{
    $\lambda \leftarrow$ a random real number in $[0, 1]$;
    for $i \leftarrow 1$ to $n$
        $z_i \leftarrow \lambda x_i + (1 - \lambda) y_i$;
    return $z = (z_1, z_2, \ldots, z_n)$;
}
```

Fig. 4. Pseudo-code of line crossover

```
extended-line-crossover$(x, y)$
{
    $m \leftarrow -\infty, M \leftarrow \infty$;
    for $i \leftarrow 1$ to $n$
        if $x_i \neq y_i$
            $t_l \leftarrow (l_i - y_i)/(x_i - y_i), t_u \leftarrow (u_i - y_i)/(x_i - y_i)$;
            $t_m \leftarrow \min(t_l, t_u), t_M \leftarrow \max(t_l, t_u)$;
            $m \leftarrow \max(m, t_m), M \leftarrow \min(M, t_M)$;
    $\lambda \leftarrow$ a random real number in $[\max(m, -\alpha), \min(M, 1 + \alpha)]$;
    for $i \leftarrow 1$ to $n$
        $z_i \leftarrow \lambda x_i + (1 - \lambda) y_i$;
    return $z = (z_1, z_2, \ldots, z_n)$;
}
// $\alpha$ is extension rate.
```

Fig. 5. Pseudo-code of extended-line crossover

The most common form of mutation for real-code vectors generates an offspring vector by adding a vector $M$ of random variables with expectation zero to the parent vector. There are two types of mutations bounded and unbounded depending on the fact that the range of the random variable is bounded or unbounded. The most frequently used bounded mutations are the creep mutation and the single-variable mutation and for the unbounded case is the Gaussian mutation. For the creep (or hyper-box) mutation $M \sim U([-a, a]^n)$ is a vector of uniform random variables, where $a$ is a parameter defining the limits of the offspring area. This operator yields offspring within a hyper-box centered in the parent vector. For the single-variable mutation $M$ is a vector in which all entries are set to zero except for a random entry which is a uniform random variable $\sim U([-a, a])$. Bounded mutation operators may get stuck in local optima. In contrast, unbounded mutation operators guarantee asymptotic global convergence. The primary unbounded mutation is the Gaussian mutation for which $M$ is a multivariate Gaussian distribution.

### 2.2 Adopted operators for this study

As crossover operators for our analysis, we adopted four representative crossovers: box, extended-box, line, and extended-line crossovers. Their pseudo-codes are shown in Figures 2, 3, 4, and 5, respectively and the possible range for each crossover is represented in Figure 1.

```
mutation($z, p$)
{
        for $i \leftarrow 1$ to $n$
                if a random number from $[0, 1]$ is less than mutation rate $p$
                    $z_i \leftarrow z_i + N(0, (u_i - l_i)/10)$;
        return $z = (z_1, z_2, \ldots, z_n)$;
}
```

Fig. 6. Pseudo-code of mutation

```
// $x$ and $y$ are parents.
fine-mutation($x, y, z, p$)
{
        for $i \leftarrow 1$ to $n$
                if a random number from $[0, 1]$ is less than mutation rate $p$
                    $z_i \leftarrow z_i + N(0, |x_i - y_i|)$;
        return $z = (z_1, z_2, \ldots, z_n)$;
}
```

Fig. 7. Pseudo-code of fine mutation

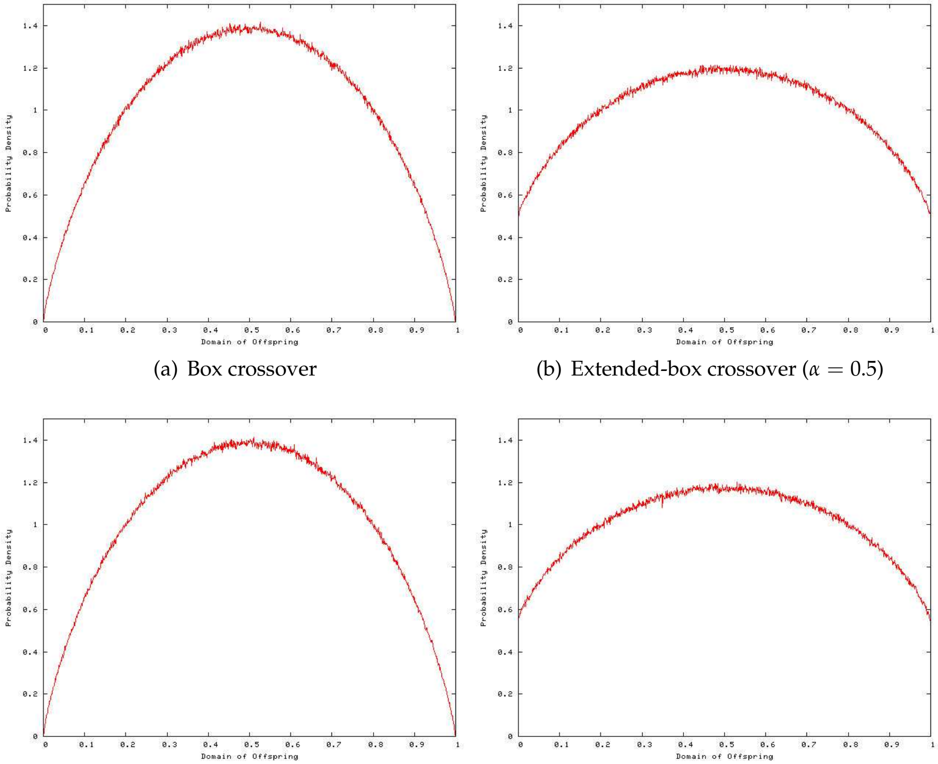
And, as mutation operators for our analysis, we adopted two kinds of mutation: Gaussian mutation and fine mutation. Their pseudo-codes are shown in Figures 6 and 7, respectively. The Gaussian mutation is a simple static Gaussian mutation, the same as in Tsutsui & Goldberg (2001). The *i*-th parameter $z_i$ of an individual is mutated by $z_i = z_i + N(0, \sigma_i)$ with a mutation rate $p$, where $N(0, \sigma_i)$ is an independent random Gaussian number with the mean of zero and the standard deviation of $\sigma_i$. In our study, $\sigma_i$ is fixed to $(u_i - l_i)/10$ - the tenth of width of given area. The fine mutation is a simple dynamic Gaussian mutation inspired from Ballester & Carter (2004b). In different with Gaussian mutation, it depends on the distance between parents and, as population converges, the strength of the mutation approaches zero.

## 3. Bias of genetic operators

Pre-existing crossovers for the real-coded representation have an inherent bias toward the center of the space. Some boundary extension techniques to reduce crossover bias have been extensively studied (Someya & Yamamura, 2005; Tsutsui, 1998; Tsutsui & Goldberg, 2001). The concept of crossover *bias* first appeared in (Eshelman et al., 1997) and it has been extensively used in (Someya & Yamamura, 2005; Tsutsui & Goldberg, 2001), in which they tried to remove the bias of real-coded crossover heuristically (and theoretically incompletely).

Notice that the notion of bias of a crossover operator has different definitions depending upon the underlying representation considered. The bias toward the center of the space considered in real-coded crossovers conceptually differs from the crossover biases on binary strings, which focus on how many bits are passed to the offspring and from which positions, which, in turn conceptually differs from the bias considered in Genetic Programming focusing on bloat.

The notion of bias so defined can be understood as being the inherent preference of a search operator for specific areas of the search space. This is an important search property of a search



(a) Box crossover

(b) Extended-box crossover ($\alpha = 0.5$)

(c) Line crossover

(d) Extended-line crossover ($\alpha = 0.5$)

Fig. 8. Crossover bias in one-dimensional bounded real space

operator: an evolutionary algorithm using that operator, without selection, is attracted to the areas the search operator prefers. Arguably, also when selection is present, the operator bias acts as a background force that makes the search keener to go toward the areas preferred by the search operator. This is not necessarily bad if the bias is toward the optimum or an area with high-quality solutions. However, it may negatively affect performance if the bias is toward an area of poor-quality solutions. If we do not know the spatial distribution of the fitness of the problem, we may prefer not to have any *a priori* bias of the search operator, and instead use only the bias of selection, which is informed by the fitness of sampled solutions that constitute empirical knowledge about promising areas obtained in the search, and which is better understood.

In this chapter, we investigate the bias caused by crossover itself and crossover combined with mutation in real-coded GAs. Intuitively, box and line crossover are biased toward the center on the Euclidean space. This intuition is easy to verify experimentally by picking a large number of pairs (ideally infinitely many) of random parents and generating offspring uniformly at random in the boxes (or lines) identified by the pairs of parents.

| Function | $n$ | Range of $x_i$: $[l_i, u_i]$ |
|---|---|---|
| Shifted sphere<br>$\sum_{i=1}^{n}(x_i - o_i)^2 - 450$ | 30 | $[-100, 100]$ |
| Shifted Schwefel<br>$\sum_{i=1}^{n}(\sum_{j=1}^{i}(x_j - o_j))^2 - 450$ | 30 | $[-100, 100]$ |
| Shifted Rosenbrock<br>$\sum_{i=1}^{n-1}(100((x_i - o_i + 1)^2 - (x_{i+1} - o_{i+1} + 1))^2 + (x_i - o_i)^2) + 390$ | 30 | $[-100, 100]$ |
| Shifted Rastrigin<br>$\sum_{i=1}^{n}((x_i - o_i)^2 - 10\cos(2\pi(x_i - o_i)) + 10) - 330$ | 30 | $[-5, 5]$ |

Table 1. Test Functions
$\boldsymbol{o} = (o_1, o_2, \ldots, o_n)$ is the optimal solution, which is randomly located in the domain.

In the Hamming space, the distribution of the offspring of uniform crossover tends in the limit to be uniform on all space, whereas in the Euclidean space the distribution of the offspring tends to be unevenly distributed on the search space and concentrates toward the center of the space. One way to compensate, but not eliminate, such bias is using extended-line and extended-box crossovers. Figure 8 visualizes the crossover bias in the one-dimensional real space by plotting frequency rates of $10^7$ offspring randomly generated by each type crossover. As we can see, box and line crossover are biased toward the center of the domain. We could also observe that extended-box and extended-line crossover largely reduce the bias but they are still biased toward the center.[2]

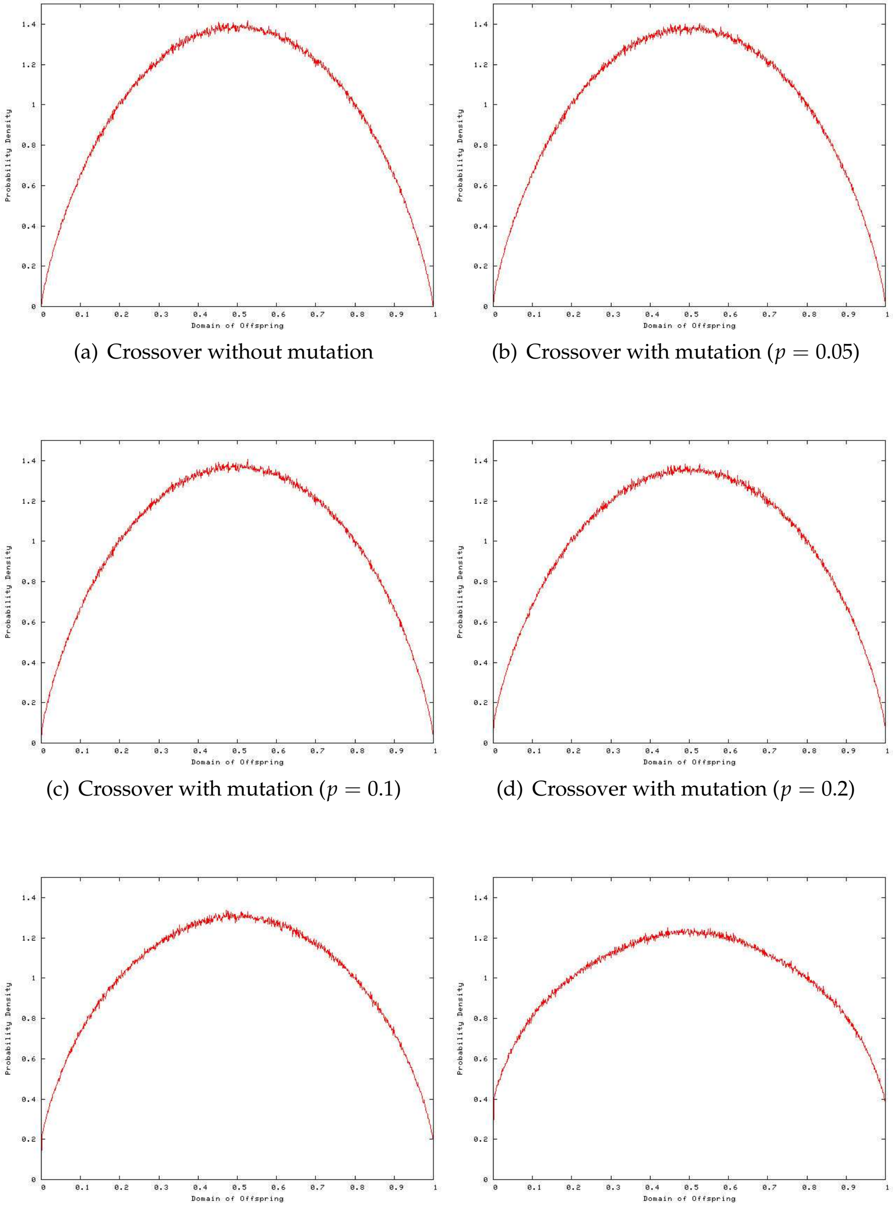
For analyzing the effect of mutation in relation with the bias, we also performed the same test using crossover combined with Gaussian mutation. We picked $10^7$ pairs of random parents, generated offspring randomly using each type crossover, and then applied Gaussian mutation. The tests are performed for various mutation rates from 0.0 to 1.0. The results for box, extended-box, line, and extended-line crossover are shown in Figures 9, 10, 11, and 12, respectively. Interestingly, for all cases, we could observe that the higher mutation rate reduces the bias more largely. However, even high mutation rates cannot eliminate the bias completely.

## 4. Combination of crossover and mutation

In this section, we try to figure out the properties of crossover and mutation through experiments using their various combinations. For our experiments, four test functions are chosen from Suganthan et al. (2005). They are described in Table 1.

We mainly followed the genetic framework by Tsutsui & Goldberg (2001). Its basic evolutionary model is quite similar to that of CHC (Eshelman, 1991) and $(\mu + \lambda)$-ES (Beyer, 2001).

[2] We can find consistent results with this in Someya & Yamamura (2005); Yoon et al. (2012).



(a) Crossover without mutation

(b) Crossover with mutation ($p = 0.05$)

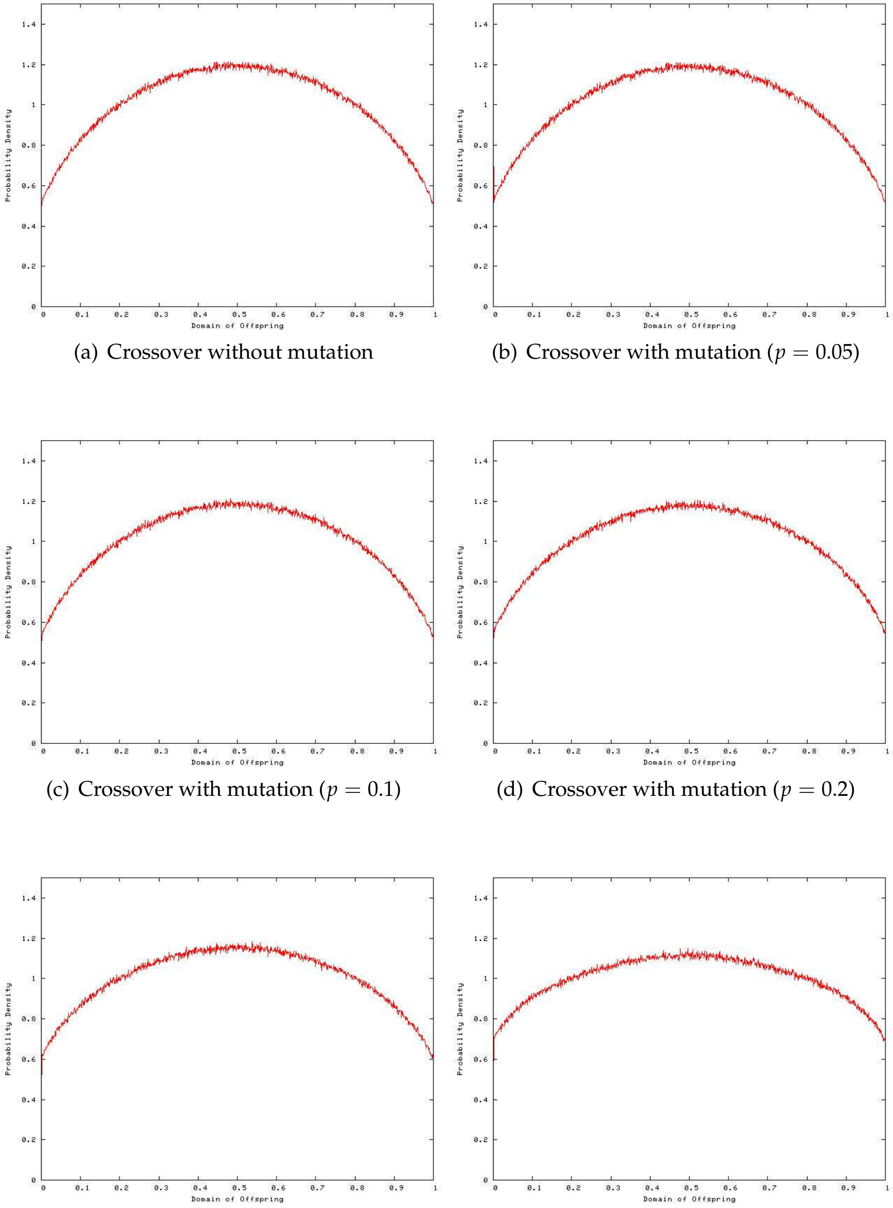
(c) Crossover with mutation ($p = 0.1$)

(d) Crossover with mutation ($p = 0.2$)

(e) Crossover with mutation ($p = 0.5$)

(f) Crossover with mutation ($p = 1.0$)

Fig. 9. Bias of box crossover with mutation



(a) Crossover without mutation

(b) Crossover with mutation ($p = 0.05$)

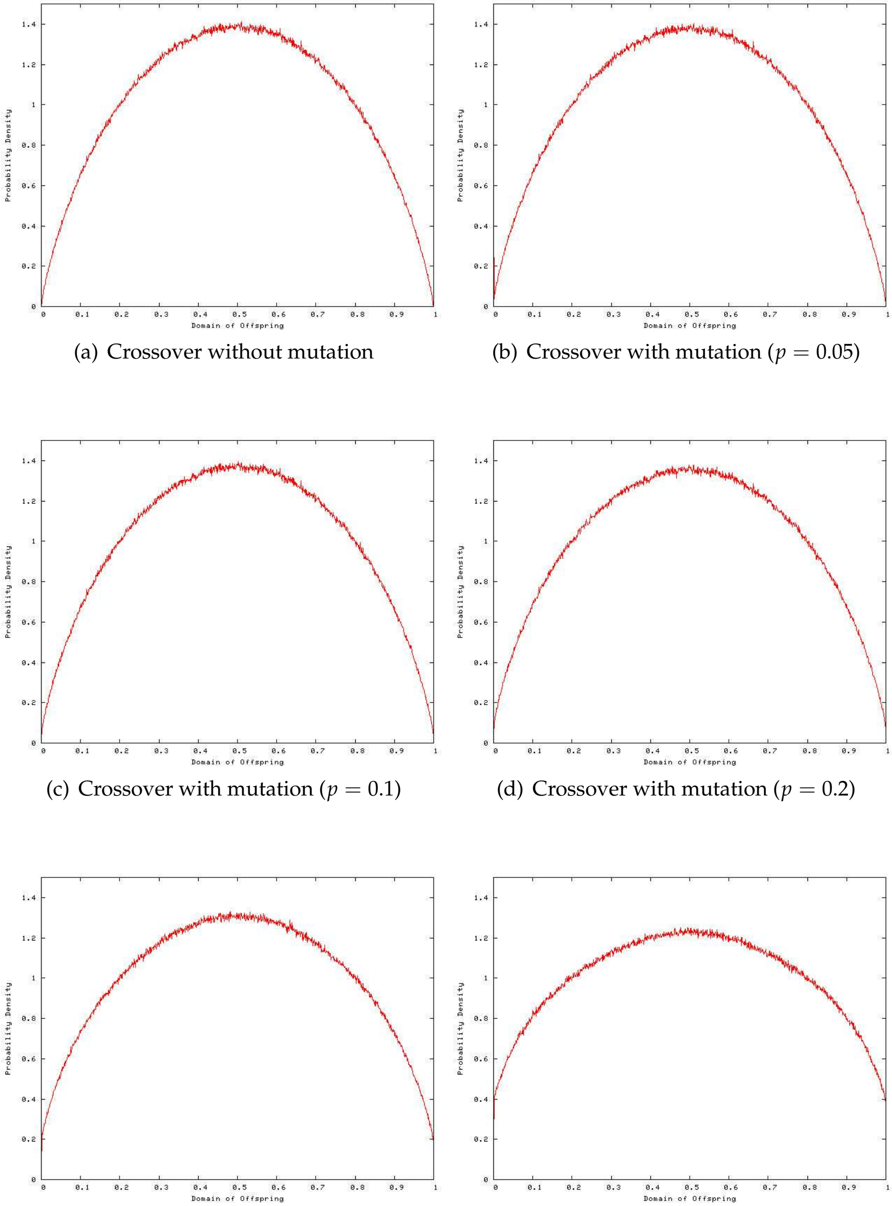
(c) Crossover with mutation ($p = 0.1$)

(d) Crossover with mutation ($p = 0.2$)

(e) Crossover with mutation ($p = 0.5$)

(f) Crossover with mutation ($p = 1.0$)

Fig. 10. Bias of extended-box crossover ($\alpha = 0.5$) with mutation



(a) Crossover without mutation

(b) Crossover with mutation ($p = 0.05$)

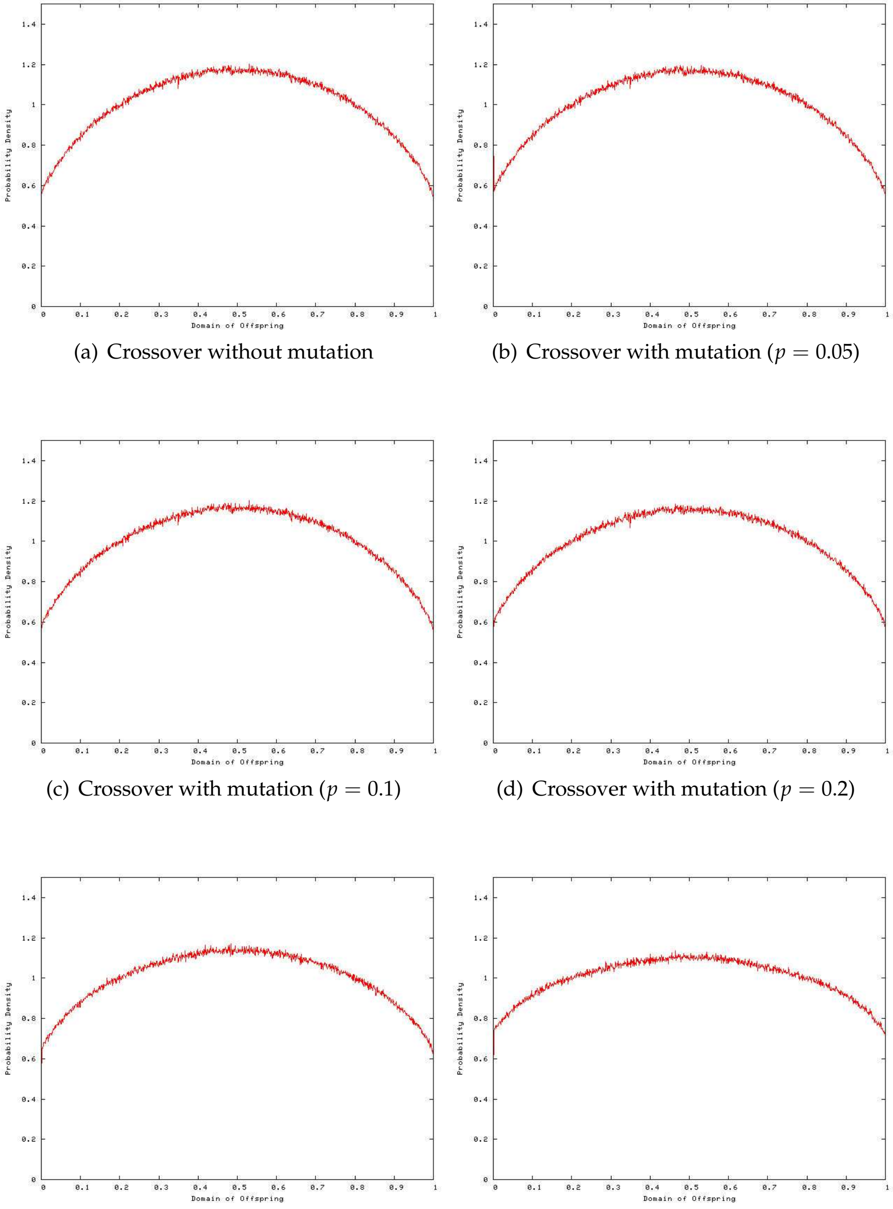
(c) Crossover with mutation ($p = 0.1$)

(d) Crossover with mutation ($p = 0.2$)

(e) Crossover with mutation ($p = 0.5$)

(f) Crossover with mutation ($p = 1.0$)

Fig. 11. Bias of line crossover with mutation



(a) Crossover without mutation

(b) Crossover with mutation ($p = 0.05$)

(c) Crossover with mutation ($p = 0.1$)

(d) Crossover with mutation ($p = 0.2$)

(e) Crossover with mutation ($p = 0.5$)

(f) Crossover with mutation ($p = 1.0$)

Fig. 12. Bias of extended-line crossover ($\alpha = 0.5$) with mutation

Let $N$ the population size. A collection of $N/2$ pairs is randomly composed, and crossover and mutation are applied to each pair, generating $N/2$ offspring. Parents and newly generated offspring are ranked and the best $N$ individuals among them are selected for the population in the next generation. The population size was 400 for all experiments. If the population has no change during $n \times r \times (1.0 - r)$ generations, it is reinitialized except for the best individual. Here, $r$ is a divergence rate and we set it to 0.25 as in Eshelman (1991). The used GA terminates when it finds the global optimum.

For crossover, we used four crossover operators: box crossover, extended-box crossover (extension rate $\alpha$: 0.5), line crossover, and extended-line crossover (extension rate $\alpha$: 0.5). After crossover, we either mutate the offspring or do not. We used two different mutation operators; Gaussian mutation and fine mutation. Different mutation rates were applied to each crossover type and the rates decrease as the number of generations increases.

Table 2 shows the results from 30 runs. Each value in 'Ave' means the average function value from 30 runs. The smaller, the better. The limit of function evaluations is 50,000, i.e., the genetic algorithm terminates after 50,000 evaluations and outputs the best solution among evaluated ones so far over generations. In the table, $k = 1 + \lfloor numberOfGenerations/100 \rfloor$ and the rate of fine mutation is $0.5/k$.

From these experiments we can obtain the following properties.

- There is no superior operator combination for all over the problem instances. For the shifted sphere, box crossover with fine mutation showed the best performance. For the shifted Schwefel, line crossover with Gaussian mutation, for the shifted Rosenbrock, extended-box crossover without mutation, and for the shifted Rastrigin, box crossover with fine mutation showed the best performances, respectively. So we can know that suitable crossover and mutation can be varied depending on the property of given problem.
- Without mutation, extended-box crossover showed the best performance. That is, when we do not know the characteristic of given problem, it is a general choice that we use extended-box crossover as a crossover operator in real-coded genetic algorithms. It is convenient since parameter tuning with mutation is not required. However, it is possible to surpass the performance of extended-box crossover using well-designed combination of crossover and mutation.
- Unusually, for extended-box crossover, the results without mutation is the best and the performance becomes worse as mutation rate increases. However, for box crossover, moderate rate of mutation has a good effect to the performance. For all cases, box crossover with mutation showed better performance than that without mutation. From this fact, we can infer that extended-box crossover contains the function of mutation in itself but box crossover does not.
- Except for extended-box crossover, the results of crossover with mutation were better than those of crossover without mutation. In particular, fine mutation was better than Gaussian mutation. Fine mutation depends on the distance between parents so, as population converges, the strength of the mutation approaches zero. That is, the amount of mutation becomes very fine as population converges. In binary encodings, the main role of mutation is perturbation effect to prevent premature convergence. However, we can know that, in real encoding, the function of fine tuning by mutation is also important from this experiment.

| | Function | Shifted Sphere | Shifted Schwefel | Shifted Rosenbrock | Shifted Rastrigin |
|---|---|---|---|---|---|
| Crossover | Mutation | Ave ($\sigma/\sqrt{n}$) | Ave ($\sigma/\sqrt{n}$) | Ave ($\sigma/\sqrt{n}$) | Ave ($\sigma/\sqrt{n}$) |
| None | $0.05/k$ | 5.46e+02 (4.59e+01) | 1.68e+04 (9.05e+02) | 1.87e+07 (1.70e+06) | -2.66e+02 (1.91e+00) |
| | $0.10/k$ | **6.40e+01** (2.17e+01) | 1.16e+04 (5.33e+02) | **4.64e+06** (3.86e+05) | **-2.68e+02** (1.65e+00) |
| | $0.20/k$ | 1.37e+02 (1.55e+01) | 7.99e+03 (4.79e+02) | 4.74e+06 (2.34e+05) | -2.64e+02 (1.58e+00) |
| | $0.50/k$ | 9.87e+02 (3.13e+01) | **7.87e+03** (3.18e+02) | 1.89e+07 (8.19e+05) | -2.27e+02 (1.28e+00) |
| | $1.00/k$ | 2.38e+03 (6.45e+01) | 1.24e+04 (3.76e+02) | 7.12e+07 (2.45e+06) | -1.91e+02 (1.72e+00) |
| Box crossover | None | 2.46e+04 (4.46e+02) | 2.47e+04 (3.03e+02) | 5.36e+09 (1.69e+08) | -1.83e+02 (1.64e+00) |
| | $0.05/k$ | -3.61e+02 (1.81e+00) | 1.44e+04 (1.96e+02) | 2.66e+05 (8.26e+03) | -2.53e+02 (1.51e+00) |
| | $0.10/k$ | -4.02e+02 (1.13e+00) | 1.22e+04 (1.42e+02) | 7.86e+04 (2.26e+03) | **-2.60e+02** (1.11e+00) |
| | $0.20/k$ | **-4.25e+02** (6.96e-01) | 9.98e+03 (1.40e+02) | **4.34e+04** (1.57e+03) | -2.48e+02 (2.37e+00) |
| | $0.50/k$ | -3.40e+02 (3.04e+00) | **8.10e+03** (1.17e+02) | 2.16e+05 (1.02e+04) | -1.94e+02 (1.90e+00) |
| | $1.00/k$ | 2.81e+02 (1.54e+01) | 8.68e+03 (2.05e+02) | 5.55e+06 (2.53e+05) | -1.71e+02 (1.48e+00) |
| | Fine | **-4.50e+02** (4.63e-03) | 8.74e+03 (1.16e+02) | **1.40e+03** (7.16e+01) | **-3.03e+02** (1.60e+00) |
| | Fine + $0.05/k$ | **-4.50e+02** (2.54e-02) | 8.43e+03 (1.36e+02) | 1.99e+03 (8.59e+01) | -2.77e+02 (3.16e+00) |
| | Fine + $0.10/k$ | -4.48e+02 (8.07e-02) | 8.10e+03 (1.42e+02) | 3.33e+03 (1.59e+02) | -2.39e+02 (3.26e+00) |
| | Fine + $0.20/k$ | -4.37e+02 (5.19e-01) | **7.83e+03** (1.06e+02) | 1.78e+04 (8.37e+02) | -2.07e+02 (1.87e+00) |
| | Fine + $0.50/k$ | -2.11e+02 (6.63e+00) | 8.83e+03 (1.72e+02) | 8.47e+05 (3.91e+04) | -1.82e+02 (1.64e+00) |
| | Fine + $1.00/k$ | 6.67e+02 (2.51e+01) | 1.20e+04 (2.27e+02) | 1.08e+07 (5.26e+05) | -1.68e+02 (1.65e+00) |
| Extended box crossover ($\alpha = 0.5$) | None | **-4.50e+02** (2.72e-04) | **8.78e+03** (1.90e+02) | **7.45e+02** (3.21e+01) | **-2.21e+02** (2.33e+00) |
| | $0.05/k$ | -4.49e+02 (3.17e-02) | 9.63e+03 (2.20e+02) | 3.14e+03 (1.67e+02) | -2.00e+02 (1.97e+00) |
| | $0.10/k$ | -4.40e+02 (2.52e-01) | 1.07e+04 (2.50e+02) | 2.24e+04 (9.14e+02) | -1.95e+02 (1.75e+00) |
| | $0.20/k$ | -3.59e+02 (2.22e+00) | 1.25e+04 (2.45e+02) | 3.35e+05 (1.86e+04) | -1.81e+02 (1.66e+00) |
| | $0.50/k$ | 2.54e+02 (1.91e+01) | 1.60e+04 (3.92e+02) | 6.93e+06 (3.35e+05) | -1.63e+02 (1.68e+00) |
| | $1.00/k$ | 1.60e+03 (5.17e+01) | 2.10e+04 (4.98e+02) | 4.65e+07 (1.85e+06) | -1.50e+02 (2.26e+00) |
| | Fine | **-4.23e+02** (7.81e-01) | **1.76e+04** (3.62e+02) | **2.07e+05** (1.21e+04) | **-1.75e+02** (1.71e+00) |
| | Fine + $0.05/k$ | -3.67e+02 (2.09e+00) | 1.80e+04 (4.84e+02) | 7.16e+05 (2.99e+04) | -1.68e+02 (1.56e+00) |
| | Fine + $0.10/k$ | -2.72e+02 (3.48e+00) | 1.85e+04 (4.39e+02) | 1.55e+06 (6.25e+04) | -1.68e+02 (1.83e+00) |
| | Fine + $0.20/k$ | -2.18e+01 (1.03e+01) | 1.84e+04 (5.26e+02) | 5.45e+06 (2.49e+05) | -1.63e+02 (2.22e+00) |
| | Fine + $0.50/k$ | 9.07e+02 (3.03e+01) | 2.24e+04 (5.74e+02) | 2.69e+07 (1.32e+06) | -1.52e+02 (1.92e+00) |
| | Fine + $1.00/k$ | 2.47e+03 (5.56e+01) | 2.55e+04 (5.66e+02) | 9.84e+07 (3.94e+06) | -1.37e+02 (2.26e+00) |
| Line crossover | None | 4.29e+04 (6.09e+02) | 3.44e+04 (8.72e+02) | 1.22e+10 (4.00e+08) | -7.32e+01 (3.09e+00) |
| | $0.05/k$ | -2.11e+02 (4.61e+00) | 1.45e+04 (3.04e+02) | 1.30e+06 (4.48e+04) | -2.24e+02 (2.19e+00) |
| | $0.10/k$ | -3.55e+02 (2.03e+00) | 1.26e+04 (2.19e+02) | 2.12e+05 (7.95e+03) | **-2.32e+02** (1.56e+00) |
| | $0.20/k$ | **-4.00e+02** (1.76e+00) | 1.00e+04 (1.80e+02) | **9.66e+04** (3.77e+03) | -2.24e+02 (3.13e+00) |
| | $0.50/k$ | -3.37e+02 (3.29e+00) | **7.26e+03** (1.59e+02) | 2.33e+05 (1.30e+04) | -1.91e+02 (1.62e+00) |
| | $1.00/k$ | 2.37e+02 (2.04e+01) | 6.93e+03 (1.46e+02) | 4.34e+06 (2.80e+05) | -1.73e+02 (1.50e+00) |
| | Fine | **-4.48e+02** (1.22e-01) | 9.38e+03 (1.38e+02) | 1.07e+04 (8.87e+02) | **-2.87e+02** (2.31e+00) |
| | Fine + $0.05/k$ | -4.47e+02 (1.52e-01) | 8.81e+03 (1.73e+02) | **8.13e+03** (4.91e+02) | -2.58e+02 (4.68e+00) |
| | Fine + $0.10/k$ | -4.43e+02 (4.00e-01) | 8.16e+03 (1.54e+02) | 1.10e+04 (6.89e+02) | -2.30e+02 (3.34e+00) |
| | Fine + $0.20/k$ | -4.32e+02 (5.20e-01) | 7.79e+03 (1.37e+02) | 2.95e+04 (1.63e+03) | -2.05e+02 (1.97e+00) |
| | Fine + $0.50/k$ | -2.27e+02 (7.78e+00) | **7.71e+03** (1.70e+02) | 7.95e+05 (3.23e+04) | -1.82e+02 (1.50e+00) |
| | Fine + $1.00/k$ | 5.76e+02 (2.37e+01) | 9.91e+03 (1.62e+02) | 9.54e+06 (5.55e+05) | -1.62e+02 (1.68e+00) |
| Extended line crossover ($\alpha = 0.5$) | None | 3.85e+04 (9.05e+02) | 3.11e+04 (6.79e+02) | 9.81e+09 (5.11e+08) | -1.17e+02 (3.40e+00) |
| | $0.05/k$ | -1.37e+02 (1.20e+01) | 1.21e+04 (2.97e+02) | 2.97e+06 (2.14e+05) | **-2.40e+02** (2.59e+00) |
| | $0.10/k$ | -3.27e+02 (4.22e+00) | 9.46e+03 (2.92e+02) | 5.23e+05 (3.33e+04) | -2.37e+02 (2.16e+00) |
| | $0.20/k$ | **-3.67e+02** (3.37e+00) | 8.19e+03 (2.09e+02) | **2.02e+05** (1.00e+04) | -2.13e+02 (2.53e+00) |
| | $0.50/k$ | -2.41e+02 (6.38e+00) | **7.03e+03** (1.43e+02) | 6.22e+05 (3.95e+04) | -1.88e+02 (1.40e+00) |
| | $1.00/k$ | 4.60e+02 (2.41e+01) | 8.00e+03 (1.61e+02) | 7.43e+06 (3.58e+05) | -1.69e+02 (1.61e+00) |
| | Fine | **-4.46e+02** (2.66e-01) | 7.67e+03 (1.98e+02) | **1.88e+04** (2.14e+03) | **-2.44e+02** (5.08e+00) |
| | Fine + $0.05/k$ | -4.39e+02 (4.12e-01) | 7.69e+03 (1.76e+02) | 2.28e+04 (1.56e+03) | -2.26e+02 (3.84e+00) |
| | Fine + $0.10/k$ | -4.29e+02 (1.03e+00) | 7.52e+03 (1.90e+02) | 3.51e+04 (2.50e+03) | -2.09e+02 (1.96e+00) |
| | Fine + $0.20/k$ | -3.92e+02 (2.23e+00) | **7.24e+03** (1.57e+02) | 1.42e+05 (7.84e+03) | -1.92e+02 (1.67e+00) |
| | Fine + $0.50/k$ | -4.34e+01 (1.06e+01) | 8.62e+03 (1.56e+02) | 1.93e+06 (8.59e+04) | -1.73e+02 (1.65e+00) |
| | Fine + $1.00/k$ | 8.69e+02 (3.33e+01) | 1.10e+04 (2.63e+02) | 1.51e+07 (9.43e+05) | -1.57e+02 (1.86e+00) |

Table 2. Results

## 5. Conclusions

In this chapter, we tried to analyze distinct roles of crossover and mutation when using real encoding in genetic algorithms. We investigated the bias of crossover and mutation. From this investigation, we could know that extended crossover and mutation can reduce the inherent bias of traditional crossover in real-coded genetic algorithms.

We also studied the functions of crossover and mutation operators through experiments for various combinations of both operators. From these experiments, we could know that extended-box crossover is good in the case of using only crossover without mutation. However, it is possible to surpass the performance of extended-box crossover using well-designed combination of crossover and mutation. In the case of other crossover operators, not only the function of perturbation but also that of fine tuning by mutation is important, but extended-box crossover contains the fine tuning function in itself.

There are many other test functions defined on real domains. We conducted experiments with limited test functions. We may obtain more reliable conclusions through experiments with more other functions. So, more extended experiments on more various test functions are needed for future work. We may also find other useful properties from those empirical study.

## 6. Acknowledgments

The authors would like to thank Dr. Alberto Moraglio for his encouragement and valuable comments in improving this study. This work was supported by the Research and Development of Advanced Weather Technology of National Institute of Meteorological Research (NIMR) of Korea in 2011.

## 7. References

Akay, B. & Karaboga, D. (2010). A modified artificial bee colony algorithm for real-parameter optimization, *Information Sciences* . doi:10.1016/j.ins.2010.07.015.

Bäck, T., Fogel, D. B. & Michalewicz, T. (eds) (2000). *Evolutionary Computation 1: Basic Algorithms and Operators*, Institute of Physics Publishing.

Ballester, P. J. & Carter, J. N. (2003). Real-parameter genetic algorithms for finding multiple optimal solutions in multi-modal optimization, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 706–717.

Ballester, P. J. & Carter, J. N. (2004a). An effective real-parameter genetic algorithm with parent centric normal crossover for multimodal optimisation, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 901–913.

Ballester, P. J. & Carter, J. N. (2004b). An effective real-parameter genetic algorithms for multimodal optimization, *Proceedings of the Adaptive Computing in Design and Manufacture VI*, pp. 359–364.

Beyer, H.-G. (2001). *Theory of Evolution Strategies*, Springer.

Beyer, H.-G. & Deb, K. (2001). On self-adaptive features in real-parameter evolutionary algorithms, *IEEE Transactions on Evolutionary Computation* 5(3): 250–270.

Chen, Y.-P., Peng, W.-C. & Jian, M.-C. (2007). Particle swarm optimization with recombination and dynamic linkage discovery, *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 37(6): 1460–1470.

Das, S. & Suganthan, P. N. (2011). Differential evolution - a survey of the state-of-the-art, *IEEE Transactions on Evolutionary Computation* 15(1): 4–31.

Dasgupta, S., Das, S., Biswas, A. & Abraham, A. (2009). On stability and convergence of the population-dynamics in differential evolution, *AI Commun.* 22(1): 1–20.

Deb, K. & Agrawal, R. B. (1995). Simulated binary crossover for continuous search space, *Complex Systems* 9(2): 115–148.

Deb, K., Anand, A. & Joshi, D. (2002). A computationally efficient evolutionary algorithm for real-parameter optimization, *Evolutionary Computation* 10(4): 371–395.

Deb, K. & Beyer, H.-G. (1999). Self-adaptation in real-parameter genetic algorithms with simulated binary crossover, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 172–179.

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

Deb, K., Sindhya, K. & Okabe, T. (2007). Self-adaptive simulated binary crossover for real-parameter optimization, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1187–1194.

Eshelman, L. J. (1991). The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination, *Proceedings of the Workshop on Foundations of Genetic Algorithms*, pp. 265–283.

Eshelman, L. J., Mathias, K. E. & Schaffer, J. D. (1997). Crossover operator biases: Exploiting the population distribution, *Proceedings of the International Conference on Genetic Algorithms*, pp. 354–361.

Eshelman, L. J. & Schaffer, J. D. (1993). Real-coded genetic algorithms and interval-schemata, *Proceedings of the Workshop on Foundations of Genetic Algorithms*, pp. 187–202.

Goldberg, D. E. (1991). Real-coded genetic algorithms, virtual alphabets, and blocking, *Complex Systems* 5: 139–167.

Hansen, N. & Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies, *Evolutionary Computation* 9: 159–195.

Herrera, F., Lozano, M. & Sánchez, A. M. (2003). A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study, *International Journal of Intelligent Systems* 18(3): 309–338.

Herrera, F., Lozano, M. & Sánchez, A. M. (2005). Hybrid crossover operators for real-coded genetic algorithms: an experimental study, *Soft Computing* 9(4): 280–298.

Herrera, F., Lozano, M. & Verdegay, J. L. (1998). Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis, *Artificial Intelligence Review* 12(4): 265–319.

Higuchi, T., Tsutsui, S. & Yamamura, M. (2000). Theoretical analysis of simplex crossover for real-coded genetic algorithms, *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature*, pp. 365–374.

Huang, H., Qin, H., Hao, Z. & Lim, A. (2010). Example-based learning particle swarm optimization for continuous optimization, *Information Sciences* . doi:10.1016/j.ins.2010. 10.018.

Igel, C., Hansen, N. & Roth, S. (2007). Covariance matrix adaptation for multi-objective optimization, *Evolutionary Computation* 15(1): 1–28.

Igel, C., Suttorp, T. & Hansen, N. (2006). A computational efficient covariance matrix update and a $(1+1)$-CMA for evolution strategies, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 453–460.

Jägersküpper, J. (2007). Algorithmic analysis of a basic evolutionary algorithm for continuous optimization, *Theoretical Computer Science* 379(3): 329–347.

Janikow, C. Z. & Michalewicz, Z. (1991). An experimental comparison of binary and floating point representations in genetic algorithms, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp. 31–36.

Juang, Y.-T., Tung, S.-L. & Chiu, H.-C. (2011). Adaptive fuzzy particle swarm optimization for global optimization of multimodal functions, *Information Sciences* (20): 4539–4549.

Kang, F., Li, J. & Ma, Z. (2011). Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions, *Information Sciences* (16): 3508–3531.

Kita, H. (2001). A comparison study of self-adaptation in evolution strategies and real-coded genetic algorithms, *Evolutionary Computation* 9(2): 223–241.

Kita, H., Ono, I. & Kobayashi, S. (1998). Theoretical analysis of the unimodal normal distribution crossover for real-coded genetic algorithms, *Proceedings of the International Conference on Evolutionary Computation*, pp. 529–534.

Kita, H., Ono, I. & Kobayashi, S. (1999). Multi-parental extension of the unimodal normal distribution crossover for real-coded genetic algorithms, *Proceedings of the Congress on Evolutionary Computation*, pp. 1581–1587.

Kramer, O. (2008a). Premature convergence in constrained continuous search spaces, *Proceedings of the Parallel Problem Solving from Nature*, pp. 62–71.

Kramer, O. (2008b). *Self-Adaptive Heuristics for Evolutionary Computation*, Springer.

Kramer, O., Gloger, B. & Goebels, A. (2007). An experimental analysis of evolution strategies and particle swarm optimisers using design of experiments, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 674–681.

Krohling, R. A. & Coelho, L. S. (2006). Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems, *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 36(6): 1407–1416.

Kukkonen, S. & Lampinen, J. (2004). An extension of generalized differential evolution for multi-objective optimization with constraints, *Proceedings of the Parallel Problem Solving from Nature*, pp. 752–761.

Kukkonen, S. & Lampinen, J. (2005). GDE3: the third evolution step of generalized differential evolution, *Proceedings of the Congress on Evolutionary Computation*, pp. 443–450.

l. Sun, C., Zeng, J. & Pan, J. (2011). An improved vector particle swarm optimization for constrained optimization problems, *Information Sciences* 181(6): 1153–1163.

Lozano, M., Herrera, F., Krasnogor, N. & Molina, D. (2004). Real-coded memetic algorithms with crossover hill-climbing, *Evolutionary Computation* 12(3): 273–302.

Meyer-Nieberg, S. & Beyer, H.-G. (2007). Self-adaptation in evolutionary algorithms, *Proceedings of the Parameter Setting in Evolutionary Algorithms*, pp. 47–75.

Mezura-Montes, E., Miranda-Varela, M. E. & d. C. Gómez-Ramón, R. (2010). Differential evolution in constrained numerical optimization: An empirical study, *Information Sciences* 180(22): 4223–4262.

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs*, Springer.

Mühlenbein, H. (1994). The breeder genetic algorithm - a provable optimal search algorithm and its application, *IEE Colloquium on Applications of Genetic Algorithms*, pp. 5/1–5/3.

Mühlenbein, H. & Schlierkamp-Voosen, D. (1993). Predictive models for the breeder genetic algorithm I: Continuous parameter optimization, *Evolutionary Computation* 1(1): 25–49.

Noman, N. & Iba, H. (2005). Enhancing differential evolution performance with local search for high dimensional function optimization, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 25–29.

Ono, I., Kita, H. & Kobayashi, S. (1999). A robust real-coded genetic algorithm using unimodal normal distribution crossover augmented by uniform crossover: Effects of self-adaptation of crossover probabilities, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 496–503.

Ono, I. & Kobayashi, S. (1997). A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pp. 246–253.

Qi, A. & Palmieri, F. (1994a). Theoretical analysis of evolutionary algorithms with an infinite population size in continuous space, Part I: Basic properties of selection and mutation, *IEEE Transactions on Neural Networks* 5(1): 102–119.

Qi, A. & Palmieri, F. (1994b). Theoretical analysis of evolutionary algorithms with an infinite population size in continuous space, Part II: Analysis of the diversification role of crossover, *IEEE Transactions on Neural Networks* 5(1): 120–129.

Reed, J., Toombs, R. & Barricelli, N. A. (1967). Simulation of biological evolution and machine learning, *Journal of Theoretical Biology* 17: 319–342.

Ripon, K. S. N., Kwong, S. & Man, K. F. (2007). A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization, *Information Sciences* 177(2): 632–654.

Rönkkönen, J., Kukkonen, S. & Price, K. (2005). Real-parameter optimization with differential evolution, *Proceedings of the Congress on Evolutionary Computation*, pp. 506–513.

Socha, K. & Dorigo, M. (2008). Ant colony optimization for continuous domains, *European Journal of Operational Research* 185(3): 1155–1173.

Someya, H. & Yamamura, M. (2005). A robust real-coded evolutionary algorithm with toroidal search space conversion, *Soft Computing* 9(4): 254–269.

Storn, R. & Price, K. (1997). Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization* 11(4): 341–359.

Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y., Auger, A. & Tiwari, S. (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization, *Technical Report NCL-TR-2005001*, Natural Computing Laboratory (NCLab), Department of Computer Science, National Chiao Tung University.

Surry, P. D. & Radcliffe, N. (1996). Real representations, *Proceedings of the Workshop on Foundations of Genetic Algorithms*, pp. 343–363.

Takahashi, M. & Kita, H. (2001). A crossover operator using independent component analysis for real-coded genetic algorithm, *Proceedings of the Congress on Evolutionary Computation*, pp. 643–649.

Tsutsui, S. (1998). Multi-parent recombination in genetic algorithms with search space boundary extension by mirroring, *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature*, pp. 428–437.

Tsutsui, S. & Goldberg, D. E. (2001). Search space boundary extension method in real-coded genetic algorithms, *Information Sciences* 133(3-4): 229–247.

Tsutsui, S. & Goldberg, D. E. (2002). Simplex crossover and linkage identification: Single-stage evolution vs. multi-stage evolution, *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 974–979.

Tsutsui, S., Goldberg, D. E. & Sastry, K. (2001). Linkage learning in real-coded GAs with simplex crossover, *Proceedings of the Fifth International Conference on Artificial Evolution*, pp. 51–58.

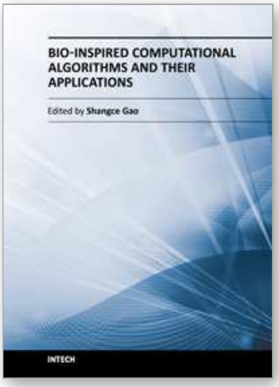
Tsutsui, S., Yamamura, M. & Higuchi, T. (1999). Multi-parent recombination with simplex crossover in real coded genetic algorithms, *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 657–664.

Wei, L., Chen, Z. & Li, J. (2011). Evolution strategies based adaptive $L_p$ LS-SVM, *Information Sciences* 181(14): 3000–3016.

Wright, A. H. (1991). Genetic algorithms for real parameter optimization, *Proceedings of the Workshop on Foundations of Genetic Algorithms*, pp. 205–218.

Yoon, Y., Kim, Y.-H., Moraglio, A. & Moon, B.-R. (2012). A theoretical and empirical study on unbiased boundary-extended crossover for real-valued representation, *Information Sciences* 183(1): 48–65.

Zhang, M., Luo, W. & Wang, X. (2008). Differential evolution with dynamic stochastic selection for constrained optimization, *Information Sciences* 178(15): 3043–3074.



BIO-INSPIRED COMPUTATIONAL ALGORITHMS AND THEIR APPLICATIONS

Edited by Shangce Gao

INTECH

**Bio-Inspired Computational Algorithms and Their Applications**
Edited by Dr. Shangce Gao

ISBN 978-953-51-0214-4
Hard cover, 420 pages
**Publisher** InTech
**Published online** 07, March, 2012
**Published in print edition** March, 2012

Bio-inspired computational algorithms are always hot research topics in artificial intelligence communities. Biology is a bewildering source of inspiration for the design of intelligent artifacts that are capable of efficient and autonomous operation in unknown and changing environments. It is difficult to resist the fascination of creating artifacts that display elements of lifelike intelligence, thus needing techniques for control, optimization, prediction, security, design, and so on. Bio-Inspired Computational Algorithms and Their Applications is a compendium that addresses this need. It integrates contrasting techniques of genetic algorithms, artificial immune systems, particle swarm optimization, and hybrid models to solve many real-world problems. The works presented in this book give insights into the creation of innovative improvements over algorithm performance, potential applications on various practical tasks, and combination of different techniques. The book provides a reference to researchers, practitioners, and students in both artificial intelligence and engineering communities, forming a foundation for the development of the field.

**InTech Europe**
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

**InTech China**
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821