A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization

Kalyanmoy Deb deb@iitk.ac.in
Kanpur Genetic Algorithms Laboratory (KanGAL), Indian Institute of Technology Kanpur, Kanpur, PIN 208 016, India

Ashish Anand ashi@iitk.ac.in
Kanpur Genetic Algorithms Laboratory (KanGAL), Indian Institute of Technology Kanpur, Kanpur, PIN 208 016, India

Dhiraj Joshi djoshi@cse.psu.edu
Department of Computer Science and Engineering, Pennsylvania State University, 2307 Plaza Drive, State College, PA 16801, USA

Abstract

Due to increasing interest in solving real-world optimization problems using evolutionary algorithms (EAs), researchers have recently developed a number of real-parameter genetic algorithms (GAs). In these studies, the main research effort is spent on developing an efficient recombination operator. Such recombination operators use probability distributions around the parent solutions to create an offspring. Some operators emphasize solutions at the center of mass of parents and some around the parents. In this paper, we propose a generic parent-centric recombination operator (PCX) and a steady-state, elite-preserving, scalable, and computationally fast population-alteration model (we call the G3 model). The performance of the G3 model with the PCX operator is investigated on three commonly used test problems and is compared with a number of evolutionary and classical optimization algorithms including other real-parameter GAs with the unimodal normal distribution crossover (UNDX) and the simplex crossover (SPX) operators, the correlated self-adaptive evolution strategy, the covariance matrix adaptation evolution strategy (CMA-ES), the differential evolution technique, and the quasi-Newton method. The proposed approach is found to consistently and reliably perform better than all other methods used in the study. A scale-up study with problem sizes up to 500 variables shows a polynomial computational complexity of the proposed approach. This extensive study clearly demonstrates the power of the proposed technique in tackling real-parameter optimization problems.

Keywords

Real-parameter optimization, simulated binary crossover, self-adaptive evolution strategy, covariance matrix adaptation, differential evolution, quasi-Newton method, parent-centric recombination, scalable evolutionary algorithms.

1 Introduction

Over the past few years, there has been a surge of studies related to real-parameter genetic algorithms (GAs), despite the existence of specific real-parameter evolutionary algorithms (EAs), such as evolution strategy and differential evolution. Although, in principle, such real-parameter GA studies have been shown to have a similar theoretical behavior on certain fitness landscapes with proper parameter tuning in an earlier

study (Beyer and Deb, 2001), in this paper we investigate the performance of a couple of popular real-parameter genetic algorithms and compare extensively with the above-mentioned real-parameter EAs and with a commonly used classical optimization method.

A good description of different real-parameter GA recombination operators can be found in Herrara et al. (1998) and Deb (2001). Of different approaches, the *unimodal normal distribution crossover* (UNDX) operator (Ono and Kobayashi, 1997), the *simplex crossover* (SPX) operator (Higuchi et al., 2000), and the *simulated binary crossover* (SBX) (Deb and Agrawal, 1995) are well studied. The UNDX operator uses multiple parents and creates offspring solutions around the center of mass of these parents. A small probability is assigned to solutions away from the center of mass. On the other hand, the SPX operator assigns a uniform probability distribution for creating offspring in a restricted search space around the region marked by the parents. These mean-centric operators have been applied with a specific GA model (the *minimum generation gap* (MGG) model suggested by Satoh et al. (1996)). The MGG model is a steady-state GA in which 200 offspring solutions are created from a few parent solutions and only two solutions are selected. Using this MGG model, a recent study (Higuchi et al., 2000) compared both UNDX and SPX and observed that the SPX operator performs better than the UNDX operator on a number of test problems. Since the SPX operator uses a uniform probability distribution for creating an offspring, a large offspring pool size (200 members) was necessary to find a useful progeny. On the other hand, the UNDX operator uses a normal probability distribution to create an offspring, giving more emphasis to solutions close to the center of mass of the parents. Therefore, such a large offspring pool size may not be optimal with the UNDX operator. Despite the extensive use of these two recombination operators, we believe that adequate parametric studies were not performed in any earlier study to establish the best parameter settings for these GAs. In this paper, we perform a parametric study by varying the offspring pool size and the overall population size and report interesting outcomes. For a fixed number of function evaluations, the UNDX operator with a biased (normal) probability distribution of creating offspring solutions around the centroid of parents works much better with a small offspring pool size and outperforms the SPX operator, which uses a uniform probability distribution over a simplex surrounding the parents.

Working with nonuniform probability distribution for creating offspring, it is not intuitively clear whether biasing the centroidal region (mean-centric approach as in the UNDX operator) or biasing the parental region (parent-centric approach as in the SBX or fuzzy recombination) is a better approach. A previous study (Deb and Agrawal, 1995) has shown that the binary-coded GAs with the single-point crossover operator, when applied to continuous search spaces, use an inherent probability distribution biasing the parental region, rather than the centroidal region. Using variable-wise recombination operators, a past study (Deb and Beyer, 2000) has clearly shown the advantage of using a parent-centric operator (SBX) over a number of other recombination operators. Motivated by these studies, in this paper, we suggest a generic parent-centric recombination (PCX) operator, which allows a large probability of creating a solution near each parent, rather than near the centroid of the parents. In order to make the MGG model computationally faster, we also suggest a *generalized generation gap* (G3) model, which replaces the roulette-wheel selection operator of the MGG model with a block selection operator. The proposed G3 model is a steady-state, elite-preserving, and computationally fast algorithm for real-parameter optimization. The efficacy of the G3 model with the proposed PCX operator is investigated by comparing it with UNDX

and SPX operators on three test problems.

To further investigate the performance of the proposed G3 model with the PCX operator, we also compare it to the correlated self-adaptive evolution strategy and the differential evolution method. To really put the proposed GA to the test, we also compare it to a commonly used classical optimization procedure – the quasi-Newton method with BFGS update procedure (Reklaitis et al., 1983). Finally, the computational complexity of the proposed GA is investigated by performing a scale-up study on three chosen test problems having as many as 500 variables.

Simulation studies show remarkable performance of the proposed GA with the PCX operator. Since the chosen test problems have been well studied, we also compare the results of this paper with past studies where significant results on these test problems have been reported. The extensive comparison of the proposed approach with a number of challenging competitors chosen from evolutionary and classical optimization literature clearly demonstrates the superiority of the proposed approach. In addition, the polynomial computational complexity observed with the proposed GA should encourage the researchers and practitioners to test and apply it to more complex and challenging real-world search and optimization problems.

2 Evolutionary Algorithms for Real-Parameter Optimization

Over the past few years, many researchers have been paying attention to real-coded evolutionary algorithms, particularly for solving real-world optimization problems. In this context, three different approaches have been popularly practiced: (i) self-adaptive evolution strategies (Bäck, 1997; Hansen and Ostermeier, 1996; Rechenberg, 1973; Schwefel, 1987), (ii) differential evolution (Storn and Price, 1997), and (iii) real-parameter genetic algorithms (Deb, 2001; Herrera et al., 1998). However, some recent studies have shown the similarities in search principles between some of these different approaches (Beyer and Deb, 2001; Kita et al., 1999) on certain fitness landscapes. Details of all these different evolutionary techniques can be found in respective studies. Here, we discuss some fundamental approaches to real-parameter GAs, as our proposed optimization algorithm falls in this category.

Among numerous studies on the development of different recombination operators for real-parameter GAs, *blend crossover* (BLX), SBX, UNDX, and SPX are commonly used. A number of other recombination operators, such as arithmetic crossover, intermediate crossover, and extended crossover are similar to the BLX operator. A detailed study of many such operators can be found elsewhere (Deb, 2001; Herrara et al., 1998). In the recent past, GAs with some of these recombination operators have been demonstrated to exhibit self-adaptive behavior similar to that observed in evolution strategy and evolutionary programming approaches.

Beyer and Deb (2001) argued that a variation operator (a combination of the recombination and the mutation operator) should have the following two properties:

1. Population mean decision variable vector should remain the same before and after the variation operator.

2. Variance of the intramember distances must increase due to the application of the variation operator.

Since variation operators usually do not use any fitness function information explicitly, the first argument makes sense. The second argument comes from the realization that the selection operator has a tendency to reduce the population variance. Thus,

population variance must be increased by the variation operator to maintain adequate diversity in the population. In the context of real-parameter optimization, a recombination operator alone can introduce arbitrary diversity in the offspring population. Since in this study we have not used any mutation operator, a real-parameter recombination operator should satisfy the above two properties.

It is interesting that the population mean can be preserved in several ways. One method would be to have individual recombination events producing offspring near the centroid of the participating parents. We call this approach *mean-centric* recombination. The other approach would be to have individual recombination events biasing offspring to be created near the parents, but assigning each parent an equal probability of creating offspring in its neighborhood. This will also allow that the expected population mean of the entire offspring population is identical to that of the parent population. We call this latter approach *parent-centric* recombination.

Recombination operators such as UNDX and BLX are mean-centric approaches, whereas the SBX and fuzzy recombination (Voigt et al., 1995) are parent-centric approaches. Beyer and Deb (2001) have also shown that these operators may exhibit similar performance if the variance growth under recombination operation can be matched by fixing their associated parameters. In this paper, we treat the UNDX operator as a representative mean-centric recombination operator and a multiparent version of the SBX operator as a parent-centric recombination operator.

2.1 Mean-Centric Recombination

In the UNDX operator, $(\mu - 1)$ parents are randomly chosen and their mean $\vec{g}$ is computed. From this mean, $(\mu - 1)$ direction vectors ($\vec{d}^{(i)} = \vec{x}^{(i)} - \vec{g}$) are formed. Let the direction cosines be $\vec{e}^{(i)} = \vec{d}^{(i)}/|\vec{d}^{(i)}|$. Thereafter, from another randomly chosen parent $\vec{x}^{(\mu)}$, the length D of the vector $(\vec{x}^{(\mu)} - \vec{g})$ orthogonal to all $\vec{e}^{(i)}$ is computed. Let $\vec{e}^{(j)}$ (for $j = \mu, \ldots, n$, where n is the size of the variable vector $\vec{x}$) be the orthonormal basis of the subspace orthogonal to the subspace spanned by all $\vec{e}^{(i)}$ for $i = 1, \ldots, (\mu - 1)$. Then, the offspring is created as follows:

$$\vec{y} = \vec{g} + \sum_{i=1}^{\mu-1} w_i |\vec{d}^{(i)}| \vec{e}^{(i)} + \sum_{i=\mu}^{n} v_i D \vec{e}^{(i)}, \tag{1}$$

where w_i and v_i are zero-mean normally distributed variables with variances σ_ζ^2 and σ_η^2, respectively. Kita and Yamamura (1999) suggested $\sigma_\zeta = 1/\sqrt{\mu - 2}$ and $\sigma_\eta = 0.35/\sqrt{n - \mu - 2}$, respectively and observed that $\mu = 3$ to 7 performed well. It is interesting to note that each offspring is created around the mean vector $\vec{g}$. The probability of creating an offspring diminishes with the distance from the mean vector, and the maximum probability is assigned at the mean vector. Figure 1 shows three parents and a few offspring created by the UNDX operator. The complexity of the above procedure in creating one offspring is $O(\mu^2)$, governed by the Gram-Schmidt orthonormalization needed in the process.

The SPX operator also creates offspring uniformly around the mean but restricts them within a predefined region (in a simplex similar but $\gamma = \sqrt{\mu + 1}$ times bigger than the parent simplex). A distinguishing aspect of the SPX from the UNDX operator is that the SPX assigns a uniform probability distribution for creating any solution in a restricted region (called the simplex). Although, in its true sense, it is not a mean-centric operator, because of its emphasis to solutions around the centroid of the participating parents we put this operator in the category of the mean-centric operators. Figure 2

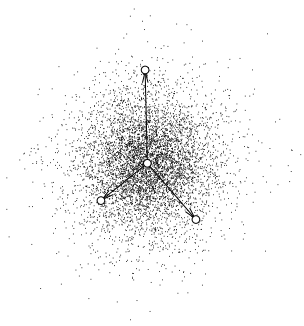

Figure 1: UNDX.

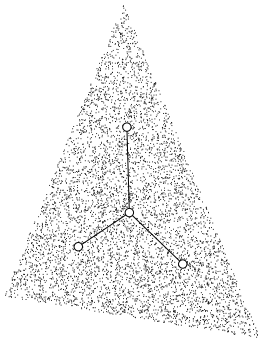

Figure 2: SPX.

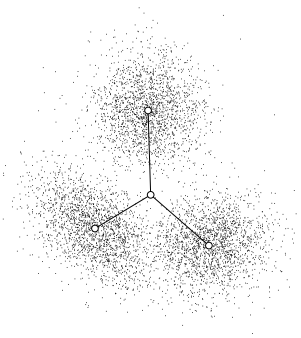

Figure 3: PCX.

shows the density of solutions with three parents for the SPX operator. The computational complexity for creating one offspring here is $O(\mu)$, thereby making the SPX operator faster than the UNDX operator.

2.2 Parent-Centric Recombination (PCX)

The SBX operator assigns a higher probability for an offspring to remain closer to the parents than away from parents. We use this parent-centric concept and modify the UNDX operator as follows. The mean vector $\vec{g}$ of the chosen μ parents is computed. For each offspring, one parent $\vec{x}^{(p)}$ is chosen with equal probability. The direction vector $\vec{d}^{(p)} = \vec{x}^{(p)} - \vec{g}$ is calculated. Thereafter, from each of the other $(\mu - 1)$ parents, perpendicular distances D_i to the line $\vec{d}^{(p)}$ are computed and their average $\bar{D}$ is found. The offspring is created as follows:

$$\vec{y} = \vec{x}^{(p)} + w_\zeta \vec{d}^{(p)} + \sum_{i=1,\ i\neq p}^{\mu} w_\eta \bar{D} \vec{e}^{(i)}, \tag{2}$$

where $\vec{e}^{(i)}$ are the $(\mu - 1)$ orthonormal bases that span the subspace perpendicular to $\vec{d}^{(p)}$. Thus, the complexity of the PCX operator to create one offspring is $O(\mu)$, instead of $O(\mu^2)$ required for the UNDX operator. The parameters w_ζ and w_η are zero-mean normally distributed variables with variance σ_ζ^2 and σ_η^2, respectively. The important distinction from the UNDX operator is that offspring solutions are centered around each parent. The probability of creating an offspring closer to the parent is higher. Figure 3 shows the distribution of offspring solutions with three parents. The motivation of the PCX operator is as follows. Since individual parents have qualified the "fitness test" in the selection operator, it can be assumed that solutions close to these parents are also potential good candidates, particularly in the context of continuous search space problems. On the contrary, it may be quite demanding to assume that the solutions close to the centroid of the participating parents are also good, especially in cases where parents are well sparsed in the search space. Creating solutions close to previously found good solutions, as emphasized by the PCX operator, should make a more reliable search. It is also intuitive that the convergence towards a local optimum can be made faster by always choosing $\vec{x}^{(p)}$ as the best parent.

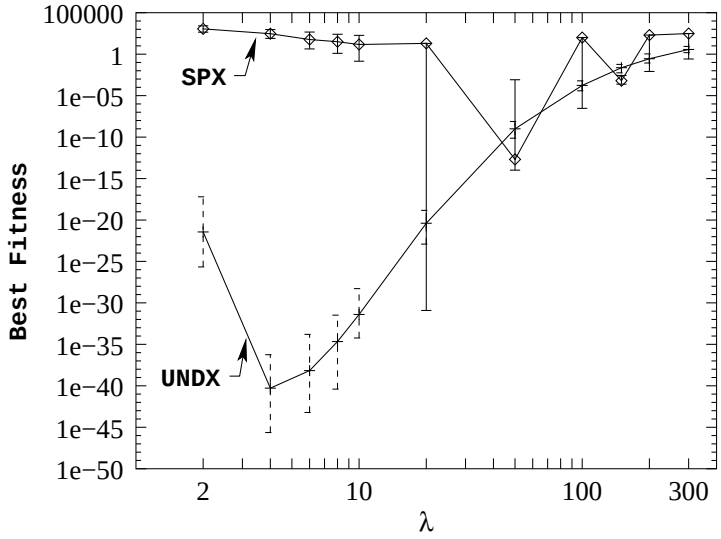

Figure 4: Best fitness for different λ on $F_{\rm elp}$ using the MGG model with SPX and UNDX operators.

3 Evolutionary Algorithm Models

Besides the recombination operator, researchers have also realized the importance of a population alteration model different from a standard genetic algorithm for real-parameter optimization. In the following, we describe a commonly used model originally suggested by Satoh et al. (1996) and later used in a number of studies (Kita et al., 1999; Tsutsui et al., 1999).

3.1 Minimal Generation Gap (MGG) Model

This is a steady-state model, where recombination and selection operators are intertwined in the following manner:

1. From the population P, select μ parents randomly.
2. Generate λ offspring from μ parents using a recombination scheme.
3. Choose two parents at random from the population P.
4. Of these two parents, one is replaced with the best of λ offspring and the other is replaced with a solution chosen by a roulette-wheel selection procedure from a combined population of λ offspring and two chosen parents.

The above procedure completes one iteration of the MGG model. A recent study (Higuchi et al., 2000) used $\mu = n + 1$ and $\lambda = 200$ for the SPX operator and $\mu = 3$ and $\lambda = 200$ for the UNDX operator. No mutation operator was used. With the above parameters, that study showed that the MGG model with the SPX operator and a population size of 300 was able to solve a number of test problems better than that using the UNDX operator. However, that study did not show any justification for using $\lambda = 200$ and for using a population size of $N = 300$. Here, we use the MGG model with both

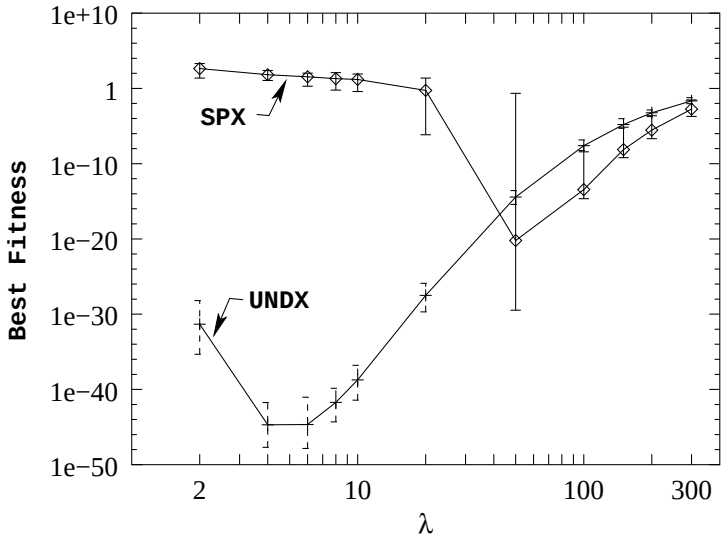

Figure 5: Best fitness on $F_{\rm sch}$ using the MGG model with SPX and UNDX.

recombination operators and perform a parametric study with λ on three standard test problems:

$$F_{\rm elp} = \sum_{i=1}^{n} i x_i^2 \quad \text{(ellipsoidal function)} \tag{3}$$

$$F_{\rm sch} = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \quad \text{(Schwefel's function)} \tag{4}$$

$$F_{\rm ros} = \sum_{i=1}^{n-1} \left(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2 \right) \quad \text{(generalized Rosenbrock's function)} \tag{5}$$

In all problems, we have used $n = 20$. The first two problems have their minimum at $x_i^* = 0$ with $F^* = 0$, and the third function has its minimum at $x_i^* = 1$ with $F^* = 0$. In order not to bias the search, we have initialized the population in $x_i \in [-10, -5]$ for all i in all problems.

First, we fix $N = 300$ and vary λ from 2 to 300. All other parameters are kept as they were used in the original study (Higuchi et al., 2000), except that in UNDX, $\mu = 6$ is used, as this value was found to produce better results. In all experiments, we ran the MGG model until a predefined number of function evaluations F^T elapsed. We used the following values of F^T for different functions: $F^T_{\rm elp} = 0.5(10^6)$, $F^T_{\rm sch} = 1(10^6)$, and $F^T_{\rm ros} = 1(10^6)$. In all experiments, 50 runs with different initial populations were taken and the smallest, median, and highest best fitness values recorded. Figure 4 shows the best fitness values obtained by the SPX and the UNDX operators on $F_{\rm elp}$ with different values of λ. The figure shows that $\lambda = 50$ produced the best reliable performance for the SPX operator. Importantly, the MGG model with $\lambda = 200$ (which was suggested in the original study) did not perform as well. Similarly, for the UNDX operator, the best

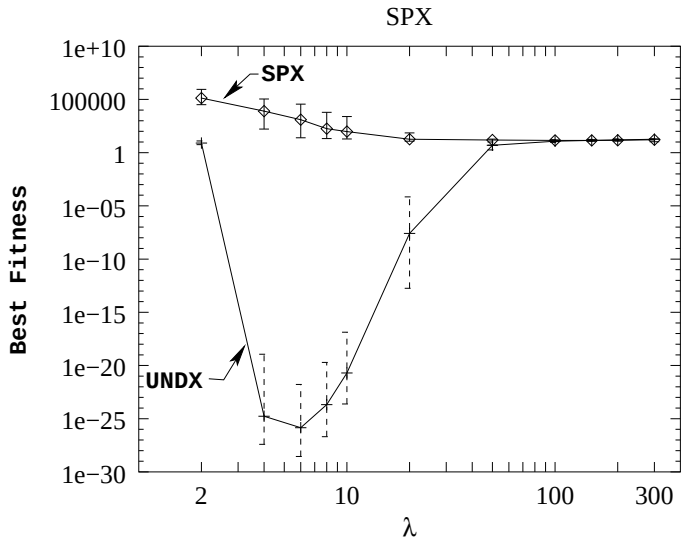

Figure 6: Best fitness on $F_{\rm ros}$ using the MGG model with SPX and UNDX.

performance is observed at $\lambda = 4$, which is much smaller than the suggested value of 200.

Figure 5 shows that in $F_{\rm sch}$, best performances are observed with identical values for λ with SPX and UNDX. Figure 6 shows the population best fitness for the MGG model with SPX and UNDX operators applied to the $F_{\rm ros}$ function. Here, the best performance is observed at $\lambda = 100$ to 300 for the SPX operator and $\lambda = 6$ for the UNDX operator.

Thus, it is clear from the above experiments that the suggested value of $\lambda = 200$ (which was recommended and used in earlier studies (Satoh et al., 1996; Kita et al., 1999)) is not optimal for either recombination operator (UNDX or SPX). Instead, a smaller value of λ has exhibited better performance. It is also clear from the figures that the SPX operator works better with a large offspring pool size, whereas the UNDX works well with a small offspring pool size. This fact can be explained as follows. Since a uniform probability distribution is used in the SPX operator, a large pool size requirement is intuitive. With a biased probability distribution, the UNDX operator does not rely on the sample size, rather it relies on a large number of iterations, each providing a careful choice of an offspring close the center of mass of the chosen parents.

3.2 Generalized Generation Gap (G3) Model

Here, we modify the MGG model to make it computationally faster by replacing the roulette-wheel selection with a block selection of the best two solutions. This model also preserves elite solutions from the previous iteration.

1. From the population P, select the best parent and $\mu - 1$ other parents randomly.
2. Generate λ offspring from the chosen μ parents using a recombination scheme.
3. Choose two parents at random from the population P.

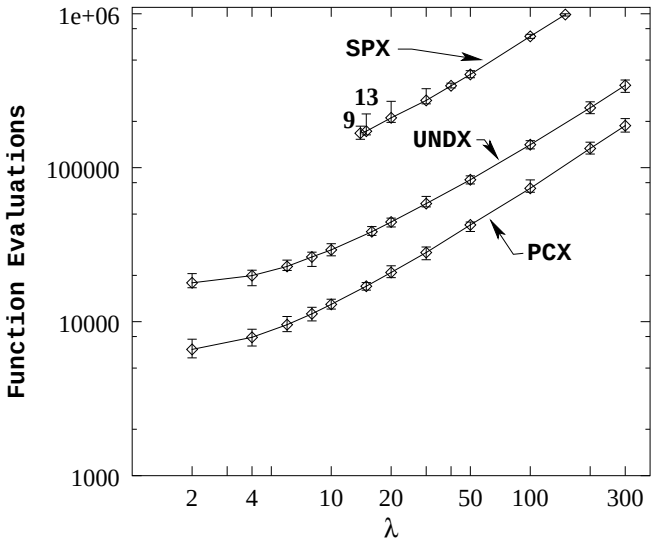

Figure 7: Function evaluations needed to find a solution of fitness 10^{-20} on F_{elp} using the G3 model with PCX, UNDX, and SPX. For PCX and UNDX operators, a population size of 100 is used, and for the SPX operator, a population size of 300 is used.

4. From a combined subpopulation of two chosen parents and λ created offspring, choose the best two solutions and replace the chosen two parents (in Step 3) with these solutions.

The above G3 model can also be modified by choosing only one parent in Step 3 (instead of two parents) and replacing the parent with the best of the combined subpopulation of λ offspring and the parent. At first, we do not make this change and continue with the above G3 model in order to keep the structure of the model similar to the MGG model. Later, we shall investigate the efficacy of this modified G3 model.

3.2.1 Simulation Results

In all experiments with the G3 model, we record the number of function evaluations needed to achieve the best fitness value equal to 10^{-20}. Figure 7 shows the performance of the G3 model with all three operators (PCX, UNDX, and SPX) on the ellipsoidal problem. For the PCX and UNDX operators, $N = 100$ is used, and for the SPX operator, $N = 300$ is used. A large population for SPX is found to be essential for better performance. In all PCX runs, we have used $\sigma_\eta = \sigma_\zeta = 0.1$. For a faster convergence, the best parent is always used to calculate the direction vector $d^{(p)}$ in Equation 2. In PCX and UNDX runs, we have used $\mu = 3$, and in SPX runs, we have used $\mu = n + 1$ or 21 (as suggested by the developers of SPX).

The minimum, median, and maximum number of required function evaluations, as shown in the figure, suggest the robustness of the G3 model with the PCX operator. The G3 model with the PCX operator has performed better (minimum required function evaluations is only 5,818) than that with the UNDX operator (minimum required function evaluations is 16,602). For the SPX operator, not all 50 runs have found a solution having a fitness value as small as the required value of 10^{-20} for $\lambda = 12$ and 15.

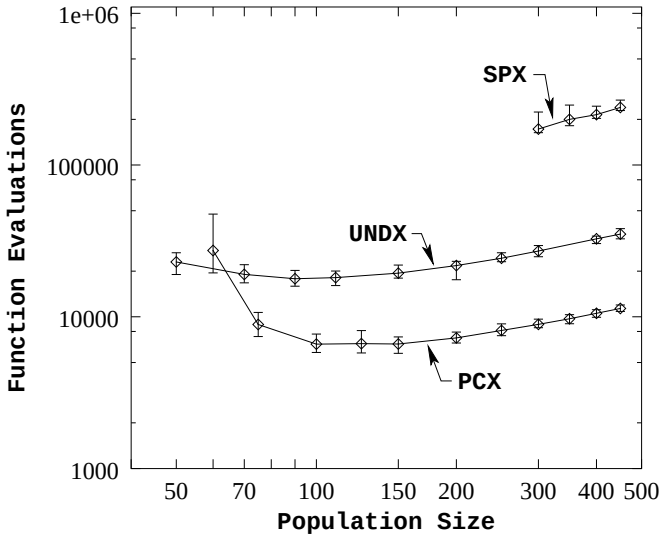

Figure 8: Function evaluations to reach a fitness 10^{-20} versus population sizes on $F_{\rm elp}$ using the G3 model with PCX, UNDX, and SPX.

The number of runs (out of 50) where such a solution was found are marked on the plot. For $\lambda < 12$, the SPX operator did not find the target solution in *any* of the 50 runs. The best run of the SPX operator (with $\lambda = 12$) required 163,380 function evaluations.

The figure shows that with smaller offspring pool size (λ), better performance with PCX and UNDX operators is achieved. Thus, we choose $\lambda = 2$ for these operators and perform a parametric study with the population size. For the SPX operator, we use $\lambda = 15$, below which satisfactory results were not found. Figure 8 shows that there exists an optimum population size at which the performance is the best for PCX (with 5,744 function evaluations) and UNDX (with 15,914 function evaluations). For the 20-variable ellipsoidal problem, $N \sim 100$ seems to be optimum for these two operators. It is also interesting to note that the optimal population size requirement for the PCX operator is larger than that for the UNDX operator. However, the solution accuracy achieved by the PCX operator is almost an order of magnitude better than that obtained by the UNDX operator. Another interesting aspect is that for the SPX operator with $\lambda = 15$, all runs with population size smaller than 300 did not find the desired solution. Since SPX creates solutions within a fixed range proportional to the location of the parents, its search power is limited. Moreover, since random samples are taken from a wide region in the search space, the success of the algorithm relies on the presence of a large population size.

Next, we apply the G3 model with all three recombination operators to $F_{\rm sch}$. Figures 9 and 10 show the parametric studies with λ and the population size for the PCX and the UNDX operators, respectively. Once again, $N = 100$ and $\lambda = 2$ are found to perform the best for both operators. However, the PCX operator is able to find the desired solution with a smaller number of function evaluations (14,643 for PCX versus 27,556 for UNDX). However, the SPX operator does not perform well at all on the Schwefel's function. The minimum function evaluations needed with any parameter

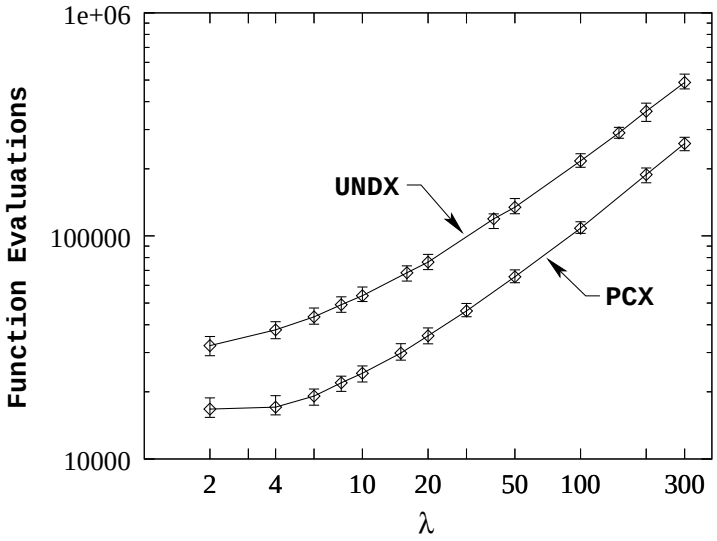

Figure 9: Function evaluations needed to find a solution of fitness 10^{-20} on F_{sch} using the G3 model with PCX and UNDX. $N = 100$ is used.

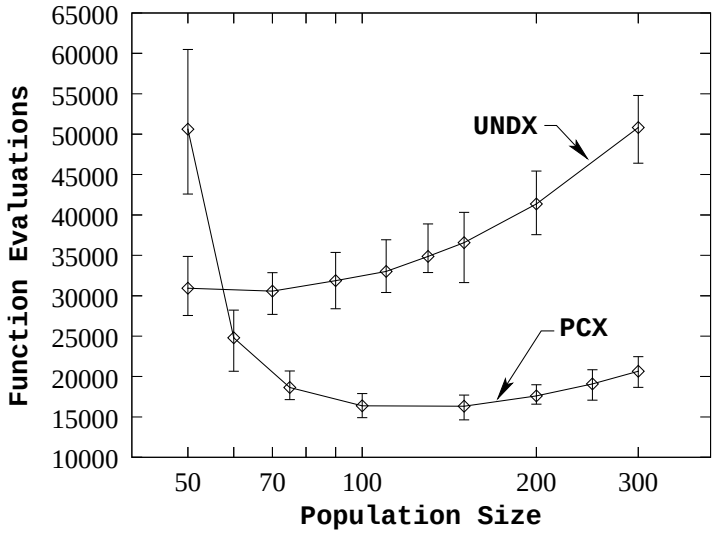

Figure 10: Function evaluations to reach a fitness 10^{-20} versus population sizes on F_{sch} using the G3 model with PCX and UNDX. $\lambda = 2$ is used.

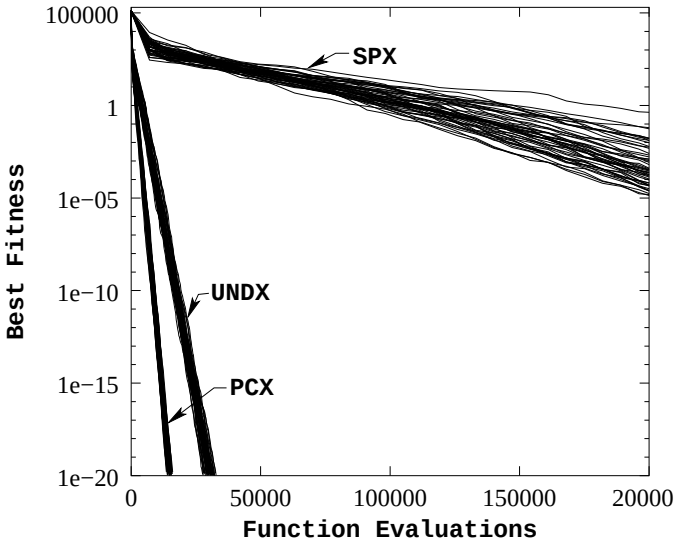

Figure 11: Best fitness versus function evaluations for $F_{\rm sch}$.

Table 1: Three minima (or near minima) for the 20-variable Rosenbrock's function.

$f(\vec{x})$	$x_i, i = 1, 2, \ldots, 20$						
0	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	
3.986624	−0.993286	0.996651	0.998330	0.999168	0.999585	0.999793	0.999897
	0.999949	0.999974	0.999987	0.999994	0.999997	0.999998	0.999999
	0.999999	0.999999	0.999999	0.999997	0.999995	0.999989	
65.025362	−0.010941	0.462100	0.707587	0.847722	0.922426	0.960915	0.980415
	0.990213	0.995115	0.997563	0.998782	0.999387	0.999682	0.999818
	0.999861	0.999834	0.999722	0.999471	0.998953	0.997907	

setting of the SPX operator to find the desired solution is 414,350, which is an order of magnitude more than the best results obtained using the PCX and the UNDX operators. Thus, we have not presented any simulation result for the SPX operator.

Figure 11 shows the population best fitness values (of 50 runs) of $F_{\rm sch}$ with number of function evaluations in the case of the G3 model with the best-performing parameters for PCX ($\lambda = 2$ and $N = 150$), UNDX ($\lambda = 2$ and $N = 70$), and SPX ($\lambda = 70$ and $N = 300$) operators. The figure shows the superiority of the PCX operator in achieving a desired accuracy with the smallest number of function evaluations.

Next, we attempt to solve the $F_{\rm ros}$ function. This function is more difficult to solve than the previous two functions. Here, no implementation is able to find a solution very close to the global optimum (with a fitness value 10^{-20}) in *all* 50 runs each within one million function evaluations. Runs with PCX and UNDX operators sometimes get stuck to a local optimum solution. Interestingly, this function is claimed to be unimodal in a number past of studies. However, for $n > 3$, this function has more than one min-

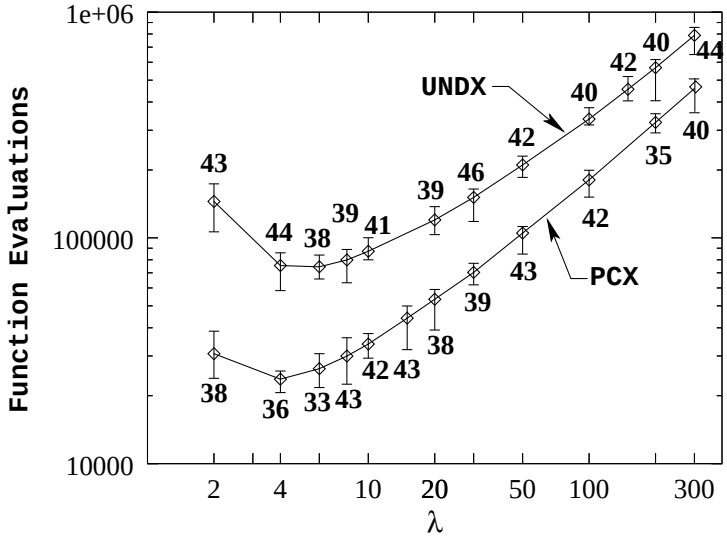

Figure 12: Function evaluations needed to find a solution of fitness 10^{-20} for different λ values on the $F_{\rm ros}$ using the G3 model with PCX and UNDX operators.

imum, of which the solution $x_i = 1$ (for all i) is the global minimum. For 20 variables, we have identified three minima with function values 0, 3.98662, and 65.025362, respectively. The corresponding variable values are tabulated in Table 1. In the figures (to be described next) involving $F_{\rm ros}$, whenever a GA did not find the global minimum, it was always attracted to the best local optimum (with the function value 3.98662). It is also important to highlight that the SPX operator failed to find the required solution in *any* of the 50 runs. However, Figures 12 and 13 show that the PCX operator (with a minimum of 14,847 function evaluations) has performed much better than the UNDX operator (with a minimum of 58,205 function evaluations). The number of times where a run has converged near the global optimum is also marked in the figures. Once again, a small λ (~ 4) and an adequate population size (100 to 150) are found to produce optimum behavior for PCX and UNDX operators in this problem.

To compare the performance of the UNDX operator when applied with the MGG and the G3 model, we compare the number of function evaluations needed to achieve a solution with fitness 10^{-20} on all three test problems. In both cases, results are obtained with their best parameter settings. The following table shows that the G3 model is an order of magnitude better than the MGG model.

Function	MGG	G3
$F_{\rm elp}$	2,97,546	18,120
$F_{\rm sch}$	5,03,838	30,568
$F_{\rm ros}$	9,38,544	73,380

The careful creation of new solutions near successful parents and the dependence of the actual distance of new solutions from parents on the diversity of the parent population allow the PCX operator to have a self-adaptive yet efficient search. The combination of such a search operator along with a fast population alteration procedure makes the overall approach much better than the previous real-parameter EAs.

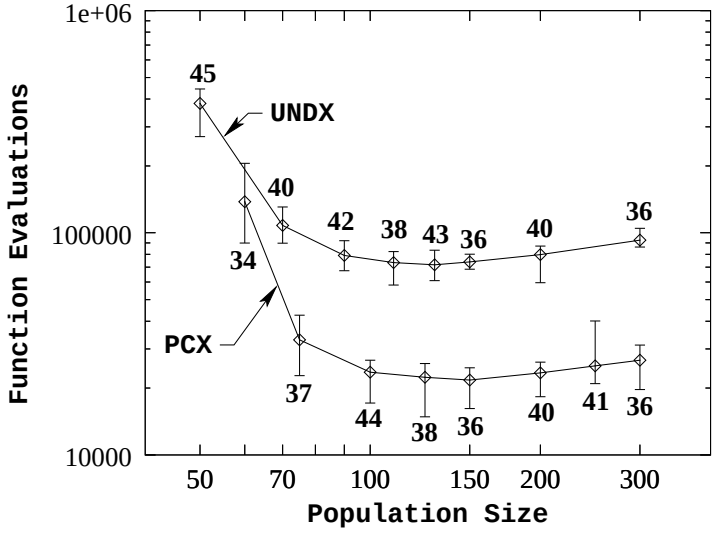

Figure 13: Function evaluations to reach a fitness 10^{-20} versus population sizes on $F_{\rm ros}$ using the G3 model with PCX and UNDX operators. For PCX and UNDX operators, a population size of 100 is used, and for the SPX operator, a population size of 300 is used.

Table 2: Comparison of the original and the modified G3 model on three test problems.

G3	$F_{\rm elp}$			$F_{\rm sch}$		
model	Best	Median	Worst	Best	Median	Worst
Original	5,744	6,624	7,372	14,643	16,326	17,712
Modified G3	5,826	6,800	7,728	13,988	15,602	17,188
G3	$F_{\rm ros}$					
model	Best	Median	Worst			
Original	14,847	22,368	25,797			
Modified G3	16,508	21,452	25,520			

3.2.2 Modified G3 Model

In this subsection, we show simulation results on all three test problems using the modified G3 model in which, instead of two parents, only one parent is chosen in Step 3 and updated in Step 4. Table 2 presents the best, median, and worst function evaluations of 50 runs for the original G3 and the modified G3 models to obtain a function value equal to 10^{-20} for all three problems. For each problem, 20 variables are used. The table shows that in the ellipsoidal function, the original G3 model performed better, and in the Schwefel's function, the modified G3 performed better. On Rosenbrock's function, the original G3 model found a better overall best solution over 50 runs, but the modified G3 found a better median solution. Based on these results, we cannot conclude which of the two G3 models is better. However, in the rest of the paper, we compare simulation results of other optimization algorithms with the modified G3 model.

Table 3: Comparison of correlated ES and CMA-ES with the proposed approach.

	F_{elp}			F_{sch}		
EA	Best	Median	Worst	Best	Median	Worst
$(1, 10)$-ES	28,030	40,850	87,070	72,330	105,630	212,870
$(15, 100)$-ES	83,200	108,400	135,900	173,100	217,200	269,500
CMA-ES	8,064	8,472	8,868	15,096	15,672	16,464
Modified G3	5,826	6,800	7,728	13,988	15,602	17,188
	F_{ros}					
EA	Best	Median	Worst			
$(1, 10)$-ES	591,400	803,800	997,500			
$(15, 100)$-ES	663,760	837,840	936,930			
CMA-ES	29,208	33,048	41,076			
Modified G3	16,508	21,452	25,520			

4 Self-Adaptive Evolution Strategies

Besides the real-parameter genetic algorithms, self-adaptive evolution strategies have also been applied to solve real-parameter optimization problems. In this section, we apply these two methods to the above three test problems and compare the results with that obtained using the proposed GA model. In all cases, we have initialized each variable x_i in the range $[-10, -5]$, so that the initial population does not bracket the global optimum solution.

There exist a number of different self-adaptive evolution strategies (ESs) (Beyer, 2001; Schwefel, 1987; Hansen and Ostermeier, 1996) for solving problems with strong linkages among parameters. In this study, we first use the correlated self-adaptive ES, in which the extra strategy parameters, such as the mutation strength of each variable and correlations between pair-wise variable interactions, are also updated along with the problem variables. Using the standard guidelines of choosing the learning rate (Beyer, 2001) for self-adaptive ES, we use $(1, 10)$-ES and $(15, 100)$-ES on all three test problems having 20 variables. Table 3 shows the best, median, and worst number of function evaluations (of 50 independent runs) needed to achieve a solution with a function value of 10^{-20} using the above two correlated ESs and the modified G3 model with the PCX operator on the three test problems. It is clear from the table that both correlated ESs require at least an order of magnitude more function evaluations than the proposed approach to achieve the same accuracy in the obtained solutions for all three test problems.

Next, we try the *covariance matrix adaptation evolution strategy* (CMA-ES) developed by Hansen and Ostermeier (1996). Collecting the information of previous mutations, the CMA-ES determines the new mutation distribution providing a larger probability for creating better solutions. The procedure is quite involved mathematically, and interested readers are encouraged to refer to the original study or Hansen and Ostermeier (2000). In this study, we use the MATLAB coded from the developer's website along with the recommended parameter settings. For the 20-variable test problems, we have used $(3_I, 12)$-CMA-ES with solutions initialized in $x_i \in [-10, -5]$. Table 3 shows the number of function evaluations needed to find a solution (in 50 runs) with a function value of 10^{-20} in all three test problems. Although the required function evaluations

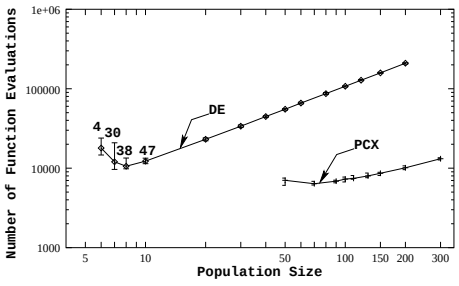

Figure 14: Function evaluations needed with differential evolution for the ellipsoidal problem.

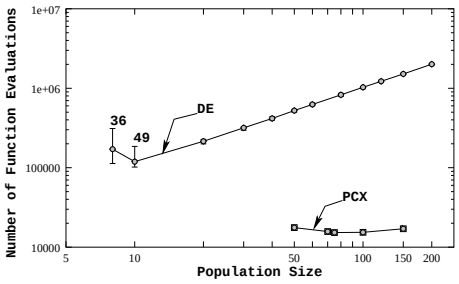

Figure 15: Function evaluations needed with differential evolution for the Schwefel's function.

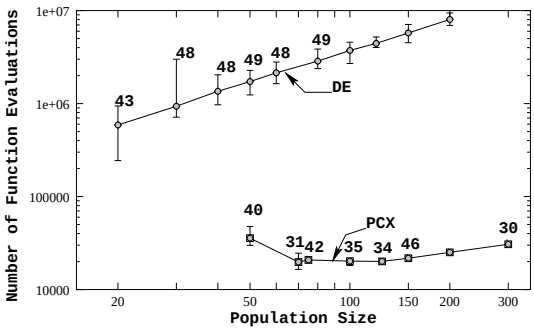

Figure 16: Function evaluations needed with differential evolution for the Rosenbrock's function.

are much better than those needed for the correlated self-adaptive ES, they are somewhat worse than those needed for the modified G3 model with the PCX operator. It is worth mentioning that the CMA-ES requires setting of many parameters, some reasonable values of which are suggested in the original study. However, in the case of the G3 model, there are two parameters σ_η and σ_ζ (in Equation 2) that a user has to set. In all studies here, we have used a constant value of 0.1 for these two parameters.

5 Differential Evolution

Differential evolution (DE) (Storn and Price, 1997) is a steady-state EA in which for every offspring a set of three parent solutions and an index parent are chosen. Thereafter, a new solution is created either by a variable-wise linear combination of three parent solutions or by simply choosing the variable from the index parent with a probability. The resulting new solution is compared with the index parent and the better of them is declared as the offspring. We have used the C-code downloaded from the DE website `http://http.icsi.berkeley.edu/~storn/` and used strategy 1 with all parameters set as suggested in the code. Moreover, this strategy is also observed to perform better than other DE strategies in our limited experimental study.

Figures 14, 15, and 16 compare the function evaluations needed with DE to achieve a solution with a function value of 10^{-20} on three test problems with the modified G3 model and the PCX operator. We have applied DE with different population sizes each performed with 50 independent runs. It is interesting to note that in all three problems, an optimal population size is observed. In the case of the Rosenbrock's function, only a few runs out of 50 runs find the true optimum with a population size smaller than 20. All three figures demonstrate that the best performance of the modified G3 with the PCX operator is better than the best performance of the DE. To highlight the best performances of both methods, we have tabulated corresponding function evaluations in the following table:

	$F_{\rm elp}$			$F_{\rm sch}$		
Method	Best	Median	Worst	Best	Median	Worst
DE	9,660	12,033	20,881	102,000	119,170	185,590
Modified G3	5,826	6,800	7,728	13,988	15,602	17,188

	$F_{\rm ros}$		
Method	Best	Median	Worst
DE	243,800	587,920	942,040
Modified G3	16,508	21,452	25,520

The table shows that except for the ellipsoidal function, the modified G3 requires an order of magnitude less number of function evaluations than DE.

6 Quasi-Newton Method

The quasi-Newton method for unconstrained optimization is a popular and efficient approach (Deb, 1995; Reklaitis et al., 1983). Here, we have used the BFGS quasi-Newton method along with a mixed quadratic-cubic polynomial line search approach available in MATLAB (Branch and Grace, 1996). The code computes gradients numerically and adjusts step sizes in each iteration adaptively for a fast convergence to the minimum. This method is found to produce the best performance among all optimization procedures coded in MATLAB, including the steepest-descent approach.

In Table 4, we present the best, median, and worst function values obtained from a set of 10 independent runs started from random solutions with $x_i \in [-10, -5]$. The maximum number of function evaluations allowed in each test problem is determined

Table 4: Solution accuracy obtained using the quasi-Newton method. FE denotes the maximum allowed function evaluations. In each case, the G3 model with the PCX operator finds a solution with a function value smaller than 10^{-20}.

Func.	FE	Best	Median	Worst
$F_{\rm elp}$	6,000	$8.819(10^{-24})$	$9.718(10^{-24})$	$2.226(10^{-23})$
$F_{\rm sch}$	15,000	$4.118(10^{-12})$	$1.021(10^{-10})$	$7.422(10^{-9})$
$F_{\rm ros}$	15,000	$6.077(10^{-17})$	$4.046(10^{-10})$	3.987
$F_{\rm elp}$	8,000	$5.994(10^{-24})$	$1.038(10^{-23})$	$2.226(10^{-23})$
$F_{\rm sch}$	18,000	$4.118(10^{-12})$	$4.132(10^{-11})$	$7.422(10^{-9})$
$F_{\rm ros}$	26,000	$6.077(10^{-17})$	$4.046(10^{-10})$	3.987

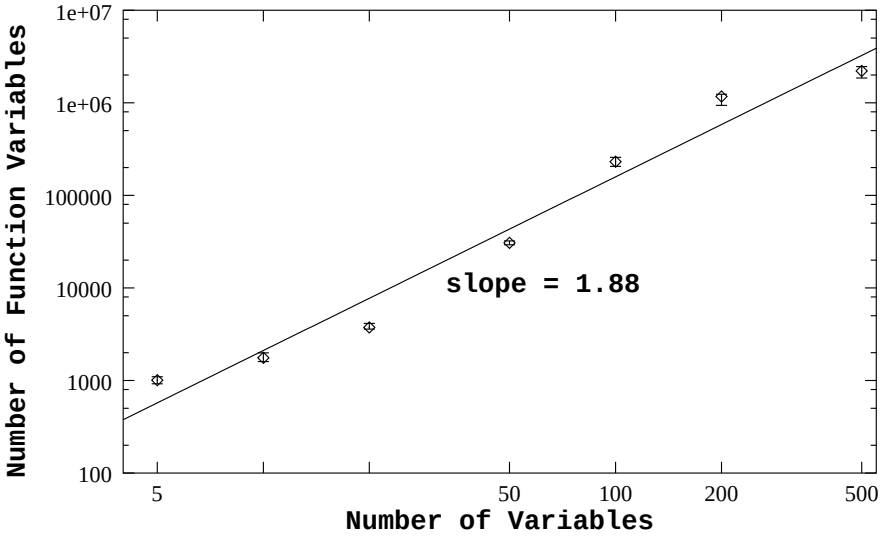

Figure 17: Scale-up study for the ellipsoidal function using the modified G3 model and the PCX operator.

from the best and worst function evaluations needed for the G3 model with the PCX operator to achieve an accuracy of 10^{-20}. These limiting function evaluations are also tabulated. The tolerances in the variables and in the function values are set to 10^{-30}. The table shows that the quasi-Newton method has outperformed the G3 model with PCX for the ellipsoidal function by achieving a better function value. Since the ellipsoidal function has no linkage among its variables, the performance of the quasi-Newton search method is difficult to match with any other method. However, it is clear from the table that the quasi-Newton method is not able to find the optimum with an accuracy of 10^{-20} within the allowed number of function evaluations in more epistatic problems (Schwefel's and Rosenbrock's functions).

7 Scalability of the Proposed Approach

In the above sections, we have considered only 20-variable problems. In order to investigate the efficacy of the proposed G3 model with the PCX operator, we have attempted to solve each of the three test problems with a different number of variables. For each case, we have chosen the population size and variances (σ_η and σ_ζ) based on some parametric studies. However, we have kept λ to a fixed value. In general, it is observed that for a problem with an increasing number of variables, a large population size and small variances are desired. The increased population size requirement with increased problem size is also in agreement with past studies (Goldberg et al., 1992; Harik et al., 1999). The reduced requirement for variances can be explained as follows. As the number of variables increases, the dimensionality of the search space increases. In order to search reliably in a large dimensional search space, smaller step sizes in variables must be chosen. Each case is run 10 times from different initial populations (initialized in $x_i \in [-10, -5]$ for all variables) and the best, median, and worst function evaluations needed to achieve a function value equal to 10^{-10} are presented.

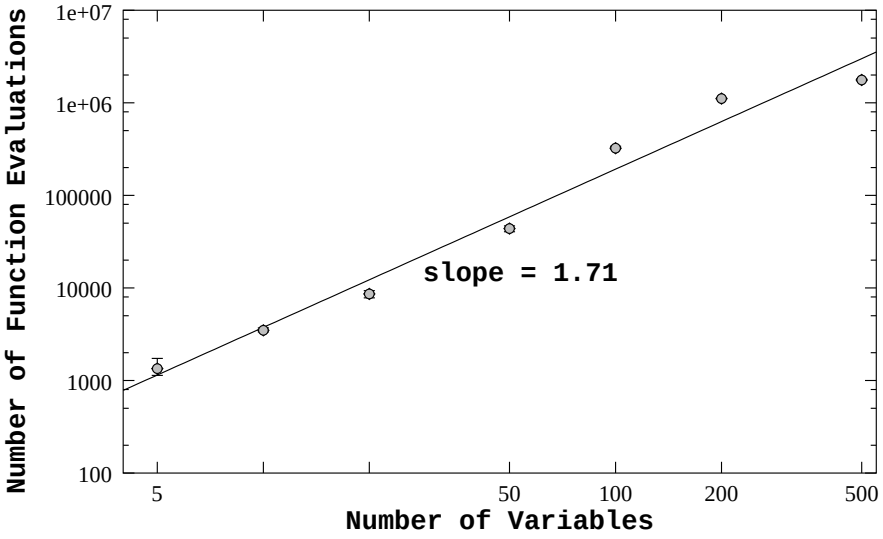

Figure 18: Scale-up study for the Schwefel's function using the modified G3 model and the PCX operator.

Figure 17 shows the experimental results for the ellipsoidal test problem having as large as 500 variables. The use of two offspring ($\lambda = 2$) is found to be the best in this case. The figure is plotted in a log-log scale. A straight line is fitted through the experimental points. The slope of the straight line is found to be 1.88, meaning that the function evaluations vary approximately polynomially as $O(n^{1.88})$ over the entire range of the problem size in $n \in [5, 500]$.

Next, we do a similar scale-up study for the Schwefel's function. Figure 18 shows the outcome of the study with $\lambda = 2$. A similar curve-fitting finds that the number of function evaluations required to obtain a solution with 10^{-10} varies as $O(n^{1.71})$ over the entire range $n \in [5, 500]$ of problem size.

Finally, we apply the modified G3 with PCX on the Rosenbrock's function. Because of the large number of function evaluations needed to solve a 500-variable Rosenbrock's function, we limit our study to a maximum of 200 variables. Figure 19 shows the simulation results and the fitted straight line on a log-log plot. The function evaluations needed to obtain a function value of 10^{-10} varies as $O(n^2)$. Interestingly, all the above experiments on three test problems show that the modified G3 model with the PCX operator needs a polynomially increasing number of function evaluations with an increasing problem size.

7.1 Rastrigin's Function

The ellipsoidal and Schwefel's functions have only one optimum, whereas the Rosenbrock's function is multimodal. To really investigate the performance of the G3 model with PCX and UNDX operators on multimodal problems further, we have chosen the

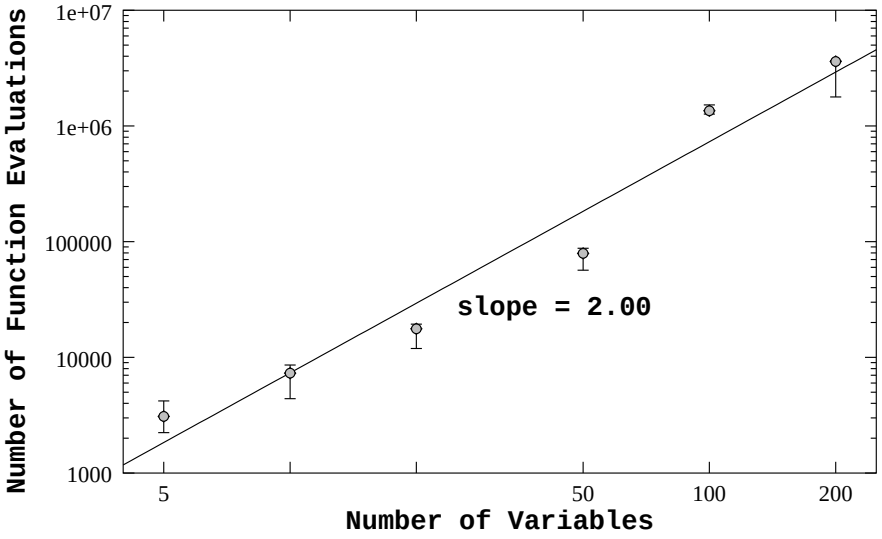

Figure 19: Scale-up study for the Rosenbrock's function using the modified G3 model and the PCX operator.

20-variable ($n = 20$) Rastrigin's function, involving many local minima:

$$F_{\text{rst}} = 10n + \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i)\right). \tag{6}$$

The global minimum function value is zero. Each integer corresponds to a local minimum. Unlike many past studies involving this function, we initialize the population randomly at $x_i \in [-10, -5]$. This initialization prevents a couple of important matters, which are ignored in many past studies involving this function:

1. The initial population is away from the global basin, thereby making sure that an algorithm must overcome a number of local minima to reach the global basin.

2. Such initialization prevents the advantage enjoyed by algorithms that have an inherent tendency to create solutions near the centroid of the parents.

Most past studies have initialized a population symmetrically about the global minimum (such as by initializing $x_i \in [-5.12, 5.12]$) to solve this function to global optimality. We consider this unfair, as a mean-centric recombination of two solutions on either side of $x_i = 0$ may result in a solution close to $x_i = 0$. Moreover, in most real-world problems, the knowledge of the exact optimum is usually not available, and the performance of an EA on a symmetric initialization may not represent the EA's true performance in solving the same problem with a different initialization or other problems.

Within the range $[-10, -5]$ for each variable, there exist six local minima. In order for an algorithm to reach the global minimum, it has to overcome four more local minima for each variable. We have tried varying different G3 model parameters, such as

λ, population size and variances. For both PCX and UNDX operators, no solution in the global basin is found in a maximum of one million function evaluations over multiple runs. From typical function values of the order of 10^3, which exist in the initial population, the G3 model with the PCX and UNDX operators finds best solutions with function values equal to 15.936 and 19.899, respectively. Since these function values are less than 20×1 (the best local minimum on each variable has a function value equal to one) or 20, at least one variable ($x_i \in [-0.07157, 0.07157]$) lies close to the global optimum value of $x_i = 0$. Although this itself is a substantial progress made by both models despite the existence of many local minima, it would be interesting to investigate if there exists a better global approach to solve this problem starting from an initial population far away from the global optimum.

8 Review of Current Results with Respect to Past Studies

There exists a plethora of past studies attempting to solve the four test problems used in this study. In this section, we put the results of this study in perspective to the past studies in which significant results on the above functions were reported.

8.1 Skewed Initialization

The need for a skewed initialization, in which the initial population is not centered around the global optimum, in tackling test problems with known optima is reported in a number of studies. Fogel and Beyer (1995) indicated that an initial population centered around the true optimum produces an undesired bias for some recombination operators. Based on this observation, Eiben and Bäck (1997) used a skewed initial population in their experimental studies with correlated self-adaptive evolution strategy. For the 30-variable Schwefel's function, an initialization of the population at $x_i \in [60, 65]$, the best reported solution (with a $(16, 100)$-ES) corresponds to a function value larger than 1.0 obtained with 100,000 function evaluations. For the 30-variable Rastrigin's function, the population was initialized at $x_i \in [4, 5]$, and the best function value larger than 10.0 was achieved with 200,000 function evaluations. Although the initialization is different from what we have used here, this study also showed the importance of using a skewed population in controlled experiments with test problems.

Chellapilla and Fogel (1999) solved the 10-variable Rastrigin's function by initializing the population at $x_i \in [8, 12]$. Compared to a symmetric initialization, this study showed negative improvement in best function values with the skewed initialization.

Patton et al. (1999) also considered a skewed initialization (but bracketing the minimum) for the 10-variable Schwefel's and Rosenbrock's functions. For a maximum of 50,000 function evaluations, their strategy found the best solution with $F_{\rm sch} = 1.2(10^{-4})$ and $F_{\rm ros} = 2.37$, which are much worse than our solutions obtained with an order of magnitude smaller number of function evaluations.

Deb and Beyer (2000) used real-parameter GAs with the SBX operator to solve an ellipsoidal function started with a skewed initial population. The convergence properties of the GA were found to be similar to that of a correlated ES.

8.2 Scale-Up Study

On the ellipsoidal problem, Salomon (1999) performed a scale-up study with 10, 100, and 1,000 variables. Since his *deterministic GA* (DGA) starts with variable-wise optimization requiring linear computational complexity, it is not surprising that the simulation study found the same for the ellipsoidal function. However, since our algorithm does not assume separability of variables, we cannot compare the performance of our

algorithm with that of the variable-wise DGA.

Higuchi et al. (2000) performed a scale-up study on the Rosenbrock's function with 10, 20, and 30 variables. In their simulation study, the population was initialized within $x_i \in [-2.048, 2.048]$. Even with this centered initialization, the MGG model with SPX required 275,896, 1,396,496, and 3,719,887 function evaluations, respectively. The original study did not mention the target accuracy used, but even if it were 10^{-20} as used here, we have found such high-accuracy solutions with much better computational complexities.

Kita (2000) reported a scale-up study on the Rosenbrock's function with 5, 10, and 20 variables. The MGG model with the UNDX operator ($\mu = 3$ and $\lambda = 100$) finds solutions with function values $2.624(10^{-20})$, $1.749(10^{-18})$, and $3.554(10^{-9})$, respectively, with a maximum of one million function evaluations. As we have shown earlier, the G3 model with the PCX operator requires much fewer function evaluations to obtain a much better solution (with a function value of 10^{-20} or smaller).

Hansen and Ostermeier (2000) applied their CMA evolution strategy to the first three functions up to 320 variables and reported interesting results. They started their algorithm one unit away from the global optimum in each variable (for example, for the Schwefel's function, $x_i^{(0)} = 1$ was used, whereas the optimum of F_{sch} is at $x_i^* = 0$). For ellipsoidal and Schewfel's functions, the scale-up is between $O(n^{1.6})$ to $O(n^{1.8})$, whereas for the Rosenbrock's function, it is $O(n^2)$. For all these functions, our study has also found similar computational complexities up to a wider range of problem sizes, despite the use of a more skewed initial population ($x_i \in [-10, -5]$) in our study. On the 20-variable Rastrigin's function, the CMA-ES was started with a solution initialized in $x_i \in [-5.12, 5.12]$, and function values within 30.0 to 100.0 were obtained with 2,000 to 3,000 function evaluations. Our study with a simple parent replacement strategy with offspring created with a computationally efficient vector-wise parent-centric recombination operator and without the need of any strategy parameter has shown a similar performance in all four test problems to that obtained using CMA-ES involving $O(n^2)$ strategy parameters to be updated in each iteration.

Wakunda and Zell (2000) did not perform a scale-up study but compared a number of real-parameter EAs to the three problems used in this study to find the corresponding optimum with an accuracy of 10^{-20}. For the 20-variable ellipsoidal function, more than 25,000 function evaluations, for the 20-variable Schwefel's function, more than 40,000 function evaluations, and for the 20-variable Rosenbrock's function, more than 90,000 function evaluations were the best results reported. Although this is the only study where solutions with an accuracy as high as that considered in our study were reported, the required function evaluations are much larger than that needed by the G3 model with the PCX operator.

9 Conclusions

The ad-hoc use of a uniformly distributed probability for creating an offspring often implemented in many real-parameter evolutionary algorithms (such as the use of SPX, BLX, or arithmetic crossover) and the use of a mean-centric probability distribution (such as that used in UNDX) have been found not to be as efficient as the proposed parent-centric approach in this paper. The parent-centric recombination operator favors solutions close to parents, rather than the region close to the centroid of the parents or any other region in the search space. Systematic studies on three 20-variable test problems started with a skewed population not bracketing the global optimum have shown that a parent-centric recombination is a meaningful and efficient way of

solving real-parameter optimization problems. In any case, the use of a uniform probability distribution for creating offspring has not been found to be efficient compared to a biased probability distribution favoring the search region represented by the parent solutions. Moreover, it has been observed that the offspring pool size used in previous real-parameter EA studies is too large to be computationally efficient.

Furthermore, the use of an elite-preserving, steady-state, scalable, and computationally fast evolutionary model (named as the G3 model) has been found to be effective with both PCX and UNDX recombination operators. In all simulation runs, the G3 model with the PCX operator has outperformed both the UNDX and SPX operators in terms of the number of function evaluations needed in achieving a desired accuracy. The proposed PCX operator has also been found to be computationally faster. Unlike most past studies on test problems, this study has stressed the importance of initializing an EA run with a skewed population.

To further investigate the performance of the G3 model with the PCX operator, we have compared the results with three other commonly used evolutionary algorithms: correlated self-adaptive evolution strategy, covariance matrix adaptation (CMA-ES), and differential evolution, and with a commonly used classical unconstrained optimization method, the quasi-Newton algorithm, with the BFGS update method. Compared to all these state-of-the-art optimization algorithms, the proposed model with the PCX operator has consistently and reliably performed better (in some cases, more than an order of magnitude better in terms of required function evaluations).

A scale-up study on three chosen problems over a wide range of problem sizes (up to 500 variables) has demonstrated a polynomial (of a maximum degree of two) computational complexity of the proposed approach. Compared to existing studies with a number of different GAs and a number of different self-adaptive evolution strategies, the proposed approach has shown better performance than most studies on the four test problems studied here. However, compared to a CMA-based evolution strategy approach (Hansen and Ostermeier, 2000), which involves an update of $O(n^2)$ strategy parameters in each iteration, our G3 approach with a computationally efficient procedure of creating offspring (with the PCX operator) and a simple parent replacement strategy has shown a similar scale-up effect to all test problems studied here. This study clearly demonstrates that the search power involved in the strategy-parameter based self-adaptive evolution strategies (CMA-ES) in solving real-parameter optimization problems can be matched with an equivalent evolutionary algorithm (G3 model) without explicitly updating any strategy parameter, but with an adaptive recombination operator. A similar outcome was also reported elsewhere (Deb and Beyer, 2000) showing the equivalence of a real-parameter GA (with the SBX operator) and the correlated ES. Based on the plethora of general-purpose optimization algorithms applied to these problems over the years, we tend to conclude that the computational complexities observed here (and that reported in the CMA-ES study as well) are probably the best that can be achieved on these problems.

Based on this extensive study and computational advantage demonstrated over other well-known optimization algorithms (evolutionary or classical), we recommend the use of the proposed G3 model with the PCX operator on more complex problems and on real-world optimization problems.

Acknowledgments

We thank Hajime Kita for letting us use his UNDX code. Simulation studies for the differential evolution strategy and the CMA-ES are performed with the C code down-

loaded from http://www.icsi.berkeley.edu/~storn/code.html and the MATLAB code downloaded from http://www.bionik.tu-berlin.de/user/niko, respectively. For the quasi-Newton study, the MATLAB optimization module is used.

References

Bäck, T. (1997). Self-adaptation. In Bäck, T., Fogel, D., and Michalewicz, Z., editors, *Handbook of Evolutionary Computation*, pages C7.1:1–15, Institute of Physics Publishing, Bristol, UK, and Oxford University Press, New York, New York.

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

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

Branch, M. A. and Grace, A. (1996). *Optimization toolbox for use with MATLAB*, The MathWorks, Inc., Natick, Massachusetts.

Chellapilla, K. and Fogel, D. B. (1999). Fitness distributions in evolutionary computation: Analysis of local extrema in the continuous domain. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC-1999)*, pages 1885–1892, IEEE Press, Piscataway, New Jersey.

Deb, K. (1995). *Optimization methods for engineering design*, Prentice-Hall, New Delhi, India.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. John Wiley and Sons, Chichester, UK.

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

Deb, K. and Beyer, H.-G. (2000). Self-adaptive genetic algorithms with simulated binary crossover. *Evolutionary Computation*, 9(2):197–221.

Eiben, A. E. and Bäck, T. (1997). Empirical investigation of multiparent recombination operators in evolution strategies. *Evolutionary Computation*, 5(3):347–365.

Fogel, D. B. and Beyer, H.-G. (1995). A note on the empirical evaluation of intermediate recombination. *Evolutionary Computation*, 3(4):491–495.

Goldberg, D. E., Deb, K., and Clark, J. H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex Systems*, 6(4):333–362.

Hansen, N. and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 312–317, IEEE Press, Piscataway, New Jersey.

Hansen, N. and Ostermeier, A. (2000). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195.

Harik, G. et al. (1999). The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–254.

Herrera, F., Lozano, M., and 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., and Yamamura, M. (2000). Theoretical analysis of simplex crossover for real-coded genetic algorithms. In Schoenauer, M. et al., editors, *Parallel Problem Solving from Nature (PPSN-VI)*, pages 365–374, Springer, Berlin, Germany.

Kita, H. (2000). A comparison study of self-adaptation in evolution strategies and real-coded genetic algorithms. *Evolutionary Computation* 9(2):223–241.

Kita, H. and Yamamura, M. (1999). A functional specialization hypothesis for designing genetic algorithms. *Proceedings of the 1999 IEEE International Conference on Systems, Man, and Cybernetics*, pages 579–584, IEEE Press, Piscataway, New Jersey.

Kita, H., Ono, I., and Kobayashi, S. (1999). Multi-parental extension of the unimodal normal distribution crossover for real-coded genetic algorithms. In Porto, V. W., editor, *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 1581–1587, IEEE Press, Piscataway, New Jersey.

Ono, I. and Kobayashi, S. (1997). A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover. In Bäck, T., editor, *Proceedings of the Seventh International Conference on Genetic Algorithms (ICGA-7)*, pages 246–253, Morgan Kaufmann, San Francisco, California.

Patton, A. L., Goodman, E. D., and Punch, W. F. (1999). Scheduling variance loss using population level annealing for evolutionary computation. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC-1999)*, pages 760–767, IEEE Press, Piscataway, New Jersey.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Frommann-Holzboog Verlag, Stuttgart, Germany.

Reklaitis, G. V., Ravindran, A., and Ragsdell, K. M. (1983). *Engineering Optimization Methods and Applications*. John Wiley and Sons, New York, New York.

Salomon, R. (1999). The deterministic genetic algorithm: Implementation details and some results. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC-1999)*, pages 695–702, IEEE Press, Piscataway, New Jersey.

Satoh, H., Yamamura, M., and Kobayashi, S. (1996). Minimal generation gap model for GAs considering both exploration and exploitation. In Yamakawa, T. and Matsumoto, G., editors, *Proceedings of the IIZUKA: Methodologies for the Conception, Design, and Application of Intelligent Systems*, pages 494–497, World Scientific, Singapore.

Schwefel, H.-P. (1987). Collective intelligence in evolving systems. In Wolff, W., Soeder, C. J., and Drepper, F., editors, *Ecodynamics – Contributions to Theoretical Ecology*, pages 95–100, Springer, Berlin, Germany.

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

Tsutsui, S., Yamamura, M., and Higuchi, T. (1999). Multi-parent recombination with simplex crossover in real-coded genetic algorithms. In Banzhaf, W. et al., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-99)*, pages 657–664, Morgan Kaufmann, San Francisco, California.

Voigt, H.-M., Mühlenbein, H., and Cvetković, D. (1995). Fuzzy recombination for the Breeder Genetic Algorithm. In Eshelman, L., editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 104–111, Morgan Kaufmann, San Francisco, California.

Wakunda, J. and Zell, A. (2000). Median-selection for parallel steady-state evolution strategies. In Schoenauer, M. et al., editors, *Proceedings of the Parallel Problem Solving from Nature, (PPSN-VI)*, pages 405–414, Springer, Berlin, Germany.