I.J. Mathematical Sciences and Computing, **2018, 4, 42-55**
Published Online November 2018 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijmsc.2018.04.04

Available online at **http://www.mecs-press.net/ijmsc**

An Efficient Genetic Algorithm for Numerical Function Optimization with Two New Crossover Operators

Abid Hussain [a,*], Yousaf Shad Muhammad [a], Muhammad Nauman Sajid [b]

[a] *Department of Statistics, Quaid-i-Azam University, Islamabad, Pakistan*
[b] *Department of Software Engineering, Foundation University, Islamabad, Pakistan*

Received: 20 April 2018; Accepted: 19 July 2018; Published: 08 November 2018

Abstract

Selection criteria, crossover and mutation are three main operators of genetic algorithm's performance. A lot of work has been done on these operators, but the crossover operator has a vital role in the operation of genetic algorithms. In literature, multiple crossover operators already exist with varying impact on the final results. In this article, we propose two new crossover operators for the genetic algorithms. One of them is based on the natural concept of crossover i.e. the upcoming offspring takes one bit from a parent and next from other parent and continuously takes bits till last one. The other proposed scheme is the extension of two-point crossover with the concept of multiplication rule. These operators are applied for eight benchmark problems in parallel with some traditional crossover operators. Empirical studies show a remarkable performance of the proposed crossover operators.

Index Terms: Genetic algorithms, Crossover operators, Benchmark functions, Comparison.

1. Introduction

Genetic algorithms (GAs) are stochastic-based approaches which depend on biological evolutionary processes proposed by John Holland in the 1960s. He discussed the GAs in his book "Adaptation in Natural and Artificial Systems" published in the 1975 [1]. GAs rely on one of the most important criteria of Darwin: survival of the fittest. GAs established some codes on population chromosomes which work under some assumptions. Goldberg [2] discussed the GAs with various applications cited in the literature.

GAs [3-10] are robust and heuristic-based optimization methods. These methods mainly focus on selection

* Corresponding author.
E-mail address: abid0100@gmail.com

processes which are very common in genetics. The individuals which are environment-friendly transfer their genetic properties to the next generations. Application of GAs is also common in operation research, sciences and engineering etc. The main advantage of GAs over other optimization techniques is that they have not stuck off on local optima most of the time. GAs separate objective function from the related constraints when it is used as search criterion. A detailed study of university course timetabling problem using multi population hybrid GAs [11].

GAs start from the randomly generated strings of the initial population. The performance of each string is measured to evaluate its departure from the optimal solution when used as an objective function. All possible candidate solutions are evaluated one by one to find the best one. Genetic information is exchanged between these candidates after they are nominated as parents. The crossover is an exchange of information from parents to offspring. Mutation randomly changes within some of these offspring, which introduces change among the strings of the population. The population created in such a manner replaces the old one after crossover and mutation operators. This process continues until its convergence criterion is met. A generation is a complete cycle of all these processes. The sketch of a typical genetic algorithm is depicted in Fig. 1.

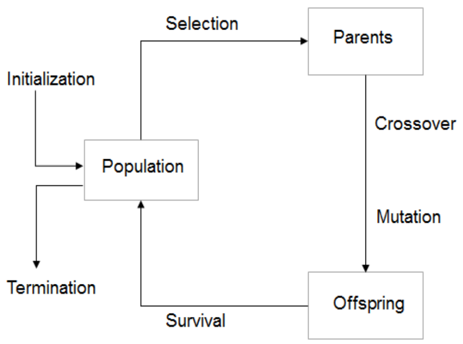

Fig.1. Layout of a Typical Genetic Algorithm

The main objective of this study is to presents the performance of crossover operators that have the major impact on the GAs process. The purpose of crossover operator is to vary the individual's quality by combining the desired characteristics from both parents. A comprehensive study about various crossover operators has been performed and some suggestions for selection among these operators have been introduced and reviewed [12]. Some of the traditional crossover operators are; one-point, two-point, multi-point, uniform, discrete, heuristic and arithmetic etc. A comprehensive study shows that with large search space, the GAs using uniform crossover outperform the ones using one-point and two-point crossover operators [13]. In general, however, their theory also suggested that GAs with crossover should outperform those without crossover operator (i.e. mutation only).

A large number of binary-represented crossover operators used in GAs and results confirmed with a high efficiency of two-point and uniform crossover operators [14]. GAs application to dimension optimization problems to compare various crossover operators and suggested that two-point crossover operator is best [15-17]. A comparison study among four binary crossover operators was done by Wu and Chow [18]. A theory suggests that multi-point crossover operator in terms of fast progress becomes very slow when compared to the one-point crossover [19]. A comparative study of different crossover operators with varying population size suggests that two-point crossover performs better when population size is large, otherwise uniform crossover operator [20]. An author showed in his study that uniform crossover operator is more efficient when compared with two-point crossover [21]. A study about the effect of different behavior of crossover operators found improved results of combining the two crossover operators [22, 23]. The other study is conducted with assessing the performance of traveling salesman problem by using various crossover operators [24]. In this study, we also proposed two new crossover operators (forward-backward and same-opposite crossover) to check the behavior of GAs with other traditional crossover operators on several benchmarks.

Rest of the paper is organized as: Section 2 reprints the background of crossover operators. In Section 3, we proposed two new operators, related benchmark functions are discussed in Section 4. Results and discussion are presented in Section 5 and Section 6 is focused on our conclusion.

2. Crossover Operators

GAs can rapidly identify discrete regions within a huge search space to concentrate the search for an optimum solution. This approach changes mutually defined parts of two selected individuals and obtained different individuals that give an opportunity to locate the optimum with new points in search space. The crossover operators used in this study are summarized below.

2.1. One-point Crossover Operator

The simplest crossover technique for GAs is one-point crossover operator. Using this technique, first, select two parents and then randomly selects a crossover site from the interval [1, *l*-1], where *l* is the fixed length of the string. All bits beyond the crossover site in either organism string is swapped between the two parent organisms [22, 25, 26]. This approach is depicted in Fig. 2.

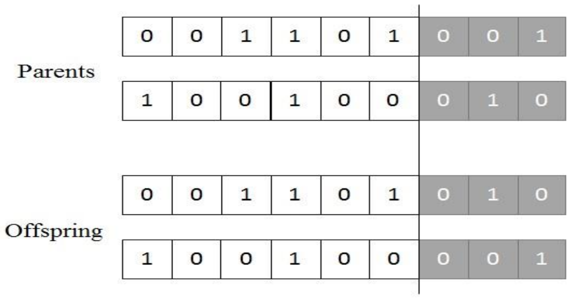

Fig.2. One-point Crossover Operator

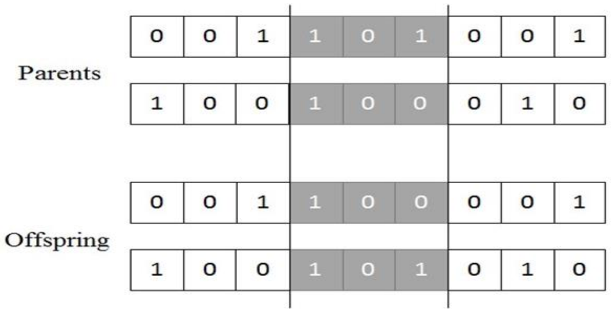

Fig.3. Two-point Crossover Operator

2.2. Two-point Crossover Operator

In this approach, select two points in the interval [1, *l*-1], where *l* is the fixed length of the string. After two selected crossover sites, the contents between these points or outer portions are exchanged between two mated individuals to produce new individuals [22, 25, 26]. Note that the resulting individuals will be the same in both cases. This crossover operator specifically results in higher chance as compared to one-point crossover for all individuals who are exchanging their important bits within their chromosomal strings [27]. This approach is depicted in Fig. 3.

2.3. Multi-point Crossover Operator

In this approach, select many cut-off points in the interval [1, l-1], where l is the fixed length of the string. In order to separate the sites into (n_c +1) portions for individuals, we will select ($n_c \geq 3$) crossover sites at first. For completion of this procedure, any one of the two groups of portions as a whole will be exchanged. The first group includes the set of {1st , 3rd … (2k - 1)th , where k = 1,2,...,int(n_c/2 + 1), while the other group comprises rest of the portions, not being selected in first group [22]. This approach is depicted in Fig. 4.

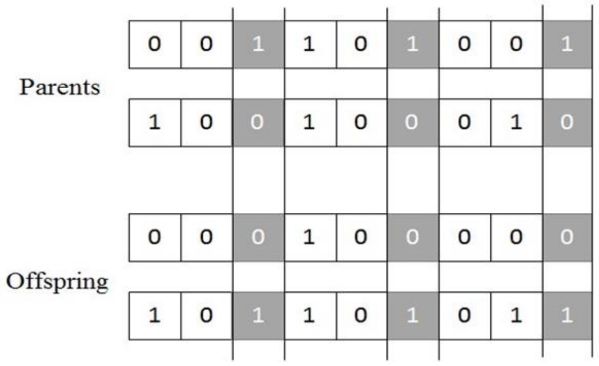

Fig.4. Multi-point Crossover Operator

2.4. Uniform Crossover Operator

This is quite different from previous crossover operators and the idea was first used Syswerda [21]. After selected parents, introduce a new randomly generated binary crossover mask according to same length of the string. The bits of offspring are duplicated from the parents as per the bits of their mask. If it is a “0” in the binary crossover mask, the bit is copied from the second parent otherwise from the first parent. The second offspring may be generated either by introducing a new mask [28] or by using complementary of the original mask [21]. This approach is depicted in Fig. 5.

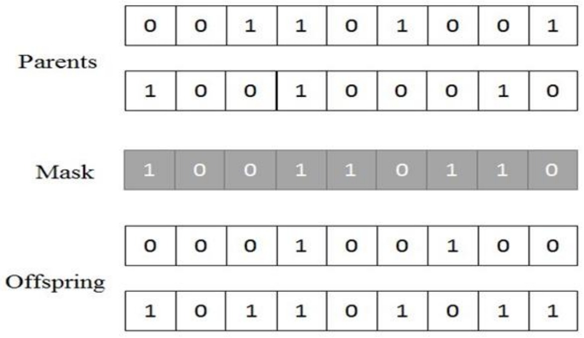

Fig.5. Uniform Crossover Operator

3. Proposed Crossover Schemes

In search of optimum, a solid crossover operator generally uses two significant techniques, namely; exploration and exploitation. In exploration strategy, a deep and swift discovery examination of the design space is required. Every authentic exploration results in preventing the search of the local optimum. However, this rule is not a sufficient condition to obtain more suitable points. Whereas the exploitation technique uses already detected points to approaches the optimum. This process generally ends in slow convergence along with increased risk of locating a local optimum. Having opposite status for these two techniques, a balanced use of these will be more efficient while searching through crossover operator [22, 29].

In this section of the study, we present two newly developed crossover operators for GAs. They are used as alternative approaches of crossover operator which have a great balance of exploration and exploitation. First one is purely based on selecting the genes for both upcoming offspring from both parents alternately. Our second approach is the extension of the two-point crossover technique. The details about these operators are in the following subsections.

3.1. Forward-Backward Crossover Operator

In nature, offspring is the mixture of two parents and normally healthier than them if parents are healthy. As we know that GAs work with binary numbers, so for offspring, one bit from a parent and the next bit from the other and the process continues till last bit. For the first offspring, take the first bit from the first parent, second bit from the second parent, third one from the first parent and so on. For this way of moving, we suggest the name of the approach is forward-backward crossover (FBX) operator. We can also suggest it another name with odd-even crossover (OEX) because of first offspring take all odd-order bits from first parent and even-order bits from second parent and vice versa for the second offspring. Given two selected chromosomes (strings) having "m" genes (bits) as parents P_u and P_v in the following form:

$$p_u = (a_{u1}, a_{u2}, \dots, a_{um})$$

$$p_v = (a_{v1}, a_{v2}, \dots, a_{vm})$$

Before exploring the scheme to produce offspring, we divide it into two cases depending on the number of bits i.e. "*m*" is even or odd. We also introduce another term "k", which is the subscript of two selected parents. In above selected parents *k* = u, v.

If "*m*" is even then offspring C_i, where "*i*" is the subscript of required offspring as:

$$C_i = \begin{cases} c_{ij} = a_{ij}, & j = 2n-1; n = 1,2,3,\dots,\dfrac{m}{2} \\ c_{ij} = a_{kj}, & j = 2n; n = 1,2,3,\dots,\dfrac{m}{2}; k \neq i \end{cases}$$

If "*m*" is odd then offspring C_i, where "*i*" is the subscript of required offspring as:

$$C_i = \begin{cases} c_{ij} = a_{ij}, j = 2n-1; n = 1,2,3,\dots,\dfrac{m+1}{2} \\ c_{ij} = a_{kj}, j = 2n-2; n = 1,2,3,\dots,\dfrac{m+1}{2}; k \neq i \end{cases}$$

For more simplicity this approach is depicted in Fig. 6.

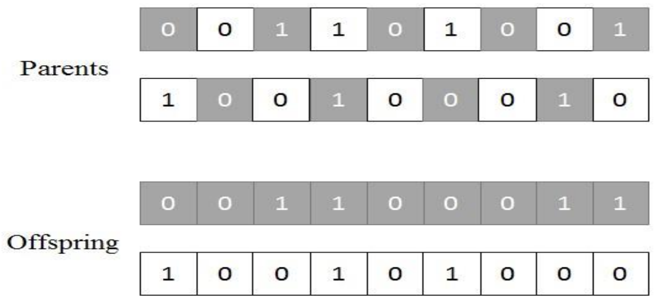

Fig.6. FBX Operator

We see later in results that this approach gives a fast convergence of some given benchmark functions. We used MATLAB software with version R2015a to apply this new approach and perform its measure along with some traditional crossover operators. To apply this crossover operator, we made a MATLAB function “xoverforwarbackward” for the GAs tool as crossover operator for custom used. The pseudo code of our algorithm to verify its fast convergence is given in the Algorithm 1.

Algorithm 1. The Pseudo-code of FBX-Operator

```
No. of kids ← Half the no. of parents
N ← 1
While N ≤ No. of kids do
        P1 ← select random parent
        P2 ← select random parent
        C1 ← P1 (1: 2: length (P1)) // Generated Child1
        C1 ← P2 (2: 2: length (P2))
        C2 ← P2 (1: 2: length (P2)) // Generated Child2
        C2 ← P1 (2: 2: length (P1))
end while
```

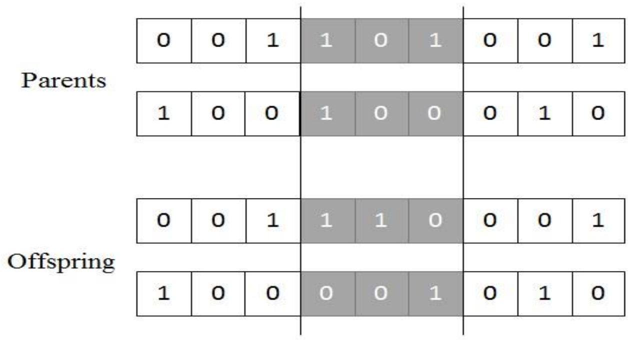

Fig.7. SOX Operator

3.2. Forward-Backward Crossover Operator

In this approach, we combined our previous work [30] and two-point crossover [22, 25, 26] operators. As usual in two-point crossover operator, select two sites in the interval [1, $l - 1$], where l is the fixed length of the string. After selecting the cut points, take "1" if both parallel bits are same and otherwise "0" for one offspring and vice versa for other offspring. We suggest the name of this novel approach is same-opposite crossover (SOX) operator. This approach is depicted in Fig. 7.

To apply this crossover operator, we made another MATLAB function "xoversameopposite" for the GAs tool as crossover operator for custom used. The pseudo code of our algorithm to verify its fast convergence is given in the Algorithm 2.

Algorithm 2. The Pseudo-code of SOX-Operator

```
No. of kids ← Half the no. of parents
N ← 1
while N ≤ No. of kids do
        P1 ← select random parent
        P2 ← select random parent
        Cut1 ← random value, $c_1 \in (0, l/2)$
        Cut2 ← random value, $c_2 \in (l/2, l)$
        Track1 ← Not[XOR(P1(Cut1:Cut2), P2(Cut1:Cut2))]
        Track2 ← XOR(P1(Cut1:Cut2), P2(Cut1:Cut2))
        Child1 ← [P1(1:Cut1), Track1, P1(Cut2:End)]
        Child2 ← [P2(1:Cut1), Track2, P2(Cut2:End)]

end while
```

4. The Benchmarks

There is not a hard-and-fast rule to choose a suitable optimization function to verify the performance of an algorithm. An application to application variation in a function depends on the nature of optimization problem, in terms of the rate of variation in the objective function, the number of local optima etc. [31]. A multimodal function has at least two local optima. In order to search for global optimum, the search process must be capable of eliminating the region around local optimum. The situation becomes more complicated in case of random distribution of local optima in search space. There is another important factor in the complexity of the problem is the dimensionality of the search space [32]. A detailed study of the dimensionality problem and its features was carried out by Friedman [33]. In order to compare the performance of the proposed crossover operators, we have used eight multimodal classical benchmark functions. These benchmarks have varying complexities that are most popular and used in several studies. The necessary information about these benchmarks are as follows.

4.1. Uneven Decreasing Maxima Function

The uneven decreasing maxima function is one of the multimodal optimization problems. There are four local optima and one global optimum but the maxima decrease in height exponentially. This function was originally proposed in [32]. It is stated as follows:

$$F_1 = \exp\left(-2\log(2)\left(\frac{\theta-0.08}{0.854}\right)^2\right) sin^6\left(5\pi\left(\theta^{\frac{3}{4}} - 0.05\right)\right) \tag{1}$$

Where $\theta \in [0, 1]$, the global maximal point of the function is at $\theta = 0.08$ and $F_1 = 1$.

4.2. Himmelblau Function

In mathematical optimization, Himmelblau's function is a multimodal function, used to test the performance of optimization algorithms, originally proposed by Himmelblau [35]. But we used the inverted version of this function which is already used in [34, 36]. The function is defined by:

$$F_2 = 200 - (\theta_1^2 + \theta_2 - 11)^2 - (\theta_1 + \theta_2^2 - 7)^2 \tag{2}$$

Where $\theta_i \in [-6, 6]$, it has no local optimum and four global optima at approximately (3.58,-1.86), (3.0, 2.0), (-2.815, 3.125), and (-3.78,-3.28) and $F_2 = 200$.

4.3. Colville Function

This benchmark has a highly scattered pattern of convergence taken from [37]. It is stated as follows:

$$F_3 = 200(\theta_2 - \theta_1^2)^2 + (1-\theta_1)^2 + 90(\theta_4 - \theta_3^2)^2 + (1-\theta_3)^2 + 10.1((\theta_2 - 1)^2 + (\theta_4 - 1)^2) + 19.8(\theta_2 - 1)(\theta_4 - 1) \tag{3}$$

Where $\theta_i \in [-10, 10]$, the global optimum point of the function is at (1, 1, 1, 1) and $F_3 = 0$. This function is highly multimodal and difficult to locate its optimum points due to its more dimensions.

4.4. Six-Hump Camel Back Function

This function has six local optima, two of them are global optima within the bounded region. It has a highly scattered pattern of convergence taken from [37]. It is stated as follows:

$$F_4 = \left(4 - 2.1\,\theta_1^2 + \frac{1}{3}\theta_1^4\right)\theta_1^2 + \theta_1\theta_2 + (-4 + 4\,\theta_2^2)\,\theta_2^2 \tag{4}$$

Where $\theta_{\mathrm{i}} \in [-3, 3]$ and $\theta_2 \in [-2, 2]$, the global optimum value of the function is $F_4 = -1.0316$ at two different points (-0.0898, 0.7126) and (0.0898, -0.7126).

4.5. Gold-Price Function

This function has been taken from [38]. It is stated as follows:

$$F_5 = [1 + (\theta_1 + \theta_2 + 1)^2(19 - 14\theta_1 + 3\theta_1^2 - 14\theta_2 + 6\theta_1\theta_2 + 3\theta_2^2)] \times [30 + (2\theta_1 - 3\theta_2)^2(18 - 32\theta_1 + 12\theta_1^2 + 48\theta_2 - 36\theta_1\theta_2 + 27\theta_2^2)] \tag{5}$$

Where $\theta_{\mathrm{i}} \in [-2, 2]$ the function has a global minimum value of the function is $F_5 = 3$ at (0, -1).

4.6. Easom Function

The Easom test function is our sixth benchmark which has the global optimum fall in a small area relative to the search space taken from [39]. It is stated as follows:

$$F_6 = -cos(\theta_1)cos(\theta_2)exp[-(\theta_1 - \pi)^2 - (\theta_2 - \pi)^2] \tag{6}$$

Where $\theta_i \in [-100,100]$, the function has a global optimum value of the function is $F_6 = -1$ at (π, π).

4.7. Rastrigin Function

The Rastrigin function, which has been taken from De Jong's standard functions [40] is being used here to produce frequently local optima with the addition of cosine modulation. Since it is a complex function and many algorithms generally are trapped on local optima, hence we have selected this function as a test function to compare our proposed operators. The optimization problem is stated as follows:

$$F_7 = \sum_{i=1}^{D}(10 + \theta_i^2 - 10\cos(2\pi\theta_i) \tag{7}$$

Where $\theta_i \in [-5.12, 5.12]$, the function has a global optimum value of the function is $F_7 = 0$ at (0,0,...,0).

4.8. Rosenbrock Function

This function is a classic optimization benchmark and also known as a banana function because of its distinctive shape in a contour plot. The global optimum lies inside a narrow, long, parabolic shaped flat valley and to find an optimum point is trivial. Owing to the difficulty in converging its global optimal, this function is commonly used to test the performances of many optimization algorithms [40]. It is stated as follows:

$$F_8 = \sum_{i=1}^{D}(100\,(\theta_i^2 - \theta_{\mathrm{i+1}})^2 \ + (1-\theta_i\,)^2\,) \tag{8}$$

Where $\theta_i \in [-2.048, 2.048]$, the function has a global optimum value of the function is $F_8 = 0$ at (1,1,...,1).

5. Experimental Results and Discussion

While comparing the proposed crossover operators with others, we used GAs tool of MATLAB version R2015a. The global optimum is the objective for all test functions. One of the main difficulties in building a practical genetic algorithm (GA) is in choosing suitable values of parameters such as population size, scaling function, selection criteria, elite count, the probability of crossover (P_c), mutation function and function tolerance etc. The selections of parameter value are varied to depend on the problem to be solved. Dimensionality of the search space D for F_7 and F_8 was set to 10. All compared crossover operators were executed 30 times (30 runs) with different and randomly chosen initial population for each function. Other control parameters of the algorithm for each experiment are given in Table 1.

Table 1. Fixed Parameters for GAs Tool

Population Size	50
Scaling Function	Proportional
Selection Operator	Roulette Wheel
Elite Count	6%
Crossover Probability	80%
Mutation Operator	Swap
Mutation Probability	10%
Function Tolerance	1e-6

In order to show the performance of the crossover operators more clearly with the help of average and standard deviation (S.D) in Table 2. For function F_1, the proposed FBX outperforms among all other operators with zero variation and all other operators work as a similar pattern. For functions F_2, F_4 and F_6, all given crossover operators give optimal results for all 30 runs. The proposed SOX outperforms among all other operators with zero variation for functions F_3 and F_8. Results indicate that it has a more robust investigation and sensitivity with respect to the number of runs for functions F_3. No other operator gives optimal result not even in a single run of this benchmark but works as a similar pattern. For function F_5, only One-point crossover give optimal result in all 30 runs and proposed operators give optimum 17 and 13 out of 30 times for this function by FBX and SOX respectively. This function is more sensitive to fewer generations but the performance is improved when generations will enhance [38]. Both proposed crossover operators give optimum value in all 30 runs for function F_7 and no other operator behave like them for this function. The simulated results show that the overall performance of the proposed operators outperforms among all operators.

Since the goal of our proposed approaches are to prevent the convergence of the GA to a local optimum also with evaluating the performance in terms of the number of runs for which the GA gets stuck at a local optimum. For this point of view, a box-and-whisker plot is one of the best choices to show the variations in the simulation results. An outlier can easily be detected with the help of a box-and-whisker plot. Typically, a value in the data set is considered to be an outlier if it is greater than the third quartile by more than 1.5 times the interquartile range or if it is less than the first quartile by more than 1.5 times the interquartile range. So we display the benchmark functions results with the help of box-and-whisker plots in two figures. Fig. 8 represents the results of functions F_1 to F_4 and Fig. 9 for functions F_5 to F_8.

In Fig. 8, one of the proposed crossover operator FBX (used in the figure as FB) shows no variation in all 30 independent runs for the function F_1. Same work by the second proposed operator SOX (used in the figure as SO) for F_3. So for these two functions, to choose the crossover operator is sensitive to get the optimum results and proposed operators are the best choice. For functions F_2 and F_4, the GA work with efficient results by using any used operator, because crossover operator is not sensitive for these functions.

In Fig. 9, one of the traditional crossover operator One-point (used in the figure as 1P) shows no variation in all 30 independent runs for function the F_5. For F_6, all operators work on a similar pattern with zero dispersion. Both proposed operators work outperforms for the function F_7. The second proposed operator is the best choice for function F_8.

For significant point of view, the One-point operator gives optimum results in four out of eight benchmarks, its mean 50% the GA stuck-off on local optima when using this operator. Also, 70% the algorithm stuck-off on local optima when used Two-point, Multi-point and Uniform crossover operators. But the proposed operators work efficiently for these benchmark functions and only stuck-off 30% and 20% on local optima by FBX and SOX respectively.

Table 2. Results Obtained by Different Crossover Operators

		Crossover Operator											
		One-point		Two-point		Multi-point		Uniform		FBX		SOX	
Function	Optimal	Average	S.D	Average	S.D	Average	S.D	Average	S.D	Average	S.D	Average	S.D
F1	1.0000	0.9829	0.0245	0.9804	0.0452	0.9474	0.0913	0.9932	0.0177	1.0000	0.0000	0.9770	0.0455
F2	200.00	200.00	0.0000	200.00	0.0000	200.00	0.0000	200.00	0.0000	200.00	0.0000	200.00	0.0000
F3	0.0000	1.9084	2.5772	3.5617	2.6618	4.0009	3.3269	4.9041	3.6706	3.2219	2.9355	0.0000	0.0000
F4	-1.0316	-1.0316	0.0000	-1.0316	0.0000	-1.0316	0.0000	-1.0316	0.0000	-1.0316	0.0000	-1.0316	0.0000
F5	3.0000	3.0000	0.0000	6.3167	7.0949	7.7461	7.1690	8.2605	7.6724	4.3323	2.3363	5.5489	3.2050
F6	-1.0000	-1.0000	0.0000	-1.0000	0.0000	-1.0000	0.0000	-1.0000	0.0000	-1.0000	0.0000	-1.0000	0.0000
F7	0.0000	0.9980	0.8304	1.3294	0.7572	0.9969	0.8289	0.8641	0.8183	0.0000	0.0000	0.0000	0.0000
F8	0.0000	3.9967	2.2158	5.1650	2.7961	4.5619	2.4692	4.4521	2.6130	2.4105	1.6580	0.0000	0.0000

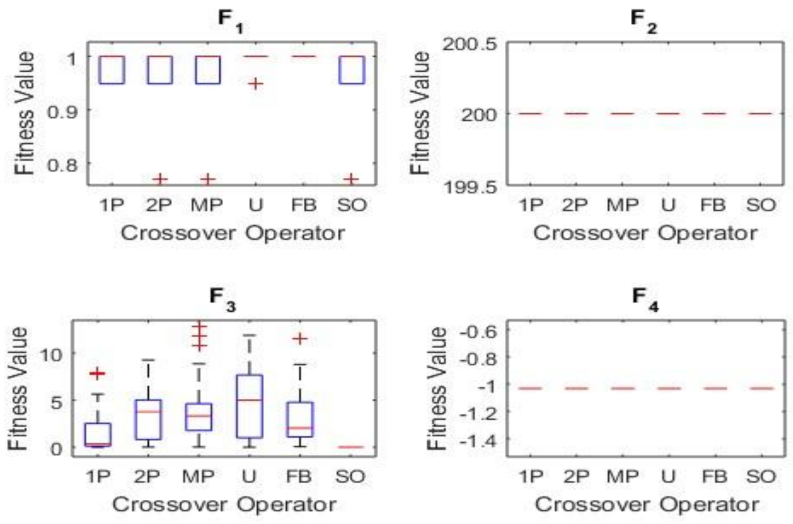

Fig.8. Dispersion among 30 runs of functions F_1 to F_4

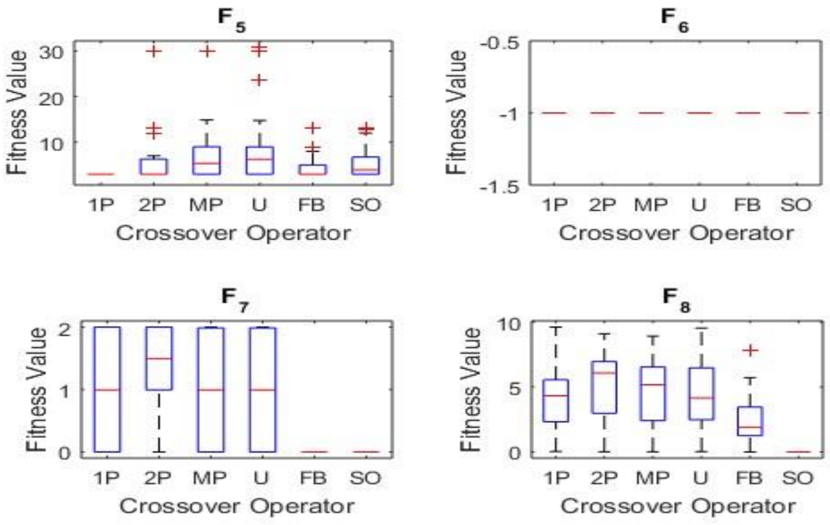

Fig.9. Dispersion among 30 runs of functions F_5 to F_8

6. Conclusions

Unlike other heuristic methods, GA uses natural rules of selection, crossover and mutation to make the computation easier and fast. These things make it more valuable, better performing and efficient algorithm over those. Various crossover operators have been presented for GAs in literature. In this article, we also introduced two new and efficient crossover schemes for GAs. The proposed approaches provide comparative convergence rate. The performance of each crossover method at the average results and also observed their sensitivity in different runs with the help of an absolute measure S.D. is compared through the implementation in a MATLAB program. On the set of multimodal testing functions, we select a set of experiments of varying difficulty levels. Our results in the comparative study provide the evidence of improvement in performance. The novel schemes might be applied to the optimization problems to compare with various benchmarks. After the results of benchmark functions of this study, we suggest that proposed crossovers may be good candidates as a crossover operator to get fast and accurate results in problems of high complexity in the fields of optimization.

Disclosure Statement

No conflict of interest is declared.

References

[1] Holland, J.H. Adaptation in Natural and Artificial Systems. Ann Arbor, MI, USA: University of Michigan Press; 1975.

[2] Goldberg, D.E. Genetic Algorithms in Search, Optimization and Machine Learning. Boston, MA, USA: Addison-Wesley Longman Publishing; 1989.

[3] Datta, D. Unit commitment problem with ramp rate constraint using a binary-real- coded genetic algorithm. Appl Soft Comput. 2013; 13:3873-3883.

[4] Dudek, G. Genetic algorithm with binary representation of generating unit start-up and shut-down times for the unit commitment problem. Expert Syst Appl. 2013; 40:6080-6086.

[5] Sun, L., Zhang, Y., Jiang, C. A matrix real-coded genetic algorithm to the unit commitment problem. Electr Pow Syst Res. 2006; 76:716-728.

[6] Dudek, G. Unit commitment by genetic algorithm with specialized search operators. Electr Pow Syst Res. 2004; 72:299-308.

[7] Swarup, K., Yamashiro, S. Unit commitment solution methodology using genetic algorithm. IEEE T Power Syst. 2002; 17:87-91.

[8] Dang, C., Li, M. A floating-point genetic algorithm for solving the unit commitment problem. Eur J Oper Res. 2007; 181:1370-1395.

[9] Pavez-Lazo, B., Soto-Cartes, J. A deterministic annular crossover genetic algorithm optimization for the unit commitment problem. Expert Syst Appl. 2011; 38:6523-6529.

[10] Bukhari, S.B., Ahmad, A., Raza, S.A., Siddique, M.N. A ring crossover genetic algorithm for the unit commitment problem. Turk J Elec Eng & Comp Sci. 2016; 24:3862-3876.

[11] Kohshori, M.S., Zeynolabedini, D., Liri, M.S., Jadidi, L. Multi Population Hybrid Genetic Algorithms for University Course Timetabling Problem. International Journal of Information Technology and Computer Science. 2012; 4(6):1-11.

[12] Caruana, R.A., Eshelman, L.J., Schaffer, J.D. Representation and hidden bias II: Eliminating defining length bias in genetic search via shuffle crossover. In: Proceedings of the 11th International Joint Conference on Artificial intelligence. 1989; 750-755.

[13] Spears, W.M., Anand, V. A study of crossover operators in genetic programming. In: International Symposium on Methodologies for Intelligent Systems; Springer, Berlin, Heidelberg. 1991;409-418.

[14] Picek, S., Golub, M. Comparison of a crossover operator in binary-coded genetic algorithms. WSEAS Trans. Comput. 2010; 9:1064-1073.

[15] Adeli, H., Cheng, N.T. Integrated genetic algorithm for optimization of space structures. J AEROSPACE ENG. 1993; 6:315-328.

[16] Adeli, H., Cheng, N.T. Augmented Lagrangian genetic algorithm for structural optimization. J AEROSPACE ENG. 1994; 7:104-118.
[17] Adeli, H., Cheng, N.T. Concurrent genetic algorithms for optimization of large structures. J AEROSPACE ENG. 1994; 7:276-296.
[18] Wu, S.J., Chow, P.T. Steady-state genetic algorithms for discrete optimization of trusses. COMPUT STRUCT. 1995; 56:979-991.
[19] Jenkins, W.M. On the application of natural algorithms to structural design optimization. ENG STRUCT. 1997; 19:302-308.
[20] DeJong, K.A., Spears, W.M. An analysis of the interacting roles of population size and crossover in genetic algorithms. In: International Conference on Parallel Problem solving from Nature; Springer, Berlin, Heidelberg. 1990; 38-47.
[21] Syswerda, G. Uniform crossover in genetic algorithms. In: Proceedings of the 3rd International Conference on Genetic Algorithms. 1989; 2-9.
[22] Hasanc ̧ebi, O., Erbatur, F. Evaluation of crossover techniques in genetic algorithm based optimum structural design. COMPUT STRUCT. 2000; 78:435-448.
[23] Zaharie, D. Influence of crossover on the behavior of differential evolution algorithms. APPL SOFT COMPUT. 2009; 9:1126-1138.
[24] Khan, I.H. Assessing Different Crossover Operators for Traveling Salesman Problem. International Journal of Intelligent Systems and Applications. 2015; 7(11):19-25.
[25] Kellegoz, T., Toklu, B., Wilson, J. Comparing efficiencies of genetic crossover operators for one machine total weighted tardiness problem. Applied Mathematics and Computation. 2008; 199:590-598.
[26] Reeves, C.R. Genetic algorithms. In Handbook of Metaheuristics, pp. 109-139. Springer, Boston, MA; 2010.
[27] Back, T. Evolutionary algorithms in theory and practice. New York, Oxford university press; 1996.
[28] Beasley, D., Bull, D.R., Martin, R.R. An overview of genetic algorithms: Part 2, research topics. University computing. 1993; 15:170-181.
[29] Kaya, M. The effects of two new crossover operators on genetic algorithm performance. Applied Soft Computing. 2011; 11:881-890.
[30] Hussain, A., Muhammad, Y.S., Nawaz, A. Optimization through genetic algorithm with a new and efficient crossover operator, International Journal of Advances in Mathematics. 2018; 1:1-14.
[31] Srinivas, M., Patnaik, L.M. Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics. 24, (1994) 656-667.
[32] Ortiz-Boyer, D., Hervas-Martinez, C., Garcia-Pedrajas, N. CIXL2: A Crossover Operator for Evolutionary Algorithms Based on Population Features. J. Artif. Intell. Res. (JAIR). 2005; 24:1-48.
[33] Friedman, J.H. An overview of predictive learning and function approximation. *In From Statistics to Neural Networks*. Springer, Berlin, Heidelberg. 1994; 1-61.
[34] Deb, K. Genetic algorithms in multimodal function optimization. PhD diss., Clearinghouse for Genetic Algorithms, Department of Engineering Mechanics, University of Alabama; 1989.
[35] Himmelblau, D.M. Applied nonlinear programming. McGraw-Hill Companies; 1972.
[36] Li, X., Engelbrecht, A., Epitropakis, M. Benchmark functions for CEC’2013 special session and competition on niching methods for multimodal function optimization, Tech. rep., RMIT University; 2013.
[37] Michalewicz, Z. Genetic Algorithms + Data Structures= Evolution Programs. 3rd ed. London, UK: Springer-Verlag; 1996.
[38] Goldstein, A.A., Price, J.F. On descent from local minima. MATH COMPUT. 1971; 25:569574.
[39] Easom, E.E. A survey of global optimization techniques. PhD, University of Louisville, Kentucky, United States; 1990.
DeJong, K.A. Analysis of the behavior of a class of genetic adaptive Systems. PhD, Dept.Computer and Communication Sciences, University of Michigan, Ann Arbor; 1975.

Authors' Profiles

Mr. Abid Hussain is currently Ph.D. (Statistics) scholar. He did his master in 2012. His areas of interest are Operation Research, Stochastic Processes, Optimization Theories and Computational Mathematics.

Dr. Yousaf Shad Muhammad is currently working as a Assistant Professor (Statistics) in Quaid-i-Azam University, Islamabad, Pakistan. He did his Ph.D. from University of Vienna, Austria in 2006. He was selected as a YSSPer, IIASA (first researcher of Pakistan). He completed a two-year postdoctoral fellowship in NTNU Norway. His research interests are Stochastic Modelling and Optimization, Network Optimization and Computing, Supply and Value Chain Design and Optimization, Scenario Generation, Credit Risk Modelling.

Mr. Muhammad Nauman Sajid did his master in Computer Science in 2013 from LUMS. Now he is the part of Foundation University, Islamabad as a lecturer. His areas of interest are Programming, Algorithms etc.

How to cite this paper: Abid Hussain, Yousaf Shad Muhammad, Muhammad Nauman Sajid,"An Efficient Genetic Algorithm for Numerical Function Optimization with Two New Crossover Operators", International Journal of Mathematical Sciences and Computing(IJMSC), Vol.4, No.4, pp.42-55, 2018.DOI: 10.5815/ijmsc.2018.04.04