Numerical Comparison of some Crossover in Genetic Algorithm for Two Dimensional Global Minimization Problem using C++

Muhamad Deni Johansyah¹, Herlina Napitupulu²*, Eddy Djauhari³, Sukono⁴, and Julita Nahar⁵
¹,²,³,⁴,⁵Department of Mathematics, FMIPA, Universitas Padjadjaran, Jatinangor 45363, Sumedang, Jawa-Barat, Indonesia

Corresponding author’s email: *napitupuluherlina@gmail.com

Abstract. In this paper we simulate the comparison of numerical results of the different crossover method in genetics algorithm for solving global minimization problem. In this paper, the objective function is for 2 two variables function. The genetics algorithm is run using C++ program and implemented to several benchmark test functions of global optimization. The performance of each crossover method is analyse according to the numerical results, which show that certain crossover method is better for certain benchmark test function.

1. Introduction

Genetic algorithms (GA) was found by John Holland and developed by David Golberg. GA is numerical optimization algorithms which its idea is from natural selection and natural genetics [3] which is regarded as a heuristic approach. This method is known as direct, parallel, as well as powerful technique in optimization methods. This method is able to optimize continuous or discrete variables, with non-derivative needed, simultaneously searches from a large domain of objective function, deals with a large number of variables, optimizes variables with extremely complex objective function, provides several optimum solutions instead of one and so on [1,4,5]. This method is applicable to various fields of optimization case. This makes GA being an attractive method to be studied [6]. Some literature are discussing the genetics algorithm for global optimization purpose such as: in [7] to achieve the global optimization the authors use rastrigin function as illustration, in [8] shows the implementation of genetics algorithm optimization to control non-linear direct torque control of induction motor drive, and in [9] the research is made to optimize the surface roughness prediction model using Genetic Algorithms (GA) to optimize the objective function.

2. Methods

One of the stochastic methods to solve the minimization problems is genetics algorithm method. Our goal is to solve the minimization problem using Genetic Algorithm method. The general genetics algorithm pseudo code, suggested by Golberg, is given below and its implementations are given in Section 3 for solving minimization problem.
Pseudo code of Genetics Algorithm

Step 1. Initialization
Step 2. Encoding
Step 3. Generate the Initial population
Step 4. Chromosome evaluation (Calculate and Evaluate of fitness value)
Step 5. Calculation of \( p_k \); \( cg = cg + 1 \)
   - Step 5.1. if \( p_k \geq \theta \) or \( cg \geq jg \geq \theta \) then decoding, go to Step 6
   - Step 5.2. else
     - Step 5.2.1. Selection (Individual Selection)
     - Step 5.2.2. Crossover
     - Step 5.2.3. Mutation
     - Step 5.2.4. Create a New Population and go to Step 4
Step 6. Termination

3. Genetic Algorithm in Global Optimization

Let \( \mathbb{R}^n \) the \( n \) –dimensional real Euclidean space, \( x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \), and let \( f: \mathbb{R} \rightarrow \mathbb{R}^n \) be a real-valued function. In this paper the objective is to find the global minimizers of \( f \), that is, the points \( x^* \in \mathbb{R}^n \) such that \( f(x^*) \leq f(x) \) for all \( x \in \mathbb{R}^n \). We restrict the scope only for \( n = 2 \).

To construct the GA program to solve the minimization problem using, we use the following criteria.

- The objective function is minimizing the function \( f \) with
- Fitness that we calculate is for minimization;
- \( N \) = population size. To compare the performance we take several values of \( N \) in this paper
- The total fitness calculated by: \( F = \sum_{k=1}^{N} eval(v_k), k = 1,2,\ldots,N \)
- The formula to calculate the cumulative probability \( q_k \) for each chromosome \( v_k \) is given by \( q_k = \sum_{k=1}^{N} p_k, k = 1,2,\ldots,N \).

The genetics algorithm processes generally separated into following important steps: encoding, selection, crossover, and mutation ([6]). The details explanation of each steps is given in the following..

Pre-step 1: Initialization

The initialization is done as follows, we initialized the number of generation \( jg \), the counter generation \( cg \), the number of population \( pop \), the crossover probability \( pc \), and number of mutation per generation \( mut \), and the threshold \( \theta \).

Pre-step 2: Encoding

The objective is to find \( x = (x_1, x_2) \) which minimize of \( f(x) \) in the interval solution

\[
x_1^L \leq x \leq x_1^U, x_2^L \leq x \leq x_2^U.
\]

In this research we use binary encoding method to represent chromosome and the gen length (denoted by \( m_j \)). We calculate \( m_j \) \((j = 1,2)\) as

\[
2^{m_{j-1}} < (x_j^U - x_j^L) \times 10^k < 2^{m_j} - 1.
\]

To convert the binary to decimal for variable \( x \), we use formula

\[
x_j = x_j^L + \text{decimal}(\text{substring}) \times \frac{(x_j^U - x_j^L)}{2^{m_j-1}}.
\]

For each case problem we have \( m_1 = m_2 \), so that we will have length of chromosome or number of gen \( m = m_1 + m_2 \).

Step 1: Initial Population
A set of $N$ initial population is obtained randomly. We use the $N = 300,600, \text{ and } 800$ random initial population, which are in the binary form of length \textit{bit}. The decimal value of each initial population is calculated by substituting the decimal of each substring to formula (1).

\textbf{Step 2: Chromosome Evaluation}
In our case, the fitness value of chromosome is minimization problem, is the value of objective function $f$. The fitness value of each chromosome is evaluated in this step, in order to see the best value of $f$ (minimum value).

\textbf{Step 3: Calculation of Population Convergence Percentage}
The Population Convergence Percentage ($pk$) defined as percentage of total number of individual with most total number of the same fitness value. It use the formula

$$pk = \frac{n}{pop} \times 100\%$$

where $n$ is the most total number with similar fitness number with $pop$ is the number of individual. The calculation of counter generation is given by formula,

$$cg = cg + 1.$$

\textbf{Step 4: Termination Criteria}
The termination holds when the counter generation $cg$ is reaching the number of generation ($jg = 1000$) or the convergence of population $pk$ reaching the threshold level ($\theta = 90\%$). After population is initialized, the value of $cg = 0$. If there is no such individual with similar fitness number, then $\theta = 10\%$, means that that the genetics algorithm process will continue.

\textbf{Step 5: Chromosome Selection}
To calculate the fitness value of minimization problem, the following formula is used:

$$eval(v_k) = \frac{1}{f(x^g)-f(x)}.$$ 

with $x^g$ as the global minimizer of $f$. The total fitness is obtained by the formula

$$F = \sum_{k=1}^{N} eval(v_k)$$

thus the probability selection $p_i$ for each chromosome $v_k$ is calculated using

$$p_k = \frac{eval(v_k)}{F}.$$

The cumulative probability is calculated by

$$q_k = \sum_{i=1}^{k} p_i.$$

By selecting 10 times the set of 10 chromosome using roulette wheel, the random numbers $r \in [0,1]$ is obtained. We apply the following criteria to addresses the position to replace the selected $k$ – th current individual,

$$if \begin{cases} q_{k-1} < r \leq q_k, & select \ v_k, \\ r < q_1, & select \ v_1. \end{cases}$$

\textbf{Step 6: Crossover}
The crossover will be done to the new population; in this paper the research is to compare the performance of the Crossover method as follows ([6]):

- Single Point Crossover (SPX). SPX is the very simple crossover by generate one random number (the value is the number from 1 to gen length of one chromosome) as the cutting point. The algorithm of SPX is as follows [5]:
  a) Generate one random number (in the interval $[1, gen\ length]$), this number will be the initial position of gen of the chromosome to be crossover
  b) Switch the allele which is noted by parent 1 and parent 2.
Two Point Crossover (TPX). TPX is the crossover with two cutting point, where two random number are generated (the value is the number from 1 to gen length of one chromosome) as cutting points. The algorithm of TPX is as follows [5]:

a) Generate two random number (in the interval [1, gen length]), in which this numbers will be used for as the beginning and ending position respectively of the chromosome to be crossover.

b) Switch the allele which is noted by parent 1 and 2.

Step 7: Mutation
Number of gen in one generation is \( m \times N \). If we want to arrange the mutation probability value equal to \( pm \), then we should have calculate number of mutation \( nm \) as

\[ nm = m \times N \times pm \]

In this simulation, we use \( pm = 0.001 \), then, in the mutation process we generate randomly \( nm = 4 \) number of integer in interval \( [1, pop \times m] \). This 4 number is used as location in which gen will be mutated. The replacement process of allele is 1 and 0, as follows

\[ \text{allele} \begin{cases} 0 \rightarrow 1 \\ 1 \rightarrow 0 \end{cases} \]

After the mutation process then the calculation in one generation is finished. The next process is to repeat the process in step 2 of the algorithm until the termination criteria is satisfied.

Step 8: Decoding
Decoding is the process to recode the gens in a chromosome so that the value is return into the value (before encoded).

4. Numerical Results and Discussion
SPX and TPX crossover methods in genetics algorithm are compared computationally. The programming is using C++ Microsoft Visual Studio 2010, which is applied to following 10 benchmark testing function of global optimization.

We initialized the number of generation \( jg = 1000 \), the counter generation \( cg = -1 \), the number of population \( pop = 300, 600, \) and \( 800 \). The crossover probability \( pc = 0.25 \), number of mutation per generation \( mut = 4 \), and the threshold \( \theta = 90\% \).

The implementation of crossover method on global optimization in this research is as follows. For SPX method, we generate one random number \( r_c \) which its value is from 1 to \( gen \) length that will be pointing the initial gen position to be crossover in both parents, thus the allele between parents are switched. The \( gen \) length = \( m \) of each problem is based on the calculation of \( m_1 + m_2 = m \) as given in step 2. For the TPX method the number of generated random number are two, with value in between 1 to \( gen \) length. Thus this numbers are used as the starting and ending position of chromosome to be crossover, thus the pointed allele are switched between parent 1 and 2.

Problem 1.
Six Hump Back Camel Function has two global minimum solutions, i.e. \( x^* = (0.08984,0.71265) \) and \( x^* = (-0.08984,-0.71265) \) where \( f(x^*) = -1.03163 \).

\[
\begin{align*}
f(x_1, x_2) &= 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 - x_1x_2 - 4x_2^2 + 4x_2^4 \\
&\text{s.t. } -3 \leq x_1, x_2 \leq 3
\end{align*}
\]

Problem 2.
Three Hump Back Camel Function has one global minimum solution, i.e. \( x^* = (0,0) \) with \( f(x^*) = 0 \)

\[
f(x_1, x_2) = 2x_1^2 - 1.05x_1^4 + \frac{1}{6}x_1^6 - x_1x_2 + x_2^2
\]
$$\begin{align*}
s.t. \ -3 \leq x_1, x_2 \leq 3
\end{align*}$$

**Problem 3.**
Shubert Function I has 760 minima with 18 global minimizers where \( f(x^*) = -186.731 \)

\[
f(x_1, x_2) = \left\{ \sum_{j=1}^{5} i \cos((i + 1)x_1 + i) \right\} \left\{ \sum_{j=1}^{5} i \cos((i + 1)x_2 + i) \right\} + 1 \big( x_1 + 1.42513 \big)^2 + (x_2 + 0.80032)^2
\]

s.t. \ -10 \leq x_1, x_2 \leq 10

**Problem 4.**
Shubert Function II has 760 minima with global minimizers \( x^* = (-1.42513, -0.800321) \) where \( f(x^*) = -186.731 \)

\[
f(x_1, x_2) = \left\{ \sum_{j=1}^{5} i \cos((i + 1)x_1 + i) \right\} \left\{ \sum_{j=1}^{5} i \cos((i + 1)x_2 + i) \right\} + 1 \big( x_1 + 1.42513 \big)^2 + (x_2 + 0.80032)^2
\]

s.t. \ -10 \leq x_1, x_2 \leq 10

**Problem 5.**
Shubert Function III has 760 minima with global minimizers \( x^* = (-1.42513, -0.800321) \) where \( f(x^*) = -186.731 \)

\[
f(x_1, x_2) = \left\{ \sum_{j=1}^{5} i \cos((i + 1)x_1 + i) \right\} \left\{ \sum_{j=1}^{5} i \cos((i + 1)x_2 + i) \right\} + (x_1 + 1.42513)^2 + (x_2 + 0.80032)^2
\]

s.t. \ -10 \leq x_1, x_2 \leq 10

**Problem 6.**
Goldstein-Price Function has 4 minimizers, and 1 among them is the global one with the value \( f(0, -1) = 3 \). Formula of Goldstein-Price function and its domain where minimizers exist are as follows

\[
f(x_1, x_2) = \left( 1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \right) \times \left( 30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2) \right)
\]

s.t. \ -3 \leq x_1, x_2 \leq 3

**Problem 7.**
Rastrigin function has one global minimizer. It has about 50 minimizers. This typical test of non-linear multimodal function is rather difficult due to its large search space and large number of local minimizers. Its global minimum is \( f(0, 0) = -2 \).

\[
f(x_1, x_2) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_1)
\]

s.t. \ -2 \leq x_1, x_2 \leq 2

**Problem 8.**
Treccani function has two symmetric global minimizer with the value \( f(-2, 0) = f(0, 0) = 0 \). The formula and its domain are as follows

\[
f(x_1, x_2) = x_1^4 + 4x_1^2 + 4x_2^2 + x_2^2
\]

s.t. \ -2 \leq x_1, x_2 \leq 2

**Problem 9.**
Two Dimensional Sin Square function is a function which exhibits many local minimizers and has one global minimizer with minimum value \( f(1, 1) = 0 \). The formula and domain of this function is as follows,

\[
f(x_1, x_2) = \frac{\pi}{2} \left( 10 \sin^2(\pi x_1) + [(x_1 - 1)^2(1 + 10 \sin^2(\pi x_2))] + (x_2 - 1)^2 \right)
\]

s.t. \ -10 \leq x_1, x_2 \leq 10

**Problem 10.**
Himmelblau function has four global minimizer \((3, 2), (3.58443, -1.84812), (-2.80512, 3.13131)\), and \((-3.77931, -3.28319)\) where \( f(x^*) = 0 \). The formula and its domain is given as follows

\[
f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2
\]

s.t. \ -6 \leq x_1, x_2 \leq 6
We assume that the algorithm is a success in obtained global minimizer if the termination point which converge to $x^k$ satisfying $|f(x^k) - f(x^*)| \leq 10^{-2}$. Each test problem are done about 20 times of trials. The numerical results are given in the Table 1 with the following symbols explanation.

- $N_{pop}$ = number of initial points/ initial populations
- $Time$ = average elapsed time until termination of program
- $Iter$ = average number of iteration until program terminated
- $ns$ = number of success in reaching the global minimizer
- $nf$ = number of failed in reaching the global minimizer

### Table 1. Numerical results of Problem 1-10 for 20 time trials

| No. | $N_{pop}$ | SPX | TPX |
|-----|-----------|-----|-----|
|     | $Time$ (second) | $Iter$ | $ns$ | $nf$ | $Time$ (second) | $Iter$ | $ns$ | $nf$ |
| 1   | 300       | 4.04495 | 270.4 | 13 | 7 | 2.8349 | 298.65 | 14 | 6 |
|     | 600       | 6.66255 | 344.85 | 18 | 2 | 4.2836 | 164.85 | 17 | 3 |
| 2   | 300       | 2.99485 | 328.85 | 11 | 19 | 2.64825 | 282.45 | 14 | 6 |
|     | 600       | 4.4118 | 197.25 | 19 | 1 | 6.5925 | 338.25 | 14 | 6 |
| 3   | 300       | 3.353 | 375.05 | 0 | 20 | 3.2943 | 374.7 | 0 | 20 |
|     | 600       | 11.7184 | 584.4 | 19 | 19 | 8.4649 | 341.85 | 3 | 17 |
| 4   | 300       | 6.4954 | 833 | 0 | 20 | 6.3335 | 787.75 | 0 | 20 |
|     | 600       | 11.9507 | 664.1 | 19 | 19 | 10.9877 | 618.7 | 0 | 20 |
|     | 800       | 21.03695 | 845.6 | 0 | 20 | 19.01645 | 698.6 | 1 | 19 |
| 5   | 300       | 5.75945 | 694.35 | 0 | 20 | 6.2136 | 745.55 | 0 | 20 |
|     | 600       | 19.0516 | 882.25 | 0 | 20 | 12.6888 | 701.2 | 1 | 19 |
|     | 800       | 21.1309 | 830.15 | 19 | 19 | 15.94935 | 600.7 | 3 | 17 |
| 6   | 300       | 4.4139 | 540.1 | 0 | 20 | 3.7866 | 442.8 | 3 | 17 |
|     | 600       | 11.58655 | 704.95 | 19 | 19 | 7.08345 | 365.6 | 6 | 14 |
|     | 800       | 9.72575 | 390.65 | 5 | 15 | 11.70215 | 439.1 | 6 | 14 |
| 7   | 300       | 3.0973 | 349.95 | 7 | 13 | 2.87425 | 300.25 | 9 | 11 |
|     | 600       | 6.5634 | 291.45 | 11 | 9 | 5.8313 | 256.35 | 13 | 7 |
|     | 800       | 7.6415 | 209.85 | 12 | 8 | 13.2184 | 448.8 | 11 | 9 |
| 8   | 300       | 2.73785 | 284.7 | 16 | 4 | 2.65685 | 267.95 | 16 | 4 |
|     | 600       | 6.27285 | 284.1 | 19 | 1 | 6.7896 | 313.2 | 19 | 1 |
|     | 800       | 7.39475 | 198.05 | 20 | 0 | 9.5255 | 340.9 | 20 | 0 |
| 9   | 300       | 5.5345 | 661.25 | 19 | 19 | 4.40175 | 505.3 | 3 | 17 |
|     | 600       | 13.1879 | 724.15 | 4 | 16 | 9.9385 | 573.25 | 7 | 13 |
|     | 800       | 18.31975 | 708.5 | 6 | 14 | 19.7462 | 764.4 | 7 | 13 |
| 10  | 300       | 4.12935 | 484.3 | 2 | 18 | 4.0643 | 462.5 | 3 | 17 |
|     | 600       | 9.3262 | 444.4 | 6 | 14 | 9.1094 | 418.95 | 7 | 13 |
|     | 800       | 11.8446 | 405.8 | 8 | 12 | 12.44535 | 405.15 | 7 | 13 |

According to the numerical results in Table 1, by comparing between SPX and TPX obtained the following summary. Based on the average time elapsed, Problem 1, 2, 4, 6, 7, 8, 9 and 10 for the 20 times trials the average of the least elapsed time is the procedure using TPX with $N_{pop}=300$. But for Problem 3 and 5, the average of the least elapsed time is the procedure using TPX with $N_{pop}=600$, since for problem 3 and 5, weather the SPX or TPX with $N_{pop}=300$ failed in reaching any global minimum.

Based on the average number of iteration, Problem 1, 3, 4, 6 have the least number of iteration on TPX with $N_{pop}=600$. Thus, for Problem 2, 5, 10 on TPX with $N_{pop}=800$, Problem 7 and 8 on SPX with $N_{pop}=800$, and Problem 9 on TPX with $N_{pop}=300$.

Based on the Average Number of success/failed in obtained global minimum, Problem 1 on SPX with $N_{pop}=600$ and TPX with $N_{pop}=800$, Problem 2 on SPX with $N_{pop}=600$. Problem 3 and 7 on TPX with $N_{pop}=600$. Problem 4 on SPX with $N_{pop}=600$ and TPX with $N_{pop}=800$. Problem 5 on TPX with
Npop=800. Problem 6 on TPX with Npop=600 and 800. Problem 8 on SPX and TPX respectively with Npop=800. Problem 9 on TPX with Npop=600 and 800. Problem 10 on SPX with Npop=800.

Generally, by comparing between SPX and TPX based on: the least average elapsed time until termination, the number of success, the least number of iteration for the given benchmark test function with given condition then the TPX method is better than SPX method.

5. Conclusion

We can conclude that for most of the benchmark test functions genetics algorithm with SPX and TPX crossover method is success in reaching the global minimum solution. Generally, by comparing the average elapsed time, the least for all problems is reached by TPX crossover method. According to the number of success, TPX is more than SPX method. For the whole benchmark test function, the least iteration is achieved in by TPX crossover method. It is clear that generally for the given conditions and the given benchmark test function, the TPX method is better than the SPX method.

References

[1] Haupt R L and Haupt S E 2004 Practical Genetic Algorithms (2nd ed.) (New Jersey: A John Wiley & Sons, Inc., Publication)
[2] Bimlendu S, Sujata D, Abhinav M, Vinay K, Prasanna K C 2016 A Review over Genetic Algorithm and Application of Wireless Network System Procedia Computer Science 78 pp 431-438
[3] Wagner F S, Claudio M N A, Pereira, Pius P M S, Roberto S 2002 Genetic algorithms applied to turbine extraction optimization of a pressurized-water reactor. Applied Energy 773 pp 217-222
[4] Kramer O 2017 Studies in Computational Intelligence Volume 679: Genetic Algorithm Essentials (Oldenburg: Springer)
[5] Weise T 2009 Global Optimization Algorithms Theory and Application (http://www.it-weise.de/, downloaded 10/10/2017)
[6] Adriana F and Fadly R L 2015 Algoritma Genetika dan Penerapannya (Yogyakarta: Teknosain)
[7] Bajpai P and Kumar M 2010 Genetic Algorithm- An Approach to Solve Global Optimization Problem Indian Journal of Computer Science and Engineering 1 (3) pp199-206
[8] Malhotra R, Singh N, and Singh Y 2011 Genetic Algorithms: Concepts, Design for Optimization of Process Controllers 4 (2) Computer and Information Science
[9] Suresh P V S, Rao P V, Deshmukh SG 2002 A genetic algorithm approach for optimization of surface roughness prediction model. International Journal of Machine Tools and Manufacture 42 6 pp 675-680

Acknowledgment

The authors thank to the Universitas Padjadjaran for the grant research funding through the scheme of Research Fundamental UNPAD (RFU) with contract number 2335/UN6.D/KS/2018