Genetic Algorithm Modification Analysis Of Mutation Operators In Max One Problem

Ummul Khair
Informatic Engineering Departement
Engineering and Computer Faculty
University of Harapan Medan
Medan, Indonesia
ummul.kh@gmail.com

Adidtya Perdana
Informatic Engineering Departement
Engineering and Computer Faculty
University of Harapan Medan
Medan, Indonesia
adid.dana@gmail.com

Arief Budiman
Informatic Engineering Departement
Engineering and Computer Faculty
University of Harapan Medan
Medan, Indonesia
ariefdiman13@gmail.com

Yuyun Dwi Lestari
Informatic Engineering Departement
Engineering and Computer Faculty
University of Harapan Medan
Medan, Indonesia
yuyun.dl@gmail.com

Dody Hidayat
Informatic Engineering Departement
Engineering and Computer Faculty
University of Harapan Medan
Medan, Indonesia
hidayatdody91@gmail.com

***Abstract*— Max One Problem is the simplest problem that only performs the calculation of the maximum value from a number of binary strings. Usually, Max One Problem is used to represent an algorithm i.e. Genetic Algorithm. This paper will discuss how genetic algorithms work to solve the problems of Max One. However, the genetic algorithm will be modified in the algebraic process stages in the mutation operator. As in modified genetic algorithms without the use of crossover processes, and modified of genetic algorithms without the use of selection and crossover processes. Later, the result will be compared with conventional genetic algorithms. The purpose of this comparison is to see whether the modified genetic algorithm process can produce different results than conventional genetic algorithms and can provide a variety of methods to solve a problem. In this paper, the variation of modified method of the genetic algorithm is expected to solve problems other than the problem of Max One.**

Keywords*— *Genetic Algorithm, Genetic Algorithm Modification, Crossover, Selection, Max One Problem

I. INTRODUCTION

One of the most common metaheuristic search methods is the Genetic Algorithm. Genetic algorithms are adopted from the branches of biological science i.e. genetics and natural selection. This algorithm has several important operators that are used in solving the problem to obtain more optimal results. These operators are selection operators, crossover operators or crossovers, and mutation operators. Each operator has a very important role that can affecting the outcome of the process[1].

One of the operator that has an important role that affects the outcome of the process is the mutation operator, the operator serves to recover the lost genetic material by replacing or shifting a randomly selected value. This operator uses a parameter i.e. the probability of mutation (pm). The mutation probability is used to determine how many genes in each chromosome will be mutated in one population in the process of solving the problem. By changing the value, smaller or greater between 0 to 1, may affect the results obtained[2][3].

There are many variations of this method, in addition to conventional genetic algorithms that commonly used. Such as hybrid genetic algorithms that combine genetic algorithms with probabilistic methods, parallel genetic algorithms, messy genetic algorithms, fuzzy genetic algorithms and etc. [2][4][5][6]. The purpose of genetic algorithm variation is to obtain the more optimal and varied process results.

Beside the variation of genetic algorithm methods that already existed, genetic algorithms can be modified as in the adoption of Hardy-Weinberg's Equilibrium Law by eliminating the process of selection operators and mutation operators, and crossover all chromosomes in populations where the processed data must be large[7]. This modification aims to get more varied results, whether the results obtained are better or worse than the conventional genetic algorithm.

To test the modification of this genetic algorithm requires a problem. One of the easiest problems which can be used to implement genetic algorithm is Max One Problem. Max One Problem is a simple problem that only searching for the maximum value in a binary string[8]. This problem can also be defined as a class on the optimization problem where the objective is to find a solution that meets all criteria with a set of maximum variable values of 1[9].

In this research, the modification of genetic algorithm will be tested, i.e. modification of the stages of genetic algorithm process at the mutation operator which applied to solve Max One problem and compare it with the result obtained from conventional genetic algorithm usage.

II. Literature Review

A. Genetic Algorithm

Genetic Algorithm was first discovered by John Holland in 1975 at the University of Michigan through his book "Adaptation in Natural and Artificial System" which explains how to apply the principles that exist in natural evolution into optimization problems and then popularized by one his students, David Goldberg through his research [1][10]. Genetic algorithm usually used to solve the optimization problem of a nonlinear string search [11]. In the genetic algorithm there are several stages of the process [7] [12]:

1. Data Input; data is defined before it is processed.
2. Population Initialization; data that has been inputted at initialization and raised as an initial population as a set of random individuals.
3. Calculating Fitness Value; after the initial population is generated randomly then each individual is calculated its fitness value.
4. Selection; the process undertaken to select the number of individuals that have the best fitness value using certain selection methods to create the parent for the next process.
5. Crossover; the process undertaken to select the string position of the selection result and replace it randomly on the parent that will generate the new individual using certain crossover methods.
6. Mutation; processes undertaken to restore the lost or damaged genetic material using certain randomly to choose mutation methods.

B. Modification of Genetic Algorithm

Modifying or combining an algorithm is a way to get the more optimal results from the issues to be solved. The result should be compared with the results from the unmodified algorithm. Some literature has been discussed about the modification or combination of genetic algorithms, either modify the operator process or combining genetic algorithms with other optimization methods.

One of the modifications made to the genetic algorithm is to apply Hardy-Weinberg Equilibrium Law from the branch of biological science [7]. In theory, the law states that a population of stable conditions, both frequency and gene ratios, will remain constant from generation to generation in a sexually-breeding population. On condition [7][13]:

1. Genes have the same viability and fertility,
2. Marriage occurs randomly,
3. No mutation of genes,
4. There is no migration and natural selection,
5. The number of individuals of a population must always be large.

In addition to modifications by applying the Hardy-Weinberg Equilibrium Law, modifications may also be made to processes or operators of genetic algorithms such as modifying the selection and mutation process. So the results given from these modifications become more varied and can be used as reference to further research.

C. Mutation Operators

Mutation is the next process after a crossover process. The goal is to prevent the process of being trapped in local optima and to recover or restore lost or damaged genetic material and randomly disturbing genetic information [1][12]. There are different types of methods for mutation operators. Each method can be used only to solve certain problems. In the binary problem representation, the method that can be used is a method of reversing gene values with a small probability [1]. Here are some of the mutation methods [1]:

1. Flipping Mutation; is a mutation method that reverses the value of 1 to 0 and 0 to 1 based on the resulting mutation chromosome.
2. Interchanging Mutation; is a mutation method that exchanges two randomly selected genes in the same individual.
3. Reversing Mutation; is a mutation method that selects the random gene position in the selected individual and swaps with the gene next to it. This process is done just like using a mirror.

D. Max One Problem

Max One Problem is the simplest problem to represent genetic algorithm. This problem focuses on calculating the maximum value of a number of binary strings [8] [14]. To form a number of binary strings can be done manually as flipping coins as much as the desired binary amount. Described as a side of a coin as a value of 1 and a coin value of money as a value of 0. If computerized, the formation of a binary string would be very easy to do by generating a number of random numbers 0 and 1 as much as needed. To calculate the maximum value of a given binary string can use the following function [14]:

$$F(x) = \sum_{i=1}^{N} x_i \quad (1)$$

Where N is the number of strings to be raised and x_i is the binary array. The equation only adds a value of 1 to a set of generated binaries.

III. Methodology

In this research, the applied methodology is to test the modification of genetic algorithm and conventional genetic algorithm. There are two modifications made in this research, including:

1. Genetic algorithms by eliminating the crossover process, a process that is done only selection and mutation.

2. Genetic algorithm by eliminating the selection process and crossover, the process is done only mutation.

The testing phase performed 10 times. Each test is based on the probability of a mutation (pm) with a value from 0.1 to 1.0 (pm = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]). In each test there were 5 experiments for each method used. So the total experiment is 150 times in solving Max One problem.

A. Data Used

The data used in this research is a number of random binary strings of 200 pieces. The binary string to be used is divided into 10 new strings and each string consists of 20-digit binaries. The following is a generated binary string:

```
S1 = [0 1 0 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 1 0]
S2 = [1 0 1 0 1 1 0 1 0 1 0 0 0 1 0 1 1 1 0 1]
S3 = [1 1 1 0 1 0 0 1 0 0 1 0 1 0 0 1 0 1 0 0]
S4 = [0 1 0 1 0 0 1 0 0 1 1 1 0 1 0 1 0 0 1 0]
S5 = [0 1 0 0 1 0 1 0 0 1 0 1 0 1 0 0 1 0 0 0]
S6 = [0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 1 1]
S7 = [0 1 0 0 1 0 1 0 0 1 1 1 1 1 0 1 0 1 0 0]
S8 = [0 1 0 0 1 1 0 1 1 0 1 0 1 0 0 1 1 1 1 0]
S9 = [0 1 0 0 1 0 1 0 0 1 0 1 0 0 0 1 1 1 1 0]
S10= [0 1 0 0 1 1 0 1 0 0 0 0 1 0 1 0 0 1 0 1]
```

The data is generated using computer programs and is used in testing to Conventional Genetic Algorithms and Modification of Genetic Algorithms.

B. Parameters of Conventional Genetic Algorithms

The parameters used in the testing process of Conventional Genetic Algorithms are as follows:

1. Number of generation parameters (G): 500
2. Population size parameters (L): 20
3. Individual size parameters (n): 10
4. Crossover Probability Parameters (pc): 0.5
5. Selection Method: Roulette Wheel
6. Crossover Method: One-Point Crossover
7. Mutation Method: Flipping Mutation

C. Parameters of Non-Crossover Genetic Algorithms

The parameters used in the testing process of Genetic Algorithm without Crossover process are as follows:

1. Number of generation parameters (G): 500
2. Population size parameters (L): 20
3. Individual size parameters (n): 10
4. Selection Method: Roulette Wheel
5. Mutation Method: Flipping Mutation

D. Parameters of Genetic Algorithms Without Selection and Crossover

The parameters used in the testing process of Genetic Algorithm without selection and crossover process are as follows:

1. Number of generation parameters (G): 500
2. Population size parameters (L): 20
3. Individual size parameters (n): 10
4. Mutation Method: Flipping Mutation

IV. TESTING

A. First Test

In the first test, it is performed with probability of mutation is 0.1 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA -x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE I. FIRST TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	159	150	124
2	151	162	120
3	153	158	120
4	150	152	124
5	152	156	122
Avg	**153**	**155.6**	**122**

B. Second Test

In the second test, it is performed with probability of mutation is 0.2 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE II. SECOND TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	131	140	124
2	134	140	124
3	133	136	120
4	132	134	120
5	136	140	120
Avg	**133.2**	**138**	**121.6**

C. Third Test

In the third test, it is performed with probability of mutation is 0.3 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE III. THIRD TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	133	132	120
2	128	130	122
3	128	128	124
4	127	130	124
5	128	134	122
Avg	**128.8**	**130.8**	**122.4**

D. Fourth Test

In the fourth test, it is performed with probability of mutation is 0.4 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE IV. FOURTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	120	122	126
2	124	122	122
3	127	124	122
4	121	124	120
5	120	125	126
Avg	**122.4**	**123.4**	**123.2**

E. Fifth Test

In the fifth test, it is performed with probability of mutation is 0.5 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE V. FIFTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	122	120	124
2	119	122	124
3	122	122	120
4	120	120	120
5	122	122	126
Avg	**121**	**121.2**	**122.8**

F. Sixth Test

In the sixth test, it is performed with probability of mutation is 0.6 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE VI. SIXTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	119	124	122
2	119	122	122
3	118	120	124
4	125	121	122
5	121	122	122
Avg	**120.4**	**121.8**	**122.4**

G. Seventh Test

In the seventh test, it is performed with probability of mutation is 0.7 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE VII. SEVENTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	123	122	120
2	116	120	124
3	118	118	124
4	122	122	120
5	117	126	122
Avg	**119.2**	**121.6**	**122**

H. Eighth Test

In the eighth test, it is performed with probability of mutation is 0.8 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE VIII. EIGHTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	129	126	126
2	117	130	120
3	115	128	124
4	121	118	122
5	119	122	120
Avg	**120.2**	**124.8**	**122.4**

I. Ninth Test

In the ninth test, it is performed with probability of mutation is 0.9 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE XI. NINTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA -*x***	**GA -*s* -*x***
1	126	132	126
2	119	136	120
3	122	128	120
4	126	130	120
5	121	126	128
Avg	**122.8**	**130.4**	**122.8**

J. *Tenth Test*

In the tenth test, it is performed with probability of mutation is 1.0 to Conventional Genetic Algorithm (abbreviated GA-Konv), Genetic Algorithm without Crossover (GA-x), and Genetic Algorithm without Selection and Crossover (GA -s-x). The test results presented in following table.

TABLE X. TENTH TESTING RESULT

Testing	Type of GA		
	GA-Konv	**GA *-x***	**GA *-s -x***
1	130	108	110
2	117	106	110
3	116	109	110
4	120	109	110
5	124	110	110
Avg	**121.4**	**108.4**	**110**

K. *Testing Result*

From the results of all tests made a chart to get the conclusion of the completed testing. This the following chart:

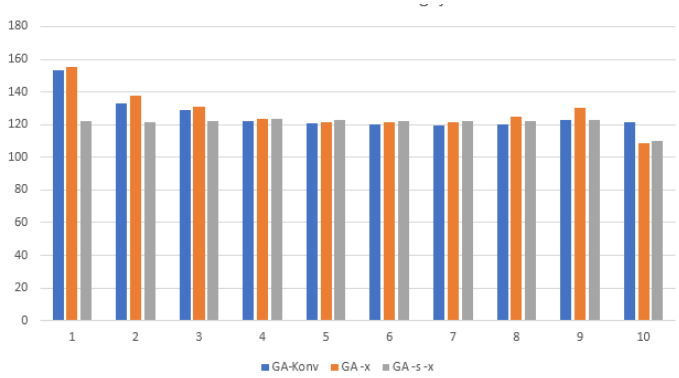

Fig. 1. Chart of All Testing Result

And the following is a chart of test results for each method used.

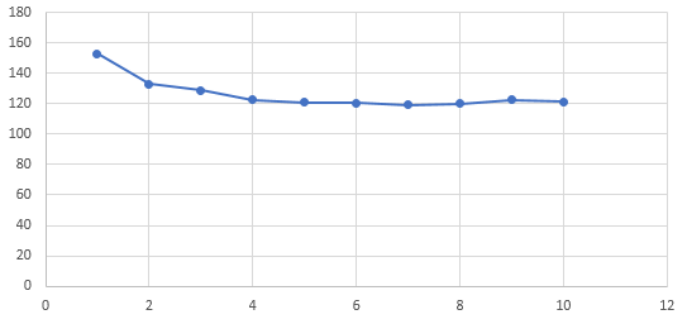

Fig. 2. Chart of Average Testing for Conventional Genetic Algorithm

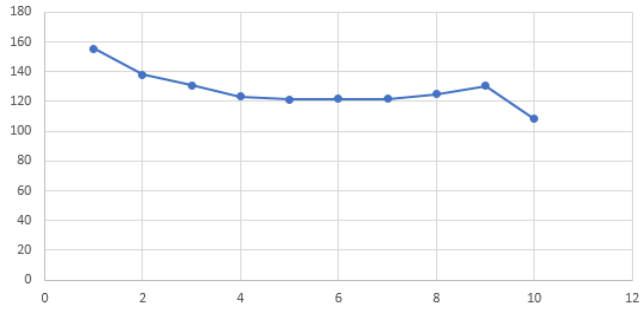

Fig. 3. Chart of Average Testing for Genetic Algorithm without Crossover

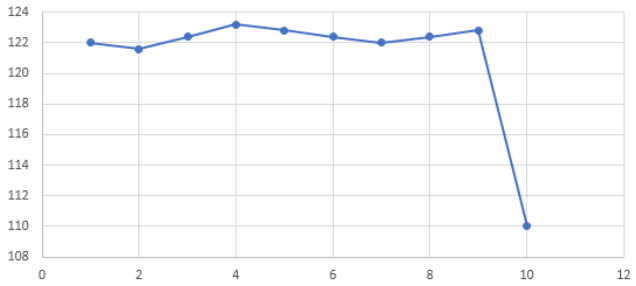

Fig. 4. Chart of Average Testing for Genetic Algorithm without Selection and Crossover

V. DISCUSSION

From the results can be seen that for Conventional Genetic Algorithm and Genetic Algorithm without Crossover, the smaller the probability of mutation the better the results displayed (give maximum result). But for Genetic Algorithms without Selection and Crossover the results shown do not provide a significant difference value of each experiment.

In the first test, genetic algorithms without crossover gave better results than the other two methods used. This is due to the small mutation probability value and the absence of a crossover process so that the genetic material can provide more optimal results.

In the second test, genetic algorithms without crossover still provide more optimal results compared to the other two methods used. For the third and fourth experiments, the results shown do not make a significant difference but the Genetic Algorithm without crossover still gives better results than the other two methods.

In the fifth, sixth and seventh tests, the genetic algorithm without selection and crossover was able to provide better results than the two methods used in the test. Because of the probability of mutations that are in the medium value, so that genetic material can be restored properly and give good results as well.

In the eighth and ninth tests, the genetic algorithm without crossover gives more optimum results than the other two methods. While in the tenth test, conventional genetic algorithms provide better results than the other two methods. In the tenth test, the genetic algorithm without selection and crossover gave constant results from the first to the fifth

experiments. This is caused by mutated genetic material repeatedly because the mutated gene is 100% (all genes present in the population are mutated) so that the maximum value generated from the process will not change.

VI. Conclution

From the results of the analysis, the above test and discussion can be concluded that from the results of the tests conducted on the Max One problem, by comparing the conventional genetic algorithm, genetic algorithm without crossover, and genetic algorithm without selection and crossover there are significant differences to the three comparison methods. Applying modifications to genetic algorithms can provide a variety of methods that can be used to solve problems.

Of the three methods used in the tests performed, modification of genetic algorithm method without crossover is able to give much better results than the other two methods. However, it will give a poor result if mutation probability is 1. This happens because the entire genetic material in the mutation so that the binary string value will be repeated with the same value.

References

[1] S. N. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms*. Berlin: Springer, 2008.

[2] J. Huang and G. A. Süer, "Rule-based Fuzzy Genetic Algorithm in Multi-Objective Job Shop Scheduling," in *Proceedings of the 2009 Industrial Engineering Research Conference*, 2009, pp. 2109–2115.

[3] A. Perdana, "Analisis Perbandingan Metode Genetic Algorithm dan Particle Swarm Optimization dalam Menilai Tingkat Optimasi Hasil Pada Bin Packing Problem Satu Dimensi Adidtya Perdana," in *Prosiding SNASTIKOM 2017*, 2017, pp. 1–6.

[4] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby, "Hybrid Genetic Algorithms : A Review," *Eng. Lett.*, vol. 11, no. August, pp. 124–137, 2006.

[5] T. Vidal, T. G. Crainic, M. Gendreau, N. Lahrichi, and W. Rei, "A Hybrid Genetic Algorithm for Multidepot and Periodic Vehicle Routing Problems," *Oper. Res.*, vol. 60, no. 3, pp. 611–624, 2012.

[6] E. Kasturi and S. L. Narayanan, "A Novel Approach to Hybrid Genetic Algorithms to Solve Symmetric TSP," *Int. J. Adv. Res. Comput. Sci. Manag. Stud.*, vol. 2, no. 2, pp. 2321–7782, 2014.

[7] A. Perdana, "Analisis Performansi Pada Penerapan Hukum Ketetapan Hardy-Weinberg Dalam Algoritma Genetika," Universitas Sumatera Utara, 2015.

[8] A. Saad, E. Avineri, K. Dahal, M. Sarfraz, and R. Roy, *Soft Computing in Industrial Applications: Recent and Emerging Methods and Techniques*. Berlin Heidelberg: Springer, 2007.

[9] S. Khanna, M. Sudan, L. Trevisan, and D. P. Williamson, "The approximability of constraint satisfaction problems," *SIAM J. Comput.*, vol. 30, no. 6, pp. 1863–1920, 2000.

[10] M. Melanie, *An introduction to genetic algorithms*. Massachusetts: MIT Press, 1999.

[11] M. Gen and R. Cheng, *Genetic Algorithms {&} Engineering Design*, vol. 3029. 1997.

[12] M. Negnevitsky, *Artificial intelligence : A Guide to Intelligent Systems*, 2nd ed. Harlow: Pearson Education Limited, 2005.

[13] A. Perdana, "Analisis Komparasi Genetic Algorithm dan Firefly Algorithm pada Permasalahan Bin Packing Problem," *SinkrOn*, vol. 1, no. April, pp. 1–6, 2017.

[14] J. D. Schaffer and L. J. Eshelman, "On Crossover as an Evolutionarily Viable Strategy," in *Proceedings of the 4th International Conference on Genetic Algorithms*, 1991, no. November 2014, pp. 61–68.