

A dynamic approach for parameter tuning in genetic algorithm using crossover and mutation ratios

Mrinmoyee Chattoraj, Udaya Rani Vinayakamurthy

School of Computing and Information Technology, REVA University, Bengaluru, India

Article Info

Article history:

Received Feb 25, 2022

Revised Jun 20, 2022

Accepted Jul 14, 2022

Keywords:

Crossover ratio
Genetic algorithm
Mutation ratio
Parameter control
Parameter selection

ABSTRACT

Genetic algorithm uses the natural selection process for any search process. It is an optimization process where integration among different vital parameters like crossover and mutation plays a major role. The parameters have an impact on the algorithm by their probabilities. In this paper we would review the different strategies used for the selection of crossover and mutation ratios and suggest a dynamic approach for modifying the ratios during runtime. We start with a mutation ratio 0% and crossover ratio 100% where the mutation ratio slowly increases and the crossover ratio decreases (MICD). The final mutation ratio will be 0% and crossover ratio will be 100% at the end of the search process. We also do the reverse process of considering the mutation ratio to be maximum and crossover ratio to be minimum and slowly decrease the mutation ratio and increase the crossover ratio (MDCI). We compare the proposed method with two pre-existing parameter tuning methods and found that this dynamic approach of incrementing the mutation and decrementing the crossover value was more effective when the size of the population was large.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Mrinmoyee Chattoraj

School of Computing and Information Technology, REVA University Kattigenahalli

Yelahanka, Bengaluru, India

Email: mrinmoyee2005@gmail.com

1. INTRODUCTION

Genetic algorithms are heuristics search algorithms which uses the technique of natural selection. It was developed by Holland in the year 1975 to solve optimization problems using evolutionary concepts of genetics [1]. Genetic algorithm is a nonlinear process which does not use any mathematical formula in order to reach the optimal solution but is an important tool to find out the optimal solution in the complex search spaces [2], [3]. Genetic algorithms are widely used by researchers in various fields such as computer network [4], image processing [5], machine learning [6]. In a traditional genetic algorithm, the process starts with a selection operator which chooses a set of individuals based on their fitness [7]. The offspring are produced from these individuals using the crossover and mutation operator and this process continues until a termination condition is reached [8]. The efficiency of the genetic algorithm is controlled basically by the size of the population, crossover and the mutation operator [9]. According to researchers it has been observed that the crossover and mutation operators plays a major role in increasing the performance of the genetic algorithms [10]. The selection and crossover operators help the genetic algorithm to converge to better solutions whereas the mutation operator helps to give the global optima by skipping the local search [11].

The values of these crossover and mutation parameter have an impact on the performance of the genetic algorithm i.e., crossover probability as 50% gives a different result when the crossover probability is changed to 100% [12], [13]. Similarly, the mutation probability also has an effect on the performance of

genetic algorithm [14]. Thus, we can see that it is very important to controlling the parameters in genetic algorithm by keeping a balance between them. There are two ways of setting the parameters which is shown in Figure 1. The first technique is a common approach to experiment on different values and finally fix the values of the parameters which gives the best result [15]. The second technique is to start the execution of the program with a value and slowly change the values of crossover and mutation ratios during runtime. The different techniques to change the values during runtime are deterministic, adaptive and self adaptive parameter control. In case of deterministic parameter control we use some strategy to change the value of the parameter. In case of adaptive parameter control we change the values based on the feedback and in case of self-adaptive parameter control the parameter values in the individuals goes through crossover and mutation. These parameter settings are used by researchers to find an optimal solution. Our proposed method is a type of deterministic parameter control technique.

Section 2 reviews the various techniques used in selecting the crossover and mutation ratios. The proposed dynamic approach to linearly change the ratios during the search process is in section 3. Section 4 consists of the results and discussions and finally the paper is concluded in section 5.

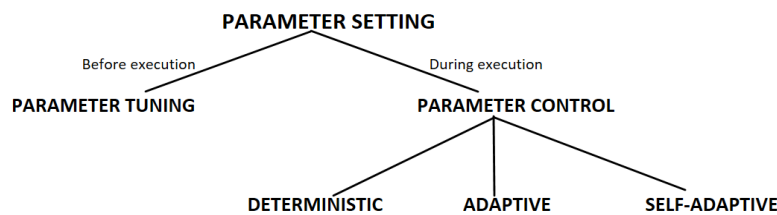


Figure 1. Parameter setting taxonomy

2. RELATED WORKS

John Holland in 1975 gave the mathematical formulation of GA [1] and after that many research work has been going on in this field and it has been proved that the crossover and mutation rates has a major role in GA [16]. According to Jong [17] the optimal range of the size of population is [50-100] and the mutation rate 0.001 and the crossover rate to be 0.6. These parameters were used in the implementations.

Grefenstette [18] used optimal parameter values and showed that if the population size is small such as 20 to 40 range, then the crossover rates will be high and mutation will be low. According to him if the mutation rate is greater than 0.05 then it was not giving the optimal performance. He also found that with a population size of 30, mutation rate 0.01 and crossover rate 0.95 was giving the near optimal results. Muhlenbein and Voosen [19] explained the dynamic behaviour of the rate of crossover and mutation. It was observed that for small population size mutation rate was effective whereas the crossover rate was depended on the size of the population. It was also observed that the combination of both the operators was giving a better result as compared to a single operator. Deb and Agrawal [20] explored the performance of GA and the results of the test discovered that for explaining simple unimodal difficulties, mutation operator played an important role. They also found out that operators such as crossover and mutation have different impact on the population size. They had conducted the experiment with initial crossover rate as 0.9 and population in the range of 2 to 500 with “zero mutation rate”. The next time test was conducted with “zero crossover rate” for the population range of 2 to 500 and mutation 1,0.5 and 0.1. The last test was conducted with mutation rate 1,0.5 and 0.1 whereas two values for crossover 0.9 and 0.0. They analysed that mutation and selection operators performs well with reasonable population size whereas selection operator and crossover work well for large population without the mutation parameter. Hong *et al.* [21] proposed a dynamic genetic algorithm with matches the crossover and mutation rates automatically according to individual evaluation results in the particular generation. Rylander and Stanley [22] had conducted experiments to find out the optimal population size. In their experiments the mutation rate was 0.1, crossover rate 0.5 and different population sizes as 100,300,400 and 600. It was determined that when the population increases the it takes more generations to converge and hence the accuracy also increases. Gomez and Hougen [23] planned a new method to find a basic population for both small as well as large population. Alfeilat *et al.* [24] proposed a dynamic technic to calculate the mutation and crossover rates using Euclidean distance formula between two chromosomes having the highest and lowest fitness. Chiroma *et al.* [25] conducted a survey and found out that the most critical operators are the population size, mutation and crossover rate. He revealed that the crossover and mutation probability are positively associated with each other but the correlation is not significant. Table 1 shows few values of the population size crossover and mutation probabilities used in the previous literatures.

Table 1. Few GA parameters used in previous works

Reference	Population size (PS)	Crossover probability (CP)	Mutation probability (MP)
Huang and Shi [26]	30	0.95	0.01
Huang and Shi [26]	80	0.45	0.01
Renders and Flasse [27]	100	0.9	1
Vavak and Fogarty [28]	100	0.8	0.005
Lizhe <i>et al.</i> [29]	30	0.9	0.1
Laboudi and Chikhi [30]	20	0.9	0.3

3. PROPOSED FRAMEWORK/METHOD

To find the optimal value for the parameters there are two levels which are involved. The first level is to select a suitable algorithm and the second level is to select the different parameters in order to get the highest efficiency. In this case our problem is to find the shortest path using a genetic algorithm which is a parametric method so we need to deal with the second method and find suitable parameters to deal with its efficiency [18]. Using control parameter is an important factor for the performance of GA in order to get optimal or approximate solutions. Some researchers recommend low mutation rate of 0.1 to 1 with high crossover rates of 80 to 95% while others suggest high rate of mutation and crossover (50%) with small population size [20]. Deterministic control parameters change the values during runtime without any feedback from the user. This technique of parameter setting finds out the behaviour of the parameter in order to find optimal solutions. In this paper we propose two deterministic parameter control techniques followed by a parameter tuning method. In the first deterministic technique (MICD) the crossover ratios are decreased from 100% to 0% and the mutation ratios are increased linearly from 0% to 100% along with the process time.

$$MR = \frac{GL}{GN} \quad \text{where } G = [1, 2, 3 \dots n] \quad (1)$$

$$M = MR * psize \quad (2)$$

When the algorithm executes then the mutation rate is changed linearly using (1) and (2) according to the number of generations. MR is the mutation rate, GL is the generation level, M is the number of chromosomes and psize is the population size. At the same time the crossover rate is also changed linearly using (3) and (4) where CR is the crossover rate and C is the number of chromosomes used during the crossover process. This method provides different crossover and mutation rates during each level of generation as shown in Table 2. In this case the low mutation rate is increased and the high crossover if decreased.

$$CR = 1 - \frac{GL}{GN} \quad (3)$$

$$C = CR * psize \quad (4)$$

Table 2. Mutation and crossover rates during increasing of low mutation and decreasing high crossover

Generation	MICD	
	Increasing mutation rate	Decreasing crossover rate
1	0	1
100	0.0612	0.935
200	0.0131	0.875
300	0.187	0.815
400	0.25	0.725
500	0.3112	0.689
600	0.375	0.625
700	0.432	0.521
800	0.521	0.432
900	0.625	0.375
1000	0.689	0.3112
1100	0.725	0.25
1200	0.815	0.187
1300	0.875	0.0131
1400	0.935	0.0612
1500	1	0

In this case we observe that in each generation the crossover rate is decreased by a particular ratio until it becomes 0%. Similarly, the mutation rate is also increased in a particular ratio so that it becomes 100% at the end of the execution of the genetic algorithm. From (1) to (4) it is clear that the mutation and crossover are complement of each other. The main aim of changing the crossover and mutation rate during runtime is a dynamic technique to bring diversity in the population rate. Figure 2 shows the relation between the crossover and mutation rate during a particular population value.

In the second deterministic technique (MDCI) the crossover ratios are increased from 0% to 100% and the mutation ratios are decreased linearly from 100% to 0% along with the process time. This technique is opposite to the technique used in MICD. Table 3 provides the different crossover and mutation rates during each generation. Figure 3 show the relation between crossover and mutation rate for each population.

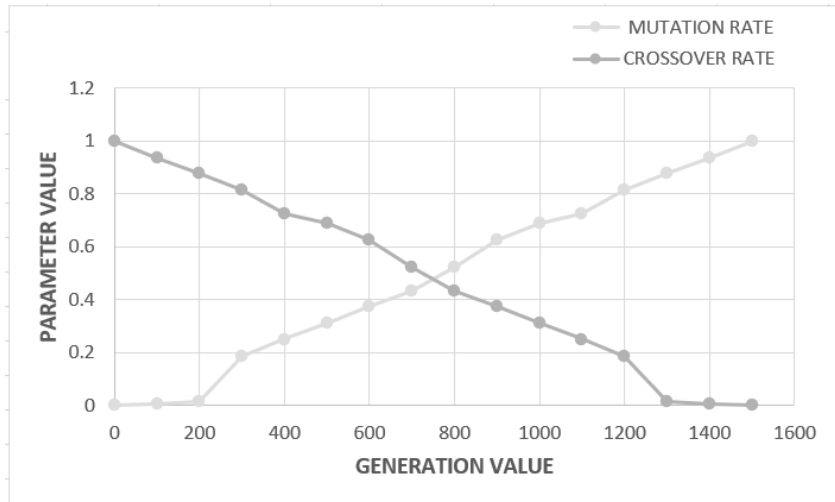


Figure 2. Mutation and crossover rates during MICD

Table 3. Mutation and crossover rates during decreasing of high mutation and increasing low crossover

Generation	MDCI	
	Decreasing Mutation Rate	Increasing Crossover Rate
1	1	0
100	0.935	0.0612
200	0.875	0.0131
300	0.815	0.187
400	0.725	0.25
500	0.689	0.3112
600	0.625	0.375
700	0.521	0.432
800	0.432	0.521
900	0.375	0.625
1000	0.3112	0.689
1100	0.25	0.725
1200	0.187	0.815
1300	0.0131	0.875
1400	0.0612	0.935
1500	0	1

In case of MDCI the mutation rate is calculated using (5) and (6).

$$MR = 1 - \frac{GL}{GN} \quad \text{where } G = [1, 2, 3 \dots n] \tag{5}$$

$$M = MR * psize \tag{6}$$

In this case the mutation rate is changed linearly using (5) and (6) according to the number of generations. MR is the mutation rate, GL is the generation level, M is the number of chromosomes and psize

is the population size. At the same time the crossover rate is also changed linearly using (7) and (8) where CR is the crossover rate and C is the number of chromosomes used during the crossover process.

$$CR = \frac{GL}{GN} \quad (7)$$

$$C = CR * psize \quad (8)$$

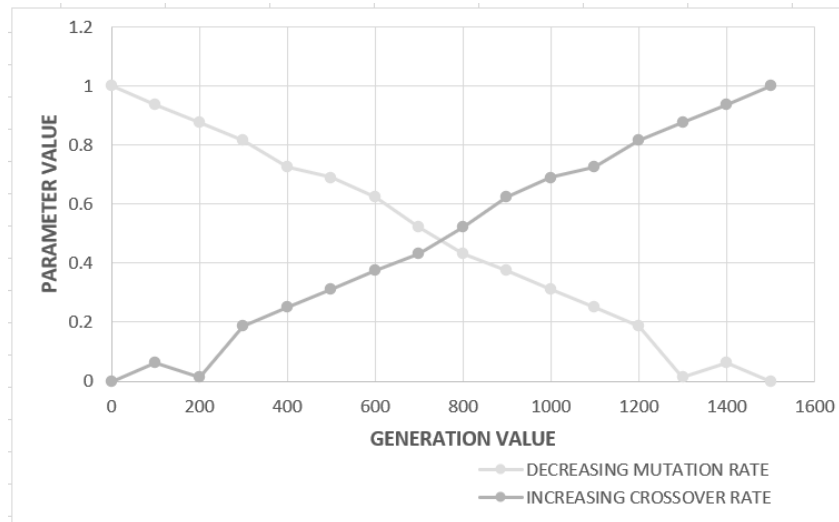


Figure 3. Mutation and crossover rates during MDCI

In case of the parameter tuning methods, we use the mutation rate MR=0.03 and the crossover rate CR=0.09 because most of the literature survey shows high crossover rate in the range 0.5 to 0.9 [30]. Also, we use the fifty-fifty mutation rate and crossover rate (FFMRCCR). Where mutation rate MR=0.50 and crossover rate CR=0.50 [20].

4. RESULTS AND DISCUSSION

Six sets of experiments were conducted in order to find the shortest paths using genetic algorithm and our proposed methods MDCI and MICD was compared with the two parameter tuning techniques MR=0.03 and the crossover rate CR=0.09 and FFMRCCR. Different population sizes were used for the various parameter setting techniques. The size of the population used was:

- Set 1 and 2-small population size of 30 and 60.
- Set 3 and 4-average population size of 100 and 200.
- Set 5 and 6-large population size of 300 and 400.

In this experiment we applied the genetic algorithm to find the shortest path for a NYC data namely A250, B150, C130, D100, E75, F50, G48, P151, Q144 and R780 respectively where the number indicates the number of cities in the route. The initial population uses the roulette wheel selection strategy and the China national petroleum corporation (CNPC) crossover strategy [31]. The termination condition is based on a fixed number of generations. For all our experiments we had set the maximum number of generations as 1500. The result of this GA will be the minimum total distance travelled.

4.1. Experiment Set1(population size=30)

In this experiment we compare the performance of our proposed methods MICD and MDCI with the common parameter tuning methods MR=0.03 and the crossover rate CR=0.09 and fifty-fifty MRCR (FFMRCCR) with a population size of 30. The GA is applied for ten set of routes and it has been observed that the algorithm is giving better convergence rate as the generation number is increasing. The performance of the various techniques is shown in Figure 4 and it has been observed that our proposed methods are giving better performance as compared to the other techniques. The GA results using different parameter settings and their standard deviation is shown in Table 4. Results from the first set of experiment shows that the

performance of MICD is better compared to the other techniques while later the performance of MDCI is better.

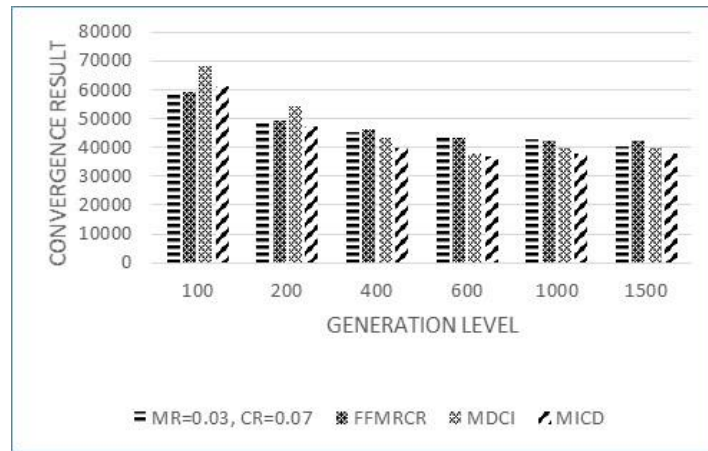


Figure 4. Average convergence of GA for G48 with a population size of 30

Table 4. Average convergence of data and standard deviation for a population size of 30 and after 1,500 generations

Problem	MR=0.03, CR=0.07	STD. Deviation	FFMRCR	STD. Deviation	MDCI	STD. Deviation	MICD	STD. Deviation
A250	78702	19654	68782	2287	68012	1881	69876	1256
B150	200551	14232	153245	14567	161203	12512	149204	9789
C130	501	36	498	29	491	25	456	11
D100	9505	562	9324	379	9023	489	8635	452
E75	151967	12823	138765	8712	146723	14215	138921	8567
F50	35421	2762	34235	3718	35127	2234	31926	1587
G48	40102	3214	42122	3012	39837	3245	37856	1326
P151	105198	6634	94563	7413	93924	7136	90894	2876
Q144	9213	398	8792	289	9187	442	9328	226
R780	10107	593	10012	895	10612	903	10378	723

4.2. Experiment Set2 (population size=60)

In the second experiment also, we observed that in most of the cases the performance of MICD was better as compared to other techniques. The results of GA using different parameter settings is shown in Table 5 with a population size of 60 individuals. For example, if we consider the route G48 we observe that the MICD technique was giving a value of 37894 which was much less and better as compared to the other cases like FFMRCR or MDCI.

Table 5. Average convergence of data for a population size of 60 and after 1,500 generations

Problem	MR=0.03, CR=0.07	FFMRCR	MDCI	MICD
A250	75892	67817	68295	71365
B150	176123	152089	158121	152901
C130	512	487	486	475
D100	9452	9154	9169	9061
E75	156123	145231	139187	134989
F50	39872	32456	32459	31921
G48	39987	41098	38973	37894
P151	102345	94267	95678	93987
Q144	9398	9261	9298	9021
R780	11398	10298	10391	10198

4.3. Experiment Set3 (population size=100)

In the third experiment also, we observed that in most of the cases the performance of MICD was better as compared to other techniques. The results of GA using different parameter settings is shown in

Table 6 with a population size of 100 individuals. Here we observed that like experiment 2 MICD was giving better result compared to the other techniques.

Table 6. Average convergence of data for a population size of 100 and after 1,500 generations

Problem	MR=0.03, CR=0.07	FFMRCR	MDCI	MICD
A250	70167	68817	67695	71065
B150	172123	157089	157121	152901
C130	497	482	478	482
D100	9204	8854	8869	8806
E75	148123	146231	149187	139989
F50	34892	32456	32459	31921
G48	41987	39898	37973	36894
P151	97945	90267	95678	98987
Q144	8998	8961	9198	9421
R780	11378	10598	10291	10198

4.4. Experiment Set4 (population size=200)

In the fourth experiment also, we observed that in most of the cases the performance of MDCI was better as compared to other techniques. The results of GA using different parameter settings is shown in Table 7 with a population size of 200 individuals. In this experiment we observed that as the population size has increased a bit MDCI is performing better compared to the other techniques like FFMRCR or MICD.

Table 7. Average convergence of data for a population size of 200 and after 1,500 generations

PROBLEM	MR=0.03, CR=0.07	FFMRCR	MDCI	MICD
A250	70457	71817	67695	71065
B150	168123	177089	155121	165901
C130	507	498	468	482
D100	9624	9354	9069	9506
E75	152123	149231	139187	143989
F50	34792	35456	32759	32492
G48	40987	40798	36973	38894
P151	94945	91267	91878	92987
Q144	9508	8926	9198	9421
R780	10378	10298	10691	10298

4.5. Experiment Set5 (population size=300)

In the fifth experiment also, we observed that in this case the performance of MDCI was better as compared to other techniques. The results of GA using different parameter settings is shown in Table 8 with a population size of 300 individuals. As the population size has increased to 300, we observed that MDCI was giving less values than MICD or other techniques.

Table 8. Average convergence of data for a population size of 300 and after 1,500 generations

Problem	MR=0.03, CR=0.07	FFMRCR	MDCI	MICD
A250	69457	70817	67995	72165
B150	178123	149189	148121	156901
C130	523	497	476	495
D100	9654	9154	8769	9216
E75	159123	146231	133187	140989
F50	35692	32438	31059	31392
G48	40727	40732	38673	38624
P151	95645	94567	91678	96787
Q144	9058	9076	8698	9321
R780	13187	10488	9987	10598

4.6. Experiment Set6 (population size=400)

In the sixth experiment also, we observed that in this case the performance of MICD was better as compared to other techniques. The results of GA using different parameter settings is shown in Table 9 with a population size of 400 individuals. Finally in experiment 6 we notice that for increased population MICD was improving and giving better values.

Table 9. Average convergence of data for a population size of 400 and after 1,500 generations

Problem	MR=0.03, CR=0.07	FFMRCR	MDCI	MICD
A250	71457	69817	67684	73165
B150	174123	149259	146321	158901
C130	520	498	477	489
D100	9254	9153	8639	9154
E75	152523	143231	135787	140989
F50	36892	34438	30659	32192
G48	43727	41732	36673	39424
P151	101645	94867	87678	96497
Q144	9258	9176	8678	9821
R780	11187	10318	9837	10291

5. CONCLUSION

The new deterministic approach to dynamically change the crossover and mutation rate was reviewed and based on the various experiments it was observed that the MICD technique was successful to give the best result for small population such as 30, 60 and 100. As the population size increased to 200,300 and 400, then in that case the MICD technique was performing better for larger population. The success of the MICD algorithm for small population is because of the lack of diversity since the search space is less. As the generation increases the algorithm prevents the genetic algorithm to get stuck at the local optima due to the increase in mutation. On the other hand, for large population the MDCI algorithm was more successful and gave better results as seen in the experiments. Since large populations are already diverse in nature so we don't need a lot of mutations hence the mutation rate is decreased. In this case to produce better offspring's we need to exchange larger portions of good parents hence the crossover rate needs to be increased so we observe that for large population the crossover rate needs to be increased and the mutation rate has to be decreased. We also observe that in both the cases whether it's a small population or large population the standard method of predefined parameter MR=0.03 and CR=0.07 did not perform well in both the cases.





REFERENCES

- [1] Holland and H. John, "Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence," MIT press, 1992, doi: 10.7551/mitpress/1090.001.0001.
- [2] K.-F. Man, K.-S. Tang, and S. Kwong, "Genetic algorithms: concepts and applications [in engineering design]," *IEEE transactions on Industrial Electronics*, vol. 43, no. 5, pp. 519-534, 1996, doi: 10.1109/41.538609.
- [3] D. Whitley, "A genetic algorithm tutorial," *Statistics and computing*, vol. 4, no. 2, pp. 65-85, 1994, doi: 10.1007/BF00175354.
- [4] A. A. Mohammed, and G. Nagib, "Optimal routing in ad-hoc network using genetic algorithm," *International Journal of Advanced Networking and Applications*, vol. 3, no. 05, pp. 1323-1328, 2012.
- [5] S. Mirjalili, J. S. Dong, A. S. Sadiq, and H. Faris, "Genetic algorithm: Theory, literature review, and application in image reconstruction," *Nature-inspired optimizers*, pp. 69-85, 2020, doi: 10.1007/978-3-030-12127-3_5.
- [6] K. M. Hamdia, X. Zhuang, and T. Rabczuk, "An efficient optimization approach for designing machine learning models based on genetic algorithm," *Neural Computing and Applications*, vol. 33, no. 6, pp. 1923-1933, 2021, doi: 10.1007/s00521-020-05035-x.
- [7] O. Kramer, "Genetic algorithms," *In Genetic algorithm essentials*, pp. 11-19, 2017, doi: 10.1007/978-3-319-52156-5_2.
- [8] S. M. Lim, A. B. M. Sultan, M. N. Sulaiman, A. Mustapha, and K. Y. Leong, "Crossover and mutation operators of genetic algorithms," *International journal of machine learning and computing*, vol. 7, no. 1, pp. 9-12, 2017, doi: 10.18178/ijmlc.2017.7.1.611.
- [9] A. E. Eiben, Z. Michalewicz, M. Schoenauer, and J. E. Smith, "Parameter control in evolutionary algorithms," *In Parameter setting in evolutionary algorithms*, pp. 19-46, 2007, doi: 10.1007/978-3-540-69432-8_2.
- [10] F. A. Zainuddin, M. F. A. Samad, and D. Tunggall, "A review of crossover methods and problem representation of genetic algorithm in recent engineering applications," *International Journal of Advanced Science and Technology*, vol. 29, no. 6s, pp. 759-769, 2020.
- [11] M. Sivaram, K. Batri, M. A. Salih, and V. Porkodi, "Exploiting the local optima in genetic algorithm using tabu search," *Indian Journal of Science and Technology*, vol. 12, no. 1, pp. 1-13, 2019, doi: 10.17485/ijst/2019/v12i1/139577.
- [12] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 5, pp. 8091-8126, 2021, doi: 10.1007/s11042-020-10139-6.
- [13] Y. Jiang, P. Wu, J. Zeng, Y. Zhang, Y. Zhang, and S. Wang, "Multi-parameter and multi-objective optimisation of articulated monorail vehicle system dynamics using genetic algorithm" *Vehicle System Dynamics*, vol. 58, no. 1, pp. 74-91, 2020, doi: 10.1080/00423114.2019.1566557.
- [14] G. Zhang, Y. Hu, J. Sun, and W. Zhang, "An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints," *Swarm and Evolutionary Computation*, vol. 54, p. 100664, 2020, doi: 10.1016/j.swevo.2020.100664.
- [15] A. Buzdalova, C. Doerr, and A. Rodionova, "Hybridizing the 1/5-th success rule with Q-learning for controlling the mutation rate of an evolutionary algorithm," *In International Conference on Parallel Problem Solving from Nature*, pp. 485-499. Springer, Cham, 2020, doi: 10.1007/978-3-030-58115-2_34.
- [16] D. Beasley, D. R. Bull, and R. R. Martin, "An overview of genetic algorithms: Part 1, fundamentals," *University computing*, vol. 15, no. 2, pp. 56-69, 1993, doi: 10.1016/0141-6359(93)90299-P.
- [17] K. D. Jong, "Adaptive system design: a genetic approach," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 10, no. 9, pp. 566-574, 1980, doi: 10.1109/TSMC.1980.4308561.
- [18] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on systems, man, and cybernetics*, vol. 16, no. 1, pp. 122-128, 1986, doi: 10.1109/TSMC.1986.289288.





- [19] H. Muhlenbein, and D. S.-Voosen, "Optimal interaction of mutation and crossover in the breeder genetic algorithm," In *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1993.
- [20] K. Deb and S. Agrawal, "Understanding interactions among genetic algorithm parameters," *Foundations of genetic algorithms*, vol. 5, no. 5, pp. 265-286, 1999.
- [21] T.-P. Hong, H.-S. Wang, W.-Y. Lin, and W.-Y. Lee, "Evolution of appropriate crossover and mutation operators in a genetic process," *Applied intelligence*, vol. 16, no. 1, pp. 7-17, 2002, doi: 10.1023/A:1012815625611.
- [22] S. G. B. Rylander, and B. Gotshall, "Optimal population size and the genetic algorithm," *Population*, vol. 100, no. 400, p. 900, 2002.
- [23] P. A. D.-Gomez, and D. F. Hougen, "Initial population for genetic algorithms: A metric approach," *In Gem*, pp. 43-49, 2007.
- [24] A. Alfeilat *et al.* "Effects of distance measure choice on k-nearest neighbor classifier performance: a review," *Big data*, vol. 7, no. 4, pp. 221-248, 2019, doi: 10.1089/big.2018.0175.
- [25] H. Chiroma, S. Abdulkareem, A. Abubakar, A. Zeki, A. Y. U. Gital, and M. J. Usman, "Correlation study of genetic algorithm operators: crossover and mutation probabilities," In *Proceedings of the International Symposium on Mathematical Sciences and Computing Research*, 2013, pp. 6-7.
- [26] Y.-P. Huang and K.-Q. Shi, "Genetic algorithms in the identification of fuzzy compensation system," In *1996 IEEE International Conference on Systems, Man and Cybernetics. Information Intelligence and Systems (Cat. No. 96CH35929)*, vol. 2, 1996, pp. 1090-1095.
- [27] J.-M. Renders and S. P. Flasse, "Hybrid methods using genetic algorithms for global optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 2, pp. 243-258, 1996, doi: 10.1109/3477.485836.
- [28] F. Vavak, and T. C. Fogarty, "Comparison of steady state and generational genetic algorithms for use in nonstationary environments," In *Proceedings of IEEE International Conference on Evolutionary Computation*, 1996, pp. 192-195, doi: 10.1109/ICEC.1996.542359.
- [29] Y. Lizhe, X. Bo, and W. Xianjie, "BP network model optimized by adaptive genetic algorithms and the application on quality evaluation for class teaching," In *2010 2nd International Conference on Future Computer and Communication*, vol. 3, 2010, pp. V3-273, doi: 10.1109/ICFCC.2010.5497635.
- [30] Z. Laboudi and S. Chikhi, "Comparison of genetic algorithm and quantum genetic algorithm," *Int. Arab J. Inf. Technol*, vol. 9, no. 3, pp. 243-249, 2012.
- [31] M. Chatteraj and U. R. Vinayakamurthy, "A self adaptive new crossover operator to improve the efficiency of the genetic algorithm to find the shortest path," *Indonesian Journal of Electrical Engineering and Computer Science (IJECS)*, vol. 23, no. 2, pp. 1011-1017, 2021, doi: 10.11591/ijeecs.v23.i2.pp1011-1017.

BIOGRAPHIES OF AUTHORS



Mrinmoyee Chatteraj     is a Research Scholar in "School of Computing & Information Technology", Reva University, Bangalore. She has 15yrs of teaching experience. Currently she is working as an Assistant Professor St. Joseph's College, Bangalore. Her research interests are Computer Network, Machine Learning and Genetic Programming. She can be contacted at email: mrinmoyee2005@gmail.com.



Dr. Udaya Rani Vinayakamurthy     is Dr. Associate Professor in Reva University, Bangalore. She holds a PhD degree in Computer Engineering with specialization in Data Mining and Warehousing She has approx 20yrs of experience and published above 20 papers in National and International Journals. She has filed two patents and have been awarded with one patent. Her research interests are Data Mining and Warehousing, Machine Learning, Big Data and Genetic Programming. She can be contacted at email: udayarani.v@reva.edu.in.