ISSN 2278-3091
Volume 13, No.2, March - April 2024
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCS1E/static/pdf/file/ijatcse091322024.pdf
https://doi.org/10.30534/ijatcse/2024/091322024

Enhancement of Genetic Algorithm by J.Zhang Applied to Tour Planning

Isabella Mae Malonzo[1], Tracy Louise Patacsil[2], Jonathan Morano[3], Vivien Agustin[4], Raymund Dioses[5]
[1]Computer Science Department, Pamantasan ng Lungsod ng Maynila, Philippines, imrmalonzo2020@plm.edu.ph
[2]Computer Science Department, Pamantasan ng Lungsod ng Maynila, Philippines, tlrpatacsil2020@plm.edu.ph
[3]Computer Science Department, Pamantasan ng Lungsod ng Maynila, Philippines, jcmorano@plm.edu.ph
[4]Computer Science Department, Pamantasan ng Lungsod ng Maynila, Philippines, vaagustin@plm.edu.ph
[5]Computer Science Department, Pamantasan ng Lungsod ng Maynila, Philippines, rmdioses@plm.edu.ph

Received Date : February 26, 2024 Accepted Date: March 29, 2024 Published Date: April 06, 2024

ABSTRACT

Following widespread lockdowns, there has been a notable increase in people's desire to travel, leading to longer and more frequent trips. This trend has created a demand for customized itineraries and tour planning. Unfortunately, manual tour planning can be challenging to optimize, time-consuming, and increasingly complex as the number of locations increases. To automate and improve tour planning, optimization methods can be used, as they leverage algorithms to find efficient routes. The Genetic Algorithm (GA), an algorithm that mimics the course of natural evolution, is adept at navigating complex search spaces and finding optimal solutions, making it suitable for solving tour planning challenges. Building upon the work of J. Zhang (2021), this study aims to improve the performance of GA by enhancing the diversity of the population, removing redundant nodes, and reducing the execution time. Two simulators were created, one for each algorithm, to test their performance. The researchers conducted tests on both the existing and enhanced algorithms. This involved the utilization of several test data that contains coordinates of several cities in the Philippines. Based on the results, the enhanced algorithm showed better results compared to the existing algorithm. In conclusion, the enhanced algorithm performed better than the exiting algorithm.

Key words: Genetic Algorithm, Tour planning, Crossover operator, Mutation operator

1. INTRODUCTION

Many consumers have a new passion for life after going through lockdowns that made international travel practically difficult, which is causing a trend that is seeing countless people travel for longer and more frequently [1]. This trend has created a demand for customized itineraries and tour planning. However, manually arranging tours can be difficult to optimize, time-consuming, and increasingly complex as the number of locations increases. To automate and improve tour planning process, optimization methods can be used, as they leverage algorithms to find efficient routes. Various algorithms can be used to solve the problem, such as Brute Force Algorithm (BFA), Ant Colony Optimization (ACO), and Genetic Algorithm (GA). This study used GA to solve tour planning problem as it adept at navigating complex search spaces and finding optimal solutions [2]. Many works have been made in GA to improve its performance, one is from J. Zhang [3]. In Zhang's GA, he introduced a new crossover process and include a local search. However, the algorithm still faces several challenges: possible occurrence of redundant nodes, low diversity, and longer execution time as the number of nodes increases. To address these challenges, the researchers modified Zhang's GA focuses on removing the redundant nodes by introducing new crossover operator, increasing the diversity using a dynamic mutation rate, and reducing the time execution by adding a validation process.

2. LITERATURE REVIEW

The genetic algorithm (GA), which is based on the principle of genetic selection, is often used to optimize search tools for challenging problems. In addition to optimization, it also aids in machine learning and research and development. With factors like selection, crossover, and mutation combined to form genetic operations that would initially be applicable to a random population, it is comparable to biology for chromosomal formation. The goal of GA is to produce solutions for succeeding generations. Success in individual production is closely correlated to the suitability of the solution it serves, ensuring that quality will improve over future generations [4].

The adoption of novel selection approaches, which eschew standard tournament selection in favor of strategies like rank-based selection and crowding distance selection, is among the most prominent improvements [5]. These techniques have shown to be more successful at locating superior solutions. Additionally, the emergence of cutting-edge crossover operators like uniform crossover and blended crossover has

helped GA more than conventional single-point and double-point crossover techniques [6]. In addition, improvements in mutation operators, such as polynomial mutation and Gaussian mutation, have shown to be more efficient in traversing the search space and identifying novel solutions. GA has been hybridized with other algorithms, like local search and machine learning methods, to further increase their capabilities. One example is Zhang's improvement on Genetic Algorithm. In his paper, he proposed a novel crossover operator, a swapping mutation operator, and incorporated a local search algorithm to expand the search space and to improve the quality of the solutions. The simulation indicates the proposed algorithm surpasses the standard GA in terms of its efficiency. It was also found that it is computationally simple and easy to carry out.

This collaborative strategy enables GA to produce a set of initial candidate solutions that are then improved by specialized algorithms [7]. Together, these improvements make GA more effective and adaptable optimization tools, with applications in machine learning, engineering, and a variety of other fields.

Despite this, there are still several weaknesses GA faces as it is used in the said problems. The traveling salesman problem is one of several optimization issues for which GA can be used to discover a solution [8]. TSP is a minimization problem that asks for the shortest path that makes exactly one stop at each city before returning to the starting point. The best chromosome, which is a tour, is returned as the solution in GA after a predetermined number of random tours have been generated and improved populations have been reached [9].

However, it has been observed that some nodes repeatedly appear in GA, having locations visited more than once. Therefore, it is impossible to achieve diversity or exploration [10]. As stated, gene repetition is one of the drawbacks seen in its application to routing problems. Repeated nodes may generate similar or identical solutions that violate the TSP constraint and can reduce population diversity, resulting in ineffective methods because of the time lost in creating and evaluating infeasible solutions [11]. The algorithm's capacity to traverse the search space and produce optimal solutions may be hampered as a result [12].

It has also been observed that the computation time of the GA increases with the problem size [13]. There are several factors in the algorithm that contribute to the increase of the execution time such as the parameters used, the choice of operators, as well as the recalculation of fitness score of each chromosome in each iteration. Katoch et al. [14], in their review of genetic algorithm, revealed the procedure of GA which includes the computation of the fitness score of each chromosome that is necessary for the selection of the parents in each generation.

In relation to the process of GA, the mutation operator is the mechanism that preserves genetic diversity from one group to the next [14]. One chromosome is all that is needed for a mutation to produce a kid chromosome because it is an asexual operator. To prevent premature convergence, these operators allow the population's evolution to retain its random aspect [15]. A beneficial mutation confers a fitness benefit on the individual, increasing the likelihood of reproduction and the ability to pass on the advantageous characteristic to future generations. On the other hand, an organism that undergoes a harmful mutation is likely to be eliminated, and the characteristic might vanish in subsequent generations [16]. Yang [17] pointed out that mutation at a single site is not very efficient as mutation can be local if the mutation rate is low, and the step sizes are very small. The choice of mutation operator will then be a major factor in the exploration of the best possible solution. In addition, Vie et al. [16] also noted that overly high mutation rates result in ineffective random search, but an excessively low mutation rate can't stop a particular portion from sticking around in the population forever or can't stop early convergence. Therefore, the choice of the mutation parameter will determine if the algorithm converges to the best possible solution.

3. METHODOLOGY

The methodology aims to address challenges in an existing algorithm, specifically node duplication and decreased diversity, through strategic modifications and enhancements.

To tackle node duplication, a novel approach utilizing a single parent in crossover operations is proposed, ensuring offspring do not contain repeated nodes, as seen in Figure 1. This prevents the emergence of infeasible solutions without the need for repair mechanisms, thus improving efficiency.

Additionally, a dynamic mutation rate is introduced to counteract decreased diversity. Mutation probability significantly impacts population variability, with high rates promoting exploration and low rates facilitating exploitation. To balance the exploration and exploitation capabilities of the operator, the proponents have decided to utilize a dynamic mutation rate that would change based on the number of generations used [18]. This ensures effective exploration of the search space while maintaining quality solutions and avoiding convergence to suboptimal outcomes. The formula used to calculate the mutation rate is;

$$MR = \frac{LG}{Gn} \quad (1)$$

$$M = MR * population\ size \quad (2)$$

where: MR is the dynamic mutation rate, LG is the current generation, Gn is the total number of generations, M is the number of individuals to be mutated, and population size is the number of individuals in a population.

Furthermore, to mitigate increased computational time, a cache system is incorporated to store previously evaluated fitness scores, as shown in Figure 2. This prevents redundant evaluations of similar individuals, optimizing the selection process and overall efficiency of the genetic algorithm.

To test the performance of both algorithms, the researchers created two simulators, one for J.Zhang's GA, which will be

called the existing algorithm, and one for the enhanced algorithm. The test data used is comprised of coordinates of cities in the Philippines with 11, 108, 502 and 1345 locations, along with TSP instances like kroA100, rat575, and pr1002. Each algorithm undergoes five trials for performance evaluation.

Parent: 123|4567|89 → 891234567 → 891234567 (Child 1)

Subset: 1~~2~~3~~45678~~9 → xxxx13469 → 257813469 (Child 2)

Figure 1: Single Parent Crossover Operator

Individual	Fitness Values
1234	5+4+2 = 11
4321	2+4+5 = 11
3214	4+5+3 = 12
2341	4+2+3 = 9

Generation 1

Individual	Fitness Values
1234	11
2341	9
1243	5+5+2 = 12
1234	5+4+2 = 11

Generation 2

Individual	Fitness Values
1234	11
4321	11
3214	12
2341	9

Storage

Figure 2: Validation Process before Evaluation Fitness Scores

4.RESULTS AND DISCUSSION

This section discusses the results of the methods that researchers have applied to enhance J. Zhang's Genetic Algorithm.

Table 1 presents that across all iterations, the enhanced algorithm consistently demonstrated superior performance compared to the existing algorithm. Specifically, the enhanced algorithm exhibited the ability to visit every node exactly once within each trial. On the over hand, the existing algorithm exhibited inefficiencies, as evidenced by instances of node duplication or omission, thus indicating its shortcomings in ensuring optimal traversal of the nodes.

In this study, average diversity refers to the average difference or range between various solutions within the population. It illustrates the varying performance or characteristics of each solution compared to the others. Figure 3 and Figure 4 shows the progression of the average diversity and best distance across generations of both existing and enhanced algorithms, respectively, with a dataset containing 1345 nodes. As seen in Figure 3, the existing algorithm is seen to have a smaller difference on the distance between nodes ranging between 30-60, indicating that the algorithm only explores a localized area of the search space limiting its likelihood to get better results. It can also be seen that its diversity significantly declines as the generation increases. On the other hand, in Figure 4, the enhanced algorithm's diversity ranged between 40-100, indicating that it has effectively explored a wider range of the search space.

In addition, the existing algorithm had a best distance of 6021.249 while the enhanced algorithm recorded 5915.147 as its best distance. This suggests that the enhanced algorithm is capable of getting better results than the existing algorithm.
The results were further evaluated by utilizing different datasets to evaluate its efficiency on nodes of varying sizes. The researchers used datasets with 11,108, and 502 nodes to challenge the findings recorded in both algorithms. The behavior of the existing algorithm's diversity across datasets were the same, observing a significant decline as the generations progress. As shown in Table 2, the enhanced algorithm consistently outperforms the existing version in providing better solutions, except in the 11-node dataset. This occurs because, as seen in Table 1, the current algorithm is only traversing nodes between 3-6 nodes disregarding 5-8 nodes resulting to a shorter path than the improved version. Moreover, it was found that the enhanced algorithm consistently provides better solutions in comparison to the existing one across all datasets, evidence that the enhanced is better in exploring the search space in finding a better solution for the given dataset.

Table 3 shows that the enhanced algorithm consistently had a shorter execution time than the existing one, and the difference became more noticeable as the number of nodes increased. These results strongly suggest that enhanced algorithm is better overall when it comes to speed and efficiency in all instances.

Table 1: Comparison of Number of Nodes Traversed

No.of Nodes	Traversed Nodes									
	Existing Algorithm					Enhanced Algorithm				
	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
11	3	3	5	5	6	11	11	11	11	11
108	106	107	105	107	107	108	108	108	108	108
502	492	498	492	492	495	502	502	502	502	502
1345	1337	1335	1335	1344	1336	1345	1345	1345	1345	1345

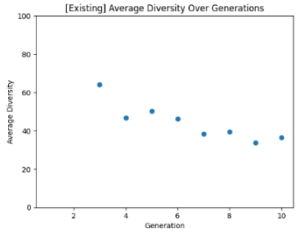

Figure 3: Existing Algorithm Diversity

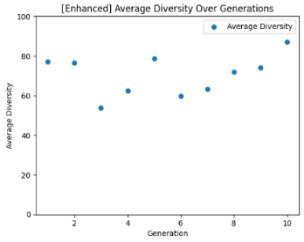

Figure 4: Enhanced Algorithm Diversity

Table 2: Comparison of Best Distance Obtained

No. of Nodes	Best Distance	
	Existing Algorithm	Enhanced Algorithm
11	0.216	2.097
108	499.787	445.156
502	2202.892	2065.199
1345	6021.249	5915.147

Table 3: Comparison of Total Execution Time

No. of Nodes	Total Execution Time (seconds)					
	Existing Algorithm			Enhanced Algorithm		
	T1	T2	T3	T1	T2	T3
11	1.82	1.79	1.77	1.71	1.72	1.74
108	3.74	2.72	2.90	2.00	1.96	1.98
502	5.24	5.16	5.18	3.13	3.09	3.12
1345	9.64	9.13	9.17	5.10	5.06	5.14

5. CONCLUSION

Genetic algorithm is a well-known algorithm used to solve optimization problems. It can effectively search over a sizable and complex search space and produce useful results. Tour planning stands as one such optimization problem that genetic algorithms can address. However, there are still improvements that were made to further enhance the performance of the algorithm.

In the enhanced algorithm, the possibility of producing invalid solutions was eliminated by introducing a single-parent crossover instead of the previous crossover operator, resulting in no repeated or missed locations. Next, the diversity of the solutions was increased using a dynamic mutation rate, causing the expansion of the search space. Lastly the computation of similar solutions was eliminated by incorporating a validation process before evaluating the fitness score. As a result, the time execution was reduced.

Overall, the points mentioned above prove that the enhanced algorithm is effective and performed better than the existing algorithm.

ACKNOWLEDGMENT

The Authors would like to thank Pamantasan ng Lungsod ng Maynila College of Information System and Technology Management and Computer Science Department for supporting this academic endeavor.

REFERENCES

1. Euronews. **Post-COVID "revenge travel" has gone big. And the revenge is sweet**. Euronews. 2023. Available from: https://www.euronews.com/2023/04/21/post-covid-revenge-travel-has-gone-big-and-the-revenge-is-sweet
2. Singh, A., Singh, R. **Exploring Travelling Salesman Problem using Genetic Algorithm**. International Journal of Engineering Research and Technology. 2014; 3(2):2032-35.
3. Zhang J. **An improved genetic algorithm with 2-opt local search for the traveling salesman problem.** In: Application of Intelligent Systems in Multi-modal Information Analytics. Cham: Springer International Publishing; 2021. p. 404–9.
4. Lambora A, Gupta K, Chopra K. **Genetic algorithm- A literature review**. In: 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon). IEEE; 2019. p. 380–4.
5. Pandey HM. **Performance evaluation of selection methods of genetic algorithm and network security concerns**. Procedia Comput Sci. 2016;78:13–8. Available from: http://dx.doi.org/10.1016/j.procs.2016.02.004
6. Dou X-A, Yang Q, Gao X-D, Lu Z-Y, Zhang J. **A comparative study on crossover operators of genetic algorithm for traveling salesman problem.** In: 2023 15th International Conference on Advanced Computational Intelligence (ICACI). IEEE; 2023.
7. Souza F, Grimes D. **Combining Local Search and Genetic Algorithm for Two-Dimensional Guillotine Bin Packing Problems with partial sequence constraint**. Ceur-ws.org. Available from: https://ceur-ws.org/Vol-2771/AICS2020_paper_34.pdf
8. Alzyadat T, Yamin M, Chetty G. **Genetic algorithms for the travelling salesman problem: a crossover comparison.** Int J Inf Technol. 2020;12(1):209–13. Available from: http://dx.doi.org/10.1007/s41870-019-00377-9
9. Fu C, Zhang L, Wang X, Qiao L. **Solving TSP problem with improved genetic algorithm**. In: AIP Conference Proceedings. Author(s); 2018.
10. Akter S, Nahar N, Shahadat M, Andersson K. **A new crossover technique to improve genetic algorithm and its application to TSP**. In: 2019 International Conference on Electrical, Computer and Communication Engineering (ECCE). IEEE; 2019.
11. Doerr B, Johannsen D, Kötzing T, Neumann F, Theile M. **More effective crossover operators for the all-pairs shortest path problem.** Theor Comput Sci. 2013;471:12–26. Available from: http://dx.doi.org/10.1016/j.tcs.2012.10.059
12. Paper R, Malik MS, Wadhwa MS. **Preventing premature convergence in genetic algorithm using DGCA and elitist technique.** 2014; Available from: https://www.semanticscholar.org/paper/db7678845db91e5cdabbd2767de11c1e85eae6ae
13. Xie X, Zheng Y, Li Y. **Genetic algorithm and its performance analysis for scheduling a single crane.** Discrete Dyn Nat Soc. 2015;2015:1–12. Available from: http://dx.doi.org/10.1155/2015/618436
14. Katoch S, Chauhan SS, Kumar V. **A review on genetic algorithm: past, present, and future**. Multimed Tools Appl. 2021;80(5):8091–126. Available from: http://dx.doi.org/10.1007/s11042-020-10139-6
15. Othman SB, Ajmi I, Quilliot A. **Agent-based architecture for task scheduling and dynamic orchestration support.** In: Logistics Engineering and Health. Elsevier; 2016. p. 139–91.

16. Vié A. **Qualities, challenges and future of genetic algorithms**. SSRN Electron J. 2020; Available from: http://dx.doi.org/10.2139/ssrn.3726035
17. Yang X-S. **Genetic Algorithms. In: Nature-Inspired Optimization Algorithms**. Elsevier; 2014. p. 77–87.
18. Hassanat A, Almohammadi K, Alkafaween E, Abunawas E, Hammouri A, Prasath VBS. **Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach.** Information (Basel). 2019;10(12):390. Available from: http://dx.doi.org/10.3390/info10120390