Computing Science

Comparative analysis of novel types of mutations in Genetic Algorithm for solving the Travelling Salesperson Problem

Julia Wiktoria Zieba

No portion of the work contained in this document has been submitted in support of an application for a degree or qualification of this or any other university or other institution of learning. All verbatim extracts have been distinguished by quotation marks, and all sources of information have been specifically acknowledged.

Signed: Julia Wiktoria Zieba

Date: November 30, 2020

CS4040 Report

Department of Computing Science **November 2020**
University of Aberdeen
King's College
Aberdeen AB24 3UE

Comparative analysis of novel types of mutations in Genetic Algorithm for solving the Travelling Salesperson Problem

Julia Wiktoria Zieba

Department of Computing Science
University of Aberdeen

November 30, 2020

Abstract: The Travelling Salesman Problem is a widely recognised optimisation problem, applicable to various real-life scenarios. Genetic Algorithm is a reliable meta-heuristic approach for solving this problem. This paper explores and compares performance of different novel mutation operators in GA in order to find the most reliable one. Investigated novel mutation operators are as follows: Inverted Exchange Mutation, Inverted Displacement Mutation, and Odd-Even Position Wise Mutation. They are compared against Reverse Sequence Mutation. Results of conducted experiments showed that Inverted Displacement Mutation provides the best performance.

1 Introduction

The Travelling Salesperson Problem (TSP) is a benchmark optimisation problem in Computing Science, considering how often is it discussed. Thus, innovative approaches to solving it are continuously being proposed. This paper will explore and compare performance of different mutation operators used in Genetic Algorithm (GA) for solving this problem, in order to find the most suitable one.

1.1 Genetic Algorithm

GA is a meta-heuristic inspired by the process of natural selection described by Charles Darwin [6]; it is an approximation algorithm - the aim is to find an optimal solution in a reasonable amount of time. GA provides a powerful tool to do so, as it has been proven to work successfully for a number of different problems, such as scheduling workflow applications [3] and image processing [10].

The main principles of GA are: spawning population, fitness assessment, selection of parents, crossover and mutation for creating new offspring, and narrowing the population down.

Mutations play a key role in assuring diversity of the population [21]. As in real life, mutations happen with a random probability to a random individual; although they might be beneficial, sometimes they result in degrading fitness of the mutated offspring.

1.2 Travelling Salesperson Problem

TSP is an optimisation problem, the goal of which is to find the shortest path between n cities, given the city pair-wise distances [11]; it is an *NP-hard* problem [15], which means that it cannot be solved exactly in a polynomial time [9]. The TSP can be solved in a deterministic way using a brute force approach, that is, by comparing all possible solutions and picking the best one. However, such an algorithm would have an complexity of $O(n!)$ - this can be easily proven using combinatorial techniques. Thus, with the increase of the number of cities, the number of possible tours increases factorially [20].

2 Background and Related Work

The use of GA for solving the TSP has been widely discussed, seeing as this meta-heuristic and its variations are efficient in finding optimal solutions to this dilemma [18].

The performance of different mutation operators for solving the Travelling Salesperson Problem has been addressed previously [2]. A number of different established mutation operators have been compared, and their effectiveness has been assessed based on the number of iterations it took to arrive at the optimal solution. This research showed that the Reverse Sequence Mutation (RSM) is the most efficient one, with the Partial Shuffle Mutation (PSM) following it closely. The main difference between the paper described above and this one is that this paper explores novel mutation operators instead of commonly-used ones.

Different novel mutation operators have been proposed in order to improve the performance of the GA. Approaches this paper is concerned with are described in following sections.

2.1 Inverted Displacement and Inverted Exchange Mutations

Inverted Displacement Mutation (IDM) and Inverted Exchange Mutation (IEM) were presented in a study conducted by Deep and Mebrahtu [7]. Both of these proposals are a combination of previously established methods, such as Inversion Mutation, Displacement Mutation, Exchange Mutation, and Insertion Mutation.

IDM selects two points in the chromosome, inverts the cities in this substring, and then inserts obtained substring at randomly chosen position.

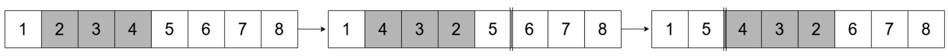


Figure 1: Inverted Displacement Mutation

IEM selects two points in the chromosome, inverts the cities in this substring, and then exchanges a randomly chosen city from this substring to a city from outside the substring.

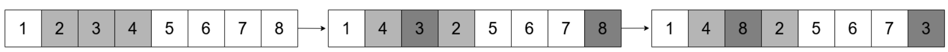


Figure 2: Inverted Exchange Mutation

It was showed that IDM and IEM operators perform better than other mutation operators discussed in this research paper. IDM was observed to be superior to other ones by giving the best performance in 7 out of 10 presented problems, with IEM following closely. However, in 3 problems, the minimal cost was obtained by using the Inversion Mutation.

2.2 Odd-Even Position Wise Mutation

In his study, Lakshmi proposes an Odd-Even (OE) Position Wise mutation operator, which is a complex combination of other mutation operators [17]. This approach was proposed in order to address and overcome disadvantages of established operators, whilst also capitalising on their benefits.

In OE, a lower and a higher bound are chosen within the chromosome - elements within those indices are considered for a position-wise interchange. Then, elements on odd positions are interchanged in an orderly manner; the same goes for the elements on even positions.

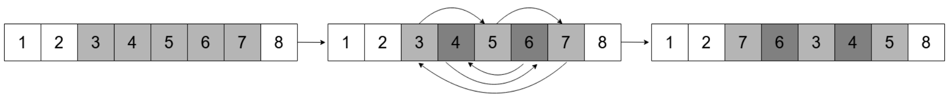


Figure 3: Odd-Even Mutation

This study showed that the newly developed OE operator is more successful than other established mutation operators with respect to the speed and error. Additionally, it was showed that OE mutation provides optimal solutions in less time than established methods, even when the problem size is large.

Many other novel operators, such as the Power Mutation (PM) [8] and Greedy Sub Tour Mutation (GSTM) [4], have been proposed. However, given limited resources, only the previously described ones are going to be considered in this paper.

3 Research Question

The purpose of this study is to investigate the performance of different mutations in GA for solving the TSP. I will compare new approaches to established ones that are proven to work efficiently. Considering their effectiveness described in research papers [8, 4, 9], these mutations have great potential for improving the computational power of solving optimisation problems.

I will inspect how different types of novel mutation operators - IDM, IEM, and OE - influence the performance of the GA for TSP and how they present in relation to the best established RSM approach. The choice of these operators was made based on the fact that all of them were developed by combining already existing approaches. IDM has showed better performance than IEM in more tests conducted by their proposers, however, both operators were chosen to be investigated, as it provides more information in relation to RSM.

Operators presented above aim to improve the time, complexity, and the overall performance of GA for TSP. Thus, research questions of this paper are as follows:

- Are novel mutation operators superior to the RSM?
- Which of these operators provides the best performance?
- How significant is the difference in performance of examined mutation operators?

The hypothesis is that new methods of implementing mutation operators in GA perform better than RMS, as they were specifically designed with the TSP in consideration. Additionally, given the time period between devising the RSM and novel methods, it might be reasonable to assume that the new ones improve the effectiveness of GA.

In order to answer research questions, an experiment consisting of a simulation of chromosome population has to be designed, an adequate representation of the problem must be described, and a number of different test data sets have to be used to assure generality. Additionally, to reliably assess performance of each operator and make valid comparisons, statistical analysis, such as investigating significance of findings, will be performed on obtained results.

Test data sets used in the experiment were adapted from the TSPLIB [12] by John Burkardt [5], and are presented in a form of an inter-city distance table. Additionally, shortest possible tour is provided for each data set to assure correctness [13]. Previously processed data sets were used to conduct experiments, as this project is not concerned with the way of calculating Euclidean distances between cities, which has to be done when using raw data from TSPLIB.

Due to time constraints, the scope of the project was narrowed down to exploring only three novel approaches. Additionally, using previously processed data sets limits the re-usability of the produced code, as it is representation-specific.

4 Experimental Design

Hypothesis that novel operators perform better than RSM and that they improve the overall performance of GA, is going to be tested by applying GA with different mutation operators implemented to the TSP problem. For this research, a Python3 program for simulating GA was developed. Pseudocode for a basic GA can be found in appendix A. Independent variables in this experiment are different mutation operators - RSM, IEM, IDM, and OEM - and dependent variables under inspection are path distance progression over 500 generations, average error in relation to the true shortest distance of each problem set, and mean run-time produced by each variant of GA.

For the experiment, a chromosome population of 20 individuals is generated, and then the GA is run on the same population for each of the mutation operators on one of the data sets. This approach is then applied to remaining data-sets. The algorithm iterates through 500 generations of chromosomes, and then terminates. This sequence is then repeated 50 times, in order to assure that obtained results represent an average and normal case. Additionally, each run of the GA with a different mutation operator is timed, to investigate performance and complexity of each operator in relation to time.

In order to conduct experiments, a variant of the TSP had to be selected. As there are many different versions of it, one has to be explicitly specified in order to make the implementation and solution clear. For this study, a Symmetric TSP was chosen, as this is the most popular, as well as the simplest variant of the problem [19]. Its main aspect is that the tour can be evaluated in both directions for the same cost [14]: if d_{ij} represents distance from city i to city j, then $d_{ij} = d_{ji}$. Representation of the TSP had to be correctly designed, as the selected type of the problem has specific assumptions, i.e. no city can be visited twice. Thus, permutation tour representation was chosen, as TSP is an ordering problem. Each position in the permutation encoded individual represents a city, and the whole chromosome represents one path [16]. This type of representation is used to present mutation operators in figures 1, 2, and 3.

Several design decisions concerning parts of GA that are not going to be examined in the experiments were made. The Ordered Crossover operator (OX) has been chosen as a crossover operator, since it has been determined to be the most efficient one out of established methods [1]. Elitist parent selection [21] has been chosen for this experiment, as there are time constraints, and this selection process is the most efficient to implement, as it is not very complex. This selection process guarantees that the fittest members of chromosome population will be chosen for breeding.

Experiments were conducted on three different data sets in order to assess performance of each mutation operator for different sizes of problems. Additionally, this approach assures that results are not biased towards a specific problem size. Inspecting a broader range of TSP instances allows for more in-depth analysis of results, as one operator might be more suitable for problems of a specific size.

Data sets chosen for the problem are as follows:

1. gr17 - set of 17 cities with the minimal distance of 2085

2. fr26 - set of 26 cities with a minimal distance of 937

3. att48 - set of 48 cities with the minimal distance of 33523

All experiments were conducted on a laptop machine with Intel(R) Core(TM) i5-7200U 2.5GHz in CPU and 16.0GB RAM with a Microsoft Windows 10 Home as an operating system.

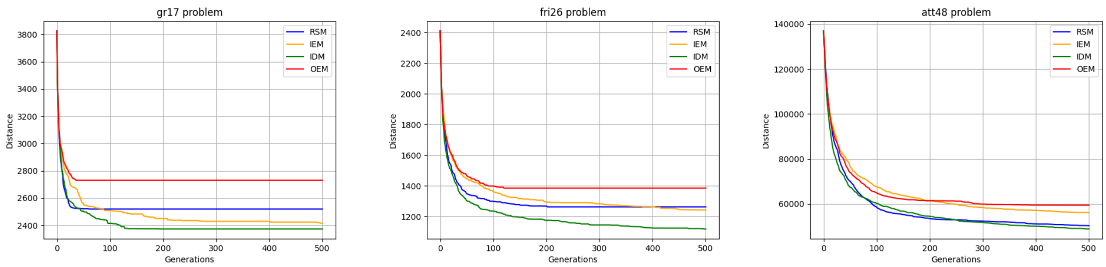
5 Results

This sections presents findings obtained during the course of experiments. Additionally, results of statistical analysis are described. The cut-off value of α used for assessing the statistical significance is 0.05.

5.1 Performance of finding minimal length path

The performance of GA implementing each mutation operator can be seen in figure 4 below. These graphs show what the best path distance is in each generation, and how it changes over time.

The IDM operator showed the best performance for each problem size - the slope on the graph line is the steepest, and it gets the closest to the actual minimal path length. Whereas IDM consistently produced results closest to the actual solution for each problem size, it can be observed that there are some changes in performance of RSM and IEM. For smaller problem sizes - gr17 and fri26 - RSM is outperformed by IEM, however, for att48 it is the RSM that produces a better solution. Lastly, OEM consistently produced the worst results out of these 4 operators.

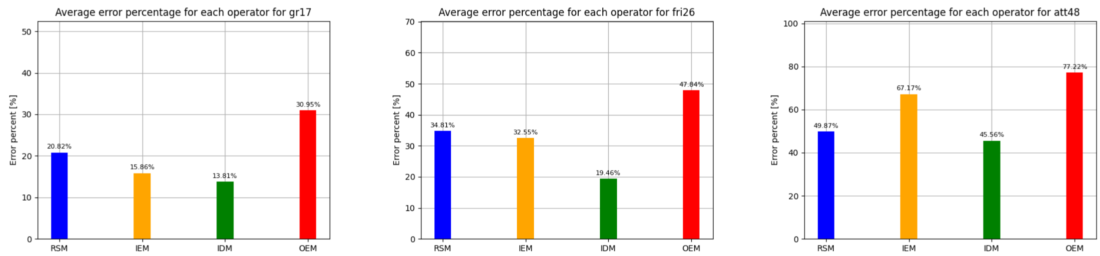


(a) (b) (c)

Figure 4: Performance for (a) gr17, (b) fri26 and (c) att48 problems

In gr17 it can be observed that the gradient of each line converges to 0 very early; in fri26 gradient of IDM and IEM is negative throughout 500 generations of GA; in att48 only the gradient of OEM line converges to 0.

The difference between the output of the GA algorithm was also compared with the actual shortest path for each mutation operator and each problem size. Average error in percents can be seen in the figure 5 below.



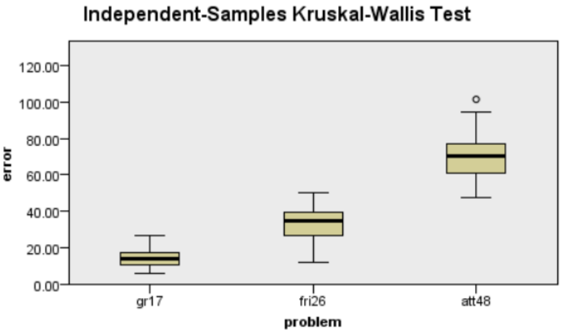
(a) (b) (c)

Figure 5: Error percentage for (a) gr17, (b) fri26 and (c) att48 problems

IDM has consequently achieved the lowest average error of 13.81%, 19.46%, and 45.56% for gr17, fri26, and att48 respectively.

The biggest average error for each problem size has been observed for OEM. Additionally, it was observed that even though IEM performed relatively well for the smallest problem size, its error percentage increased significantly for bigger problems. The Shapiro-Wilk test showed that the error data for gr17 problem is not distributed normally, $W(50) = 0.94$, $p = 0.014$. Thus, Kruskal-Wallis test was used to asses statistical significance of error increase, as it does not require the same normality assumption as ANOVA.

In the figure 6 presented below it can be observed that p values for pairwise comparison between problem sizes also adhere to the $p < \alpha$ rule, as all of them are smaller than 0.00. Thus, differences between distinct data-sets are statistically significant.



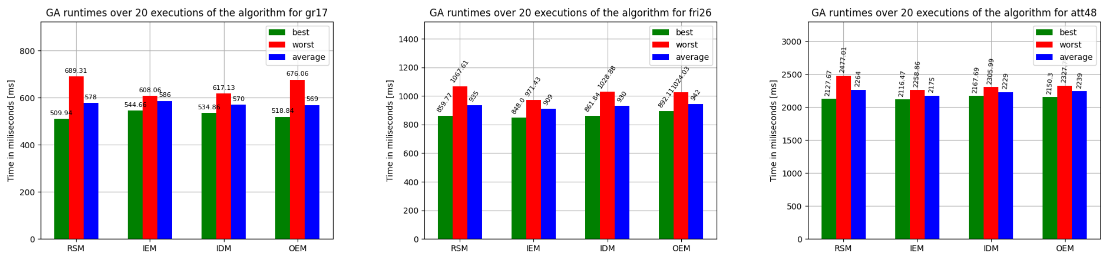
Each node shows the sample average rank of problem.

Sample1-Sample2	Test Statistic	Std. Error	Std. Test Statistic	Sig.	Adj.Sig.
gr17-fri26	-46.840	8.689	-5.391	.000	.000
gr17-att48	-98.360	8.689	-11.320	.000	.000
fri26-att48	-51.520	8.689	-5.929	.000	.000

Figure 6: Kruskal-Wallis test results for increase in IEM error

5.2 Run-time analysis

Run-time analysis of each mutation operator can be seen in figure 7 below.



(a) (b) (c)

Figure 7: Run-time for (a) gr17, (b) fri26 and (c) problems

Run-times of GA with each operator do not show any significant discrepancies. In the gr17 problem, RSM had the worst average run-time of 689.31 milliseconds, and IEM had the best one equal to 608.06 milliseconds. The biggest differences in run-time can be observed for the smallest problem size - gr17.

It was impossible to assess significance of the run-time data using a standard one-way ANOVA, as it was not normally distributed, which violated one of assumptions needed for performing this test. In the figure 8 below results of normality testing can be observed:

Tests of Normality

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
rsm	.074	50	.200*	.950	50	.034
iem	.139	50	.017	.964	50	.128
idm	.159	50	.003	.866	50	.000
oem	.184	50	.000	.768	50	.000

Figure 8: Normality test for fri26 runtime data

Running normality test showed that the p values of the Shapiro-Wilk Test for RSM, IDM, and OEM are smaller than 0.05, which means that the data significantly deviate from a normal distribution.

5.3 Performance with respect to RSM

In none of the inspected instances of TSP did RSM achieve the worst performance out of all tested operators. In fri26, RMS performed on the same level as IEM, and in gr17 it was second to last, with only OEM performing worse. However, in fri26 it can be observed that the slope of the IEM line is steeper that the slope of RSM line, which might suggest that given more generations IEM would outperform RSM. For gr17 and fri26, RSM has achieved significantly worse performance than the best operator. However, in att48, RSM produced results very close to IDM - the difference in average error (figure 5) was only 4.31%.

On average, RMS produced 20.82% error over 50 runs of GA for the gr17 problem, and 34.81% and 49.87% for fri26 and att28 respectively (figure 5). In comparison to that, IDM produced 13.81%, 19.46% , and 45.56% error for each of the problem sizes.

6 Discussion

From the conducted experiment it can be concluded that IDM is the most suitable mutation operator in GA for solving TSP. It has consequently produced very good and promising results, as it provided solutions closest to true ones for each problem size, and consequently generated smallest amount of error.

After inspecting the slope of the IDM line in the graph illustrating how the shortest path distance changes over the course of the GA run (figure 4) it can be concluded that this operator does not converge to a local minimum as quickly as other mutation operators. Thus, the solution is being improved for a longer amount of time and has a bigger chance to come closer to the actual shortest path.

Additionally, the hypothesis that novel mutation operators provide an overall better performance than RSM was rejected, as the results showed it does come close to minimal path distances in each problem. RSM performs closely to IDM, and can still be considered to be one of the benchmark approaches to TSP.

As for each run of the GA with a different mutation operator the same population and data-sets were used, the difference in performance - final error percentage, etc. - can be attributed to the specific mutation operator.

Narrowing execution of GA to only 500 generations and a population of 20 chromosomes creates some limitations, as it might be difficult for GA to get close enough to a correct shortest path distance in larger problem sizes, as seen in figure 5. For the data-set with the largest number of cities (att48), it can be observed that the error percentage is high, even for the best performing IDM (45.56%), which might suggest that the number of generations in GA was insufficient to minimise the solution. Experimenting with different population sizes and number of generations might be beneficial for creating more reliable results.

Additionally, implementing pure elitist selection algorithm poses further limitations, such as a possible premature convergence to a local minimum, as this approach increases the exploitability of the GA. This could be overcome by implementing a different parent selection method, such as Roulette Wheel Selection, or Tournament Selection [21].

Conducting experiments only on one machine also could affect the outcome. Obtained results are biased towards specific machine specifications, and do not represent a broader spectrum of hardware. However, as this machine is a very common one and does not have any extra improvements, it shows how accessible performing such problem-solving operations is for most users.

7 Conclusion & Future Work

This research was concerned with comparing novel mutation operators and finding which one of them performs the best in solving the TSP in order to establish new best way of using GA for TSP. Experiments have showed that IDM performs the best for all inspected problem sizes - solution produced by IDM is on average closer to the true solution than solutions produced by other operators in the same amount of time - which makes it a very versatile operator.

The hypothesis that all novel mutation operators would perform better than RSM was not supported, as this approach has consequently dominated over OEM in all cases. This research showed that even though new operators are being developed with solving TSP in mind, RSM still produces solid results in relation to new ones.

As it was previously mentioned, there are numerous new mutation operators, that could not be covered by this research. This indicates a potential for further research in this direction. Next steps would be to perform an exhaustive comparative analysis of a large number of novel mutation operators. Additionally, incorporating different parent selection techniques could also be taken into consideration.

Another direction for continuing research in this area would be developing a mutation operator that is even more superior to the tested ones. This research would possibly focus on extracting benefits of each operator and overcoming their limitations.

8 Reflective Analysis

As there is not a project that does not come across any obstacles, creation of this paper was a mixture of small successes, as well as failures. During this project too much time was spent on making cosmetic changes to the graphical results. If one was to start over, it could be beneficial to focus on gathering more diverse data and performing more statistical tests on it. One could also benefit from more robust and thought-through experiment design and result analysis design, as not having an explicit plan took a significant toll on time management.

Appendices

A GA pseudocode

Algorithm 1 Genetic Algorithm

```
population generation
fitness measurement
for iteration = 1, 2, ..., 500 do
    parent selection
    crossover
    mutation
    fitness measurement
    narrowing population
end for
```

B Word count

Word count according to Overleaf:

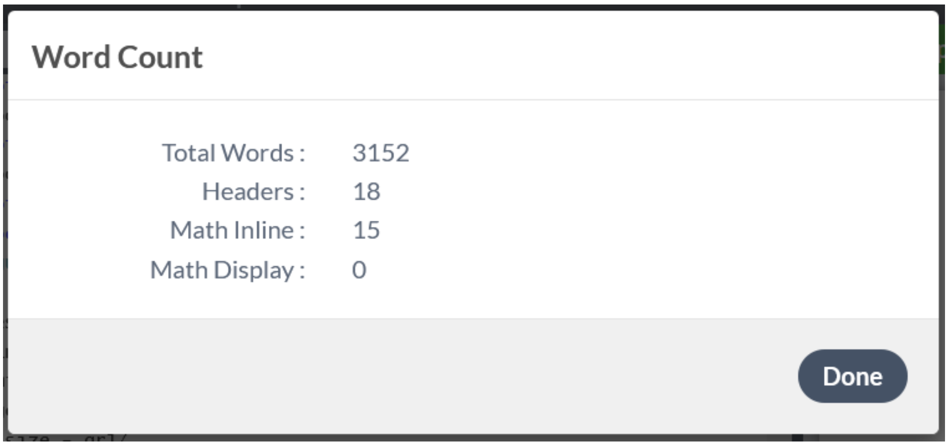


Figure 9: Word count

References

[1] Otman Abdoun and Jaafar Abouchabaka. A comparative study of adaptive crossover operators for genetic algorithms to resolve the traveling salesman problem. *arXiv preprint arXiv:1203.3097*, 2012.

[2] Otman Abdoun, Jaafar Abouchabaka, and Chakir Tajani. Analyzing the performance of mutation operators to solve the travelling salesman problem. *arXiv preprint arXiv:1203.3099*, 2012.

[3] Saima Gulzar Ahmad, Chee Sun Liew, Ehsan Ullah Munir, Tan Fong Ang, and Samee U Khan. A hybrid genetic algorithm for optimization of scheduling workflow applications in heterogeneous computing systems. *Journal of Parallel and Distributed Computing*, 87:80–90, 2016.

[4] Murat Albayrak and Novruz Allahverdi. Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms. *Expert Systems with Applications*, 38(3):1313–1320, 2011.

[5] John Burkardt. Cities: City distance datasets. `https://people.sc.fsu.edu/~jburkardt/datasets/cities/cities.html`. Accessed: 23 Oct, 2020.

[6] Charles Darwin. *The origin of species*. PF Collier & son New York, 1909.

[7] Kusum Deep and Hadush Mebrahtu. Combined mutation operators of genetic algorithm for the travelling salesman problem. *International Journal of Combinatorial Optimization Problems and Informatics*, 2(3):1–23, 2011.

[8] Kusum Deep and Manoj Thakur. A new mutation operator for real coded genetic algorithms. *Applied mathematics and Computation*, 193(1):211–230, 2007.

[9] Varshika Dwivedi, Taruna Chauhan, Sanu Saxena, and Princie Agrawal. Travelling salesman problem using genetic algorithm. *IJCA Proceedings on Development of Reliable Information Systems, Techniques and Related Issues (DRISTI 2012)*, 1:25, 2012.

[10] Minglun Gong and Yee-Hong Yang. Multi-resolution stereo matching using genetic algorithm. In *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, pages 21–29. IEEE, 2001.

[11] Sanchit Goyal. A survey on travelling salesman problem. In *Midwest Instruction and Computing Symposium*, pages 1–9, 2010.

[12] Ruprecht-Karls-Universität Heidelberg. Discrete and combinatorial optimization: Tsplib. `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`. Accessed: 22 Oct, 2020.

[13] Ruprecht-Karls-Universität Heidelberg. Optimal solutions for symmetric tsps. `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html`. Accessed: 22 Oct, 2020.

[14] K Ilavarasi and K Suresh Joseph. Variants of travelling salesman problem: A survey. In *International Conference on Information Communication and Embedded Systems (ICICES2014)*, pages 1–7. IEEE, 2014.

[15] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[16] Anit Kumar. Encoding schemes in genetic algorithm. *International Journal of Advanced Research in IT and Engineering*, 2(3):1–7, 2013.

[17] R Lakshmi. Effective oe position-wise mutation technique for permutation encoded genetic algorithm to solve school bus routing problem: mtsp approach. *International Journal of Computer Science and Information Security (IJCSIS)*, 14, 2016.

[18] Bao Lin Lin, Xiaoyan Sun, and Sana Salous. Solving travelling salesman problem with an improved hybrid genetic algorithm. *Journal of computer and communications.*, 4(15):98–106, 2016.

[19] Rajesh Matai, Surya Prakash Singh, and Murari Lal Mittal. Traveling salesman problem: an overview of applications, formulations, and solution approaches. *Traveling salesman problem, theory and applications*, 1, 2010.

[20] Antima Sahalot and Sapna Shrimali. A comparative study of brute force method, nearest neighbour and greedy algorithms to solve the travelling salesman problem. *International Journal of Research in Engineering & Technology*, 2(6):59–72, 2014.

[21] Anita Thengade and Rucha Dondal. Genetic algorithm-survey paper. In *MPGI National Multi Conference*, pages 7–8. Citeseer, 2012.