Mehran University Research Journal of Engineering and Technology

https://doi.org/10.22581/muet1982.2555

2024, 43(2) 01-07

Path and cost optimization using genetic algorithm: an application perspective

Yousaf Ali [a, *], Syed Waqar Shah [a], Gul Muhammad [a], Wasim Ahmed Khan [b]

[a] *Department of Electrical Engineering University of Engineering and Technology Peshawar, Pakistan*

[b] *Faculty of Mechanical Engineering Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi, Pakistan*

[*] Corresponding author: Yousaf Ali, Email: yousafa.eec@uetpeshawar.edu.pk

Received: 12 September 2022, Accepted: 28 March 2024, Published: 01 April 2024

KEYWORDS

Genetic Algorithm

Optimization

Evolutionary

Travelling Salesman Problem

Mutation

ABSTRACT

Genetic Algorithm is an optimization technique inspired by nature. The technique has been used by scientists and engineers for real-life search and optimization problems. This work makes use of the genetic algorithm for the solution of the traveling salesman problem. This work focuses on real-time problems; the algorithm is used to find the optimum path for sales travelers inside the Khyber Pakhtunkhwa Province of Pakistan. The solution provides the shortest distance between the cities to be traveled and gives the optimal route. The coding is done in Python-3 and software is developed for the traveling salesmen, where the salesmen can select the cities, they want to travel, and the software will provide the optimal path and the distance.

1. Introduction

Genetic algorithms are bio-inspired stochastic search approaches based on biological evolutionary processes that belong to the class of evolutionary algorithms [1]. Genetic algorithms were first proposed by John Holland [2]. In nature, the species that are more adapt to the environment are more likely to survive and mate to reproduce the next generation. The reproduced generation will be much healthier and fitter than the previous one. A lot of research and applications have been done about Genetic Algorithms in a frequently cited book by Golberg [3]. Genetic Algorithms work with population of individuals called chromosomes that consist of some underlying parameters set codes called as Gene. The algorithm starts with the initial population of suitable solution to the problem. The population consists of individual chromosomes. The Genetic algorithm produces a new generation from the previous generation with a set of functions to exploit the search space, they are selection, crossover, and mutation functions [4]. The Selection function typically selects the most competitive individuals' chromosomes from the mating pool based on the survival strategy, The survival strategy is the fitness score of the individual. Crossover and mutation functions primarily provide better searching diversification in the solution space [5]. The Genetic algorithm is divided into the following steps, details are provided in the following paragraphs.

- Initialization of Population
- Fitness Calculations of each individual
- Selection of the fittest Individual
- Crossover
- Mutation
- Termination

1.1 Population Initialization

The genetic algorithm starts with the initialization of the population. The population is the set of chromosomes. Every individual chromosome is the possible solution to the problem addressed by the genetic algorithm [6]. The individual chromosome consists of a set of parameters called genes [7]. Genes are joined into a string to form a Chromosome (solution). In a genetic algorithm, the genes in the chromosomes are represented by numbers, alphabets, or binary bit strings. We will represent the gene by the decimal numbers starting from 1 to 24. The numbers (gene) will be selected to form the chromosome and hence the genes are encoded to form chromosomes. Fig. 1 shows how gene combines to form chromosomes and multiple chromosomes result in the population.

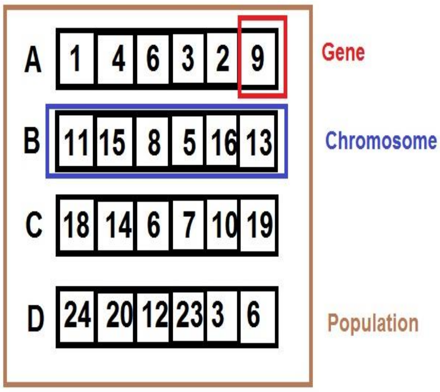


Fig 1. Genes and chromosomes

1.2 Fitness Calculation

Fitness indicates the rank of the individual chromosome to compete with other chromosomes [8]. Each individual chromosome is assigned a fitness score. The fitness function is used to calculate the fitness score. The probability that a chromosome will be selected for reproduction is based on its fitness score. The better the fitness score the higher is the probability of selection for reproduction.

1.3 Selection

The fitness individual will be selected to pass their genes to the next generation. The selected chromosomes are called parents.

1.4 Crossover

The crossover is the crucial phase in a genetic algorithm. The pairs of parents selected earlier are mated and a crossover point is randomly selected. The crossover point is selected to be 3 in the Fig. 2 as shown below.

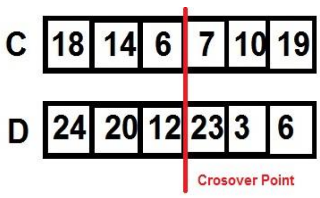


Fig. 2. Crossover

The genes are exchanged between two parents until the crossover point is reached (all the genes are exchanged). The newly generated individuals are called offspring or children. The new offspring are added to the population. Fig. 3 and Fig. 4. show the mating process and creation of new offspring.

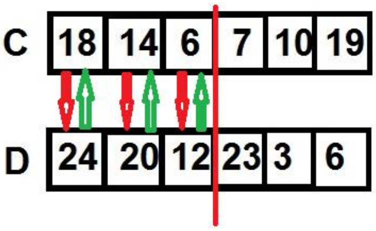


Fig. 3. Mating

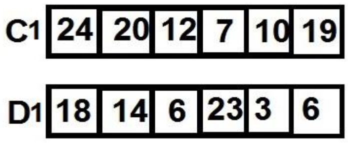


Fig. 4. New offspring

1.5 Mutation

The newly generated offspring are subjected to a mutation process. The mutation process refers to the flipping of the gene of the offspring with the gene of other chromosomes [9]. The mutation rate is normally very low. Mutation occurs to prevent premature convergence within the population [10].

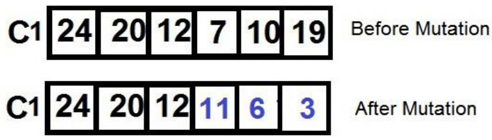


Fig. 5. Mutation

1.6 Termination

The processes of crossover, mating, and mutation will be repeated until the population has converged [11] the processes do not create new offspring with genes significantly different from the previously generated offspring. The algorithm terminates if the population has converged. After the population is converged, the genetic algorithm is said to have provided the best possible solution [12].

1.7 Problem statement

In several domains including but not limited to logistics, transportation, robotics, and computer network routing path and cost optimization plays a crucial role in increasing efficiency and better utilization of resources. Traditional optimization techniques often face challenges in handling complex and dynamic environment conditions. The objective of this research is to develop and apply the robust optimization technique the Genetic Algorithm for path and cost optimization for traveling salesman. Genetic Algorithm is a powerful optimization technique inspired by the process of natural selection and genetics. By iteratively evolving a population of potential solutions, Genetic Algorithm has demonstrated effectiveness in finding near-optimal solutions to complex optimization problems.

This research aims to address the following key objectives.

- Designing a genetic representation scheme to encode feasible paths and associated costs within the optimization framework.
- Developing fitness evaluation functions that accurately assess the quality of candidate solutions based on path optimality and cost minimization criteria.
- Implementing genetic operators, including selection, crossover, and mutation, to iteratively evolve the population towards better solutions.
- Evaluating the performance of the proposed Genetic Algorithm approach through extensive experimentation and comparison with existing optimization techniques.
- Validating the applicability and scalability of the optimization framework across diverse real-world scenarios and problem instances.

Khyber Pakhtunkhwa has been selected as a case, the study is general and applies to every geography.

2. Literature Review

The travelling salesman problem has received the attention of scientists and engineers over past few decades; various techniques have been proposed and applied to solve the path and cost optimization problem of traveling salesmen. Author in [13] use Ant colony optimization (ACO) with symbiotic organism search is used to solve the traveling salesman problem. The symbiotic organism search (SOS) is used to optimize the parameters of the Ant colony optimization algorithm. This helped them accelerate the convergence and get improved quality of solution. They observe that using SOS for optimization of ACO better results can be obtained than using ACO alone.in [14] Mohsen et al. explored the traveling salesman problem including scheduling the job time and called it as Traveling salesman problem with job time (TSPJT). He provided a mathematical mode for TSPJT using a commercial mixed integer solver. The dynamic programming approach has been used by Paul Bouman in [15] for the path optimization of drones. They were able to demonstrate that the dynamic programming approach can provide solutions to larger and more complex problems than the mathematical programming approaches.

Chen Yang [16] provided a solution to the traveling salesman problem using a multi-agent system. The multi-agent system partitioned the cities into *n* groups and assigned an agent to each group. The agent was used to find the shortest path within the group. The author’s in [17] used a decomposition algorithm for a consistent traveling salesman problem. The consistent TSP provides a minimum cost route to be followed by a single vehicle. The algorithm they proposed decomposes the complex problem into a sequence of small single period TSP problems. The algorithm was competitive to provide solutions to 100 customers over five period horizons. The artificial Bee Colony algorithm is used by Venkatesh et al. in [18] for solving the traveling salesman problem. The author’s demonstrated that the solution is efficient in terms of quality and computation resources. Further, the author’s in [19] used the branch and bound technique, particle swarm optimization in [20] the cutting plan technique in [21], 2-opt was used by the author’s in [22]. The other techniques include Ant Colony optimization [23], artificial neural networks [24], and Genetic algorithm by author’s in [25]. Stephan et al. in [26] used dynamic Gaussian process regression to find the best route. The author finds the solution less expensive in terms of computational cost. The African Buffalo optimization has been used by author’s in [27] to solve the traveling salesman problem, the author’s state that the African buffalo displays uncommon intelligence, organizational skills, and exceptional navigational capabilities. The author’s modeled the algorithm and solved more than 30 symmetric salesman problems with high accuracy. Nicola Bisozzie et al. in [28] solved the traveling salesman problem for the bus transit system for the city of Castallenza in Italy. This practical approach also makes use of a branch and bound algorithm for optimal route calculation. Most of the algorithms mentioned above are computationally expensive. Some algorithms depend upon another tool for parameter optimization. Genetic algorithms are preferred due to their effectiveness in solving complex problems and being computationally less expensive.

3. Methodology

The methodology starts with the selection of cities by the traveling salesman as the gene for the genetic algorithm. The following Fig. shows how the salesman selects the cities from the user interface. The initial population is created by the random process that is the possible routes from the starting city to the last city to be traveled. The number of initial populations is defined to be 1000; one thousand possible routes are generated initially. Each city is represented by a number ranging from 1 to 28, 1 corresponds to the first city in the selection user interface and 28 to the last city in the user interface as shown in the Fig. 6a and Fig. 6b.

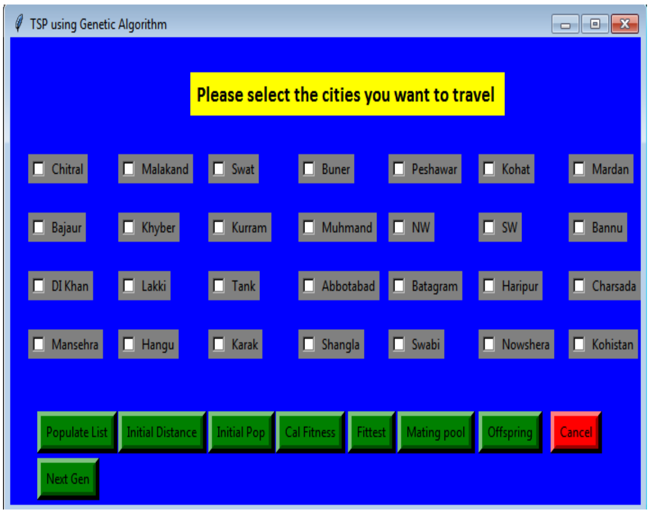


Fig. 6a. User Interface for Travelers

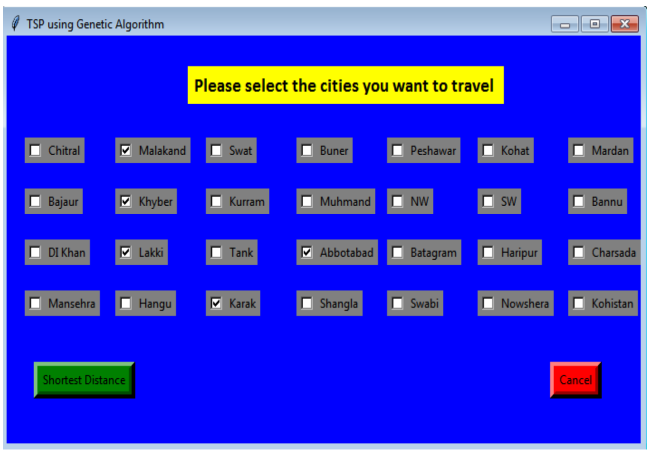


Fig. 6b. Final user Interface for Travelers

The finesses for all the initially generated routes are calculated. Fitness is the distance in kilometers from the starting city to the end city, the shorter the distance the better is the fitness. The next step is to select the fittest routes depending upon the shortest distance values. Twenty fittest paths are selected to be the elite of the fitness function. The mating process is started using the chromosomes i-e the cities in the fittest class and hence the new offspring are generated. The mutation is performed as explained in the introduction section and a new population is generated. The steps continue to be repeated until the population is converged. The following flow Fig. 7 gives the genetic algorithm process.

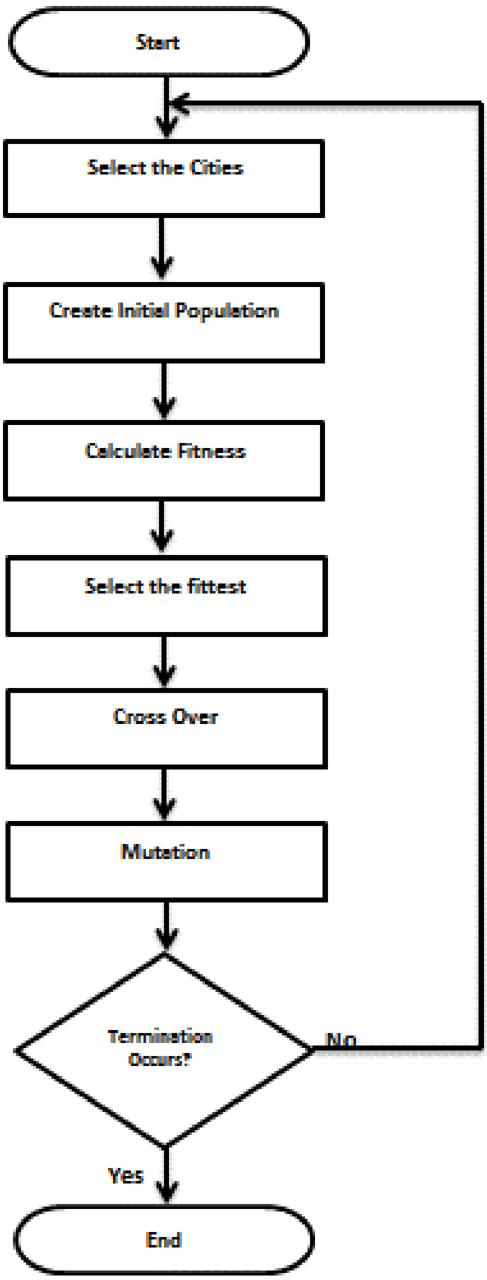
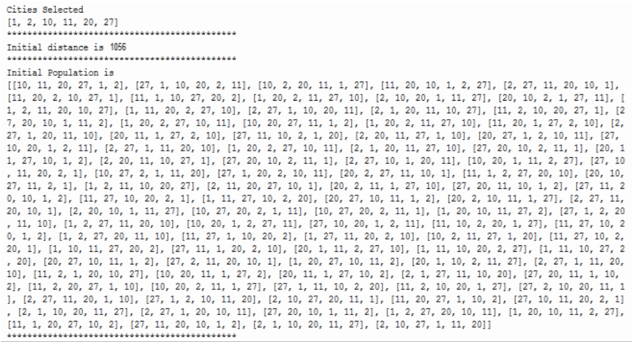


Fig. 7. Flowchart of the Algorithm

4. Results

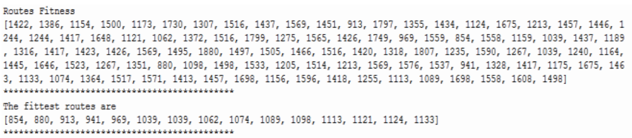
The algorithm is coded in Python language and graphic user interface is also provided for the user to select among the cities they want to travel. The result of each step of the genetic algorithm for the travelling salesman problem is provided in the following figures.



```
Cities Selected
[1, 2, 10, 11, 20, 27]
*********************************************
Initial distance is 1056
*********************************************
Initial Population is
[[10, 11, 20, 27, 1, 2], [27, 1, 10, 20, 2, 11], [10, 2, 20, 11, 1, 27], [11, 20, 10, 1, 2, 27], [2, 27, 11, 20, 10, 1],
[11, 20, 2, 10, 27, 1], [11, 1, 10, 27, 20, 2], [1, 20, 2, 11, 27, 10], [2, 10, 20, 1, 11, 27], [20, 10, 2, 1, 27, 11], [
1, 2, 11, 20, 10, 27], [1, 11, 20, 2, 27, 10], [2, 27, 1, 10, 20, 11], [2, 1, 20, 11, 10, 27], [11, 2, 10, 20, 27, 1], [2
7, 20, 10, 1, 11, 2], [1, 20, 2, 27, 10, 11], [10, 20, 27, 11, 1, 2], [1, 20, 2, 11, 27, 10], [11, 20, 1, 27, 2, 10], [2,
27, 1, 20, 11, 10], [20, 11, 1, 27, 2, 10], [27, 11, 10, 2, 1, 20], [2, 20, 11, 27, 1, 10], [20, 27, 1, 2, 10, 11], [27,
10, 20, 1, 2, 11], [2, 27, 1, 11, 20, 10], [1, 20, 2, 27, 10, 11], [2, 1, 20, 11, 27, 10], [27, 20, 10, 2, 11, 1], [20, 1
1, 27, 10, 1, 2], [2, 20, 11, 10, 27, 1], [27, 20, 10, 2, 11, 1], [2, 27, 10, 1, 20, 11], [10, 20, 1, 11, 2, 27], [27, 10
, 11, 20, 2, 1], [10, 27, 2, 1, 11, 20], [27, 1, 20, 2, 10, 11], [20, 2, 27, 11, 10, 1], [11, 1, 2, 27, 20, 10], [20, 10,
27, 11, 2, 1], [1, 2, 11, 10, 20, 27], [2, 11, 20, 27, 10, 1], [20, 2, 11, 1, 27, 10], [27, 20, 11, 10, 1, 2], [27, 11, 2
0, 10, 1, 2], [11, 27, 10, 20, 2, 1], [1, 11, 27, 10, 2, 20], [20, 27, 10, 11, 1, 2], [20, 2, 10, 11, 1, 27], [2, 27, 11,
20, 10, 1], [2, 20, 10, 1, 11, 27], [10, 27, 20, 2, 1, 11], [10, 27, 20, 2, 11, 1], [1, 20, 10, 11, 27, 2], [27, 1, 2, 20
, 11, 10], [1, 2, 27, 11, 20, 10], [10, 20, 1, 2, 27, 11], [27, 10, 20, 1, 2, 11], [11, 10, 2, 20, 1, 27], [11, 27, 10, 2
0, 1, 2], [1, 2, 27, 20, 11, 10], [11, 27, 1, 10, 20, 2], [1, 27, 11, 20, 2, 10], [10, 2, 11, 27, 1, 20], [11, 27, 10, 2,
20, 1], [1, 10, 11, 27, 20, 2], [27, 11, 1, 20, 2, 10], [20, 1, 11, 2, 27, 10], [1, 11, 10, 20, 2, 27], [1, 11, 10, 27, 2
, 20], [20, 27, 10, 11, 1, 2], [27, 2, 11, 20, 10, 1], [1, 20, 27, 10, 11, 2], [20, 1, 10, 2, 11, 27], [2, 27, 1, 11, 20,
10], [11, 2, 1, 20, 10, 27], [10, 20, 11, 1, 27, 2], [20, 11, 1, 27, 10, 2], [2, 1, 27, 11, 10, 20], [27, 20, 11, 1, 10,
2], [11, 2, 20, 27, 1, 10], [10, 20, 2, 11, 1, 27], [27, 1, 11, 10, 2, 20], [11, 2, 10, 20, 1, 27], [27, 2, 10, 20, 11, 1
], [2, 27, 11, 20, 1, 10], [27, 1, 2, 10, 11, 20], [2, 10, 27, 20, 11, 1], [11, 20, 27, 1, 10, 2], [27, 10, 11, 20, 2, 1]
, [2, 1, 10, 20, 11, 27], [2, 27, 1, 20, 10, 11], [27, 20, 10, 1, 11, 2], [1, 2, 27, 20, 10, 11], [1, 20, 10, 11, 2, 27],
[11, 1, 20, 27, 10, 2], [27, 11, 20, 10, 1, 2], [2, 1, 10, 20, 11, 27], [2, 10, 27, 1, 11, 20]]
*********************************************
```

Fig. 8. Cities and Initial Population

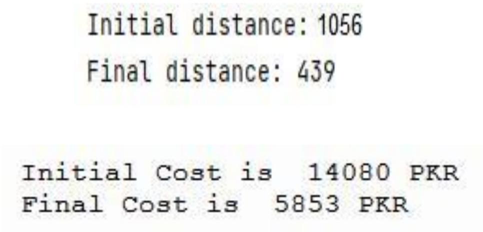
Fig. 8 above shows the selected cities and the number associated with the cities is shown. The Initial Population and initial distance are calculated and shown in the Fig. above. All the fitness calculation, selection and mutation processes are performed, and the optimum path is generated as shown in Fig. 9. below.



```
Routes Fitness
[1422, 1386, 1154, 1500, 1173, 1730, 1307, 1516, 1437, 1569, 1451, 913, 1797, 1355, 1434, 1124, 1675, 1213, 1457, 1446, 1
244, 1244, 1417, 1648, 1121, 1062, 1372, 1516, 1799, 1275, 1565, 1426, 1749, 969, 1559, 854, 1558, 1159, 1039, 1437, 1189
, 1316, 1417, 1423, 1426, 1569, 1495, 1880, 1497, 1505, 1466, 1516, 1420, 1318, 1807, 1235, 1590, 1267, 1039, 1240, 1164,
1445, 1646, 1523, 1267, 1351, 880, 1098, 1498, 1533, 1205, 1514, 1213, 1569, 1576, 1537, 941, 1328, 1417, 1175, 1675, 146
3, 1133, 1074, 1364, 1517, 1571, 1413, 1457, 1698, 1156, 1596, 1418, 1255, 1113, 1089, 1698, 1558, 1608, 1498]
*********************************************
The fittest routes are
[854, 880, 913, 941, 969, 1039, 1039, 1062, 1074, 1089, 1098, 1113, 1121, 1124, 1133]
*********************************************
```

Fig. 9. Fitness Calculations

The result with initial and final distances and cost in PKR (Pakistani Rupees) is shown in the Fig. 10. The number of generations vs. final distance is shown in Fig. 11 below.



```
Initial distance: 1056
Final distance: 439

Initial Cost is  14080 PKR
Final Cost is  5853 PKR
```

Fig. 10. Initial and Final Results

Fig. 10 above shows that the initial distance at the first generation was 1056km and the algorithm converges at the 500th generation with a final distance of 439km. If the fuel price per liter is 237 PKR and the fuel economy of the vehicle is 18km/l then the initial and final costs are 14,080 PKR and 5853 PKR. respectively. This indicates the path and cost have been optimized by factors of 2.4.

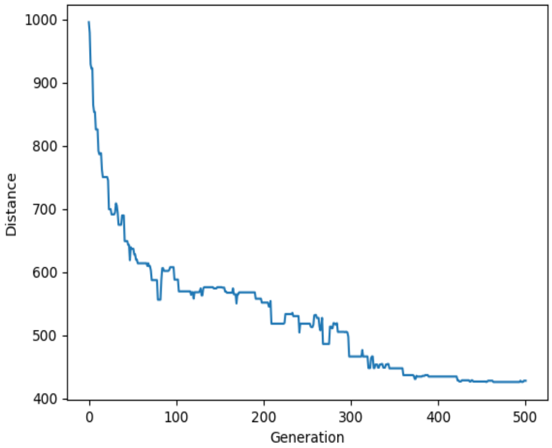


Fig. 11. Distances Evolving VS Generation

5. Conclusion

Genetic algorithm is used to get the optimized and economic path for the traveling salesman. This work focuses on the practical approach to get the shortest path for the travelers of the Khyber Pakhtunkhwa province of Pakistan. The solution provided in the paper was able to save time and cost at a factor of 2.4. The developed Graphic user interface can be used by the courier companies to get the optimized path and the cost of travel for the optimized path. The problem is solved using Python programming and the results are generated compared and presented in this work. The study can be extended to other geographies as Khyber Pakhtunkhwa has been used as a test case. Further, the solution can be deployed for logistics, transportation, mobile robots in industries, drone applications, and routing of large circuit boards.

6. References

[1] T. C. Dwivedi, S. Saxena, and P. Agrawal, "Travelling salesman problem using genetic algorithm," Int. J. Comput. Appl. (IJCA), pp. 25-30, 2012.

[2] J. H. Holland, Adaptation in Natural and Artificial Systems: “An introductory analysis with applications to biology, control, and artificial intelligence” University of Michigan Press, Oxford, UK, 1975.

[3] D. E. Golberg, "Genetic algorithms in search, optimization and machine learning," Addison-Wesley Publishing Company, 1989.

[4] S. Kirkpatrick and G. Toulouse, "Configuration space analysis of travelling salesman problems," Le Journal de Physique, vol. 46, no. 8, pp. 1277-1292, 1985.

[5] D. L. Applegate, "The traveling salesman problem: a computational study," Princeton University Press, Princeton Series in Applied Mathematics, 2006.

[6] S. S. Juneja, P. Saraswat, K. Singh, J. Sharma, R. Majumdar, and S. Chowdhary, "Travelling salesman problem optimization using genetic algorithm," 2019 Amity International Conference on Artificial Intelligence (AICAI), pp. 264-268, 2019. doi: 10.1109/AICAI.2019.8701246.

[7] S. M. Gustafson, W. Langdon, and J. Koza, "Bibliography on genetic programming, 11, 2021," [Online]. Available: https://liinwww.ira.uka.de/bibliography/Ai/genetic.programming.html.

[8] H. Cai, "User preference adaptive fitness of interactive genetic algorithm based ceramic disk pattern generation method," IEEE Access, vol. 8, pp. 95978-95986, 2020. doi: 10.1109/ACCESS.2020.2995765.

[9] M. K. Vassil S. markov and S. Penev, "Genetic operators' crossover and mutation in solving the TSP problem," International Conference on Computer Systems and Technologies, 2005.

[10] K. N. Kumar and R. Kumar, "A genetic algorithm approach to study travelling salesman problem," Journal of Global Research in Computer Science, vol. 3, no. 3, 2012.

[11] D. S. Johnson and L. A. McGeoch, "The traveling salesman problem: A case study in local optimization," Local search in combinatorial optimization, vol. 1, pp. 215-310, John Wiley and Sons, London.

[12] J. R. Koza, "Genetic programming: on the programming of computers by means of natural selection," vol. 1, MIT Press.

[13] Y. Wang and Z. Han, "Ant colony optimization for traveling salesman problem based on parameters optimization," Applied Soft Computing, vol. 107, p. 107439, 2021.

[14] M. Mosayebi, M. Sodhi, and T. A. Wettergren, "The traveling salesman problem with job-times (tspj)," Computers & Operations Research, vol. 129, p. 105226, 2021.

[15] P. Bouman, N. Agatz, and M. Schmidt, "Dynamic programming approaches for the traveling salesman problem with drone," Networks, vol. 72, no. 4, pp. 528-542, 2018.

[16]. C. Yang and K. Y. Szeto, "Solving the traveling salesman problem with a multi-agent system," 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 2019, pp. 158-165, doi: 10.1109/CEC.2019.8789895.

[17] A. Subramanyam and C. E. Gounaris, "A decomposition algorithm for the consistent traveling salesman problem with vehicle idling," Transportation Science, vol. 52, no. 2, pp. 386-401, 2018.

[18] V. Pandiri and A. Singh, "A swarm intelligence approach for the colored traveling salesman problem," Applied Intelligence, vol. 48, pp. 4412-4428, 2018.

[19] G. Finke, A. Claus, and E. Gunn, "A two-commodity network flow approach to the traveling salesman problem," vol. 41, pp. 167-178.

[20] P. Miliotis, "Using cutting planes to solve the symmetric travelling salesman problem," Mathematical Programming, vol. 15, no. 1, pp. 177-188, 1978.

[21] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," Operations Research, vol. 21, pp. 498-516, 1973.

[22] J. Kennedy, R. C. Eberhart, and Y. Shi, "Swarm intelligence," Morgan Kaufmann Publishers, Inc., San Francisco, CA, USA, 2001.

[23] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 53-66, 1997.

[24] S. Bhide, N. John, and M. R. Kabuka, "A boolean neural network approach for the traveling salesman problem," IEEE Transactions on Computers, vol. 42, no. 10, pp. 1271-1278, 1993.

[25] Z. Michalewicz, "Genetic Algorithms + Data Structures = Evolution Programs," Springer, New York, NY, USA, 1996.

[26] S. M. Akandwanaho, A. O. Adewumi, and A. A. Adebiyi, "Solving dynamic traveling salesman problem using dynamic gaussian process regression," Journal of Applied Mathematics, vol. 2014, article ID 818529, pp. 1-10, 2014. https://doi.org/10.1155/2014/818529

[27] J. B. Odili and M. N. M. Kahar, "Solving the traveling salesman’s problem using the African buffalo optimization," Computational Intelligence and Neuroscience, vol. 2016, pp. 1-12, 2016. https://doi.org/10.1155/2016/1510256.

[28] N. Besozzi, L. Ruschetti, C. Rossignoli, and F. Strozzi, "The traveling salesman game for cost allocation: the case study of the bus service in castellanza," Gametheory, vol. 2014, pp. 1-6, 2014. https://doi.org/10.1155/2014/696945.

