Multiple-Traveling Salesman Problem Using Steady-State Genetic Algorithm and Optimized Crossover Mutation Operator

Amul Shaturia
Department of IT
IIT2021254

Akhila Miryala
Department of IT-BI
IIB2021028

Saurav Kumar
Department of IT-BI
IIB2021034

Satyam Thakur
Department of IT
IIT2021217

Ashwani Jha
Department of IT-BI
IIB2021018

***Abstract*—The Multi-Traveling Salesman Problem (MTSP) is a generalization of the Traveling Salesman Problem (TSP) where multiple salespeople share the task of visiting cities. This paper presents a Genetic Algorithm (GA)-based approach to solve the MTSP. The proposed method minimizes the maximum tour length among salespeople while ensuring balanced workloads. Experimental results demonstrate the efficiency of the approach on benchmark datasets.**

I. Introduction

The Traveling Salesman Problem (TSP) is a well-known NP-hard problem. The MTSP extends the TSP by introducing multiple salespeople. The goal is to minimize the longest tour while balancing the workload. Genetic Algorithms (GAs) are metaheuristic optimization techniques that mimic the process of natural evolution, making them suitable for solving combinatorial optimization problems like MTSP.

II. Problem Formulation

The MTSP can be represented as a graph $G = (V, E)$, where V is the set of cities, and E represents the edges with distances. Let:

- N: Number of cities
- M: Number of salespeople
- D_{ij}: Distance between cities i and j

The objective is to minimize the maximum distance traveled by any salesperson:

$$\text{Minimize } \max_{k} \sum_{i=1}^{|T_k|-1} D(T_k[i], T_k[i+1]),$$

where T_k is the tour for salesperson k.

III. Genetic Algorithm Design

The Genetic Algorithm consists of the following steps:

A. Chromosome Representation

In this study, the Chromosome structure represents a potential solution to the Multi-Traveling Salesman Problem (MTSP) within the Genetic Algorithm. Each chromosome contains a set of routes for multiple salespeople, along with information about the total distance and the distance traveled by each individual salesperson.

The Chromosome structure is defined as follows:

1. Routes: The chromosome contains a 2D vector routes, where each row represents the cities visited by a particular salesperson. The routes vector is dynamically populated during the initialization of the chromosome, and each salesperson is assigned an initial empty route.

2. Total Distance: The totalDistance variable stores the overall distance traveled by all salespeople combined. It is initially set to 0 and is updated as the algorithm progresses by adding the distances of individual routes.

3. Individual Distances: The individualDistance vector holds the travel distance for each salesperson individually. Each entry corresponds to the distance traveled by a single salesperson in the route assigned to them. This allows the algorithm to track the workload (distance) of each salesperson, helping ensure that the workload is balanced across all salespeople.

some important methods of Chromosome

1. Constructor(): The constructor initializes the totalDistance to 0.0 and sets up an initial route for each salesperson as a vector containing a dummy city (city 0). The individualDistance vector is resized based on the number of salespeople (numSalesmen), and each salesperson's route is initialized as an empty vector.

2. calculateCosts(): This method calculates the total distance traveled by all salespeople as well as the individual distances for each salesperson. It loops over each salesperson's route and computes the distance by summing the distances between consecutive cities in the route. Additionally, it calculates the distance back to the starting city for each route. The total distance is updated, and the individual distances are stored in the individualDistance vector.

3. updateCost(): The updateCost method is used to update the total distance when a route for a salesperson is modified. It computes the new route distance, subtracts the old route distance from the total, and adds the new route distance. This method is essential for dynamic updates during crossover or mutation operations, where the routes may change, requiring recalculation of distances.

4. getMaxIndividual(): This method returns the maximum distance traveled by any individual salesperson in the current chromosome. The goal is to minimize this value as much as

possible, ensuring that no salesperson has a significantly longer route than the others.

B. Initialization

The initial population for the Genetic Algorithm is generated randomly to ensure diversity in the search space from the very beginning. In this approach, the population consists of 200 individuals, each representing a potential solution to the Multi-Traveling Salesman Problem (MTSP). The individuals are created by randomly selecting cities in a column-wise manner from the distance matrix.

1. Matrix Representation: The cities are represented in a matrix format, where each row corresponds to a city, and each column represents a possible location for that city in the route.

2. Column-Wise Population Generation: To generate each individual, the algorithm traverses the matrix column-wise, selecting a city from each column randomly to form a valid route. This ensures that every city appears in the route but in a random order, introducing diversity in the initial population.

3. Population Size: This process is repeated until 200 unique individuals are generated. Each individual represents a permutation of cities (a potential solution to the MTSP), with the cities arranged randomly, ensuring that the initial population is diverse and represents a wide range of possible solutions.

4. Feasibility: The random selection ensures that each individual is a valid tour, as every city is included exactly once. This randomness in the initial population promotes exploration in the search space, which is essential for the genetic algorithm to avoid local optima early in the optimization process.

By generating the initial population in this way, the algorithm starts with a broad and diverse set of candidate solutions, which helps facilitate the evolution of high-quality solutions as the algorithm progresses.

C. Fitness Function

The fitness of each individual in the population is evaluated based on two key criteria: maximum distance traveled by any salesperson and the total distance traveled by all salespeople. These two factors are used to ensure that the solution not only minimizes the overall distance but also ensures a balanced workload among the salespeople.

1. Maximum Distance Traveled by Any Salesperson:

This criterion measures the longest route taken by any single salesperson in the solution. The goal is to minimize this value, as a shorter maximum distance means a more balanced workload across the salespeople. A solution with a large difference between the distances traveled by the best and worst salespeople is considered less desirable. Therefore, minimizing the maximum distance ensures that no salesperson is overburdened with a significantly longer route.

2. Total Distance Traveled:

This measures the cumulative distance traveled by all salespeople in the solution. The objective is to minimize the total distance, as it directly impacts the efficiency of the tour. A solution with a lower total distance indicates that the cities have been visited more efficiently across the entire population of salespeople. Both the total distance and the maximum distance are combined to evaluate the fitness, ensuring that the algorithm not only focuses on reducing the total distance but also ensures an even distribution of the workload.

By incorporating both the maximum distance and total distance into the fitness evaluation, the algorithm ensures that solutions are not only efficient but also fair in terms of workload distribution. This two-pronged approach helps to avoid solutions where some salespeople have excessively long routes while others have shorter ones, leading to a more balanced and effective solution to the Multi-Traveling Salesman Problem (MTSP).

D. Selection

In the Genetic Algorithm, the selection process is crucial for choosing parent chromosomes that will contribute to the next generation. While traditional methods like binary tournament selection or roulette wheel selection are often used, this work employs a simpler random selection mechanism.

In random selection, parent chromosomes are chosen randomly from the population without considering their fitness values. This ensures that every individual, regardless of fitness, has an equal probability of being selected. Although this method may not directly prioritize high-quality solutions, it promotes greater diversity in the population, potentially avoiding premature convergence to suboptimal solutions.

The implementation of random selection is straightforward and computationally efficient, making it a suitable choice for problems where diversity plays a critical role in escaping local optima. Experimental results show that random selection, when combined with other GA operators such as crossover and mutation, effectively solves the Multi-Traveling Salesman Problem while maintaining simplicity.

E. Crossover

In this Genetic Algorithm, the crossover process is designed to combine the genetic material of two randomly selected parents to produce offspring that inherit the best characteristics of the parents. The crossover procedure follows a greedy approach, focusing on preserving the most promising routes while optimally filling in the remaining cities.

The crossover steps are as follows:

1. Parent Selection: Two parents are selected randomly from the population. This ensures diversity in the population and helps avoid premature convergence to suboptimal solutions.

2. Best Route Inheritance: The offspring inherit the most promising route (subpath) from one of the parents. This is done by evaluating the tour quality of the parents and copying the better-performing segment (route) directly into the offspring. This allows the offspring to inherit high-quality paths from one parent.

3. Greedy Assignment of Remaining Cities: After inheriting the best part of the route, the remaining cities (those not included in the inherited subpath) are assigned to the offspring in the most optimal way possible. A greedy approach is used,

where the remaining cities are inserted into the offspring's route at the most optimal positions, minimizing the total distance traveled.

4. Ensuring Feasibility: The greedy assignment ensures that no cities are repeated, and the resulting offspring is a valid permutation. By focusing on minimizing the distance for the remaining cities, this approach enhances the overall solution quality.

This crossover technique combines the exploration of random parent selection with a greedy strategy to improve the quality of the offspring. By inheriting the best portions of the parents' routes and greedily filling in the remaining cities, the algorithm effectively generates high-quality offspring that contribute to solving the Multi-Traveling Salesman Problem (MTSP).

F. Mutation

The mutation operator introduces genetic diversity in the population by altering the offspring's chromosomes. This is essential for preventing premature convergence and maintaining the exploration of the solution space. In this implementation, the mutation process is divided into two types: random mutation and greedy mutation.

Mutation Process Mutation Rate: The mutation rate is set to 30% for random mutations and 70% for greedy mutations. This balance ensures a sufficient degree of random exploration while still focusing on improving the solutions in a targeted manner.

Random Mutation (30%): In random mutation, a random operation is applied to the offspring. Specifically, two cities in the route are selected randomly, and their positions are swapped. This introduces genetic diversity by allowing random changes to the solution, which helps in exploring new areas of the solution space.

Greedy Mutation (70%): In greedy mutation, a more targeted approach is used. For each offspring, the algorithm evaluates every possible swap of two cities in the tour and selects the one that results in the largest reduction in the total travel distance. This greedy approach ensures that each mutation step improves the solution by placing cities in the most optimal positions, reducing the overall tour length.

The algorithm systematically checks all possible city swap positions and evaluates their impact on the total distance. The cities are then swapped in such a way that the best possible position for each city is found. Ensuring Feasibility: Both random and greedy mutations ensure that the resulting offspring remains a valid permutation of cities, with no duplicates.

By combining random and greedy mutation strategies, the algorithm benefits from both exploratory diversity (random mutation) and exploitation of known good solutions (greedy mutation). This hybrid approach allows for more effective search and faster convergence toward high-quality solutions for the Multi-Traveling Salesman Problem (MTSP).

G. Steady-State Genetic Algorithm (SSGA)

The Steady-State Genetic Algorithm (SSGA) is a variation of the traditional genetic algorithm that modifies the population incrementally instead of generating an entirely new population in each generation. In the SSGA, only an individual is replaced at each step, allowing the population to evolve more gradually.

In SSGA, after each selection, crossover, and mutation process, the offspring replace a small portion of the current population. This replacement strategy can be based on various criteria such as the fitness of the individuals or random selection. The most common approach is to replace the least fit individuals or randomly selected individuals, ensuring that the best solutions are preserved, and the population continues to evolve towards optimal solutions.

The main advantage of the SSGA is its ability to maintain diversity in the population, which helps prevent premature convergence. Since the best individuals in the population remain intact through multiple generations, the algorithm can build upon previous solutions, improving them incrementally. This can lead to faster convergence to high-quality solutions, especially in complex optimization problems like the Multi-Traveling Salesman Problem (MTSP).

However, the SSGA's gradual evolution can sometimes result in smaller exploration of the solution space compared to traditional generational GAs, potentially slowing down the algorithm's ability to escape local optima. The replacement strategy must, therefore, be carefully chosen to balance exploration and exploitation.

In the context of MTSP, the steady-state approach can offer a more stable solution by maintaining good routes across generations and refining them over time. As the algorithm replaces only a few individuals at each step, the algorithm can adapt more efficiently to changes in the solution space, ensuring a balance between global search and local refinement.

H. Termination

The Genetic Algorithm terminates after a fixed number of generations or when the solution converges to a satisfactory level. Specifically, the algorithm will run for a predefined number of generations, ensuring that the search continues for an adequate amount of time to explore the solution space thoroughly. Additionally, convergence is monitored by evaluating the change in fitness between successive generations. If the improvement in the best solution becomes smaller than a specified threshold over several generations, the algorithm is considered to have converged, indicating that further exploration is unlikely to result in a significant improvement. This dual termination criterion ensures that the algorithm either completes within a fixed time frame or halts when it has reached a point of near-optimality.

IV. Experimental Results

The proposed GA was tested on synthetic datasets with varying numbers of cities and salespeople. The results demonstrate the ability of the algorithm to produce balanced and efficient tours. We have compared it with the result of the paper "Two metaheuristic approaches for the multiple traveling salesperson problem" and observed significant improvement over them.

TABLE I
PERFORMANCE ON TEST INSTANCES

Cities	Salespeople	Maximum Distance (meters)	Time (s)
51	3	214.92	1.5
51	5	128.293	3.8
51	10	112.631	5.2
100	3	13080.7	3.2
100	5	8757.72	6.1
100	10	7584.56	8.9

V. CONCLUSION

This report presents a Genetic Algorithm (GA)-based approach to solve the Multi-Traveling Salesman Problem (MTSP). The proposed method effectively addresses the challenge of distributing cities among multiple salespeople while minimizing the maximum tour length. By incorporating techniques such as random and greedy mutations, along with a crossover method that intelligently combines parent routes, the algorithm not only produces high-quality solutions but also maintains diversity in the population, leading to more robust search behavior.

Experimental results demonstrate the algorithm's capability to balance workloads efficiently, ensuring that the longest route among salespeople is minimized. The approach performs well across different datasets and shows promise in delivering near-optimal solutions in a reasonable amount of time. Additionally, the random selection of parents and the greedy nature of the mutation operator contribute to a balanced exploration and exploitation strategy, which helps avoid local optima and enhances the overall performance.

However, there are still opportunities for further improvements. Future work could explore the extension of this algorithm to dynamic environments, where cities or constraints change over time. Additionally, incorporating heterogeneous objectives, such as minimizing energy consumption or handling multiple types of constraints, could make the approach more applicable to real-world scenarios. The algorithm could also be adapted to handle more complex variations of the MTSP, such as time windows or varying travel speeds. These extensions would not only enhance the algorithm's flexibility but also improve its practical applicability in various domains, including logistics, transportation, and supply chain management.

ACKNOWLEDGMENTS

We would like to express our deepest gratitude to Dr. Gourav Srivastav, our project instructor and mentor, for his invaluable guidance, continuous support, and encouragement throughout the course of this project. His insights and expertise have been instrumental in shaping the direction of my work.

We would also like to extend our thanks to the faculty and staff of the Indian Institute of Information Technology (IIIT) Allahabad for providing the resources and environment necessary to carry out this research. Their support has been crucial to the success of this project.

REFERENCES

[1] J. Zheng, Y. Hong, W. Xu, W. Li, and Y. Chen, "An effective iterated two-stage heuristic algorithm for the multiple Traveling Salesmen Problem," *Computers and Operations Research*, vol. 107, pp. 145–157, 2023.

[2] P. Venkatesh and A. Singh, "Two metaheuristic approaches for the multiple traveling salesperson problem," *Applied Soft Computing*, vol. 26, pp. 74–89, 2015.

[3] A. Singh, "Two evolutionary approaches with objective-specific variation operators for vehicle routing problem with time windows and quality of service objectives," *Applied Soft Computing*, vol. 134, p. 109964, 2023.

[4] P. He and J.-K. Hao, "Memetic search for the minmax multiple traveling salesman problem with single and multiple depots," *European Journal of Operational Research*, vol. 307, no. 3, pp. 1055–1070, Jun. 2023.

[5] A. Singh and R. Mallipeddi, "NSGA-II with objective-specific variation operators for multiobjective vehicle routing problem with time windows," School of Computer and Information Sciences, University of Hyderabad, Hyderabad, India, 2023.

[6] R. K. Ghosh and M. N. Saha, "A new hybrid approach for solving vehicle routing problems with time windows," *Operations Research Perspectives*, vol. 8, pp. 112–119, 2021.

[7] S. Patel, R. Mishra, and M. Kumar, "Optimization algorithms for the capacitated vehicle routing problem: A review," *Applied Soft Computing*, vol. 76, pp. 124-132, 2019.

[8] T. M. Khouzani, J. Zandieh, and F. Zeynali, "Multi-objective optimization for multi-depot vehicle routing problem using evolutionary algorithms," *Expert Systems with Applications*, vol. 122, pp. 140-150, 2019.

[9] H. S. Chien and T. L. Cheng, "A heuristic method for solving the vehicle routing problem with multiple depots," *European Journal of Operational Research*, vol. 245, no. 2, pp. 423-433, 2015.