Software Engineering Department
ORT Braude College

Capstone Project Phase A – 61998

Enhancing Electric Vehicle Routing Efficiency Through K-Means Clustering and Genetic Algorithm Optimization

24-1-R-20

Supervisor: Miri Weiss-Cohen

Team members: Yovel Gabay, Tal Gaon

Git link: https://github.com/Yovelgabay/Electric-Vehicle-Routing.git

Table of Contents

Abstract
Efficient route planning for electric vehicles (EVs) presents a critical challenge in the transportation sector, particularly with the rising concerns over greenhouse gas emissions and energy sustainability. This project addresses the challenge of determining the most efficient route from point A to point B, considering stops at charging stations by integrating K-Means clustering and Genetic Algorithm (GA) optimization techniques to enhance routing efficiency. Traditional methods often overlook real-world challenges such as queuing delays at charging stations and computational complexity. Our approach tackles these issues through two main strategies: K-Means clustering to partition charging stations into clusters based on geographical proximity and then Genetic Algorithm optimization to minimize travel distance by strategically selecting routes with minimal charging stops. By integrating these techniques, our approach ensures efficient charging and minimizes overall travel distance, addressing the unique challenges posed by EVs. This research contributes to advancing sustainable transportation solutions, supporting the transition towards a greener and more efficient transportation sector.

1. Introduction
The vehicle routing problem (VRP) poses a significant challenge in optimizing delivery routes from a depot to scattered customers while maintaining various constraints [1]. Dantzig and Ramser defined the problem for the first time in 1959 as the Truck Dispatching Problem [2]. VRP is a generalization of the well-known traveling salesman problem (TSP) that seeks the shortest route to visit all customers. VRP encompasses various problem variants like capacitated VRP (CVRP), VRP with time windows (VRPTW), and multidepot VRP (MDVRP), each tailored to specific real-world scenarios and constraints. [3]

In the context of modern transportation, The transportation sector faces a critical crossroads, with conventional vehicles contributing 75% of its greenhouse gas emissions (IEA, 2022) [4] This alarming reality concerns climate change, resource depletion, and the future of our planet. While electric vehicles (EVs) have emerged as a potential solution, their environmental impact remains under examination. Although they are known for being emissions-free during operation, their dependence on electricity, often generated from fossil fuels, raises questions about their overall environmental benefit. Their rapid adoption could also strain energy grids, further complicating the equation. Despite these challenges, EVs present undeniable advantages. Government incentives and regulations actively encourage their use, driving a rapidly growing market with a 72% annual average increase in global stock from 2017 to 2021 (IEA, 2022) [4]. However, the environmental benefits of EVs are contingent on efficient route planning and utilization.

This introduces the Electric Vehicle Routing Problem (EVRP), an extension of VRP tailored to the unique requirements of EVs [5]. EVRP aims to optimize routes for

electric vehicles while considering charging constraints, energy consumption, and other specific factors like travel distance, time, and charging station logistics.
In our project, we focus on two main objectives:

1. Incorporating Real-World Charging Constraints: Many studies often assume that recharging happens immediately when EVs arrive at charging stations. However, real-world scenarios often involve limited charger availability, resulting in queuing before recharging. This waiting time can significantly impact routing decisions.[6] We aim to provide more realistic and practical routing solutions by integrating queuing times into our model.
2. Enhancing Computational Efficiency through Clustering: To reduce computation time, we use clustering techniques to limit the number of charging stations considered in the graph. This approach aims to increase the speed of computation while maintaining the quality of the solution, enabling quicker and more efficient route planning. [7]

Based on the EVRP problem, our problem is described as follows: Given a start point A and destination point B, **each EV can complete the journey while scheduling the best routes within its battery charge level limits.** This entails strategically planning charging stops along the route to recharge the EVs' batteries. Finding the best route depends on stopping at the most efficient charging stations, including considering:

1. Closest charging station
2. The queuing times at charging stations

Our approach involves two phases:

1. Partitioning Charging Stations into Clusters: We will use the K-Means clustering algorithm to divide the charging stations into clusters based on geographical proximity. Figure 1 provides a visual representation of the clustering of recharging stations. Each cluster will have known waiting times for its charging stations. We will assume an average waiting time for stations with unknown waiting times. At any point along the route, we can access waiting times exclusively for the charging stations within the current cluster.
2. Optimizing the Total Travel Distance: In the second phase, we aim to optimize the total travel distance by determining the best charging stations for the EV to stop along the route. This will be achieved using a Genetic Algorithm (GA) to strategically select the route with charging stations, considering factors such as the distance to the station and the potential queue of cars waiting for recharging. Whenever a driver enters a new cluster, our model will be re-evaluated to find a new optimal route, considering the known waiting times of the nearby stations. This dynamic reassessment ensures that the driver always follows the most efficient route based on the information on the latest charging station waiting times.

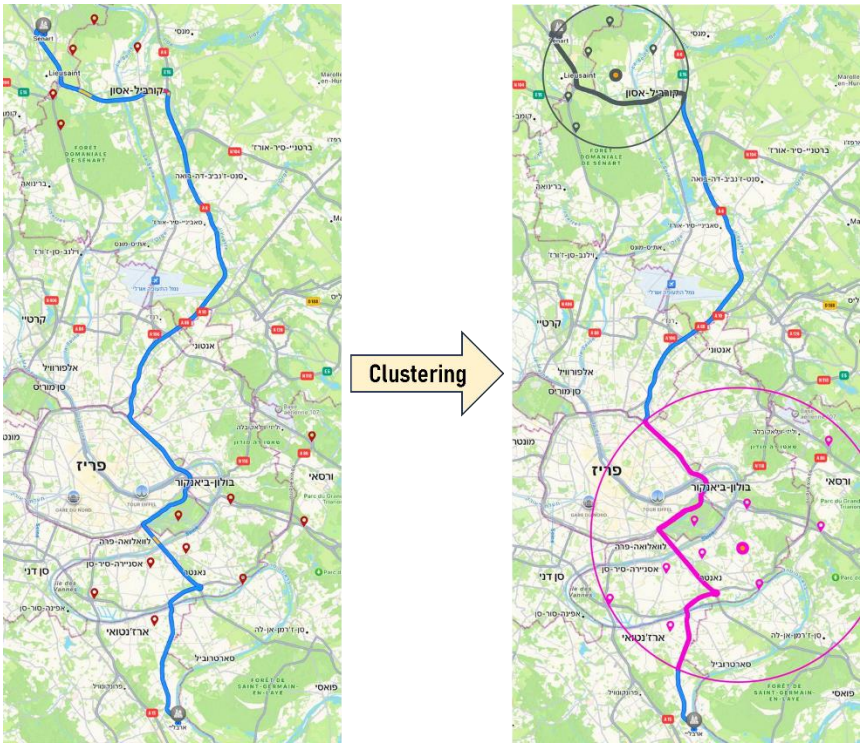


Fig. 1. Clustering of recharging stations.

2. Related Work

Optimizing Electric Vehicle Routing with Non-Linear Charging Efficient routing strategies for electric vehicles (EVs) are essential in minimizing environmental impact and operational costs. Several notable research endeavors address Electric Vehicle Routing Problems (EVRPs).

Sašo Karakatič's [8] paper addresses the Electric Vehicle Multi-Depot Vehicle Routing Problem with Time Windows and Nonlinear Charging Times (E-MDVRPTW-NL). It proposes an innovative solution using a genetic algorithm (GA) with a novel genotype representation. The problem arises from the need to optimize routes for electric vehicles considering multiple objectives: route optimization, selecting appropriate charging stations, and determining optimal recharging times.

The key challenge the paper addresses is the complexity of the E-MDVRPTW-NL problem, which requires simultaneously optimizing both discrete (route sequence) and continuous (charging station selection and recharging times) variables. Existing optimization methods for vehicle routing problems are not directly applicable due to the problem's mixed nature and the lack of suitable solution representations.

To address this challenge, the paper proposes a two-layer genotype representation, where the first layer represents the route sequence of customer visits, and the second layer optimizes charging station selection and recharging times. A two-layer genetic algorithm is then employed, alternating between modifying each layer using mutation and crossover operators tailored to its specific characteristics.
the overview of the whole optimization process with the genetic algorithm is presented in Fig. 2.

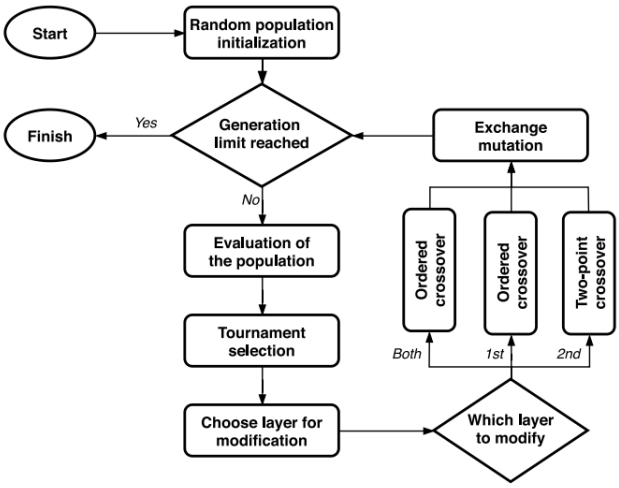


Fig. 2. The whole evolution process.

The contributions of the paper are three-fold:

1. Proposition of the E-MDVRPTW-NL problem, which closely resembles real-world situations and can be extended for further research.

2. Proposition of a two-layer problem representation, allowing for simultaneous optimization of discrete and continuous variables, along with a tailored genetic algorithm solution.

3. Presentation of problem instances that can serve as benchmarks for future research and comparison with the proposed solution.

The results demonstrate the effectiveness of the proposed approach in solving the E-MDVRPTW-NL problem, providing a valuable contribution to the field of electric vehicle routing optimization. The proposed solution offers a novel and efficient method for addressing the unique challenges posed by electric vehicle routing problems, paving the way for further advancements in this area.

The study by Yong Jun Kim and Byung Do Chung (2022) [9] investigates the Electric Vehicle Routing Problem (EVRP) domain by addressing the intricacies of State of Charge (SoC)-dependent charging and discharging rates (EVRP-SoC-CD).

The methodology involves the development of optimal charging policies that account for SoC-dependent rates, exploring four distinct scenarios: constant charging and discharging rates, SoC-dependent charging and constant discharging rates, constant charging and SoC-dependent discharging rates, and SoC-dependent charging and SoC-dependent discharging rates. Case 4 emerges as the most realistic scenario, highlighting the necessity of accounting for these dynamic factors in EV operations. This exploration ensures a more accurate representation of real-world EV operations.

In addition, the study also demonstrates that a strict charging policy, commonly used in the literature for constant discharging rates, may not be energy-efficient in real-world scenarios where the charging and discharging rates are SoC-dependent. This finding underscores the need for optimal charging policies adjusted to cases with

SoC-dependent rates. Firstly, regardless of the type of charging station, the EV tends to charge more than the strict charging policy dictates. This behavior stems from the fact that maintaining a higher SoC level during charging can reduce energy consumption. Furthermore, analysis of the order of visits to the CS reveals that EVs tend to charge more at the first charging station on the route than at subsequent stations. This phenomenon occurs because a longer charging time at the first station helps maintain a higher SoC level for the remaining trip, resulting in lower energy consumption.

The study's results underscore the significance of SoC-dependent discharging rates through a series of experiments, revealing that neglecting these rates could lead to infeasible or suboptimal solutions.

The paper by Wang, Z. (2021) [10] addresses the Hybrid Charging Strategy based Dynamic Vehicle Routing Problem (HCS-DVRP), which arises from the need to optimize Electric Vehicle (EV) routes considering both traditional Plug-in Charging Technology (PCT) and Wireless Charging Technology (WCT) options. The problem involves changing charging station availability dynamically and the trade-off between charging cost and travel time.

To tackle this challenge, the paper proposes an innovative framework called the Transfer Knee points based Multi-Criteria Decision Making (TK-MCDM) algorithm. This framework leverages knee points, a subset of non-dominated solutions with higher HyperVolume (HV) values, to guide the search for optimal solutions. The proposed algorithm accelerates convergence, maintains population diversity, and reduces the cognitive burden on decision-makers (DMs) by reusing a few high-quality knee solutions. Fig. 3 illustrates the benefits of the knee point.

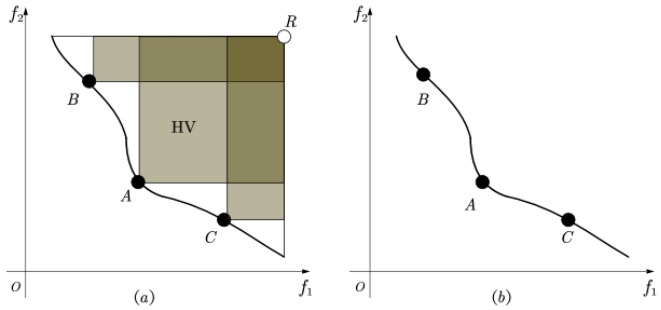


Fig. 3. (a) Compared with other non-dominated solutions B and C, knee point A has a higher HV value to the reference point R. (b) Compared with other solutions, knee point A is mostly preferred by DMs.

The framework comprises three critical components: knee selection, collection of training solutions, and reuse function learning. First, knee points are identified in each objective space division. Second, training samples are collected to train the reuse function, considering the imbalanced problem caused by the lack of knee points. Finally, a reuse function is learned to predict initial knee points in the changing environment by exploiting the collected training solutions.

The proposed framework's contributions include utilizing knee points to accelerate convergence and relieve DMs' burden in selecting solutions, designing an adaptive

learning mechanism to efficiently reuse knee points, and addressing the imbalanced problem in solution generation.

The proposed TK-MCDM algorithm's results demonstrate its effectiveness in efficiently solving the HCS-DVRP. The algorithm achieves accelerated convergence by reusing high-quality knee solutions, maintains diversity, and reduces the cognitive burden on DMs. Overall, the framework offers a promising approach to efficiently tackle the challenges posed by HCS-DVRP in real-world dynamic environments, contributing significantly to the field of vehicle routing optimization, particularly in the context of electric vehicles and hybrid charging strategies.

Merve Keskin's paper [6] addresses the Electric Vehicle Routing Problem with Time-Dependent Waiting Times at Recharging Stations, an extension of the Vehicle Routing Problem tailored to electric vehicles (EVs). The key challenge addressed in this study revolves around the practical consideration of waiting times at recharging stations, which impact routing decisions due to limited chargers and unpredictable queuing dynamics. To tackle this challenge, the authors propose extending the Electric Vehicle Routing Problem with Time Windows (EVRPTW), incorporating an M/M/1 queueing system to predict queue lengths, nonlinear charging functions, and penalties for late arrivals. Additionally, the planning horizon is divided into time intervals to account for varying EV arrival rates and queue lengths. The solution methodology integrates Adaptive Large Neighborhood Search (ALNS) with exact methods, optimizing charging station selection and recharge quantity decisions using mixed integer programming.

The paper's contributions lie in offering a more realistic model for EVRPTW, providing an effective solution methodology that integrates queueing dynamics, and presenting insights into optimizing electric vehicle routing.

Merve Jaël Champagne Gareau, Éric Beaudry, and Vladimir Makarenko's paper [7] addresses the challenging problem of Electric Vehicle Path Planning with Intermediate Recharges (EVPP-R). The EVPP-R problem uses careful planning to determine optimal charging stations during long journeys, considering factors such as charging station availability, battery characteristics, and terrain topology.

To tackle these challenges, the authors propose a novel clustering technique to solve the EVPP-R problem efficiently. The key idea is to reduce the computational complexity of path planning by clustering charging stations based on their proximity and characteristics. The number of nodes considered in the planning graph is significantly reduced by grouping stations into clusters, thereby expediting the path-planning process.

The paper's main contributions include:

1. Introduction of a fast and intuitive clustering technique for solving the EVPP-R problem.

2. Evaluation of the proposed technique using real-world data to assess its performance and effectiveness.

3. Demonstration of significant reductions in computation time while maintaining solution quality through the clustering approach.

The authors' empirical evaluation validates their proposed approach's effectiveness. Using real data from Québec, Canada, the paper demonstrates notable computation time decreases without sacrificing the solutions' quality. By comparing various parameter settings and conducting thorough analyses, the authors provide valuable insights into the practical implementation of their clustering-based approach for EV route planning systems.

3. Background

3.1. Genetic Algorithm

3.1.1. Introduction

Genetic algorithms (GAs) are a powerful tool inspired by the process of natural selection, first introduced in 1970 by John Holland. They have proven to be particularly effective at solving complex optimization problems. Much like nature's evolution, the GA process begins with selecting individuals for the first generation, manipulating them through crossover and mutation techniques to generate subsequent generations. The idea is that if parents have better fitness, their offspring will be better than the parents and have a better chance of surviving. Through an iterative process, this continual cycle ultimately yields a generation populated by the most fit individuals. [11]

3.1.2. Genetic algorithm vocabulary

1. **Chromosome (Individual):** Represents a potential solution as a string of genes.
2. **Gene:** Component of a chromosome that encodes a particular part of the solution.
3. **Population:** The population consists of a collection of chromosomes.
4. **Fitness Function**: Evaluate each chromosome by assigning a fitness score, reflecting how well it meets the problem's objectives or constraints.
5. **Locus:** Position of gene in chromosome

3.1.3. Pseudo Code for Genetic Algorithm [11]

1. **Initialization**- Generate n chromosomes for the initial population p
2. **Loop** until the end condition is met
    a. **Evaluation-** Evaluate the fitness of each chromosome in p
    b. **While** n offspring have not been created, **do**
        i. **Selection-** Select 2 parents from P based on their fitness
        ii. **Crossover-** Perform crossover to create offspring. If no crossover was performed, the offspring is an exact copy of the parents
        iii. **Mutation**- With a mutation probability, mutate new offspring at each locus
        iv. **Accepting**- Place new offspring in a new population
    c. **Replace**- Replace old population with the new population

3. **Return** the best solution

3.1.4. Initial Population

Generate a random population of n chromosomes, each representing a potential solution to the problem. Each chromosome contains a set of variables that mimic genes. The primary aim during the initialization step is to spread the solutions evenly across the search space, enhancing the diversity of the population and improving the likelihood of discovering promising regions.[12]

3.1.5. Genetic Algorithm Operators

Genetic Algorithms (GAs) initiate with a population of random strings representing solutions. The population is then operated by three main operators - reproduction, crossover, and mutation - to generate a new population of solutions. GAs aim to maximize the fitness function by assessing several solution vectors. The operators aim to create new solution vectors by selecting, combining, or altering existing ones that have proven to be good temporary solutions. The new population is further evaluated and tested until termination. If the termination criterion is not met, the population is iteratively operated by the above three operators and evaluated. This procedure is continued until the termination criterion is met [13]. One cycle of these operations and the subsequent evaluation procedure is known as a generation in GAs terminology. The operators are described in the following steps.

Selection Operation

Selection is an operator that chooses elite individuals from the current population to serve as parents for producing offspring [11]. The fitness function is used to evaluate the fitness of each chromosome in the population and determine how suitable it is for solving the problem. Drawing inspiration from natural selection, which favors the fittest individuals in nature, this process ensures that individuals with higher fitness values have a greater chance of being selected for generating the next generation. Various methods exist for selecting the best chromosomes:

1. **Tournament Selection** - Tournament selection within GAs entails organizing a contest among a predetermined number of individuals. The winner of each tournament, identified as the individual with the highest fitness among the competitors, is then chosen to join the mating pool. By adjusting the tournament size, the level of selection pressure can be modified: larger tournament sizes intensify selection pressure, thereby favoring individuals with superior fitness. This approach effectively facilitates the selection of high-quality individuals for reproduction, thereby enhancing the overall fitness of subsequent generations. However, balancing the tournament size is essential to avoid either slow convergence or premature convergence to suboptimal solutions. Adjusting the tournament size enables fine-tuning the selection pressure to optimize the GA's performance in identifying optimal or near-optimal solutions [14].
2. **Roulette Wheel Selection -** parents are chosen based on their fitness within the population. Essentially, the fitter the chromosome, the greater its likelihood of being selected. This selection method is analogous to a roulette wheel, where each chromosome occupies a portion of the wheel corresponding

to its fitness score, an illustration of the Roulette Wheel Selection in Fig. 4 [12].

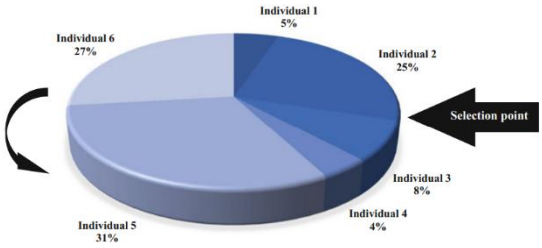


Fig. 4
Roulette wheel selection. The best individual (#5) has the largest share of the roulette wheel, while the worst individual (#4) has the lowest share.

3. **Rank Selection -** In situations where there's a significant disparity in fitness levels among chromosomes, for instance, in the traditional roulette wheel selection, if the highest fitness chromosome occupies 90% of the roulette wheel, others have minimal chances of selection. The Rank selection addresses this issue by initially sorting the population based on fitness, assigning each chromosome a rank corresponding to its position in the sorted list. The worst-ranked chromosome receives a fitness score of 1, the second worst 2, and so on, with the best-ranked receiving a fitness score equal to the population size. Subsequently, all chromosomes have an opportunity for selection, with the probability of selection being proportional to their rank rather than their fitness. However, this approach may lead to slower convergence as the differences between the best and other chromosomes are less pronounced [11].

4. **Steady State Selection –** Steady-state selection prioritizes the survival of high-fitness chromosomes into the next generation. Instead of following a specific method for selecting parents, it maintains population stability while favoring the fittest individuals. During each generation, a subset of high-fitness chromosomes produces new offspring. Low-fitness individuals are removed to make space for the offspring, with the remaining population continuing to the next generation. This approach preserves favorable traits, gradually improving overall fitness by allowing the best-performing individuals to propagate [15].

5. **Elitism Selection** - This selection method ensures that the best-performing individuals are preserved across generations. When creating a new population, there's a risk of losing the best chromosome due to chance factors. Elitism addresses this by copying the best chromosome, or a few top performers, into the new population before generating the rest using standard methods. This approach rapidly enhances GA performance by safeguarding the best-found solutions from being lost [11].

Crossover Operations

The generation of successors in a GA is determined by a set of operators that recombine and mutate selected members of the current population after selecting the individuals using a selection operator. The two most common operators are crossover and mutation. The crossover operator produces two new offspring from two-parent strings by copying selected bits from each parent. Each offspring's bit at position i is copied from the bit at position i in one of the two parents. The choice of which parent contributes the bit for position i is determined by an additional string called the crossover mask [11]. A crossover mask is simply a string of bits the size of the chromosomes to be crossed. The parity of each bit in the mask determines which parent will receive that bit for each corresponding bit in a child. There are different techniques for the crossover operator, including:

1. **Standard single-point and double-point crossovers:** In the single-point crossover, the chromosomes of two parent solutions are swapped before and after a single point. The double-point crossover has two crossover points, and the chromosomes between the points are swapped only. Illustration of the crossovers in Fig. 5 [11].

2. **Uniform crossover:** Uniform crossover involves combining bits randomly sampled from both parents. A crossover mask is generated as a random bit string, with each bit independently chosen randomly, as shown in Fig. 5 [11].

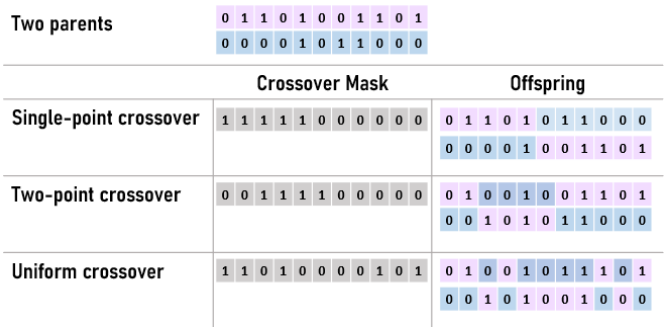


Fig. 5. An example of Single-point, Double-point, and Uniform crossovers.

3. **Ordered crossover (OX):** Order Crossover (OX) is a permutation-based crossover method. The illustration of OX in Fig. 6 showcases the crossover process between two parent sequences. Initially, two random crossover points are selected within the parent sequences. The segment between these crossover points in the first parent is highlighted in pink and copied directly to the first offspring. Subsequently, the process continues with the second parent: starting from the second crossover point, only the unused numbers (depicted in blue, with the red numbers indicating those already used) are selected and copied to the first child, potentially wrapping around the sequence. This sequence of steps is then repeated for the second child, with the roles of the parents reversed [16].

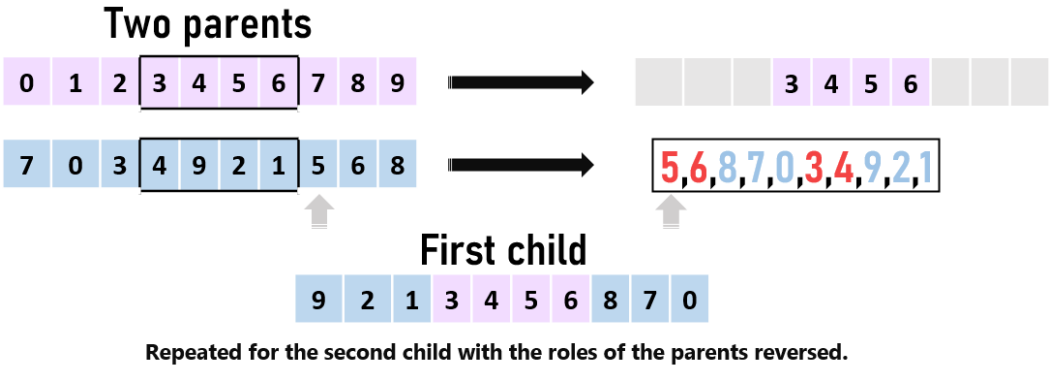


Fig. 6. An illustration of ordered crossover.

Mutation

Mutation typically occurs after the crossover stage, where recombination operators combine genetic material from two parents to create offspring. The mutation operator randomly chooses a single bit and changes its value. Mutation serves as a complementary process to recombination, introducing small random changes to the genetic makeup of the resulting offspring. This randomness is crucial for maintaining diversity within the population and preventing premature convergence to suboptimal solutions. For example, the following population consists of 8-bit strings: 01101011, 00111101, 00010110, 01111100. It can be noticed that all four strings have a 0 in the leftmost bit position. If the true optimum solution requires 1 in that position, then neither the reproduction nor crossover operator described above will be able to create 1 in that position.[13]

Common mutation operators (Illustration in Fig. 7):

Bit Flip Mutation- select one or more random bits and flip them.

Exchange Mutation- select two positions on the chromosome at random and interchange the values.

Scramble Mutation- a subset of genes is chosen, and their values are scrambled or shuffled randomly.

Inversion Mutation- select a subset of genes like in scramble mutation, but instead of shuffling the subset, we merely invert the entire string in the subset.

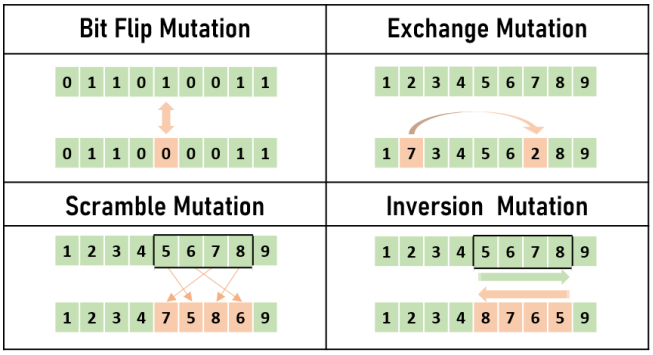


Fig. 7. An illustration of common mutation operators.

Genetic algorithms balance exploration and exploitation by incorporating crossover and mutation, ensuring an effective solution space search while avoiding local optima.

3.2.Clustering

Clustering is an unsupervised learning technique to uncover natural structures or patterns within a dataset without predefined labels. The goal is to group similar data points and separate dissimilar ones, as shown in Fig. 8. Doing so helps understand the data's natural organization and identify meaningful patterns. Clustering involves partitioning the dataset into clusters such that objects within the same cluster are more similar than objects in different clusters. The similarity between data points is typically measured using distance metrics, and various clustering algorithms exist to perform this task. [17]

3.2.1. Applications [18]:

1. Clustering simplifies large datasets by condensing them into representative clusters, aiding in data management and analysis.
2. It assists in creating classification schemes, notably in biology, facilitating species classification based on shared characteristics.
3. It unveils data patterns, suggesting hypotheses about data structure for deeper analysis.

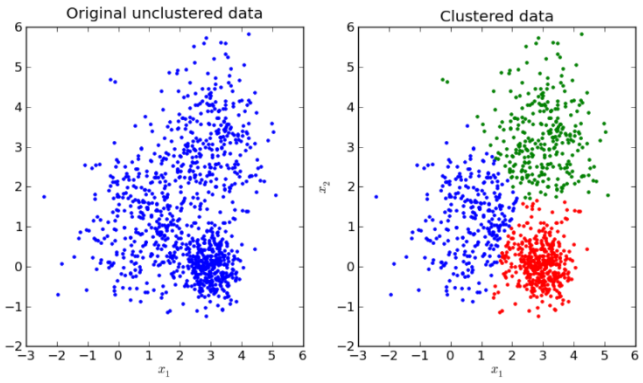


Fig. 8. Visualization of data clustering.

3.2.2. The Clustering Problem: [18]

Clustering in N-dimensional Euclidean space $\mathbb{R}^N$ is the process of partitioning given n points from the data set $S = \{x_1, x_2 \dots, x_n\}$ into K clusters $C_1, C_2 \dots, C_n$ based on some similarity metrics.

Cluster Conditions:

1. Each cluster C_i should be non-empty: $C_i \neq \emptyset$ for $i = 1, \dots, k$
2. No two clusters should overlap: $C_i \cap C_i \neq \emptyset$ for $i = 1, \dots, k$, $j = 1, \dots, k$ and $i \neq j$
3. The union of all clusters should cover the entire dataset: $S = \bigcup_{i=1}^{K} C_i$

3.2.3. K-means:

One of the most popular and widely used clustering methods is K-means. [18] The k-means algorithm aims to minimize the sum of squared distances between each data

point and its assigned cluster centroid. The objective function represents the total within-cluster variance, where cluster centers are updated iteratively to minimize this variance. [19]

K-means algorithm: [18]

1. select K initial cluster centers $z_1, z_2 \dots, z_n$ from the data set $S = \{x_1, x_2 \dots, x_n\}$
2. For each data point x_i calculate its distance to each cluster center z_j using a distance metric. $i = 1, \dots, n$, $j = 1, \dots, k$.
   Assign the data point x_i to the cluster represented by the nearest cluster center z_j.
3. Compute new cluster centers $z_{H1}, z_{H2} \dots, z_{HK} : z_{Hi} = \frac{\sum x_j}{n_i}\ x_j \in C_i, i = 1, .. k$
   z_{Hi} for cluster C_i is the mean of all data points x_j in cluster C_i
4. For all the clusters: If $z_{Hi} = z_i, , i = 1, .. k$ then terminate. Otherwise, continue from step 2.

3.2.4. K-Value Selection

Selecting the appropriate value of K in the K-Means algorithm is a crucial step that significantly impacts the clustering performance. Several methods exist for determining K and assessing the validity of the resulting clusters. There are a few approaches, such as:

User-specified K value: in this method, the user manually specifies the value of K in the algorithm implementation. This typically requires multiple iterations with different K values to achieve a satisfactory clustering result. However, this approach may lead to subjective interpretations and lacks objective validation.

Elbow method algorithm: The Elbow Method algorithm is a simple yet effective technique for determining the optimal number of clusters (K) in K-means clustering. It systematically tests different K values, typically ranging from 1 to a set maximum of N. The Within-Cluster Sum of Square (WCSS) is calculated for each K value, representing the variance within clusters. WCSS is a square of the distance between the sample points in each cluster and the cluster's centroid. A curve resembling an elbow is obtained by plotting WCSS against the number of clusters. The point where the rate of decrease in WCSS significantly slows down is considered the optimal K value, as the illustration in Fig 9 shows. This indicates a balance between clustering performance and model complexity, as further increasing K provides diminishing returns in reducing WCSS [19].

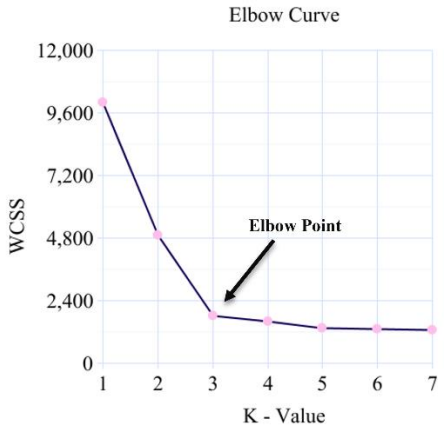


Fig. 9. An illustration of an elbow curve.

The Silhouette Coefficient Algorithm:

The Silhouette Coefficient algorithm combines the concepts of cohesion (similarity within clusters) and separation (difference between clusters) to provide a measure of how well individual data points fit into their assigned clusters [19].

The Silhouette Coefficient $s(i)$ ranges from -1 to 1, where:

- $s(i)$ close to 1 indicates that the data point is well-clustered.
- $s(i)$ close to -1 indicates that the data point may have been assigned to the wrong cluster.
- $s(i)$ around 0 indicates that the data point is close to the decision boundary between two clusters.

The Calculation Method:

1. Calculate Intra-cluster Dissimilarity $a(i)$:
    - For each data point i, calculate the average distance to other data points within the same cluster.
    - $a(i)$ represents the average dissimilarity (distance) between data point i and other points in the same cluster.
2. Calculate Inter-cluster Dissimilarity $b(i)$:
    - For each data point i, calculate the average distance to all data points in the nearest neighboring cluster.
    - $b(i)$ represents the average dissimilarity (distance) between data point i and points in the nearest neighboring cluster.
3. Calculate Silhouette Coefficient $s(i)$:
    - Compute $s(i)$ using the formula: $s(i) = \frac{b(i)-a(i)}{\max(a(i),b(i))}$

4. Expected Achievements:

The problem presented in our study is to ensure EVs can complete their journeys by planning the best routes within battery charge limits and strategically scheduling charging stops. Our approach uses the K-Means clustering algorithm to partition charging stations into clusters based on geographical proximity. Additionally, we will employ a Genetic Algorithm (GA) to optimize the total travel distance by selecting

the best charging stations along the route, considering distances and queuing times. This dynamic model will continuously re-evaluate the optimal route as the vehicle progresses, ensuring drivers follow the most efficient path based on the latest information. Furthermore, we aim to develop a flexible and scalable system capable of delivering excellent results across various inputs, including changes in the number of charging stations, long travel distances, and low battery capacities. The expected outcome of this project is a system that receives a start and destination point and clusters charging stations. It calculates the optimal route with charging stops, providing a practical, efficient, and adaptable solution for EV drivers.

5. Proposed approach:

5.1. Research Process

Our proposed research begins by examining electric vehicles (EVs) and focusing on their operational limitations. We investigate EV characteristics to understand the constraints that shape their performance and usability. This basic understanding guides our next research process.

Building upon this understanding, we start researching the Electric Vehicle Routing Problem (EVRP). We explore the complexities of EVRP, which involves optimizing routes for electric vehicles while considering charging constraints, energy consumption, and other factors. By analyzing existing literature and methodologies in EVRP, we identify key challenges and opportunities for innovation.

To identify the optimal charging stations, we minimize the total travel distance while factoring in the waiting times at these stations. We use a clustering algorithm to partition the charging stations into clusters, each with known waiting times for its stations. Drivers are assigned to clusters based on proximity, defined by a specific distance threshold from the station. If the driver approaches a station within this threshold distance, they enter the corresponding cluster. For stations outside this cluster, we assume an average waiting time.

Whenever a driver enters a new cluster, our model is recalibrated to determine the most optimal route, considering the updated waiting times of nearby charging stations. This dynamic approach ensures that route planning adapts in real time, optimizing the driver's journey based on current information.

5.1.1. Clustering

We employ K-Means clustering to efficiently group charging stations, reducing computation time by limiting the number of charging stations considered in the graph during optimization. To refine our routing strategy by optimizing station grouping based on proximity, we investigate two methods to determine the appropriate value for K:

- The Silhouette Coefficient Algorithm
- The Elbow Method Algorithm

5.1.2. Genetic Algorithm

Our research delves into applying genetic algorithms (GAs) to address optimization challenges.

Key parameters:

Input parameters:

P Maximum capacity of the EV (distance that can be traveled on a full charge).

A, B start and destination point.

Penalties:

Q_i Penalty incurred due to queuing at charging stations.

Z_i Penalty for recharging at a station.

R_i Distance of Recharging station from the EV.

Other parameters:

d_j represents the distances between consecutive charging stations along the route. We will use d_j to verify the validity of the chromosome by ensuring that the distance between the two stations does not exceed the EV's battery capacity P.

$1 < j < n-1, 1 < i < n, n = number\ of\ chosen\ charging\ station$

Chromosome

We propose to encode the chromosome as follows [20]: Each chromosome represents a solution for the routing path for an EV from the starting point to the destination, with stops at charging stations. The chromosome comprises sequences of positive integers corresponding to the IDs of charging stations along the route. Each locus of the chromosome signifies the order of a charging station in the routing path, with the first locus reserved for the starting point and the last for the destination point. The length of the chromosome is **variable**, but it should not exceed the maximum length (the total number of charging stations). The genes are selected based on d_j parameter, i.e. the condition that the distance between two adjacent charging stations does not exceed the EV's battery capacity, ensuring that the route is feasible for the vehicle's energy constraints. Figure 10 shows an example of a chromosome with length l that represents the route $A \rightarrow CS_1 \rightarrow CS_2 \rightarrow \cdots \rightarrow CS_n \rightarrow B$.

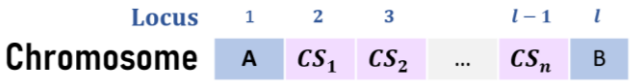


Fig. 10. Example of routing path with the chosen charging stations and its encoding scheme.

Fitness Function

Our approach involves the development of a fitness function aimed at comprehensively evaluating the quality of candidate solutions. The factors included in the fitness function are:

- The distance to the charging station from the route.

- Queue model- The queue will be modeled as a penalty. The longer the queue, the greater the penalty for charging at this station. In the fitness function, we will enter the queue times and calculate the distance to the charging station. We want to investigate how much more severe the penalty should be. We will investigate the optimal ratio between the waiting time at the charging station and the number of kilometers that will be added to the total distance.
- Charging penalty- We will explore the ideal penalty in kilometers to add to the total distance within the fitness score when stopping at the charging station for recharging. We want this to estimate charging times. For example, if there is a charging station with negligible distance from the route, we want the algorithm to decide not to charge if it doesn't need to because it adds time.
- We assume the EV leaves with 100% battery from each charging station.

The fitness function is as follows:

$$F = \frac{1}{\sum_i^n (R_i + Q_i + Z_i)}$$

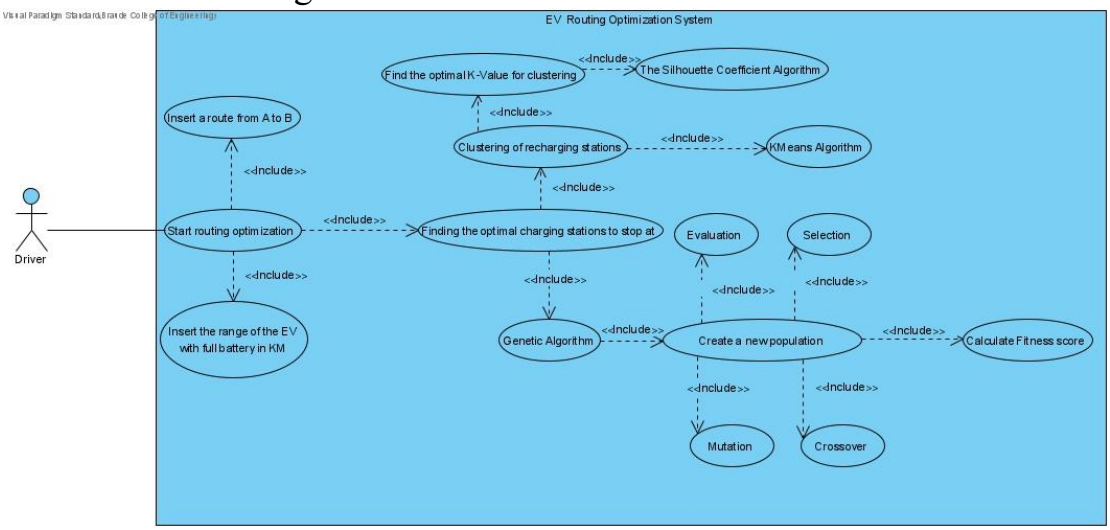
By integrating these factors and parameters into our fitness function, we aim to develop a robust optimization framework for the problem, enhancing the efficiency and practicality of electric vehicle routing.

Genetic Operators

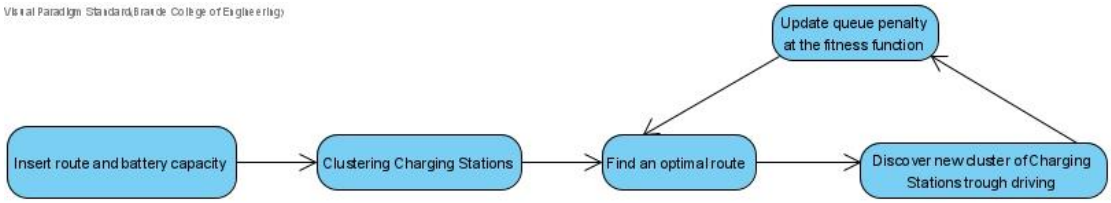
To enhance the effectiveness of our genetic algorithm, we will analyze genetic operators utilized in the optimization process. Selection, mutation, and crossover play crucial roles in promoting the evolution of candidate solutions towards more optimal configurations. This research stage will improve our solutions, convergence speed and quality, ensuring we effectively address the unique challenges of our optimization domain.

5.2.Product

5.2.1. Use Case diagram.



5.2.2. Flow Chart



5.2.3. GUI

Figure 11 presents the initial screen where users input their route data and EV details.

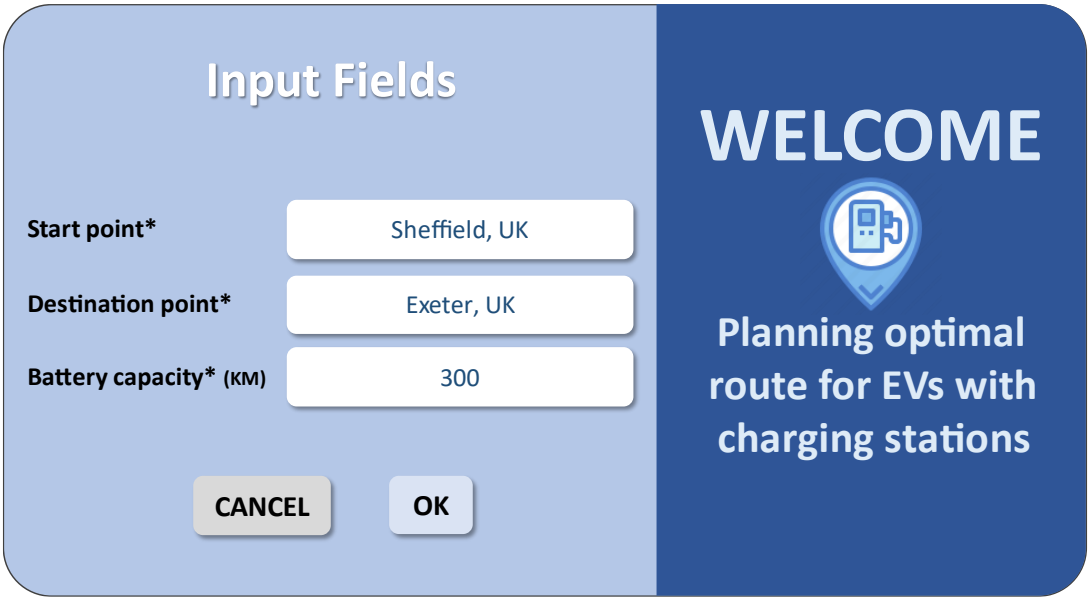


Fig. 11. WELCOME screen + input data.

Once the data is inserted, as shown in Figure 12, the route information is displayed:

1. User input details
2. The length of the route without stops for charging and its map.
3. List of optional charging stations for stopping.

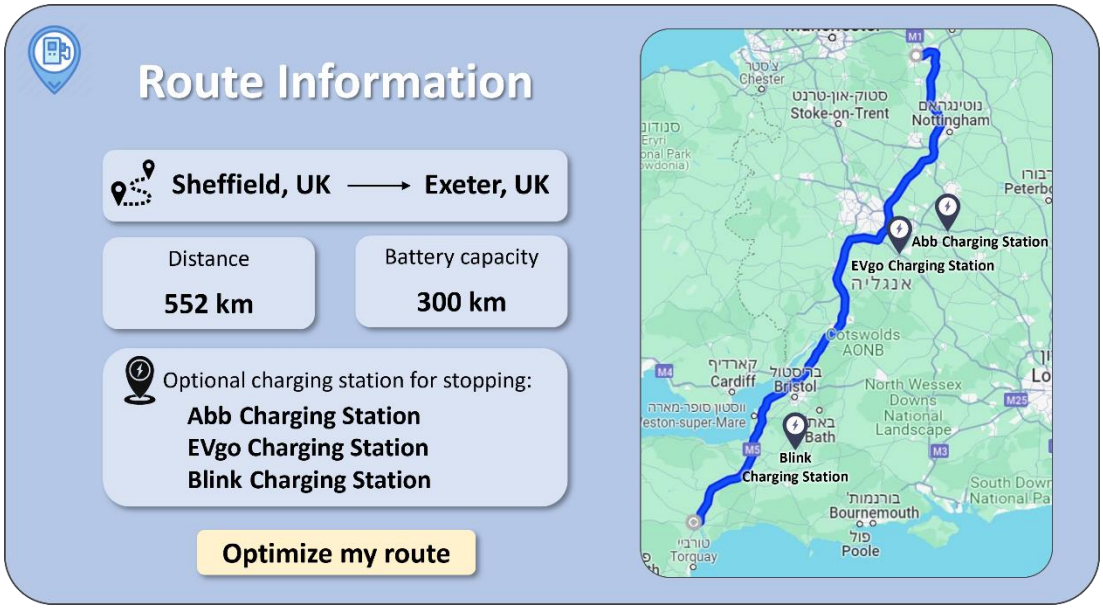


Fig. 12. Route information before the optimization process of adding a stop at charging stations.

After the optimization process, the optimal route with stops at the charging stations and their estimated waiting times is displayed, as shown in Figure 13. This initial planned route may be updated in the future according to the updated waiting times at the charging stations.

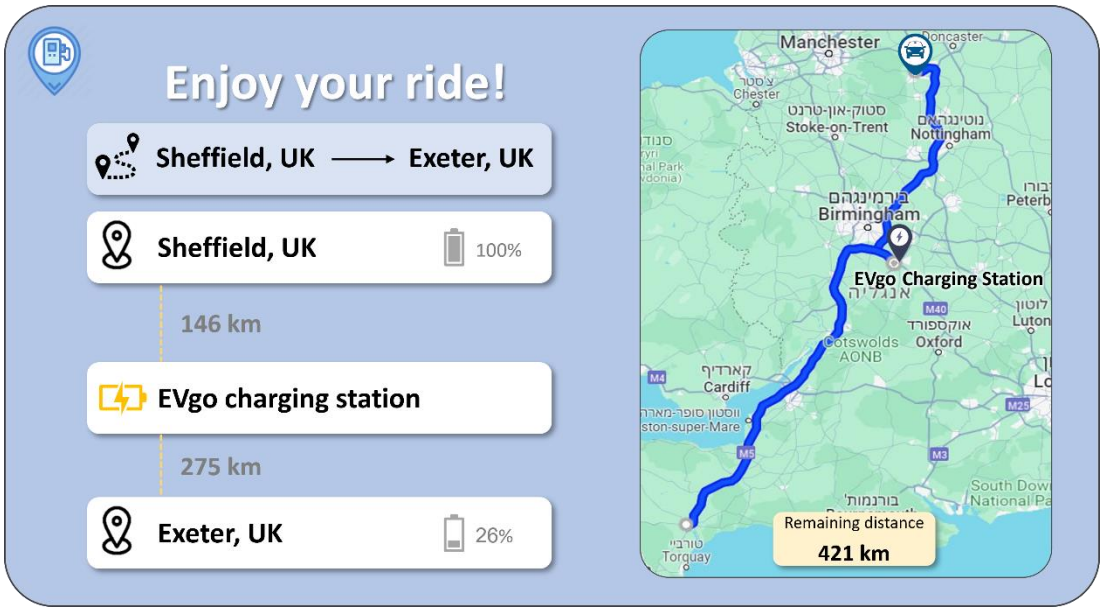


Fig. 13. Route information after the optimization process and adding a stop at charging stations.

As shown in Figure 14, during the route, due to entering the cluster area, the waiting times of the charging stations were updated, and another charging station was chosen as the optimal station.

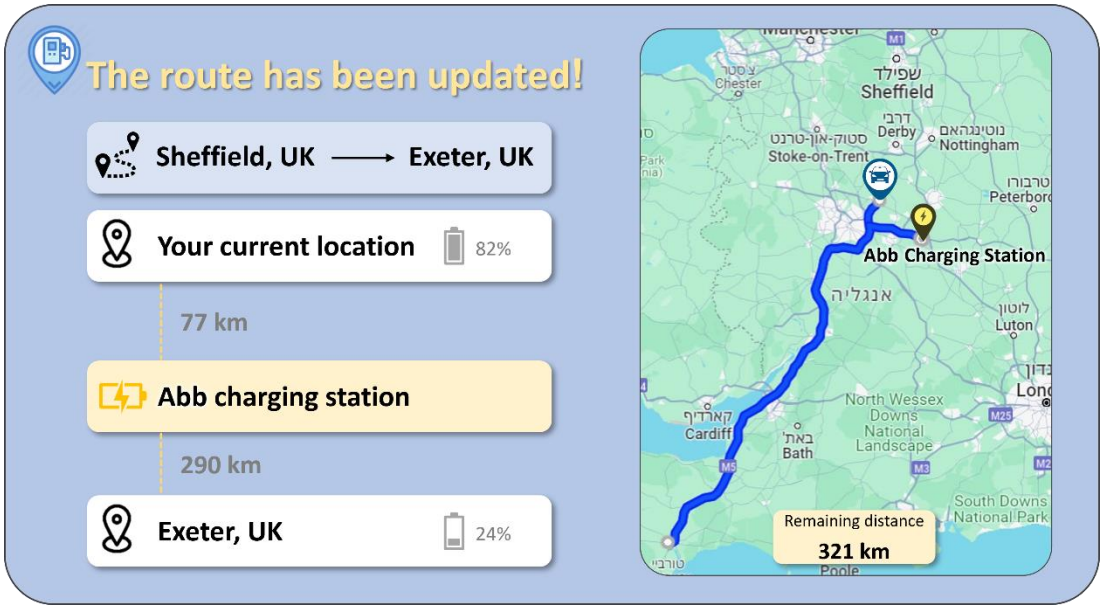


Fig. 14. Updated route information.

6. Test Plan:

	Test Case	Expected Result
Input Validation	Insert invalid start or destination points (e.g., non-existent locations or invalid format).	Error message: "Invalid start/destination point."
	Enter a negative or zero value for battery charge capacity.	Error message: "Invalid battery charge capacity."
	The user doesn’t insert value for start or destination points.	The "OK" button is disabled.
	The user doesn’t insert a value for battery charge capacity.	The "OK" button is disabled.
	Impossible route (e.g., there are not enough charging stations to complete the route).	Output: "Unable to complete the route due to insufficient charging stations. Please select a different route or adjust the battery charge capacity."
User interaction	Click on the “CANCEL” button.	Empty the fields.
	Click the "OK" button. The start and destination points are valid, and it is a possible route.	Display the ‘Route Information’ screen.
	Click the “Optimize my route” button.	Display the ‘Output’ screen, which shows the route's stages, the charging stations, and their waiting times.

7. References:

1. Liu, Fei, Chengyu Lu, Lin Gui, Qingfu Zhang, Xialiang Tong, and Mingxuan Yuan. "Heuristics for vehicle routing problem: A survey and recent advances." arXiv preprint arXiv:2303.04147 (2023). https://arxiv.org/pdf/2303.04147
2. Dantzig, George B., and John H. Ramser. "The truck dispatching problem." Management Science 6, no. 1 (1959): 80-91. https://pubsonline.informs.org/doi/10.1287/mnsc.6.1.80
3. Gintaras Vaira. "GENETIC ALGORITHM FOR VEHICLE ROUTING PROBLEM". Doctoral Dissertation Technological Sciences, Informatics Engineering (07 T) (2014). https://www.mii.lt/files/doc/lt/doktorantura/apgintos_disertacijos/mii_dis_2014_vaira.pdf
4. IEA. "Fast Facts on Transportation Greenhouse Gas Emissions" (2023). https://www.epa.gov/greenvehicles/fast-facts-transportation-greenhouse-gas-emissions
5. Ilker Kucukoglu, Reginald Dewil and Dirk Cattrysse. "The electric vehicle routing problem and its variations: A literature review". Expert Systems with Applications Volume 161, 107650 (2021). https://www.sciencedirect.com/science/article/abs/pii/S0360835221005544?via%3Dihub
6. Merve Keskin, Gilbert Laporte and Bülent Çatay. "Electric Vehicle Routing Problem with Time-Dependent Waiting Times at Recharging Stations". Computers & Operations Research Volume 107, Pages 77-94 (2019). https://www.sciencedirect.com/science/article/abs/pii/S030505481930053X
7. Jaël Champagne Gareau, Éric Beaudry, and Vladimir Makarenkov. "A Fast Electric Vehicle Planner Using Clustering". Data Analysis and Rationality in a Complex World (pp.17-25) (2021). https://www.jaelgareau.com/fr/publication/gareau-ifcs19/gareau-IFCS19.pdf
8. Sašo Karakatič. " Optimizing nonlinear charging times of electric vehicle routing with genetic algorithm". Expert Systems with Applications Volume 164, 114039 (2021). https://www.sciencedirect.com/science/article/abs/pii/S0957417420308083?via%3Dihub
9. Yong Jun Kim, Byung Do Chung. "Energy consumption optimization for the electric vehicle routing problem with state-of-charge-dependent discharging rates". Journal of Cleaner Production Volume 385, 135703 (2023). https://www.sciencedirect.com/science/article/abs/pii/S0959652622052775?via%3Dihub
10. Zhenzhong Wanga, Kai Ye, Min Jiang, Junfeng Yao. "Solving hybrid charging strategy electric vehicle-based dynamic routing problem via evolutionary multi-objective optimization". Swarm and Evolutionary Computation Volume 68, 100975 (2022). https://www.sciencedirect.com/science/article/abs/pii/S2210650221001371?via%3Dihub
11. L. Haldurai, T. Madhubala and R. Rajalakshmi. "A Study on Genetic Algorithm and its Applications". International Journal of Computer Sciences and Engineering Volume-4, Issue-10 (2016). https://www.ijcseonline.org/pub_paper/22-IJCSE-01860-4.pdf

12. Seyedali Mirjalili. "Evolutionary Algorithms and Neural Networks". Pages 43-55 (2019). https://link.springer.com/book/10.1007/978-3-319-93025-1
13. Tom V. Mathew. "Genetic Algorithm". https://datajobs.com/data-science-repo/Genetic-Algorithm-Guide-[Tom-Mathew].pdf
14. Brad L. Miller and David E.Goldberg. " Genetic Algorithms, Tournament Selection, and the Effects of Noise". Com plex Systems 9, 193- 2 (1995). https://wpmedia.wolfram.com/sites/13/2018/02/09-3-2.pdf
15. Dahal, Keshav P. and McDonald, J.R. "Generational and steady-state genetic algorithms for generator maintenance scheduling problems". Artificial Neural Nets and Genetic Algorithms pp 259–263 (1998). https://link.springer.com/chapter/10.1007/978-3-7091-6492-1_57
16. Kusum Deep and Hadush Mebrahtu. " New Variations of Order Crossover for Travelling Salesman Problem". (2021). https://www.researchgate.net/publication/50923713_New_Variations_of_Order_Crossover_for_Travelling_Salesman_Problem
17. Rowena Marie Cole. "CLUSTERING WITH GENETIC ALGORITHMS". (1998). https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=39c596ef9d69334d8e6bb1b3a0fca20028a432e0
18. Ujjwal Maulik and Sanghamitra Bandyopadhyay. "Genetic algorithm-based clustering technique". Pattern Recognition Volume 33, Issue 9, Pages 1455-1465 (2000). https://www.sciencedirect.com/science/article/abs/pii/S0031320399001375
19. Chunhui Yuan and Haitao Yang. "Research on K-Value Selection Method of K-Means Clustering Algorithm". (2019). https://www.mdpi.com/2571-8800/2/2/16
20. Chang Wook Ahn “A Genetic Algorithm for Shortest Path Routing Problem and the Sizing of Populations”. (2022). https://ieeexplore.ieee.org/abstract/document/1134124