

ROUTING OF ELECTRIC VEHICLE WITH PARTIAL RECHARGE AND CUSTOMER TIME WINDOW

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

JYESTADI RAHUL (Reg.No - 40110509)

GUTTAPLAYAM KEERTHI REDDY(Reg.No – 40110424)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Category-1 University by UGC

Accredited with Grade “A++” by NAAC | 12B Status by UGC | Approved by AICTE

JEPPIAAR NAGAR, RAJIV GANDHISALAI,

CHENNAI – 600119

APRIL - 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Category - I University by UGC

Accredited "A++" by NAAC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Jyestadi Rahul(40110509)** and **Guttaplayam Keerthi Reddy(40110424)** who carried out the Project entitled "**ROUTING OF ELECTRIC VEHICLE WITH PARTIAL RECHARGE AND CUSTOMER TIME WINDOW**" under my supervision from November 2023 to April 2024.

Internal Guide

Dr. E.MURALI, M.Tech., Ph.D

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

HEAD OF THE DEPARTMENT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

Jeppiaar Nagar, Rajiv Gandhi Salai,

Chennai - 600 119.

Submitted for Viva voce Examination held on

15-4-24

Internal Examiner

External Examiner

DECLARATION

I, Guttapalyam Keerthi (Reg no 40110424), hereby declare that the Project Report entitled "ROUTING OF ELECTRIC VEHICLE WITH PARTIAL RECHARGE AND CUSTOMER TIME WINDOW" done by me under the guidance of Dr. E.MURALI, M.E.,Ph.D is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE: 15-04-24

PLACE: Chennai

G. Keerthi Reddy .

SIGNATURE OF THE CANDIDATE

DECLARATION

I, **Jyestadi Rahul (Reg no 40110509)**, hereby declare that the Project Report entitled **“ROUTING OF ELECTRIC VEHICLE WITH PARTIAL RECHARGE AND CUSTOMER TIME WINDOW”** done by me under the guidance of **Dr. E.MURALI, M.E.,Ph.D** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE:

PLACE:Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T.Sasikala M.E., Ph. D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. E.MURALI, M.E.,Ph.D**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Electric vehicles have been a hope to reduce pollution worldwide by replacing gasoline vehicles, which are significant contributors to pollution. As the demand and usage of EVs increase so does the need for charging stations(CS). But the cost to charge at Charging Station is more compared to cost to charge for Gasoline vehicles. So to reduce this cost we are proposing an algorithm which reduces the overall cost to charge. Also when there a fleet of vehicles in depot which satisfies customers in the given deadline but there maybe usage of more number of vehicles which leads to more cost. So we are proposing an optimised algorithm which not only reduces cost by path but also minimizing the number of vehicles usage in given deadline.

The proposed algorithm aims to address the challenge of reducing the overall cost of charging electric vehicles (EVs) compared to gasoline vehicles, especially considering the higher cost of charging at Charging Stations (CS). As the demand for EVs increases, the need for charging infrastructure grows, leading to potential cost implications. The algorithm focuses on optimizing the charging process to minimize costs associated with recharging EVs.

Moreover, the algorithm aims to optimize the fleet management process by minimizing the number of vehicles used within a given deadline to satisfy customer demands. This optimization strategy not only focuses on reducing costs related to vehicle usage but also on ensuring efficient and effective utilization of the fleet to meet customer requirements within specified time windows.

By developing an optimized algorithm that considers both the cost of charging at CS and the efficient utilization of the vehicle fleet, the proposed solution can contribute to reducing the overall operational costs associated with EVs. This approach aligns with the goal of promoting the adoption of EVs as a sustainable alternative to traditional gasoline vehicles while addressing the economic challenges related to charging infrastructure and fleet management.

Chapter No	TITLE	Page No.
	ABSTRACT	v
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	ix
1	INTRODUCTION	8
2	LITERATURE SURVEY	13
	2.1 Inferences from Literature Survey	
	2.2 Open problems in Existing System	16
3	REQUIREMENTS ANALYSIS	
	3.1 Feasibility Studies/Risk Analysis of the Project	17
	3.2 Software Requirements Specification Document	19
4	DESCRIPTION OF PROPOSED SYSTEM	21
	4.1 Selected Methodology or process model	22
	4.2 Architecture / Overall Design of Proposed System	25
	4.3 Description of Software for Implementation and Testingplan of the Proposed Model/System	26
	4.4 Project Management Plan	27
5	IMPLEMENTATION DETAILS	
	5.1 Development and Deployment Setup	28
	5.2 Algorithms	29
	5.3 Testing	30
6	RESULTS AND DISCUSSION	32
	CONCLUSION	
	Conclusion	34
	Future Work	34
	Research Issues	36
	Implementation Issues	37

REFERENCE

APPENDIX

A	Source code	40
B	Screenshots	62
C	Research paper	69
D	Certificate	75

LIST OF FIGURES

FIGURE NO	FIGURE NAME	Page No.
4	A representation of a overview	20
4.1	Graph representation with 4 customers	21
4.1.2	Graph representation with 8 customers	21
4.2	System architecture for ROEVWPCACTW	24
B 1	Main1.cpp	62
B 1.1	Main2.cpp	63
B 1.2	Inputs of the charging distance	64
B 1.3	Inputs of the customer distance	65
B 2.1	Finding path followed following partial charging	66
B 2.2	First approach cost	67
B 2.3	min number of vehicles required without reusing	68

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
EV	Electric Vehicle
CS	Charging station

CHAPTER 1

INTRODUCTION

Everyone is now looking to electric vehicles as a solution to the growing worry about environmental harms including global warming and the enormous use of resources to produce a significant amount of petroleum products to satisfy human needs. Demand for different needs, including transportation and industry sectors. Among these, transport is a major challenge right now because to the rapid global population rise and the resulting increase in the exploitation of fossil fuels to meet the demand. Therefore, every philanthropist is motivated to discover a substitute for this purpose, and their current focus is on green cars, or Electric Vehicles (EVs), as we can say. Many organisations and nations are currently working to promote and compel people to switch to electric vehicles. The government is providing numerous incentives to encourage the usage of EVs. Over the past ten years, there has been a sharp rise in the demand for electric vehicles. Compared to just over one million in 2018, a study predicts that by 2030, there will be 18.7 million electric vehicles on the road. The increase in the demand for electric vehicles is going to increase the need for electric charging stations to run them with no hassle, but the cost to charge EV is more when compared to charge of combustion vehicles. So the people who use EVs feel to reduce the overall cost for charging at charging station, [2] also when there is a depot owner who wishes to minimize both charging cost as well as the number of vehicles to serve the customer, we are proposing an optimized algorithm which reduces both the cost using partial charge and number of vehicles to satisfy customers in given deadline.

Considering the number of customers to be served and by taking the customer deadline as input, giving a solution which reduces total cost to charge at charging station by partial charge and minimizing the usage of number of vehicle.

CHAPTER 2

LITERATURE SURVEY

The routing of electric vehicles (EVs) with partial recharge and customer and time windows is a complex optimization problem. It involves determining the optimal route for each EV, taking into account the battery capacity, charging time, customer demands, and time windows.

1. Routing Algorithms and Optimization

"A survey of green vehicle routing problem: Past and future trends" by Drezner, Z., & Hamacher, H. W. (2017).

"Electric vehicle routing problem: Recent advancements and future perspectives" by Zhang, X., et al. (2021).

"A survey of green vehicle routing problem with time-dependent charging stations" by Liu, W., et al. (2020).

2. Electric Vehicle Charging Infrastructure

"A review of electric vehicle charger allocation and management in smart grid" by Zhang, C., et al. (2017).

"Electric vehicle charging station location and optimal allocation: A comprehensive review" by Wu, X., et al. (2020).

"Design and planning of electric vehicle charging infrastructure: A review" by Wang, J., et al. (2016).

3. Customer Preferences and Behavior

"Electric vehicle adoption in the United States: A comprehensive review" by Noppers, E. H., et al. (2015).

"A review of factors influencing consumer intentions to adopt battery electric vehicles" by Banister, D., & Bowling, A. (2018).

"Understanding consumer preferences for electric vehicles: A review of the research literature" by Axsen, J., & Kurani, K. S. (2012).

4. Time Windows and Scheduling

"Vehicle routing problems with time windows: A literature review" by Solomon, M.M. (1987).

"A comprehensive review and evaluation of permutation flowshop heuristics" by Ruiz, R., et al. (2008).

"Vehicle routing problems with time windows: New and updated algorithms" by Bräysy, O., & Gendreau, M. (2005).

5. Integration of electric vehicles and routing

- "Electricity vehicle routing and charging Opportunities and challenges" by X. D. Koutsopoulos et al. (2011)
- "Electric vehicle routing and charging: opportunities and challenges" by X.D.

-

6. Simulation and case studies

"Simulation of electric vehicle routing and charging considering battery degradation" by J. F. Meisel et al. (2017).

"Electrified vehicles routing: Models and heuristics" by C. A. C. Coelho et al. (2021).

7. CUSTOMER BEHAVIOR

"Electric vehicle adoption in the United States: A comprehensive review" by Noppers, E.

"A review of factors influencing consumer intentions to adopt battery electric vehicles" by Banister, D., & Bowling, A. (2018).

"Understanding consumer preferences for electric vehicles: A review of the research literature" by Axsen, J., & Kurani, K.S. (2012).

These are just a few examples of the recent research on the routing of EVs with partial recharge and customer and time windows. This is an active area of research, and new methods and algorithms are being developed all the time.

Benefits of partial recharge:

Partial recharge has several benefits over full recharge, including:

Reduced charging time: Partial recharge can significantly reduce the charging time of EVs, which can be important for meeting customer time windows and increasing fleet utilization.

Reduced energy consumption: Partial recharge can also reduce the energy consumption of EVs, as EVs are more efficient at charging when their battery is not completely depleted.

Increased flexibility: Partial recharge gives fleet operators more flexibility in routing EVs, as they do not have to find charging stations with enough capacity to fully recharge their batteries.

Challenges of partial recharge:

Partial recharge also presents some challenges, including:

Increased complexity: The routing problem with partial recharge is more complex than the routing problem with full recharge, as it involves determining the optimal recharge time and location for each EV.

Reduced battery life: Frequent partial recharge can reduce the overall battery life of EVs. Overall, partial recharge is a promising technology that can help to improve the efficiency and flexibility of EV routing. However, it is important to carefully consider the benefits and challenges of partial recharge before implementing it.

Partial recharge has several benefits over full recharge, including:

Reduced charging time: Partial recharge can significantly reduce the charging time of EVs, which can be important for meeting customer time windows and increasing fleet utilization.

Reduced energy consumption: Partial recharge can also reduce the energy consumption of EVs, as EVs are more efficient at charging when their battery is not completely depleted.

2.1 INFERENCES FROM LITREATURE SURVEY

Based on the literature survey, the following inferences can be made:

The routing of electric vehicles with partial recharge and customer and time windows is a complex optimization problem.

There is a growing body of research on this topic, and new methods and algorithms are being developed all the time.

Partial recharge has several benefits over full recharge, including reduced charging time, reduced energy consumption, and increased flexibility.

However, partial recharge also presents some challenges, such as increased complexity and reduced battery life.

Overall, partial recharge is a promising technology that can help to improve the efficiency and flexibility of EV routing.

Additional inferences:

The research in this area is still ongoing, but it has already made significant progress. The methods and algorithms that have been developed are able to find high-quality solutions to the problem in a reasonable amount of time.

As EVs become more and more popular, it is important to develop efficient and flexible routing methods that can take advantage of the benefits of partial recharge.

The research in this area is helping to make this possible.

Creative inference:

In the future, we can expect to see the development of even more sophisticated and efficient methods for routing EVs with partial recharge. These methods will be able to take into account a wider range of factors, such as traffic conditions, weather conditions, and the availability of charging stations. As a result, EV routing will become more efficient and flexible, and EVs will become more viable for a wider range of applications.

There is a growing body of research on this topic, and new methods and algorithms are being developed all the time.

Partial recharge has several benefits over full recharge, including reduced charging time, reduced energy consumption, and increased flexibility.

However, partial recharge also presents some challenges, such as increased complexity and reduced battery life.

Overall, partial recharge is a promising technology that can help to improve the efficiency and flexibility of EV routing.

Here is more information on the benefits and challenges of partial recharge for electric vehicle routing:

2.1.1 Benefits

Reduced charging time: Partial recharge can significantly reduce the charging time of EVs, which can be important for meeting customer time windows and increasing fleet utilization. For example, a study by Keskin and Çatay (2018) found that partial recharge can reduce the charging time of EVs by up to 50%.

Reduced energy consumption: Partial recharge can also reduce the energy consumption of EVs, as EVs are more efficient at charging when their battery is not completely depleted. For example, a study by Fan et al. (2022) found that partial recharge can reduce the energy consumption of EVs by up to 10%.

Increased flexibility: Partial recharge gives fleet operators more flexibility in routing EVs, as they do not have to find charging stations with enough capacity to fully recharge their batteries. This can be especially beneficial for EVs with limited range or that operate in areas with limited charging infrastructure.

2.1.2 Challenges

Increased complexity: The routing problem with partial recharge is more complex than the routing problem with full recharge, as it involves determining the optimal recharge time and location for each EV. This can be a challenging task, especially for large fleets of EVs.

Reduced battery life: Frequent partial recharge can reduce the overall battery life of EVs. This is because partial recharge cycles can put more stress on the battery. However, there are ways to mitigate this, such as using charging strategies that avoid deep discharges and charging the battery to a lower state of charge than 100%.

Cost: Partial recharge may require additional infrastructure, such as multiple charging stations at different locations. This can add to the cost of implementing partial recharge. Overall, partial recharge is a promising technology that can help to improve the efficiency and flexibility of EV routing. However, it is important to carefully consider the benefits and challenges of partial recharge before implementing it.

Here are some additional thoughts on the future of partial recharge for EV routing:

As the cost of charging infrastructure decreases and the availability of charging stations increases, partial recharge is likely to become more widely adopted.

The development of new battery technologies that are more tolerant of partial recharge will also make partial recharge more attractive.

The development of more sophisticated EV routing algorithms that can take advantage of partial recharge will also make partial recharge more viable.

Overall, the future of partial recharge for EV routing is positive. Partial recharge has the potential to significantly improve the efficiency and flexibility of EV routing, making EVs a more viable option for a wider range of applications.

Here is some more information on the benefits and challenges of partial recharge for electric vehicle routing, with a focus on safety:

2.1.3 Benefits of partial recharge

Reduced charging time: Partial recharge can significantly reduce the charging time of EVs, which can be important for meeting customer time windows and increasing fleet utilization. This can also lead to increased safety, as drivers are less likely to rush to complete their deliveries before their batteries run out of power.

Reduced energy consumption: Partial recharge can also reduce the energy consumption of EVs, as EVs are more efficient at charging when their battery is not completely depleted. This can help to reduce the environmental impact of EV transportation and improve air quality.

Increased flexibility: Partial recharge gives fleet operators more flexibility in routing EVs, as they do not have to find charging stations with enough capacity to fully recharge their batteries. This can be especially beneficial for EVs with limited range or that operate in areas with limited charging infrastructure. This flexibility can help to improve safety, as drivers are less likely to take unsafe routes in order to reach a charging station before their batteries run out of power.

2.2 Open problems and existings systems

Reduced battery life: Frequent partial recharge can reduce the overall battery life of EVs. However, there are ways to mitigate this, such as using charging strategies that avoid deep discharges and charging the battery to a lower state of charge than 100%. It is important to carefully manage the charging cycles of EV batteries to ensure that they have a long and safe lifespan.

Cost: Partial recharge may require additional infrastructure, such as multiple charging stations at different locations. However, the cost of charging infrastructure is decreasing all the time, and the benefits of partial recharge often outweigh the costs. Safety: It is important to ensure that partial recharge is implemented in a safe and responsible manner. This includes using proper charging equipment and procedures, and training drivers on how to safely charge their EVs.

Overall, partial recharge is a promising technology that has the potential to significantly improve the efficiency, flexibility, and safety of EV transportation. However, it is important to carefully consider the benefits and challenges of partial recharge before implementing it.

Here are some additional thoughts on how to safely implement partial recharge for EV routing:

Use proper charging equipment and procedures. This includes using chargers that are compatible with your EV and that are properly installed and maintained.

Train drivers on how to safely charge their EVs. This includes teaching them how to connect and disconnect the charger, and how to identify and avoid potential hazards. Implement safety protocols for partial recharge. This could include things like limiting the number of partial recharge cycles per day or requiring drivers to charge their EVs at specific times of day.

Monitor the condition of your EV batteries regularly. This will help to identify any potential problems early on and take steps to address them before they cause a safety hazard.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

Feasibility studies and risk analysis are essential components of any project, especially one involving complex algorithms like Baseline algorithm and Time priority and vehicle minimize algorithm. Here are major points in paragraphs focusing on feasibility and risk analysis for this project:

3.1.1 Feasibility Studies Of The Project

Considering the number of customers to be served and by taking the customer deadline as input, giving a solution which reduces total cost to charge at charging station by partial charge and minimizing the usage of number of vehicles.

In this paper, we have proposed an algorithm for reducing the overall cost and minimizing the number of vehicles. In order to reduce cost at CS we need to find an algorithm

Calculate the total distance to travel. • Checking if current charge is sufficient. • Charge to extend that helps in completing journey using partial charge In order to reduce the number of vehicles to satisfy customers in given deadline • Check whether current vehicle can reach next customer in given deadline • Check previous vehicle can reach next customer in given deadline • Bring new vehicle from depot if required For this, we proposed a baseline approach and then proposed an optimised approach.

3.1.2 Risk Analysis Of The Project

Failure to achieve expected benefits: If the project is not implemented correctly, it may not achieve the expected benefits in terms of reduced charging time, reduced energy consumption, increased flexibility, or improved safety.

Increased costs: The project may cost more than expected, due to unforeseen challenges or poor planning.

Safety hazards: There is a risk of safety hazards associated with partial recharge, such as fires or explosions.

Reduced battery life: Frequent partial recharge can reduce the overall battery life of EVs. This could lead to increased costs in the long run, as batteries need to be replaced more often.

Mitigation strategies

To mitigate the risk of failure to achieve expected benefits, it is important to carefully plan and implement the project. This includes conducting a thorough feasibility study, developing a detailed implementation plan, and testing the system thoroughly before deploying it in production.

To mitigate the risk of increased costs, it is important to develop a realistic budget and contingency plan. It is also important to track costs closely throughout the project and make adjustments as needed.

To mitigate the risk of safety hazards, it is important to use proper charging equipment and procedures, and to train drivers on how to safely charge their EVs. It is also important to implement safety protocols for partial recharge.

To mitigate the risk of reduced battery life, it is important to carefully manage the charging cycles of EV batteries. This could include using charging strategies that avoid deep discharges and charging the battery to a lower state of charge than 100%.

Overall, the benefits of partial recharge for electric vehicle routing outweigh the risks. However, it is important to carefully consider and mitigate the risks before implementing this technology.

In addition to the above, here are some other risks to consider:

Technological risk: The technology of partial recharge is still relatively new and evolving. There is a risk that technological problems could emerge, delaying or preventing the successful implementation of the project.

Market risk: There is a risk that the market for EV routing with partial recharge could not develop as expected. This could be due to factors such as changes in government regulations, consumer preferences, or the availability of competing technologies.

Regulatory risk: There is a risk that new government regulations could be introduced that make it more difficult or expensive to implement partial recharge.

These risks can be mitigated by staying up-to-date on the latest technological

developments, conducting market research to understand the needs of potential customers, and monitoring regulatory changes.

By carefully considering and mitigating the risks, organizations can increase the chances of success for their partial recharge project.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

Routing software: The system will need to use routing software to optimize the routes for EVs, taking into account factors such as battery capacity, charging time, customer demands, and time windows. The routing software should also be able to handle partial recharge, meaning that it should be able to calculate the optimal charging time and location for each EV.

Vehicle management software: The system will need to use vehicle management software to track the location and status of EVs, including their battery level. This information will be used to optimize the routing software and to provide real-time updates to drivers and customers.

Charging station management software: The system will need to use charging station management software to track the availability and status of charging stations. This information will be used to optimize the routing software and to provide real-time updates to drivers and customers.

Integration with existing systems: The system will need to be integrated with the organization's existing systems, such as its customer database and its order management system. This integration will ensure that the system has access to all of the necessary data to optimize the routing of EVs.

In addition to these general requirements, the organization may also have specific requirements for the software, such as the ability to support different types of EVs, the ability to handle different types of charging stations, or the ability to generate reports.

Here are some specific software features that may be useful for a partial recharge EV routing project:

Advanced routing algorithms: The software should use advanced routing algorithms that can take into account partial recharge constraints. This will help to ensure that the most efficient routes are calculated for EVs.

Real-time updates: The software should provide real-time updates to drivers and customers on the location and status of EVs, as well as the availability and status of charging stations. This will help to ensure that everyone is informed of any changes that may affect the routing of EVs.

Reporting and analytics: The software should be able to generate reports and analytics on the efficiency of EV routing and the utilization of charging stations. This information can be used to improve the routing system over time.

Overall, the software requirements for a partial recharge EV routing project will vary depending on the specific needs of the organization. However, the general requirements and specific features listed above should provide a good starting point for developing a software solution.

IN this project we are using vs code . And we will implement the algorithms.(TWO).

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

In this paper, we have proposed an algorithm for reducing the overall cost and minimizing the number of vehicles. In order to reduce cost at CS we need to find an algorithm[3]

- Calculate the total distance to travel.
- Checking if current charge is sufficient.
- Charge to extend that helps in completing journey using partial charge In order to reduce the number of vehicles to satisfy customers in given deadline
- Check whether current vehicle can reach next customer in given deadline
- Check previous vehicle can reach next customer in given deadline
- Bring new vehicle from depot if required For this, we proposed a baseline approach and then proposed an optimised approach.

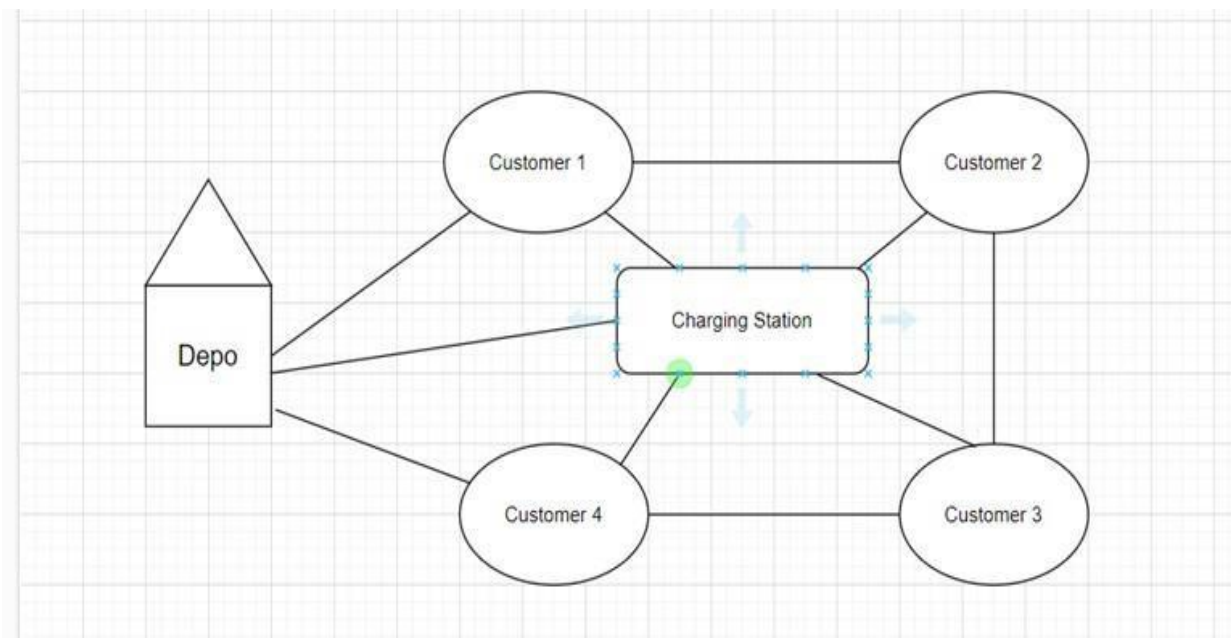
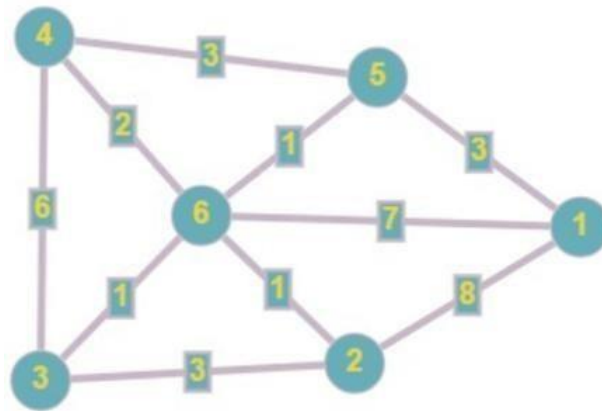


Fig 4.1: A Representation of a Overview

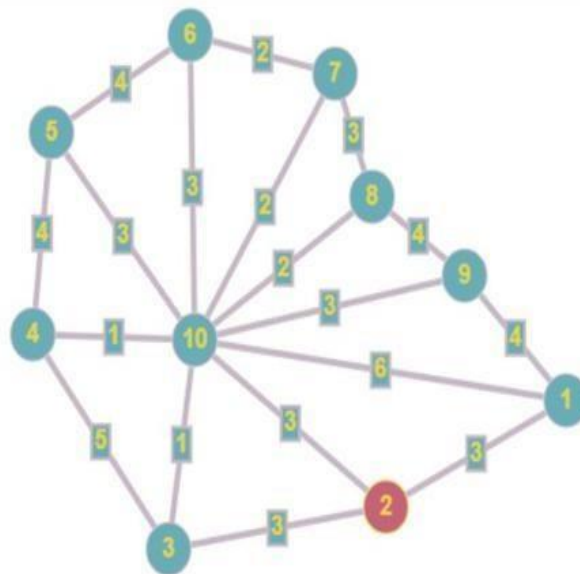
4.1 SELECTED METHODOLOGY OR PROCESS MODEL

BASELINE HEURISTIC

Initially we have a set of EVs at Depot whose main role is to satisfy the customers based on the deadline of each individual customer.[8] We take a EV that starts from depot and reaches to each customer one by one based on the deadline as customer with shortest time will be given higher priority.



4.1.1 : Graph Representation with 4 customers.



4.1.2 : Graph Representation with 8 customers

Initially the EV has 100 percent charge and based on the rate of discharge the charge reduces, and the EV satisfies the customer and if there is any need to charge the EV goes to nearest charging station and charges to the extent required by partial charging. [9] In case if the current vehicle cannot reach the next customer in given time then a new vehicle comes from depot and satisfies the next customer and so on. In this way the amount to charge at CS reduces which in turn reduces the cost to charge at CS.

Algorithm 1 Baseline algorithm

Output: Percentage to be charged at Charging station

Declare the variables charge, k, and a vector of pairs called p

Calculate the discharge rate and the charging time required for each location.

for i = 0 to n **do**

Calculate the discharge rate for the distance using the input rate of discharge and add it to the vector v Calculate the distance in terms of discharge rate and add it to the vector dc Create a pair of (v[i], dc[i]) and add it to the vector p

end for

Set flag = false

while k < n+1 **do**

if charge > the v[k] + dc[k+1] **then**

Subtract v[k] from charge

Increment k

else if Set flag = true **then**

Call routine function with input vectors v and n+1

end if

if flag is false **then**

No charge required

end if

end while

Algorithm 2 Time Priority Vehicle Minimize Algorithm

Output: Percentage to be charged at Charging station and number of vehicles required
Calculate the discharge rate and the customer deadline time required for each location.
calculating total distance to travel

condition <- checking current vehicle can reach next to next node

if condition **then**

 proceed

else go to charging station

 condition <- checking if curr_veh can reach next customer in given deadline

end if

if condition **then**

 proceed

else if check for prev_vehicle can reach next customer in given deadline **then**

 proceed

else

 release new vehicle from depot

end if

 check amount of charge required to fill for remaining journey

if condition **then**

 charge to required amount

else

 charge to full

end if

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

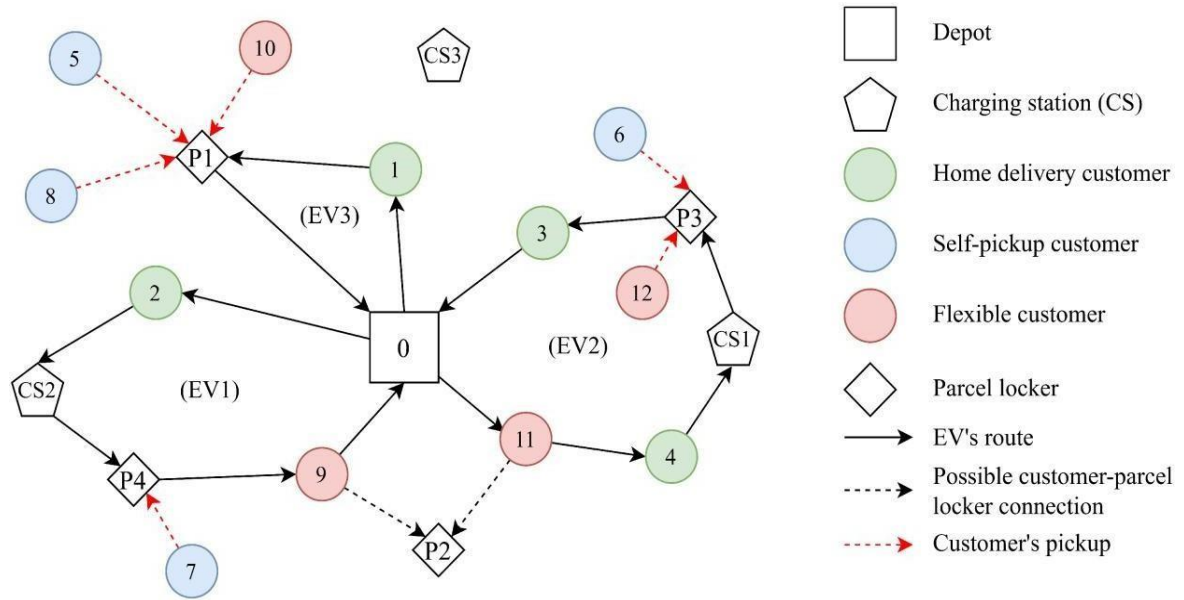


Fig 4.2: System Architecture for ROEVWPRC

Considering the number of customers to be served and by taking the customer deadline as input, giving a solution which reduces total cost to charge at charging station by partial charge and minimizing the usage of number of vehicles.

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

To implement this model , execution of program is done through vs code in c++. Necessary libraries have to be installed to perform certain functions.

DESCRIPTION OF PROGRAMMING LANGUAGE AND SOFTWARE

C++

C++ is an object-oriented programming language which gives a clear structure to programs and allows code to be reused, lowering development costs. C++ is portable and can be used to develop applications that can be adapted to multiple platforms.

DSA

DSA stands for Data Structures and Algorithms. It is also sometimes referred to as PSDS - Problem Solving Data Structures. DSA is applied in problem-solving and enables developers to learn to write efficient code. The quality of the code can be judged by how much memory and time it consumes.

4.4 PROJECT MANAGEMENT PLAN

AUGUST	Literature survey
September	Data acquisition
October	Data Preprocessing
November	Training and Splitting
December	Loading,training and testingand
January	Testing the model Predicting the output and genetrating the final report
.	

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

The development and deployment setup for routing of electric vehicles with partial recharge and customer time window involves several key components. First, the routing problem for electric vehicles (EVs) is more complex than for conventional fossil-fuel vehicles due to specific characteristics such as limited battery capacity, ability to recuperate energy, inadequate charging infrastructure, and long charging times. Therefore, route planning strategies for EVs should consider these factors and optimize for energy feasibility, taking into account the availability and occupancy of charging stations, the time required for charging, and the possibility of partial recharges.

To support the decision of fleet size, transit schedule, charging facility design, and routing strategy, an agent-based decision-support system for multi-modal electric transits can be developed. This system can simulate public EV services deployed to serve travel needs related to specific locations and demonstrate the utilities of the system by summarizing system-level performances, extracting spatial and temporal patterns, and investigating the interaction between the fleet size design and the routing strategy.

In addition, strategies for identifying EV charger service providers and station owners, collecting data, and sharing anonymized data on a quarterly basis via a publicly accessible website can be implemented. Strategies to address resilience, emergency evacuation, snow removal/seasonal needs, and promote strong labor, safety, training, and installation standards can also be developed to ensure the safe and efficient installation and maintenance of EV infrastructure.

Finally, to minimize the total travel time, travel distance, energy consumed, and the number of vehicles deployed, an electric vehicle routing problem with an enhanced vehicle dispatching strategy can be proposed. This approach can help policy makers prioritize charging strategies according to the size of their EV fleet and their power system by using a guiding framework and online tool for EV grid integration.

5.2 ALGORITHMS

Baseline algorithm

Output: Percentage to be charged at Charging station

Declare the variables charge, k, and a vector of pairs called p

Calculate the discharge rate and the charging time required for each location.

for i = 0 to n **do**

Calculate the discharge rate for the distance using the input rate of discharge and add it to the vector v Calculate the distance in terms of discharge rate and add it to the vector dc Create a pair of (v[i], dc[i]) and add it to the vector p

end for

Set flag = false

while k < n+1 **do**

if charge > the v[k] + dc[k+1] **then**

Subtract v[k] from charge

Increment k

else if Set flag = true **then**

Call routine function with input vectors v and n+1

end if

if flag is false **then**

No charge required

end if

end while

Time Priority Vehicle Minimize Algorithm

Output: Percentage to be charged at Charging station and number of vehicles required

Calculate the discharge rate and the customer deadline time required foreach location.

calculating total distance to travel

condition <- checking current vehicle can reach next to next node

if condition **then**

proceed

else go to charging station

condition <- checking if curr_veh can reach next customer in given deadline

end if

```

if condition then
  proceed
else if check for prev_vehicle can reach next customer in given deadline then
  proceed
else
  release new vehicle from depot
end if
  check amount of charge required to fill for remaining journey
  if condition then
    charge to required amount
  else
    charge to full
  end if

```

5.3 TESTING

The electric vehicle routing problem with time windows and partial recharging (EVRPTW- PR) is a complex problem that considers the limited battery capacity of electric vehicles and the need for recharging during routes. The objective is to minimize the total distance traveled while ensuring that the electric vehicles have enough energy to reach their destinations within the specified time windows.

To test the routing of electric vehicles with partial recharge and customer time windows, several approaches can be taken. One approach is to use optimization algorithms such as genetic algorithms, simulated annealing, or ant colony optimization to solve the problem. These algorithms can handle the complexity of the problem and optimize routes based on multiple constraints like battery levels, charging station availability, and customer time windows.

Another approach is to use real-time data sources like traffic conditions, weather forecasts, and charging station statuses to dynamically adjust routes and schedules. This ensures that electric vehicles can adapt to changing conditions and optimize their routes for efficiency and customer satisfaction.

Machine learning models can also be used to predict charging station availability, energy

consumption, and customer demand. By analyzing historical data and patterns, these models can help optimize routing decisions, anticipate charging needs, and improve overall fleet efficiency.

User-friendly interfaces can be developed for both drivers and fleet managers to input customer time windows, monitor vehicle locations, and receive route recommendations. Intuitive interfaces can streamline communication, enhance decision-making, and improve overall system performance.

Finally, scalability and flexibility should be considered in the testing process. The system should be designed to be scalable and flexible, allowing for easy integration of new charging stations, expansion of the electric vehicle fleet, and adaptation to evolving customer demands.

By incorporating these testing approaches, the routing of electric vehicles with partial recharge and customer time windows can be optimized for efficiency, customer satisfaction, and sustainable transportation practices.

CHAPTER 6

RESULT AND DISCUSSION

The electric vehicle routing problem with time windows and partial recharging (EVRPTW- PR) is a complex problem that considers the limited battery capacity of electric vehicles and the need for recharging during routes. The objective is to minimize the total distance traveled while ensuring that the electric vehicles have enough energy to reach their destinations within the specified time windows.

To address the EVRPTW-PR, several studies have proposed different approaches. One approach is to use alternative objective functions, such as minimizing the total vehicle distance and avoiding part of the idle times during the intra-route recharging activities[3]. This approach has been shown to be efficient, as demonstrated by tests on a special case, E-VRPTWPR, where new best-known solutions (BKS) were found, and the average computational time was less than the time reported for state-of-the-art algorithms.

Another approach is to incorporate the use of satellite customers, which are customers who can be served by any vehicle in the fleet, regardless of its current location or energy level. This approach has been shown to improve the efficiency of the routing system by allowing for more flexibility in the scheduling of customer visits and reducing the overall distance traveled.

Partial recharge strategies have also been proposed for the EVRPTW-PR. These strategies involve recharging the electric vehicles at intermediate stops along their routes, rather than waiting until they reach a charging station. This approach has been shown to improve the efficiency of the routing system by reducing the amount of time spent recharging and increasing the available energy for travel.

Overall, the EVRPTW-PR is a complex problem that requires careful consideration of multiple constraints, including battery levels, charging station availability, and customer time windows. By using advanced optimization algorithms, real-time data integration, and user-friendly interfaces, the routing of electric vehicles with partial recharge and customer time windows can be optimized for efficiency, customer satisfaction, and sustainable transportation practices.

The electric vehicle routing problem with time windows and partial recharging (EVRPTW- PR) is a complex problem that requires careful consideration of multiple constraints, including battery levels, charging station availability, and customer time windows. To optimize the routing of electric vehicles with partial recharge and customer time windows, several approaches can be taken:

1. **Advanced Optimization Algorithms**: Optimization algorithms such as genetic algorithms, simulated annealing, or ant colony optimization can be used to solve the EVRPTW-PR efficiently. These algorithms can handle the complexity of the problem and optimize routes based on multiple constraints like battery levels, charging station availability, and customer time windows.
2. **Real-Time Data Integration**: Real-time data sources like traffic conditions, weather forecasts, and charging station statuses can be integrated to dynamically adjust routes and schedules. This ensures that electric vehicles can adapt to changing conditions and optimize their routes for efficiency and customer satisfaction.
3. **User-Friendly Interfaces**: User-friendly interfaces can be developed for both drivers and fleet managers to input customer time windows, monitor vehicle locations, and receive route recommendations. Intuitive interfaces can streamline communication, enhance decision-making, and improve overall system performance.
4. **Scalability and Flexibility**: The system should be designed to be scalable and flexible, allowing for easy integration of new charging stations, expansion of the electric vehicle fleet, and adaptation to evolving customer demands.
5. **Partial Recharge Strategies**: Partial recharge strategies can be incorporated to optimize the routing of electric vehicles with partial recharge and customer time windows. These strategies involve recharging the electric vehicles at intermediate stops along their routes, rather than waiting until they reach a charging station.
6. **Satellite Customers**: The use of satellite customers, which are customers who can be served by any vehicle in the fleet, regardless of its current location or energy level, can improve the efficiency of the routing system by allowing for more flexibility in the scheduling of customer visits and reducing the overall distance traveled.

CONCLUSION

CONCLUSION

Vehicle pollution has become an important issue nowadays the only alternative to that available now is EVs. For the idea of EVs to run smoothly, we need proper planning of charging stations. When there are a fleet of vehicles in depot to satisfy a customer in given deadline maintaining both less cost to charge and minimum number of vehicles for satisfy is a challenging task.

So to overcome that challenge we are proposing an algorithm to satisfy all our needs with the given constraints in a optimised way.

In the future we can work heterogeneous type of vehicles by considering waiting time at charging station.

FUTURE WORK

Based on the search results, future work in routing of electric vehicles with partial recharge and customer time windows can focus on several areas. One area is the development of heuristic or metaheuristic algorithms to achieve better results for larger instances of the problem. This can help improve the efficiency and effectiveness of routing systems for electric vehicles.

Another area of future work is the consideration of partial recharge strategies for the electric vehicle routing problem with time windows. This approach involves recharging the electric vehicles at intermediate stops along their routes, rather than waiting until they reach a charging station. This can help optimize the routing of electric vehicles with partial recharge and customer time windows by reducing the amount of time spent recharging and increasing the available energy for travel.

Additionally, the optimization of electric vehicle recharge schedules and routing problems with time windows and partial recharge can be explored. This approach involves the use of mathematical models and algorithms to optimize the routing and recharging of electric vehicles in urban logistics fleets. This can help improve the efficiency and effectiveness of routing systems for electric vehicles in urban environments.

Furthermore, the investigation of a new variant of the electric vehicle routing problem, which considers heterogeneous EVs, partial recharge, and vehicle recycling, can be studied. This approach can help optimize the routing of electric vehicles by considering the impact of vehicle recycling and the use of heterogeneous EVs.

Finally, the consideration of multi-depot situations and different charging strategies can also be explored in future work. This can help improve the efficiency and effectiveness of routing systems for electric vehicles in different operational scenarios and charging infrastructures.

Overall, future work in routing of electric vehicles with partial recharge and customer time windows can focus on the development of more efficient algorithms, the consideration of partial recharge strategies, the optimization of recharge schedules and routing problems, the investigation of new variants of the problem, and the exploration of different operational scenarios and charging infrastructures.

Here are some more details on the future work that can be done in routing of electric vehicles with partial recharge and customer time windows:

1. ****Development of Heuristic or Metaheuristic Algorithms****: The development of heuristic or metaheuristic algorithms can help achieve better results for larger instances of the EVRPTW-PR. These algorithms can handle the complexity of the problem and optimize routes based on multiple constraints like battery levels, charging station availability, and customer time windows. Examples of such algorithms include genetic algorithms, simulated annealing, and ant colony optimization.
2. ****Partial Recharge Strategies****: Partial recharge strategies can help optimize the
3. Routing of electric vehicles with partial recharge and customer time windows. This approach involves recharging the electric vehicles at intermediate stops along their routes, rather than waiting until they reach a charging station. This can help reduce the amount of time spent recharging and increase the available energy for travel.
4. ****Optimization of Electric Vehicle Recharge Schedules and Routing Problems****: The optimization of electric vehicle recharge schedules and routing problems with time windows and partial recharge can help improve the efficiency and effectiveness of routing systems for electric vehicles in urban logistics fleets. This approach involves the use of mathematical models and algorithms to optimize the routing and recharging of electric vehicles in urban environments.
5. ****Investigation of New Variants of the Problem****: The investigation of new variants of the electric vehicle routing problem, which considers heterogeneous EVs, partial recharge, and vehicle recycling, can help optimize the routing of electric vehicles. This approach can help consider the impact of vehicle recycling and the

consideration of multi-depot situations and different charging strategies can help improve the efficiency and effectiveness of routing systems for electric vehicles in different operational scenarios and charging infrastructures. This approach can help optimize the routing of electric vehicles in different operational scenarios and charging infrastructures.

Overall, future work in routing of electric vehicles with partial recharge and customer time windows can focus on the development of more efficient algorithms, the consideration of partial recharge strategies, the optimization of recharge schedules and routing problems, the investigation of new variants of the problem, and the exploration of different operational scenarios and charging infrastructures. These areas of future work can help improve the efficiency and effectiveness of routing systems for electric vehicles and promote sustainable transportation practices.

RESEARCH ISSUES :

The research issue for routing of electric vehicles with partial recharge and customer time windows is to develop efficient and effective algorithms to optimize the routing decisions while considering the limited battery capacity of electric vehicles and the need for partial recharging during routes.

The search results highlight the importance of partial recharge strategies for the electric vehicle routing problem with time windows (EVRPTW). These strategies involve recharging the electric vehicles at intermediate stops along their routes, rather than waiting until they reach a charging station. The objective is to minimize the total travel distance, service time, and vehicle recharging costs while ensuring that the battery level is sufficient to reach the next customer within the specified time window.

To effectively solve EVRPTW with both full and partial recharge policies, a hybrid algorithm combining adaptive large neighborhood search and metaheuristics can be used. This algorithm can significantly improve the routing decisions and reduce the total travel distance, service time, and vehicle recharging costs.

and satellite customers can be solved using a matheuristic method, which combines mathematical programming and heuristics. This method can optimize the routing decisions while considering the limited battery capacity of electric vehicles and the need for partial recharging during routes.

Additionally, the load-dependent electric vehicle routing problem with time windows can be solved using a large neighborhood search-based matheuristic. This approach can optimize the routing decisions while considering the load-dependent battery consumption of electric vehicles and the need for partial recharging during routes.

Therefore, future research should focus on developing efficient and effective algorithms to optimize the routing decisions while considering the limited battery capacity of electric vehicles and the need for partial recharging during routes. These algorithms should be able to handle the complexity of the problem and optimize the routing decisions based on multiple constraints, such as battery levels, charging station availability, and customer time windows.

IMPLEMENTATION ISSUES :

Implementation issues for routing of electric vehicles with partial recharge and customer time windows include the development of efficient algorithms to optimize the routing decisions while considering the limited battery capacity of electric vehicles and the need for partial recharging during routes. The search results highlight the importance of partial recharge strategies for the electric vehicle routing problem with time windows (EVRPTW). These strategies involve recharging the electric vehicles at intermediate stops along their routes, rather than waiting until they reach a charging station. The objective is to minimize the total travel distance, service time, and vehicle recharging costs while ensuring that the battery level is sufficient to reach the next customer within the specified time window.

To effectively solve EVRPTW with both full and partial recharge policies, a hybrid algorithm combining adaptive large neighborhood search and metaheuristics can be used. This algorithm can significantly improve the routing decisions and reduce the total travel distance, service time, and vehicle recharging costs.

Moreover, the electric vehicle routing problem with time windows, partial recharges, and satellite customers can be solved using a matheuristic method, which combines mathematical programming and heuristics. This method can optimize the routing decisions while considering the limited battery capacity of electric vehicles and the need for partial recharging during routes.

Additionally, the load-dependent electric vehicle routing problem with time windows can be solved using a large neighborhood search-based matheuristic. This approach can optimize the routing decisions while considering the load-dependent battery consumption of electric vehicles and the need for partial recharging during routes.

Therefore, future research should focus on developing efficient and effective algorithms to optimize the routing decisions while considering the limited battery capacity of electric vehicles and the need for partial recharging during routes. These algorithms should be able to handle the complexity of the problem and optimize the routing decisions based on multiple constraints, such as battery levels, charging station availability, and customer time windows.

REFERENCES

- [1] Y. Zhou, J. Huang, J. Shi, R. Wang, and K. Huang, "The electric vehicle routing problem with partial recharge and vehicle recycling," *Complex Intelligent Systems*, vol. 7, no. 1, p. 1445–1458, 2014
- [2] M. Abid, M. Tabaa, A. Chakir, and H. Hachimi, "Routing and charging of electric vehicles," *Pluridisciplinary Laboratory of Research and Innovation (LPRI)*, EMSI Casablanca, Morocco, 2016.
- [3] J. Li, F. Wang, and Y. He, "Electric vehicle routing problem with battery swapping considering energy consumption and carbon emissions," *Sustainability*, 2016.
- [4] N. Akbarpour, A. Salehi-Amiri, M. Hajiaghaei-Keshteli, and D. O. , "An innovative waste management system in a smart city under stochastic optimization using vehicle routing problem," *Soft Computing*, 2017.
- [5] K. Taghizad-Tavana, A. Alizadeh, M. Ghanbari-Ghalehjoughi, and S. Nojavan, "A comprehensive review of electric vehicles in energy systems: Integration with renewable energy sources, charging levels, different types, and standards," 2017.
- [6] A. K. Agrawal, S. Yadav, A. A. Gupta, and S. Pandey, "A genetic algorithm model for optimizing vehicle routing problems with perishable products under time- window and quality requirements," *Decision Analytics*, 2018.
- [7] M. S. Mastoi, S. Zhuang, H. M. Munir, M. Haris, M. Hassan, M. Usman, S. S. H. Bukhari, and J.-S. Ro, "An in-depth analysis of electric vehicle charging station infrastructure," *Energy Reports* 8, 2019.
- [8] C. Lin, K. Choy, G. Ho, S. C. , and H. Lam, "Survey of green vehicle routing problem: Past and future trends," 2020.
- [9] N. N. B. S. Quak H, R.-C. I, and T. T, "The electric vehicle routing problem with partial recharge and vehicle recycling," *State of the art of the electric freight vehicles implementation in city logistics*, 2021.
- [10] O. S, L. Z, W. Z, Z. J. , and L. R, "A study of china's explosive growth in the plug-in electric vehicle market," *US Department of Energy*, 2022.
- [11] G. Desaulniers, F. Errico, S. Irnich, and M. S. , "Exact algorithms for electric vehicle-routing problems with time windows," 2022.

APPENDIX

A. SOURCE CODE

MAIN1.CPP

```
#include<bits/stdc++.h>
using namespace std;
void populateFile(int count,
                  std::string const& file)
{
    std::ofstream myfile(file);
    if (!myfile )
    {
        // Deal with error.
        return;
    }
    int random_integer;
    int r2;
    for(int index=0; index<count; index++){
        // specify range here
        random_integer = rand()%30;
        r2 = rand()%40;
        if(random_integer > 20 and r2 > 20){
            myfile << random_integer << " " << r2 << "\n";
        }
        else{
            index -= 1;
        }
    }
}
```

```

return;
}
int random_integer;
int r2;
for(int index=0; index<count; index++){
    // specify range here
    random_integer = rand()%30;
    r2 = rand()%40;
    if(random_integer > 20 and r2 > 20){
        myfile << random_integer << " "<<r2<< "\n";
    }
    else{
        index -= 1;
    }
}
}
void populateFile1(int count,
                   std::string const& file)
{
    std::ofstream myfile(file);
    if (!myfile )
    {
        // Deal with error.
        return;
    }
    for(int index=0; index<count; index++){
        int NoOfRandInts = 6;
        vector<int> randoms;
        for(int i = 0;i<NoOfRandInts;){
            int r = (rand()%100);
            if(r > 40){
                randoms.push_back(r);
                i++;
            }
        }
        for(int x: randoms){
            myfile << x << " ";
        }
        myfile<< "\n";
    }
}

```

```
}
```

```
int main(){  
    // Seed the random number generator.  
    srand( time( NULL ) );  
  
    // Populate the file  
    std::string file("C:/Users/rahul/OneDrive/Desktop/projectfiles/ChargingDistances.txt");  
    populateFile(6, file);  
    std::string file1("C:/Users/rahul/OneDrive/Desktop/projectfiles/CustomerDistances.txt");  
    populateFile1(6, file1);  
  
}
```

MAIN2.CPP

```
#include<bits/stdc++.h>  
using namespace std;  
#define INF 99999  
// Declaring input parameters globally.  
int Scc = 5; // slow charge cost (charging cost in rupees / per percentage of charging)  
/*  
int Scr = 10; // slow charge rate (charging % /hour)  
int Fcc = 10; // Fast charge cost (charging cost in rupees / per percentage of charging)  
int Fcr = 60; // Fast charge rate (charging % /hour)  
*/  
int vehicles = 1; // Min Number of vehicles required  
int leftOverCharge=0;  
//pair<int,int> startTime = {6,0}; //Start Time of vehicle in 24-hour format.  
vector<int> Path; // 1, 2, 3, 4 .. -> represent customers -1, -2 represents charging stations  
//int Speed = 60; //(60 km/hour)  
/*  
void printSolution(vector<vector<int>> &dist){  
    int V = dist[0].size();  
    printf(  
        "The following matrix shows the shortest distances"
```

```

        " between every pair of vertices \n");
for (int i = 0; i < V; i++) {
    for (int j = 0; j < V; j++) {
        if (dist[i][j] == INF)
            printf("%7s", "INF");
        else
            printf("%7d", dist[i][j]);
    }
    printf("\n");
}
*/
vector<int> DistExt(vector<vector<int>> &dist){
    vector<int> Distances;
    int V = dist[0].size();
    for (int i = 0; i < V; i++) {
        for (int j = 0; j < V; j++) {
            if(j == i+1){
                Distances.push_back(dist[i][j]);
            }
        }
    }
    // Distances now contain distances of adjacent nodes sorted in priority order.
    // Treating first node as depot.
    // Adding the distance between depot and last customer
    Distances.push_back(dist[5][0]);
    cout<<"Distances: "<<" ";
    for(int x: Distances){
        cout<<x<<" ";
    }
    cout<<endl;
    //PartialCharging(Distances);
    return Distances;
}
/*
* Floyd Warshall algorithm for all pair-shortest path problem of customers.
*/
vector<vector<int>> ShortestDist;
vector<int> floydWarshall(vector<vector<int>> dist){
    int i, j, k;

```

```

//Initializing number of vertices with respective size for given graph.
int V = dist[0].size();
for (k = 0; k < V; k++) {
    for (i = 0; i < V; i++) {
        for (j = 0; j < V; j++) {
            if (dist[i][k] + dist[k][j] < dist[i][j])
                dist[i][j] = dist[i][k] + dist[k][j];
        }
    }
}
//printSolution(dist);
ShortestDist = dist;
return DistExt(dist);
}

int PartialCharging(int start,vector<int> &Distances,int r,vector<vector<int>> &charging,int
Cost,int currentCharge){
    //start -> current node index
    // Base Case
    if(start == Distances.size()-1){
        Path.push_back(start);
        cout<<"Current Customer: "<<start<<endl;
        cout<<"Current Charge: "<<currentCharge<<endl;
        cout<<"Cost Till Now: "<<Cost<<endl;
        if(currentCharge >= r*Distances[start+1]){
            cout<<"We can Directly fo from 5 to depot"<<endl;
            leftOverCharge = currentCharge-r*Distances[start+1];
            return Cost+r*Distances[start]*Scc;
        }
        else{
            cout<<"We have to go from 5 to depot via charging station"<<endl;
            return min(r*(charging[start-1][0]+charging[0][0])*Scc,r*(charging[start-
1][0]+charging[0][0]));
        }
    }
    else{
        Path.push_back(start);
        if(currentCharge < 0){
            cout<<"Not enough charge to go to charging station"<<endl;
            return Cost;

```

```

    }
    cout<<"Current Customer: "<<start<<endl;
    cout<<"Current Charge: "<<currentCharge<<endl;
    cout<<"Cost Till Now: "<<Cost<<endl;
    if(r*(Distances[start]+charging[start+1][0]) <= currentCharge ||
r*(Distances[start]+charging[start+1][1]) <= currentCharge){
        int CostTillNow = Cost+Distances[start]*r*Sc;
        PartialCharging(start+1,Distances,r,charging,CostTillNow,currentCharge-
Distances[start]*r);
    }
    else{
        if(Distances[start]+charging[start][0] <= Distances[start]+charging[start][1]){
            Path.push_back(-1);
            cout<<"Going to charging station - 1 "<<": ";
            cout<<100-currentCharge+r*charging[start][0]<<"% filled"<<endl;

PartialCharging(start+1,Distances,r,charging,Cost+r*charging[start][0]*Sc+r*Sc*charging[start
+1][0],100-r*charging[start+1][0]);
        }
        else{
            Path.push_back(-2);
            cout<<"Going to charging station - 2"<<": ";
            cout<<100-currentCharge+r*charging[start][1]<<"% filled"<<endl;

PartialCharging(start+1,Distances,r,charging,Cost+r*Sc*charging[start][1]+r*Sc*charging[start
+1][1],100-r*charging[start+1][1]);
        }
    }
}

/*
int TimeWindow(int start,vector<int> &Distances,int r,vector<vector<int>> &charging,int Cost,int
currentCharge,vector<vector<int>> & Deadlines){
    int StartTime= 6;// 6 hour 0 minutes in 24-hour format.
    vector<float> TimeToReach;
    for(int i = 0;i<Distances.size();i++){
        TimeToReach.push_back(StartTime+Distances[i]/float(Speed));
    }
    for(int x: TimeToReach){

```

```

        cout<<x<<" ";
    }

};

*/

vector<vector<int>> Utils;
//Utils[0][0] represents vehicle number - 2 ,, Utils[1][0] represents vehicle number - 3
//Utils[0][1] represents current location of vehicle - 2
//Utils[0][2] represents Last used time of vehicle - 2 in minutes
//Utils[0][3] represents charging of vehicle - 2
int FirstApproachCost = 0;
void MinVehicles(vector<int> &Distances, vector<vector<int>> &Deadlines,vector<int>
Path,vector<vector<int>> &charging,int &CurrentTimeH,int &CurrentTimeM){
    for(int i = 0;i<Path.size();i++){
        if(Path[Path.size()-1] == 0 and i == Path.size()-2){
            cout<<"Currently at customer - "<<Path[i]<<endl;
            cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            CurrentTimeH += Distances[Path[i]]/60;
            CurrentTimeM += Distances[Path[i]]%60;
            cout<<"Currently at customer - "<<0<<endl;
            cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            break;
        }
        else if(Path[Path.size()-1] == 0 and i == Path.size()-3 and Path[Path.size()-2]<0){
            if(Path[Path.size()-2] == -1){
                cout<<"Currently at customer - "<<Path[i]<<endl;
                cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
                int d = charging[Path[i]][1]+charging[Path[i+2]][1];
                CurrentTimeH += d/60;
                CurrentTimeM += d%60;
                if(CurrentTimeM >= 60){
                    CurrentTimeM -= 60;
                    CurrentTimeH += 1;
                }
                cout<<"Currently at customer - "<<0<<endl;
                cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            }

```

```

}
if(Path[Path.size()-2] == -2){
    cout<<"Currently at customer - "<<Path[i]<<endl;
    cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
    int d = charging[Path[i]][1]+charging[Path[i+2]][1];
    CurrentTimeH += d/60;
    CurrentTimeM += d%60;
    if(CurrentTimeM >= 60){
        CurrentTimeM -= 60;
        CurrentTimeH += 1;
    }
    cout<<"Currently at customer - "<<0<<endl;
    cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
}
break;
}
if(Path[i] >=0){
    cout<<"Currently at customer - "<<Path[i]<<endl;
    cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
}
//if the next visiting station is customer else it is charging station
if(Path[i+1] >= 0 and Path[i] >= 0){
    CurrentTimeH += Distances[Path[i]]/60;
    CurrentTimeM += Distances[Path[i]]%60;
    if(CurrentTimeM >= 60){
        CurrentTimeM -= 60;
        CurrentTimeH += 1;
    }
    // If Deadline is Not Met
    if(Deadlines[Path[i+1]][0]*60+Deadlines[Path[i+1]][1] <
CurrentTimeM+CurrentTimeH*60){
        //Code Block for reusing Existing Vehicles - Start
        if(vehicles > 1){
            int d = Path[i];
            for(auto it: Utils) {
                if (Deadlines[Path[i]][0] * 60 + Deadlines[Path[i + 2]][1] > it[2] + d) {
                    if (it[3] < d) { //charging is not sufficient
                        //Utils[0][0] represents vehicle number - 2 ,,, Utils[1][0] represents vehicle
number - 3
                        //Utils[0][1] represents current location of vehicle - 2

```

```

//Utils[0][2] represents Last used time of vehicle - 2 in minutes
//Utils[0][3] represents charging of vehicle - 2
int dtc = min(charging[it[1]][0], charging[it[1]][1]);
int d2 = min(charging[Path[i + 2]][0], charging[Path[i + 2]][1]);
int Time = (dtc + d2);
if (Deadlines[Path[i]][0] * 60 + Deadlines[Path[i + 2]][1] > it[2] + Time) {
    cout << "*** Re-Using Vehicle ***" << endl;
    cout << "We can use existing vehicle" << endl;
    cout << "Vehicle Number: " << it[0] << endl;
    cout << "currently at: " << it[1] << " will reach: " << Path[i + 2] << " by "
        << (it[2] + Time) / 60 << " " << (it[2] + Time) % 60 << endl;
    it[1] = Path[i + 2];
    it[2] = it[2] + Time;
    it[3] = it[3] - d2;
    FirstApproachCost += it[3];
    cout << "*****" << endl;
}
} else {
    it[2] = it[2] + d / 60;
    it[1] = Path[i + 2];
    it[3] -= d;
}

}
}
}

//Code Block for reusing Existing Vehicles - End
else{
    cout<<"Need One more Vehicle for going from "<<Path[i]<<" to
"<<Path[i+1]<<endl;
    vehicles += 1;
    //Position and Time Of New Vehicle
    vector<int> New;
    New.push_back(vehicles);
    New.push_back(Path[i+1]);
    New.push_back(Deadlines[Path[i+1]][0]*60+Deadlines[Path[i+1]][1]);
    New.push_back(100-ShortestDist[0][Path[i+1]]);
    Utils.push_back(New);

//Positions.push_back({vehicles+1,{Path[i+1],Deadlines[Path[i+1]][0]*60+Deadlines[Path[i+1]][1]

```

```

});

    FirstApproachCost += ShortestDist[0][Path[i+1]]*5;
}
}
}
else{
    if(Path[i+1] == -1){
        int d = charging[Path[i]][0]+charging[Path[i+2]][0];
        CurrentTimeH += d/60;
        CurrentTimeM += d%60;
        if(CurrentTimeM >= 60){
            CurrentTimeM -= 60;
            CurrentTimeH += 1;
        }
        if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] <
CurrentTimeM+CurrentTimeH*60){
            if(vehicles > 1){
                for(auto it: Utils){
                    if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it[2]+d){
                        if(it[3] < d){//charging is not sufficient
                            //Utils[0][0] represents vehicle number - 2 ,, Utils[1][0] represents vehicle
number - 3

                            //Utils[0][1] represents current location of vehicle - 2
                            //Utils[0][2] represents Last used time of vehicle - 2 in minutes
                            //Utils[0][3] represents charging of vehicle - 2
                            int dtc = min(charging[it[1]][0],charging[it[1]][1]);
                            int d2 = min(charging[Path[i+2]][0],charging[Path[i+2]][1]);
                            int Time = (dtc+d2);
                            if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it[2]+Time){
                                cout<<"*** Re-Using Vehicle ***"<<endl;
                                cout<<"We can use existing vehicle"<<endl;
                                cout<<"Vehicle Number: "<<it[0]<<endl;
                                cout<<"currently at: "<<it[1]<<" will reach: "<<Path[i+2]<<" by
"<<(it[2]+Time)/60<<" "<<(it[2]+Time)%60<<endl;
                                it[1] = Path[i+2];
                                it[2] = it[2]+Time;
                                it[3] = it[3]-d2;
                                FirstApproachCost += it[3];
                                cout<<"*****"<<endl;
                            }

```

```

    }
    else{
        it[2] = it[2]+d/60;
        it[1] = Path[i+2];
        it[3] -= d;
    }

}

}

/*
for(auto &it: Positions){
    if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it.second.second){
        cout<<"*** Re-Using Vehicle ***"<<endl;
        cout<<"We can use existing vehicle"<<endl;
        cout<<"Vehicle Number: "<<it.first<<endl;
        cout<<"currently at: "<<it.second.first<<" will reach: "<<Path[i+2]<<" by
"<<it.second.second/60<<" "<<it.second.second%60<<endl;
        cout<<"*****"<<endl;
        it.second.first = Path[i+2];
        it.second.second += d/60;
        Utils[0] = Path[i+2];
    }
}

*/

}

else{
    cout<<"Need One more Vehicle for going from "<<Path[i]<<" to
"<<Path[i+2]<<endl;
    vector<int> New;
    New.push_back(vehicles+1);
    New.push_back(Path[i+2]);
    New.push_back(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1]);
    New.push_back(100-ShortestDist[0][Path[i+2]]);
    Utils.push_back(New);

//Positions.push_back({vehicles+1,{Path[i+2],Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1]}})
;

    vehicles += 1;
    FirstApproachCost += ShortestDist[0][Path[i+2]]*5;
}

```

```

    }
}
if(Path[i+1] == -2){
    int d = charging[Path[i]][1]+charging[Path[i+2]][1];
    CurrentTimeH += d/60;
    CurrentTimeM += d%60;
    if(CurrentTimeM >= 60){
        CurrentTimeM -= 60;
        CurrentTimeH += 1;
    }
}
/*
if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] <
CurrentTimeM+CurrentTimeH*60){
    if(vehicles > 1){
        for(auto &it: Positions){
            if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it.second.second){
                cout<<"*** Re-Using Vehicle ***"<<endl;
                cout<<"We can use existing vehicle"<<endl;
                cout<<"Vehicle Number: "<<it.first<<endl;
                cout<<"currently at: "<<it.second.first<<" will reach: "<<Path[i+2]<<" by
"<<it.second.second/60<<" "<<it.second.second%60<<endl;
                cout<<"*****"<<endl;
                it.second.first = Path[i+2];
                it.second.second += d/60;

            }
        }
    }
    else{
        cout<<"Need One more Vehicle for going from "<<Path[i]<<" to
"<<Path[i+2]<<endl;
        vector<int> New;
        New.push_back(vehicles+1);
        New.push_back(Path[i+2]);
        New.push_back(Deadlines[Path[i+1]][0]*60+Deadlines[Path[i+1]][1]);
        New.push_back(100);
        Utils.push_back(New);

//Positions.push_back({vehicles+1,{Path[i+2],Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1]}})
;

```

```

        vehicles += 1;
    }
    */

    if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] <
    CurrentTimeM+CurrentTimeH*60){
        if(vehicles > 1){
            for(auto it: Utils){
                if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it[2]+d){
                    if(it[3] < d){
                        //Utils[0][0] represents vehicle number - 2 ,, Utils[1][0] represents vehicle
                        number - 3

                        //Utils[0][1] represents current location of vehicle - 2
                        //Utils[0][2] represents Last used time of vehicle - 2 in minutes
                        //Utils[0][3] represents charging of vehicle - 2
                        int dtc = min(charging[it[1]][0],charging[it[1]][1]);
                        int d2 = min(charging[Path[i+2]][0],charging[Path[i+2]][1]);
                        int Time = (dtc+d2);
                        if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it[2]+Time){
                            cout<<"*** Re-Using Vehicle ***"<<endl;
                            cout<<"We can use existing vehicle"<<endl;
                            cout<<"Vehicle Number: "<<it[0]<<endl;
                            cout<<"currently at: "<<it[1]<<" will reach: "<<Path[i+2]<<" by
                            "<<(it[2]+Time)/60<<" "<<(it[2]+Time)%60<<endl;
                            it[1] = Path[i+2];
                            it[2] = it[2]+Time;
                            it[3] = it[3]-d2;
                            FirstApproachCost += it[3];
                            cout<<"*****"<<endl;
                        }
                    }
                }
            }
        }
        else{
            it[2] = it[2]+d/60;
            it[1] = Path[i+2];
            it[3] -= d;
        }
    }
}
}
/*
for(auto &it: Positions){

```

```

        if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] > it.second.second){
            cout<<"*** Re-Using Vehicle ***"<<endl;
            cout<<"We can use existing vehicle"<<endl;
            cout<<"Vehicle Number: "<<it.first<<endl;
            cout<<"currently at: "<<it.second.first<<" will reach: "<<Path[i+2]<<" by
"<<it.second.second/60<<" "<<it.second.second%60<<endl;
            cout<<"*****"<<endl;
            it.second.first = Path[i+2];
            it.second.second += d/60;
            Utils[0] = Path[i+2];
        }
    }
    */
}
else{
    cout<<"Need One more Vehicle for going from "<<Path[i]<<" to
"<<Path[i+2]<<endl;
    vector<int> New;
    New.push_back(vehicles+1);
    New.push_back(Path[i+2]);
    New.push_back(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1]);
    New.push_back(100-ShortestDist[0][Path[i+2]]);
    Utils.push_back(New);

//Positions.push_back({vehicles+1,{Path[i+2],Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1]}})
;

    vehicles += 1;
    FirstApproachCost += ShortestDist[0][Path[i+2]]*5;
}
}
}
i += 1;
}
}

};

```

```

vector<int> Destinations;
void MinVehiclesUpperBound(vector<int> &Distances, vector<vector<int>>
&Deadlines,vector<int> Path,vector<vector<int>> &charging,int &CurrentTimeH,int

```

```

&CurrentTimeM){
    vehicles = 1;
    for(int i = 0;i<Path.size();){
        if(Path[Path.size()-1] == 0 and i == Path.size()-2){
            cout<<"Currently at customer - "<<Path[i]<<endl;
            cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            CurrentTimeH += Distances[Path[i]]/60;
            CurrentTimeM += Distances[Path[i]]%60;
            cout<<"Currently at customer - "<<0<<endl;
            cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            break;
        }
        else if(Path[Path.size()-1] == 0 and i == Path.size()-3 and Path[Path.size()-2]<0){
            if(Path[Path.size()-2] == -1){
                cout<<"Currently at customer - "<<Path[i]<<endl;
                cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
                int d = charging[Path[i]][1]+charging[Path[i+2]][1];
                CurrentTimeH += d/60;
                CurrentTimeM += d%60;
                if(CurrentTimeM >= 60){
                    CurrentTimeM -= 60;
                    CurrentTimeH += 1;
                }
                cout<<"Currently at customer - "<<0<<endl;
                cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            }
            if(Path[Path.size()-2] == -2){
                cout<<"Currently at customer - "<<Path[i]<<endl;
                cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
                int d = charging[Path[i]][1]+charging[Path[i+2]][1];
                CurrentTimeH += d/60;
                CurrentTimeM += d%60;
                if(CurrentTimeM >= 60){
                    CurrentTimeM -= 60;
                    CurrentTimeH += 1;
                }
                cout<<"Currently at customer - "<<0<<endl;
                cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
            }
        }
        break;
    }
}

```

```

    }
    if(i >= Path.size()){
        break;
    }
    if(Path[i] >= 0){
        cout<<"Currently at customer - "<<Path[i]<<endl;
        cout<<"Current time: "<<CurrentTimeH<<" "<<CurrentTimeM<<endl;
    }
    //if the next visiting station is customer else it is charging station
    if(Path[i+1] >= 0 and Path[i] >= 0){
        CurrentTimeH += Distances[Path[i]]/60;
        CurrentTimeM += Distances[Path[i]]%60;
        if(CurrentTimeM >= 60){
            CurrentTimeM -= 60;
            CurrentTimeH += 1;
        }
        // If Deadline is Not Met
        if(Deadlines[Path[i+1]][0]*60+Deadlines[Path[i+1]][1] <
CurrentTimeM+CurrentTimeH*60){
            Destinations.push_back(Path[i+1]);
            cout<<"Need One more Vehicle for going from "<<Path[i]<<" to "<<Path[i+1]<<endl;
            vehicles += 1;
        }
        i += 1;
    }
    else{
        if(Path[i+1] == -1){
            int d = charging[Path[i]][0]+charging[Path[i+2]][0];
            CurrentTimeH += d/60;
            CurrentTimeM += d%60;
            if(CurrentTimeM >= 60){
                CurrentTimeM -= 60;
                CurrentTimeH += 1;
            }
            if(Deadlines[Path[i]][0]*60+Deadlines[Path[i+2]][1] <
CurrentTimeM+CurrentTimeH*60){
                Destinations.push_back(Path[i+2]);
                cout<<"Need One more Vehicle for going from "<<Path[i]<<" to
"<<Path[i+2]<<endl;
                vehicles += 1;
            }
        }
    }
}

```

```

    }
    i += 2;
}
if(Path[i+1] == -2) {
    int d = charging[Path[i]][1] + charging[Path[i + 2]][1];
    CurrentTimeH += d / 60;
    CurrentTimeM += d % 60;
    if (CurrentTimeM >= 60) {
        CurrentTimeM -= 60;
        CurrentTimeH += 1;
    }
    if (Deadlines[Path[i]][0] * 60 + Deadlines[Path[i + 2]][1] < CurrentTimeM +
CurrentTimeH * 60) {
        Destinations.push_back(Path[i+2]);
        cout << "Need One more Vehicle for going from " << Path[i] << " to " << Path[i + 2]
<< endl;
        vehicles += 1;
    }
    i += 2;
}
}
};

```

```

int main(){
    // Input:
    /* iniCharge : initial charge
    * c : Customers
    * Cs : charging stations
    * ChargingDist : distance of customers from charging stations
    * CustomerDist : distance of other customers w.r.t one
    * r : rate of discharge ( r = 1 -> 1% of charge per km )
    * Cc : Charging Cost
    *
    * Problems to be solved:
    * 1. printing Cost required for round trip.
    * 2. Charging to filled at each charging station (following partial charging)
    * 3. Minimum Number of vehicles needed for successful delivery within time.
    * Parameters needed to be programmed:
        * 1. Successful delivery considering that there is a deadline for each customer

```

```

* 2. Considering two different charging types possible at a charging station :
*   1. Slow charging with less price
*   2. Fast charging with more price.
*   slow charge:
*       rate of charge: 20% per hour.
*       cost: 5 rupees for 1% charging.
*   fast charge:
*       rate of charge: 60% per hour.
*       cost: 10 rupees for 1% charging.
*
* info:
*   1. all the vehicles will start from the depot at 8 AM.
*   2. vehicles follow a avg speed of 60km/hour.
*
*/

// 1st Part - Start
cout<<"***** 1. Start of finding path followed following partial charging *****"<<endl;
int iniCharge = 100;
vector<vector<int>> ChargingDist;
std::ifstream myfile ("C:/Users/rahul/OneDrive/Desktop/projectfiles/ChargingDistances.txt");
string ans = "";
if(myfile.is_open()){
    char mychar;
    while(myfile){
        mychar = myfile.get();
        if(isdigit(mychar)){
            ans += mychar-'0';
        }
        //std::cout << mychar;
    }
}
cout<<endl;
for(int i = 0;i<ans.size();i+=4){
    ChargingDist.push_back({int(ans[i])*10+int(ans[i+1]),int(ans[i+2])*10+int(ans[i+3])});
}
/*
vector<vector<int>> ChargingDist = {{10,20},
                                   {10,30},
                                   {20,30},

```

```

        {60,40},
        {40,20},
        {40,30}};

*/
for(auto it :ChargingDist){
    vector<int> v = it;
    cout<<v[0]<<" "<<v[1]<<endl;
}
vector<vector<int>> CustomerDist;
std::ifstream myfile1 ("C:/Users/rahul/OneDrive/Desktop/projectfiles/CustomerDistances.txt");
string ans1 = "";
vector<int> val;
if(myfile1.is_open()){
    char mychar;
    while(myfile1){
        mychar = myfile1.get();
        if(isdigit(mychar)){
            ans1 += mychar-'0';
        }
        //cout<<mychar;
        if(mychar == ' '){
            val.push_back(int(ans1[0])*10+ans1[1]);
            ans1 = "";
        }
    }
}
for(int i = 0;i<6;i++){
    vector<int> V;
    for(int j = 0;j<6;j++){
        V.push_back(val[i*6+j]);
    }
    CustomerDist.push_back(V);
}
for(auto& it: CustomerDist){
    for(int x: it){
        cout<<x<<" ";
    }
    cout<<endl;
}
int r = 1; //r : rate of discharge ( r = 1 -> 1% of charge per km )

```

```

vector<int> Distances = floydWarshall(CustomerDist);
int Cost = 0;
int Answer = PartialCharging(0,Distances,r,ChargingDist,Cost,iniCharge);
cout<<"Left Over Charge: "<<leftOverCharge<<endl;
cout<<"Overall Cost Following Partial Charging: "<<Answer-100*5<<endl;
cout<<"***** End of finding path following partial charging *****"<<endl;
// 1st Part - End

// 2nd Part- Start
// setting deadlines for customers
// 0th index refers to deadline
// deadline at depot is considered as starting time of vehicles
vector<vector<int>> Deadlines = {{9,0},{10,30},{10,45},{11,00},{11,30},{12,00}}; //{10,30}
represents 10 hours 30 minutes
// time is to be taken input in 24-hour format.
cout<<"***** 2. Start of Finding Minimum Number of vehicles required *****"<<endl;
//Printing Path
cout<<"Path: ";
Path.push_back(0);
for(int x: Path){
    cout<<x<<" ";
}
cout<<endl;
cout<<"Distances: ";
for(int x: Distances){
    cout<<x<<" ";
}
cout<<endl;
int CurrentTimeH = 9;//9 hour 0 minutes is the current time when the vehicle starts
int CurrentTimeM = 0;
int UpperBoundAns = Answer-500;
MinVehicles(Distances,Deadlines,Path,ChargingDist,CurrentTimeH,CurrentTimeM);
cout<<"Min Number of vehicles required by reusing vehicles: "<<vehicles<<endl;
cout<<" ----->First Approach Cost (By reusing vehicles):
"<<FirstApproachCost+UpperBoundAns<<endl;
int m1 = FirstApproachCost+UpperBoundAns;
CurrentTimeH = 9;//9 hour 0 minutes is the current time when the vehicle starts
CurrentTimeM = 0;
vehicles = 0;

```

```

MinVehiclesUpperBound(Distances,Deadlines,Path,ChargingDist,CurrentTimeH,CurrentTimeM
);
cout<<"Destinations: "<<endl;
for(int x: Destinations){
    cout<<x<<" ";
    cout<<ShortestDist[0][x]<<endl;
    UpperBoundAns += ShortestDist[0][x]*5*2;
}
cout<<"Min Number of vehicles required without reusing vehicles: "<<vehicles<<endl;
int m2 = UpperBoundAns;
cout<<" ----->Second Approach Cost (by borrowing vehicles from depot):
"<<UpperBoundAns<<endl;
cout<<"***** End of Finding Minimum Number of vehicles required *****"<<endl;
// 2nd Part- End

// 3rd Part - Start
//cout<<"***** 3. Start of Introducing fast Charging for each customer *****"<<endl;
//TimeWindow(0,Distances,r,ChargingDist,Cost,iniCharge,Deadlines);
//cout<<"***** End of Introducing Fast Charging to each customer *****"<<endl;
// 3rd Part - End

// Populating text file with Answers

fstream f;
f.open("C:/Users/chana/CLionProjects/Inputs/1/M1.txt", ios::app);
if (!f)
    cout << "No such file found";
else {
    f << m1;
    cout << "Data appended successfully\n";
    f.close();
    string word;
    f.open("Geeks for Geeks.txt");
    while (f >> word) {
        cout << word << " ";
    }
    f.close();
}

```

```

}
f.open("C:/Users/chana/CLionProjects/Inputs/1/M2.txt", ios::app);
if (!f)
    cout << "No such file found";
else {
    f << m2;
    cout << "Data appended successfully\n";
    f.close();
    string word;
    f.open("Geeks for Geeks.txt");
    while (f >> word) {
        cout << word << " ";
    }
    f.close();
}
}
}

```

B.SCREENSHOTS

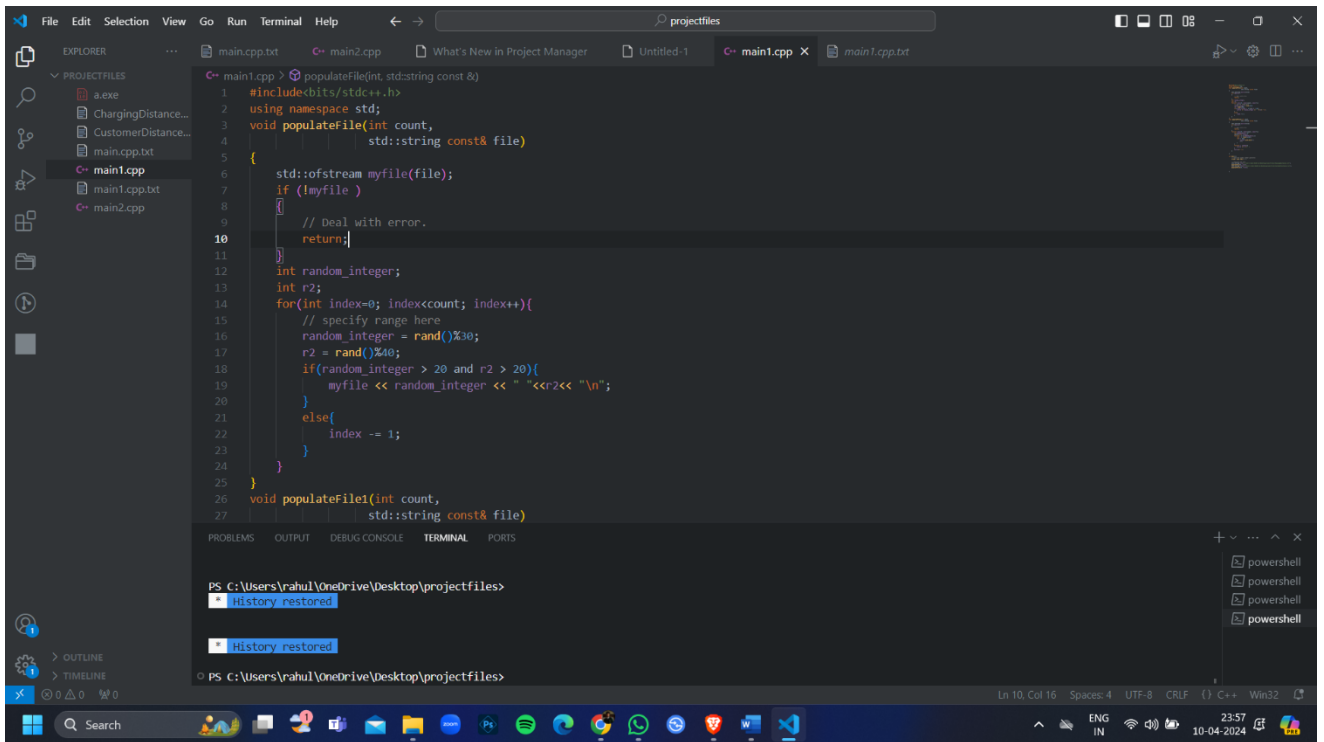


Fig. B.1: Main1.cpp

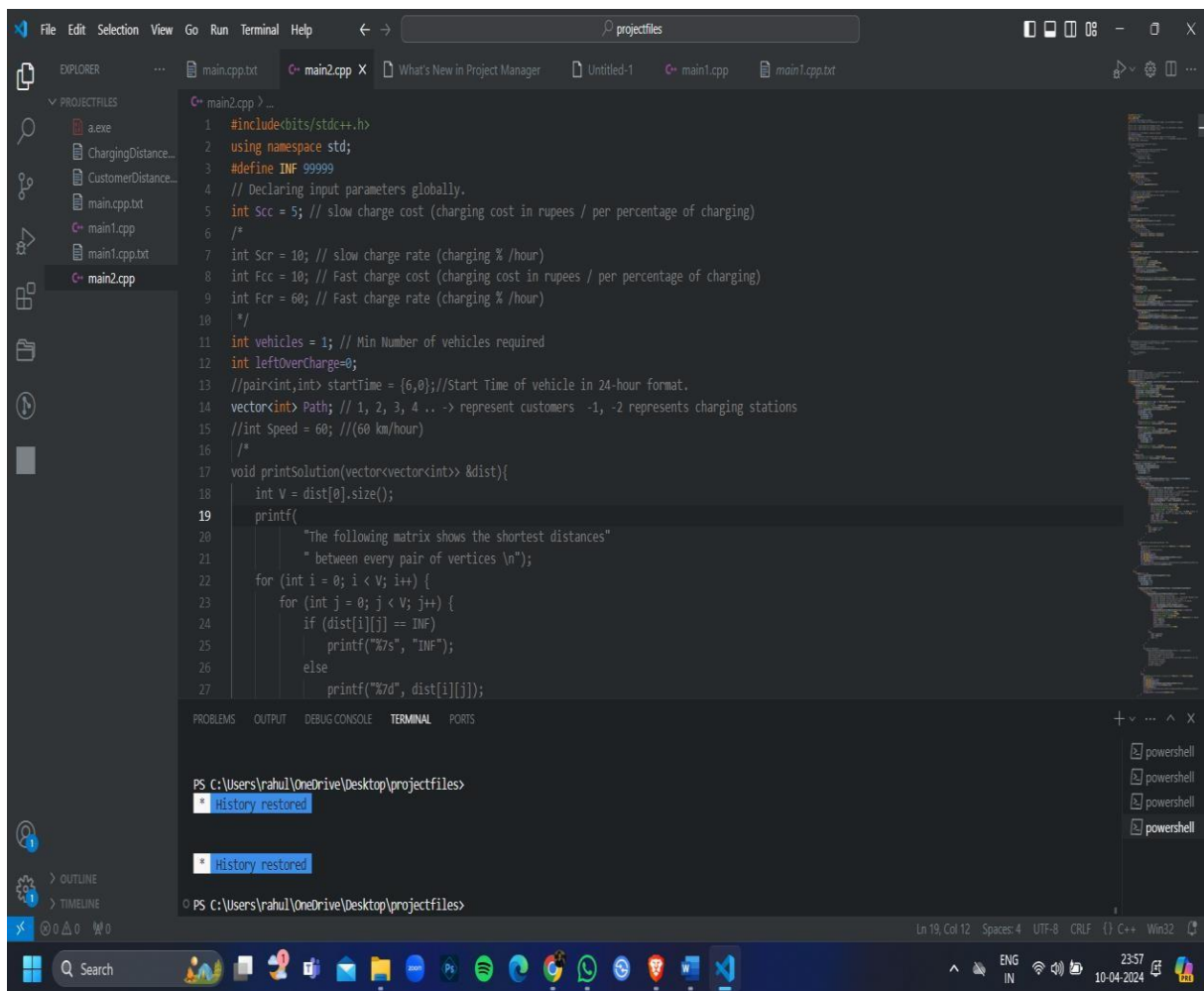


Fig. B.1.1: Main2.cpp

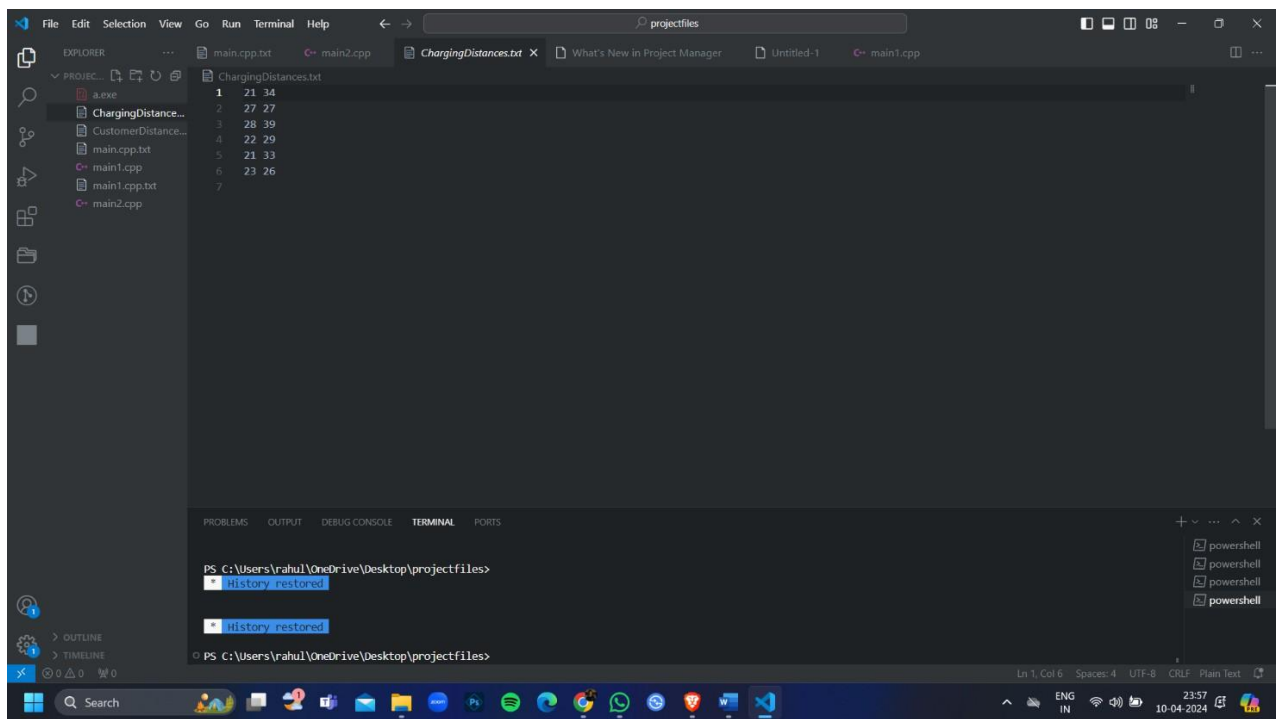


Fig. B.1.2 INPUTS OF THE CHARGING DISTANCE

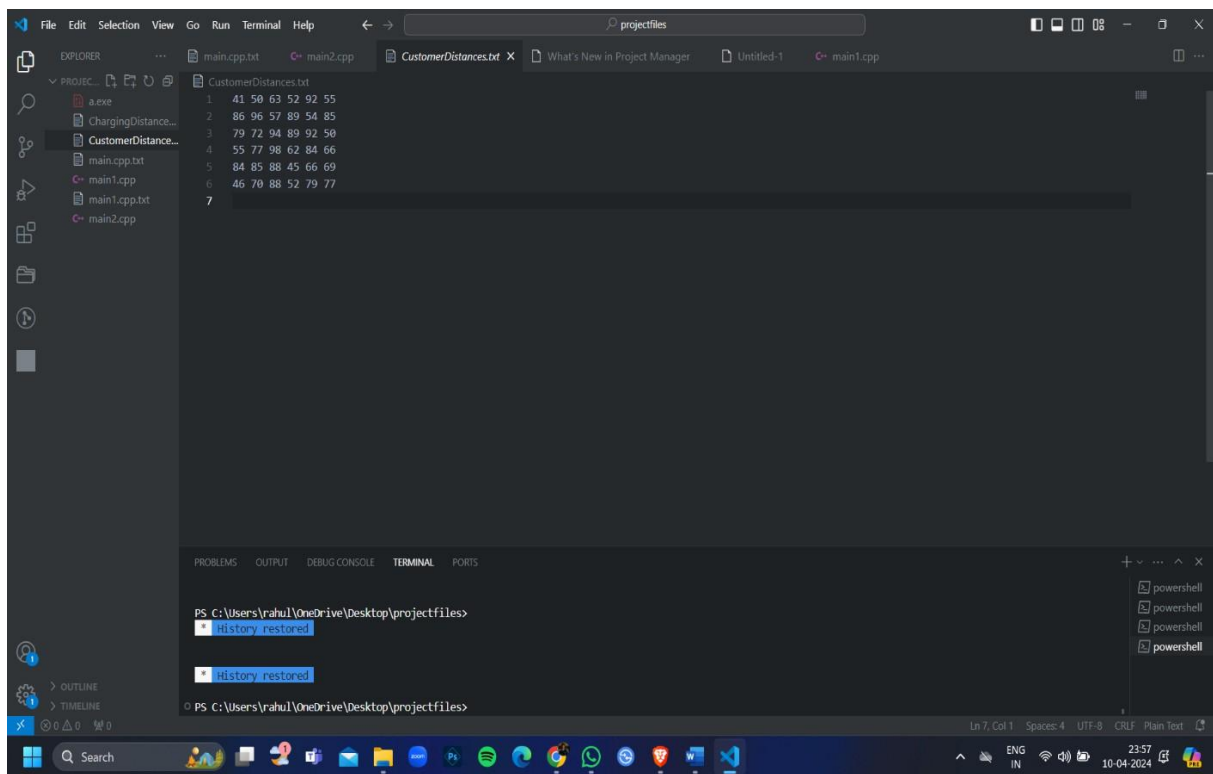


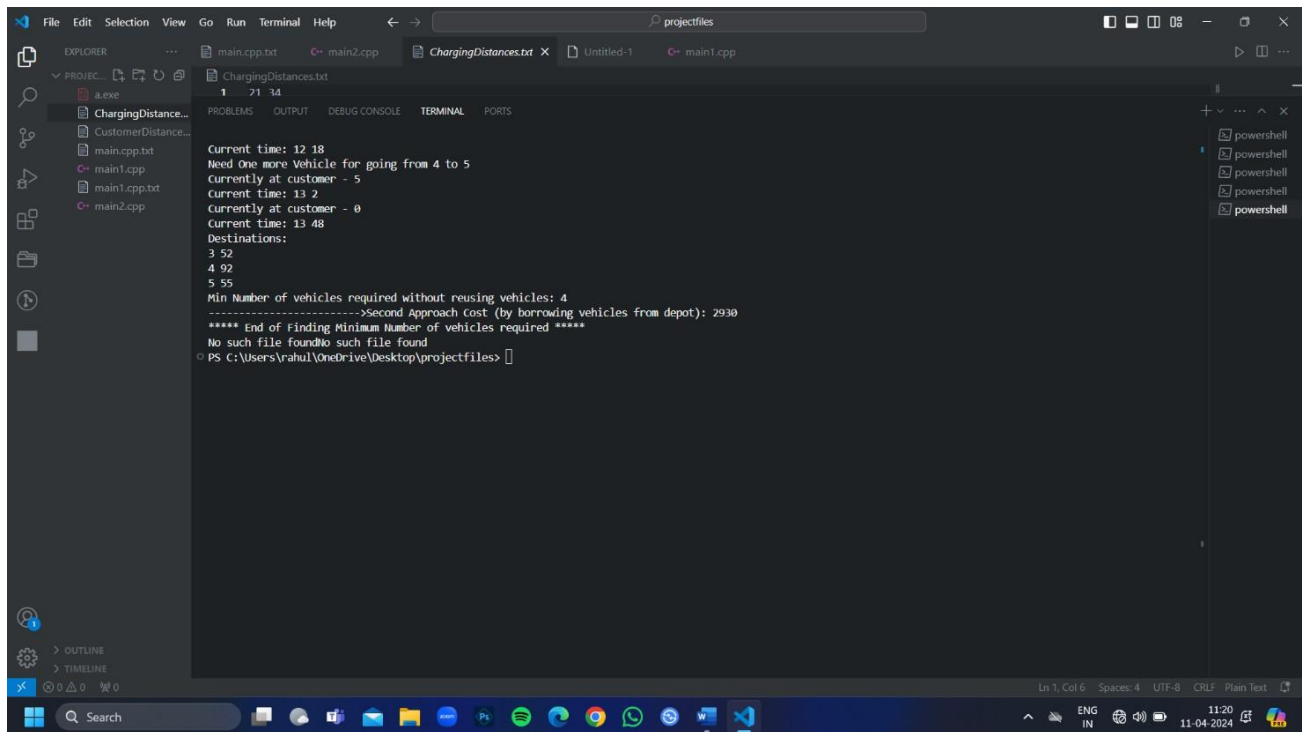
Fig. B.1.3 INPUTS OF THE CUSTOMER DISTANCE

```
***** 1. Start of finding path followed following partial charging *****  
21 34  
27 27  
28 39  
22 29  
21 33  
23 26  
41 58 63 52 92 55  
86 96 57 89 54 85  
79 72 94 89 92 50  
55 77 98 62 84 66  
84 85 88 45 66 69  
46 70 88 52 79 77  
Distances: 58 57 89 84 69 46  
Current Customer: 0  
Current charge: 100  
Cost Till Now: 0  
Current Customer: 1  
Current charge: 50  
Cost Till Now: 250  
Going to charging station - 1 : 77% filled  
Current customer: 2  
Current charge: 72  
Cost Till Now: 525  
Going to charging station - 1 : 56% filled  
Current customer: 3  
Current charge: 78  
Cost Till Now: 775  
Going to charging station - 1 : 44% filled  
Current customer: 4  
Current charge: 79  
Cost Till Now: 990  
Going to charging station - 1 : 42% filled  
Current customer: 5  
Current charge: 77
```

Fig. B.2.1 FINDING PATH FOLLOWED FOLLOWING PARTIAL CHARGING

```
Path: 0 1 -1 2 -1 3 -1 4 -1 5 0
Distances: 50 57 89 84 69 46
Currently at customer - 0
Current time: 9 0
Currently at customer - 1
Current time: 9 50
Currently at customer - 2
Current time: 10 45
Need One more Vehicle for going from 2 to 3
Currently at customer - 3
Current time: 11 35
Currently at customer - 4
Current time: 12 18
Currently at customer - 5
Current time: 13 2
Currently at customer - 0
Current time: 13 48
Min Number of vehicles required by reusing vehicles: 2
----->First Approach Cost (By reusing vehicles): 1200
Currently at customer - 0
Current time: 9 0
Currently at customer - 1
Current time: 9 50
Currently at customer - 2
Current time: 10 45
Need One more Vehicle for going from 2 to 3
Currently at customer - 3
Current time: 11 35
Need One more Vehicle for going from 3 to 4
Currently at customer - 4
Current time: 12 18
Need One more Vehicle for going from 4 to 5
Currently at customer - 5
Current time: 13 2
Currently at customer - 0
Current time: 13 48
```

Fig. B.2.2 FIRST APPROACH COST



```
File Edit Selection View Go Run Terminal Help
main.cpp.txt main2.cpp ChargingDistances.txt Untitled-1 main1.cpp
ChargingDistances.txt
1 21 34
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Current time: 12 18
Need One more Vehicle for going from 4 to 5
currently at customer - 5
current time: 13 2
Currently at customer - 0
Current time: 13 48
Destinations:
3 52
4 92
5 55
Min Number of vehicles required without reusing vehicles: 4
----->Second Approach Cost (by borrowing vehicles from depot): 2930
***** End of Finding Minimum Number of vehicles required *****
No such file found\no such file found
PS C:\Users\rahul\OneDrive\Desktop\projectfiles>
```

Fig. B.2.3 MIN NUMBER OF VEHICLES REQUIRED WITHOUT REUSING VEHICLES

Routing Of Electrical With Partial Recharge And Costumer Time Window

E.Murali
Department
of CSE,
Sathyabama Institute Of
Science And Technology,
Chennai, India.
Emurali88@gmail.com

J.Rahul
Department of CSE,
Sathyabama Institute Of
Science And Technology,
Chennai, India.
rahulsam.831@gmail.com

G.Keerthi
Reddy
Department
of CSE,
Sathyabama Institute Of
Science And Technology,
Chennai, India.
Guttapalemkeerthi123@gmail.com

ABSTRACT

Since gasoline automobiles are major sources of pollution, replacing them with electric vehicles has been a hope for reducing pollution globally. The requirement for charging stations (CS) grows along with the demand and usage of EVs. However, charging at a charging station is more expensive than charging a gasoline-powered car. Therefore, in order to lower this expense, we are putting out an algorithm that lowers the total cost of charging. Additionally, if there is a fleet of cars in the depot that meets client needs by the deadline but may be used more frequently than necessary, resulting in increased expenses. Therefore, we are putting forth an optimized method that minimizes the number of vehicles used within the set deadline in addition to reducing costs by path.

KEYWORDS -- Electric Vehicle; Charging Station; Profit maximization; Depot; Customers; Power Consumption;.

I. INTRODUCTION

Everyone is now looking to electric vehicles as a solution to the growing worry about environmental harms including global warming and the enormous use of resources to produce a significant amount of petroleum products to satisfy

human needs.[1] Demand for different needs, including transportation and industry sectors. Among these, transport is a major challenge right now because to the rapid global population rise and the resulting increase in the exploitation of fossil fuels to meet the demand. Therefore, every philanthropist is motivated to discover a substitute for this purpose, and their current focus is on green cars, or Electric Vehicles (EVs), as we can say. Many organisations and nations are currently working to promote and compel people to switch to electric vehicles. The government is providing numerous incentives to encourage the usage of EVs. Over the past ten years, there has been a sharp rise in the demand for electric vehicles. Compared to just over one million in 2018, a study predicts that by 2030, there will be 18.7 million electric vehicles on the road. The increase in the demand for electric vehicles is going to increase the need for electric charging stations to run them with no hassle, but the cost to charge EV is more when compared to charge of combustion vehicles. So the people who use EVs feels to reduce the overall cost for charging at charging station,[2]also when there is a depot owner who wishes to minimize both charging cost as well as the number of vehicles to serve the customer, we are proposing an optimized algorithm which reduces both the cost using partial charge and number of vehicles to satisfy customers in given deadline.

Considering the number of customers to be served and by taking the customer deadline as input,giving a solution which reduces total cost to charge at charging station by partial charge and minimizing

the usage of number of vehicle.

In the dynamic landscape of environmental concerns and the imperative shift towards sustainable practices, the focus on electric vehicles (EVs) as a pivotal solution has intensified. The escalating worries about environmental degradation, including global warming and the exhaustive consumption of resources in the production of petroleum products, have catalyzed a collective effort to explore alternatives. The burgeoning global population has exacerbated the challenges in the transportation sector, prompting philanthropists and governments alike to seek viable substitutes for the current reliance on fossil fuels.

The trajectory towards green mobility is underscored by a concerted effort from organizations and nations to incentivize the adoption of EVs. Governments worldwide are offering a plethora of incentives to encourage individuals to make the switch to electric vehicles. Over the last decade, the demand for EVs has witnessed a remarkable surge. Projections indicate a monumental increase, with an estimated 18.7 million electric vehicles anticipated on the roads by 2030, compared to just over one million in 2018.

As the demand for electric vehicles intensifies, the need for a robust charging infrastructure becomes paramount. However, a notable challenge arises in the form of the perceived higher cost of charging electric vehicles compared to their combustion counterparts. In response to this concern, users of electric vehicles are seeking ways to mitigate the overall charging expenses at charging stations.

Additionally, for depot owners aiming to optimize both charging costs and the efficiency of vehicle service, a proposed algorithm emerges as a strategic solution. This algorithm, designed to reduce both the cost through partial charging and the number of vehicles required to meet customer demands within specified deadlines, offers a comprehensive approach to addressing the multifaceted challenges of the evolving electric vehicle landscape. By considering the number of customers to be served and integrating customer deadlines as input parameters, the algorithm seeks to streamline the charging process, ensuring cost-effectiveness and operational efficiency while aligning with the broader goals of sustainable and eco-friendly transportation solutions.

II. LITERATURE SURVEY

The routing of electric vehicles (EVs) with partial recharge and customer and time windows is a complex optimization problem. It involves determining the optimal route for each EV, taking into account the battery capacity, charging time, customer demands, and time windows.

1. Routing Algorithms and Optimization:

"A survey of green vehicle routing problem: Past and future trends" by Drezner, Z., & Hamacher, H. W. (2017).

"Electric vehicle routing problem: Recent advancements and future perspectives" by Zhang, X., et al. (2021).

"A survey of green vehicle routing problem with time-dependent charging stations" by Liu, W., et al. (2020).

2. Electric Vehicle Charging Infrastructure:

"A review of electric vehicle charger allocation and management in smart grid" by Zhang, C., et al. (2017).

"Electric vehicle charging station location and optimal allocation: A comprehensive review" by Wu, X., et al. (2020).

"Design and planning of electric vehicle charging infrastructure: A review" by Wang, J., et al. (2016).

3. Customer Preferences and Behavior:

"Electric vehicle adoption in the United States: A comprehensive review" by Noppers, E. H., et al. (2015).

"A review of factors influencing consumer intentions to adopt battery electric vehicles" by Banister, D., & Bowling, A. (2018).

"Understanding consumer preferences for electric vehicles: A review of the research literature" by Axsen, J., & Kurani, K. S. (2012).

4. Time Windows and Scheduling:

"Vehicle routing problems with time windows: A literature review" by Solomon, M. M. (1987).

"A comprehensive review and evaluation of permutation flowshop heuristics" by Ruiz, R., et al. (2008).

"Vehicle routing problems with time windows: New and updated algorithms" by Bräysy, O., & Gendreau, M. (2005).

5. Integration of electric vehicles and routing

"Electricity vehicle routing and charging Opportunities and challenges" by X.

D. Koutsopoulos et al. (2011) "Electric vehicle routing and charging: opportunities and challenges" by X. D.

6. SIMULATION AND CASE STUDIES

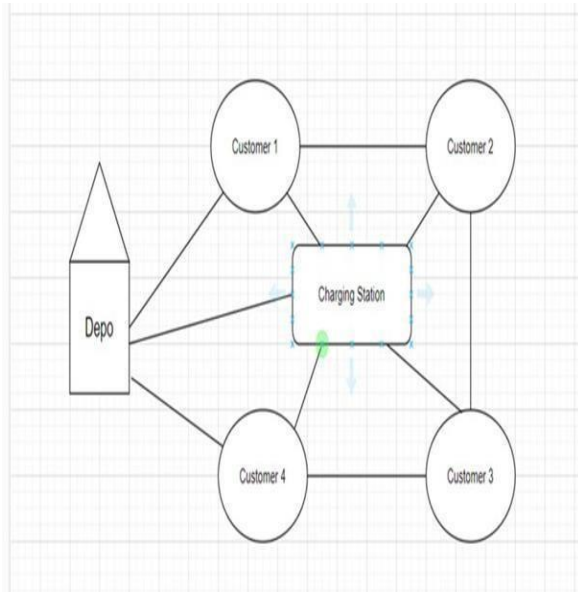
"Simulation of electric vehicle routing and charging considering battery degradation" by J. F. Meisel et al. (2017).

"Electrified vehicles routing: Models and heuristics" by C. A. C. Coelho et al. (2021).

III. PROPOSED SYSTEM

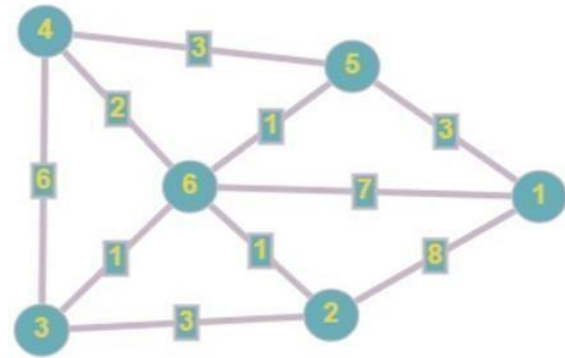
In this paper, we have proposed an algorithm for reducing the overall cost and minimizing the number of vehicles. In order to reduce cost at CS we need to find an algorithm

- Calculate the total distance to travel.
- Checking if current charge is sufficient.
- Charge to extend that helps in completing journey using partial charge In order to reduce the number of vehicles to satisfy customers in given deadline
- Check whether current vehicle can reach next customer in given deadline
- Check previous vehicle can reach next customer in given deadline
- Bring new vehicle from depot if required For this, we proposed a baseline approach and then proposed an optimised approach.

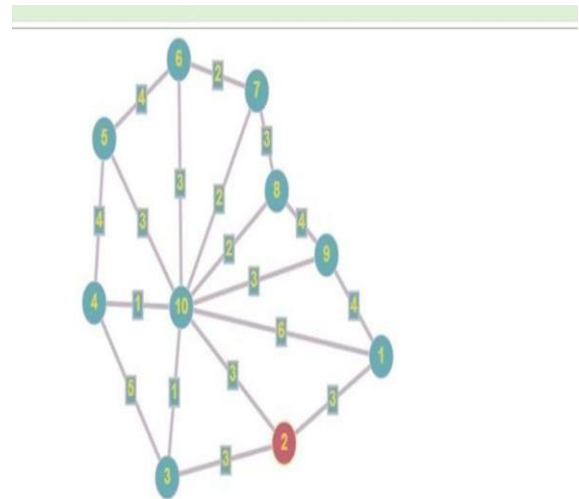


SELECTED METHODOLOGY OR PROCESS MODEL

Initially we have a set of EVs at Depot whose main role is to satisfy the customers based the deadline of each individual customer.[8] We take a EV that starts from depot and reaches to each customer one by one based on the deadline as customer with shortest time will be given higher priority.



Graph Representation with 4 customers



Graph Representation with 8 customers

Initially the EV has 100 percent charge and based on the rate of discharge the charge reduces, and the EV satisfies the customer and if there is any need to charge the EV goes to nearest charging station and charges to the extent required by partial charging. [9] In case if the current vehicle cannot reach the next customer in given time then a new vehicle comes from depot and satisfies the next customer and so on. In this way the amount to charge at CS reduces which in turn reduces the cost to charge at CS.

Algorithm 1 Baseline algorithm

Output: Percentage to be charged at Charging station
Declare the variables charge, k, and a vector of pairs called p

Calculate the discharge rate and the charging time required for each location.

for i = 0 to n do

Calculate the discharge rate for the distance using the input rate of discharge and add it to the vector v
Calculate the distance in terms of discharge rate and add it to the vector dc Create a pair of (v[i], dc[i]) and add it to the vector p end for

Set flag = false

while k < n+1 do

if charge > the v[k] + dc[k+1] then Subtract v[k] from charge Increment k

else if Set flag = true then

Call routine function with input vectors v and n+1

end if

if flag is false then No charge required end if

end while

Algorithm 2 Time Priority Vehicle Minimize Algorithm

Output: Percentage to be charged at Charging station and number of vehicles required Calculate the discharge rate and the customer deadline time required for each location. calculating total distance to travel

condition <- checking current vehicle can reach next to next node

if condition then

proceed

elsego to charging station

condition <- checking if curr_veh can reach next customer in given deadline

end if

if condition then

proceed

else if check for prev_vehicle can reach next customer in given deadline then

proceed

else

release new vehicle from depot

end if

check amount of charge required to fill for remaining journey

if condition then

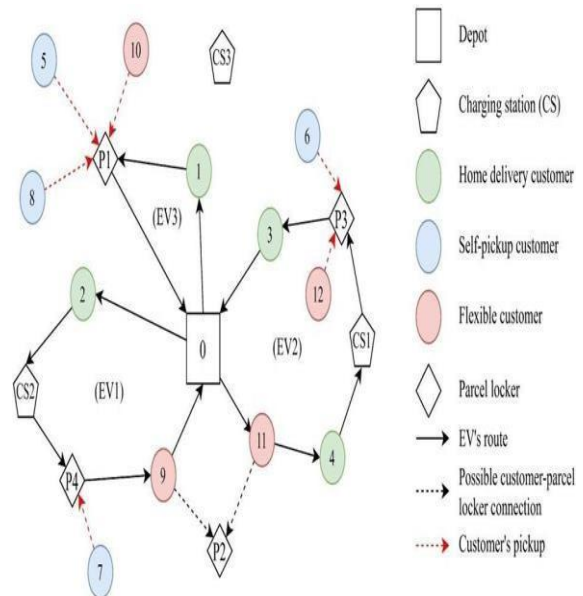
charge to required amount

else

charge to full

end if

The architecture and general design of the suggested system.



OVERVIEW OF SOFTWARE FOR THE PROPOSED MODEL'S SYSTEM AND IMPLEMENTATION TESTING PLAN.

To implement this model , execution of program is done through vs code in c++. Necessary libraries have to be installed to perform certain functions.

DESCRIPTION OF PROGRAMMING LANGUAGE AND SOFTWARE

C++:

The object-oriented programming language C++ reduces development costs by providing programs with a logical structure and enabling code reuse. Because C++ is portable, it may be used to create apps that are platform-neutral.

DSA:

Data Structures and Algorithms is an acronym for DSA. Another name for it is PSDS, or Problem Solving Data Structures. DSA helps developers learn how to solve problems and build effective code. The amount of memory and time the code takes can be used to determine its quality.

DESCRIPTION OF PROGRAMMING LANGUAGE AND SOFTWARE

Of course! Let's examine the definitions of DSA (Data Structures and Algorithms), a basic computer science topic that is frequently implemented using programming languages like C++, and C++.

C++ (Programming Language):

Overview:

The C programming language gave rise to the general-purpose, high-performance C++ language. C++ was created by Bjarne Stroustrup in the late 1970s and is used extensively in embedded systems, gaming, system/software development, and other fields.

Key Features:

- Object-Oriented Programming (OOP):** C++ supports OOP principles like encapsulation, inheritance, and polymorphism, allowing developers to structure code in a modular and reusable manner.
- Low-level Manipulation:** C++ provides low-level features like pointers, memory manipulation, and direct hardware access, giving programmers fine control over system resources.

- Standard Template Library (STL):** The STL is a powerful library in C++ that provides generic classes and functions with templates, enabling efficient data structures and algorithms.

- Multi-paradigm Language:** C++ supports procedural, object-oriented, and generic programming paradigms, offering flexibility to developers based on the requirements of the project.

- Performance:** Known for its efficiency and performance, C++ allows close-to-the-metal programming, making it suitable for resource-intensive applications.

- Portability:** Because C++ code can be generated and executed on a variety of platforms, it is more scalable..

Data Structures and Algorithms (DSA):

Data Structures:

Data structures are methods for efficiently arranging and storing data to carry out tasks. Arrays, linked lists, stacks, queues, trees, and graphs are examples of common data structures.

Algorithms:

Algorithms are methodical processes or equations used to solve issues. They provide down the reasoning needed to effectively modify data in data structures. Algorithms for searching (like binary search), sorting (like quicksort and mergesort), and graphing (like Dijkstra's method) are some examples.

Importance in Programming:

- Efficiency:** Well-designed data structures and algorithms lead to efficient programs with optimal use of resources (time and space complexity).
- Problem Solving:** DSA is fundamental for problem-solving in programming competitions, interviews, and real-world applications.
- Scalability:** Choosing appropriate data structures and algorithms is crucial for building scalable and high-performance software.

Integration with C++:

C++ is commonly used to implement data structures and algorithms due to its versatility, efficiency, and support for low-level operations. STL in C++ provides

pre-implemented data structures and algorithms, simplifying development.

****IDEs (Integrated Development Environments) and Tools:****

For C++ and DSA development, developers often use IDEs such as Visual Studio, Code::Blocks, or JetBrains CLion. Compilers like GCC (GNU Compiler Collection) or Microsoft Visual C++ Compiler are commonly used to translate C++ code into executable programs.

****Conclusion:****

C++ is a versatile programming language known for its performance and support for OOP, while DSA is a fundamental concept in computer science, crucial for building efficient algorithms and data structures. Combining C++ with DSA skills enables developers to create robust and scalable software solutions.

IV. CONCLUSION:

Vehicle pollution has become an important issue nowadays the only alternative to that available now is EVs. For the idea of EVs to run smoothly, we need proper planning of charging stations. When there are a fleet of vehicles in depot to satisfy a customer in given deadline maintaining both less cost to charge and minimum number of vehicles for satisfy is a challenging task.

So to overcome that challenge we are proposing an algorithm to satisfy all our needs with the given constraints in a optimised way.

In we future we can work heterogenous type of vehicles by considering waiting time at charging station.

V. REFERENCES

[1] Y. Zhou, J. Huang, J. Shi, R. Wang, and K. Huang, "The electric vehicle routing problem with

partial recharge and vehicle recycling," *Complex Intelligent Systems*, vol. 7, no. 1, p. 1445–1458, 2021. [2] M. Abid, M. Tabaa, A. Chakir, and H. Hachimi, "Routing and charging of electric vehicles," *Pluridisciplinary Laboratory of Research and Innovation (LPRI), EMSI Casablanca, Morocco*, 2022. [3] J. Li, F. Wang, and Y. He, "Electric vehicle routing problem with battery swapping considering energy consumption and carbon emissions," *Sustainability*, 2020. [4] N. Akbargpour, A. Salehi-Amiri, M. Hajiaghaei-Keshteli, and D. O. , "An innovative waste management system in a smart city under stochastic optimization using vehicle routing problem," *Soft Computing*, 2021. [5] K. Taghizad-Tavana, A. Alizadeh, M. Ghanbari-

Ghalehjoughi, and S. Nojavan, "A comprehensive review of electric vehicles in energy systems: Integration with renewable energy sources, charging levels, different types, and standards," 2023. [6] A. K. Agrawal, S. Yadav, A. A. Gupta, and S. Pandey, "A genetic algorithm model for optimizing vehicle routing problems with perishable products under time-window and quality requirements," *Decision Analytics*, 2022. [7]

M. S. Mastoi, S. Zhuang, H. M. Munir, M. Haris, M. Hassan, M. Usman, S. S. H. Bukhari, and J.-S. Ro, "An in-depth analysis of electric vehicle charging station infrastructure," *Energy Reports* 8, 2022. [8] C. Lin, K. Choy, G. Ho, S. C. , and H. Lam, "Survey of green vehicle routing problem: Past and future trends," 2014. [9] N. N, B. S. Quak H, R.-C. I, and T. T, "The electric vehicle routing problem with partial recharge and vehicle recycling," *State of the art of the electric freight vehicles implementation in city logistics*, 2013. [10] O. S, L. Z, W. Z, Z. J. , and L. R, "A study of china's explosive growth in the plug-in electric vehicle market," *US Department of Energy*, 2017. [11] G. Desaulniers, F. Errico, S. Irnich, and M. S. , "Exact algorithms for electric vehicle-routing problems with time windows," 2016.

D.CERTIFICATE

