Performance Assessment of Electric Vehicle Charging in a Charging Station Considering Charging Time and Price

MOHAMMAD FAHAD ULLAH

SUPERVISORS

Mohan Lal Kolhe(University of Agder, Norway)
K.M.S.Y Konara (University Of Ruhuna, Srilanka)
Souman Rudra (University of Agder, Norway)
Md. Saiful Islam (Chittagong University of Engineering and Technology, Bangladesh)

University of Agder, [2023]
Faculty of Engineering and Science
Department of Engineering Sciences

Abstract

Climate change is a buzz word in recent times. The whole world is suffering from it, temperature rise, sea level increase and many more are the consequences of it. One of the main reasons of it is the exploitation of fossil fuel. By burning fossil fuel, it emits different types of detrimental gases like CO_2,CH_4 etc in combination these are called Green House Gas (GHG). But 17% of this GHG & 20% of CO_2 is produced from transportation sector. To mitigate this the whole world is emphasizing at electrification of the transportation sector. And EV can be a proper solution for it. But there are some obstacles regarding to its purchase and its convenience. As the price of the electric vehicle is still high than fuel-based vehicle, time consumption of an electric vehicle for charging is numerous, inadequate amount of charging station infrastructure and another important issue is range anxiety of the EV drivers. As of now there are 16.5 million EV's are operating in the world and the IEA estimates that 300 million EVs will be in use by 2030, in accordance with the scenario of net zero emissions by 2050.To ensure uninterrupted service to this huge fleet of EV a lot of planning and implementation is needed. In this paper we proposed time-based price and charging-rate varying plan regarding for EVCS. which consider the effect on grid with the addition of huge EV fleet and EVCS financial profit. And we compared it with the conventional Charging system where charging price is volatile but charging time is fixed and constant charging system where both price and charging time are constant. We prepared a probabilistic conditional statement based optimizing algorithm for three of the system. And we considered 24hr scenario for all the systems. Results indicate that nothing controls consumer choices in the constant method, whereas only pricing may do so in the conventional way. This can result in a sudden increase in load at the charging station. However, with our approach, price, charging rate, and a combination of both helped spread out customers throughout various time periods while keeping the system in balance. Additionally, both traditional and constant systems have experienced consumer losses, which resulted in a financial loss for the charging station.

.

Abstrakt

Klimaendringer er et buzz-ord i nyere tid. Hele verden lider av under det, med temperaturstigning, havnivåøkning og mange flere erandre konsekvensene av det. En av hovedårsakene til det er utnyttelsen av fossilt brensel som v. Ved forbrenning avgir ulike typer å brenne fossilt brensel avgir det forskjellige typer skadelige gasser som , osv. i kombinasjon, dDisse kalles Green House Gas (GHG). Transportsektoren står for 17 % av disse drivhusgassene og 20 % av CO2-utslippene.
Men 17 % av disse drivhusgassene og 20 % av er produsert fra transportsektoren. For å redusere dette fokuserer verden på elektrifisering av transportsektoren, der elbiler kan være en løsning. Men det er noen hindringer når det gjelder kjøp og bekvemmelighet. For å dempe dette legger hele verden vekt på elektrifisering av transportsektoren. Og EV kan være en skikkelig løsning for det. Men det er noen hindringer for kjøpet og dets bekvemmelighet. Siden pPrisen på det elektriske kjøretøyet fortsatt er høyere enn et drivstoffbasert kjøretøy, som bruker fossilt brensel, lading av elbiler tar tid, det er utilstrekkelig ladestasjonsinfrastruktur, og det er også et viktig problem med rekkeviddeangst blant elbilførerne. er tidsforbruket til et elektrisk kjøretøy for lading mange, utilstrekkelig mengde ladestasjonsinfrastruktur og et annet viktig problem er rekkeviddeangst hos elbilførerne. Pr det trengs mye planlegging og gjennomføring Det kreves nøye planlegging og gjennomføring for å håndtere disse utfordringene. I denne artikkelen foreslo vi tidsbasert pris- og ladehastighetsvarierende plan for EVCS, . som vurderer effekten på nettet med tillegg av enorm EV-flåte og EVCS økonomisk fortjeneste. Og vVi sammenlignet det med det konvensjonelle ladesystemet der ladeprisen er ustabil, men ladetiden er fast og konstant ladesystem der både pris og ladetid er konstant. Vi utarbeidet en probabilistisk betinget
setning basert optimaliseringsalgoritme for tre av systemet,. Oog vi vurderte 24-timers scenario for alle systemene. Resultatene indikerer at ingenting styrer forbrukernes valg i den konstante metoden, mens bare prising kan gjøre det på den konvensjonelle måten. Dette kan resultere i en plutselig økning i belastningen ved ladestasjonen. Med vår tilnærming bidrar imidlertid pris, ladehastighet og en kombinasjon av begge til å fordele kundene jevnt over ulike tidsperioder samtidig som systemet opprettholder balanse.Med vår tilnærming bidro imidlertid pris, ladehastighet og en kombinasjon av begge til å spre kunder over ulike tidsperioder samtidig som systemet holdt balanse. I tillegg har både de tradisjonelle og konstante systemene r opplevd forbrukertaptap av kunder, noe som resulterte i et økonomisk tap for ladestasjonen.

Preface

My name is Mohammad Fahad Ullah, and I am an exchange student participating in the NORPART-2021/10355 (CARE) project.
This collaborative project is jointly conductedby the University of Agder and Chittagong University of Engineering &Technology.
I would like to express my sincere gratitude to HK-dir for their generous funding during my stay at the University of Agder.
The purpose of this study revolves around the projected increase in the number of electric vehicles (EVs) in the global transportation sector. Our aim is to develop a comprehensive plan that addresses the accommodation of this substantial EV fleet from the perspective of Electric Vehicle Charging Stations (EVCS).

I would like to extend my heartfelt appreciation to Prof. Mohan Lal Kohle for his invaluable guidance and support throughout my thesis journey. I am also grateful to Mr. Konara and Prof. Souman Rudra for their unwavering assistance and mentorship during this process. Additionally, I would like to express my gratitude to the authorities at CUET, particularly Prof. Abu Shadat Sayem and Prof. Saiful Islam, for selecting me to participate in this project.

Furthermore, I would like to thank the University of Agder for providing us with the opportunity to contribute to such a productive project.

Lastly, I would like to acknowledge the support and understanding of my cohabitant and family, who have been instrumental throughout the writing phase of my masters thesis.

Individual/group Mandatory Declaration

The individual student or group of students is responsible for the use of legal tools, guidelines for using these and rules on source usage. The statement will make the students aware of their responsibilities and the consequences of cheating. Missing statement does not release students from their responsibility.

1.	I/We hereby declare that my/our report is my/our own work and that I/We have not used any other sources or have received any other help than mentioned in the thesis.	☒
2.	I/we further declare that this thesis: - has not been used for another exam at another department/university/university college in Norway or abroad; - does not refer to the work of others without it being stated; - does not refer to own previous work without it being stated; - have all the references given in the literature list; - is not a copy, duplicate or copy of another's work or manuscript.	☒
3.	I/we am/are aware that violation of the above is regarded as cheating and may result in cancellation of exams and exclusion from universities and colleges in Norway, see Universitets- og høgskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	☒
4.	I/we am/are aware that all submitted theses may be checked for plagiarism.	☒
5.	I/we am/are aware that the University of Agder will deal with all cases where there is suspicion of cheating according to the university's guidelines for dealing with cases of cheating.	☒
6.	I/we have incorporated the rules and guidelines in the use of sources and references on the library's web pages.	☒

Publishing Agreement

Table of Contents

List of Figures

Abbreviations

EVCS	Electric Vehicle Charging Station
SOC	Sate of Charge
$P_{(EVCSOP)}$	off peak price
$P_{(EVCSP)}$	Peak price
t_{peak}	peak time
$t_{off\ peak}$	off-peak time
i	1 to 24 , 24 time slot
$b_i t_i$	Is the consumer load for a particular time slot
Battery-j	Battery size of a particular consumer
pt_i	per unit Price for a particular time slot
ct_i	Charger in KW for a particular time slot
j	1 to 20 is the number of consumer

1 Introduction

The environment is like a mother, which is why we call it Mother Nature. Our nature is facing annihilation because of our deeds. Yes, the overuse of once-beneficial fossil fuels has now become the villain; they are detrimental to the environment. Greenhouse gasses, which consist of CO_2,CH_4 etc., are basically responsible for accumulating the heat that is dissipating from our earth and making our world much warmer. And the consequences of these are a rise in sea level, imbalance in the seasons, bush fires, severe effects on ecology, etc. To address this massive problem, world leaders have launched an initiative known as green energy or renewable energy. But 17% of this GHG & 20% of $CO_2$is produced from transportation sector.[1] To reduce this amount electric vehicle can be an ideal solution. EV's operating cost is cheaper than combustion engine vehicle and can be replenished from renewable resources.[2] Around 1 in 250 vehicles on the road are electric, according to estimates. 16.5 million EV are active in the world.[3]

1.1 Motivation

In the US, 328,118 electric vehicles were sold in 2018, representing a 75% year-over-year increase in the number of electric vehicle sales. Among them 153,443 automobiles were sold in California.[3] Furthermore, compared to China, which is the largest contributor to worldwide plug-in sales in terms of growth, plug-in EV sales in the US account for less than 2% of the total EV market in this country. Norway leads the world in EV new car sales, accounting for 58% of all EV sales in 2019.[3]Only Japan is predicted to have 210,000 electric vehicles on the road by 2030. [3] The IEA reports that the number is expected to reach 300 million worldwide by 2030 under the "net zero emissions by 2050 scenario.".[4] Now the challenge is to fulfill the charging needs of this huge fleet, which will be added in the future. A scheme that can persuade consumers toward equal distribution can be effective.

1.2 Problem

Future increases in the number of electric vehicles (EVs) could cause unsatisfactory charging situations and possibly discourage EV owners if their vehicles are charged inefficiently. Transformers may experience excessive pressure from uncoordinated charging and erroneous arrivals, leading to system instabilities.

1.3 Main Goal

The aim of this paper is to create more charging options for consumers regarding charging stations, keeping in mind that the EVCS has to earn revenue from them and that consumers can choose options that will vary in money rate and charging rate.

Objective1: To create more charging options for consumers at charging stations.

Objective2: To ensure charging stations can accommodate the maximum number of consumers.

Objective3: To ensure charging stations can earn profit.

1.4 Solution

To accomplish the stated goal, a variable charging rate and price rate scheme can be used, and it will be compared to traditional and fixed charging and pricing rate schemes. This strategy seeks to improve our comprehension of the utility of the variable scheme. Clarifying the purpose, designing the variable scheme while taking into account numerous elements, putting the scheme into practice and gathering pertinent data, comparing the results with the traditional and constant scheme, and analyzing the performance to identify advantages and areas for development are all part of the organized process. A probabilistic conditional statement-based algorithm can be implemented in Python for three of the schemes mentioned above. And in order to have a clear understanding, the output of these three systems can be compared.

1.5 Thesis Outline

The aim of this thesis is to create more charging options for a future elevated EV fleet. Simultaneously, maintaining the charging station’s profit. This report is divided into chapters, which are structured in the manner described below:

Chapter 2: Relevant theories

Chapter 3: Methods which Shows the structure of suggested solution and also the other mentioned plans.

Chapter 4: Shows the results of the three cases and a comparison of their simulation output and discussion.

Chapter 5: Conclusion then Chapter 6 have some future work suggestion.

1.6 Literature Review

In [5] the authors have proposed an optimal charging scheduling method considering that there is limited amount of charger in the charging station to reduce the operating cost of the charging station.in [6] The authors presented a decentralized scheduling system for heterogeneous and homogeneous EV fleets based on pricing variation.in [7] To tackle a discrete optimization issue centered on charging EVs at discrete charge rates, a variation of the ODC algorithm is created. In [8] provides an online algorithm to control EV loads via an indirect aggregator who releases a charging reference using the aggregate EV load in real-time. EVs then make binary charging decisions by comparing their SOC to the reference signal. According to a decentralized (T2) model, in [9] the authors propose a non-iterative method of sequentially scheduling one EV at a time. The algorithm seeks to reduce the aggregate load's maximum peak as well as its variance. The study in [10] also makes use of a sequential scheduling method to create a decentralized (T2) charging system that seeks to reduce the mean square error between the aggregate load in real time and a reference operating point calculated offline using information about non-EV load and EV mobility. The integrated placement planning strategy described in [11] by the authors maximizes PEV charging convenience while preserving the integrity of the power system. The suggested model also accounts for the long-term costs of crucial grid assets brought on by charging demand irrationality and uncertainty. How to efficiently plan the electric vehicle (EV) charging power to lower the cost of running the charging station when the number a critical issue because the number of chargers was constrained. To address this problem Authors in [12] offer a system for scheduling charging that responds to the price of time-of-use (TOU) power. In a review paper [13], Influencing EV users through power costs is one possible strategy to fill the load valley. In light of this, an aggregator may broadcast control signals, such as

signals that resemble prices and change in accordance with total demand. As a result, each EV user will selfishly want to save charging prices by planning the EV's charging at a period when the load valley is at its highest. A transformer's windings can become too hot from sustained overloading, which causes the transformer to break early. A distribution overload constraint must be included. In [13] it is suggested that limiting the maximum power that a transformer can carry at any given moment will help to reduce congestion.

But there are some constraints while buying electric vehicle, the price of the EV is still high.[14] EV drivers experience range anxiety because if they are on a long trip and don't find a charging station in the middle of it, the battery can be completely depleted.[15] Another issue is charging time how long a driver has to wait to fully charge their vehicle. Given the aforementioned circumstances, we must overhaul the system by developing new schemes that will benefit consumers and charging station. Through this scheme, we have to create multiple charging options for consumers. Simultaneously, create a time-based price varying and adaptive charger scheme that would be optimal for all the participants, including users, charging station owners, and grid operators or suppliers. Here we will focus on charging station perspective how they accommodate the participants and make a profit out of it.

1.7 Research Questions

The future holds a promising increase in the number of electric vehicles (EVs) for several reasons. Firstly, the growing concern for the environment and the urgent need to reduce emissions propel the transition towards cleaner transportation options. Secondly, ongoing advancements in battery technology are steadily improving the performance and range of EVs, making them more practical and appealing to consumers. Additionally, governments worldwide are implementing supportive policies and providing incentives to encourage the adoption of EVs, making them more affordable and accessible. Furthermore, the development of charging infrastructure, including fast-charging networks, is addressing range anxiety and facilitating convenient long-distance travel. With the long-term cost efficiency, reduced maintenance requirements, and continuous commitment of the automotive industry to electrification, the future of electric vehicles looks brighter than ever.As mentioned in the introduction by 2030 it is anticipated that close to 300 million EV would be on the road worldwide.

Research Question 1: How can Electric Vehicle Charging Stations (EVCSs) be optimized

to achieve profitability while effectively accommodating the increasing charging

demands of a rapidly growing Electric Vehicle (EV) fleet?

Research Question 2: To what extent are the existing conventional charging norms

capable of supporting the anticipated surge in the number of Electric Vehicles (EVs) in

the future?

Research Question 3: How does the implementation of a constant price and charge-rate method compare to alternative strategies in terms of addressing the challenges posed by the growing EV fleet?

2 Theory:

2.1 Probability

The systematic investigation of results in random experiments is called probability theory. It entails picturing every potential conclusion and accepting that the actual result cannot be foreseen. Depending on the nature of the experiment or previous outcomes, probabilities can be assigned. A balanced die roll, for instance, has a probability of 1/6 for each outcome. Even though the sets of potential outcomes for events like the lifespan of an electric bulb or city temperatures may be large, probabilities can still be determined using empirical data.[17]

2.2 Conditional Statement

The truth values of the conditional statement's parts alone cannot establish if it is true. Propositions that depend on two other propositions make up conditional sentences. This has sparked debates and philosophical controversies. The logical issue of conditionals primarily focuses on the formal properties of the conditional function, indicated by "if...then," which accepts pairs of assertions and generates new propositions. The idea that alternative truth values might be consistent with the overall facts must be taken into account when determining the truth value of counterfactual claims. The pragmatic difficulty of counterfactuals is this. To differentiate between semantic and pragmatic criteria, which can assist define the separation between the two portions of the notion, a semantic theory is used.

The three primary issues with conditionals are the difficulty of justified belief, the pragmatic issue of counterfactuals, and the epistemological issue of unrealized possibilities. Justified belief requirements are pragmatic, but the epistemological conundrum casts doubt on the strength of empirical and contrary-to-fact conditionals.[18]

3 Methods

This paper aims to create more charging options for consumers regarding charging station, keeping in mind that the EVCS has to earn revenue from it, and consumers can choose options that will vary in time, money rate, and charging rate. Here we prepared three cases for this, suppose there is an area, let's assume a transformer through which the electricity is conveyed to the area's households and charging stations. In the first case charging station uses fix charger to charge their consumers. And price of charging is also constant. For case 1, we assume the charger value is 25 kw and the price per unit is 8 NOK. And charging stations have a limitation on supply per hour. Here, there are two types of consumers: local consumers, who are the dwellers of the area, and opportunists, who basically traveled from another area to this area after a long distance of journey. So, in this study, a 24-hour scenario is considered with 20 consumers, where 15 of them are local consumers and 5 are opportunists. And this 24-hour scenario would be divided into 24-time slots. EVCS has a capacity for each time slot.

$b_i t_i$ = Is the consumer load for a particular time slot

All consumers have different battery sizes. Each consumer would pay a different fee and charge at a same rate, keeping the battery size in mind. The price of charging is fixed for case 1, but if a consumer's battery size is large, his overall cost would be on the higher side than a consumer with a small battery.Depending on their trip and level of awareness, each customer has a different state of charge (SOC). SOC would be generated randomly by each consumer

Battery size of a consumer is,

Battery-j = Battery size of a particular consumer

Where $b_i t_i$ calculates the consumer load in a time slot based on the consumer battery size

$$b_i t_i = \sum_{j=1}^{20} \text{Battery} - \text{j} \qquad 3.1$$

The total load of a time slot should not exceed the transformer capacity.[13]

$$b_i t_i < \, b_i t_i c \qquad 3.2$$

Where,

$b_i t_i c$ = Is the capacity of the transformer for a particular time slot

Here i= 1 to 24 , 24-time slot

Every consumer will have a different SOC. Regarding their SOC, their battery charge requirement would be diverse. The required battery charge is,

Real_Battery-j= Battery-j*soc 3.3

Where,

j=1 to 20 is the number of consumer

The charging time of a consumer depends on its battery size and SOC. From equation (3.3), we can understand that every consumer's charging requirement is distinct. Charging time is

j-Charging time-i= Real_Battery-j /ct_i 3.4

Where,

ct_i = Charger in KW for a particular time slot

As a result, each consumer would have 24 different charging rates based on the charging rate provided by the EVCS. As mentioned early for case1, EVCS provides fixed charging rate then a consumer would have 24 same values of charging rate.

The cost would vary for every customer depending on the size of the battery and price provided by the charging station. The expense of billing a customer would be.

j-Cost of charging-i= Real_Battery-j *pt_i 3.5

Where,

pt_i = per unit Price for a particular time slot

Thus, depending on the EVCS services that are offered, each consumer would pay a variation of prices. In case 1 charging station provides 24 resemble prices for every slot. So, a consumer will get 24 similar prices for charging his vehicle.

Combination is another crucial element, since it makes consumers believe that they will pay relatively less over time and that the process would go more quickly. Customers will search for both time and money in this section.

So, the combination would be

j-combo-i= j-Charging rate-i* j-Cost of charging-i 3.6

So, as per above a consumer also have 24 different combinations according to the available service of the EVCS. In case 1, charging station provides fixed charging rate and price. So, a consumer will get 24 identical combination.

Here, Combination , Cost of charging and charge rate are related ,

j=1 indicates consumer1, now for consumer1:

in equation (3.4), if i=1 then it indicates charging rate for 1st time-slot. in equation (3.5), if i=1 then it indicates charging cost for 1st time slot. in equation (3.6), if i=1 then it indicates combination for 1st time-slot.

if a consumer chose the 1st time-slot charging -rate then he has to pay for 1st time-slot cost of charging.

if a consumer chose the 1st time-slot combination then he has to pay for 1st time-slot cost of charging.

if a consumer chose the 1st time-slot cost of charging, he has to pay for 1st time-slot cost of charging

Consequently, there are three elements to our task.

In the first task, As we are working on 20 consumers and each consumer ows a different size of battery. Additionally, these consumers will produce their state of charge (soc) at random. Therefore, as described in equation (3.3), the generated SOC for each consumer will determine how much he must pay for charging and how much time it will take to fully charge his battery. On the consumer side, 15 of them are regarded as local consumers; they can be divided into three groups. consumer who prioritizes time. Time will take precedence over everything else for these shoppers. Customers who will give financial perspectives priority.

Therefore, the scenario is such that every consumer will give priority to time 33% of the time.Consumers will select a financial perspective about 33% of the time. A consumer's decision is influenced by combination 33% of the time. As a result, some consumers will prioritize their time, others their finances, and yet others may prioritize both. These local customers will receive a rang of costs depending on their battery size and other factors, as illustrated in equation (3.4). depending on their SOC, battery size, and charge rate provided by the EVCS.Here in case 1 charging stations price and charging time is fixed. So, a consumer will produce fix charging time and charging price from these two values of equations (3.4), (3.5) . A local consumer will get 24 identical combination value as charging rate and price rate are same for case 1. As that is pointed at equation (3.6).The remaining five customers who are classified as opportunists fall into two categories: setting money and time priorities. These opportunity seekers will also generate fix charging-rate and costs. They will choose their option randomly.

Here for case 1 as we mentioned earlier the price and charging rate is fixed so each consumer will receive 24 identical price , charging time and combination for 24-time slot. Following that each consumer will choose a price randomly from 24 identical prices. A consumer will do the same for both charging time and combination. There selected price would be stored in a variable chosen price . same way selected time would be stored in variable chosen time and selected combination in variable chosen combination.

A time slot would be indicated by the price, combination, or charging rate that each consumer or opportunist choose. Based on the price of that time slot, they would be required to pay a price, and that price would be added to a variable name expected. Through this, a charging-station will have an expected income value from the consumers.

If a consumer prioritizes time, regarding to case 1 he will choose a charging time randomly among the values he will generate from equation (3.4).

For that, if j-Charging time-i is related to j-Cost of charging-i as mentioned earlier, so

j-Cost of charging-i value would be added in expected variable. Through this charging station will estimate their expected revenue. Expected revenue is,

$$\text{Expected} \mathrel{+}= \text{j-Cost of charging-i} \qquad 3.7$$

For each consumer and opportunist based on their choice Same thing will be executed.

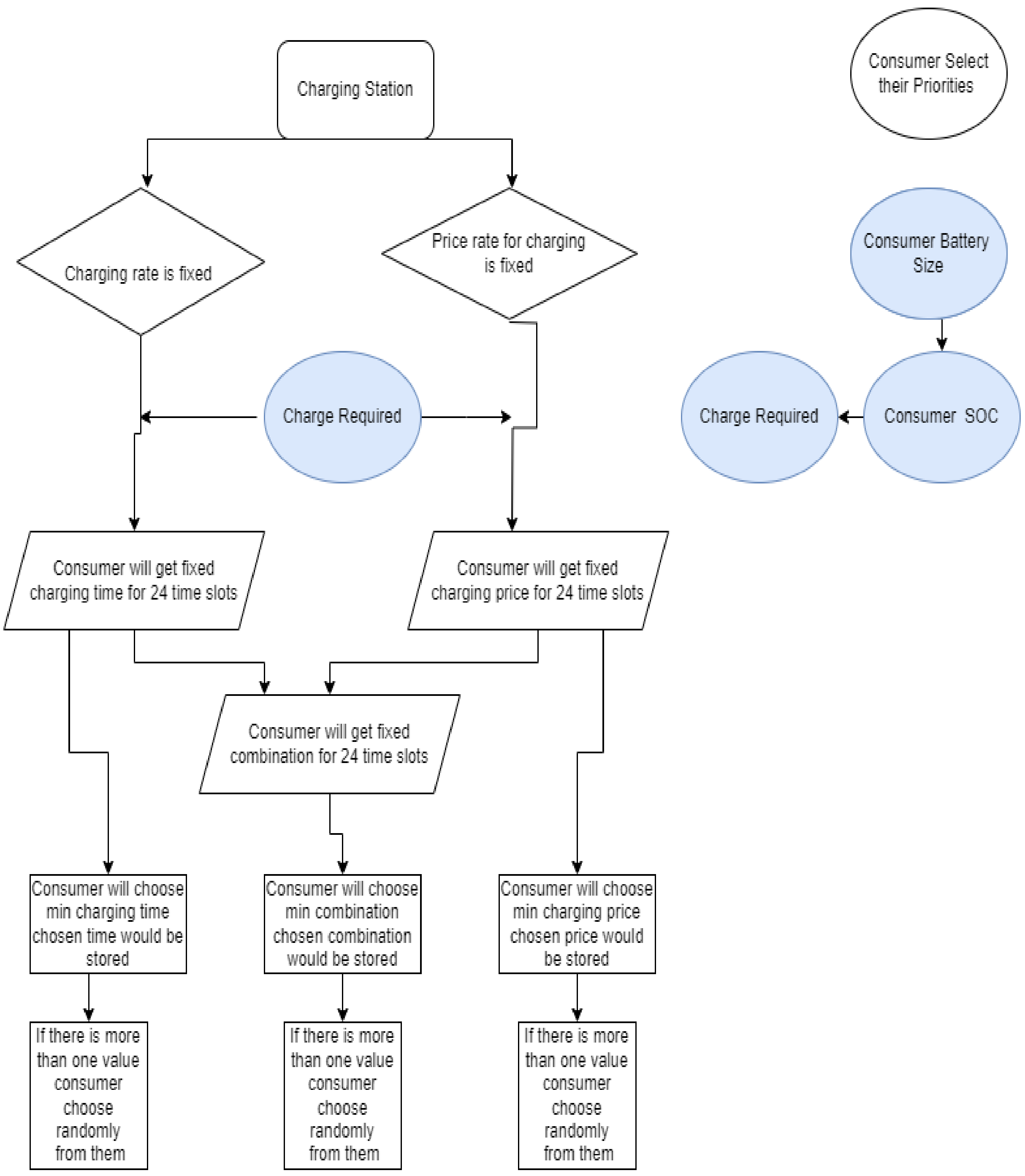
Charging Station
Consumer Select their Priorities
Charging rate is fixed
Price rate for charging is fixed
Consumer Battery Size
Charge Required
Charge Required
Consumer SOC
Consumer will get fixed charging time for 24 time slots
Consumer will get fixed charging price for 24 time slots
Consumer will get fixed combination for 24 time slots
Consumer will choose min charging time chosen time would be stored
Consumer will choose min combination chosen combination would be stored
Consumer will choose min charging price chosen price would be stored
If there is more than one value consumer choose randomly from them
If there is more than one value consumer choose randomly from them
If there is more than one value consumer choose randomly from them

Fig 3.1: Flow chart of the first part of case 1 where consumer choose their priority and store there chosen price , chosen charging time and combination

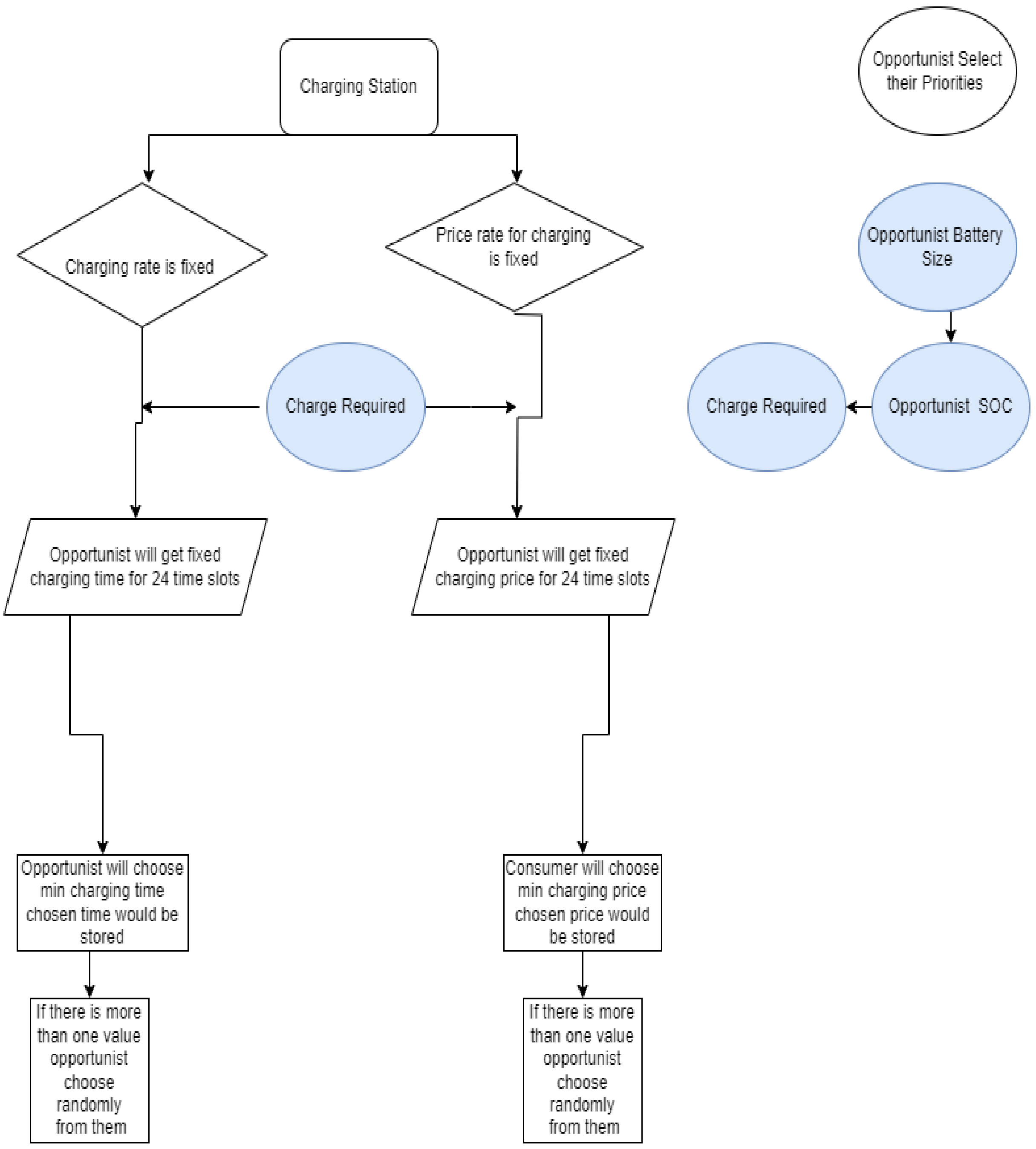
Charging Station
Charging rate is fixed
Price rate for charging is fixed
Charge Required
Opportunist will get fixed charging time for 24 time slots
Opportunist will get fixed charging price for 24 time slots
Opportunist will choose min charging time chosen time would be stored
Consumer will choose min charging price chosen price would be stored
If there is more than one value opportunist choose randomly from them
If there is more than one value opportunist choose randomly from them
Opportunist Select their Priorities
Opportunist Battery Size
Charge Required
Opportunist SOC

Fig 3.2: Flow chart of the first part of case 1 where opportunist choose their priority and store there chosen price , chosen charging time and combination

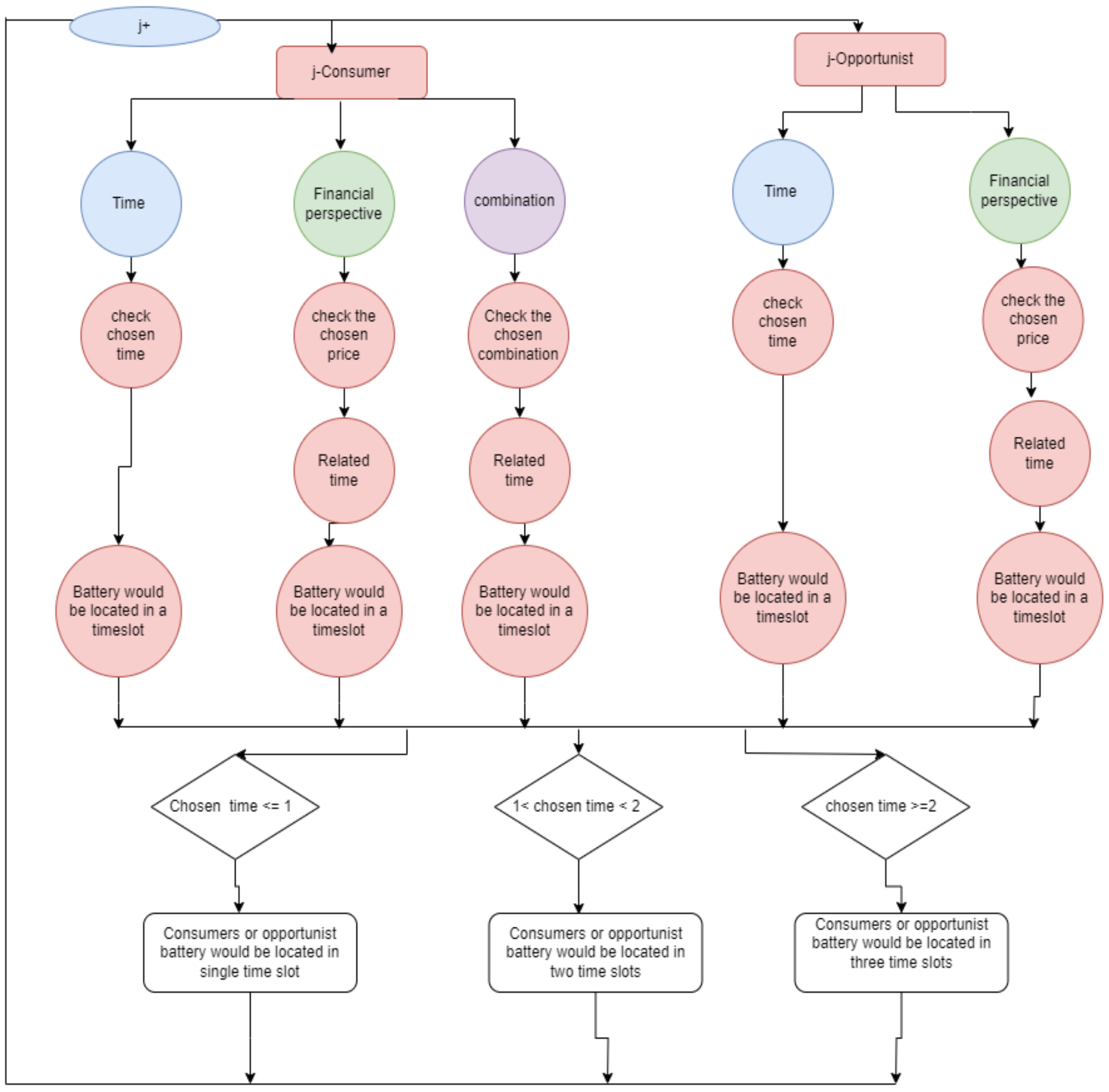
j+
j-Consumer
j-Opportunist
Time
Financial perspective
combination
Time
Financial perspective
check chosen time
check the chosen price
Check the chosen combination
check chosen time
check the chosen price
Related time
Related time
Related time
Battery would be located in a timeslot
Chosen time <= 1
1< chosen time < 2
chosen time >=2
Consumers or opportunist battery would be located in single time slot
Consumers or opportunist battery would be located in two time slots
Consumers or opportunist battery would be located in three time slots

Fig 3.3 : Flow chart of the 2and part of the case1 where consumers are located in different time slots

In the 2and part of our task,

Each consumer based on their choice they have made in the 1st task which is been stored in respective variable, their battery would be located to a time slot $b_i t_i$ as mentioned in equation (3.1), if a consumer or opportunist prioritize price then his battery would be located to a time-slot related to his chosen price. if a consumer or opportunist prioritize combination then his battery would be located to a time-slot related to his chosen combination. if a consumer or opportunist prioritize time then his battery would be located to a time-slot related to his chosen time. If a consumer's or opportunist's required time value to fully charge the battery is 1 or less than 1, consumers' or opportunist's batteries would be located in a single time slot. If the required time is between 1 and 2, it would be located in two time slots. If required, time is more than 2 his battery would be located in three time slots.

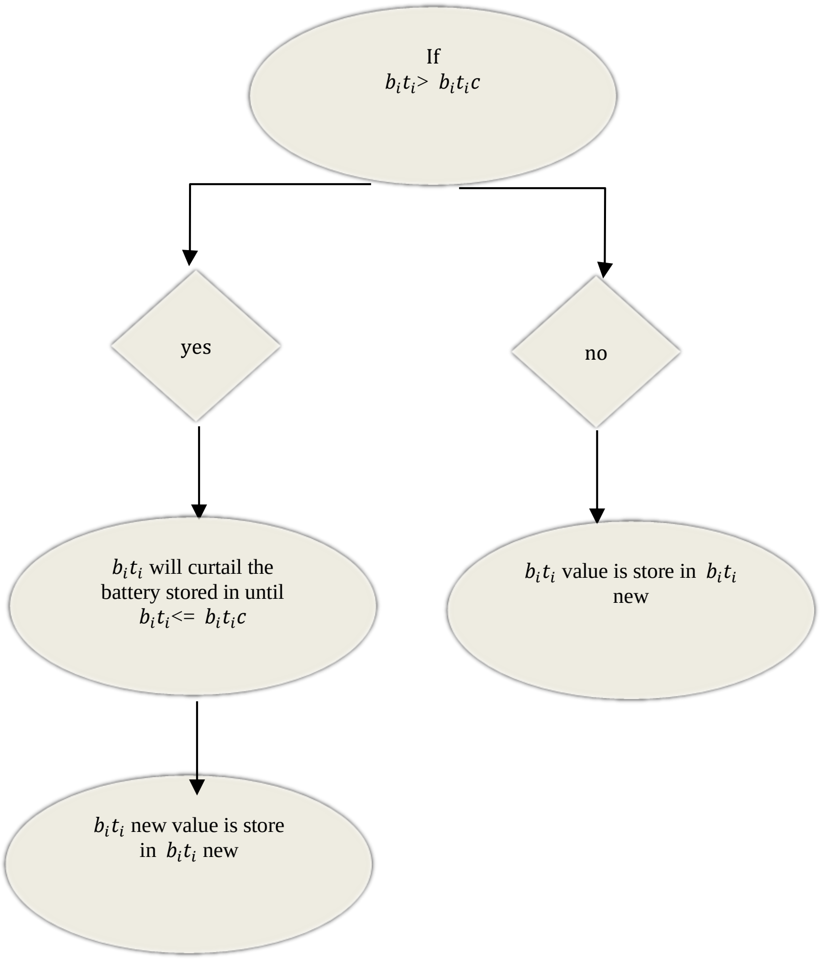
If
$b_i t_i > b_i t_i c$
yes
no
$b_i t_i$ will curtail the battery stored in until $b_i t_i <= b_i t_i c$
$b_i t_i$ value is store in $b_i t_i$ new
$b_i t_i$ new value is store in $b_i t_i$ new

Fig 3.4: Flow Chart of the third part of case1 where EVCS curtail consumer from a time slot if a time slot is overloaded

In our 3rd task,

The government's power distribution board will offer the charging station a load capacity for each hour. The capacity cannot be exceeded under any circumstances, as shown in equation (3.2).

In Fig.3.4, if the load of a timeslot becomes more than the capacity which is estimated by government's power distribution board.

Then that slot will curtail the load until the load become less or equal to capacity. After the curtailment, the new value of load for that time slot would be stored in $b_i t_i new$. And the load which is curtailed that related revenue would be added in the variable loss. Loss is ,

$$\text{Loss += Curtailed load price} \qquad 3.8$$

So, the final income of a charging station would be calculated after subtracting the loss from expected value . Final income is,

$$\text{Final Income = Expected – Loss} \qquad 3.9$$

Case2 :

For case 2, we assume the charger value is 25 kw and the price per unit varies from 5 NOK to 12 NOK.

In case 2, the charging rate of a charging station will remain constant for all the time slots, but the charging price will vary over the 24 time slots. so, according to equation (3.5)

j-Cost of charging-i= Real_Battery-j *pt_i 3.5

The consumer will get 24 different price values to charge his battery. As the price rate is varying, consumers will get diverse combinations as well. They will choose the minimum price, and that will be stored at a variable chosen price.According to equation (3.6)

j-combo-i= j-Charging rate-i* j-Cost of charging-i 3.6

For case 2, the consumer will get 24 different combinations, choose the minimum combination, and that will be stored at variable chosen combination.

Regarding to charging rate, case 2 have fixed charging rate for all time slots

j-Charging time-i= Real_Battery-j /ct_i 3.4

From equation (3.4), each consumer in case 2 will get 24 identical charging values, and the consumer will choose a charging time randomly among them. That chosen time will be stored in chosen time.The expected income is also calculated here as shown in equation (3.7)

Expected += j-Cost of charging-i 3.7

Task 1 and 2 for case 2 would remain the same as all the consumers based on their choices would be located to different time slots.

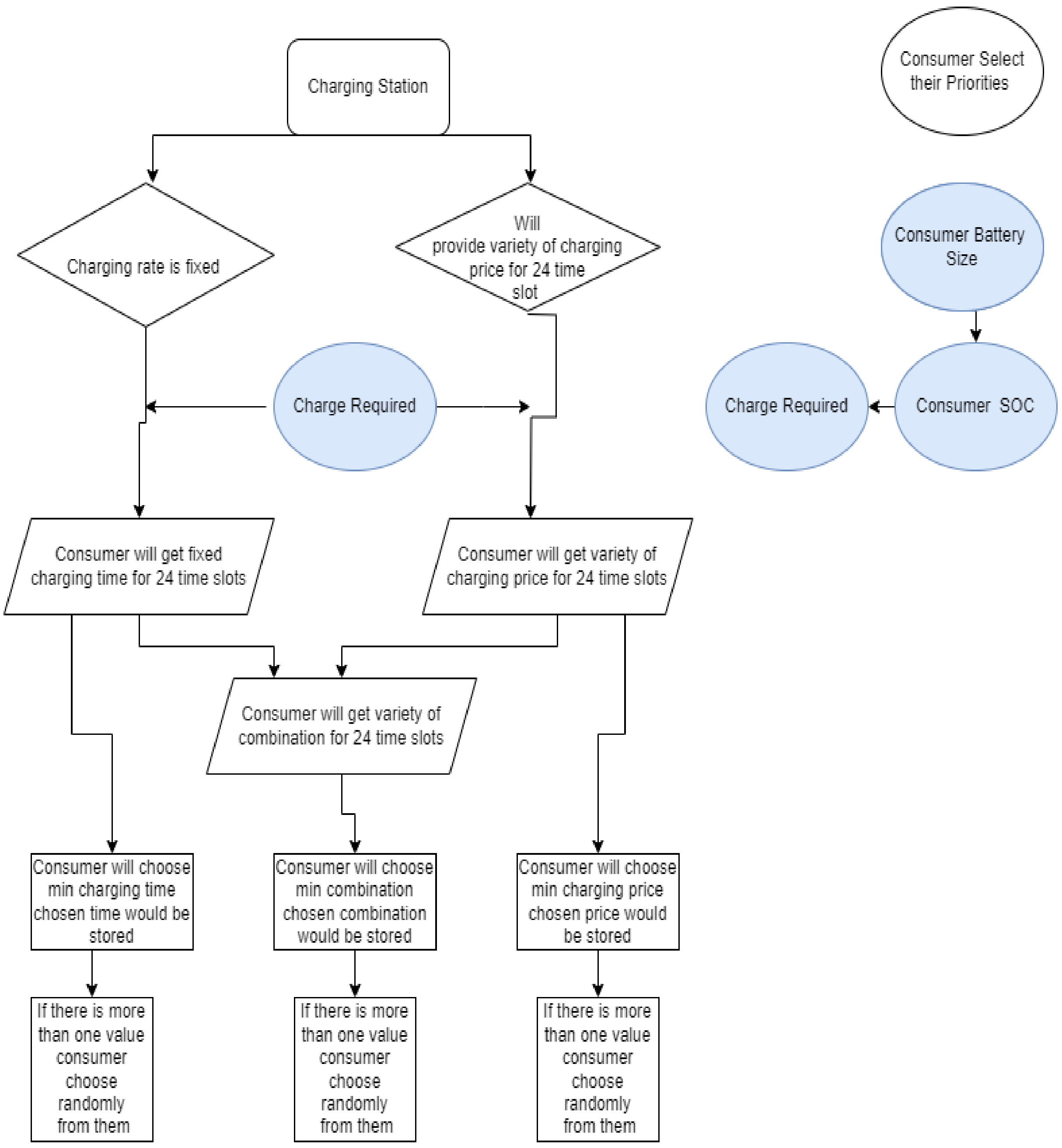
Charging Station
Charging rate is fixed
Will provide variety of charging price for 24 time slot
Charge Required
Consumer will get fixed charging time for 24 time slots
Consumer will get variety of charging price for 24 time slots
Consumer will get variety of combination for 24 time slots
Consumer will choose min charging time chosen time would be stored
Consumer will choose min combination chosen combination would be stored
Consumer will choose min charging price chosen price would be stored
If there is more than one value consumer choose randomly from them
If there is more than one value consumer choose randomly from them
If there is more than one value consumer choose randomly from them
Consumer Select their Priorities
Consumer Battery Size
Consumer SOC
Charge Required

Fig 3.5: Flow chart of the first part of case 2 where consumer choose their priority and store there chosen price , chosen charging time and combination

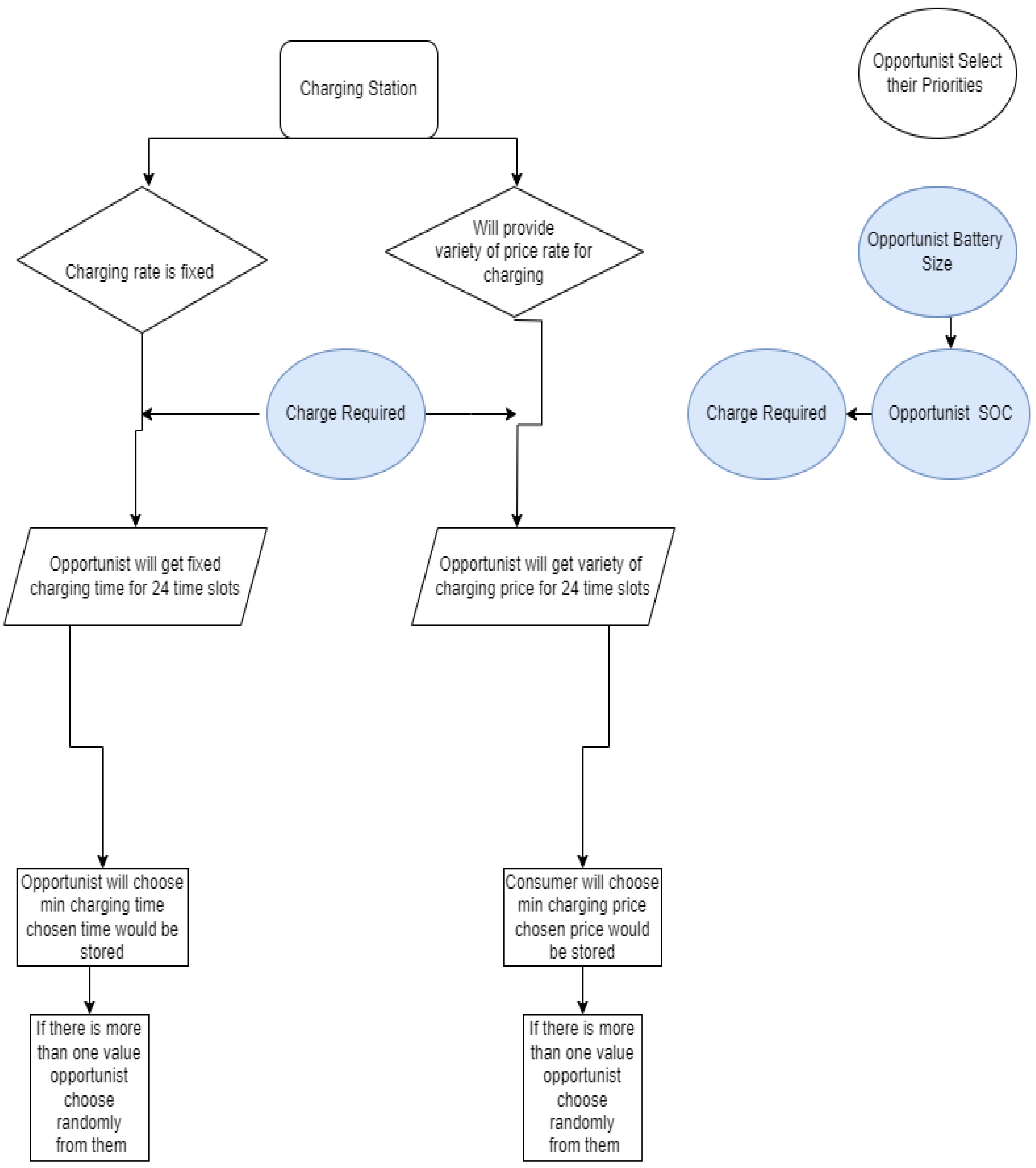
Charging Station
Charging rate is fixed
Will provide variety of price rate for charging
Charge Required
Opportunist will get fixed charging time for 24 time slots
Opportunist will get variety of charging price for 24 time slots
Opportunist will choose min charging time chosen time would be stored
Consumer will choose min charging price chosen price would be stored
If there is more than one value opportunist choose randomly from them
If there is more than one value opportunist choose randomly from them
Opportunist Select their Priorities
Opportunist Battery Size
Charge Required
Opportunist SOC

Fig 3.6: Flow chart of the first part of case 2 where opportunist choose their priority and store there chosen price , chosen charging time and combination

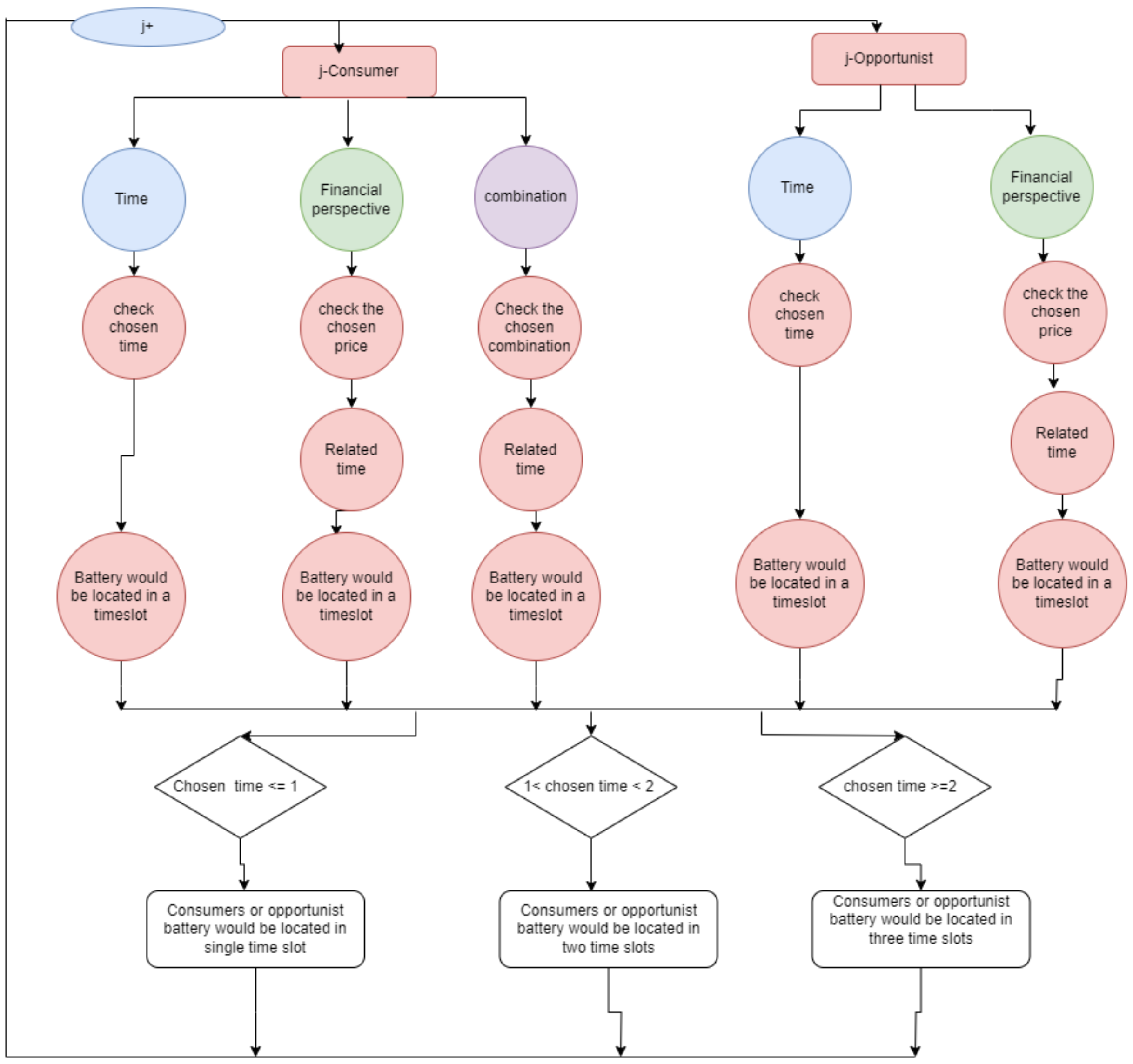
j+
j-Consumer
j-Opportunist
Time
Financial perspective
combination
Time
Financial perspective
check chosen time
check the chosen price
Check the chosen combination
check chosen time
check the chosen price
Related time
Related time
Related time
Battery would be located in a timeslot
Battery would be located in a timeslot
Battery would be located in a timeslot
Battery would be located in a timeslot
Battery would be located in a timeslot
Chosen time <= 1
1< chosen time < 2
chosen time >=2
Consumers or opportunist battery would be located in single time slot
Consumers or opportunist battery would be located in two time slots
Consumers or opportunist battery would be located in three time slots

Fig 3.7 : Flow chart of the 2and part of the case2 where consumers are located in different time slots

Task 3 for case 2 is similar to case 1, in that the charging station will curtail consumers from a time slot if the time slot is overloaded, as in equation (3.2). For case 2, as in case 1, loss and final income would be calculated as shown in equations (3.8) and (3.9).

$$\text{Loss} \mathrel{+}= \text{Curtailed load price} \tag{3.8}$$

$$\text{Final Income} = \text{Expected} - \text{Loss} \tag{3.9}$$

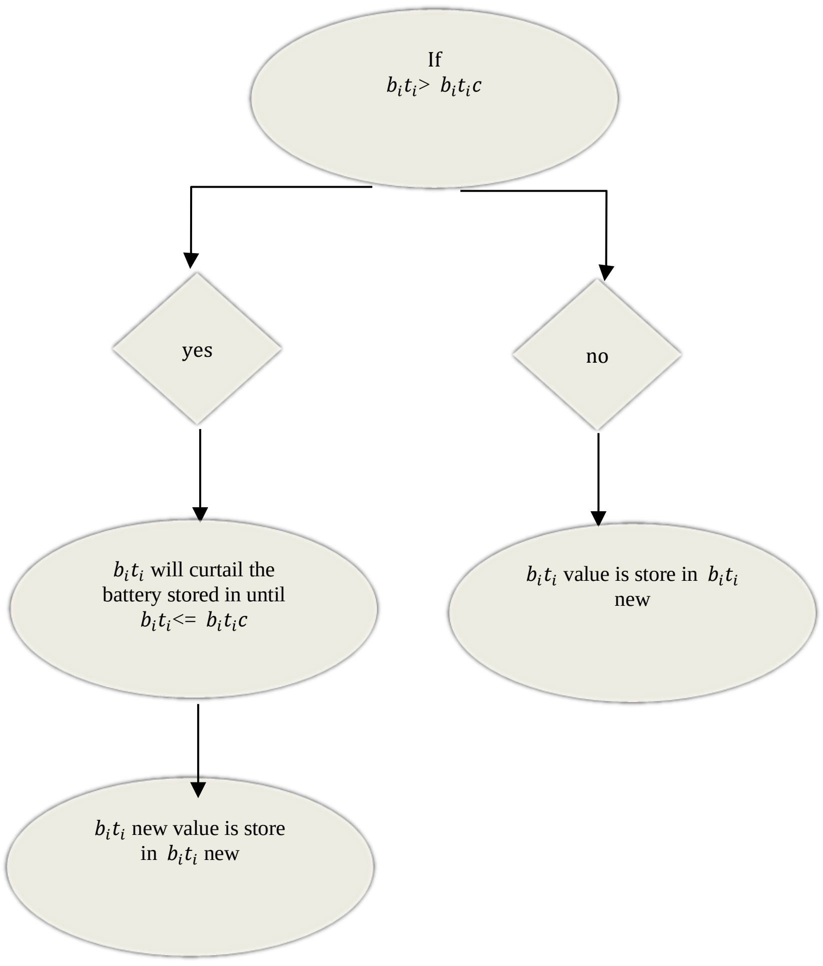
If $b_i t_i > b_i t_i c$
yes
no
$b_i t_i$ will curtail the battery stored in until $b_i t_i <= b_i t_i c$
$b_i t_i$ value is store in $b_i t_i$ new
$b_i t_i$ new value is store in $b_i t_i$ new

Fig 3.8: Flow Chart of the third part of case2 where EVCS curtail consumer from a time slot if a time slot is overloaded

Case3:

For case 3, we assume the charger value varies from 12 kw to 40kw and the price per unit varies from 5 NOK to 12 NOK.

In the third case charging station uses an adaptive charger to charge their consumers. Which varies from slow charging to fast charging.

At peak times, a charging station will provide slow charging, and at off-peak times, it will provide fast charging to its consumers. The EVCS have different prices for charging at peak load and at off-peak load. Peak price is $P_{(EVCSP)}$, and off peak price is $P_{(EVCSOP)}$. For a charging station perspective, let charging duration for EV at peak time be t_{peak} and charging duration for EV at off-peak time be $t_{off\ peak}$.

For charging duration of EVs at EVCS in peak load situations and off-peak load situations, there is a relation.

$$t_{peak} >> t_{off\ peak} \qquad 3.10$$

Means an EV have to charge at a very slow rate at peak load situation. And it can charge more faster at off-peak situation.

However, the difference in price between the peak and off-peak periods for the EVCS is,

$$P_{(EVCSOP)} > P_{(EVCSP)} \qquad 3.11$$

This implies that EVs must pay extra during off-peak hours. Additionally, charge less when demand is high.

Consequently, given the above relationship Off-peak charging costs an EV driver more money, with peak charging pricing at EVCS being the lowest.

There will be a variety of prices and charging rates for 24 distinct time slots in the third section since Charging time and price both are volatile in case 3.

The charging station will provide a price and charging rate for each hour. So, every consumer will get 24 different charging rate from equation (3.4)

j-Charging time-i= Real_Battery-j /ct_i 3.4

And minimum charging time would be stored in chosen time. Every consumer will get 24 different price as shown in equation (3.5)

j-Cost of charging-i= Real_Battery-j *pt_i 3.5

And minimum price would be stored in variable chosen price.

Same way every consumer will get 24 different combination as shown in equation (3.6)

j-combo-i= j-Charging rate-i* j-Cost of charging-i 3.6

And the minimum combination would be stored in chosen combination.All the tasks are the same for Case 3 as for Case 1 and Case 2.

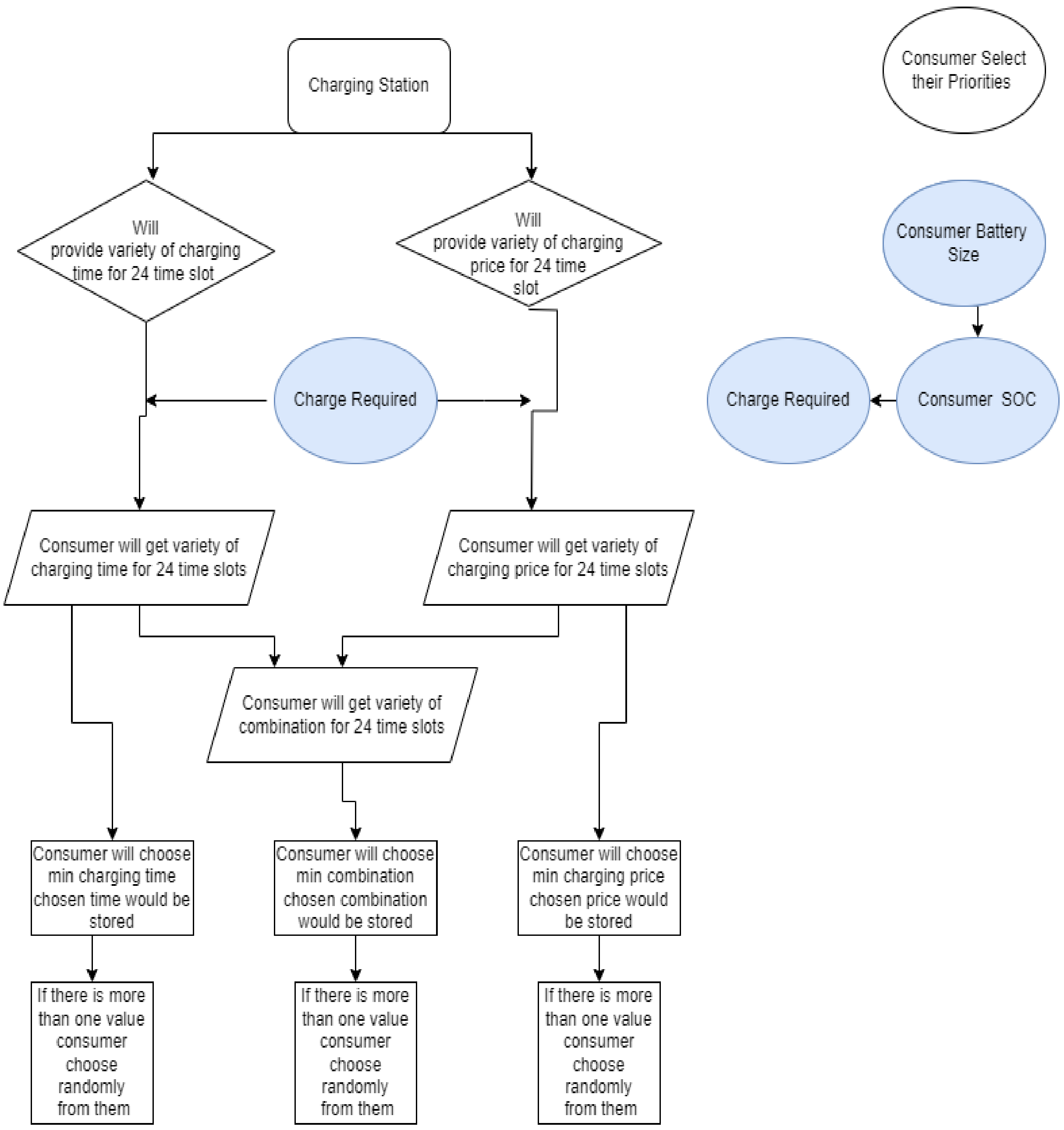
Charging Station
Will provide variety of charging time for 24 time slot
Will provide variety of charging price for 24 time slot
Charge Required
Consumer will get variety of charging time for 24 time slots
Consumer will get variety of charging price for 24 time slots
Consumer will get variety of combination for 24 time slots
Consumer will choose min charging time chosen time would be stored
Consumer will choose min combination chosen combination would be stored
Consumer will choose min charging price chosen price would be stored
If there is more than one value consumer choose randomly from them
If there is more than one value consumer choose randomly from them
If there is more than one value consumer choose randomly from them
Consumer Select their Priorities
Consumer Battery Size
Consumer SOC
Charge Required

Fig 3.9: Flow chart of the first part of case 3 where consumer choose their priority and store there chosen price, chosen charging time and combination

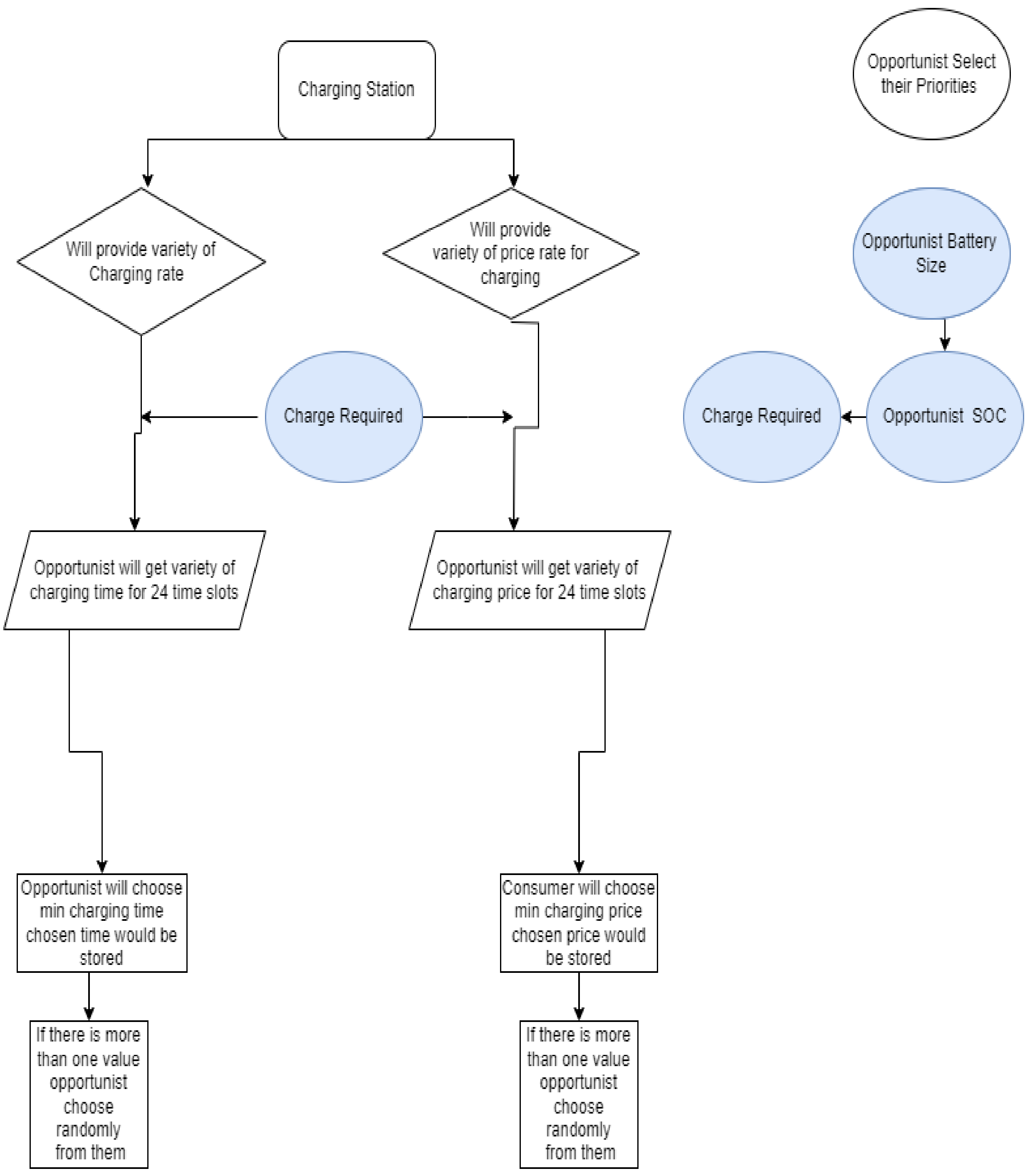
Charging Station
Opportunist Select their Priorities
Will provide variety of Charging rate
Will provide variety of price rate for charging
Opportunist Battery Size
Charge Required
Charge Required
Opportunist SOC
Opportunist will get variety of charging time for 24 time slots
Opportunist will get variety of charging price for 24 time slots
Opportunist will choose min charging time chosen time would be stored
Consumer will choose min charging price chosen price would be stored
If there is more than one value opportunist choose randomly from them
If there is more than one value opportunist choose randomly from them

Fig 3.10: Flow chart of the first part of case 2 where opportunist choose their priority and store there chosen price , chosen charging time and combination

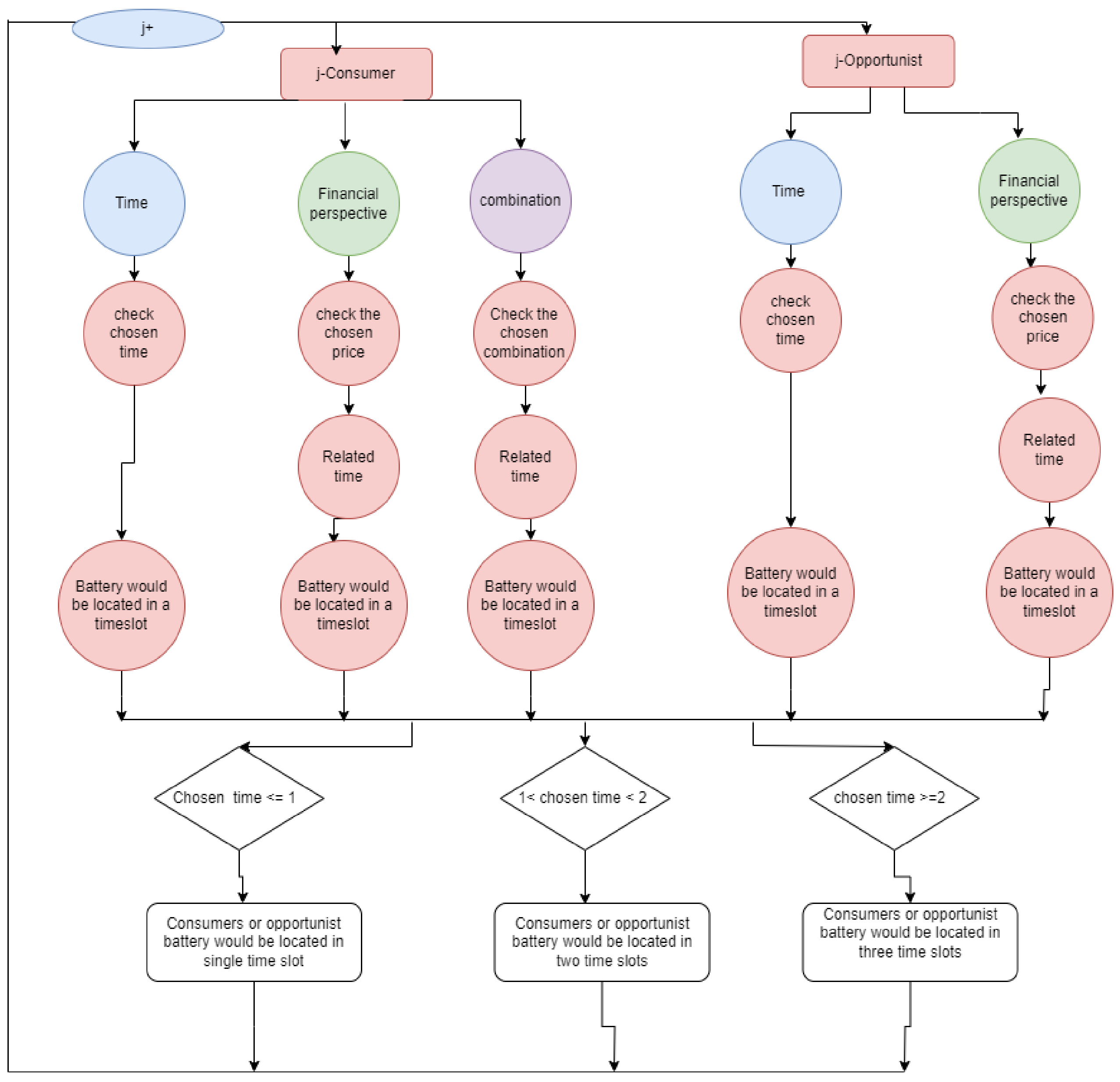
j+
j-Consumer
j-Opportunist
Time
Financial perspective
combination
Time
Financial perspective
check chosen time
check the chosen price
Check the chosen combination
check chosen time
check the chosen price
Related time
Related time
Related time
Battery would be located in a timeslot
Battery would be located in a timeslot
Battery would be located in a timeslot
Battery would be located in a timeslot
Battery would be located in a timeslot
Chosen time <= 1
1< chosen time < 2
chosen time >=2
Consumers or opportunist battery would be located in single time slot
Consumers or opportunist battery would be located in two time slots
Consumers or opportunist battery would be located in three time slots

Fig 3.11 : Flow chart of the 2and part of the case3 where consumers are located in different time slots

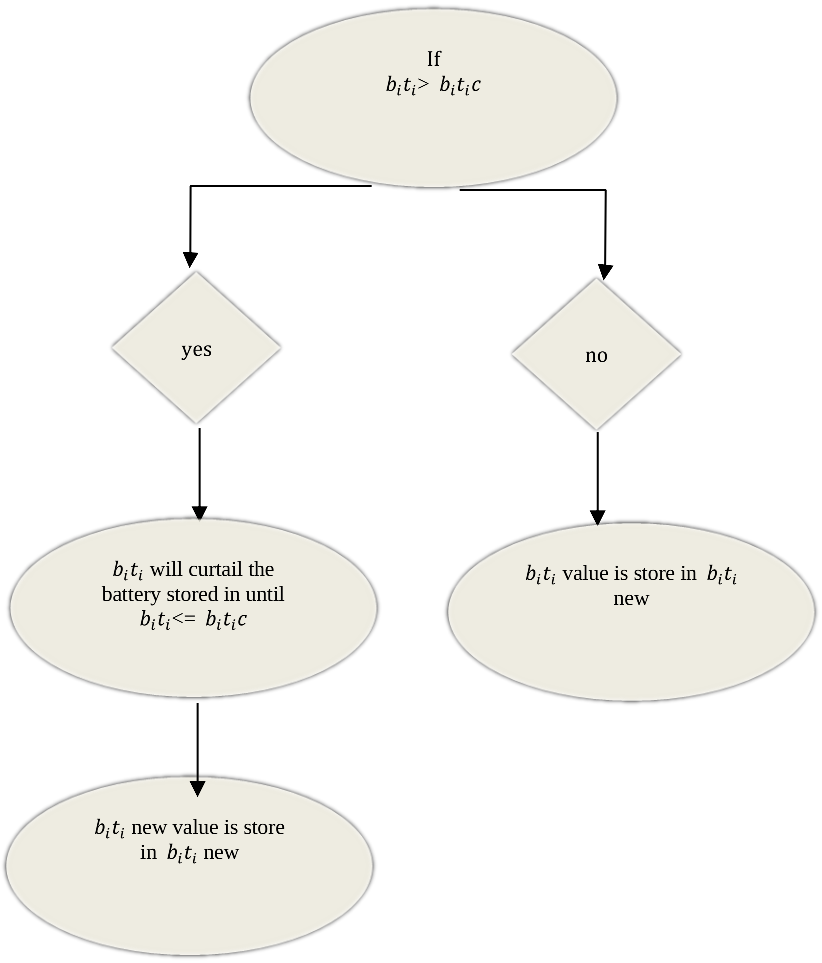
If
$b_i t_i > b_i t_i c$
yes
no
$b_i t_i$ will curtail the battery stored in until $b_i t_i <= b_i t_i c$
$b_i t_i$ value is store in $b_i t_i$ new
$b_i t_i$ new value is store in $b_i t_i$ new

Fig 3.12: Flow Chart of the third part of case3 where EVCS curtail consumer from a time slot if a time slot is overloaded

4 Results and Discussion:

Three individual codes were prepared for the three cases. After operating on all three cases in Python, some results were found. The capacity of each time slot is similar in all cases.

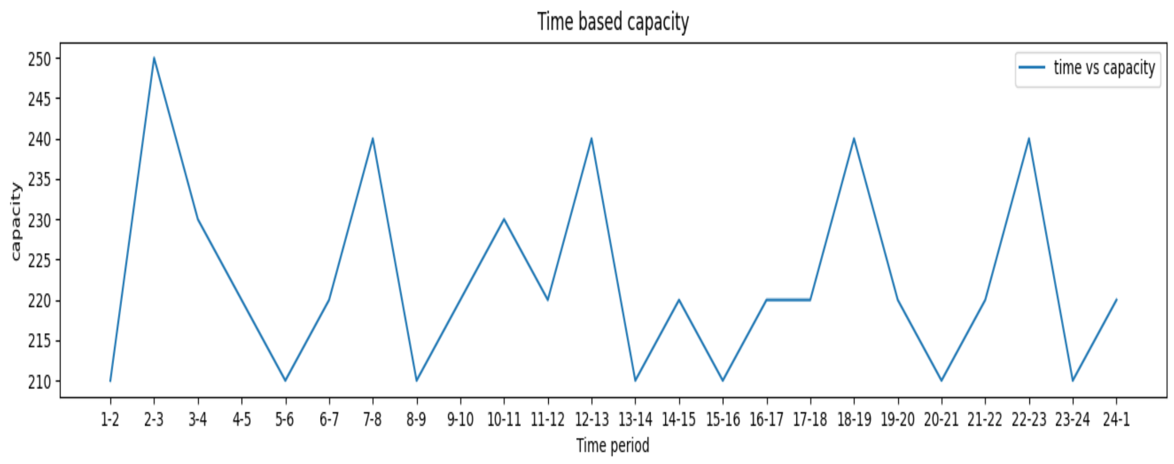

Fig 4.1: Capacity of each time slot for case1

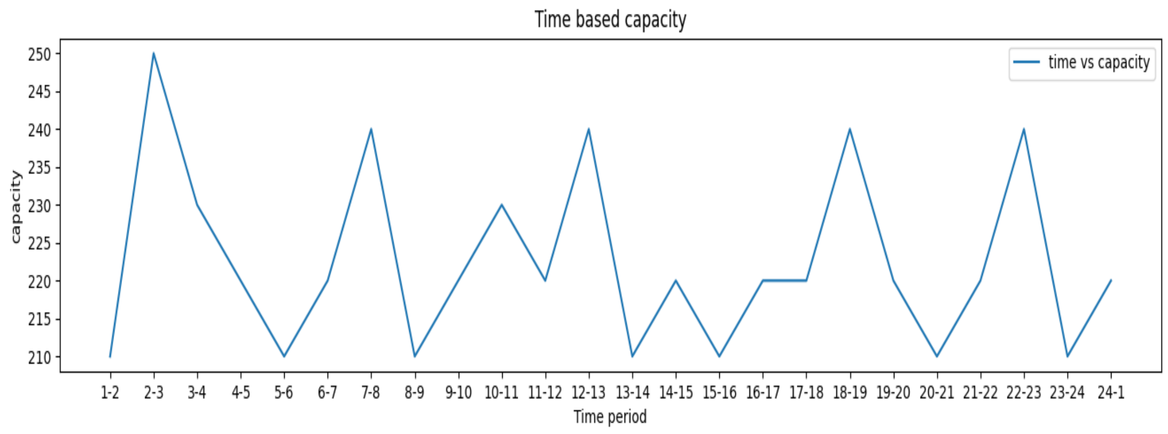

Fig 4.2: Capacity of each time slot for case2

Case 1 has a price and charge rate that are fixed for all slots, as shown in Fig 4.4. As mentioned in case 3, the price and charging rate both vary with time, while in the conventional method, which is case 2, the price varies over time but the charging rate remains the same for each time slot, as we can see from Figs 4.6 and 4.5. To understand it better, we will see it from the consumer's perspective. Because the Fig 4.6 gives the visuals of charging rate and price vs. time, it implies that whenever the charging rate goes high, the price goes high, and when the charging rate goes down, the price also goes down.

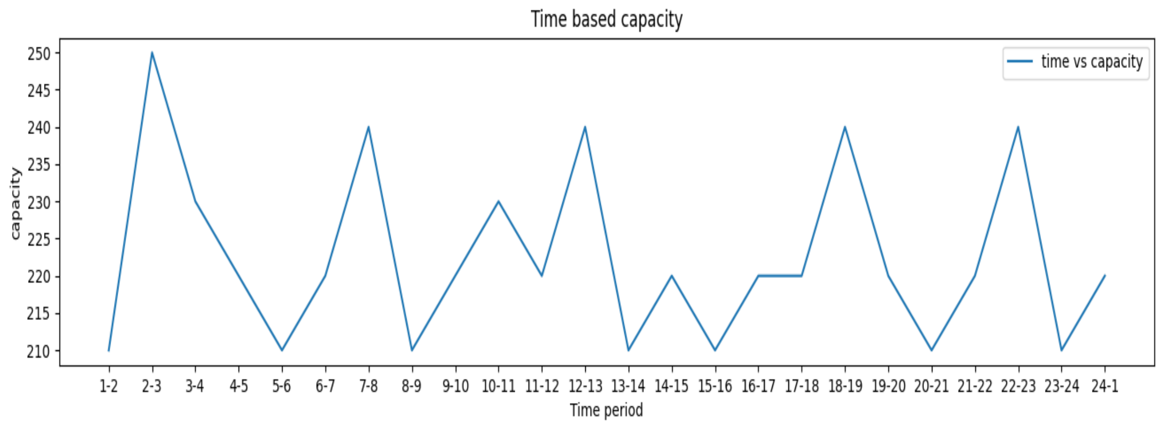

Fig 4.3: Capacity of each time slot for case3

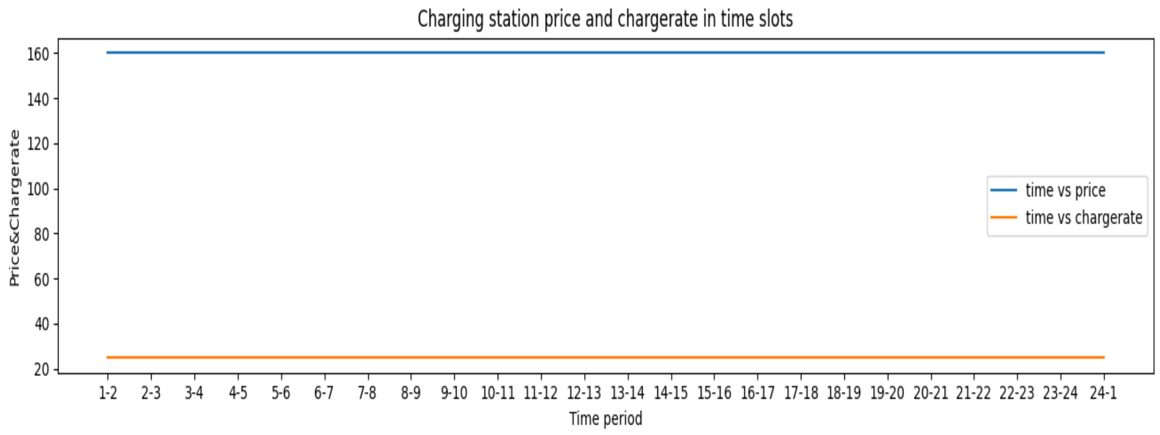

Fig 4.4: Charging station price and charge-rate in time slots in case 1

When we see this method from the consumer's perspective, we can see in Fig 4.9 for case 3 that when a consumer can charge his vehicle quickly, the required time is less, and the consumer has to pay more. When the required charging time is high, the consumer has to pay less.

According to the circumstances listed in case 2 in Fig 4.8, the price is erratic but the charge rate is constant. We can observe from Fig 4.7 which represents case 1 that both price and charging time are consistent.

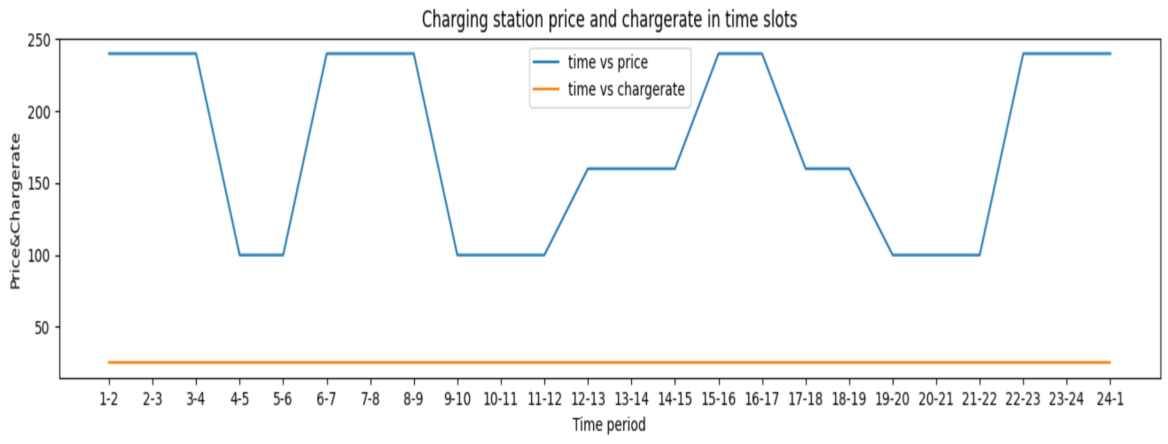

Fig 4.5: Charging station price and charge-rate in time slots in case 2

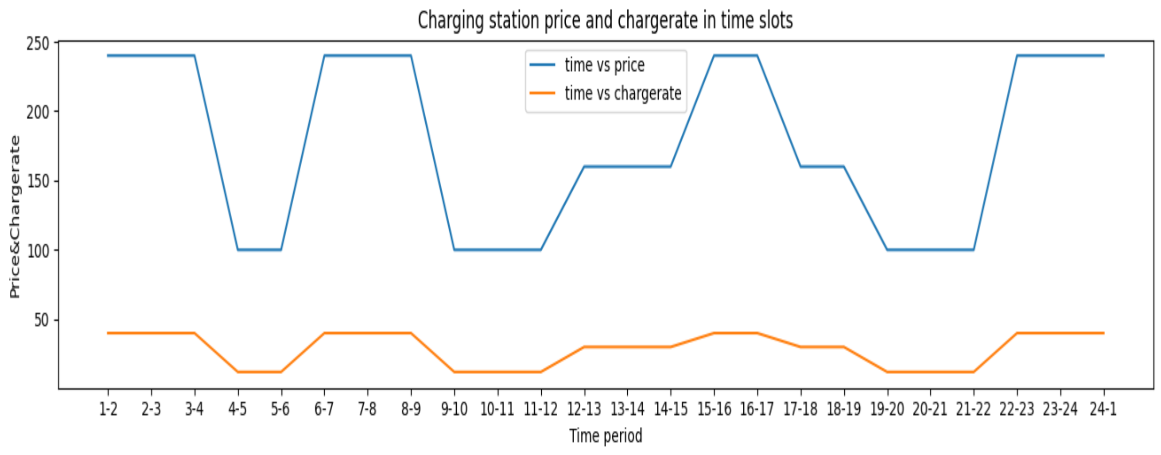

Fig 4.6: Charging station price and charge-rate in time slots in case 3

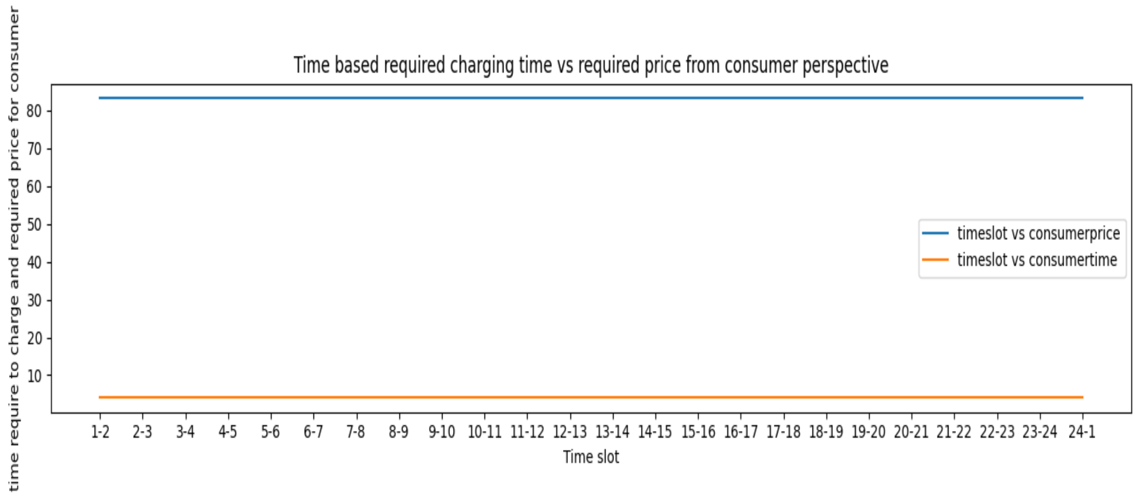

Fig 4.7: Required charging time and price in timeslots from consumer perspective case1

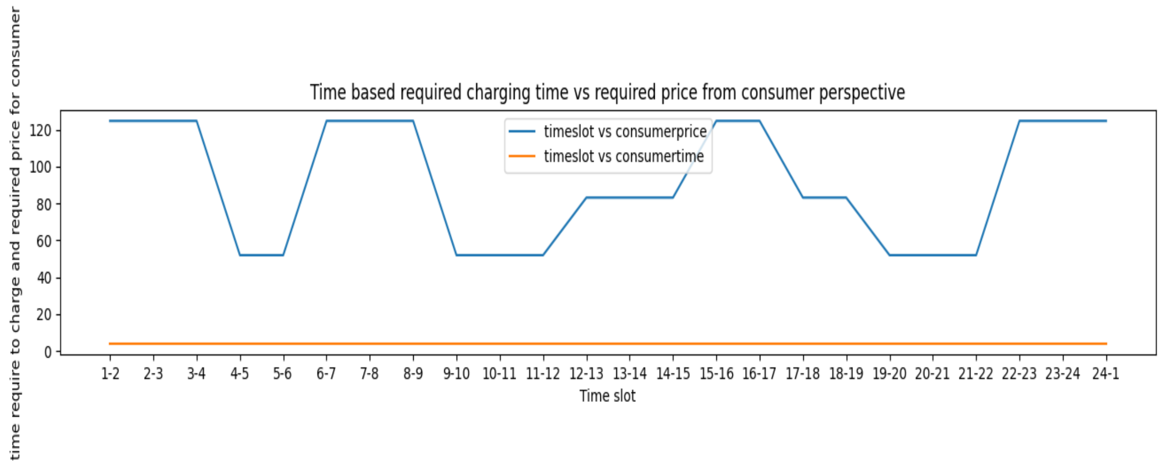

Fig 4.8: Required charging time and price in timeslots from consumer perspective case2

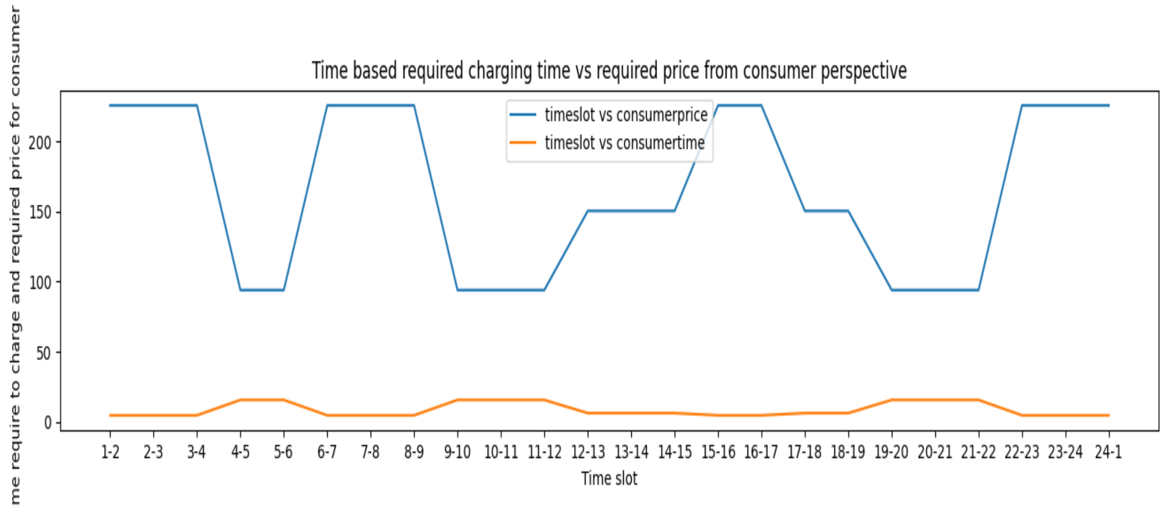

Fig 4.9: Required charging time and price in timeslots from consumer perspective case3

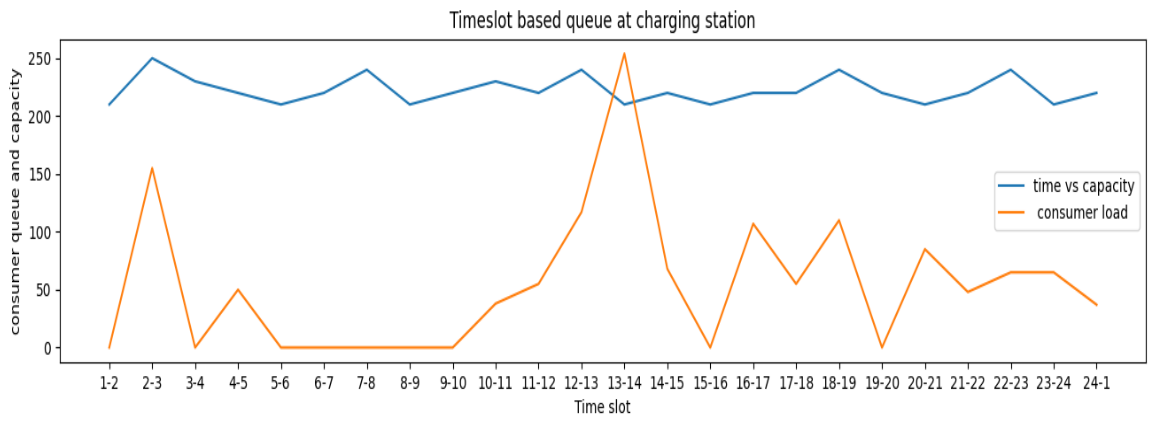

Fig 4.10: Queue at charging station at different time-slot case1

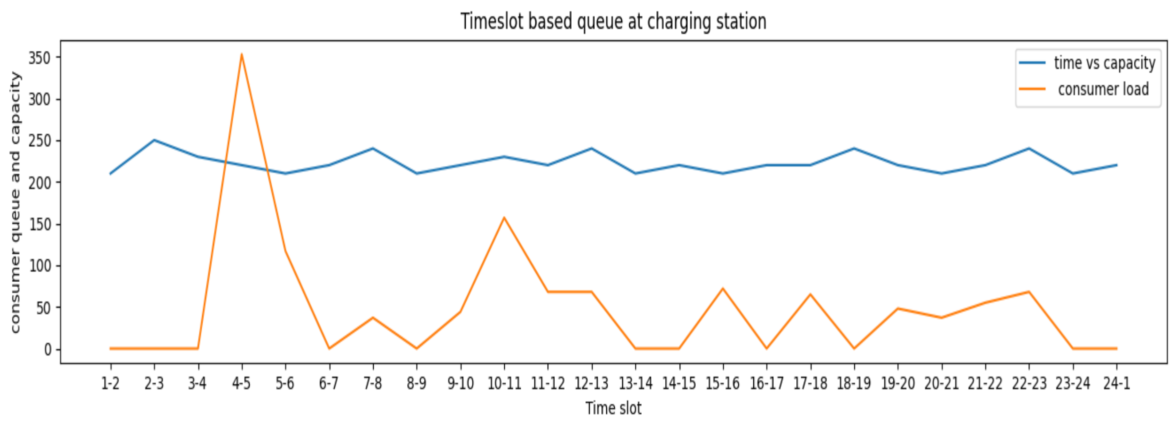

Fig 4.11: Queue at charging station at different time-slot case2

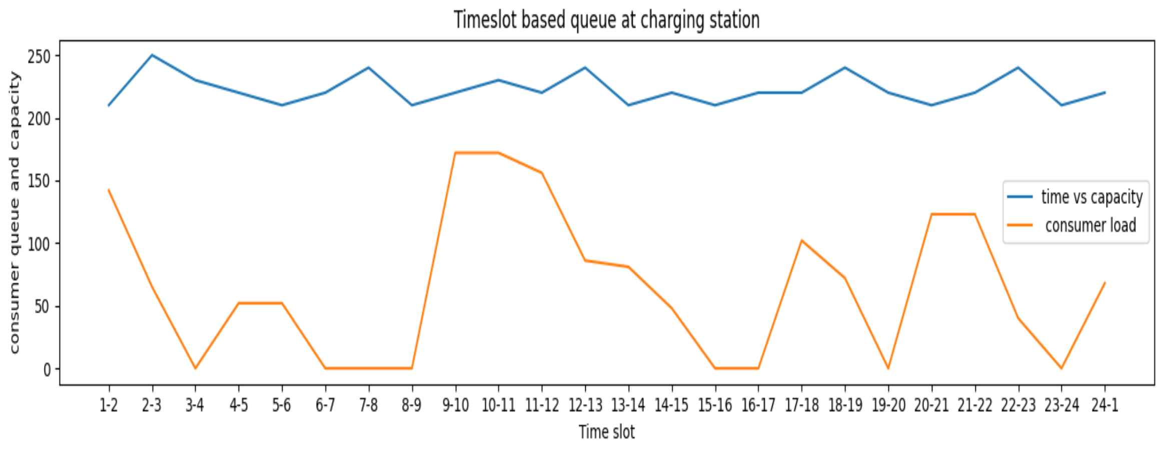

Fig 4.12: Queue at charging station at different time-slot case3

The outcome of the three cases is shown in Figs 4.10, 4.11, and 4.12. As we can see in Fig 4.11, which represents the output of case 2, there is a spike at the 3 to 6 time slots, which is breaching the capacity line. So, the charging station had to curtail some consumers and also face some losses, as described in the methodology section. The result of case1 is shown in Fig. 4.10, which also exhibits a spike between time slots 12 to 15. As a result, charging stations for case 1 must reduce some consumer spending and suffer some losses. On the other hand, Fig 4.12 demonstrates improved consumer queue coordination. where no customers must be removed from any slots by the charging station. In case 2 charging station limited the consumer's queue to modify the load within the capacity restriction, as shown in Figs 4.14 and 4.17. As seen in Figs.4.13 and 4.16, a similar amount of work is done in case 1.

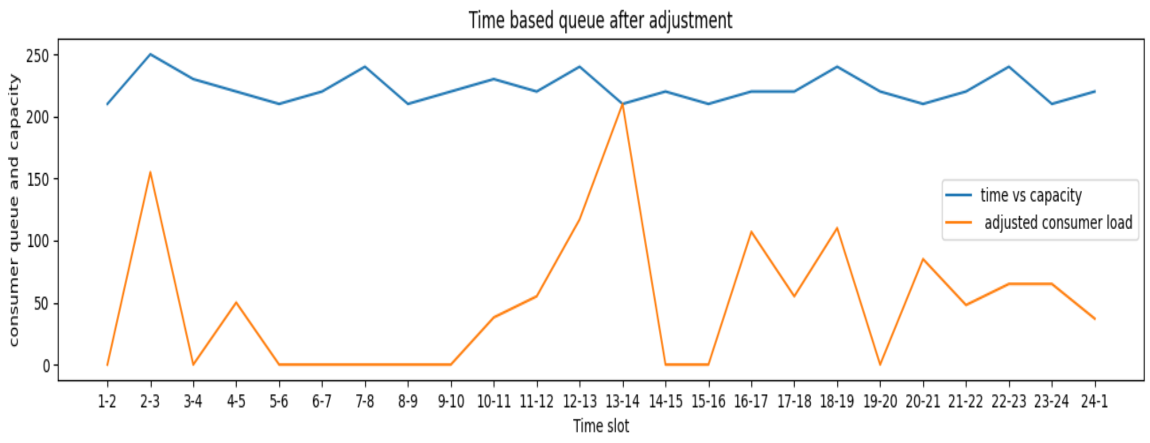

Fig 4.13: Queue at charging station after adjustment case1

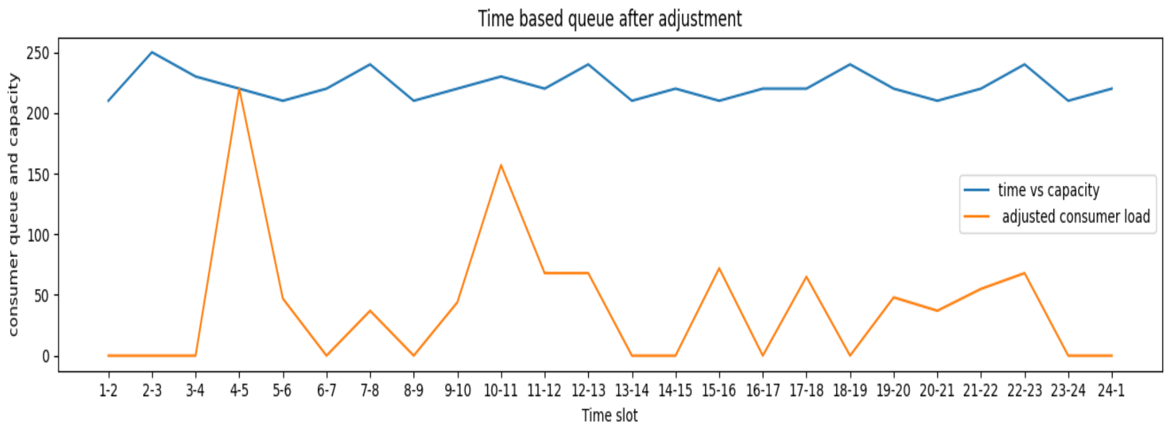

Fig 4.14: Queue at charging station after adjustment case2

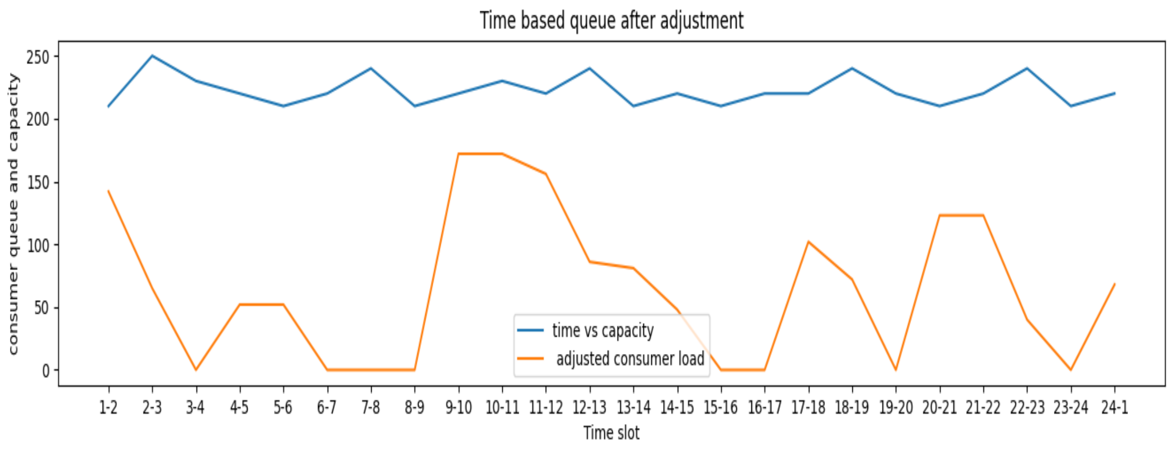

Fig 4.15: Queue at charging station after adjustment case3

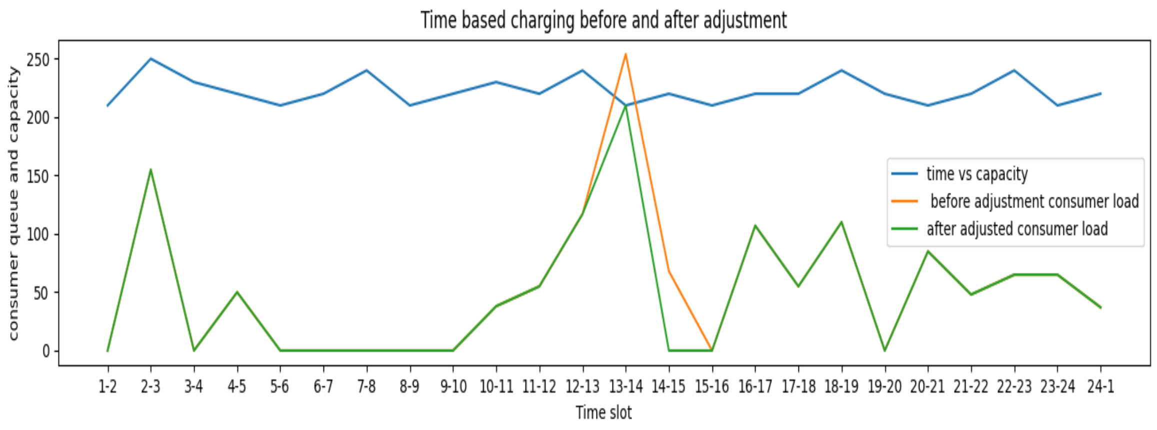

Fig 4.16: Queue at charging station before and after adjustment case1

But as can be seen from Figs 4.15 and 4.18, the charging station didn't need to change its queue for case 3. The before and after graphs for case 3 are therefore the same.

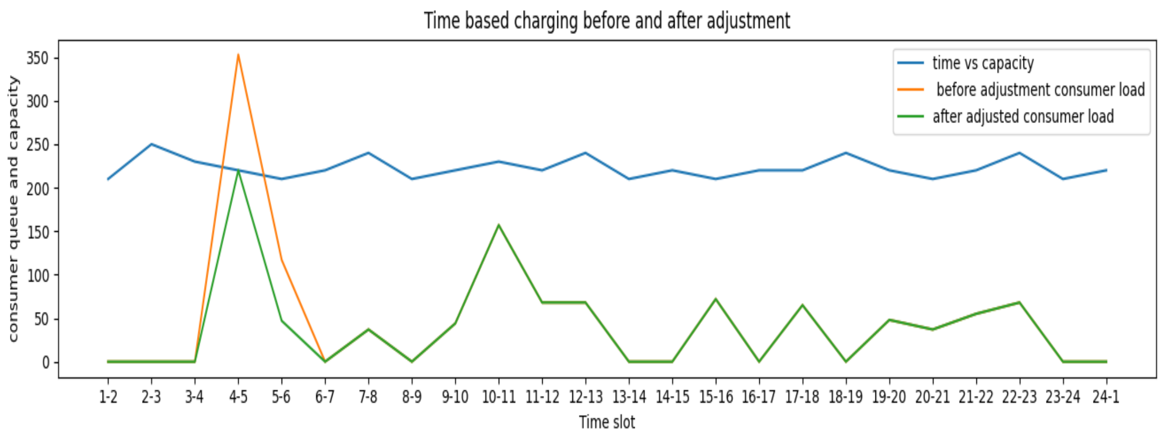

Fig 4.17: Queue at charging station before and after adjustment case2

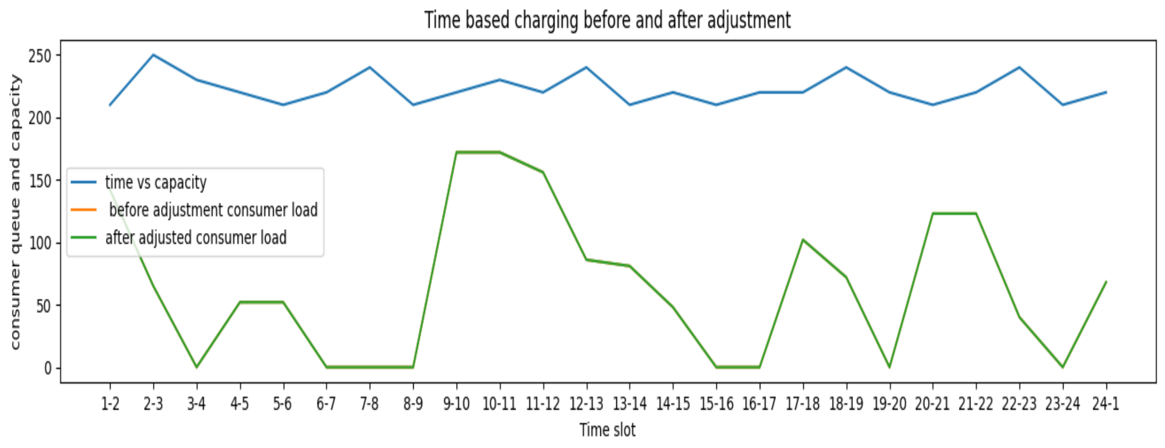

Fig 4.18: Queue at charging station before and after adjustment case3

to more clearly grasp why this is taking place. First, customer choice is to be time-prioritized, combination-based, or financial, as stated in the approach; it is random for all the cases. The consumer's selection of a time period is likewise arbitrary. It appears that all three cases have the same conditions for the bulk of the cases. The charge rate, however, distinguishes the three cases. Case 1 has a fixed charge and price rate for each time slot, in contrast to case 2, which has a variable price rate but a set charging rate. In case 3, charge rates, prices, and combinations are involved in dividing the customers among the various time slots.

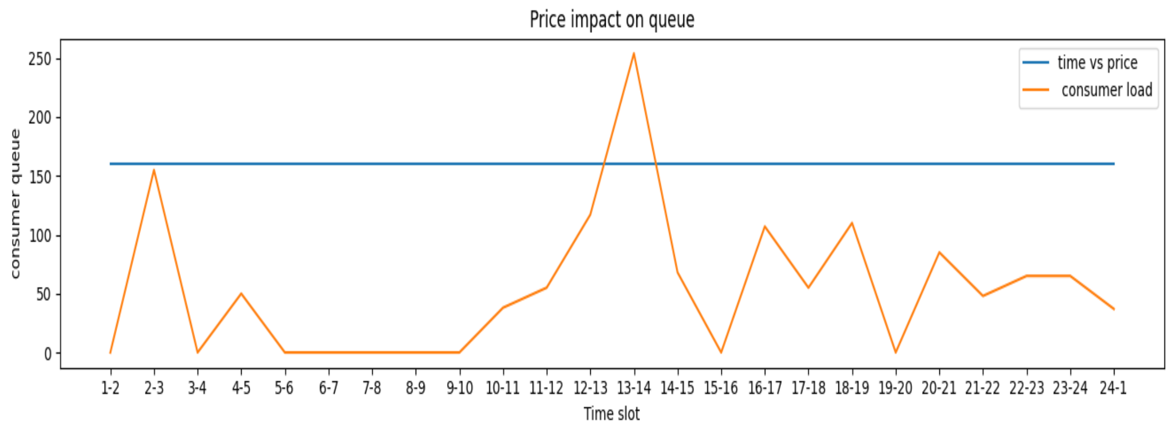

Fig 4.19: Price impact on attracting consumer in different time slot case1

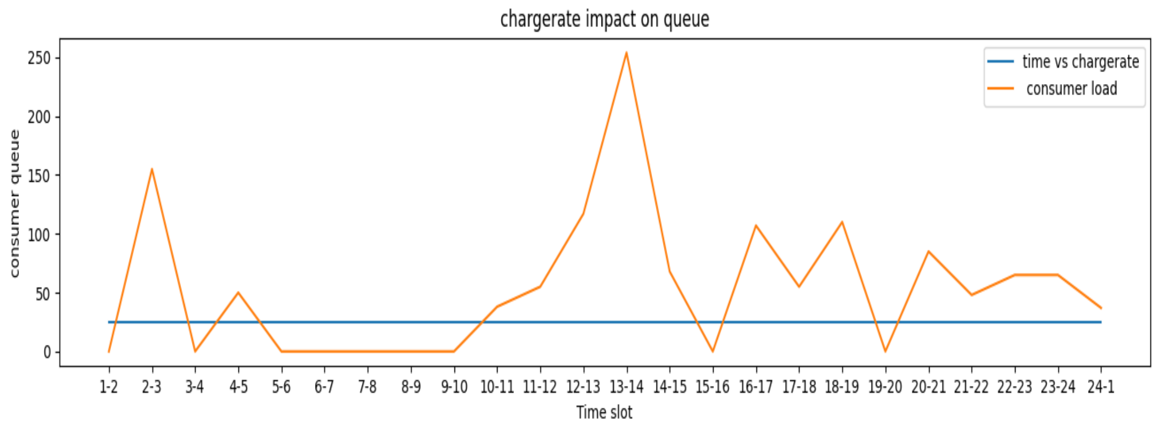

Fig 4.20: charge-rate impact on attracting consumer in different time slot case1

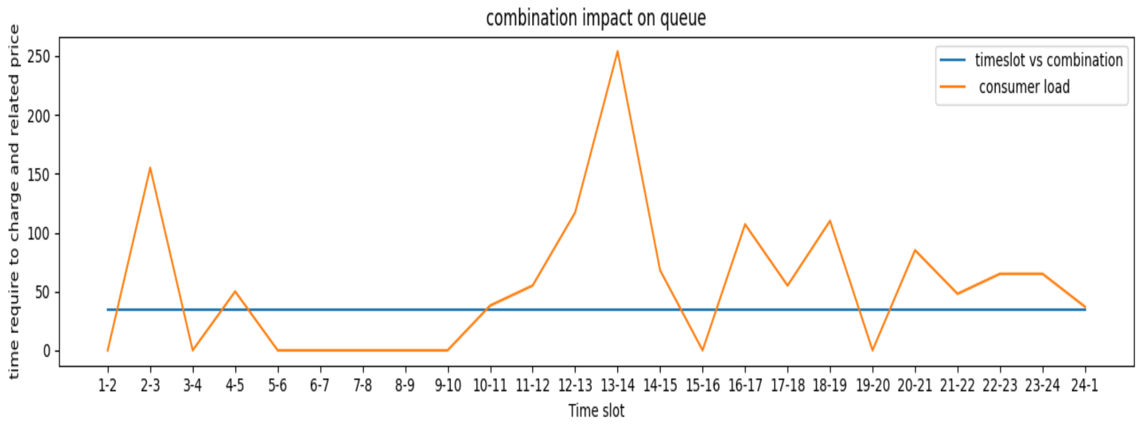

Fig 4.21: Combination impact on attracting consumer in different time slot case1

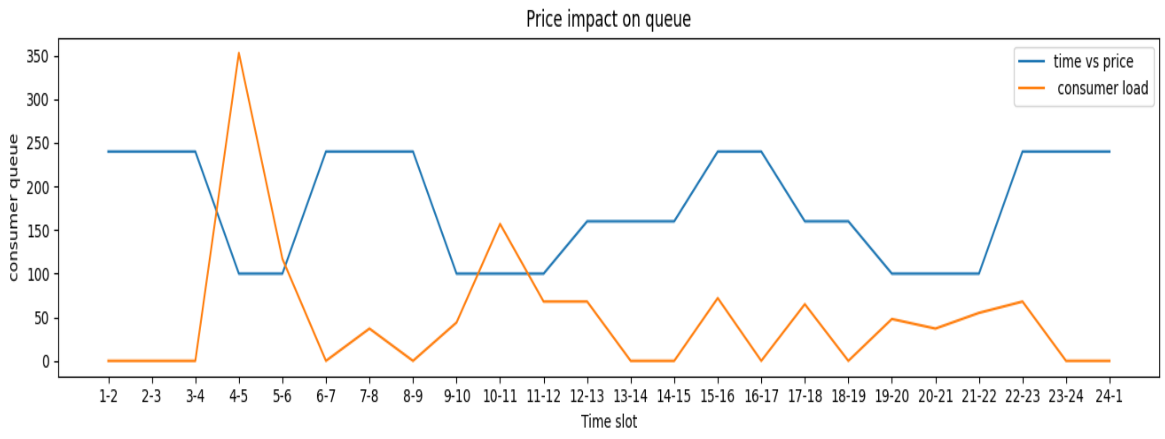

Fig 4.22: Price impact on attracting consumer in different time slot case2

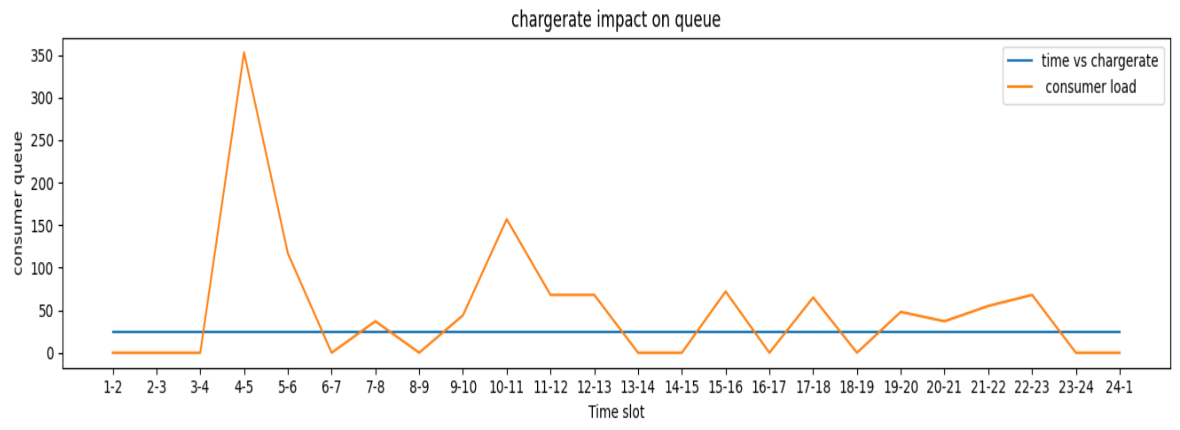

Fig 4.23: Charge-rate impact on attracting consumer in different time slot case2

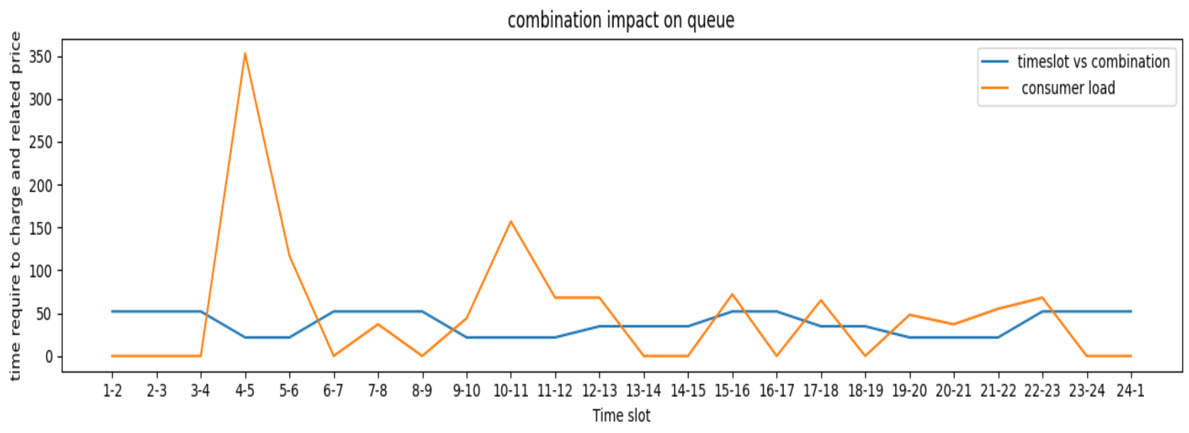

Fig 4.24: Combination impact on attracting consumer in different time slot case2

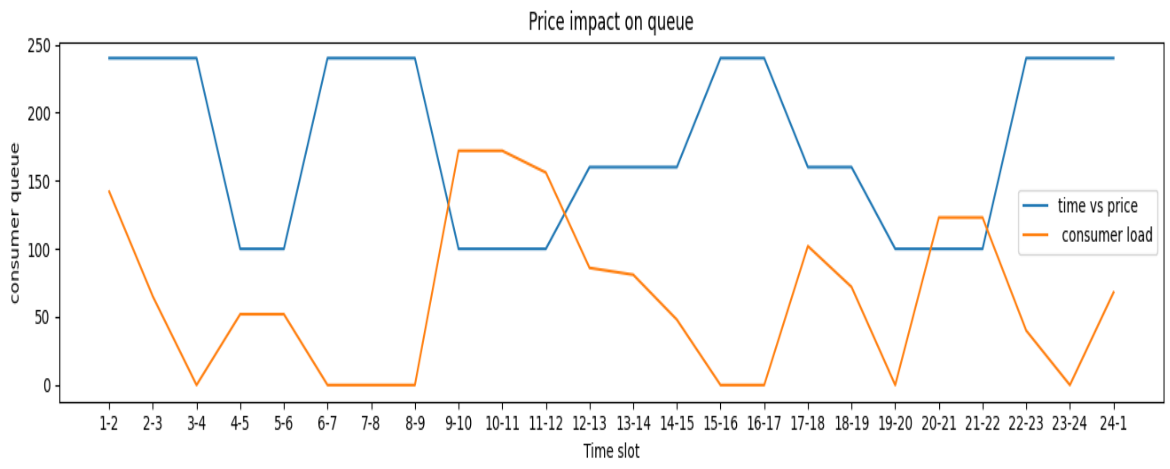

Fig 4.25: Price impact on attracting consumer in different time slot case3

From Figs 4.19, 4.20, and 4.21, which represent case 1, it is difficult to say which one attracted how many consumers, as price, charge-rate, and combination lines are flat. Here, all the consumers choose their time slot randomly.

Fig 4.22 shows in case 2 how price has been able to attract the consumer toward 3 to 6, 9 to 12, and 20 to 23 time slots. Fig 4.23 shows that for case 2, the charging rate is constant, so it doesn't make changes in the queue. But Fig 4.24 shows that the case2 combination attracted consumers in the slots 12 to 13, 17 to 19, and 20 to 22.

As we can see from Fig 4.25, for case 3, in time slots 4 to 6, 9 to 12, 20 to 23, prices have enticed the consumer to choose these slots. As we can see from Fig 4.26 of Case 3, in time slots 1 to 3, and 24 to 1, the charging rate was able to persuade the consumers to choose these slots. From Fig 4.27 of Case 3, it is visible that the time-slot 13–15 and 17–19 combinations have been able to attract the consumer to choose the respective slots.

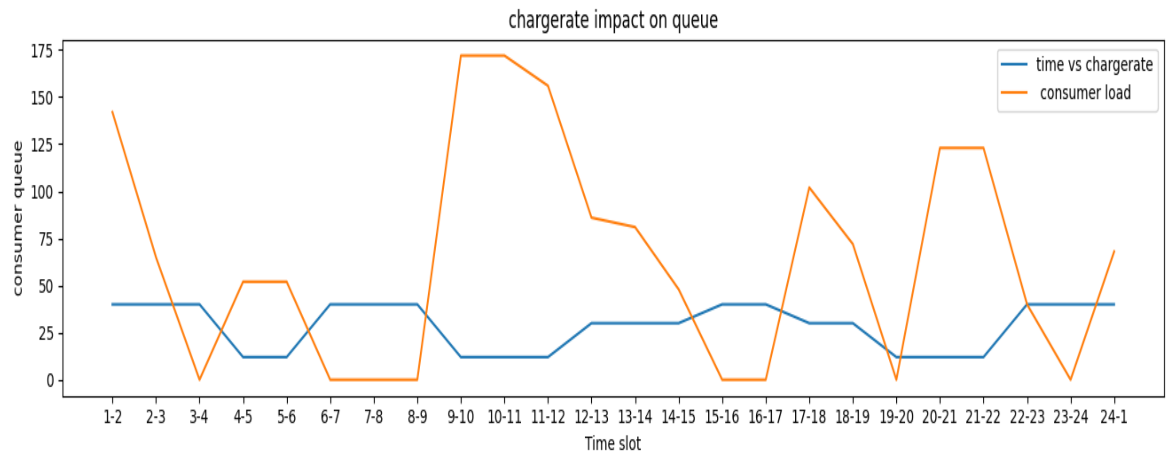

Fig 4.26: Charge-rate impact on attracting consumer in different time slot case3

Though, from Fig 4.23, we can observe that the charging rate is constant, it still attracts some consumers. In case 2, each consumer, depending on their battery size and soc, gets 24 similar values, so consumers who choose to prioritize time in case 2 choose a time slot randomly from 24 options.

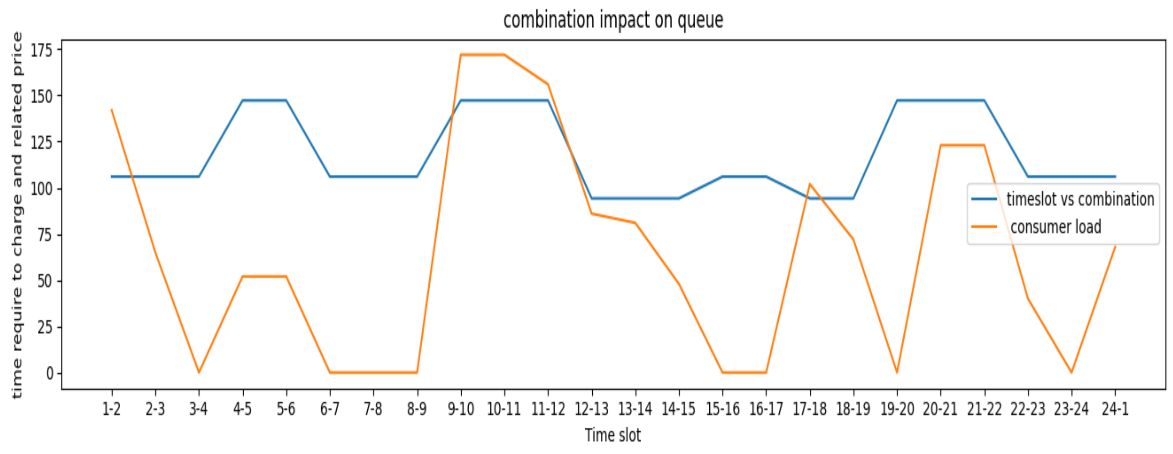

Fig 4.27: Combination impact on attracting consumer in different time slot case3

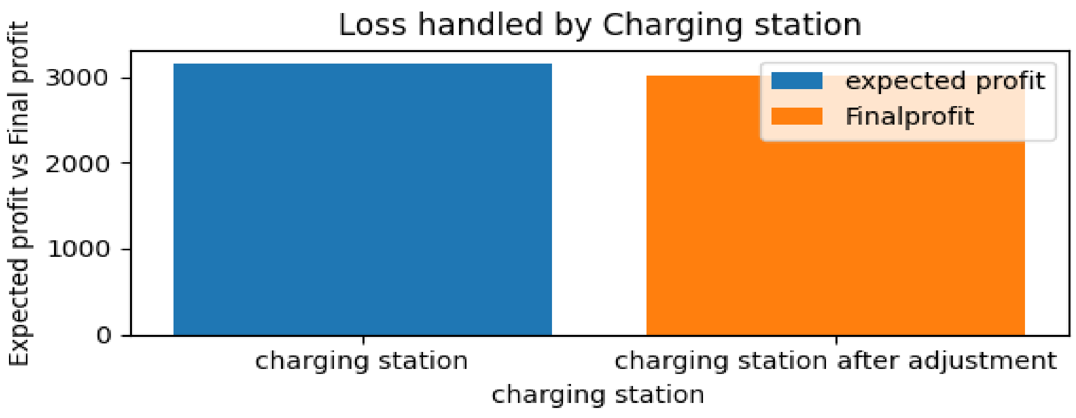

Fig4.28: Comparison of expected revenue and final revenue for case1

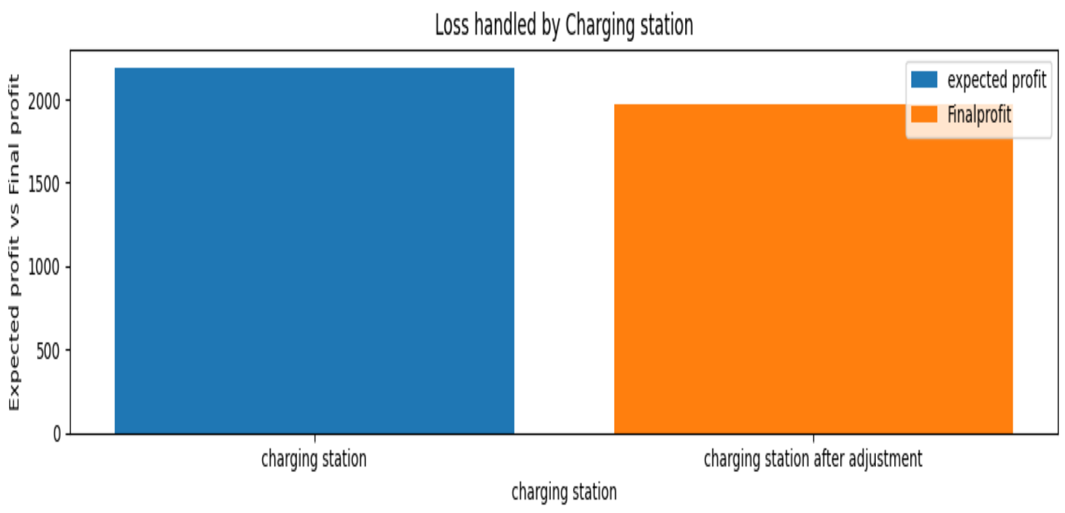

Fig 4.29: Comparison of expected revenue and final revenue for case2

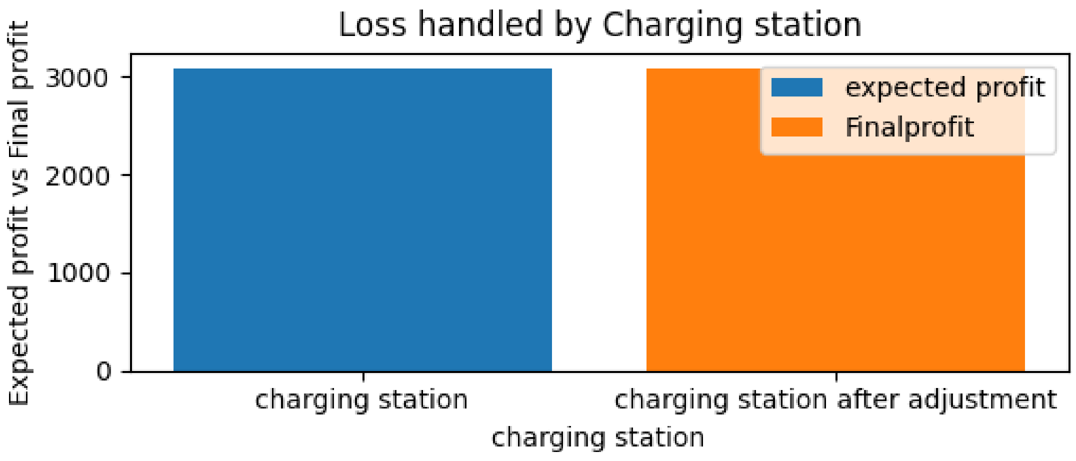

Fig 4.30: Comparison of expected revenue and final revenue for case3

Table 4.1

		Expected profit	Final profit	Loss percentage
Charging station	Case1	2700	2442	10%
Charging Station	Case2	2193	1973	10%
Charging station	Case3	3517.9	3517.9	0%

We can observe from Fig 4.30 that for case 3, the charging station generated the money they had anticipated. As can be seen in Figs 4.29 and 4.28, in cases 2 and 1, charging stations didn't generate the promised money since they had to limit some clients, and from Table 1, we can observe that in both cases they lost 10% of it.

According to the findings, in case 1, all three factors are constant, so none of them are actually playing a part in the distribution of consumers in different time slots. As consumers choose their slot randomly.

whereas case 2's manipulation of the consumer relies mostly on pricing. Although combination plays a role, it is also the result of charging time and price being multiplied. Where the charge period is fixed, the price is changeable, creating unique combinations for each customer.

We can see that case 3's manipulation of the consumer into selecting a different time slot involved three factors: price, time, and combination. All these factors are participating in distributing the consumers in different time slots, which helps the charging station accommodate all the consumers and fulfill the target of anticipated profit.

In this research, a probabilistic conditional statement-based algorithm was built for three cases. In case 1, the charging rate and price of EVCS remain constant. Case 2 is the conventional EVCS system for charging the electric vehicle where charging rate is fixed but the price of charging is unstable, and case 3 is the price and charge rate varying EVCS system, which is slightly different from the conventional one.

An analysis of all systems is done for a 24-hour period. There are 24 time slots spread across the 24 hours. The charging station in the traditional EVCS system offers a range of fees for recharging an electric vehicle. For 24 time periods, the charging rate or chargers are fixed.

While there are many different price rates in the EVCS system for case 3, there are also 24 variations in the charge rates. But in case 1 as mentioned earlier price and charging-rate both are constant.

With 20 consumers, we ran this simulation. We have divided our consumers into two groups. 15 of them are local consumers, and the rest are opportunists. The local consumers code is shown in Figs. 3.1, 3.5, and 3.9 for the respective cases. The remaining five are opportunists, Opportunists code is shown in Fig 3.2,3.6 and 3.10 for respective cases.

According to Figs 3.3, 3.7 and 3.11, local customers can be divided into three groups based on their priorities: those who place a higher value on time; those who place a higher value on money; and those who place a higher value on both time and money. The opportunist, however, has two options: financial or time perspectives.

Figures 4.1, 4.2 and 4.2's in result section shows that the two systems have the same capacities for each time slot. From Figs 4.4,4.5 and 4.6, we can see the distinction between the two scenarios, where case 1 has both charging-rate and price fixed, case 3's charging rate and price rate both are different and the traditional one's charging rate stays the same but price rate varies. Figs 4.7,4.8 and 4.9 show this distinction from the viewpoint of the consumer. Fig. 4.7 shows that in case 1, the price and charging rate remain constant from the consumer's perspective. Fig. 4.8 shows that in case 2, the price rate changes for 24 time slots, but the charging rate is fixed for the consumers. Fig. 4.9 shows that in case 3, consumers have diverse charging and price rates for 24 time slots. Consumers must spend more when they approach rapid charging, while they pay less when they approach slow charging.

As seen in Fig 4.10 and 4.11, the charging station saw a surge rise in the 12 to 15 and 3 to 6 timeslots in the usual outcome, rendering the charging station unbalanced. To achieve balance, EVCS reduced some consumers in Fig 4.16 and 4.17, which resulted in the charging station handling a loss. Which we can observe from Fig 4.28 and 4.29.

On the other hand, the price and charge rate varying EVCS system didn't lose a customer; Rather, it demonstrates comparatively better charging coordination. And Figs. 4.12, 4.15, and 4.18 show that in case 3, the EVCS system didn't lose any consumers. As a result, the charging station achieved its anticipated target, which is clear in Fig. 4.30. It is evident from Table 4.1 that in cases 1 and 2, the charging station suffered a 10% reduction in revenue compared to what was anticipated for the conventional and constant methods.

The primary manipulator in the traditional EVCS system is price volatility. Consumers that prioritize their finances choose the cheapest option available, as seen in Fig 4.22. Although some buyers are being attracted by combinations.

Combination: Time * Price

since the time is set here. The combination value decreases when the price does. It increases whenever the combination value likewise increases.

The only variable in this charging rate is price, therefore combination is likewise reliant on price change (Fig 4.24). Even if the charging rate is set, some customers are nevertheless drawn to it. Consumers who value their time will receive 24 identical time slots from which they can choose one at random. Because they don't have to consider when they can charge rapidly, consumers who value their time are more likely to visit the charging station at random.

Figs4.19, 4.20, and 4.21 demonstrate that in case 1, buyers choose their time slot at random while the price and charge rate are both constant. So, in case 1 all of the consumers will select a particular slot randomly, and a large number of them may select the same slot, resulting in overload. And customers may suffer as a result. Therefore, the loss is on the charging station. Which we can see in Fig 4.16

On the other hand, it is clear from Figs 4.25, 4.26, and 4.27 shows that in case 3 variable charging time, variable prices, and variable combination all help to distribute consumers among different time slots. Here, the wide range of alternatives available at charging stations enabled buyers to choose their own time windows. Simultaneously, in case 3, the charging station ensures the predicted profit.

It shows that for an increasing electric vehicle fleet in the future, case 3 is better for EVCS system as it can manipulate consumers arrival at charging stations according to their requirements.

Although the chances are spread equally for each consumer in our simulation, which had 20 consumers, the actual situation can differ. Additionally, if we can increase the number of consumers in the simulation, the results might provide us with more realistic viewpoints. If the charging time and price can be generated randomly within a range for every time slot, The result would be considerably more realistic.

5 Conclusion

The electrification of electric vehicles has become integral to the modern generation due to its positive impact on the environment, advancements in battery technology, and government support. With zero tailpipe emissions and low operating cost. EVs offer a sustainable and convenient transportation solution that aligns with the global push for cleaner energy and a reduced carbon footprint. As we came to know, according to resources, close to 300 million EVs would be on the road within 2030. In this study an attempt was given to facilitate this increasing EV fleet through charging stations while maintaining the profit figure. Regarding the research question, the second question was whether this future-added, huge EV fleet could be served by the conventional method. After observing the results, it is evident that for certain limits, conventional methods can be used, as price is the only key here that can persuade consumers to choose them, and we saw conventional methods face an overloading situation that cost the charging station a loss. If majority of the consumer prioritize the financial perspective than conventional method would be effective. As we saw in the conventional method outcome, price plays a crucial role in distributing consumers. If more consumers choose price regarding case 2, then the charging station can control their choice through varying prices in different slots.

Our last question was whether the constant price and charge-rate method could be a better solution compared to others. After seeing the results and the early discussion, it won't be a great option, as consumers will choose a time slot randomly. If a large number of consumers choose a same slot, then the charging station for that time slot would be overloaded and system will become imbalanced, and charging stations should face a financial loss. Simultaneously, vendor-consumer relations, which are very important in business, would be spoiled. Because of the random choice of consumers, it remains uncertain how often it can happen. Now that we've addressed the first query, according to the results, case 3 can be a better response for the hypothetical future situation. From the outcomes, it is visible that in case 3, charging time, pricing, and combination are all influencing consumers in choosing their time slots. And assisting in spreading them among various time slots, which prevents the charging station from becoming overloaded and, at the same time, enables the charging station to turn a profit.

6 Recommendations

In this study, we tried to create more charging options for the consumer regarding charging stations. In the future, we suggest this work be done from the consumer's perspective. The work will create more charging options for consumers, comprising diverse options that can include home charging or any other charging option. A storage system can be introduced in EVCS where EVCS can produce energy from renewable resources and store it. In an emergency situation, EVCS can use its stored energy for consumer convenience.

7 List of References

[1] Statistica Transportation emissions worldwide - statistics & facts. Accessed: 18/05/2023. [Online].Available: https://www.statista.com/topics/7476/transportation-emissions-worldwide/#topicOverview

[2] Sortomme, E. and El-Sharkawi, M.A., 2010. Optimal charging strategies for unidirectional vehicle-to-grid. *IEEE Transactions on Smart Grid*, *2*(1), pp.131-138.

3] 8billiontrees How Many Electric Cars in the World? Accessed : 18/05/2023. [Online] . Available :https://8billiontrees.com/carbon-offsets-credits/cars/how-many-electric-cars-in-the-world/

4] Internation Energy Agency report Electric car sales took off across major car markets in 2021. Accessed: 18/05/2023 . [Online] . Available : https://www.iea.org/reports/electric-vehicles

[5] Liu, J., Lin, G., Huang, S., Zhou, Y., Li, Y. and Rehtanz, C., 2020. Optimal EV charging scheduling by considering the limited number of chargers. *IEEE Transactions on Transportation Electrification*, *7*(3), pp.1112-1122.

[6] Gan, L., Topcu, U. and Low, S.H., 2012. Optimal decentralized protocol for electric vehicle charging. *IEEE Transactions on Power Systems*, *28*(2), pp.940-951.

[7] Gan, L., Topcu, U. and Low, S.H., 2012, July. Stochastic distributed protocol for electric vehicle charging with discrete charging rate. In *2012 IEEE Power and Energy Society General Meeting* (pp. 1-8). IEEE.

[8] Li, Q., Cui, T., Negi, R., Franchetti, F. and Ilic, M.D., 2011. On-line decentralized charging of plug-in electric vehicles in power systems. *arXiv preprint arXiv:1106.5063*.

[9] Binetti, G., Davoudi, A., Naso, D., Turchiano, B. and Lewis, F.L., 2015. Scalable real-time electric vehicles charging with discrete charging rates. *IEEE Transactions on Smart Grid*, *6*(5), pp.2211-2220.

[10] Kisacikoglu, M.C., Erden, F. and Erdogan, N., 2017. Distributed control of PEV charging based on energy demand forecast. *IEEE Transactions on Industrial Informatics*, *14*(1), pp.332-341.

[11] Mao, D., Tan, J. and Wang, J., 2020. Location planning of PEV fast charging station: An integrated approach under traffic and power grid requirements. *IEEE Transactions on Intelligent Transportation Systems*, *22*(1), pp.483-492.

[12] Liu, J., Lin, G., Huang, S., Zhou, Y., Li, Y. and Rehtanz, C., 2020. Optimal EV charging scheduling by considering the limited number of chargers. *IEEE Transactions on Transportation Electrification*, *7*(3), pp.1112-1122.

[13] Nimalsiri, N.I., Mediwaththe, C.P., Ratnam, E.L., Shaw, M., Smith, D.B. and Halgamuge, S.K., 2019. A survey of algorithms for distributed charging control of electric vehicles in smart grid. *IEEE Transactions on Intelligent Transportation Systems*, *21*(11), pp.4497-4515.

[14] Car and Driver EV vs Gas: Which Cars Are Cheaper to Own? Accessed: 18/05/2023 . [Online] . Available :https://www.caranddriver.com/shopping-advice/a32494027/ev-vs-gas-cheaper-to-own/

[15] Business Standard what is range anxiety and how does it affect ev buyers. Accessed: 18/05/2023 . [Online] . Available:https://www.business-standard.com/podcast/automobile/what-is-range-anxiety-and-how-does-it-affect-ev-buyers-122051600045_1.html

[16] Allaboutcircuits technical article The Four EV charging Modes in the IEC 61851 Standard? Accessed: 18/05/2023 . [Online] . Available:https://www.allaboutcircuits.com/technical-articles/four-ev-charging-modes-iec61851-standard/

[17] P.K. Bhattacharya, Prabir Burman,1 - Probability Theory,Editor(s): P.K. Bhattacharya, Prabir Burman,Theory and Methods of Statistics,Academic Press,2016,Pages 1-24,ISBN 9780128024409,

[18] Harper, W.L., Pearce, G.A. and Stalnaker, R. eds., 2012. *Ifs: Conditionals, belief, decision, chance and time* (Vol. 15). Springer Science & Business Media.

8 Appendices

In this study We discussed three cases. For optimizing each of the cases we prepared three seperate codes. Each code comprise of 30,000 lines, as per page limitation we are unable share the whole code. Here we are sharing some important parts of the code. Every single case contains close to 30,000 lines . Sharing the whole case won't be possible,

This is thye first part of the code 1st part of code As explained in the methodology this part is similar in every code

```
import numpy
import matplotlib.pyplot as plt
import random
x=20
expected=0
t0= "0"
b0t0c=0
t1 = "1-2"
b1t1=0
b1t1c=210
t2 = "2-3"
b2t2=0
b2t2c=250
t3 = "3-4"
b3t3=0
b3t3c=230
t4 = "4-5"
b4t4=0
b4t4c=220
t5 = "5-6"
b5t5=0
b5t5c=210
t6 = "6-7"
b6t6=0
b6t6c=220
t7 = "7-8"
b7t7=0
b7t7c=240
t8 = "8-9"
b8t8=0
b8t8c=210
t9 = "9-10"
b9t9=0
b9t9c=220
t10 = "10-11"
b10t10=0
b10t10c=230
t11 = "11-12"
```

```
b11t11=0
b11t11c=220
t12 = "12-13"
b12t12=0
b12t12c=240
t13 = "13-14"
b13t13=0
b13t13c=210
t14 = "14-15"
b14t14=0
b14t14c=220
t15 = "15-16"
b15t15=0
b15t15c=210
t16 = "16-17"
b16t16=0
b16t16c=220
t17 = "17-18"
b17t17=0
b17t17c=220
t18 = "18-19"
b18t18=0
b18t18c=240
t19 = "19-20"
b19t19=0
b19t19c=220
t20 = "20-21"
b20t20=0
b20t20c=210
t21 = "21-22"
b21t21=0
b21t21c=220
t22 = "22-23"
b22t22=0
b22t22c=240
t23 = "23-24"
b23t23=0
b23t23c=210
t24 = "24-1"
b24t24=0
b24t24c=220
This part is for case 1 where charging rate and price both are fixed
# per unit prices for different time slot
# Charger in KW
j=25
pt0= 0
ct0=0
pt1 = 8
```

```
ct1 = j
pt2 = 8
ct2 = j
pt3 = 8
ct3 = j
pt4 = 8
ct4 = j
pt5 = 8
ct5 = j
pt6 = 8
ct6 = j
pt7 = 8
ct7 = j
pt8 = 8
ct8 = j
pt9 = 8
ct9 = j
pt10 =8
ct10 = j
pt11 = 8
ct11 = j
pt12 = 8
ct12 = j
pt13 = 8
ct13 = j
pt14 = 8
ct14 = j
pt15 = 8
ct15 = j
pt16 = 8
ct16 = j
pt17 = 8
ct17 = j
pt18 = 8
ct18 = j
pt19 = 8
ct19 = j
pt20 = 8
ct20 = j
pt21 = 8
ct21 = j
pt22 = 8
ct22 = j
pt23 = 8
ct23 = j
pt24 = 8
ct24 = j
pt25 = 0
```

```
This part is for case 2 where charging rate is fixed but price rate is changing
# per unit prices for different time slot
# Charger in KW
j=25
pt0= 0
ct0=0
pt1 = 12
ct1 = j
pt2 = 12
ct2 = j
pt3 = 12
ct3 = j
pt4 = 5
ct4 = j
pt5 = 5
ct5 = j
pt6 = 12
ct6 = j
pt7 = 12
ct7 = j
pt8 = 12
ct8 = j
pt9 = 5
ct9 = j
pt10 = 5
ct10 = j
pt11 = 5
ct11 = j
pt12 = 8
ct12 = j
pt13 = 8
ct13 = j
pt14 = 8
ct14 = j
pt15 = 12
ct15 = j
pt16 = 12
ct16 = j
pt17 = 8
ct17 = j
pt18 = 8
ct18 = j
pt19 = 5
ct19 = j
pt20 = 5
ct20 = j
pt21 = 5
ct21 = j
```

```
pt22 = 12
ct22 = j
pt23 = 12
ct23 = j
pt24 = 12
ct24 = j
pt
This part is for case 3 where charging time and price both are charging
# per unit prices for different time slot
# Charger in KW
pt1 = 12
ct1 = 40
pt2 = 12
ct2 = 40
pt3 = 12
ct3 = 40
pt4 = 5
ct4 = 12
pt5 = 5
ct5 = 12
pt6 = 12
ct6 = 40
pt7 = 12
ct7 = 40
pt8 = 12
ct8 = 40
pt9 = 5
ct9 = 12
pt10 = 5
ct10 = 12
pt11 = 5
ct11 = 12
pt12 = 8
ct12 = 30
pt13 = 8
ct13 = 30
pt14 = 8
ct14 = 30
pt15 = 12
ct15 = 40
pt16 = 12
ct16 = 40
pt17 = 8
ct17 = 30
pt18 = 8
ct18 = 30
pt19 = 5
ct19 = 12
```

```
pt20 = 5
ct20 = 12
pt21 = 5
ct21 = 12
pt22 = 12
ct22 = 40
pt23 = 12
ct23 = 40
pt24 = 12
ct24 = 40
```

As we have page limitations we showed the code for one consumer from here all the have similar conditions
In this part consumer choose their priorities , produce there soc and based on their battery size and soc they choose charging time , price and combination and store them

```
# Consumer1
rand_num = random.randint(1,100)
if rand_num <= 34:
    consumer_1 = 1
elif rand_num <= 66:
    consumer_1 = 2
else:
    consumer_1 = 3
print("consumer1:", consumer_1)
socconsumer_1 = random.randrange(20, 50)
Battery_1 = 40
realBattery1 = (Battery_1*(socconsumer_1/100))
# prices for consumer1
c1price1 = realBattery1*pt1
print(c1price1)
c1price2 = realBattery1*pt2
c1price3 = realBattery1 * pt3
c1price4 = realBattery1*pt4
c1price5 = realBattery1*pt5
c1price6 = realBattery1*pt6
c1price7 = realBattery1*pt7
c1price8 = realBattery1*pt8
c1price9 = realBattery1*pt9
c1price10 = realBattery1*pt10
c1price11 = realBattery1*pt11
c1price12 = realBattery1*pt12
c1price13 = realBattery1*pt13
c1price14 = realBattery1*pt14
c1price15 = realBattery1*pt15
c1price16 = realBattery1*pt16
c1price17 = realBattery1*pt17
c1price18 = realBattery1*pt18
```

```
c1price19 = realBattery1*pt19
c1price20 = realBattery1*pt20
c1price21 = realBattery1*pt21
c1price22 = realBattery1*pt22
c1price23 = realBattery1*pt23
c1price24 = realBattery1*pt24
# choosing minimum prices
smallest = min(c1price1, c1price2, c1price3, c1price4, c1price5, c1price6, c1price7, c1price8,
c1price9, c1price10, c1price12, c1price11, c1price13, c1price14, c1price15, c1price16, c1price17,
c1price18, c1price19, c1price20, c1price21, c1price22, c1price23, c1price24)
print("The smallest number is:", smallest)
variables = [c1price1, c1price2, c1price3, c1price4, c1price5, c1price6, c1price7, c1price8, c1price9,
c1price10, c1price12, c1price11, c1price13, c1price14, c1price15, c1price16, c1price17, c1price18,
c1price19, c1price20, c1price21, c1price22, c1price23, c1price24]
smallest_value = min(variables)
smallest_variables = []
if c1price1 == smallest_value:
    smallest_variables.append('c1price1')
if c1price2 == smallest_value:
    smallest_variables.append('c1price2')
if c1price3 == smallest_value:
    smallest_variables.append('c1price3')
if c1price4 == smallest_value:
    smallest_variables.append('c1price4')
if c1price5 == smallest_value:
    smallest_variables.append('c1price5')
if c1price6 == smallest_value:
    smallest_variables.append('c1price6')
if c1price7 == smallest_value:
    smallest_variables.append('c1price7')
if c1price8 == smallest_value:
    smallest_variables.append('c1price8')
if c1price9 == smallest_value:
    smallest_variables.append('c1price9')
if c1price10 == smallest_value:
    smallest_variables.append('c1price10')
if c1price11 == smallest_value:
    smallest_variables.append('c1price11')
if c1price12 == smallest_value:
    smallest_variables.append('c1price12')
if c1price13 == smallest_value:
    smallest_variables.append('c1price13')
if c1price14 == smallest_value:
    smallest_variables.append('c1price14')
if c1price15 == smallest_value:
    smallest_variables.append('c1price15')
if c1price16 == smallest_value:
    smallest_variables.append('c1price16')
```

```
if c1price17 == smallest_value:
    smallest_variables.append('c1price17')
if c1price18 == smallest_value:
    smallest_variables.append('c1price18')
if c1price19 == smallest_value:
    smallest_variables.append('c1price19')
if c1price20 == smallest_value:
    smallest_variables.append('c1price20')
if c1price21 == smallest_value:
    smallest_variables.append('c1price21')
if c1price22 == smallest_value:
    smallest_variables.append('c1price22')
if c1price23 == smallest_value:
    smallest_variables.append('c1price23')
if c1price24 == smallest_value:
    smallest_variables.append('c1price24')
# choosing one of the minimum price
print("The smallest variables are:", smallest_variables)
chosen_variable = random.choice(smallest_variables)
print(chosen_variable)
print("The chosen variable is:", chosen_variable)
# times
c1time1 = realBattery1/ct1
print(c1time1)
c1time2 = realBattery1/ct2
c1time3 = realBattery1/ct3
c1time4 = realBattery1/ct4
c1time5 = realBattery1/ct5
c1time6 = realBattery1/ct6
c1time7 = realBattery1/ct7
c1time8 = realBattery1/ct8
c1time9 = realBattery1/ct9
c1time10 = realBattery1/ct10
c1time11 = realBattery1/ct11
c1time12 = realBattery1/ct12
c1time13 = realBattery1/ct13
c1time14 = realBattery1/ct14
c1time15 = realBattery1/ct15
c1time16 = realBattery1/ct16
c1time17 = realBattery1/ct17
c1time18 = realBattery1/ct18
c1time19 = realBattery1/ct19
c1time20 = realBattery1/ct20
c1time21 = realBattery1/ct21
c1time22 = realBattery1/ct22
c1time23 = realBattery1/ct23
c1time24 = realBattery1/ct24
# choosing minimum times required for charge
```

```
smallest = min(c1time1, c1time2, c1time3, c1time4, c1time5, c1time6, c1time7, c1time8, c1time9,
c1time10, c1time12, c1time11, c1time13, c1time14, c1time15, c1time16, c1time17, c1time18,
c1time19, c1time20, c1time21, c1time22, c1time23, c1time24)
print("The smallest number is:", smallest)
times1 = [c1time1, c1time2, c1time3, c1time4, c1time5, c1time6, c1time7, c1time8, c1time9,
c1time10, c1time12, c1time11, c1time13, c1time14, c1time15, c1time16, c1time17, c1time18,
c1time19, c1time20, c1time21, c1time22, c1time23, c1time24]
smallest_time1 = min(times1)
smallest_times1 = []
if c1time1 == smallest_time1:
   smallest_times1.append('c1time1')
if c1time2 == smallest_time1:
   smallest_times1.append('c1time2')
if c1time3 == smallest_time1:
   smallest_times1.append('c1time3')
if c1time4 == smallest_time1:
   smallest_times1.append('c1time4')
if c1time5 == smallest_time1:
   smallest_times1.append('c1time5')
if c1time6 == smallest_time1:
   smallest_times1.append('c1time6')
if c1time7 == smallest_time1:
   smallest_times1.append('c1time7')
if c1time8 == smallest_time1:
   smallest_times1.append('c1time8')
if c1time9 == smallest_time1:
   smallest_times1.append('c1time9')
if c1time10 == smallest_time1:
   smallest_times1.append('c1time10')
if c1time11 == smallest_time1:
   smallest_times1.append('c1time11')
if c1time12 == smallest_time1:
   smallest_times1.append('c1time12')
if c1time13 == smallest_time1:
   smallest_times1.append('c1time13')
if c1time14 == smallest_time1:
   smallest_times1.append('c1time14')
if c1time15 == smallest_time1:
   smallest_times1.append('c1time15')
if c1time16 == smallest_time1:
   smallest_times1.append('c1time16')
if c1time17 == smallest_time1:
   smallest_times1.append('c1time17')
if c1time18 == smallest_time1:
   smallest_times1.append('c1time18')
if c1time19 == smallest_time1:
   smallest_times1.append('c1time19')
if c1time20 == smallest_time1:
    smallest_times1.append('c1time20')
if c1time21 == smallest_time1:
    smallest_times1.append('c1time21')
if c1time22 == smallest_time1:
    smallest_times1.append('c1time22')
if c1time23 == smallest_time1:
    smallest_times1.append('c1time23')
if c1time24 == smallest_time1:
    smallest_times1.append('c1time24')
# choosing one from the minimum times
print("The smallest_times1 are:", smallest_times1)
chosen_time1 = random.choice(smallest_times1)
print("The chosen_time1 is:", chosen_time1)
# combination of price and time
c1combo1 = c1time1*c1price1
c1combo2 = c1time2*c1price2
c1combo3 = c1time3*c1price3
c1combo4 = c1time4*c1price4
c1combo5 = c1time5*c1price5
c1combo6 = c1time6*c1price6
c1combo7 = c1time7*c1price7
c1combo8 = c1time8*c1price8
c1combo9 = c1time9*c1price9
c1combo10 = c1time10*c1price10
c1combo11 = c1time11*c1price11
c1combo12 = c1time12*c1price12
c1combo13 = c1time13*c1price13
c1combo14 = c1time14*c1price14
c1combo15 = c1time15*c1price15
c1combo16 = c1time16*c1price16
c1combo17 = c1time17*c1price17
c1combo18 = c1time18*c1price18
c1combo19 = c1time19*c1price19
c1combo20 = c1time20*c1price20
c1combo21 = c1time21*c1price21
c1combo22 = c1time22*c1price22
c1combo23 = c1time23*c1price23
c1combo24 = c1time24*c1price24
smallest = min(c1combo1, c1combo2, c1combo3, c1combo4, c1combo5, c1combo6, c1combo7,
c1combo8, c1combo9, c1combo10, c1combo12, c1combo11, c1combo13, c1combo14, c1combo15,
c1combo16, c1combo17, c1combo18, c1combo19, c1combo20, c1combo21, c1combo22,
c1combo23, c1combo24)
print("The smallest number is:", smallest)
combinations1 = [c1combo1, c1combo2, c1combo3, c1combo4, c1combo5, c1combo6, c1combo7,
c1combo8, c1combo9, c1combo10, c1combo12, c1combo11, c1combo13, c1combo14, c1combo15,
c1combo16, c1combo17, c1combo18, c1combo19, c1combo20, c1combo21, c1combo22,
c1combo23, c1combo24]
smallest_combination1 = min(combinations1)
```

```
smallest_combinations1 = []
if c1combo1 == smallest_combination1:
    smallest_combinations1.append('c1combo1')
if c1combo2 == smallest_combination1:
    smallest_combinations1.append('c1combo2')
if c1combo3 == smallest_combination1:
    smallest_combinations1.append('c1combo3')
if c1combo4 == smallest_combination1:
    smallest_combinations1.append('c1combo4')
if c1combo5 == smallest_combination1:
    smallest_combinations1.append('c1combo5')
if c1combo6 == smallest_combination1:
    smallest_combinations1.append('c1combo6')
if c1combo7 == smallest_combination1:
    smallest_combinations1.append('c1combo7')
if c1combo8 == smallest_combination1:
    smallest_combinations1.append('c1combo8')
if c1combo9 == smallest_combination1:
    smallest_combinations1.append('c1combo9')
if c1combo10 == smallest_combination1:
    smallest_combinations1.append('c1combo10')
if c1combo11 == smallest_combination1:
    smallest_combinations1.append('c1combo11')
if c1combo12 == smallest_combination1:
    smallest_combinations1.append('c1combo12')
if c1combo13 == smallest_combination1:
    smallest_combinations1.append('c1combo13')
if c1combo14 == smallest_combination1:
    smallest_combinations1.append('c1combo14')
if c1combo15 == smallest_combination1:
    smallest_combinations1.append('c1combo15')
if c1combo16 == smallest_combination1:
    smallest_combinations1.append('c1combo16')
if c1combo17 == smallest_combination1:
    smallest_combinations1.append('c1combo17')
if c1combo18 == smallest_combination1:
    smallest_combinations1.append('c1combo18')
if c1combo19 == smallest_combination1:
    smallest_combinations1.append('c1combo19')
if c1combo20 == smallest_combination1:
    smallest_combinations1.append('c1combo20')
if c1combo21 == smallest_combination1:
    smallest_combinations1.append('c1combo21')
if c1combo22 == smallest_combination1:
    smallest_combinations1.append('c1combo22')
if c1combo23 == smallest_combination1:
    smallest_combinations1.append('c1combo23')
if c1combo24 == smallest_combination1:
    smallest_combinations1.append('c1combo24')
# choosing one from the minimum times
print("The smallest_combinations1 are:", smallest_combinations1)
chosen_combination1 = random.choice(smallest_combinations1)
print("The chosen_combination1 is:", chosen_combination1)
if consumer_1==1:
    if chosen_time1 == 'c1time1':
        expected += c1price1
    if chosen_time1 == 'c1time2':
        expected += c1price2
    if chosen_time1 == 'c1time3':
        expected += c1price3
    if chosen_time1 == 'c1time4':
        expected += c1price4
    if chosen_time1 == 'c1time5':
        expected += c1price5
    if chosen_time1 == 'c1time6':
        expected += c1price6
    if chosen_time1 == 'c1time7':
        expected += c1price7
    if chosen_time1 == 'c1time8':
        expected += c1price8
    if chosen_time1 == 'c1time9':
        expected += c1price9
    if chosen_time1 == 'c1time10':
        expected += c1price10
    if chosen_time1 == 'c1time11':
        expected += c1price11
    if chosen_time1 == 'c1time12':
        expected += c1price12
    if chosen_time1 == 'c1time13':
        expected += c1price13
    if chosen_time1 == 'c1time14':
        expected += c1price14
    if chosen_time1 == 'c1time15':
        expected += c1price15
    if chosen_time1 == 'c1time16':
        expected += c1price16
    if chosen_time1 == 'c1time17':
        expected += c1price17
    if chosen_time1 == 'c1time18':
        expected += c1price18
    if chosen_time1 == 'c1time19':
        expected += c1price19
    if chosen_time1 == 'c1time20':
        expected += c1price20
    if chosen_time1 == 'c1time21':
        expected += c1price21
    if chosen_time1 == 'c1time22':
        expected += c1price22
    if chosen_time1 == 'c1time23':
        expected += c1price23
    if chosen_time1 == 'c1time24':
        expected += c1price24
if consumer_1==2:
    if chosen_variable == 'c1price1':
        expected += c1price1
    if chosen_variable == 'c1price2':
        expected += c1price2
    if chosen_variable == 'c1price3':
        expected += c1price3
    if chosen_variable == 'c1price4':
        expected += c1price4
    if chosen_variable == 'c1price5':
        expected += c1price5
    if chosen_variable == 'c1price6':
        expected += c1price6
    if chosen_variable == 'c1price7':
        expected += c1price7
    if chosen_variable == 'c1price8':
        expected += c1price8
    if chosen_variable == 'c1price9':
        expected += c1price9
    if chosen_variable == 'c1price10':
        expected += c1price10
    if chosen_variable == 'c1price11':
        expected += c1price11
    if chosen_variable == 'c1price12':
        expected += c1price12
    if chosen_variable == 'c1price13':
        expected += c1price13
    if chosen_variable == 'c1price14':
        expected += c1price14
    if chosen_variable == 'c1price15':
        expected += c1price15
    if chosen_variable == 'c1price16':
        expected += c1price16
    if chosen_variable == 'c1price17':
        expected += c1price17
    if chosen_variable == 'c1price18':
        expected += c1price18
    if chosen_variable == 'c1price19':
        expected += c1price19
    if chosen_variable == 'c1price20':
        expected += c1price20
    if chosen_variable == 'c1price21':
        expected += c1price21
    if chosen_variable == 'c1price22':
        expected += c1price22
    if chosen_variable == 'c1price23':
        expected += c1price23
    if chosen_variable == 'c1price24':
        expected += c1price24
if consumer_1==3:
    if chosen_combination1 == 'c1combo1':
        expected += c1price1
    if chosen_combination1 == 'c1combo2':
        expected += c1price2
    if chosen_combination1 == 'c1combo3':
        expected += c1price3
    if chosen_combination1 == 'c1combo4':
        expected += c1price4
    if chosen_combination1 == 'c1combo5':
        expected += c1price5
    if chosen_combination1 == 'c1combo6':
        expected += c1price6
    if chosen_combination1 == 'c1combo7':
        expected += c1price7
    if chosen_combination1 == 'c1combo8':
        expected += c1price8
    if chosen_combination1 == 'c1combo9':
        expected += c1price9
    if chosen_combination1 == 'c1combo10':
        expected += c1price10
    if chosen_combination1 == 'c1combo11':
        expected += c1price11
    if chosen_combination1 == 'c1combo12':
        expected += c1price12
    if chosen_combination1 == 'c1combo13':
        expected += c1price13
    if chosen_combination1 == 'c1combo14':
        expected += c1price14
    if chosen_combination1 == 'c1combo15':
        expected += c1price15
    if chosen_combination1 == 'c1combo16':
        expected += c1price16
    if chosen_combination1 == 'c1combo17':
        expected += c1price17
    if chosen_combination1 == 'c1combo18':
        expected += c1price18
    if chosen_combination1 == 'c1combo19':
        expected += c1price19
    if chosen_combination1 == 'c1combo20':
        expected += c1price20
    if chosen_combination1 == 'c1combo21':
        expected += c1price21
    if chosen_combination1 == 'c1combo22':
        expected += c1price22
    if chosen_combination1 == 'c1combo23':
        expected += c1price23
    if chosen_combination1 == 'c1combo24':
        expected += c1price24

2and part here consumers are located to different time slots as per their choice
# conditions for time
# consumer1
if consumer_1 == 1:
    if chosen_time1 == 'c1time1':
        if c1time1 < 1:
                b1t1 += Battery_1
        if 1<c1time1 < 2:
                b1t1 += Battery_1
                b2t2 += Battery_1
        if c1time1 >= 2:
                b1t1 += Battery_1
                b2t2 += Battery_1
                b3t3 += Battery_1
    if chosen_time1 == 'c1time2':
        if c1time2 < 1:
                b2t2 += Battery_1
        if 1<c1time2 < 2:
                b2t2 += Battery_1
                b3t3 += Battery_1
        if c1time2 >= 2:
                b4t4 += Battery_1
                b2t2 += Battery_1
                b3t3 += Battery_1
    if chosen_time1 == 'c1time3':
        if c1time3 <= 1:
                b3t3 += Battery_1
        if 1<c1time3 < 2:
                b4t4 += Battery_1
                b3t3 += Battery_1
        if c1time3 >= 2:
                b4t4 += Battery_1
                b5t5 += Battery_1
                b3t3 += Battery_1
    if chosen_time1 == 'c1time4':
        if c1time4 <= 1:
            b4t4 += Battery_1
        if 1<c1time4< 2:
            b4t4 += Battery_1
            b5t5 += Battery_1
        if c1time4 >= 2:
            b4t4 += Battery_1
            b5t5 += Battery_1
            b6t6 += Battery_1
    if chosen_time1 == 'c1time5':
        if c1time5 <= 1:
            b5t5 += Battery_1
        if 1<c1time5< 2:
            b6t6 += Battery_1
            b5t5 += Battery_1
        if c1time5 >= 2:
            b7t7 += Battery_1
            b5t5 += Battery_1
            b6t6 += Battery_1
    if chosen_time1 == 'c1time6':
        if c1time6 <= 1:
            b6t6 += Battery_1
        if 1<c1time6 < 2:
            b6t6 += Battery_1
            b7t7 += Battery_1
        if c1time6 >= 2:
            b7t7 += Battery_1
            b8t8 += Battery_1
            b6t6 += Battery_1
    if chosen_time1 == 'c1time7':
        if c1time7 <= 1:
            b7t7 += Battery_1
        if 1<c1time7 < 2:
            b8t8 += Battery_1
            b7t7 += Battery_1
        if c1time7 >= 2:
            b7t7 += Battery_1
            b8t8 += Battery_1
            b9t9 += Battery_1
    if chosen_time1 == 'c1time8':
        if c1time8 <= 1:
            b8t8 += Battery_1
        if 1<c1time8< 2:
            b8t8 += Battery_1
            b9t9 += Battery_1
        if c1time8 >= 2:
            b10t10 += Battery_1
            b8t8 += Battery_1
            b9t9 += Battery_1
    if chosen_time1 == 'c1time9':
        if c1time9 <= 1:
            b9t9 += Battery_1
        if 1<c1time9 <2:
            b10t10 += Battery_1
            b9t9 += Battery_1
        if c1time9 >= 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
            b9t9 += Battery_1
    if chosen_time1 == 'c1time10':
        if c1time10 <= 1:
            b10t10 += Battery_1
        if 1<c1time10 < 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
        if c1time10 >= 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
            b12t12 += Battery_1

    if chosen_time1 == 'c1time11':
        if c1time11 <= 1:
            b11t11 += Battery_1
        if 1<c1time11 < 2:
            b12t12 += Battery_1
            b11t11 += Battery_1
        if c1time11 >= 2:
            b13t13 += Battery_1
            b11t11 += Battery_1
            b12t12 += Battery_1
    if chosen_time1 == 'c1time12':
        if c1time12 <= 1:
            b12t12 += Battery_1
        if 1<c1time12 < 2:
            b12t12 += Battery_1
            b13t13 += Battery_1
        if c1time12 >= 2:
            b13t13 += Battery_1
            b14t14 += Battery_1
            b12t12 += Battery_1
    if chosen_time1 == 'c1time13':
        if c1time13 <= 1:
            b13t13 += Battery_1
        if 1< c1time13 < 2:
            b14t14 += Battery_1
            b13t13 += Battery_1
        if c1time13 >= 2:
            b13t13 += Battery_1
            b14t14 += Battery_1
            b15t15 += Battery_1
```

```
if chosen_time1 == 'c1time14':
    if c1time14 <= 1:
        b14t14 += Battery_1
    if 1<c1time14 < 2:
        b14t14 += Battery_1
        b15t15 += Battery_1
    if c1time14 >= 2:
        b16t16 += Battery_1
        b14t14 += Battery_1
        b15t15 += Battery_1
if chosen_time1 == 'c1time15':
    if c1time15 <= 1:
        b15t15 += Battery_1
    if 1< c1time15 < 2:
        b16t16 += Battery_1
        b15t15 += Battery_1
    if c1time15 >= 2:
        b16t16 += Battery_1
        b17t17 += Battery_1
        b15t15 += Battery_1
if chosen_time1 == 'c1time16':
    if c1time16 <= 1:
        b16t16 += Battery_1
    if 1<c1time16 < 2:
        b16t16 += Battery_1
        b17t17 += Battery_1
    if c1time16 >= 2:
        b16t16 += Battery_1
        b17t17 += Battery_1
        b18t18 += Battery_1
if chosen_time1 == 'c1time17':
    if c1time17 <= 1:
        b17t17 += Battery_1
    if 1<c1time17 < 2:
        b18t18 += Battery_1
        b17t17 += Battery_1
    if c1time17 >= 2:
        b19t19 += Battery_1
        b17t17 += Battery_1
        b18t18 += Battery_1
if chosen_time1 == 'c1time18':
    if c1time18 <= 1:
        b18t18 += Battery_1
    if 1<c1time18 < 2:
        b18t18 += Battery_1
        b19t19 += Battery_1
    if c1time18 >= 2:
        b19t19 += Battery_1
```

```
            b20t20 += Battery_1
            b18t18 += Battery_1
    if chosen_time1 == 'c1time19':
        if c1time19 <= 1:
            b19t19 += Battery_1
        if 1<c1time19 < 2:
            b20t20 += Battery_1
            b19t19 += Battery_1
        if c1time19 >= 2:
            b19t19 += Battery_1
            b20t20 += Battery_1
            b21t21 += Battery_1
    if chosen_time1 == 'c1time20':
        if c1time20 <= 1:
            b20t20 += Battery_1
        if 1<c1time20 < 2:
            b20t20 += Battery_1
            b21t21 += Battery_1
        if c1time20 >= 2:
            b22t22 += Battery_1
            b20t20 += Battery_1
            b21t21 += Battery_1
    if chosen_time1 == 'c1time21':
        if c1time21 <= 1:
            b21t21 += Battery_1
        if 1<c1time21 < 2:
            b22t22 += Battery_1
            b21t21 += Battery_1
        if c1time21 >= 2:
            b22t22 += Battery_1
            b23t23 += Battery_1
            b21t21 += Battery_1
    if chosen_time1 == 'c1time22':
        if c1time22 <= 1:
            b22t22 += Battery_1
        if 1<c1time22 < 2:
            b22t22 += Battery_1
            b23t23 += Battery_1
        if c1time22 >= 2:
            b22t22 += Battery_1
            b23t23 += Battery_1
            b24t24 += Battery_1
    if chosen_time1 == 'c1time23':
        if c1time23 <= 1:
            b23t23 += Battery_1
        if 1<c1time23 <2:
            b24t24 += Battery_1
            b23t23 += Battery_1
        if c1time23 >= 2:
            b23t23 += Battery_1
            b24t24 += Battery_1
    if chosen_time1 == 'c1time24':
        if c1time24 <= 1:
            b24t24 += Battery_1
        if 1<c1time24 < 2:
            b24t24 += Battery_1
        if c1time24 >= 2:
            b24t24 += Battery_1
# conditions for price
# consumer1
if consumer_1 == 2:
    if chosen_variable == 'c1price1':
        if c1time1 <= 1:
                b1t1 += Battery_1
        if 1 < c1time1 < 2:
                b1t1 += Battery_1
                b2t2 += Battery_1
        if c1time1 >= 2:
                b1t1 += Battery_1
                b2t2 += Battery_1
                b3t3 += Battery_1
    if chosen_variable == 'c1price2':
        if c1time2 <= 1:
                b2t2 += Battery_1
        if 1 < c1time2 < 2:
                b2t2 += Battery_1
                b3t3 += Battery_1
        if  c1time2 >= 2:
                b4t4 += Battery_1
                b2t2 += Battery_1
                b3t3 += Battery_1
    if chosen_variable == 'c1price3':
        if c1time3 <= 1:
                b3t3 += Battery_1
        if 1 < c1time3 < 2:
                b4t4 += Battery_1
                b3t3 += Battery_1
        if c1time3 >= 2:
                b4t4 += Battery_1
                b5t5 += Battery_1
                b3t3 += Battery_1
    if chosen_variable == 'c1price4':
        if c1time4 <= 1:
            b4t4 += Battery_1
        if 1 < c1time4 < 2:
            b4t4 += Battery_1
            b5t5 += Battery_1
        if c1time4 >= 2:
            b4t4 += Battery_1
            b5t5 += Battery_1
            b6t6 += Battery_1
    if chosen_variable == 'c1price5':
        if c1time5 <= 1:
            b5t5 += Battery_1
        if 1 < c1time5 < 2:
            b6t6 += Battery_1
            b5t5 += Battery_1
        if c1time5 >= 3:
            b7t7 += Battery_1
            b5t5 += Battery_1
            b6t6 += Battery_1
    if chosen_variable == 'c1price6':
        if c1time6 <= 1:
            b6t6 += Battery_1
        if 1 < c1time6 < 2:
            b6t6 += Battery_1
            b7t7 += Battery_1
        if c1time6 >= 3:
            b7t7 += Battery_1
            b8t8 += Battery_1
            b6t6 += Battery_1
    if chosen_variable == 'c1price7':
        if c1time7 <= 1:
            b7t7 += Battery_1
        if 1 < c1time7 < 2:
            b8t8 += Battery_1
            b7t7 += Battery_1
        if c1time7 >= 3:
            b7t7 += Battery_1
            b8t8 += Battery_1
            b9t9 += Battery_1
    if chosen_variable == 'c1price8':
        if c1time8 <= 1:
            b8t8 += Battery_1
        if 1 < c1time8 < 2:
            b8t8 += Battery_1
            b9t9 += Battery_1
        if c1time8 >= 2:
            b10t10 += Battery_1
            b8t8 += Battery_1
            b9t9 += Battery_1
    if chosen_variable == 'c1price9':
        if c1time9 <= 1:
            b9t9 += Battery_1
        if 1 < c1time9 < 2:
            b10t10 += Battery_1
            b9t9 += Battery_1
        if c1time9 >= 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
            b9t9 += Battery_1
    if chosen_variable == 'c1price10':
        if c1time10 < 1:
            b10t10 += Battery_1
        if 1 < c1time10 < 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
        if c1time10 >= 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
            b12t12 += Battery_1
    if chosen_variable == 'c1price11':
        if c1time11 <= 1:
            b11t11 += Battery_1
        if 1 < c1time11 < 2:
            b12t12 += Battery_1
            b11t11 += Battery_1
        if c1time11 >= 2:
            b13t13 += Battery_1
            b11t11 += Battery_1
            b12t12 += Battery_1
    if chosen_variable == 'c1price12':
        if c1time12 <= 1:
            b12t12 += Battery_1
        if 1 < c1time12 < 2:
            b12t12 += Battery_1
            b13t13 += Battery_1
        if c1time12 >= 2:
            b13t13 += Battery_1
            b14t14 += Battery_1
            b12t12 += Battery_1
    if chosen_variable == 'c1price13':
        if c1time13 <= 1:
            b13t13 += Battery_1
        if 1 < c1time13 < 2:
            b14t14 += Battery_1
            b13t13 += Battery_1
        if c1time13 >= 2:
            b13t13 += Battery_1
            b14t14 += Battery_1
            b15t15 += Battery_1
    if chosen_variable == 'c1price14':
        if c1time14 <= 1:
            b14t14 += Battery_1
        if 1 < c1time14 < 2:
            b14t14 += Battery_1
            b15t15 += Battery_1
        if c1time14 >= 3:
            b16t16 += Battery_1
            b14t14 += Battery_1
            b15t15 += Battery_1
    if chosen_variable == 'c1price15':
        if c1time15 <= 1:
            b15t15 += Battery_1
        if 1 < c1time15 < 2:
            b16t16 += Battery_1
            b15t15 += Battery_1
        if c1time15 >= 2:
            b16t16 += Battery_1
            b17t17 += Battery_1
            b15t15 += Battery_1
    if chosen_variable == 'c1price16':
        if c1time16 <= 1:
            b16t16 += Battery_1
        if 1 < c1time16 < 2:
            b16t16 += Battery_1
            b17t17 += Battery_1
        if c1time16 >= 2:
            b16t16 += Battery_1
            b17t17 += Battery_1
            b18t18 += Battery_1
    if chosen_variable == 'c1price17':
        if c1time17 <= 1:
            b17t17 += Battery_1
        if 1 < c1time17 < 2:
            b18t18 += Battery_1
            b17t17 += Battery_1
        if c1time17 >= 2:
            b19t19 += Battery_1
            b17t17 += Battery_1
            b18t18 += Battery_1
    if chosen_variable == 'c1price18':
        if c1time18 <= 1:
            b18t18 += Battery_1
        if 1 < c1time18 < 2:
            b18t18 += Battery_1
            b19t19 += Battery_1
        if c1time18 >= 2:
            b19t19 += Battery_1
            b20t20 += Battery_1
            b18t18 += Battery_1
    if chosen_variable == 'c1price19':
        if c1time19 <= 1:
            b19t19 += Battery_1
        if 1 < c1time19 < 2:
            b20t20 += Battery_1
            b19t19 += Battery_1
        if c1time19 >= 2:
            b19t19 += Battery_1
            b20t20 += Battery_1
            b21t21 += Battery_1
    if chosen_variable == 'c1price20':
        if c1time20 <= 1:
            b20t20 += Battery_1
        if 1 < c1time20 < 2:
            b20t20 += Battery_1
            b21t21 += Battery_1
        if c1time20 >= 2:
            b22t22 += Battery_1
            b20t20 += Battery_1
            b21t21 += Battery_1
    if chosen_variable == 'c1price21':
        if c1time21 <= 1:
            b21t21 += Battery_1
        if 1 < c1time21 < 2:
            b22t22 += Battery_1
            b21t21 += Battery_1
        if c1time21 >= 2:
            b22t22 += Battery_1
            b23t23 += Battery_1
            b21t21 += Battery_1
    if chosen_variable == 'c1price22':
        if c1time22 <= 1:
            b22t22 += Battery_1
        if 1 < c1time22 < 2:
            b22t22 += Battery_1
            b23t23 += Battery_1
        if c1time22 >= 2:
            b22t22 += Battery_1
            b23t23 += Battery_1
            b24t24 += Battery_1
    if chosen_variable == 'c1price23':
        if c1time23 <= 1:
            b23t23 += Battery_1
        if 1 < c1time23 < 2:
            b24t24 += Battery_1
            b23t23 += Battery_1
        if c1time23 >= 2:
            b23t23 += Battery_1
            b24t24 += Battery_1
    if chosen_variable == 'c1price24':
        if c1time24 <= 1:
            b24t24 += Battery_1
        if 1 < c1time24 < 2:
            b24t24 += Battery_1
        if c1time24 >= 2:
            b24t24 += Battery_1
# conditions for combination
# consumer1
if consumer_1 == 3:
    if chosen_combination1 == 'c1combo1':
        if c1time1 <= 1:
                b1t1 += Battery_1
        if 1 < c1time1 < 2:
                b1t1 += Battery_1
                b2t2 += Battery_1
        if c1time1 >= 2:
                b1t1 += Battery_1
                b2t2 += Battery_1
                b3t3 += Battery_1
    if chosen_combination1 == 'c1combo2':
        if c1time2 <= 1:
                b2t2 += Battery_1
        if 1 < c1time2 < 2:
                b2t2 += Battery_1
                b3t3 += Battery_1
        if  c1time2 >= 2:
                b4t4 += Battery_1
                b2t2 += Battery_1
                b3t3 += Battery_1
    if chosen_combination1 == 'c1combo3':
        if c1time3 <= 1:
                b3t3 += Battery_1
        if 1 < c1time3 < 2:
                b4t4 += Battery_1
                b3t3 += Battery_1
        if c1time3 >= 2:
                b4t4 += Battery_1
                b5t5 += Battery_1
                b3t3 += Battery_1
    if chosen_combination1 == 'c1combo4':
        if c1time4 <= 1:
            b4t4 += Battery_1
        if 1 < c1time4 < 2:
            b4t4 += Battery_1
            b5t5 += Battery_1
        if c1time4 >= 2:
            b4t4 += Battery_1
            b5t5 += Battery_1
            b6t6 += Battery_1
    if chosen_combination1 == 'c1combo5':
        if c1time5 <= 1:
            b5t5 += Battery_1
        if 1 < c1time5 < 2:
            b6t6 += Battery_1
            b5t5 += Battery_1
        if c1time5 >= 3:
            b7t7 += Battery_1
            b5t5 += Battery_1
            b6t6 += Battery_1
    if chosen_combination1 == 'c1combo6':
        if c1time6 <= 1:
            b6t6 += Battery_1
        if 1 < c1time6 < 2:
            b6t6 += Battery_1
            b7t7 += Battery_1
        if c1time6 >= 3:
            b7t7 += Battery_1
            b8t8 += Battery_1
            b6t6 += Battery_1
    if chosen_combination1 == 'c1combo7':
        if c1time7 <= 1:
            b7t7 += Battery_1
        if 1 < c1time7 < 2:
            b8t8 += Battery_1
            b7t7 += Battery_1
        if c1time7 >= 3:
            b7t7 += Battery_1
            b8t8 += Battery_1
            b9t9 += Battery_1
    if chosen_combination1 == 'c1combo8':
        if c1time8 <= 1:
            b8t8 += Battery_1
        if 1 < c1time8 < 2:
            b8t8 += Battery_1
            b9t9 += Battery_1
        if c1time8 >= 2:
            b10t10 += Battery_1
            b8t8 += Battery_1
            b9t9 += Battery_1
    if chosen_combination1 == 'c1combo9':
        if c1time9 <= 1:
            b9t9 += Battery_1
        if 1 < c1time9 < 2:
            b10t10 += Battery_1
            b9t9 += Battery_1
        if c1time9 >= 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
            b9t9 += Battery_1
    if chosen_combination1 == 'c1combo10':
        if c1time10 < 1:
            b10t10 += Battery_1
        if 1 < c1time10 < 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
        if c1time10 >= 2:
            b10t10 += Battery_1
            b11t11 += Battery_1
            b12t12 += Battery_1
    if chosen_combination1 == 'c1combo11':
        if c1time11 <= 1:
            b11t11 += Battery_1
        if 1 < c1time11 < 2:
            b12t12 += Battery_1
            b11t11 += Battery_1
        if c1time11 >= 2:
            b13t13 += Battery_1
            b11t11 += Battery_1
            b12t12 += Battery_1
    if chosen_combination1 == 'c1combo12':
        if c1time12 <= 1:
            b12t12 += Battery_1
        if 1 < c1time12 < 2:
            b12t12 += Battery_1
            b13t13 += Battery_1
        if c1time12 >= 2:
            b13t13 += Battery_1
            b14t14 += Battery_1
            b12t12 += Battery_1
    if chosen_combination1 == 'c1combo13':
        if c1time13 <= 1:
            b13t13 += Battery_1
        if 1 < c1time13 < 2:
            b14t14 += Battery_1
            b13t13 += Battery_1
        if c1time13 >= 2:
            b13t13 += Battery_1
            b14t14 += Battery_1
            b15t15 += Battery_1
    if chosen_combination1 == 'c1combo14':
        if c1time14 <= 1:
            b14t14 += Battery_1
        if 1 < c1time14 < 2:
            b14t14 += Battery_1
            b15t15 += Battery_1
        if c1time14 >= 3:
            b16t16 += Battery_1
            b14t14 += Battery_1
            b15t15 += Battery_1
    if chosen_combination1 == 'c1combo15':
        if c1time15 <= 1:
            b15t15 += Battery_1
        if 1 < c1time15 < 2:
            b16t16 += Battery_1
            b15t15 += Battery_1
        if c1time15 >= 2:
            b16t16 += Battery_1
            b17t17 += Battery_1
            b15t15 += Battery_1
    if chosen_combination1 == 'c1combo16':
        if c1time16 <= 1:
            b16t16 += Battery_1
        if 1 < c1time16 < 2:
            b16t16 += Battery_1
            b17t17 += Battery_1
        if c1time16 >= 2:
            b16t16 += Battery_1
            b17t17 += Battery_1
            b18t18 += Battery_1
    if chosen_combination1 == 'c1combo17':
        if c1time17 <= 1:
            b17t17 += Battery_1
        if 1 < c1time17 < 2:
            b18t18 += Battery_1
            b17t17 += Battery_1
        if c1time17 >= 2:
            b19t19 += Battery_1
            b17t17 += Battery_1
            b18t18 += Battery_1
    if chosen_combination1 == 'c1combo18':
        if c1time18 <= 1:
            b18t18 += Battery_1
        if 1 < c1time18 < 2:
            b18t18 += Battery_1
            b19t19 += Battery_1
        if c1time18 >= 2:
            b19t19 += Battery_1
            b20t20 += Battery_1
            b18t18 += Battery_1
```

```
if chosen_combination1 == 'c1combo19':
    if c1time19 <= 1:
        b19t19 += Battery_1
    if 1 < c1time19 < 2:
        b20t20 += Battery_1
        b19t19 += Battery_1
    if c1time19 >= 2:
        b19t19 += Battery_1
        b20t20 += Battery_1
        b21t21 += Battery_1
if chosen_combination1 == 'c1combo20':
    if c1time20 <= 1:
        b20t20 += Battery_1
    if 1 < c1time20 < 2:
        b20t20 += Battery_1
        b21t21 += Battery_1
    if c1time20 >= 2:
        b22t22 += Battery_1
        b20t20 += Battery_1
        b21t21 += Battery_1
if chosen_combination1 == 'c1combo21':
    if c1time21 <= 1:
        b21t21 += Battery_1
    if 1 < c1time21 < 2:
        b22t22 += Battery_1
        b21t21 += Battery_1
    if c1time21 >= 2:
        b22t22 += Battery_1
        b23t23 += Battery_1
        b21t21 += Battery_1
if chosen_combination1 == 'c1combo22':
    if c1time22 <= 1:
        b22t22 += Battery_1
    if 1 < c1time22 < 2:
        b22t22 += Battery_1
        b23t23 += Battery_1
    if c1time22 >= 2:
        b22t22 += Battery_1
        b23t23 += Battery_1
        b24t24 += Battery_1
if chosen_combination1 == 'c1combo23':
    if c1time23 <= 1:
        b23t23 += Battery_1
    if 1 < c1time23 < 2:
        b24t24 += Battery_1
        b23t23 += Battery_1
    if c1time23 >= 2:
        b23t23 += Battery_1
```

```
            b24t24 += Battery_1
    if chosen_combination1 == 'c1combo24':
        if c1time24 <= 1:
            b24t24 += Battery_1
        if 1 < c1time24 < 2:
            b24t24 += Battery_1
        if c1time24 >= 2:
            b24t24 += Battery_1
```

3rd part of the where a charging station curtail the consumer if a timeslot get overloaded

```
if b1t1> b1t1c:
   print("Charging Station is overloaded")
   if (consumer_1==1 and chosen_time1 == 'c1time1') or (consumer_1==2 and
chosen_variable=='c1price1') or (consumer_1==3 and 'chosen_combination1==c1combo1'):
       if c1time1 <= 1:
           Onetcon1 = Battery_1
       if 1 < c1time1 < 2:
           Onetcon1 = Battery_1
           Twotcon1 = Battery_1
       if c1time1 >= 2:
           Onetcon1 = Battery_1
           Twotcon1 = Battery_1
           Threetcon1 = Battery_1
   if (consumer_2 == 1 and chosen_time2 == 'c2time1') or (consumer_2==2 and
chosen_price2=='c2price1') or (consumer_2==3 and chosen_combination2=='c2combo1'):
       if c2time1 <= 1:
           Onetcon2 = Battery_2
       if 1 < c2time1 < 2:
           Onetcon2 = Battery_2
           Twotcon2 = Battery_2
       if c2time1 >= 2:
           Onetcon2 = Battery_2
           Twotcon2 = Battery_2
           Threetcon2 = Battery_2
   if (consumer_3 == 1 and chosen_time3 == 'c3time1') or (consumer_3==2 and
chosen_price3=='c3price1') or (consumer_3==3 and chosen_combination3=='c3combo1'):
       if c3time1 <= 1:
           Onetcon3 = Battery_3
       if 1 < c3time1 < 2:
            Onetcon3 = Battery_3
            Twotcon3 = Battery_3
       if c3time1 >= 2:
             Onetcon3 = Battery_3
             Twotcon3 = Battery_3
             Threetcon3 = Battery_3
   if (consumer_4==1 and chosen_time4 == 'c4time1') or (consumer_4==2 and
chosen_price4=='c4price1') or (consumer_4==3 and chosen_combination4=='c4combo1'):
      if c4time1 <= 1:
         Onetcon4 = Battery_4
      if 1 < c4time1 < 2:
         Onetcon4 = Battery_4
         Twotcon4 = Battery_4
      if c4time1 >= 2:
         Onetcon4 = Battery_4
         Twotcon4 = Battery_4
         Onetcon4 = Battery_4
   if (consumer_5==1 and chosen_time5 == 'c5time1') or (consumer_5==2 and
chosen_price5=='c5price1') or (consumer_5==3 and chosen_combination5=='c5combo1'):
      if c5time1 <= 1:
         Onetcon5 = Battery_5
      if 1 < c5time1 < 2:
          Onetcon5 = Battery_5
          Twotcon5 = Battery_5
      if c5time1 >= 2:
          Onetcon5 = Battery_5
          Twotcon5 = Battery_5
          Threetcon5 = Battery_5
   if (consumer_6==1 and chosen_time6 == 'c6time1') or (consumer_6==2 and
chosen_price6=='c6price1') or (consumer_6==3 and chosen_combination6=='c6combo1'):
      if c6time1 <= 1:
         Onetcon6 = Battery_6
      if 1 < c6time1 < 2:
         Onetcon6 = Battery_6
         Twotcon6 = Battery_6
      if c6time1 >= 2:
         Onetcon6 = Battery_6
         Twotcon6 = Battery_6
         Threetcon6 = Battery_6
   if (consumer_7==1 and chosen_time7 == 'c7time1') or (consumer_7==2 and
chosen_price7=='c7price1') or (consumer_7==3 and chosen_combination7=='c7combo1'):
      if c7time1 <= 1:
         Onetcon7 = Battery_7
      if 1 < c7time1 < 2:
         Onetcon7 = Battery_7
         Twotcon7 = Battery_7
      if c7time1 >= 2:
         Onetcon7 = Battery_7
         Twotcon7 = Battery_7
         Threetcon7 = Battery_7
   if (consumer_8==1 and chosen_time8 == 'c8time1') or (consumer_8==2 and
chosen_price8=='c8price1') or (consumer_8==3 and chosen_combination8=='c8combo1'):
      if c8time1 <= 1:
         Onetcon8 = Battery_8
      if 1 < c8time1 < 2:
         Onetcon8 = Battery_8
```

```
        Twotcon8 = Battery_8
      if c8time1 >= 2:
        Onetcon8 = Battery_8
        Twotcon8 = Battery_8
        Threetcon8 = Battery_8
   if (consumer_9==1 and chosen_time9 == 'c9time1') or (consumer_9==2 and
chosen_price9=='c9price1') or (consumer_9==3 and chosen_combination9=='c9combo1'):
      if c9time1 <= 1:
        Onetcon9 = Battery_9
      if 1 < c9time1 < 2:
        Onetcon9 = Battery_9
        Twotcon9 = Battery_9
      if c9time1 >= 2:
        Onetcon9 = Battery_9
        Twotcon9 = Battery_9
        Threetcon9 = Battery_9
   if (consumer_10==1 and chosen_time10 == 'c10time1') or (consumer_10==2 and
chosen_price10=='c10price1') or (consumer_10==3 and chosen_combination10=='c10combo1'):
      if c10time1 <= 1:
        Onetcon10 = Battery_10
      if 1 < c10time1 < 2:
        Twotcon10 = Battery_10
        Onetcon10 = Battery_10
      if c10time1 >= 2:
        Onetcon10 = Battery_10
        Twotcon10 = Battery_10
        Threetcon10 = Battery_10
   if (consumer_11 == 1 and chosen_time11 == 'c11time1') or (consumer_11==2 and
chosen_price11=='c11price1') or (consumer_11==3 and chosen_combination11=='c11combo1'):
      if c11time1 <= 1:
        Onetcon11 = Battery_11
      if 1 < c11time1 < 2:
        Onetcon11 = Battery_11
        Twotcon11 = Battery_11
      if c11time1 >= 2:
        Onetcon11 = Battery_11
        Twotcon11 = Battery_11
        Threetcon11 = Battery_11
   if (consumer_12 == 1 and chosen_time12 == 'c12time1') or (consumer_12==2 and
chosen_price12=='c12price1') or (consumer_12==3 and chosen_combination12=='c12combo1'):
      if c12time1 <= 1:
        Onetcon12 = Battery_12
      if 1 < c12time1 < 2:
        Onetcon12 = Battery_12
        Twotcon2 = Battery_12
      if c12time1 >= 2:
        Onetcon12 = Battery_12
        Twotcon12 = Battery_12
        Threetcon12 = Battery_12
  if (consumer_13 == 1 and chosen_time13 == 'c13time1') or (consumer_13==2 and
chosen_price13=='c13price1') or (consumer_13==3 and chosen_combination13=='c13combo1'):
      if c13time1 <= 1:
        Onetcon13 = Battery_13
      if 1 < c13time1 < 2:
        Onetcon13 = Battery_13
        Twotcon13 = Battery_13
      if c13time1 >= 2:
        Onetcon13 = Battery_13
        Twotcon13 = Battery_13
        Threetcon3 = Battery_13
  if (consumer_14 == 1 and chosen_time14 == 'c14time1') or (consumer_14==2 and
chosen_price14=='c14price1') or (consumer_14==3 and chosen_combination14=='c14combo1'):
      if c14time1 <= 1:
        Onetcon14 = Battery_14
      if 1 < c14time1 < 2:
        Onetcon14 = Battery_14
        Twotcon14 = Battery_14
      if c14time1 >= 2:
        Onetcon14 = Battery_14
        Twotcon14 = Battery_14
        Threetcon14 = Battery_14
  if (consumer_15 == 1 and chosen_time15 == 'c15time1') or (consumer_15==2 and
chosen_price15=='c15price1' )or (consumer_15==3 and chosen_combination15=='c15combo1'):
      if c15time1 <= 1:
        Onetcon15 = Battery_15
      if 1 < c15time1 < 2:
        Onetcon15 = Battery_15
        Twotcon15 = Battery_15
      if c15time1 >= 3:
        Onetcon15 = Battery_15
        Twotcon15 = Battery_15
        Threetcon15 = Battery_15
  if (consumer_16 == 1 and chosen_price16 == 'c16price1' )or (consumer_16 == 2 and
chosen_time16== 'c16time1'):
      if c16time1 <= 1:
        Onetcon16 = Battery_16
      if 1 < c16time1 < 2:
        Onetcon16 = Battery_16
        Twotcon16 = Battery_16
      if c16time1 >= 2:
        Onetcon16 = Battery_16
        Twotcon16 = Battery_16
        Threetcon16 = Battery_16
  if (consumer_17 == 1 and chosen_price17 == 'c17price1') or (consumer_17 == 2 and
chosen_time17== 'c17time1'):
      if c17time1 <= 1:
            Onetcon17 = Battery_17
        if 1 < c17time1 < 2:
            Onetcon17 = Battery_17
            Twotcon17 = Battery_17
        if c17time1 >= 2:
            Onetcon17 = Battery_17
            Twotcon17 = Battery_17
            Threetcon17 = Battery_17
    if (consumer_18 == 1 and chosen_price18 == 'c18price1') or (consumer_18 == 2 and
chosen_time18== 'c18time1'):
        if c18time1 <= 1:
            Onetcon18 = Battery_18
        if 1 < c18time1 < 2:
            Onetcon18 = Battery_18
            Twotcon18 = Battery_18
        if c18time1 >= 2:
            Onetcon18 = Battery_18
            Twotcon18 = Battery_18
            Threetcon18 = Battery_18
    if (consumer_19 == 1 and chosen_price19 == 'c19price1') or (consumer_19 == 2 and
chosen_time19== 'c19time1'):
        if c19time1 <= 1:
            Onetcon19 = Battery_19
        if 1 < c19time1 < 2:
            Onetcon19 = Battery_19
            Twotcon19 = Battery_19
        if c19time1 >= 2:
            Onetcon19 = Battery_19
            Twotcon19 = Battery_19
            Threetcon19 = Battery_19
    if (consumer_20 == 1 and chosen_price20 == 'c20price1') or (consumer_20 == 2 and chosen_time20
== 'c20time1'):
        if c20time1 <= 1:
            Onetcon20 = Battery_20
        if 1 < c20time1 < 2:
            Onetcon20 = Battery_20
            Twotcon20 = Battery_20
        if c20time1 >= 2:
            Onetcon20 = Battery_20
            Twotcon20 = Battery_20
            Threetcon20 = Battery_20

non_zero_Onetcon_vars = [Onetcon for Onetcon in [Onetcon1, Onetcon2, Onetcon3, Onetcon4,
Onetcon5, Onetcon6, Onetcon7, Onetcon8, Onetcon9, Onetcon10, Onetcon11, Onetcon12,
Because of page limitation rest of the code cannot be shared
```