UNIVERSITY OF CALIFORNIA
RIVERSIDE

Optimal Dispatch of Fleet Electric Vehicles

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Electrical Engineering

by

Mengfei Wang

March 2017

Thesis Committee:

Dr. Hamed Mohsenian-Rad, Chairperson
Dr. Mahnoosh Alizadeh
Dr. Matthew Barth

The Thesis of Mengfei Wang is approved:

_____

_____

_____

Committee Chairperson

University of California, Riverside

To my parents for all their love and support.

ABSTRACT OF THE THESIS

Optimal Dispatch of Fleet Electric Vehicles

by

Mengfei Wang

Master of Science, Graduate Program in Electrical Engineering
University of California, Riverside, March 2017
Dr. Hamed Mohsenian-Rad, Chairperson

Electric vehicle is popular and people try to find the optimal energy strategy for pure PHEV and hybrid EV. However, the routine for personal PHEV is rather fixed between home work, and commercial places, etc. There are studies about optimal dispatch on fleet. But few of them is on optimal dispatch for PHEV. Hence what we should be looking for is a vehicle pattern that constantly travels from one place to another, and the origins and destinations may not the same. Taxi service is what we need. The purpose of this thesis is to find out the effects on travel cost introduced by autonomous and electric taxis. Specifically, I analyzed the travel pattern for 536 cabs in San Francisco, and optimized their routine when they are not occupied. The goal is to reduce energy cost while serving all passengers in time. In the meanwhile, taxi will be charged when needed during the best time window.

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

With the development of self-driving technology, autonomous vehicle attracts public attention and more opportunities to enhance the energy efficiency as well as driving comfort. More than ten high tech companies invested in autonomous vehicles, including Volvo, Google, and Tesla [1]. Survey [2] reveals that the majority of respondents had not only previously heard of autonomous or self-driving vehicles, but also had a positive initial opinion of the technology, and had high expectations about the benefits of the technology. However, the majority of respondents also expressed concerns about riding in self-driving vehicles, as they may not perform as well as actual drivers. Hence one aspect of this thesis is to prove that autonomous vehicles can benefit the public with lower energy cost while keep the same level of driving performance.

As for now, the challenges brought by autonomous vehicle and self-driving technology are sensing and navigating. As far as navigating, one has to know its departure time, departure location, expected arrival time, and arrival location, so as to calculate the trajectory. This is usually referred to as the optimal routing problem. Google Maps provides optimal routing as well as additional information with advanced technology [3]. However, how autonomous vehicles perform compared to human drivers is hard to measure, unless one can show that autonomous vehicles can satisfy the same level of requirements as human drivers do.

Electric vehicle is another hot topic where people try to find the optimal energy strategy for pure PHEV and hybrid EV. However, the routine for personal PHEV is rather fixed between home, work, and commercial places, etc [4]. It is the same reason why charging station is usually constructed at places with high population and long staying time, such as shopping malls at downtown. However, the relatively unchangeable routine for PHEV with personal usage is somehow boring in a sense that it may not make any difference by applying optimal routing strategy to achieve optimal energy consumption.

What we should be looking for is a flexible driving pattern that requires vehicles to constantly travel from one place to another, but the departure and arrival place may change from time to time. It is the definition of taxi services.

Electric vehicles have already been in considered on taxi services 0. One big advantage is that electric vehicles are more environmental friendly with less fuel consumption for plug-in hybrid electric vehicle (PHEV), and literally zero fuel consumption for pure EV. However, electric vehicles on taxi services also have two challenges. The first is when and where to go to charging stations so as to maximize the usage of the battery, and the second is that how to dispatch the cabs energy efficiently.

This thesis is to solve both optimal dispatch and optimal charging problems. It is to find out the effects on travel cost introduced by autonomous and electric vehicles on taxi services. On the one hand, optimal dispatch is to find the best match between cabs and passenger requests so that all passengers are satisfied in time with minimum energy cost. On the other hand, optimal charging is to find the best time window which has minimum

impact in responding to passenger requests. Both problems are answered in my algorithms.

After constructing the algorithms to solve PHEV problem, the next step is always to find applicable data to test the methods. However, the penetration level of EVs is still very low, so we are currently in lack of detailed PHEV data sets. One of the alternatives is to use existing data sets for conventional vehicles and test what will happen if combining them with specific PHEV features to synthesize PHEV data sets [6][7][8][9].

Specifically, I analyzed the travel pattern for 536 cabs in San Francisco, and how to optimize their routine when they are not occupied. The goal is to see how much effects it will bring in terms of energy saving.

The thesis is organized as follows. Section I is introduction for why and what it is in the thesis. Section II is literature review. In Section III, I process the GPS data for 536 cabs and generate the historical data for passenger requests. I also approximate the coordinates of cabs and divide them into geographic regions. I calculate the travel distance and traffic time on Google Map Distance Matrix API. I calculated the power consumption based on different features of four PHEV. In Section IV, I designed a real-time optimal dispatch algorithm. In Section V, I further extended that algorithm on time-series with pre-booked cabs. In Section VI, I simulate several cases and discuss the results. I conclude the thesis in Section VII.

Chapter 2. Literature Review

There are studies about using incentives to change the driving patterns of people. [10][11][12] However, they are mainly on residential side for personal usage. They do not discuss commercial use such as taxi services.

People have been thinking of taxi dispatch already. [13][14][15] proposed multi-agent system to dispatch taxi. [23] focused on minimizing total customer waiting time by concurrently dispatching multiple taxis and allowing taxis to exchange their booking assignments. A shortest time path taxi dispatch system based on real-time traffic conditions is proposed in [18]. [19][21][22] maximize cab profits by providing routing recommendations. However, none of them is about PHEVs.

PHEV is popular nowadays mainly in what impacts it might bring to the power grid. There are studies that examine the opportunities that PHEV may offer to better operating the electric grid. In fact, people the role of PHEVs in particular at the distribution level, some as a potential source of energy storage, others as a means to improve power quality and reliability. There are studies that examine the adverse impact of EV charging load on distribution feeders [18] [19] [25]. The possibility of using PEVs to discharge electricity back to the grid has been studied in vehicle-to-grid (V2G) systems [26] [27] [28] [29] [30] [31]. More recently, it has been shown that PEVs may also offer reactive power compensation, not only in a V2G mode but also during a regular charging cycle, with minimum impact on the EV battery lifetime [32] [33] [34] [35] [36]. However, none of them discuss the usage of PHEV on taxis. [37] might be the only one combining optimal

dispatch and charging for fleet of electric taxi. However, its main goal is to lower the waiting time of power recharging and thus increase the workable hours for taxi drivers. It does not minimize the total energy cost while serving passengers

As for the data set for PHEV, we lack the actual data for PHEV due to the low occupation of PHEV. An alternative way to analyze PHEV is to use the data from conventional vehicles. [7] analyzed 536 GPS-equipped fleet of conventional vehicle taxies in San Francisco, and developed a PHEV test data set by combining each trip information from the original data set with additional PHEV features and characteristics. The developed data set was then used to analyze the PHEV charging load. Similarly, [38] converted one day travel trace data of 229 conventional vehicles in Austin to PHEV data set and analyzed the impact of charging network coverage for PHEV energy consumption and energy cost. [39] utilized a sample of real world driving data from the National Household Travel Survey, and applied the trip profile and drive cycle information to simulate on-road PHEV electricity and fuel consumption. [40] converted 830 days of driving traces from Southeast Michigan into a PHEV-resembled data set, and then proposed a statistical model to for generating PHEV daily driving missions.

The thesis utilized the same data set as [7] but has a huge difference. [7] does not extract the information whether the cab is occupied or free, which is the core of my thesis. Because the main idea of my thesis is keep passenger trips unchanged, but optimized cab behavior when not occupied. Therefore it is essential for us to know when a cab is available or not.

Chapter 3. Data Processing

2.1 Passenger Request History Record

The first step of this project is to extract the passenger request records out from the raw data. The raw data contains the timestamp in UNIX format, latitude and longitude of its current location, and whether the cab is occupied for each of all 536 cabs.

I notice that the status of the cab is represented by a binary variable Fare. If it is 1, it means the cab is occupied. If it is 0, it means the cab is free now. Therefore I use differential method to calculate the difference between two consecutive fare values. For example, if the Fare series is 0, 0, 1, 1, 0, 0, the difference should be 0, 1, 0,-1, 0. Clearly, it could take values among -1, 0, and 1. If it is -1, it means the status of cab is changed from 1 to 0, which is a drop-off. If it is 0, it means en route with or without passengers (whether it is occupied or free). If it is 1, it means a pick-up. Therefore I record the time when 1 and -1 appears, and pairs them together as a passenger request.

Therefore I have to consider four conditions starting and ending with 1 or 0. If the beginning and ending of fare records are both free, this is the desired case, I do nothing. But if the beginning and ending of fare are both free, or if the beginning and ending of fare are both occupied, or if the beginning of fare is free but the ending is occupied, or if the beginning of fare is occupied but the ending is free, I have to add one element to the beginning or ending of pick-up records or drop-off records. Then I sort the $Request$ table by pick-up time. The table contains 464138 records. A portion of the table is as below.

Table 1 Passenger Request History Records

Cab ID	Pick Up Time	Pick Up Latitude	Pick Up Longitude	Pick Up Section Number	Drop Off Time	Drop Off Latitude	Drop Off Longitude	Drop Off Section Number
aldhidd	1211018404	37.78781	-122.39055	55	1211018594	37.78788	-122.39074	45
enkkand	1211018404	37.79611	-122.42011	54	1211018686	37.79988	-122.41387	44
ucgewft	1211018404	37.78577	-122.42846	54	1211019067	37.77141	-122.43037	3
omdrid	1211018405	37.7735	-122.43753	48	1211018413	37.77328	-122.43636	48
aslagni	1211018407	37.78697	-122.404	54	1211019791	37.71035	-122.468	2
egreosko	1211018407	37.7398	-122.46467	37	1211019264	37.78122	-122.40841	54
ewbglo	1211018407	37.78673	-122.45131	53	1211018599	37.78678	-122.45858	53
icagpony	1211018407	37.78559	-122.41549	34	1211018547	37.78691	-122.41921	44

Plot the occurrence of pick-up and drop-off time as below.

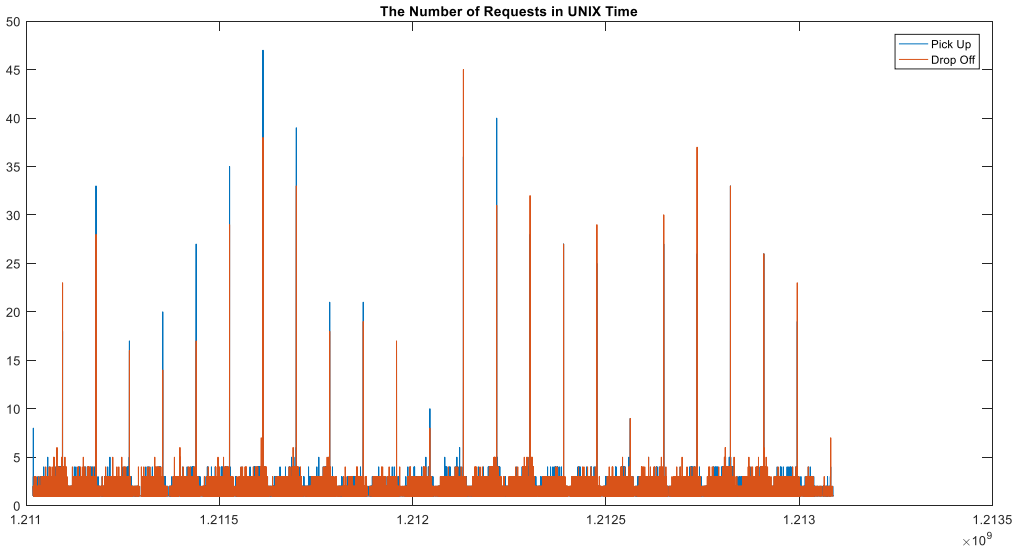

Figure 1 The Number of Requests in UNIX Time

The reason why it has so many "spikes" is because the data is incomplete. All the cabs have no time records right before every spikes. For example, there are 46 seconds missing in front of the first spikes. Therefore the spikes are actually very flatten through timeline.

2.2 Geographical Approximation

The latitude and longitude can be approximated to a small region and hence use the section number of that region to represent all the coordinates within. Since most of the coordinates are within latitudes 37.6 and 37.82 and longitudes −122.52 and −122.37. I can draw the same rectangular area as used in [7]. I have checked that among all the coordinates where pick-up and drop-off happen, there only exist 0.5283% coordinates

whose latitude is less than the minimum latitude and 0.4006% greater than the maximum latitude. There only exist 0.0308% of all the coordinates whose longitude is less than minimum longitude and 0.9780% whose longitude is greater than maximum longitude. So the boundary assumption is reasonable.

It is reasonable to relocate the pick-up location to the center of the section where the passenger is. Because in reality, passenger can wait at a cab station to be picked up.

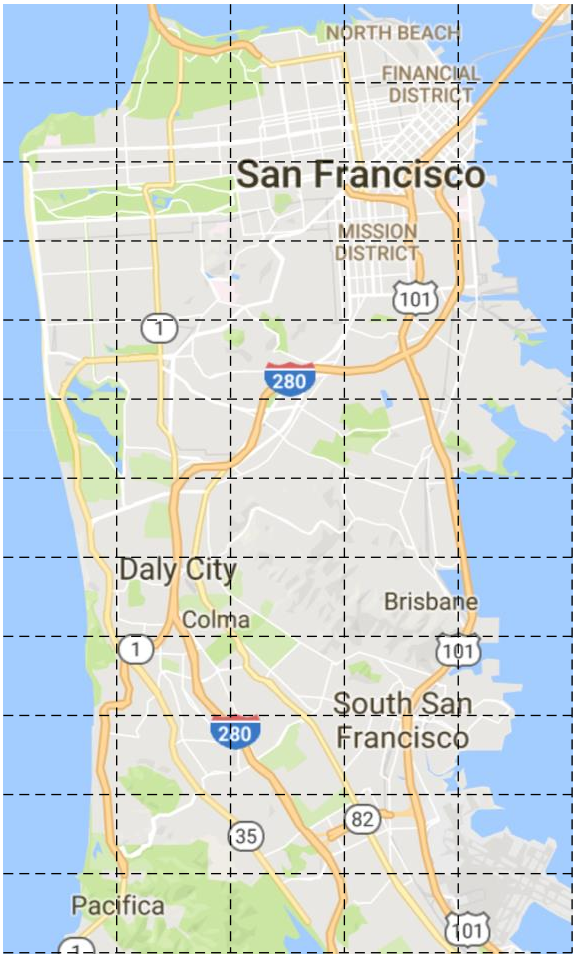

Figure 2 Dividing San Francisco into 60 Sections

[7] just wanted to know where cabs park more. And hence the paper found three locations. Interestingly, the three locations are close to San Francisco downtown, the cab headquarter and San Francisco International Airport (SFO). That is where the paper set the charging stations. I hence use the address of SFO for the first charging station. I further confirmed that the cab headquarter is Yellow Cab and I use its address for the second charging station. I choose the largest parking lot in San Francisco with 2574 parking spaces in total, which happens to be in downtown, as the third charging station. Luckily, both parking garage and SFO have PHEV chargers already. The geographical information is as below. I hence draw the occurrence of where a pick-up or drop-off is requested. The plot is as below.

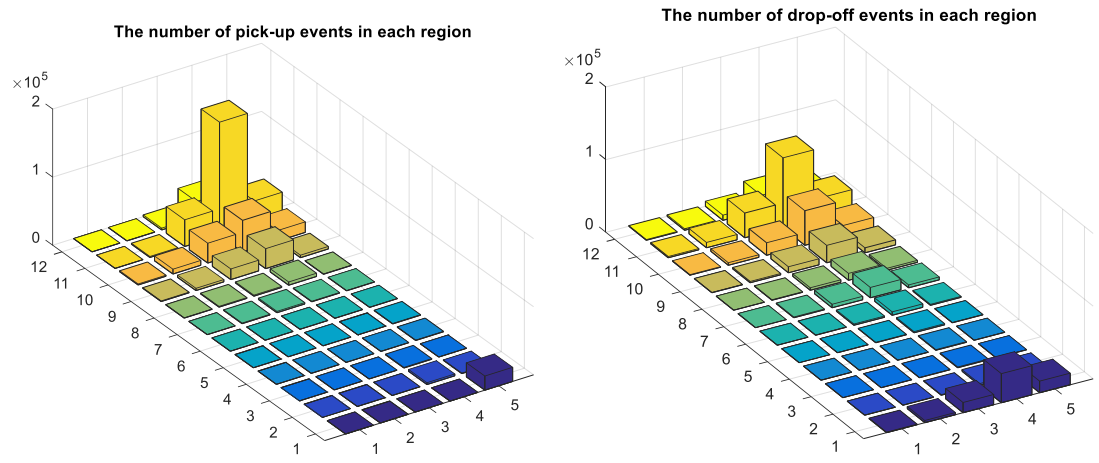

Figure 3 Occurrence of Where a Pick-up or Drop-off is Requested

Since the exact address of all three charging stations are known, I can use the exact locations of charging stations to calculate the travel distance and traffic time in Google Map API when trying to find the cost heading for a charging station, rather than roughly assuming that these charging locations are in some cells.

Optimal charging station placement is outside the scope of this thesis and so we keep the charging stations at popular locations where the cabs used to stop for long intervals of time.

Table 2 Charging Station Locations

	Charging Station #1	Charging Station #2	Charging Station #3
Name	San Francisco International Airport	Yellow Cab Co-op Taxi Headquarter	Fifth and Mission Parking Garage
Address	780 S Airport Blvd, San Francisco, CA 94128	1200 Mississippi St, San Francisco, CA 94107	833 Mission St, San Francisco, CA 94103
Coordinates	37.6213, -122.3790	37.751152, -122.394355	37.783314, -122.404563

As for the distance travelled between the successive GPS readings, such distance is approximated by a straight path between each two coordinates. Such approximation is reasonable for the purpose of our study since the reading intervals in the data set are fairly small.

I further uses the 'haversine' formula to calculate the great-circle distance between two points. That is the shortest distance over the earth's surface, ignoring any hills they fly over. If performance is an issue and accuracy less important, for small distances Pythagoras' theorem can be used on an equi-rectangular projection.

Let λ_1 and λ_2 denote the latitudes of two successive records of the GPS coordinates. Assume that φ_1 and φ_2 denote the longitudes of those coordinates. The direct distance traversed between the two points are calculated as

$$Dist_{1,2} = R\sqrt{((\varphi_2 - \varphi_1)\cos(\frac{\lambda_1 + \lambda_2}{2}))^2 + (\lambda_2 - \lambda_1)^2} \quad (1)$$

where R denotes the radius of the earth that is 6371 km.

2.3 Travel Distance and Traffic Time

After having the approximated geographical locations, I need to get the travel distance and traffic time between two locations. I use Google Map Distance Matrix API to get such information. It can "access travel distance and time for a matrix of origins and destinations with the Google Maps Distance Matrix API. The information returned is based on the recommended route between start and end points and consists of rows containing duration and distance values for each pair."[3]

Therefore I use it to create a 63×63 cell, where each entry is a 24×2 cell, which contains the distance or time between two geographical locations for 24 hourly times. In addition, Google Map Distance Matrix API can not take a past time and only provides traffic information estimated in the future. The time elapse in $Request$ table is from 05/17/2008 10:00:04 to 06/10/2008 09:17:21. Therefore I choose 05/25/2017 (Thursday) and 05/28/2017 (Saturday) as reference time for weekdays and weekends. For example,

the traffic information from charging station 1 to all the charging stations (including charging station 1 itself) is as below. The time is 05/25/2017 00:00:00, equally 1495670400 in UNIX format.

Table 3 Travel Information Between Charging Station 1 and Others

	Charging Station 1	Chrg Station 2	Chrg Station 3
Charging Station 1	[0, 0]	[17945, 1161]	[21950, 1865]

The first element in each matrix is travel distance in meters, and the second element is traffic time in seconds while choosing driving mode. For example, [17945, 1161] means it's 18 kilometers between charging station 1 and 2, and it takes 20 minutes from charging station 1 to 2 at 05/25/2017 00:00:00.

San Francisco, CA is in Pacific Time Zone (UTC-08:00). According to Google Map Distance Matrix API, "departure_time is an integer in seconds since midnight, January 1, 1970 UTC." Coordinated Universal Time (UTC) is 8 hours ahead of San Francisco, meaning that the local time in San Francisco at 05/25/2017 00:00:00 PST equals to 05/25/2017 08:00:00 UTC.

I then draw the traffic time in both weekends and weekdays as below. Red line describes weekdays, and blue line is for weekends. Clearly, red line has two surges for on-duty and off-duty rush hours at 8 am and 5 pm, respectively. On the other hand, blue line doesn't rise until 9 am because nobody gets up early on Saturday. The line reaches its peak at 12

pm, then drops down to normal around 8 pm. What's more, the traffic load on weekdays is far more than that on weekends.

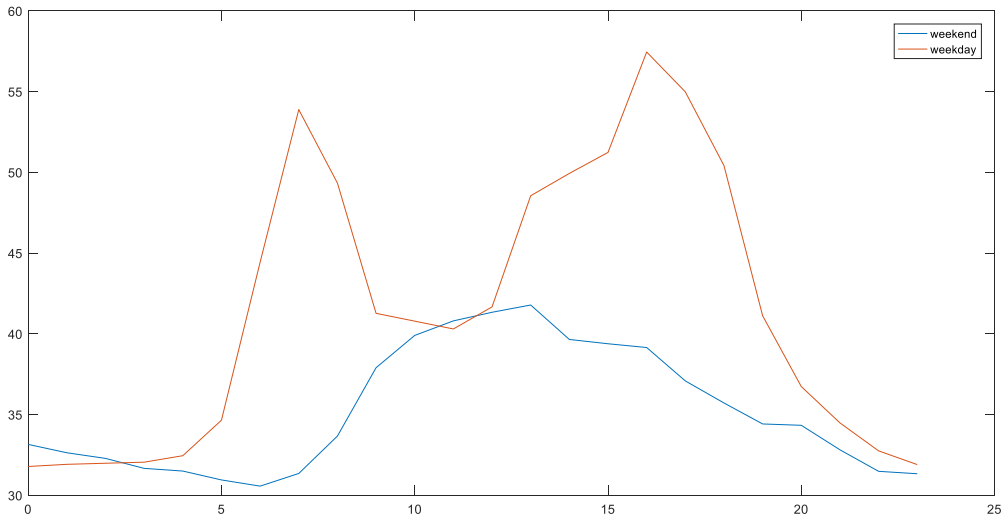

Figure 4 Traffic Time in Weekdays and Weekends

2.4 Power Consumption

I need gasoline and electricity power consumption for a typical vehicle. I found the formula in Jiarui Liu's thesis. In general, four different forces affect (impede) vehicle movement:

(1) Rolling resistance.

(2) Aerodynamic drag resistance.

(3) Inertial resistance.

(4) Grade resistance.

The timely total power requirement P_{tot} can be calculated as the sum of all resistance force multiplied by the vehicle's forward speed V. Accordingly, the energy usage E_{tot} in each second can be calculated as below.

$$P_{tot} = (F_a \cdot V + F_{air} \cdot V + F_c \cdot V + F_r \cdot V)/C_c \quad (2)$$

Where

$$F_a = m \cdot a \quad (3)$$

$$F_{air} = \frac{1}{2} \cdot \rho_{air} \cdot C_d \cdot Area \cdot V^2 \quad (4)$$

$$F_c = m \cdot g \cdot \sin\theta \quad (5)$$

$$F_r = C_r \cdot m \cdot g \quad (6)$$

Therefore

$$E_{tot} = P_{tot} \times t \quad (7)$$

Here F_a, F_{air}, and F_r denote, respectively, the force for acceleration, the aerodynamic force, a load related to steepness, and rolling resistance.

- m Vehicle weight (kg).
- a Vehicle acceleration ($\mathrm{m/s^2}$).
- V Vehicle speed ($\mathrm{m/s}$).

- ρ_{air} Air density ($\mathrm{kg/m^3}$). Normally $1.2041\ \mathrm{kg/m^3}$ for dry air at 20 °C and 101.325 kPa.
- C_d Aerodynamic drag coefficient (unitless). It is different between different vehicle model.
- $Area$ Frontal area ($\mathrm{m^2}$). Different between each vehicle model.
- g The acceleration of gravity. Normally $9.80665\ \mathrm{m/s^2}$.
- θ The slope angle (rad).
- C_r Rolling resistance coefficient (unitless). Normally equal to 0.0165.
- t Time duration (second). The time duration is 1 second for my simulation.
- C_c The energy converting efficiency ratio (unitless), representing how much energy stored in the battery can be converted into kinetic energy. It is set to be 80 % in our analysis.

Note that the road grade is not given in our available data set. In addition, I also calculate that the acceleration speed does not take much account in the energy. Hence I neglect acceleration speed and only consider the aerodynamic force and rolling resistance in my energy usage calculation.

The unit of E_{tot} is Joule. I need to convert the unit to kWh for electric energy consumption ($1\ \mathrm{kWh} = 3600000$ J).

In addition, fuel efficiency for petrol (gasoline) engine is about 20%.

Table 4 Vehicle Characteristics

Brand	Chevrolet	Honda	Ford	Toyota
Model	Volt	Accord	Fusion	Prius
Weight (kg)	1717.3	1576.2	1774.9	1451.5
Aerodynamic Drag Coefficient C_d	0.28	0.29	0.28	0.26
Frontal Area Area (m^2)	2.20	2.21	2.21	2.22
Battery Capacity (kWh)	16	6.6	7.6	4.4
Available Energy (kWh)	8.8	3.8	7.1	3.2
Average Electric Range (Mile)	37	13	21	11
Max Charging Rate (kW)	3.5	6.6	3.5	3.5
Electricity Consumption (kWh/ 100 Miles)	36	29	34	29
Gas Consumption (Gallon/100 Miles)	2.7	2.2	2.3	2
Power Train Index *	D	B	D	B
* D: Charging Depleting. B: Charging Blending				

Chapter 4 Real-time Optimal Dispatch

In order to run a time-series agent-based simulation, I have to treat each cab as an induvial agent. Each agent has attributes. I define matrix *A* to describe such attributes, where the rows represent each agent, arranged by alphabetical order of Cab ID, and the columns represent cab ID, brand-model, available time window, initial position, speed, progress, electric consumption, fuel consumption, and SoC for each agent from left to right.

In addition, each agent has 6 statuses: Not Available, Being Idle, En Route with Passenger, Pre-booked, En Route to Charging Station, and Being Charged. The status transition is as below.

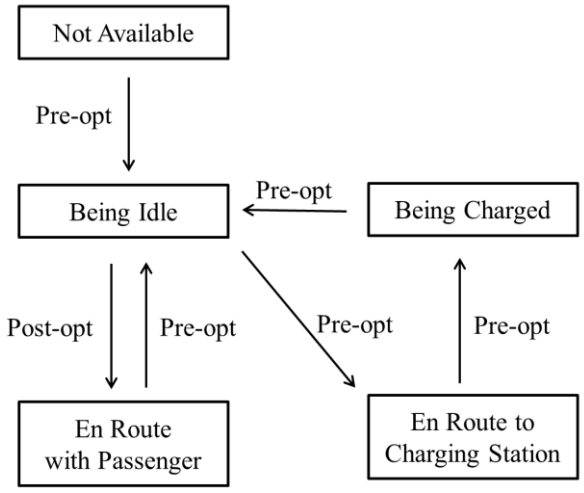

Figure 5 Real-time Optimization Status Transition

Pre-opt and post-opt means pre-optimization and post-optimization, which will be discussed later.

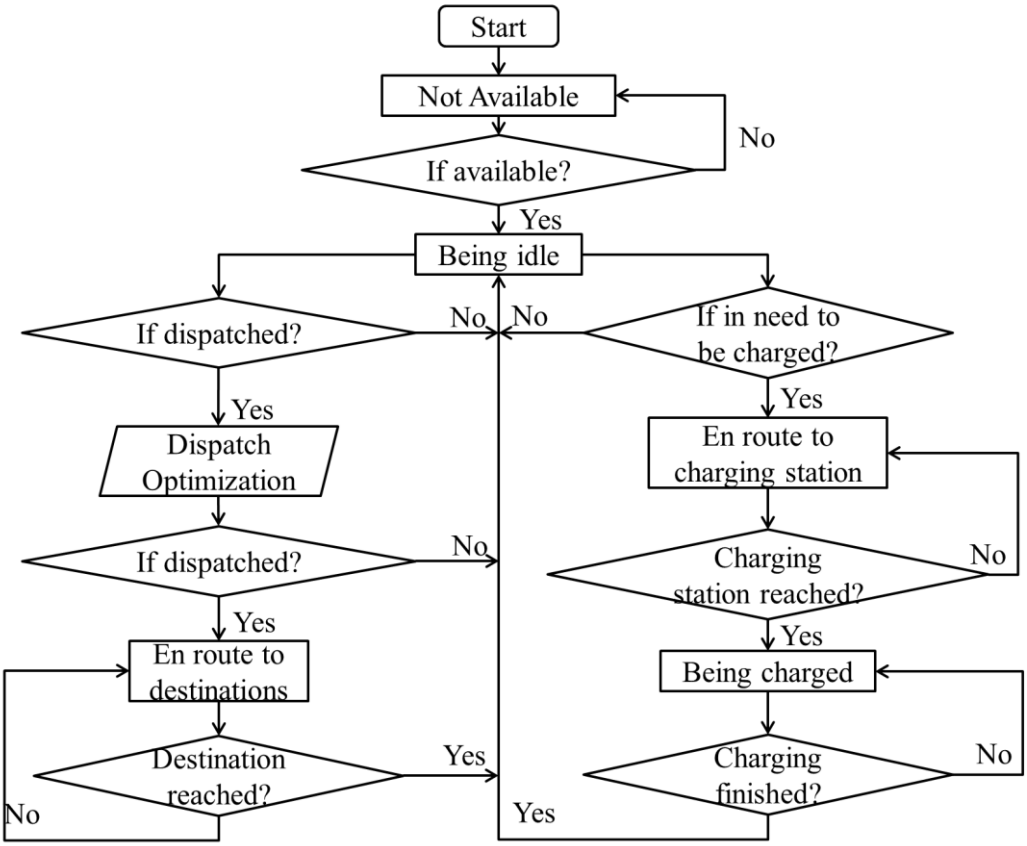

Figure 6 Real-time Optimization Flow Chart

For each cab, I can use a 5-digit number to represent its status.

Table 5 Real-time Status and Numeric Representation

Status	Numeric Representation
Not Available	10000
Being Idle	01000
En Route with Passenger	00100
En Route to Charging Station	00010
Being Charged	00001

Hence I use a matrix $S_{totCab\times 5}$ to represent the 5 statuses of all cabs, where each row represents each cab. Each column represents each of the 5 statuses. Denote $S = [s_{c,j}]$, then

$$s_{c,j} \in \{0,1\}, \forall\, c = 1{:}\, totCab, j = 1{:}\, 5 \tag{8}$$

$$\sum_{j=1}^{6} s_{c,j} = 1, \forall\, c = 1{:}\, totCab \tag{9}$$

Formula (8) makes S a binary matrix. Formula (9) means an agent must be at on one and only one status. Note that status is merely a way to better describe cab attributes. Our goal is to find the most cost effective way to dispatch and allocate cabs. Now the question is to determine which cab should change status and how to change its status.

The procedure for such question in each time slot is as below. When $t = t_1$,

Step 1: Pre-Optimization

The main purpose of this step is to traverse all cabs and calculate their energy consumption first, and then update their status and attributes if needed.

Update position, speed, electric energy consumption, fuel consumption, and SoC in A.

1.1 energy consumption calculation

Calculate electric consumption, fuel consumption and update SoC for moving and charging cabs. The energy consumption for moving cabs are based on vehicle characteristic, speed and SoC level.

The energy consumption for charging cabs is based on the maximum charging rate of vehicle or maximum charging rate at the charging station, whichever is smaller. Because the actual charge rate for each PHEV is limited by its own charger interface. Therefore, we must calculate SoC during the charging period specifically based on the characteristics of each particular PHEV. Once a PHEV departs a charging station, its SoC will start to decrease based on its driving pattern and also its power train type.

1.2 status change from “not available” to “being idle”

Check if any cab is available online now by comparing the current simulation time with the available time window of each cab.

Note that an off-duty cab only exists for human-driver case. With self-driving case, all cabs are always working whenever they are online. Also, if we don't see the GPS data for a cab in your data set, it is mostly because the car was turned off. Again, this does not happen in self-driving case. Therefore I can assume that a cab joins my fleet only when it becomes available based on the database. I have few cabs online at the beginning of my simulation. This is totally feasible because the population of passengers originally came from the cab data. That means I have fewer passengers at the beginning too.

For those just available online,

- Change the corresponding cab status from “not available” to “being idle”, which is to change the corresponding row in S from 10000 to 01000.
- Update the speed to zero.

1.3 status change from “en route with passenger” to “being idle”

Check drop-off time in $Response$ table to determine which cab should drop off its passenger. For those who finish their en route with passengers,

- Change the corresponding cab status from “en route with passenger” to “being idle”, which is to change the corresponding row in S from 00100 to 01000.
- Update cab position to where it drops off its passenger.
- Update cab speed to zero.

1.4 status change from “being idle” to “en route to charging station”

Check SoC in A for those cabs being idle to determine which cab should be charged.

For whose to be charged,

- Change the corresponding cab status from “being idle” to “en route to charging station”, which is to change the corresponding row in S from 01000 to 00010.
- Such cabs are not available for dispatch optimization in the next step, unless they arrive at charging station, finish charging, and go back to being idle again.

- Or the number of cabs being idle is less than the number of requests at that time slot. This means I will have two cost functions later in my next step.
- Update the speed.

Update ER2CS (En Route to Charging Station) table by adding in one row in the format below:

Cab ID	Departure Time	Departure Position	Arrival Time	Charging Station

Sort En Route to Charging Station table by Arrival Time.

1.5 status change from "en route to charging station" to "being charged"

Check Arrival Time in $En\ Route\ to\ Charging\ Station$ table to determine which cab arrives at the current time slot. For those who just arrives,

- Change the corresponding cab status from "en route to charging station" to "being charged", which is to change the corresponding row in S from 00010 to 00001.

Update Charging Station History Record table by adding in one row in the format below:

Cab ID	Charging Station	Start Time	End Time

This table has no use in our simulation, but is for overall charging loads analysis per charging station later.

- Update cab position to its corresponding charging station.

- Update cab speed to zero.

1.6 status change from "being charged" to "being idle"

Check if cabs being charged finish charging. For those cabs finishing charging,

- Change the corresponding cab status from "being charged" to "being idle", which is to change the corresponding row in S from 00001 to 01000.
- Update the speed to zero.

Step 2: Dispatch Optimization

Determine which cab should be dispatched to pick up a request for those "being idle".

Check $Request$ table to determine how many requests is needed at this time slot and where they are. The number of requests determines the number of rows of our optimization matrix variable. Denote there are $totPax$ number of passenger requests needed in this time slot. This number may change as time goes by, but it is determined within each time slot before the optimization. Denote $totCab$ is the total number of cabs. This is a constant number throughout the simulation.

I thereby define a matrix $D_{totPax \times totCab}$ to represent if a cab is dispatched in response to a request. Denote $D = [d_{p,c}], p = 1{:}\, totPax, c = 1{:}\, totCab$, then

$$d_{p,c} \in \{0,1\}, \forall\, p = 1{:}\, totPax, c = 1{:}\, totCab \tag{10}$$

Constraint (10) makes D a binary matrix. $d_{p,c} = 1$ means cab c is going to pick up passenger p, and its status should change from "being idle" to "en route with passenger". $d_{p,c} = 0$ means cab c is not going to pick up passenger p.

$$d_{p,c} = \begin{cases} 1, \text{cab } c \text{ picks up passenger } p \\ 0, \text{otherwise} \end{cases} \tag{11}$$

$$\sum_{c=1}^{totCab} d_{p,c} = 1, \forall\, p = 1{:}\, totPax \tag{12}$$

Constraint (12) means that each request must be satisfied by exactly one cab, which implies that no request should be left aside. It also indicates that no passenger should be picked up by more than one cab.

$$\sum_{p=1}^{totPax} d_{p,c} \leq 1, \forall\, c = 1{:}\, totCab \tag{13}$$

Constraint (13) ensures that one cab can only respond to at most one request, which indicates no car sharing in my simulation.

Note that matrix D is relatively huge and sparse as $totPax$ and $totCab$ grow. There are two ways to construct such variables more efficiently. One way is to have $totPax$ number of variables Y. Denote $Y = [y_p], p = 1{:}\, totPax$, then y_p represents the number of cab dispatched, from 1 to $totPax$. Constraint (10) does not apply for y_p, and constraint (12) is satisfied naturally. Constraint (13) is expressed as $y_{p_1} \neq y_{p_2}$, where $p_1 \neq p_2, \forall\, p_1 = 1{:}\, totPax, p_2 = 1{:}\, totPax$. However, this inequality constraint is no

more a convex constraint, since not-equal is non-convex. Hence it is harder to find the optimal solution.

The other way to construct such variables is to have $totCab$ number of variables Z. Denote $Z = [z_c], c = 1{:}\, totCab$, then z_c represents which passenger the cab shall pick up, from 1 to $totPax$, or the cab is not dispatched, simply $z_c = 0$. Again, constraint (10) does not apply for z_c. But constraint (12) is hard to describe.

In conclusion, Y and Z are denser but harder to describe or solve, hence matrix D is the best way to construct such variables. Matrix D also brings in another advantage in constraint (13) because it can be easily extended to car sharing case. For example, $\sum_{p=1}^{totPax} d_{p,c} \leq 2, \forall\, c = 1{:}\, totCab$ means one cab can pick up at most 2 requests at the same time. I will explain how to construct such sparse matrix with less memory usage and solve the optimization problem by reducing its computational complexity later.

I then record the geographical section number where each request happens in a matrix named R with $totPax$ number of rows. The first column of the matrix is the geographic section number for each request happened in this time slot, arranged by alphabetical order of Cab ID in the $Request$ table. The matrix R also contains when the passenger should arrive, and where the cab should drop off, according to historical data. Note that I use historical date to describe the driving behavior when the cab is occupied with passenger, and I use Google Map Distance Matrix API as reference for travel distance and traffic time when the cab is free.

The objective of this optimization is to find the cab which takes less energy to pick up a passenger and with high SoC. We can assume that a cab that consumes less energy to pick up a passenger also takes less time. So both the passenger and the taxi service are happy. If several cabs are in the same block as the passenger, the en route energy cost without passenger becomes 0. Therefore the cost function becomes zeros. There is no preference between cabs to pick up the passenger in the same block. That is why I add the SoC so that we will choose the cab with higher SoC. Sometimes the passenger we want to pick up wants to travel only a short distance and hence we do not really need high SoC to serve that request. But this is a minor affect.

$$\min_D \sum_{p=1}^{totPax} \sum_{c=1}^{totCab} d_{p,c}(\alpha \cdot EC(A_{c,1}, r_{p,1}) + (1-\alpha) \cdot FC(A_{c,1}, r_{p,1}) - \beta \times A_{c,5})) \quad (14)$$

α and β are adjustable coefficient, where $0 \leq \alpha \leq 1$. $A_{c,1}$ is the current position of each cab, and $A_{c,5}$ is their corresponding SoC. $r_{p,1}$ is the position for passenger p.

$EC(\cdot)$ and $FC(\cdot)$ are determined functions to calculate electric energy consumption and fuel consumption, respectively, from where the cab is to where the cab picks up the passenger.

Total energy can be obtained based on cab characteristics and average speed, which is derived from geographical distance and traffic time from where the available cab is to where the request happens. Then I need to determine how much electric energy and fuel energy allocated from total energy consumption. This is based on cab characteristics,

especially whether it is charge depleting or charge blending, its SoC, and its speed. This can be expressed in the following pseudo-code.

```
if cab is charge depleting,
    if (SoC * capacity – energy) >= available energy,
        electric = total energy / electric efficiency,
        fuel = 0,
    if (SoC * capacity – energy) < available energy,
        electric = (total energy – available energy) / electric efficiency,
        fuel = (energy - electric * electric efficiency) / fuel efficiency,
if cab is charge blending,
    if cab speed > 60 mph,
        if (SoC * capacity – energy) >= available energy,
            electric = total energy / electric efficiency,
            fuel = 0,
        if (SoC*capacity – energy) < available energy,
```

electric = (total energy – available energy) / electric efficiency,

fuel = (energy - electric * electric efficiency) / fuel efficiency,

if cab speed <= 60 mph,

electric = 0,

fuel = energy / fuel efficiency.

The objective is an integer linear optimization problem for binary matrix D.

In our optimization, I define only cabs that are being idle can be dispatched to pick up passengers. It is because cabs being idle can be easily located. However, a cab that is transferring a passenger and going towards the next pick up location to drop off that passenger may be more suitable. We can "project" a time when the taxi becomes available. This is discussed in the next chapter.

For now, we stick with cabs being idle. It means for $\forall\, c = 1{:}\, totCab$,

If $s_{c,2} = 0$, then $d_{p,c} = 0$, $\forall\, p = 1{:}\, totPax$;

If $s_{c,2} = 1$, then $d_{p,c} = 0$ or 1, $\forall\, p = 1{:}\, totPax$.

Mathematically, this can be represented as

$$(1 - s_{c,2}) \times d_{p,c} = 0, \forall\, p = 1{:}\, totPax, c = 1{:}\, totCab \qquad (15)$$

It can be simplified as

$$d_{p,c} \leq s_{c,2}, \forall\, p = 1{:}\, totPax, c = 1{:}\, totCab \tag{16}$$

In addition, I also want to limit the waiting time in response to a request so that no passenger should wait too long before picked up. The waiting time can be obtained by Google Map Distance Matrix API as the traffic time from one section where the cab is to the section where the passenger is. This constraint can be represented as below.

$$Time\left(A_{c,5}, r_{p,1}\right) \times d_{p,c} \leq t_{th}, \forall\, p = 1{:}\, totPax, c = 1{:}\, totCab \tag{17}$$

$Time(\cdot)$ is a table containing constant numbers to find the distance between two geographical sections based on current time. $A(:,5)$ is the 5th column in A recoding the positions of all cabs. Matrix R records where each request happens. Note that the time is rounded to its closest hourly time for convenience. Also I distinguish weekday times from weekend times for two time tables, because of different traffic pattern. t_{th} serves as a threshold for maximum waiting time.

Therefore the optimization problem is to solve:

Objective: (14)

Subject to: (11) (12) (13) (16) (17)

To use $intlinprog$ in MATLAB for mixed-integer linear programming (MILP) solver, I transform the matrix variable D to a vector of variables X. Denote $X = [x_i]$, then

$$X = [x_1\ x_2\ \dots\ x_{totPax \times totCab}\ ]$$
$$= [d_{1,1}\ \dots\ d_{totPax,1}\ d_{1,2}\ \dots\ d_{totPax,2}\ \dots\ d_{1,totCab}\ \dots\ d_{totPax,totCab}] \quad (18)$$

X can be achieved by $X = D(:)$ in MATLAB.

Remember that matrix D contains $(totPax \times totCab)$ elements. The total number of elements in vector X contains the same number. I will keep check the number of equality and inequality constraints as illustrated below.

The objective function (14) can be rewritten as:

$$\min_X f^T X \quad (19)$$

Denote $f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{totPax \times totCab} \end{bmatrix}$, then for $\forall\ k = 1{:}\ (totPax \times totCab)$,

$$f_k = \alpha \cdot EC\left(A_{c,5}, r_{p,1}\right) + (1 - \alpha) \cdot FC\left(A_{c,5}, r_{p,1}\right) - \beta \cdot A_{c,10} \quad (20)$$

where

$$k = p + (c - 1) \times totPax, p \in \mathbb{I}, c \in \mathbb{I}, p \in [1, totPax], p \in [1, totCab] \quad (21)$$

Constraint (10) can be rewritten as:

$$x_i \in \{0,1\}, \forall\ i = 1{:}\ (totPax \times totCab) \quad (22)$$

To rewrite constraint (12), I construct $totCab$ numbers of diagonal identity matrix, each with dimension $totPax$ by $totPax$. I then concatenate them horizontally as below.

$$M = [I, I, \dots I] \tag{23}$$

The dimension of matrix M is $totPax$ by $(totPax \times totCab)$.

Constraint (12) can be rewritten as:

$$A_{eq} \times X = b_{eq} \tag{24}$$

where $A_{eq} = M$, $b_{eq} = I_{totPax \times 1}$.

Recall that constraint (12) consists of $totPax$ equalities. Constraint (24) has the same number of equalities.

To rewrite constraint (13), I construct a $totCab$ by $(totPax \times totCab)$ diagonal matrix N where $I_{1 \times totPax}$ as block diagonal matrix and 0 for the rest of the elements as below.

$$N = \begin{bmatrix} I & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & I \end{bmatrix} \tag{25}$$

Such matrix can be achieved in MATLAB by $kron(eye(totCab),\ ones(1, totPax))$. However, $totCab$ and $totPax$ can be huge in some cases. So I use sparse matrix to represent N. In MATLAB, sparse matrix N can be obtained by $kron(speye(totCab), ones(1, totPax))$.

Constraint (13) is thereby rewritten as:

$$N \times X \leq I \tag{26}$$

Where the dimension of I is $totCab$ by 1.

Recall that constraint (13) consists of $totCab$ inequalities. Constraint (26) has the same number of inequalities.

To rewrite constraint (16), I extract the second column of matrix S and repeat it for $totPax$ times as below. The length of vector P is $(totPax \times totCab)$.

$$P = \begin{bmatrix} s_{1,2} \\ \vdots \\ s_{1,2} \\ s_{2,2} \\ \vdots \\ s_{2,2} \\ \vdots \\ s_{totCab\,,2} \\ \vdots \\ s_{totCab\,,2} \end{bmatrix} \tag{27}$$

This vector can be achieved in MATLAB by $repelem(S(:,2), totPax)$.

Constraint (16) can be rewritten as:

$$I \times X \leq P \tag{28}$$

Where I is a $(totPax \times toCab)$ by $(totPax \times totCab)$ identity matrix. Such matrix can be achieved in MATLAB by $eye(totPax \times totCab)$. Similarly, sparse matrix I can be obtained in MATLAB by $speye(totPax \times totCab)$.

Recall that constraint (16) contains $(totPax \times totCab)$ inequalities. Constraint (28) has the same number of inequalities.

Constraint (17) can be rewritten as:

$$Q \times X \leq t_{th} \times I \tag{29}$$

where the dimension of I is $(totPax \times totCab)$ by 1.

Matrix Q is a $(totPax \times totCab)$ by $(totPax \times totCab)$ diagonal matrix. The off-diagonal elements in Q are all zeros. Denote $Q = [q_{p,c}]$, then

$$q_{k,k} = Time(A_{c,5}, r_{p,1}), \forall\ k = 1{:}\ (totPax \times totCab) \tag{30}$$

where

$$k = p + (c-1) \times totPax, p \in \mathbb{I}, c \in \mathbb{I}, p \in [1, totPax], c \in [1, totCab] \tag{31}$$

Constraints (26) (28) (29) are all inequality constraints. They are all combined as:

$$A_{neq} \times X \leq b_{neq} \tag{32}$$

where

$$A_{neq} = \begin{bmatrix} N \\ I_1 \\ Q \end{bmatrix} \tag{33}$$

$$b_{neq} = \begin{bmatrix} I_2 \\ P \\ t_{th} \times I_3 \end{bmatrix} \tag{34}$$

Note that the dimensions of I_1, I_2, and I_3 are $(totPax \times totCab)$ by $(totPax \times totCab)$, $totCab$ by 1, and $totPax$ by 1, respectively. And the dimension of A_{neq} and b_{neq} are $(2 \times (totPax \times totCab) + totCab)$ by $(totPax \times totCab)$, and $(2 \times (totPax \times totCab) + totCab)$ by 1, respectively.

Therefore, the integer optimization in MATLAB is transformed into:

Objective: (19)

Subject to: (22) (24) (32)

The computational complexity to implement above optimization in MATLAB is very high and hence very time-consuming as the number of $totPax$ grows. However, there is a shortcut to express constraints (28) and (29) in a time efficient way.

Note that constraints (28) and (29) are supposed to reduce the number of variables by enforcing some of them to zeros.

For constraint (16), I can enforce the upper bound of $d_{p,c}$ to be 0 if $s_{c,2} = 0$, for $\forall\, p = 1{:}\, totPax$. Hence for constraint (28) derived from constraint (16), I can enforce the upper

bound of x_k to be 0 if $p_k = 0$ for $\forall\ k = 1{:}\ (totReq \times totTaxi)$. p_k is the k-th element of vector P.

Similarly for constraint (17), I can enforce the upper bound of $d_{p,c}$ to be 0 if $Time(A_{c,5}, r_{p,1}) \leq t_{th}$, for $\forall\ p = 1{:}\ totPax, c = 1{:}\ totCab$. Hence for constraint (29) derived from constraint (17), I can enforce the upper bound of x_k to be 0 if $q_k > t_{th}$ for $\forall\ k = 1{:}\ (totPax \times totCab)$. q_k is the k-th diagonal element of matrix Q.

After the optimization, some cabs will change their status from "en route with passenger" to "being idle". They will not be back to optimization, which is to pick up a new passenger later, until they drop off their passengers and become being idle again. After they becomes being idle again. They stay where they drop off their passengers.

Step 3: Post-Optimization

Update $Response$ table by adding in new cabs dispatched to pick up the request in the following format:

cabId	dispT	dispPos	pickUpT	pickUpPos	dispV	dropOffT	dropOffPos	ERV

From left to right, they are: cab ID, dispatch time, dispatch location, pick-up time, pick-up position, dispatch speed, drop-off time, drop-off location, and en route speed.

dispPos is where the cab is when a request rises. pickUpPos is where the passenger is.

Use Google Map Distance Matrix to describe the trip in response to a request. Use historical data, more specifically, the drop off time and drop off position for en route with passenger. The distance is still based on Google. The time is based on historical data.

The pick-up time is the current simulation time. The pick-up location is where the cab is right now. The drop-off time and drop-off location can be obtained from matrix R, since it records where a request happens, when the cab arrives at its desired destination, and where the destination is. Sort $Response$ table by $Dropoff\ Time$.

Change the corresponding cab status from “being idle” to “en route with passenger”, which is to change the corresponding row in S from 01000 to 00100.

Update position and speed for en route cabs.

Chapter 5. Pre-booked Optimal Dispatch and Charging Decision

5.1 Discussion about real-time optimal dispatch algorithm

The previous algorithm solves optimal dispatch in a sense that it can dispatch the best cab that uses least energy to pick up a passenger. However, it does not consider the energy needed to finish the en route trip with passenger. This means it might end up using a cab with high SoC to pick up a passenger that only needs to travel a short distance. Also it does not include charging decision into the optimization. Not considering the energy needed to finish the en route trip with passenger is understandable, because it is unchangeable according to the data set. However, the energy allocated to electricity and fuel must be taken into consideration, so this is the first shortage of the previous algorithm. What is more, the algorithm does not include charging decision into the optimization problem, which makes it insufficient for PHEV. Both shortages are due to the fact that the optimization does not include SoC in it, and SoC is iterative with time, meaning that we can never its change before the optimization. Therefore I must include SoC as one of the variables, and I must consider time as a critical factor in my optimization below.

5.2 Variables in pre-booked optimal dispatch and charging

The new decision variable is developed based on matrix $D_{totPax \times totCab}$. I further extend it to a third dimension with time involved. Denote the new decision variable is $\theta_{c,p}[t]$.

$$\theta_{c,p}[t] = \begin{cases} 1, \text{cab } c \text{ is with passenger } p \text{ at time } t \\ 0, \text{otherwise} \end{cases} \tag{35}$$

$$\forall\, c = 1{:}\, totCab, p = 1{:}\, totPax, t = 1{:}\, totTime$$

$\theta_{c,p}[t]$ is a time-series of integer variables where $t = 1{:}\, totTime$. $totTime$ is the total time slots for the horizon of this optimization problem. Horizon is how far away we are planning ahead in my optimization.

Each passenger request can be represented as line segments in the third dimension with several ones. We also define a set of variables δ_{c,p_1,p_2} to indicate that cab c picks up both p_1 and p_2 and no other passengers in between.

$$\delta_{c,p_1,p_2} = \begin{cases} 1, \text{cab c pick up both } p_1 \text{ and } p_2 \text{ and no other trips in between} \\ 0, \text{otherwise} \end{cases} \tag{36}$$

$$\forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax$$

We further define a new set of decision variables φ_{c,p_1,p_2} for charging decision.

$$\varphi_{c,p_1,p_2} = \begin{cases} 1, \text{cab } c \text{ goes to charging station } p_1 \text{ and } p_2 \\ 0, \text{otherwise} \end{cases} \tag{37}$$

$$\forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax$$

Note that φ_{c,p_1,p_2} has nothing to do with time t, because it is linked to a certain passenger request p, and we know when that request happens. The set of φ_{c,p_1,p_2} is about

SoC going up, while the set of $\theta_{c,p}[t]$ is about SoC going down. There are certain constraints linking with $\theta_{c,p}[t]$ and certain constrains with φ_{c,p_1,p_2} . And there are certain constraints to link both sets of variable together. Further information about variable φ_{c,p_1,p_2} will be introduced later.

To fully describe the change of SoC, and charging and discharging behavior for each cab, I define $elec_c[t]$ for how much electric energy one cab should have consumed per time slot. I also define $chrg_c[t]$ for how much electric energy charged to a cab in a time slot.

Last but not least, we introduce a set of auxiliary variables for SoC, namely $SoC_c[t]$. Since the SoC of each cab at each time slot before the optimization is unknown, we should treat SoC as variables rather than parameters. Note that SoC is just auxiliary variables, meaning that it is determined by other decision variables. However, it is bounded in feasible set. Also note that SoC is a continuous variable from 0 to 1 rather than binary variable that can only pick from 0 or 1. Further information about variable SoC will be introduced later.

5.3 Constraints in pre-booked optimal dispatch and charging

(1) Each request must be satisfied by exactly one cab.

The first constraint is that passenger p must be picked up by one and only one cab. Hence at time t_p^s,

$$\sum_{c=1}^{totCab} \theta_{c,p}[t_p^s] = 1, \forall\, p = 1{:}\, totPax \tag{38}$$

Here t_p^s represents the start time of request for passenger p, as it is known prior to the optimization. Also note that the start time of charging decision is unknown ahead of time. I use constraint (e) to describe the behaviors of charging decisions later.

The total number of equalities is $totPax$.

(2) Each request must be conducted thoroughly by one cab and cannot change to another cab in the middle of the service.

The second constraint is that when passenger p is picked up by cab c during its desired time interval $[t_i^s, t_i^e]$, the request must be conducted thoroughly. Here t_i^s and t_i^e represent the start and end time of request for passenger p, respectively. This constraint ensures that no request can shift to a different cab in the middle of the service. It also means that there should be no breaking points for request p in the time interval, no matter the request is conducted by which cab. Therefore the constraint can be expressed as below.

$$\theta_{c,p}[t] - \theta_{c,p}[t+1] = 0$$
$$\forall\, c = 1{:}\, totCab, p = 1{:}\, totPax, t_p^s \le t \le t_p^e - 1 \tag{39}$$

Note that the start time and location and the end time and location of request for passenger p are known prior to the optimization because the requests are pre-booked ahead of time.

The total number of inequalities is $totCab \times (\sum_{p=1}^{p=totPax}(t_p^e - t_p^s))$.

(3) No cab can take multi-tasks.

The third constraint is that no cab can take multi-tasks. In other words, no car sharing is allowed. Specifically, at any time t, no cab can pick up more than one passenger. Here I assume every request happens at the beginning of each time slot. Then it can be expressed as below.

$$\sum_{p=1}^{totPax} \theta_{c,p}[t] \leq 1, \forall\, c = 1{:}\, totCab, t = 1{:}\, totTime \tag{40}$$

The total number of inequalities is $(totCab \times totTime)$.

Note that this constraint can be easily modified for car sharing by changing less than 1 to less than any number of car sharings.

(4) Only certain pairs of requests can be taken by one cab in time.

The fourth constraint is the core for optimal dispatch in pre-booked optimization problem. It ensures that every cab must have enough time to move from one request to another. Since I know exactly when and where a request happens and when and where it

ends, I should be able to know prior to the optimization that whether or not one cab can take both the request for passenger p_1 and the request for passenger p_2.

If one cab, no matter which cab it is, cannot pick up both requests, it can be explained by two reasons. First, both requests may happen simultaneously so they overlap in time. Second, both requests do not overlap in time, denote request p_1 ends before request p_2 starts, one cab does not have enough time to travel from the drop-off location when request p_1 ends to the pick-up location when request p_2 starts. The travel time is calculated based on Google Map Distance Matrix API.

Therefore I define a constant symmetric matrix B beforehand, which describes the availability to travel between any two requests for a cab. Matrix B is a $totPax$-by-$totPax$ binary matrix. Denote $B = [b(p_1, p_2)]$.

$$b(p_1, p_2) = \begin{cases} 1, \text{one cab can take both } p_1 \text{ and } p_2 \text{ in time} \\ 0, \text{otherwise} \end{cases} \tag{41}$$

Note that matrix B is constant and symmetric, which does not impose two-way relationship between passenger p_1 and passenger p_2. Denote request for passenger p_1 ends before request for passenger p_2 starts. Hence no cab can travel from request p_2 to request p_1, even if one cab can travel from request p_1 to pick up request p_2. It seems that $b(p_1, p_2) = 1$ while $b(p_2, p_1) = 0$. However, in practice, no one cares about travelling from request p_2 to request p_1 because no one can travel back in time. In fact, if one cab have to pick up request p_1 and p_2, and cab knows prior to the optimization that request p_1

ends before request p_2 starts, the cab only cares about one question. That is whether it can drop off passenger p_1 first, and then pick up passenger p_2 in time. That is why $b(p_2, p_1) = b(p_1, p_2) = 1$. Therefore matrix B must be symmetric.

If $b(p_1, p_2) = 1$, then one cab can pick up both request p_1 and request p_2, no constraint is applied.

If $b(p_1, p_2) = 0$, then one cab can at most pick one from p_1 and p_2.

$$b(p_1, p_2) = \begin{cases} 1, t_{p_2}^s - t_{p_1}^e \geq t_{p_1,p_2} \text{ or } t_{p_1}^s - t_{p_2}^e \geq t_{p_2,p_1} \\ 0, \text{otherwise} \end{cases} \quad (42)$$

where t_{p_1,p_2} is the travel time from the drop-off location of request for passenger p_1 to the pick-up location of request for passenger p_2. Similarly, t_{p_2,p_1} is the travel time from the drop-off location of request for passenger p_2 to the pick-up location of request for passenger p_1. t_{p_1,p_2} and t_{p_2,p_1} can be obtained from Google Map Distance Matrix API.

Next, I apply matrix B to the decision variable $\theta_{c,p}[t]$ as below.

$$\theta_{c,p_1}\left[t_{p_1}^s\right] + \theta_{c,p_2}\left[t_{p_2}^s\right] \leq 1, \forall\, b(p_1, p_2) = 0, p_1, p_2 = 1{:}\, totPax, c = 1{:}\, totCab \quad (43)$$

This constraint means, if it is impossible to pick up both p_1 and p_2 for one cab, which is represented in matrix B, then the cab cannot make both $\theta_{c,p_1}\left[t_{p_1}^s\right]$ and $\theta_{c,p_2}\left[t_{p_2}^s\right]$ be 1.

This inequality can be achieved by the following pseudo code.

For $p_1 = 1: totPax$

For $p_2 = (p_1 + 1): totPax$

If $b(p_1, p_2) = 0$

For $c = 1: totCab$

$\theta_{c,p_1}[t^s_{p_1}] + \theta_{c,p_2}[t^s_{p_2}] \leq 1$

The total number of inequalities depends on the total number of zeros in the upper triangle of matrix B.

Here I must also consider whether the cab can pick up the request for passenger p from its initial position. Note that the initial position of each cab is known. The pick-up and pick-up location of p is also known. I can hence set the upper bound for $\theta_{c,p}[t^s_p]$ to be 0, if cab c cannot pick up p in time.

(5) Requests leave time windows for cab to charge.

The fifth constraint comes to the optimal charging decision. The previous constraint gives us an opportunity to select certain combinations of passengers for each cab. Therefore one cab can pick up both requests because there is enough time to drop off one request and pick up the next. If the time in between is big enough, this actually leaves us a time window where we can insert a possibility to charge the cab. We use φ_{c,p_1,p_2} to indicate whether to charge or not in that time window.

Recall that $\varphi_{c,p_1,p_2} = 1$ if cab c goes to charging station right before picking up passenger p. Otherwise $\varphi_{c,p_1,p_2} = 0$. It indicates that if the cab does not pick up passenger p, then there will be no charging at all before that, because there is no action. Therefore charging decision is linked to picking-up-passenger decision.

Note that φ_{c,p_1,p_2} is not related to time. Because in terms of timeline, a cab just picks up a passenger, then becomes idle, then picks up another passenger, then becomes idle, so on and so forth. As long as we know the arrangement, all we need to know about charging decision is whether the cab is going to charge before picking up each passenger. If the cab will charge before picking up a certain passenger, the idle time before that pick-up becomes charging time.

To ensure the availability to be charged before certain request, we need know when and where each request starts. The traffic time from each charging station to each request can be obtained from Google Map Distance Matrix API. If we ensure that all charging decision must fully charge a cab, then the cab c cannot be charged for a certain time amount prior to request i because the cab is en route to pick up request p.

Similarly, when and where each request ends is given. The traffic time from each request to each charging station can be obtained from Google Map Distance Matrix API. So cab c cannot be charged for a certain time after the cab drops off request p.

By doing so, there may be a time window to charge the cab. It does not have to be fully charged in the time window. Whether to charge or not in the time window is can be implied through cost function.

To find out where exists such possibility to insert a time window to charge, we need to use the matrix B and construct matrix C with same dimension, $(totPax + 1)$-by-$(totPax + 1)$. Denote $C = [c(p_1, p_2)]$.

$$c(p_1, p_2) = \begin{cases} 1, \text{one cab can be fully charged between } p_1 \text{ and } p_2 \\ 0, \text{otherwise} \end{cases} \quad (44)$$

Remember that $b(p_1, p_2) = 1$ means that one cab can take both request p_1 and request p_2. $t_{p_1}^s$, $t_{p_1}^e$, $t_{p_2}^s$, and $t_{p_2}^e$ are the time when request p_1 starts and ends, and the time when request p_2 starts and ends, respectively. We further define $t_{p_1}^{cs}$ as the time when one cab departures from the charging station closest to request p_1 and arrives at request p_1 at $t_{p_1}^s$. $t_{p_1}^{ec}$ is the time when one cab drops off request p_1 at $t_{p_1}^e$ and arrives at charging station closest to request p_1. Similarly, $t_{p_2}^{cs}$ is the time when one cab departures from the closest charging station to request p_2 and arrives at request p_2 at $t_{p_2}^s$. $t_{p_2}^{ec}$ is the time when one cab drops off request p_2 at $t_{p_2}^e$ and arrives at charging station closest to request p_2. $t_{p_1}^s$, $t_{p_1}^e$, $t_{p_2}^s$, and $t_{p_2}^e$ are given in historical data. $t_{p_1}^{cs}$, $t_{p_1}^{ec}$, $t_{p_2}^{cs}$, and $t_{p_2}^{ec}$ are calculated based on Google map Distance Matrix API.

$$c(p_1, p_2) = \begin{cases} 1, t_{p_2}^{cs} - t_{p_1}^{ec} \geq t_c \text{ or } t_{p_1}^{cs} - t_{p_2}^{ec} \geq t_c \\ 0, \text{otherwise} \end{cases} \quad (45)$$

t_c is the charging time to fully charge a fully depleted battery for an EV.

Only when $c(p_1, p_2) = 1$ can a cab have time to drop off request p_1, for the charging station, and then pick up request p_2, or the opposite way. Therefore the constraint can be written as below.

$$\varphi_{c,p_1,p_2} \leq C(p_1, p_2), \forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax \quad (46)$$

Note that we do not have to write this constraint into inequality linear constraint matrix. Instead, because matrix C is known, we can write this constraint into upper bound of φ_{c,p_1,p_2}. In other words, if $C(p_1, p_2) = 0$, it will force the upper of φ_{c,p_1,p_2} to be 0.

Matrix B and C should have the same dimension, which is $totReq$-by-$totReq$. Matrix C is sparser than matrix B in terms of more zeros. It is because only when $b(p_1, p_2) = 1$ can $c(p_1, p_2) = 1$ becomes possible.

(6) The charging decision between two requests is possible only if the cab takes both requests.

Only when cab c picks up both p_1 and p_2, can cab c charging between p_1 and p_2 becomes possible.

Table 6 Pre-booked Optimization Variable Relationships

$\theta_{c,p_1}[t^s_{p_1}]$	$\theta_{c,p_2}[t^s_{p_2}]$	φ_{c,p_1,p_2}
$\theta_{c,p_1}[t^s_{p_1}] = 1$	$\theta_{c,p_2}[t^s_{p_2}] = 1$	φ_{c,p_1,p_2} can be 1 or 0
$\theta_{c,p_1}[t^s_{p_1}] = 0$	$\theta_{c,p_2}[t^s_{p_2}] = 1$	φ_{c,p_1,p_2} must be 0
$\theta_{c,p_1}[t^s_{p_1}] = 1$	$\theta_{c,p_2}[t^s_{p_2}] = 0$	φ_{c,p_1,p_2} must be 0
$\theta_{c,p_1}[t^s_{p_1}] = 0$	$\theta_{c,p_2}[t^s_{p_2}] = 0$	φ_{c,p_1,p_2} must be 0

Mathematically, the table above can be expressed as below.

$$\varphi_{c,p_1,p_2} \leq \theta_{c,p_1}[t^s_{p_1}], \forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax \tag{47}$$

$$\varphi_{c,p_1,p_2} \leq \theta_{c,p_2}[t^s_{p_2}], \forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax \tag{48}$$

The total number of inequalities is $2 \times totCab \times totPax \times (totPax - 1)$.

(7) The charging decision between two requests is possible only if the cab takes no request in between.

$$\varphi_{c,p_1,p_2} \leq 1 - \theta_{c,p}[t^s_p] \tag{49}$$

$$\forall\, c = 1{:}\, totCab, p_1 < p < p_2, p_1, p_2 = 1{:}\, totPax$$

(8) Define the variable to describe the process between passenger trips.

Before we jump into SoC iteration, it is better if we first take a look at what are the scenarios where PHEV charges or gets charged.

There are three scenarios where PHEV charges:

(a) within a passenger trip

(b) from dropping off a passenger to picking up another passenger

(c) from dropping off a passenger to a charging station, and from a charging station to picking up the next passenger

So far we have expressed (a) and (c). Now we will look at the process between trips.

Similar to define a variable to describe whether to get charged between p_1 and p_2, I define a variable δ_{c,p_1,p_2} meaning that cab will pick up both p_1 and p_2 with no passenger request in between. The reason why this variable exists is to easily express the energy consumption between trips. Therefore the constraints should be similarly to φ_{c,p_1,p_2}.

$$\delta_{c,p_1,p_2} \leq \theta_{c,p_1}\left[t_{p_1}^{s}\right], \forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax \tag{50}$$

$$\delta_{c,p_1,p_2} \leq \theta_{c,p_2}\left[t_{p_2}^{s}\right], \forall\, c = 1{:}\, totCab, p_1, p_2 = 1{:}\, totPax \tag{51}$$

$$\delta_{c,p_1,p_2} \leq 1 - \theta_{c,p}\left[t_p^s\right]$$

$$\forall\, c = 1{:}\, totCab, p_1 < p < p_2, p_1, p_2 = 1{:}\, totPax \tag{52}$$

However, δ_{c,p_1,p_2} cannot be as flexible as φ_{c,p_1,p_2}. Because when $\theta_{c,p_1}\left[t_{p_1}^s\right]$ and $\theta_{c,p_2}\left[t_{p_2}^s\right]$ are both 1, φ_{c,p_1,p_2} can be 1 or 0. On the other hand, δ_{c,p_1,p_2} must be 1, if both $\theta_{c,p_1}\left[t_{p_1}^s\right]$ and $\theta_{c,p_2}\left[t_{p_2}^s\right]$ are 1, and for all the $p_1 < p < p_2$, $\theta_{c,p}\left[t_p^s\right] = 0$.

Therefore I must add in a new constraint as below.

$$\delta_{c,p_1,p_2} \geq \frac{1}{2}\left(\theta_{c,p_1}\left[t_{p_1}^s\right] + \theta_{c,p_2}\left[t_{p_2}^s\right] - 1\right) - \sum_p \theta_{c,p}\left[t_p^s\right]$$

$$\forall\, c = 1{:}\, totCab, p_1 < p < p_2, p_1, p_2 = 1{:}\, totPax \tag{53}$$

(9) Define variables to express electric consumption for each cab at each time slot.

I define a variable $e_c[t]$ to express the electric consumption for cab c at time t.

(10) Define a variable to express the average charging rate.

I define a variable $chrg_{c,p_1,p_2}$ to express the average charging rate if cab c will go to charging station between picking up p_1 and p_2. Of course $chrg_{c,p_1,p_2}$ is meaningful only when $\varphi_{c,p_1,p_2} = 1$.

(11) SoC iterates and must be within acceptable range.

We express all the variables needed to calculate electric consumption and come to the SoC variable. Because we do not know the SoC of each cab before the optimization, we should treat SoC as variables rather than parameters. Note that SoC is just auxiliary variables, meaning that it is determined by other decision variables. Also note that SoC is a continuous variable from 0 to 1 rather than binary variable that can only pick from 0 or 1.

Recall that some previous constraints make it possible for cab to charge during a time window. Here we assume that the cab will stay at the charging station long enough that it can fully charge a fully depleted battery.

SoC should have two subscripts for cab number and time slot, namely $SoC_c[t]$.

SoC must always be between 0 and 100% all the time. This is a bounding constraint that we must consider for electric vehicles.

$$0 \leq SoC_c[t] \leq 100\%, \forall\, c = 1{:}\, totCab, t = 1{:}\, totTime \tag{54}$$

Note that SoC cannot bound the feasible set unless it is related to other decision variables. Also note that SoC is iterative, meaning that the SoC of cab c at time slot t depends on its SoC at previous time and its decision on previous time slot t. The key here is to properly express the iteration over time for SoC, especially list all the scenarios where a PHEV charges or gets charged.

At the first time slot t_0, I assume that the SoC for every cab c is 100%.

$$SoC_c[t_0] = 100\% \tag{55}$$

This constraint is implemented by setting both the upper bound and lower bound of $SoC_c[t_0]$ to be 1.

The iteration can be expressed as below.

$$SoC_c[t+1] = SoC_c[t] - e_c[t]/capacity + chrg_c[t]/capacity$$
$$\forall c, t = 1{:}\, totTime - 1 \tag{56}$$

$e_c[t]$ is the amount of discharge for cab c at time slot t. $chrg_c[t]$ is the amount of charge for cab c at time slot t. $capacity$ is the battery capacity for the EV.

For $\forall\ c = 1{:}\, totCab, t = 1{:}\, totTime$, $e_c[t]$ should have the following constraints.

$$0 \le e_c[t]$$
$$\le \sum_{p=1}^{totPax} \theta_{c,p}[t] \times L_p[t] + \sum_{p_2=p_1+1}^{totPax} \sum_{p_1=1}^{totPax-1} \delta_{c,p_1,p_2} \times M_{p_1,p_2}[t] \tag{57}$$
$$+ \sum_{p_2=p_1+1}^{totPax} \sum_{p_1=1}^{totPax-1} \varphi_{c,p_1,p_2} \times \left(P_{p_1,p_2}[t] + Q_{p_1,p_2}[t]\right)$$

First, the electricity consumption must be greater than zero obviously. Second, the electricity consumption cannot be greater than the energy cost at any time slot t, no matter if cab c will pick up p. Note that $e_c[t]$ is a continuous variable.

The total number of inequalities is $totCab \times totTime$.

Here I must explain the meaning of $L_p[t]$, $M_{p_1,p_2}[t]$, $P_{p_1,p_2}[t]$, and $Q_{p_1,p_2}[t]$.

$L_p[t]$ is the energy cost for trip p at time interval t.

$M_{p_1,p_2}[t]$ is the energy cost between p_1 and p_2 at time interval t. There is no other trip or charging decision between p_1 and p_2.

$P_{p_1,p_2}[t]$ is the energy cost from p_1 to the charging station that is closest to p_1 and p_2 at time interval t.

$Q_{p_1,p_2}[t]$ is the energy cost from the charging station that is closest to p_1 and p_2, to p_1 at time interval t.

Back to SoC iteration, $chrg_c[t]$ should have the following constraint.

$$0 \leq chrg_c[t] \leq \sum_{p_2=p_1+1}^{totPax} \sum_{p_1=1}^{totPax-1} \varphi_{c,p_1,p_2} \times N_{p_1,p_2}[t] \tag{58}$$

$N_{p_1,p_2}[t]$ is the charging rate where the charging station can fully charge a fully depleted battery within one time interval, if one cab goes to charging station between p_1 and p_2.

We should have enough time to charge the PHEV based on whatever SoC we have. If we pick up p_1 and p_2, we should have enough time to go to the charging station, stay there enough to fully charge a fully depleted battery, and leave. The charging time is hence the battery capacity divided by the charging rate. So for some cases, we can go there partially charged. When the cab is fully charged, it just stays there for the remaining time.

$L_p[t]$, $M_{p_1,p_2}[t]$, $P_{p_1,p_2}[t]$, and $Q_{p_1,p_2}[t]$ are all constant matrices.

Note that $chrg_c[t]$ is a continuous variable. Since it is not in cost function, it is a loss variable, meaning that it may not have a meaningful value at certain time t. Therefore I have to set the value for SoC at the end of the charging process to be exactly 100%. By doing so, we do not really care how much energy charged into EV.

If $\varphi_{c,p_1,p_2} = 1$, $SoC_c[t_c^e] = 100\%$.

$$\varphi_{c,p_1,p_2} \times 100\% \leq SoC_c[t_c^e] \leq \left(2 - \varphi_{c,p_1,p_2}\right) \times 100\%, \forall\, p_1, p_2 \tag{59}$$

5.4 Objective function and optimization problem

Cost function is as below.

$$\max_{\theta,\varphi,\delta,e,chrg,SoC} \sum_{c=1}^{totCab} \sum_{t=1}^{totTime} e_c[t] - \sum_{c=1}^{totCab} \sum_{p_2=p_1+1}^{totPax} \sum_{p_1=1}^{totPax-1} \left(\varphi_{c,p_1,p_2} \times \alpha_{p_1,p_2} + \theta_{c,p_1,p_2} \times \beta_{p_1,p_2}\right) \tag{60}$$

The cost function tries to maximize the usage of electricity cost while minimizing the travel distance without passenger. This will enforce PHEV to use electricity before fuel, and reduce all the unnecessary travel while serving all the passengers in time. α_{p_1,p_2} is a coefficient for travel distance between the drop-off location p_1 and pick-up location p_2. β_{p_1,p_2} is a coefficient for travel distance between the drop-off location p_1 and the charging station that is closest to both p_1 and p_2, plus the travel distance between the charging station and pick-up location p_2.

Therefore the pre-booked optimization problem is as below.

Objective: (60)

Subjective to: (38)(39)(40)(43)(46)(47)(48)(49)(50)(51)(52)(53)(54)(56)(57)(58)(59)

Chapter 6. Simulation Results

6.1 Simulation for real-time optimization problem for one hour

I use the data set for 536 cabs in San Fransisco.

$$totTaxi = 536 \tag{61}$$

Within all the top selling PHEV brands in the United States, Nissan Leaf is the best-selling all-electric vehicle, which has sold 82,138 units through June 2015, followed by Tesla Model S, which has sold 49,720 units since June 2012. As for PHEV selling, Chevrolet Volt is the most popular PHEV model, which has sold 78,979 units since it launched in market on December 2010, followed by Toyota Prius PHV – 40,992 units, and Ford Fusion Energi – 21,929 units, another model which we considered in our later analysis – Honda Accord PHEV, has sold 1,034 through 2015 in the entire United States [17].

[7] assumed the following mixture of different PHEV brands:

- Chevrolet Volt: 161 vehicles,
- Honda Accord Plug-in: 125 vehicles,
- Ford Fusion Energi: 125 vehicles,
- Toyota Prius Plug-in: 125 vehicles.

But based on the sale records of four EVs, the actual distribution should be as below:

Chevrolet Volt:

- $78979 / (78979 + 1034 + 21929 + 40992) \times 536 = 296.1699 = 296$

Honda Accord Plug-in:

- $1034 / (78979 + 1034 + 21929 + 40992) \times 536 = 3.8775 = 4$

Ford Fusion Energi:

- $21929 / (78979 + 1034 + 21929 + 40992) \times 536 = 82.2334 = 82$

Toyota Prius Plug-in:

- $40992 / (78979 + 1034 + 21929 + 40992) \times 536 = 153.7193 = 154$

To illustrate the real-time design, I use the first 5 cabs in the 536 cabs and simulate it for an hour from 05/18/2008 07:00:00 am to 05/18/2008 08:00:00 am. There are in total 10 trips as Table 7. After the simulation, the result is as Table 8.

From both tables, you can see that by using real-time optimization, all passengers can be picked up within desired time and the total energy consumption in transition is minimum.

Table 7 Historical Trip Data

Taxi ID	Pick-up Time	Pick-up Location	Drop-off Time	Drop-off Location	En Route Distance	En Route Time	En Route Speed
'abgibo'	1211123025	54	1211123618	43	4473.9412	593	7.5445
'abjoolaw'	1211123072	54	1211123421	49	1653.7381	349	4.7385
'abgibo'	1211123765	48	1211124275	43	3141.3497	510	6.1595
'abjoolaw'	1211124005	54	1211124826	49	2789.7036	821	3.3979
'abgibo'	1211124280	53	1211124804	43	3734.0316	524	7.126
'abdremlu'	1211125012	54	1211125515	54	3199.2311	503	6.3603
'abgibo'	1211125173	53	1211125293	43	1363.9755	120	11.3664
'abgibo'	1211125861	53	1211126302	43	3139.6926	441	7.1194
'abdremlu'	1211126035	53	1211126637	54	3731.3318	602	6.1982
'abjoolaw'	1211126062	54	1211130701	50	30200.305	4639	6.51

Table 8 Real-time Optimization

Cab ID	Dispatch Time	Dispatch Location	Dispatch Speed	Pick-up Time	Pick-up Location	Drop-off Time	Drop-off Location	En Route Speed
'abcoij'	1211123025	59	12.4204	1211123618	43	1211123052	54	7.8740
'abdremlu'	1211123072	45	10.4842	1211123421	49	1211123136	54	9.8562
'abgibo'	1211123765	48	5.5235	1211124275	43	1211123765	48	0
'abdremlu'	1211124005	63	3.8455	1211124826	49	1211124049	54	14.1522
'abcoij'	1211124280	63	11.3558	1211124804	43	1211124374	53	4.7170
'abcoij'	1211125012	63	0	1211125515	54	1211125056	54	14.1522
'abdremlu'	1211125173	63	257	1211125293	43	1211125274	53	4.3900
'abdremlu'	1211125861	63	14.3617	1211126302	43	1211125962	53	4.3900
'abcoij'	1211126035	63	15.9880	1211126637	54	1211126136	53	4.3900
'abjoolaw'	1211126062	45	0.2372	1211130701	50	1211126128	54	9.5575

6.2 Simulation Results for 5 Cabs with 10 Trips in 1 Hour

I keep using the first 5 cabs listed in the 536 cabs to simulate for pre-booked optimization problem for one hour. There are 10 trips in the one hour time window from 05/18/2008 07:00:00 to 05/18/2008 08:00:00.

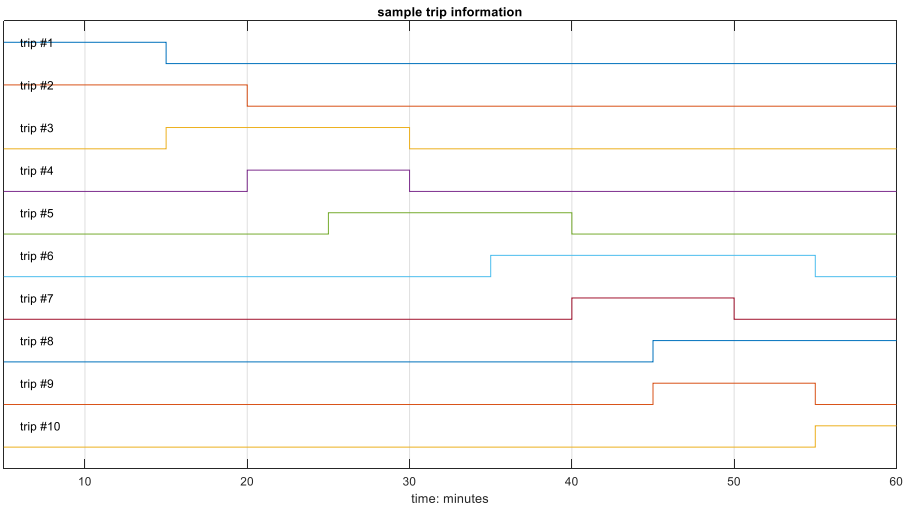

Figure 7 Test Case Trip Timeline

I further modified the time so as to cover all kinds of combinations for different trips. You can see from the figure that there are trips that departure at the same time slot, such as trip #1 and #2, #8 and #9. There are trips that arrive at the time slot, such as #3 and #4. There are also trips that departure at the time when another arrives, such as #1 and #3, #2 and #4, #5 and #7, #9 and #10. There are trips that full covers the time for another trip, such as #3 and #4, #6 and #7, #6 and #9.

The simulation results are as below.

First is about which cab picks up which passenger.

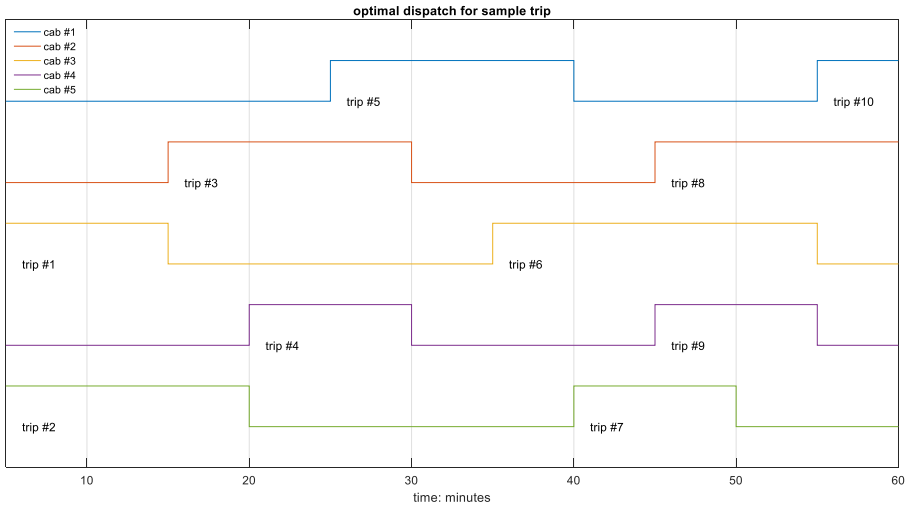

Figure 8 Cab Assignment

From the figure, we know that cab #1 picks up passenger #5 and #10. Cab #2 picks up passenger #3 and #8. Cab #3 picks up passenger #1 and #6. Cab #4 picks up passenger #4 and #9. Cab #5 picks up passenger #2 and #7.

Second is about the SoC, discharge and charge amount for each cab. The result is shown as below. It clearly show that cab #2 to cab #5 get charged in the middle. Their SoC drops when discharge, then comes back to 100% after charging, and then drops because of new trips. Cab #1 # 3 and #5 do not go to charging station. As for cab #5, t is because cab #5 picks up passenger #3 and #6. There is not enough time to go from #3 to charging station, and then come back to pick up #6 in time.

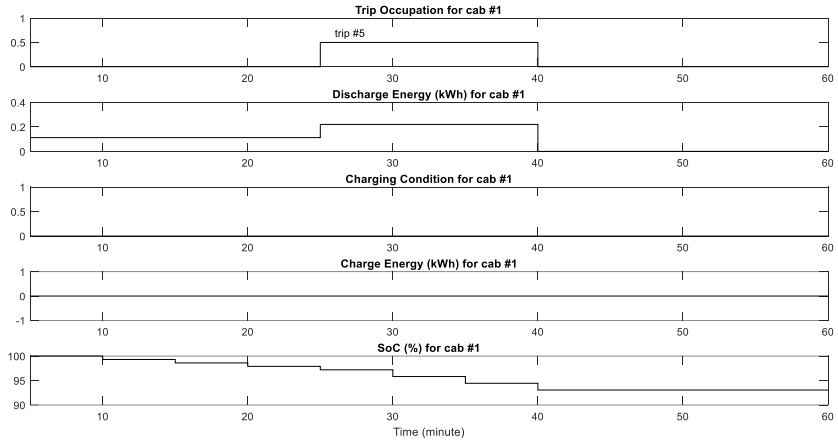

Figure 9 SoC, Discharge and Charge Amount over Time for Cab #1

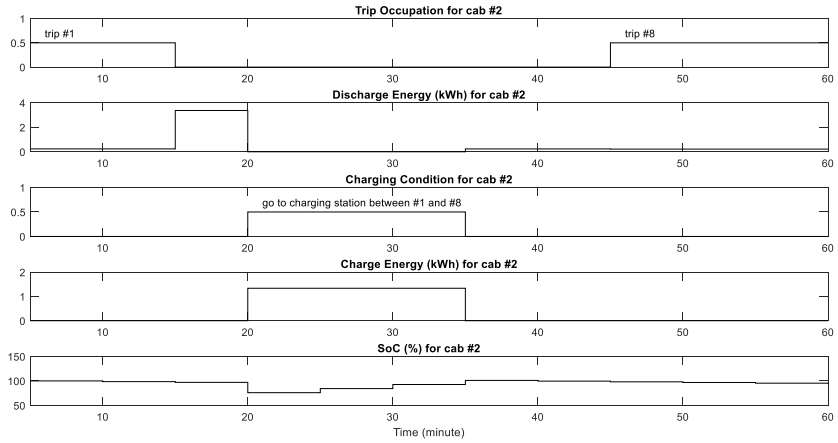

Figure 10 SoC, Discharge and Charge Amount over Time for Cab #2

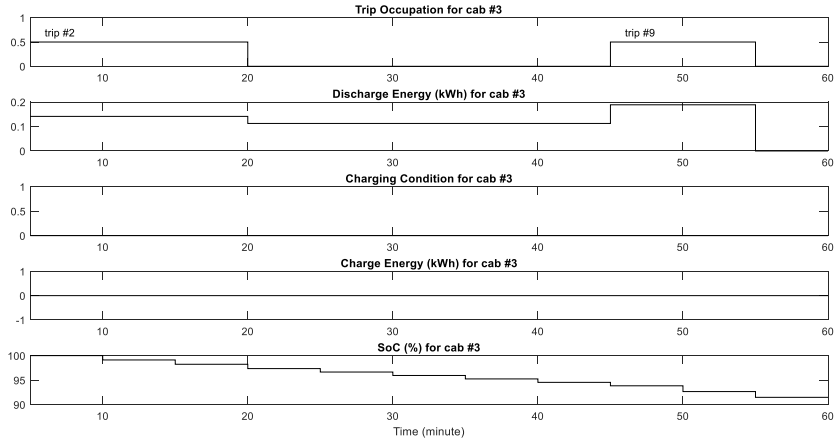

Figure 11 SoC, Discharge and Charge Amount over Time for Cab #3

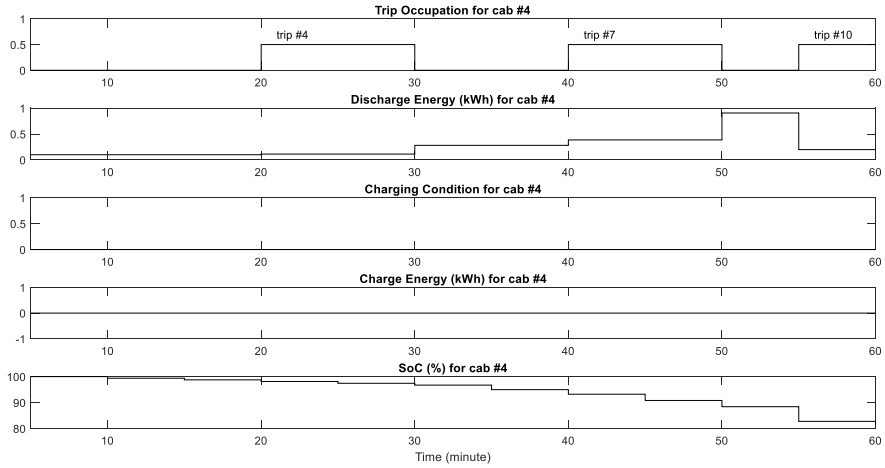

Figure 12 SoC, Discharge and Charge Amount over Time for Cab #4

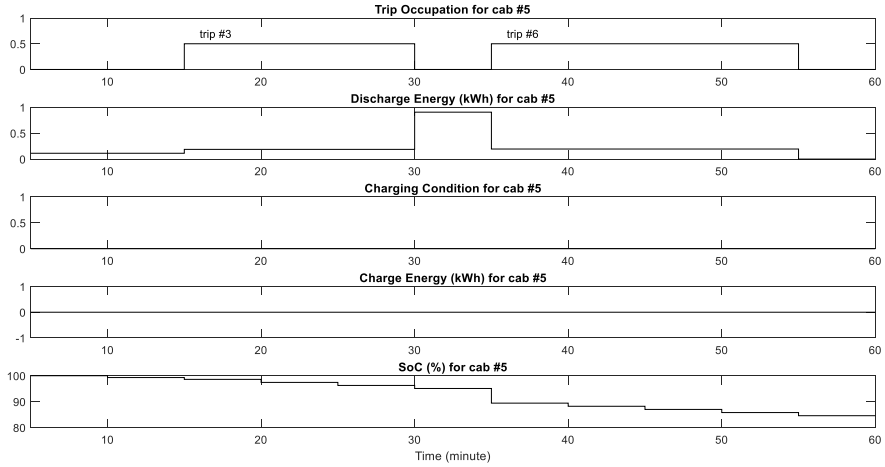

Figure 13 SoC, Discharge and Charge Amount over Time for Cab #5

Previously I modified the start and end time a little bit just to show that the optimization can cover all the conditions. The figure below shows the real start and end time for 5 cabs with 10 passengers in 1 hour. The trip information is shown in figure 14. The historical dispatch is shown in figure 15. The optimal dispatch is shown in figure 16.

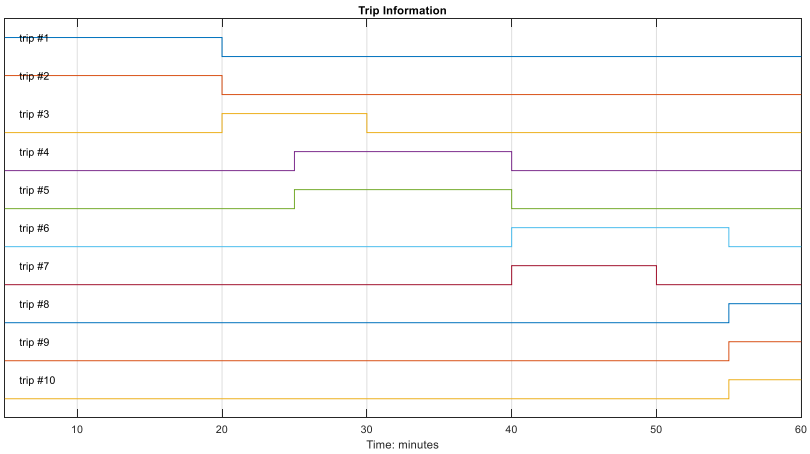

Figure 14 Historical Data Timeline

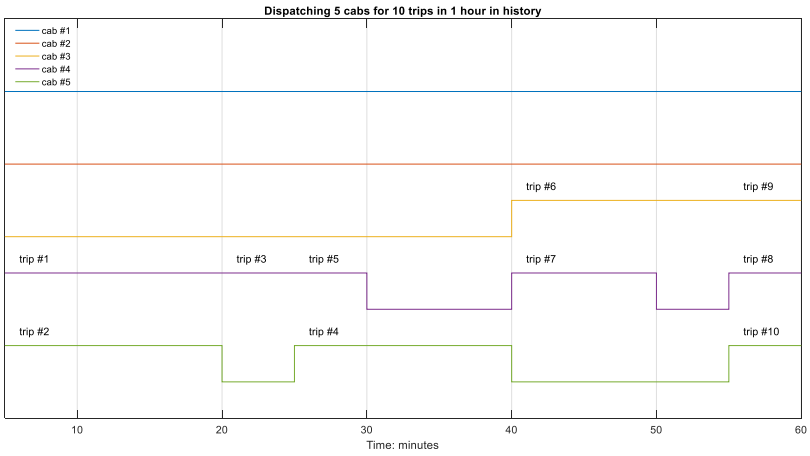

Figure 15 Historical Dispatch

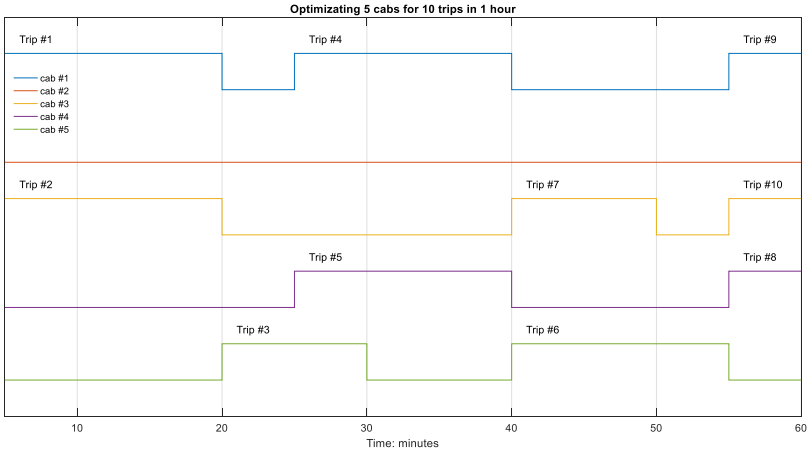

Figure 16 Optimal Dispatch

6.3 Simulation Results for Energy versus Cab

We already make sure that the optimization can dispatch cabs to pick up passengers in time and make cabs go to charging station in between if needed. We further want to know how much energy it can save compared to historical data. I hence used different number of cabs in 1 hour with their corresponding trips to see the effects.

Table 9 Number of Cabs and Their Corresponding Trips in 1 Hour

Number of Cabs	Number of Trips
4	7
8	13
12	16
16	21

The total energy consumed in 1 hour is described as below. The blue line is for historical data, and the red is for optimization.

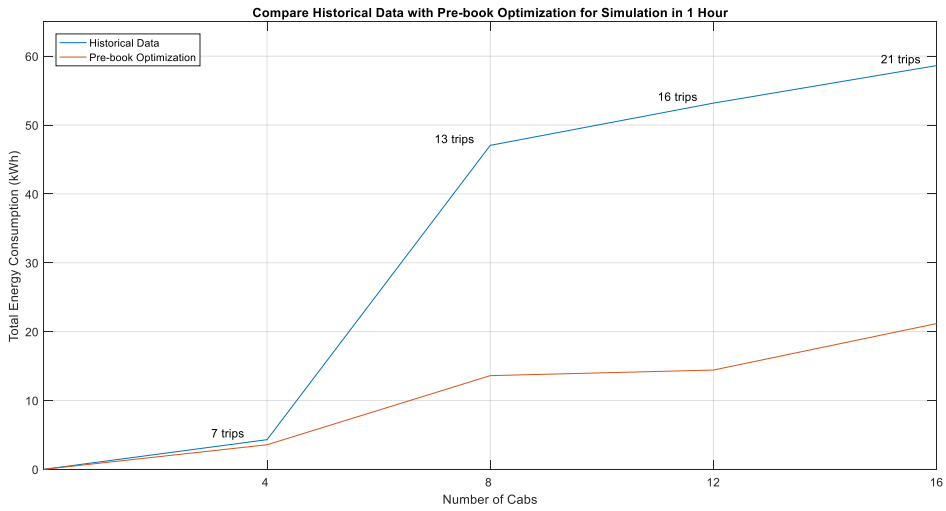

Figure 17 Energy versus Cabs for Pre-book Optimization

6.4 Simulation Results for Energy versus Time

In the previous section, we discuss the effect on energy with different number of cabs. In this section, I used the first 5 cabs in different time duration. The number of trips is generated by the 5 cabs in the time window.

Table 10 Time and Corresponding Trips for 5 cabs

Time	Number of Trips
30	5
60	10
90	14
120	15
150	17

The total energy is described as below. The blue line is for historical data, and the red is for optimization.

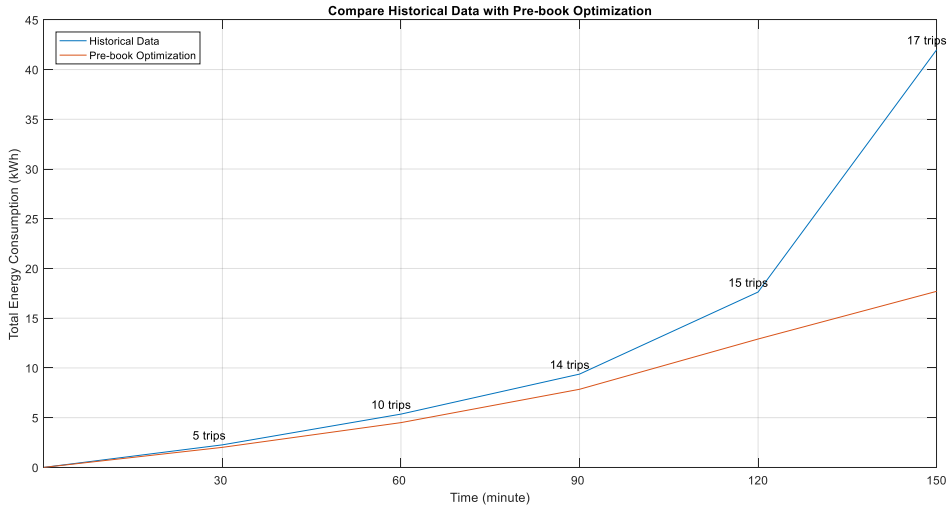

Figure 18 Energy versus Time for Pre-book Optimization

After that, I also plotted the electricity consumption versus total energy for pre-book optimization as below.

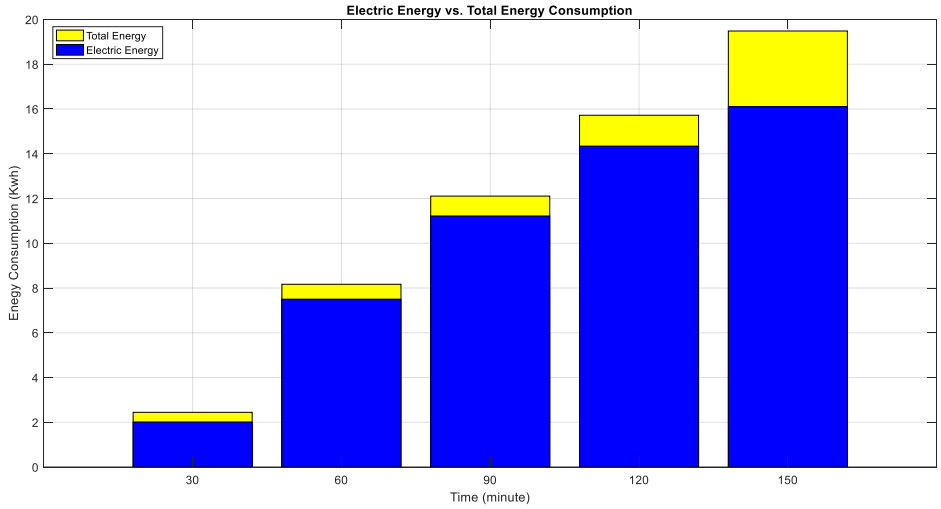

Figure 19 Electricity versus Total Energy for Pre-book Optimization

Chapter 7 Conclusions and Future Work

In this thesis, I propose two optimization methods to optimize the operation, including dispatching and charging decision, of autonomous and electric taxis, so as to maximize electricity usage while minimizing the total travel distance and serving all passengers.

First, real-time optimization is to optimize dispatch strategy. Second, pre-book optimization is to optimize both dispatch and charging strategy. I project for a certain time ahead, and consider customer demand, electricity and geographic distance in between. Charging decision is optimized between passenger requests.

The simulation results show that pre-book optimization reduces energy cost while serving all passengers in time.

For future work, one can further discuss how much energy is charged at the charging station instead of charging to 100%. Also, one can further combine the charging curve of SoC with that time.

On the other hand, I choose the closest charging station between two trips in time. One can further consider the waiting time at charging station, the floating charging price at different charging station, and what effect this might bring to the charging decision.

Reference

[1] http://www.techworld.com/picture-gallery/big-data/-companies-working-on-driverless-cars-3641537/

[2] https://deepblue.lib.umich.edu/handle/2027.42/108384

[3] https://www.google.com/maps/about/

[4] G. Hill, P. T. Blythe and C. Higgins, "Deviations in Markov chain modeled electric vehicle charging patterns from real world data," 2012 15th International IEEE Conference on Intelligent Transportation Systems, Anchorage, AK, 2012, pp. 1072-1077.

[5] http://www.nytimes.com/2007/05/23/nyregion/23taxi.html

[6] Ramteen Sioshansi, Riccardo Fagiani, Vincenzo Marano, “Cost and emissions impacts of Plug-In Hybrid Vehicles on the Ohio power system,” Energy Policy, vol. 38, issue 11, Nov. 2010.

[7] Hossein Akhavan-Hejazi, Hamed Mohsenian-Rad, “Developing a Test Data Set for Electric Vehicle Applications In Smart Grid Research,” Vehicular Technology Conference (VTC Fall), 2014 IEEE 80th.

[8] Mobashwir Khan, Kara M. Kockelman, “Predicting the market potential of plug-in electric vehicles using multiday GPS data,” Energy Policy, vol. 46, July 2012.

[9] Jarod C. Kelly, Jason S. MacDonald, Gregory A. Keoleian, “Time-dependent plug-in hybrid electric vehicle charging based on national driving patterns and demographics”, Applied Energy, vol. 94, June 2012.

[10] M. Rahmani-andebili and Haiying Shen, "Traffic and grid-based parking lot allocation for PEVs considering driver behavioral model," 2017 International Conference on Computing, Networking and Communications (ICNC), Silicon Valley, CA, USA, 2017, pp. 599-603.

[11] Sibi Krishnan K., Sunitha R. and P. Pathiyil, "Driver classification for Hybrid Electric Vehicles based on Fuel Consumption Index," 2016 International Conference on Computation of Power, Energy Information and Commuincation (ICCPEIC), Chennai, 2016, pp. 321-325.

[12] T. P. Dirienzo, N. A. Krishnan, Srija and J. R. Santos, "Effects of smart appliances on residential consumption patterns," 2014 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, 2014, pp. 188-192.

[13] K. T. Seow, N. H. Dang and D. H. Lee, "Towards An Automated Multiagent Taxi-Dispatch System," 2007 IEEE International Conference on Automation Science and Engineering, Scottsdale, AZ, 2007, pp. 1045-1050.

[14] K. T. Seow and D. H. Lee, "Performance of Multiagent Taxi Dispatch on Extended-Runtime Taxi Availability: A Simulation Study," in IEEE Transactions on Intelligent Transportation Systems, vol. 11, no. 1, pp. 231-236, March 2010.

[15] K. T. Seow, N. H. Dang and D. H. Lee, "A Collaborative Multiagent Taxi-Dispatch System," in IEEE Transactions on Automation Science and Engineering, vol. 7, no. 3, pp. 607-616, July 2010.

[16] F. Miao et al., "Taxi Dispatch With Real-Time Sensing Data in Metropolitan Areas: A Receding Horizon Control Approach," in IEEE Transactions on Automation Science and Engineering, vol. 13, no. 2, pp. 463-478, April 2016.

[17] https://en.wikipedia.org/wiki/Electric_car_use_by_country#United_States

[18] L. P. Fernandez, S. A. Enagas, T.G.S. Roman, R. Cossent, C. M. Domingo, P. Frias, “Assessment of the Impact of Plug-in Electric Vehicles on Distribution Networks”, IEEE Trans. on Power Systems, Vol. 26, No. 1, pp. 206-213, February 2011.

[19] Y. Huang and J. W. Powell, “Detecting regions of disequilibrium in taxi services under uncertainty,” in Proc. 20th Int. Conf. Adv. Geographic Inform. Syst., 2012, pp. 139–148.

[20] D.-H. Lee, R. Cheu, and S. Teo, “Taxi dispatch system based on current demands and real-time traffic conditions,” Transp. Res. Rec.: J. Transp. Res. Board, vol. 8, no. 1882, pp. 193–200, 2004.

[21] J. W. Powell, Y. Huang, F. Bastani, and M. Ji, “Towards reducing taxicab cruising time using spatio-temporal profitability maps,” in Proc. 12th Int. Conf. Adv. Spatial Temporal Databases, 2011, pp. 242–260.

[22] M. Qu, H. Zhu, J. Liu, G. Liu, and H. Xiong, “A cost-effective recommender system for taxi drivers,” in Proc. 20th ACM SIGKDD Int. Conf. KDD, 2014, pp. 45–54.

[23] K.-T. Seow, N. H. Dang, and D.-H. Lee, “A collaborative multiagent taxi-dispatch system,” IEEE Trans. Autom. Sci. Eng., vol. 7, no. 3, pp. 607–616, Jul. 2010.

[24] K. Clement-Nyns, E. Haesen, J. Driesen, “The Impact of Charging Plug-In Hybrid Electric Vehicles on a Residential Distribution Grid”, IEEE Trans. on Power Systems, Vol. 25, No. 1, pp. 371 - 380, December 2009.

[25] R. C. Green, L.Wang, M. Alam, “The Impact of Plug-in Hybrid Electric Vehicles on Distribution Networks: A Review and Outlook”, Renewable and Sustainable Energy Reviews, Vol. 15, No. 1, pp. 544-553, January 2011.

[26] S. Han, S. H. Han, K. Sezaki, “Development of an Optimal Vehicle-to-grid Aggregator for Frequency Regulation”, IEEE Trans. on Smart Grid, Vol. 1, No. 1, pp. 65-72, June 2010.

[27] P. Kumar, I. N. Kar : “Implementation of Vehicle to Grid Infrastructure Using Fuzzy Logic Controller”, in Proc. of IEEE Transportation Electrification Conference and Expo, Dearborn, MI, June 2012.

[28] Y. C. Ma, T. Houghton, A. J. Cruden, D. G. Infield, “Modeling the Benefits of Vehicle-to-grid Technology to a Power System”, IEEE Trans. on Power Systems, Vol. 27, No. 2, pp. 1012-1020, May 2012.

[29] Y. Ota, H. Taniguchi, T. Nakajima, K. M. Liyanage, J. Baba, A. Yokoyama “Autonomous Distributed V2G (Vehicle-to-grid) Satisfying Scheduled Charging”, IEEE Trans. on Smart Grid, Vol. 4, No. 1, pp. 559-564, March 2012.

[30] M. Singh, P. Kumar, I. N. Kar, “Coordination of Multi Charging Station for Electric Vehicles and its Utilization for Vehicle to Grid Scenario”, IEEE Trans. on Smart Grid, Vol. 4, No. 1, pp. 434-442, March 2012.

[31] C. Wu, H. Mohsenian-Rad, J. Huang, “Vehicle-to-Aggregator Interaction Game”, IEEE Trans. on Smart Grid, Vol. 4, No. 1, pp. 434-442, March 2012.

[32] M. C. Kisacikoglu, B. Ozpineci, L. M. Tolbert, “Examination of a PHEV Bidirectional Charger System for V2G Reactive Power Compensation”, in Proc. of the IEEE Applied Power Electronics Conference (APEC), Palm Springs, CA, Feb 2010.

[33] Y. Mitsukuri, R. Hara, H. Kita, E. Kamiya, N. Hiraiwa, E. Kogure, “Voltage Regulation in Distribution System Utilizing Electric Vehicles and Communication”, in Proc. of the IEEE T&D Conference, May 2012.

[34] C. Wu, H. Mohsenian-Rad, J. Huang, “PEV-based Reactive Power Compensation for Wind DG Units: A Stackelberg Game Approach”, in Proc. of IEEE Smart Grid Comm, Taiwan, Nov. 2012.

[35] C. Wu, H. Mohsenian-Rad, J. Huang, J. Jatskevich, “PEV-Based Combined Frequency and Voltage Regulation for Smart Grid”, in Proc. of IEEE Conference Innovative Smart Grid Technologies, Washington, DC, January 2012.

[36] C. Wu, H. Akhavan-Hejazi, H. Mohsenian-Rad, J. Huang, “PEV-based P-Q Control in Line Distribution Networks with High Requirement for Reactive Power Compensation”, in Proc. of the IEEE PES Conference on Innovative Smart Grid Technologies, Washington, DC, February 2014.

[37] J. L. Lu, M. Y. Yeh, Y. C. Hsu, S. N. Yang, C. H. Gan and M. S. Chen, "Operating electric taxi fleets: A new dispatching strategy with charging plans," 2012 IEEE International Electric Vehicle Conference, Greenville, SC, 2012, pp. 1-8.

[38] Jing Dong, zhenhon Lin,” Within-day recharge of plug-in hybrid electric vehicles: Energy impact of public charging infrastructure”, Transportation Research, Part D, 2012.

[39] Jarod C. Kelly, Jason S. MacDonald, Gregory A. Keoleian, “Time-dependent plug-in hybrid electric vehicle charging based on national driving patterns and demographics”, Applied Energy, vol. 94, June 2012.

[40] Tae-Kyung Lee, et al,” Stochastic Modeling for Studies of Real-World PHEV Usage: Driving Schedule and Daily Temporal Distribution”, IEEE Transactions on vehicular technology, vol. 61, No.4, May 2012.