5-2020

Learning & Planning for Self-Driving Ride-Hailing Fleets

Jack Morris

Learning & Planning for Self-Driving Ride-Hailing Fleets

A thesis submitted in partial fulfillment of the requirement
for the degree of Bachelor of Science in the Department of Mathematics from
The College of William and Mary

by

Jack Morris

Accepted for __Honors__________________________________
(Honors/Not Honors)

____
Anh T. Ninh, Director

____
Daniel A. Cristol

____
Anke van Zuylen

Williamsburg, VA
May 4, 2020

Learning & Planning for Self-Driving Ride-Hailing Fleets

Jack Morris

Department of Mathematics

William & Mary

Williamsburg, VA 23187-8795, USA

Email: jfmorris@email.wm.edu

Acknowledgements

I would like to thank my advisor, Professor Anh Ninh, for his guidance, our many Skype and Zoom calls, and for letting me follow my own creative direction in this project. He taught me everything I know about reinforcement learning which was the inspiration and foundation of this project. Thanks also to Professor Anke van Zuylen who, after leading me to pursue operations research with her incredible teaching, taught me all about the network algorithms at work in this project. Lastly, thanks to Professor Dan Cristol, Kim Van Deusen, and the 1693 Scholars Program for their unbounded academic and personal support.

Abstract

Through simulation, we demonstrate that incorporation of self-driving vehicles into ride-hailing fleets can greatly improve urban mobility. After modeling existing driver-rider matching algorithms including Uber's Batched Matching and Didi Chuxing's Learning and Planning approach, we develop a novel algorithm adapting the latter to a fleet of *Autos* – self-driving ride-hailing vehicles – and *Garages* – specialized hubs for storage and refueling. By compiling driver-rider matching, idling, storage, refueling, and redistribution decisions in one unifying framework, we enable a system-wide optimization approach for self-driving ride-hailing previously unseen in the literature. In contrast with existing literature that labeled driverless taxis as economically infeasible, we found that substituting Autos for conventionally driven vehicles stands to increase platform earnings between 90.4% and 99.0% even while bearing the cost of vehicle financing, licensing, maintenance, cleaning, fuel, and oversight previously paid by contracted drivers. Along with increased earnings, the substitution can lower pickup times, improve match rates, and decrease emissions. By adjusting parameters, it is possible to incentivize matching decisions that lower traffic congestion or street parking usage. Our sensitivity analysis indicates that these results are resilient to changing circumstances including high gas prices and policy regulations. We conclude by stating avenues for further improving the model and recommending that city governments take a proactive role in self-driving ride-hailing transitions in order to capitalize on the benefits of the technology while effectively mitigating its harms.

LEARNING & PLANNING FOR SELF-DRIVING RIDE-HAILING FLEETS

William & Mary Department of Mathematics

JACK MORRIS

Contents

List of Figures

List of Tables

1 Background

1.1 *Urbanization*

The world has undergone rapid urbanization in the last century. In 2007, the world's urban population surpassed its rural population for the first time in history. In 1950, less than one third of people worldwide lived in urban settlements, and by 2050, that proportion is estimated to reach two thirds. The United States is more urbanized than the world average with 84% of its inhabitants living in urban developments in 2020.[7]

According to the UN Department of Economic and Social Affairs, *urbanization* consists of two main components: (1) **the conversion of rural land into urban space**, and (2) **population redistribution from rural areas to urban areas**. Urbanization is fueled with public and private investment and molded by urban planning. By redefining the ways people live and work across space, it transforms our demographics and social structures in both rural and urban areas.[7] By concentrating people and infrastructure, urbanization also concentrates innovation and commerce. Cities become regional hubs for transportation, trade, and information where services are both more accessible and higher quality compared to surrounding rural areas.[7]

Increased urbanization poses a challenge in managing the growth and meeting the needs of an ever-increasing number of people, but it also presents an opportunity for sustainable development. With proper planning, cities can leverage this period of growth and concentration of people, infrastructure, commerce, and innovation to satisfy human needs more efficiently at never-before-seen scales.[7]

1.2 *Urban Mobility*

Municipalities face an explosion of mobility options: Transportation Network Companies (TNCs) like Uber and Lyft, "microtransit" companies like Via and Chariot, bike sharing services like Capital Bikeshare in Washington, DC or Citi Bike in New York City, and scooter sharing services like Bird and Lime. [8, 9]

City governments struggle to determine how well new services align with city goals. Do they move people around safely, equitably, and sustainably? Will cruising contractors congest city streets or will dockless scooters litter the sidewalks? Will new mobility ridership take from personal vehicle transit or public transit?[10, 1]

A report by the National Association of City Transportation Officials found that station-based and dockless bike share and scooter share accounted for 84 million rides in 2018, double that of the previous year. They found that station-based bike sharing occured more often during the week whereas scooters were used more often on weekends. As seen in Figure 1, annual bike share members' usage peaked during morning and evening rush hours where casual riders' usage was more spread throughout the day. More than 60% of riders use bike share to connect to transit compared to 28% for scooters.[1]

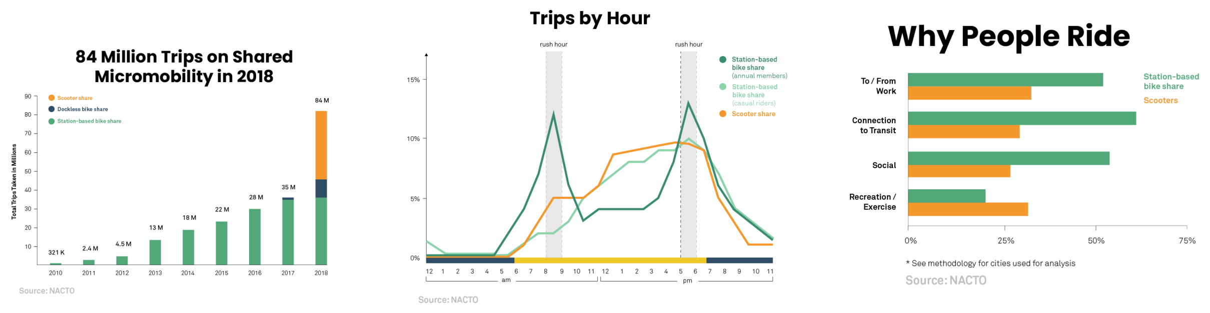

Figure 1: The urban micromobility explosion has helped to decongest rush hour traffic and connect riders with public transit.[1]

Micromobility usage reasoning is analyzed to determine impact on commute. In the United States, the commute has been relatively unchanged since 2014 with roughly 76% driving to work alone and 9% carpooling. Given that roughly 5% work from home, that leaves only 10% for public transit, biking, and walking.[11] City planners want to maximize public transit ridership for its benefits in sustainability, efficiency, and congestion, but commuters are inclined to drive for personal freedom, comfort, and door-to-door connection. Planners want new mobility to solve the "first mile/last mile," connecting commuters from their front door to public transit and from public transit to the workplace.[10]

By cutting into the 76% of single drivers and not the 10% taking public transit, walking, and biking, new mobility services could make the commute much more sustainable. As seen in Figure 2, the United States transportation sector accounts for 29% of greenhouse gas emissions, more than any other sector. More than half of that portion is emitted by light-duty vehicles including personal vehicles and taxis.[12]

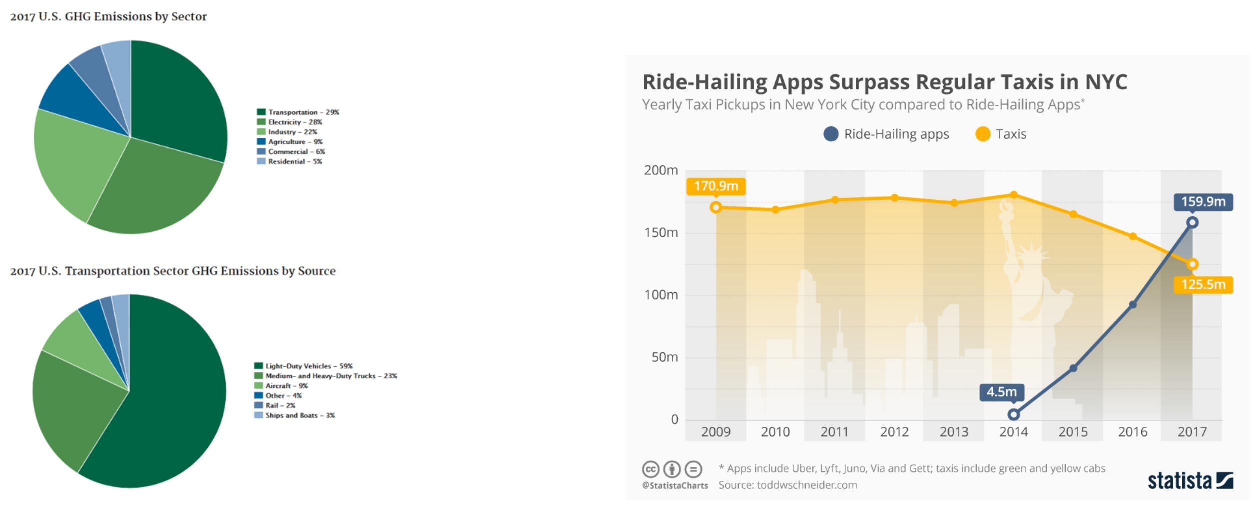

Figure 2: US Carbon Emissions by Sector and TNC and Taxi Ridership in New York City since 2009

1.3 *Ride-Hailing*

For the purposes of this project, we define ride-hailing - related to or often referred to as ridesharing, ridesourcing, or ridematching - as a service that arranges trips for riders by connecting them with drivers at an estimated price by means of a smartphone app. Since 2013 when the California Public Utilities Commission took the first steps in regulating ride-hailing services, local governments have taken to grouping ride-hailing platforms under the title of "Transportation Network Companies", or TNCs.[13]

In the United States, TNCs including Uber and Lyft connected 2.61 billion passengers to their destination in 2017, 37% more than the previous year and double that of 2012.[10] For context, bus ridership was at 4.63 billion passengers and declining that same year.[14]

Hailing rides is not new, but GPS and smartphone ubiquity has driven the recent surge. Driver and passenger ratings build trust, route planning decreases waiting time and increases vehicle utilization rates, automatic payment saves time and effort, and dynamic pricing matches supply and demand more efficiently. Platforms offer consumers options for pooled rides, luxury rides, rides for pets, and more. All of these factors work towards a significantly improved passenger experience.[15]

Drivers laud the platforms for their flexibility, allowing them to work at their leisure. Some see it as a supplementary income stream, and others use it to ease transitions between jobs.[15]

TNCs are dependent on large, dense urban environments. In the United States, 70% of Uber and Lyft rides are concentrated in just nine cities - Boston, Chicago, Los Angeles, Miami, New York, Philadelphia, San Francisco, Seattle, and Washington DC. Cities are also lean on TNCs to serve transit demand. In eight of these cities (excluding New York), TNC trips are 9 times the number of traditional taxi trips. However in rural and suburban areas, traditional taxis still account for more demand than TNCs.[10]

Experts disagree on how ride-hailing will impact carbon emissions due to counteracting factors. Higher utilization rates compared to taxis and personal vehicles decreases the emissions per vehicle mile by individual passenger, but the lower cost of transportation may increase demand and thereby induce more vehicle miles traveled in total. Likewise with congestion, a large portion of traffic in many big cities is attributed to vehicles seeking out parking spots. Ride-hailing eliminates this need and therefore decreases congestion. However because of the increase in usage, congestion will increase, especially so if the usage increase steals from public transit as opposed to personal vehicles or taxis.[15]

Schaller Consulting in New York reported in 2018 that "about 60 percent of TNC users in large, dense cities would have taken public transportation, walked, biked or not made the trip if TNCs had not been available for the trip", indicating that ride-hailing likely takes more demand from more sustainable and less congesting transit modes.[10] They also found that "shared ride services such as UberPOOL, Uber Express POOL and Lyft Shared Rides, while touted as reducing traffic, in fact add mileage to city streets" and therefore do not improve net congestion.[10]

One study (funded by Uber) compared UberX and yellow cab service in various Los Angeles neighborhoods. They found that the average UberX ride was $6.40 compared to $14.63 for yellow cabs and that UberX passengers waited 6 minutes and 49 seconds for their ride compared to 17 minutes and 42 seconds for the cab on average. Notably, the differences were the greatest in Los Angeles' poorer neighborhoods. If this result is representative of American cities and ride-hailing platforms, it is a strong argument for ride-hailing in urban mobility. Brookings reports that 75% of low- and middle-skilled jobs necessitate more than 90 minutes of public transport per passenger per day, so this result would demonstrate increased economic opportunity in disadvantaged communities.[15]

This is not to say that ride-hailing is the image of equity. Studies in Boston and Seattle found that cancellations were much more frequent among riders with African American-sounding names compared to white-sounding names and that women pay more for rides than men. However, it is still unknown how these outcomes compare to those of traditional taxi services.[16]

Not everyone shares the benefits of ride-hailing evenly. In 2018, individuals "with a bachelor's degree, over $50,000 in household income, and age 25 to 34 [used] TNCs at least twice or even three times as often as less affluent, less educated and older persons."[10]. These results may point to ride-hailing as a driver of division, but this alarm may be too early sounded. There is some historical precedent, notably among smartphones and personal computers, that earlier adopters of technological trends are often more affluent and more highly educated. It remains possible that the distribution of ride-hailing ridership will become more equitable over time.[15]

With unmet promises in sustainability, efficiency, and equity, city planners remain uncertain how ride-hailing aligns with the goals of our cities. One hope is that self-driving vehicles can change the equation.

1.4 *Self-Driving Vehicles*

In 2015, The Guardian promised that "From 2020, you will be a permanent backseat driver".[17] In 2016, Business Insider reported that "10 million self-driving cars will be on the road by 2020".[18] In 2019, Tesla CEO Elon Musk declared full self-driving capability "by the end of the year".[19] Now in 2020, where are the self-driving cars?

The principle of the problem is not too complicated. After equipping the cameras and equipment that allow the car to "see", feed it training data (driving footage) until it learns to track surrounding objects and respond correctly to events. In essence, this is just like any other deep reinforcement learning problem like AlphaGo or AlphaStar. Even the simpler driving tasks can be difficult, but some events of utmost importance - particularly traffic collisions - are relatively rare. Projects like Alphabet's Waymo resort to engineering the more dangerous and sparse situations to bolster the training data.[20]

The progress of self-driving projects are measured in total miles traveled and in the number of disengagements - occurrences where a human driver had to take control when the self-driving computer could not respond effectively to a situation. Leading in those categories is Waymo with 20 million miles driven without any fatal incidents and only 0.09 disengagements per 1000 miles. However, given that human driving typically results in one traffic fatality every 100 million miles, even Waymo has a long way to go before demonstrating the supposed significant benefits to safety of self-driving technology.[20]

Self-driving progress is also measured on a 0-5 scale - seen in Figure 3 - where Level 0 corresponds to no automation whatsoever and Level 5 corresponds to full driving capability without any human occupants in all conditions.[19] The state of self-driving vehicles today might exist between Levels 3 and 4, but if we pass Level 4, there are innumerable potential impacts to transit, both for better or for worse.

High-level self-driving vehicles primarily offer improved road safety. In the United States, 4 out of 10 unintentional deaths are from traffic collisions. By switching out fallible humans for

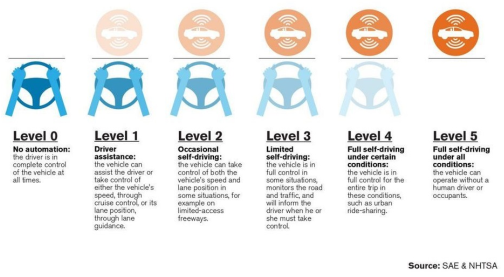
Level 0
No automation: the driver is in complete control of the vehicle at all times.
Level 1
Driver assistance: the vehicle can assist the driver or take control of either the vehicle's speed, through cruise control, or its lane position, through lane guidance.
Level 2
Occasional self-driving: the vehicle can take control of both the vehicle's speed and lane position in some situations, for example on limited-access freeways.
Level 3
Limited self-driving: the vehicle is in full control in some situations, monitors the road and traffic, and will inform the driver when he or she must take control.
Level 4
Full self-driving under certain conditions: the vehicle is in full control for the entire trip in these conditions, such as urban ride-sharing.
Level 5
Full self-driving under all conditions: the vehicle can operate without a human driver or occupants.
Source: SAE & NHTSA

Figure 3: The Five Levels of Vehicle Autonomy

computers that never get tired, never drive under the influence of alcohol or drugs, and never get distracted or lose focus, the United States stands to reclaim up to 40,000 lives per year. Secondly, due to "eco-driving" - highly optimized operating efficiency impossible for human drivers - self-driving vehicles stand to save up to 20% in emissions working to reach our climate goals and improve air quality in dense urban areas.[21][22] Made economically competitive by lowering costs and transferring earnings from drivers to the platform, employing self-driving vehicles in ride-hailing fleets may reduce congestion and lower the price of travel.[15]

Others argue that without intervening policy, self-driving technology will only compound present trends with "more traffic, less transit, and less equity and environmental sustainability" and that employing it in ride-hailing fleets does not improve its cause. Uber and Lyft remain low-density transit options. Enabling these platforms with lower costs and less accountability - even with pooled options - only stands to congest our streets even more. They argue that these services will only pull passengers from high density and more efficient options and not from their personal vehicles. Steering self-driving policy to reduce personal automobile use, complement existing high-density transit options, and ensure that fleet vehicles use public space efficiently will help mitigate harms.[10]

In our Background, we discussed (1) the growing importance of urban mobility, (2) how ride-hailing may provide it but with some questions to its effectiveness and equity, and (3) how self-driving vehicles might help resolve the problems with conventional ride-hailing. Thus, our research objective is to investigate the feasibility of self-driving ride-hailing fleets as they relate to urban mobility. The remainder of the thesis is as follows: Section 2 briefly reviews related literature. Section 3 presents the problem and rationalizes necessary assumptions. Section 4 discusses the development of our model. Section 5 conducts extensive numerical analyses of costs, sensitivity, and error in the model. Section 6 concludes the thesis, states the strengths and weaknesses of our contribution, proposes policy recommendations, and discusses future research considerations.

2 Related Literature

2.1 *Uber: Batched Matching*

Uber answers the question "How does Uber match riders with drivers?" on their Marketplace website. Originally, the company paired passengers with the closest vehicle immediately. They learned through practice that the closest is not always the quickest, and the quickest for you is likely not the quickest for everyone.

Uber works under the principle that the best results occur with the shortest pickup times: "Riders don't like waiting. Drivers earn more when there is less down time between rides."[2]

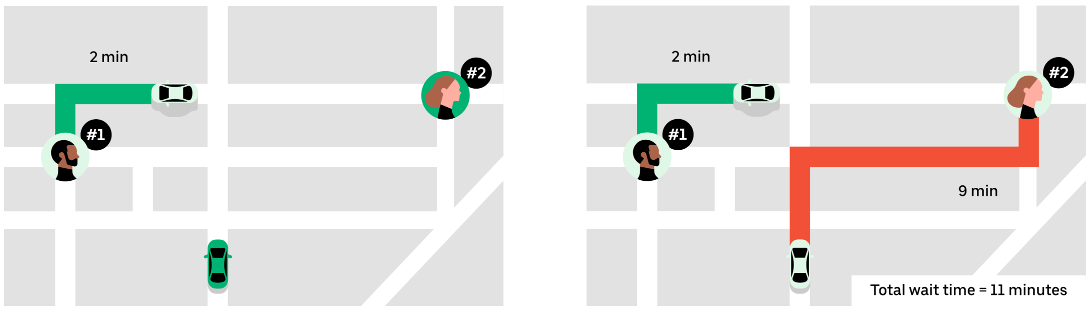

Figure 4: Uber matches two riders in sequence to the closest available Driver — 11 minutes waiting.[2]

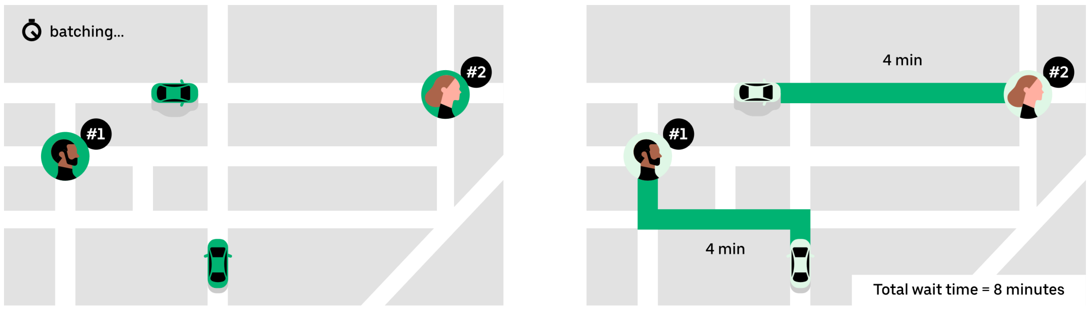

Figure 5: Uber waits briefly before matching two riders to drivers simultaneously — 8 minutes waiting.[2]

When immediately matching to the closest driver, Uber notes the frequent occurrence of unnecessarily long pickup times as visualized in Figure 4. Had the platform waited the briefest moment after rider #1 requested a ride, there would exist a significantly better pairing in terms of total waiting. Thus, Uber coined *Batched Matching*.

As visualized in Figure 5, Uber waits a moment, gathers available drivers and riders, and then settles them all in one *batch*. This waiting and batching step allows for pickup time improvements in excess of the added waiting time.[2]

2.2 *Didi: Learning & Planning*

Xu et al. from AI Labs at Didi Chuxing presented a new matching algorithm for ride-hailing platforms.[3] They witness that traditional algorithms such as Uber's Batched Matching focus too heavily on *immediate* customer satisfaction. They argue that resources may be allocated more efficiently and customers may be more satisfied by defining the goal more broadly, using driver-rider matching to position demand to meet future supply imbalances.

The authors solve this sequential order dispatch decision problem by *learning* and *planning* using a centralized algorithm that coordinates many driver agents. The Learning Step aims to understand spatiotemporal distribution of demand and supply in the ride-hailing environment which the Planning Step uses to make decisions throughout the ride-hailing process. In particular,

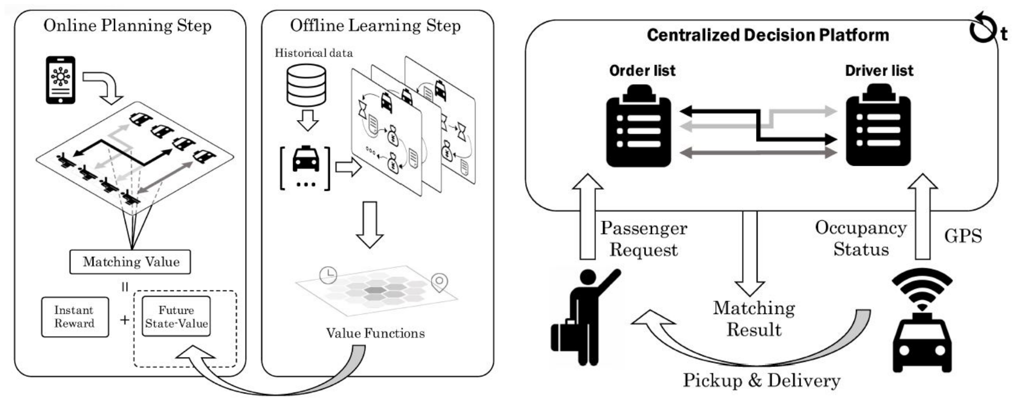

Figure 6: Learning and Planning Steps for Didi's order dispatch algorithm and the architecture of the ride-hailing centralized decision platform[3]

in the Learning Step, the authors use historical data to "spatiotemporally quantize" states - i.e., to determine the value of a vehicle being in a specific place at a specific time. Then, conducted in real-time, the Planning Step matches drivers to riders to maximize both immediate and expected future rewards estimated using the spatiotemporal values found in the Learning Step.

2.2.1 *The Learning Step*

The Learning Step is built upon a Markov Decision Process (MDP) where each individual driver is an agent. In the MDP, an *agent* lives and behaves in an *environment*. Based on its *state*, the agent takes *actions* that result in a *reward* and impact the agent's state. By observing the sequence of state-action pairs, they evaluate the value of states and actions.[3]

Each agent's *state* $s = (g, t)$ is a state-time pair where $g \in G$ is the driver's location index and $t \in T$ is the time index. Intuitively, a vehicle in Midtown Manhattan at 5:00pm is not in the same state as another in Midtown at 5:00am or yet another on Staten Island even at 5:00pm.

An agent can take two main kinds of *actions* a. By idling, the agent can remain in the same place until the next timestep, not earning but also not using fuel. By accepting an order, the agent will accept the fare to drive to the pickup location and take the rider to their destination. This is visualized in the left graphic in Figure 7.

The *reward* is the optimization goal of the algorithm. By taking actions, the agent seeks to maximize its reward. In this environment, reward is defined by ride fares and per-mile costs.

Once historical data has been collected and framed in terms of state-action-reward-next_state transitions (s, a, r, s'), they iterate through the transitions using *Policy Evaluation* to determine the value of each state $V(s)$.[3]

2.2.2 *The Planning Step*

Like Uber's Batched Matching, the Planning Step gathers available drivers and riders over a set batch time, matches them all simultaneously, and dispatches orders. What primarily sets it apart is its objective. Instead of minimizing immediate pickup times, Didi's Planning Step maximizes future profits.[3]

As seen in the right graphic of Figure 7, the algorithm first constructs a bipartite graph with available actions on one side and available agents on other. The action side includes all pending

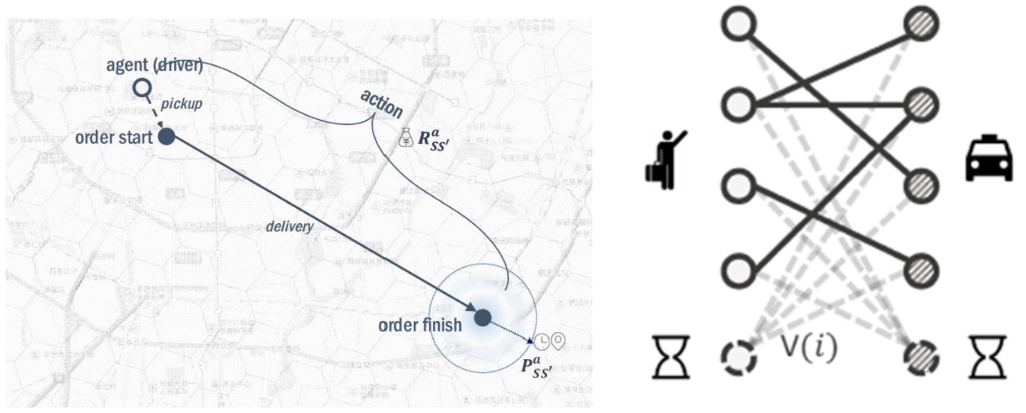

Figure 7: LEFT: Driver agent in state s accepts a ride request a to transition into new state s' and earn a reward $R^{a}_{s,s'}$. RIGHT: Bipartite graph of available actions and agents. [3]

ride requests as well as the option for an agent to idle in their current state. The agent side includes all agents currently without an assigned order as well as the option for ride requests to go unmatched and carry over into the next iteration. Using the state values $V(s)$ obtained in the Learning Step, the algorithm draws edges from each action to each agent weighted by their immediate and expected future reward. Using the Kuhn-Munkres algorithm to find the matching between the action and agent sides that maximizes edge weights, Didi determines driver-rider pairings that optimize not only immediate reward but future gain as well.[3]

The article also includes an alternate explanation for their process by framing the problem in the reinforcement learning context. The one centralized agent learns how to maximize gain in the sequential decision-making problem by combining the two steps - iteratively using the Learning Step to update the policy and the Planning Step to take actions according to the policy.

2.3 *"Autonomous Vehicles and Public Health: High Cost or High Opportunity Cost?"*

Authors Ashley Nunes and Kristen D. Hernandez of the MIT Energy Initiative released their study in April 2019 concerning self-driving vehicles serving as taxis. They observe the dangers of passenger vehicles on public health - notably road safety and air quality - and hypothesize that driverless taxis may help mitigate these dangers. Before this practice can proliferate, it must be economically feasible.[21]

They explore the economic feasibility of driverless taxis by conducting an in-depth cost analysis on operating self-driving vehicles in the San Francisco market. They analyze the costs of financing, licensing, insuring, maintaining, cleaning, fueling, and overseeing self-driving taxis and find that the high operating costs prohibit profitability and therefore the desired public health outcomes. They claim that only by considering additional opportunity costs can self-driving taxis be seen as economically competitive.[21] We replicate Nunes & Hernandez's analyses in our Cost Analysis section below, tailoring them to the Chicago market.

3 Problem Description

We aim to determine how incorporating self-driving vehicles into ride-hailing fleets can impact urban mobility by simulating incorporation in two existing driver-rider pairing frameworks: Uber's Batched Matching and Didi Chuxing's Learning and Planning. After performing cost analyses to evaluate cost-effectiveness of self-driving ride-hailing, we compare self-driving options with conventionally driven counterparts to determine potential benefits and harms to urban mobility.

We capture urban mobility with key metrics including pickup times, platform earnings, and vehicle miles traveled (VMT) which reflect passenger satisfaction, cost-effectiveness, and sustainability.

3.1 *Assumptions & Rationales*

1. **Self-driving vehicles are safe and functional.** Although it is unclear when this threshold will be met, we assume for the sake of the study that self-driving vehicles can fully operate without any human occupants in at least some environments. Given that we limit our study to urban regions, it would be sufficient that a vehicle is trained to be fully functional within one specific city.

2. **Autos drive like conventional human drivers.** For the tractability of our study, we assume self-driving ride-hailing fleet vehicles, or *Autos* as we will refer to them hereon, are essentially conventionally driven vehicles without the human driver. Route planning does not improve. We do not consider inter-Auto communication that alleviate traffic.

3. **Rides both begin and end within the city limits.** Matching supply with demand in sparser environments becomes increasingly difficult. While other studies may consider suburban and rural areas, we choose to limit our scope to urban regions.

4. **Riders may only match with available drivers and Autos.** While existing ride-hailing platforms frequently match riders to drivers who already have passengers, we choose to neglect this option to decrease computational complexity.

5. **We consider only one type of ride request.** While Uber has UberX, UberXL, UberSELECT, UberBLACK, UberSUV, and more to meet varying need of their customers, we consider just one ride type for simplicity.[23]

6. **All rides begin and end in the center of a *Community Area*.** To reduce computational complexity to a feasible level, we chose to partition the geography of simulated cities into discrete zones which we refer to as Community Areas. Because the data we input into the model is not sufficiently granular, beginning and ending rides in the centers of these Community Areas is the best achievable option.

7. **Riders renege if they aren't assigned a rider in two timesteps.** Many riders will not persevere in requesting a ride after several refusals by an insufficient platform. We cannot gauge the behavior of particular riders given only the time and place of their ride request, so we chose this simple adjustment to accommodate reneging after failed matching attempts.

4 Model Development

Symbol	Explanation
$x_{ij} \in C_X$	Distance between Community Areas i and j
$t_{ij} \in C_T$	Time between Community Areas i and j
$d_i \in D$	Available Drivers in Community Area i
$r_i \in R$	Active Ride Requests in Community Area i
$d_{it} \in D_A$	Drivers Arriving in Community Area i in t Timesteps
τ	Timestep Length, Batch Time
w_{ij}	Pickup Time between Driver i to Rider j
G_t	Gain
$V(s)$	State-Value Function
$s \in S$	State
$g \in G$	State's Spatial Index by Community Area
$t \in T$	State's Time Index
$a \in A$	Action
$f(i,j)$	Fare Function
p_0, p_{min}	Base Fare, Minimum Fare
p_x, p_t	Fare per Mile, Fare per Minute
c_m	Cost per Mile
$R(i,j,k)$	Undiscounted Reward Function
$r = R^a_\gamma$	Discounted Reward for Taking Action a
γ	Discount Factor
α	Learning Rate defined by state counter $N(s)$
A_{ij}	Advantage Function from Agent i to Order j
a_{ij}	Decision to Match Driver i to Rider j
$occ(i)$	Occupancy of Garage i
$cap(i)$	Capacity of Garage i
$in(i)$	Number of Vehicles Dispatched to Garage i
$\mathcal{S}$, $\mathcal{D}$	Supply and Demand Sides of Bipartition
$G_\mathcal{S}$, $G_\mathcal{D}$	Garage Set

Table 1: Model Notation

4.1 *Simulated Environment*

Before we started pairing riders with drivers, we needed to build the environment. We took a city and partitioned it into discrete zones that we will henceforth refer to as *community areas*. The smaller these community areas are, the more precise your results will be, but it will also be much more computationally costly. We decided to divide urban areas by existing divisions. For instance, Chicago has 77 community areas from Rogers Park in the north to Hegewisch in the south.

Whenever an environment is built, there are trade-offs between complexity and cost. In order to address our research question with reasonable computational cost, we avoided route planning.

Even with our historical data, it is impossible to say the exact state of traffic at the given moment, especially if we are altering the state of every ride-hailing vehicle in the city. So instead of devising the exact routes in the moment for every possible order dispatch, we used a Google Maps Directions API to determine the distance and time between any two community areas. Therefore, we can represent the urban environment with n community areas by distance and time matrices C_D and C_T:

$$C_D = \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{1,2} & \ddots & & \\ \vdots & & \ddots & \\ x_{n,1} & & & x_{n,n} \end{bmatrix}, C_T = \begin{bmatrix} t_{1,1} & t_{1,2} & \dots & t_{1,n} \\ t_{1,2} & \ddots & & \\ \vdots & & \ddots & \\ t_{n,1} & & & t_{n,n} \end{bmatrix},$$

where $d_{i,j}$ represents the driven distance from community area i to community area j, and $t_{i,j}$ represents the driving time from community area i to community area j. Traffic differences can be approximately accounted for by varying between *optimistic* and *pessimistic* traffic conditions in Google Maps queries [24].

Then with an interconnected environment, we populated it. Until we had a simulator with all its moving pieces in working order, the initial conditions we set for riders and drivers matter less. Once we used the environment to simulate ride-hailing based off of our Chicago taxi data, we adapted these conditions to match. Until then, we can represent the available driver distribution using an array of length n:

$$D = \begin{bmatrix} d_1 & d_2 & \dots & d_n \end{bmatrix},$$

where d_i is the number of available vehicles in community area i. We can establish these initial driver conditions either manually or randomly.

Then with drivers to service demand, we populated the environment with ride requests. Given the likelihoods of requests beginning and ending in particular community areas, we can randomly generate demand where each ride request is represented by an ordered pair of pickup location and destination by community area. Thus, we can represent the active ride request distribution with an array of length n:

$$R = \begin{bmatrix} r_1 & r_2 & \dots & r_n \end{bmatrix},$$

where r_i is the number of active ride requests with pickup locations in community area i.

Lastly, before we can start pairing our supply and demand, we need to allow for the environment to evolve through time. We chose to model the problem with discrete timesteps. We started with a timestep length of $\tau = 30$ seconds. So if a ride is predicted to take 7 minutes, it will take 14 timesteps. Thus, we can construct an array of "arriving drivers":

$$D_A = \begin{bmatrix} d_{1,1} & d_{2,1} & \dots & d_{n,1} \\ d_{1,2} & d_{2,2} & \dots & d_{n,2} \\ \vdots & \vdots & \dots & \vdots \\ d_{1,t} & d_{2,t} & \dots & d_{n,t} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix},$$

where d_{it} represents the number of drivers that will arrive in community area i in t time steps.

To give an example, consider John Doe, a University of Chicago student who has just gotten out of his last exam for the semester. He packs his bags, steps outside his dorm, takes out his smartphone, and submits a ride request to take him to the airport. Our algorithm pairs him with the closest available driver, 3 minutes away in Woodlawn. It will take the driver 3 minutes ($t_{41,42}$ = 6 timesteps) to get from Woodlawn (C.A. 41) to the pickup location in Hyde Park (C.A. 42), and then it will take another 40 minutes ($t_{42,76}$ = 80 timesteps) to get to O'Hare International Airport (C.A. 76). To reflect this pairing in the model, we subtract one from the number of available drivers in Woodlawn d_{41}, we subtract one from the number of active ride requests in Hyde Park r_{42}, and we add one to the number of drivers arriving at O'Hare 86 timesteps from now $d_{76,86}$.

After we make all the pairings in one timestep, we "dispatch" all the paired drivers from D, remove paired riders from the list of active requests R, and we transition into the next timestep. In this transition, all of the drivers arriving in that timestep $d_{i,1}$ "arrive", so we add them to the number of available drivers d_i. We effectively pop the top row off the arriving drivers matrix D_A, add it to the array of available drivers D, and append a new array of zeros to the bottom of the arriving drivers matrix. Lastly, new rides are requested, and the process starts over again.

$$D = \begin{bmatrix} d_1 & d_2 & \dots & d_n \end{bmatrix} \leftarrow \qquad D_A = \begin{bmatrix} \cancel{d_{1,1}} & \cancel{d_{2,1}} & \cancel{\dots} & \cancel{d_{n,1}} \\ d_{1,2} & d_{2,2} & \dots & d_{n,2} \\ \vdots & \vdots & \dots & \vdots \\ d_{1,t} & d_{2,t} & \dots & d_{n,t} \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix} \leftarrow \begin{bmatrix} 0 & 0 & \dots & 0 \end{bmatrix}$$

With an interconnected city environment, ride demand, vehicles to service that demand, and an simulator that evolves over time to capture the supply and demand, we started matching drivers to riders.

While we got the simulator in working order, we implemented the simplest matching algorithm first - match each ride request to the closest driver. In later algorithms, the timestep length τ - used as the "batch time" - means nothing for this algorithm, yet ride requests still arrive at discrete intervals. To consider each one individually, we select one rider at a time at random under the assumption that this represents the order in which the requests were submitted. This is not an ideal algorithm where some rides are egregiously long, but it does function and allow the simulator to progress through the course of a day. This allows us to further refine the simulator before engaging with the real decision-making models.

4.1.1 *Refining the Simulation*

One poor result would occur when "leftover" riders overlap with otherwise equivalent new riders, a leftover rider being one who had gone unpaired in the previous timestep and rolled over into the next. This algorithm was equally likely to pair a driver to the leftover or to the new rider. This resulted in some riders waiting for unnecessarily long periods while others did not wait at all.

To make rider behavior a bit more realistic, we allow riders to "renege" after trying and failing to pair with a driver for a full minute. In queue theory, reneging refers to a customer tiring of waiting and abandoning the queue before service. Thus if our algorithm fails to pair a driver to a ride request, we assume that the rider will switch to another platform, hail a conventional taxi, find another mode of transit, or give up on the destination altogether.

To minimize reneging, we altered the Closest Match algorithm to select "leftover" riders first, creating a subclass of riders with priority for being left unpaired in previous timesteps. Because of this, those "otherwise equivalent" cases will not result in a lost user.

Now with a simulation that runs from the initial conditions to the time horizon (typically midnight to midnight), we wanted to see results. How much are the drivers earning? What proportion of riders renege? Are riders mostly paired in their first timestep? How many miles are they driving?

Most of this information couldn't be appropriately collected conveniently, so we decided to replace our lists of integer riders and drivers R and D with lists of specialized car objects R and D. One car object represents one driver, and they collect information from each ride including fares earned, pickup time, miles to the pickup location, miles to the destination, and more. By aggregating all of this information after the fact, we can evaluate the customer experience in terms of waiting time and reneging likelihood, platform earnings in terms of fares earned and fuel expenses, and sustainability and efficiency in terms of total miles, nonempty miles, and vehicle usage rates.

4.2 *Uber's Batched Matching*

According to Uber's website:

> In the early days, a rider was immediately matched with the closest available driver. It worked well for most riders but sometimes led to long wait times for others. Across a whole city, those longer wait times really added up.
>
> But if we wait just a few seconds after a request, it can make a big difference. It's enough time for a batch of potential rider-driver matches to accumulate. The result is better matches, and everyone's collective wait time is shorter.

The former is what we implemented in the last section, but the latter is what Uber calls *Batched Matching*. It's a new driver-rider matching algorithm that waits a few seconds to gather available drivers and riders and settles them all in one "batch" to minimize pickup times.[2]

Given that we were implementing a algorithm that matches elements in one set to elements in another by minimizing a property of individual matches, Batched Matching is ideal to be modeled with the minimum-weight bipartite matching problem.

To introduce some basic terminology, a graph $G = (N, E)$ is composed of a set of nodes N and a set of edges E. An edge $e = (u, v)$ is incident upon (i.e. connects) two nodes u and v. Edges can be either directed (i.e. traveling strictly from u to v) or undirected. A graph $G = (N, E)$ is *bipartite* if the node set N can be partitioned into two node sets A and B such that there exists no edge $e = (u, v)$ where both endpoints are in the same side of the $A - B$ bipartition. A matching M is a subset of the edges E such that every node $n \in N$ is incident to at most one edge in M. A matching is *perfect* if $|A| = |B|$ - the two sets of the bipartition have the same number of nodes - such that every node is incident upon exactly one edge in the matching. In a perfect matching, every node $a \in A$ is matched to a node $b \in B$.[25]

In our matching problem, our bipartition is the set of drivers and riders accumulated in one batch, and every edge from driver d to rider r represents driver d accepting rider r's ride request. We can visualize this with the bipartite graph in Figure 8.

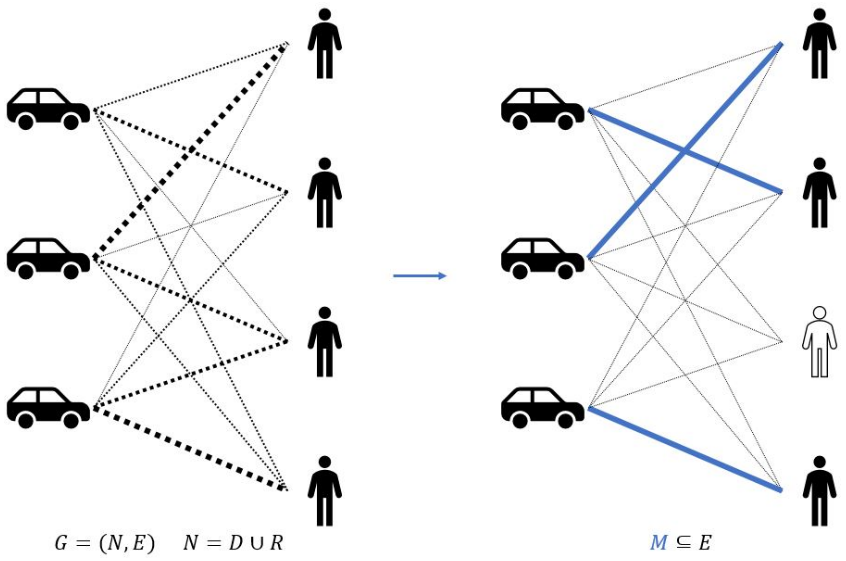

Figure 8: Constructing a Matching $M \subseteq E$ from a Bipartite Graph $G = (N, E)$ of Drivers D and Riders R

If we attribute a *weight* to each edge $e = (d, r)$ equal to the time it takes for driver d to reach rider r, then our goal would be to find a matching of maximum size and minimum total weight - the minimum weight bipartite matching problem. Our problem is equivalent to the following integer program:

$$\begin{aligned} &\text{argmin}_{a_{ij}} && \sum_{i \in D, j \in R} w_{ij} a_{ij} && && \\ &\text{subject to} && \sum_{i \in D} a_{ij} \leqslant 1, && j \in R && (1) \\ & && \sum_{j \in R} a_{ij} \leqslant 1, && i \in D && (2) \\ & && \sum_{i \in D, j \in R} a_{ij} = \min(|D|, |R|) && && (3) \end{aligned}$$

where w_{ij} is the pickup time from driver i to rider j and

$$a_{ij} = \begin{cases} 1, & \text{if rider } j \text{ is assigned to driver } i \\ 0, & \text{if rider } j \text{ is not assigned to driver } i \end{cases}$$

Constraint (1) ensures that no rider j is assigned to multiple drivers. Constraint (2) ensures that no driver i is assigned to multiple riders. Constraint (3) ensures that we designate a maximum matching - i.e., if there are fewer riders than drivers, every rider will be assigned, but if there are fewer drivers than riders, every driver will be assigned. Binary a_{ij} values assure that a match is made or it is not - e.g., rider j will not receive 40% of a ride from driver i_1 and 60% of a ride from driver i_2.

A minimum weight bipartite matching problem such as this can be solved using the Kuhn-Munkres algorithm, but we solved it using Python library NetworkX as a reduction to the minimum cost flow problem.[25]

To apply this algorithm, a ride-hailing platform would wait and gather available drivers and riders during a specified "batch time", solve the above integer program with the gathered drivers and riders as input sets D and R, dispatch orders to drivers, and repeat indefinitely.

One preliminary result we noticed was the existence of unpopular requests that originate far from the city center. Whenever there are too few available drivers in the timestep to match, these ride requests often go ignored, and they likely renege. However, whenever there were more available drivers than active requests, this algorithm matches all riders to a driver no matter how poor the matching. This resulted in an occasional worst-case matching, unprofitable for the platform and unacceptable waiting time for the customer.

Though it is not inherent to the Batched Matching algorithm, Uber engages in some form of demand-side admission control to ensure that such ride requests do not disrupt the system as a whole.[26] This project does not engage with admission control for this algorithm nor any other matching algorithm.

4.3 *Didi's Learning & Planning*

Didi's learning and planning approach differs mostly from Uber's Batched Matching by its consideration of future reward in immediate decision-making.

4.3.1 *The Learning Step*

In the learning step, we find a *spatiotemportal quantization* of supply and demand patterns in the ride-hailing environment.[3] In other words, we use historical data to gauge the value of a ride-hailing vehicle being at a particular place at a particular time. To do this, we apply a Markov Decision Process (MDP), a sequential decision-making tool used to model the behavior of an **agent** in its **environment**.

AGENT: In this MDP, we take a local view where each individual driver is modeled as an **agent**, and its **environment** is the city populated with ride requests. This differs from a global view where the whole system is one complicated agent. Although the local view can become costly when modeling thousands of agents at each planning step, most of this is mitigated by treating the driver agents as identical and by making their decisions cooperatively. The advantage of this perspective is a more meaningful view of state transitions, rewards, and actions.

The goal of the agent is to maximize its **gain** $G_t = \sum_{i=t}^{T} R_{t+1}$. In words, gain is the expected rewards the agent will accumulate starting from a particular time t. In order to achieve this goal, we learn the value of the agent being in a particular state s. We represent this expected gain using the state-value function $V(s)$:

$$V(s) = \mathbb{E}[G_t | s_t = s].$$

Equipped with a state-value function $V(s)$, the agents can make decisions that most effectively maximize their gain.

STATE: - A driver agent's **state** $s = (g, t)$ is represented by its spatiotemporal position with a spatial index $g \in G$ and a time index $t \in T$. Each state s has a different value in the state-value function $V(s)$. This is intuitive given that a vehicle in the Loop at 5:00pm does not have the same prospects as another in the Loop at 5:00am or yet another at O'Hare at 5:00pm.

Given its state, an agent may take a number of **actions**.

ACTION: - In this environment, there are two main types of actions. First, a driver may take a *serving action* - accept a ride request, pick up a passenger, and drive them to their destination. Second, a driver take an *idle action* - not accepting a ride request in this timestep and instead waiting in the same location for another request.

Taking an action - serving or idle - alters the agent's state. In an idle action, the agent's state $s = (g_i, t_i)$ transitions to a new state $s' = (g_i, t_{i+1})$ because it waits for one timestep to pass and remains in the same spatial location. In an idle action, both indices likely change. The agent's state $s = (g_i, t_i)$ transitions to a new state $s' = (g_j, t_{i+\Delta t})$ where Δt is the amount of time between accepting the ride request and dropping the passenger off at their destination g_j. State transitions by action type are displayed in Figure 9.

In the Learning Step, we only observe past actions. Selecting actions based on state-values comes later in the Planning Step.

REWARD: - Depending on the action taken, the agent will receive some **reward**, the entire optimization goal of the system. In this environment, reward is defined by fares earned and per-mile costs paid.

Our fare calculations take after Uber's standard, using a base fare, additional rates per mile and per minute, and a minimum that all fares must exceed. In Chicago, UberX \$0.20 per minute and \$0.90 per mile on top of a \$1.70 base fare, and all fares must exceed the base fare of \$2.70.[27] Taking community areas i and j as input, our fare function $f(i, j)$ is

$$f(i, j) = \max(\ p_0 + p_x x_{ij} + p_t t_{ij},\ p_{min}\ ),$$

where p_0 is the base fare, p_x is the price per mile, p_t is the price per minute, and p_{min} is the minimum fare. For our Chicago simulations, we use

$$f(i, j) = \max(\ 1.7 + 0.9x_{ij} + 0.2t_{ij},\ 2.7\ ).$$

After calculating the fare, per-mile costs must be subtracted away to get the action's reward. If we consider the pairing of a driver in i to a rider in j who wants to get to k, then our undiscounted reward can be calculated as follows:

$$R(i, j, k) = f(j, k) - c_m(x_{ij} + x_{jk}),$$

where c_m represents the cost per mile which is applied the the distance to pickup x_{ij} and the distance to the destination x_{jk}.

DISCOUNT FACTOR: - The discount factor γ regulates how far the model looks into the future. Due to the time value of money and uncertainty throughout the ride-hailing environment, equal monetary value has a greater value sooner rather than later. For example, a discount factor of $\gamma = 0.99$ means that the value of reward in the next time step is 99% the value of immediate reward. Additionally, larger discount factors drive greater variance in the state values, making consistent planning difficult. Serving actions earn fares from the passenger. We apportion reward to the agent over the course of the ride, so we can model the reward earned by the agent with the following

$$R^a_\gamma = \sum_{t=0}^{T-1} \gamma^t \frac{R(a)}{T},$$

where $R(a)$ is ride a's undiscounted fare and T is the total number of timesteps the ride takes.

Let's consider a rider at the Art Institute of Chicago (in the Loop, Community Area 32) hailing a ride to their home in Lincoln Park (C.A. 7). We can calculate the fare of this 4.6-mile, 10-minute ride using our fare function $f(32, 7) = \max(\ 1.7 + 0.9 \cdot 4.6 + 0.2 \cdot 10,\ 2.7\ ) = \$7.84$. The rider would pay \$7.84 for this ride.

Let's assume that we paired a driver who was idling in the Near South Side (C.A. 33). Therefore, it will take 6 minutes and 1.4 miles to reach the pickup location. To get the undiscounted reward, we have to subtract the per-mile costs. If we accept \$0.05 per mile, then we have $R(33, 32, 7) = f(32, 7) - 0.05(1.4 + 4.6) = \$7.54$. After receiving a \$7.84 fare, the driver would net \$7.54 after paying 30 cents in fuel.

The action takes a total of 16 minutes: 6 minutes to pickup and 10 minutes to destination. For ease of calculation, let's assume our timesteps are 4 minutes long with a discount factor of $\gamma = 0.9$. Thus, we can discount the reward on this 4-timestep ride with the following:

$$r = \sum_{t=0}^{4-1} \gamma^t \frac{7.54}{4} = 0.9\frac{7.54}{4} + 0.9^1\frac{7.54}{4} + 0.9^2\frac{7.54}{4} + 0.9^3\frac{7.54}{4} = 6.4825.$$

Thus, this serving action pairing d_{33} to $r_{32,7}$ resulted in a reward of 6.4825.

POLICY EVALUATION: - Because the model treats each driver agent as identical, they can all learn through all of their experiences. This allows us to compile all of the historical data and to garner from it state-values that can be equally used for any driver in their pairing decisions.

To fit the MDP model, we break down the historical data into its state transition pairs (s, a, s', r) which we can iterate through to calculate state-values. Any of these pairs represents either an idle action or a serving action. As seen in Figure 9, an idle action earns no immediate reward, and the agent will remain in this same location until the next timestep. Thus, we can use the following Temporal-Difference (TD) update rule for the agent's state following an idle action:

$$V(s) \leftarrow V(s) + \alpha[0 + \gamma V(s') - V(s)],$$

where $s = (g, t)$ is the space-time state in which the agent currently resides, $s' = (g, t')$ is the space-time state to which the agent moves, $V(s)$ and $V(s')$ are the estimated values of those states, and α is the learning rate. On the other hand, a serving action does earn immediate

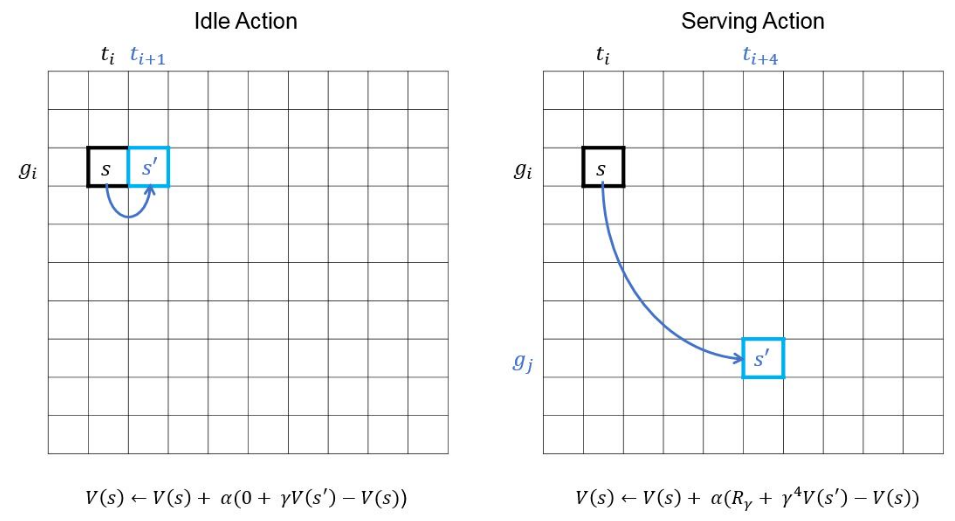
Idle Action
Serving Action
t_i
t_{i+1}
t_{i+4}
g_i
g_j
s
s'
$V(s) \leftarrow V(s) + \alpha(0 + \gamma V(s') - V(s))$
$V(s) \leftarrow V(s) + \alpha(R_\gamma + \gamma^4 V(s') - V(s))$

Figure 9: Temporal Difference (TD) update for both kinds of actions - Idle and Serve[3]

reward and the next state $s' = (g', t')$ likely consists of a different spatial location. Likewise, we have the following TD update rule for serving actions:

$$V(s) \leftarrow V(s) + \alpha[R^a_\gamma + \gamma^{\Delta t} V(s') - V(s)],$$

where R^a_γ is the discounted reward earned through the ride and Δt is the number of timesteps the ride takes including both pickup time and ride time.

We aim to solve for the state-values using dynamic programming. To do this, we choose to define our MDP with a finite horizon where one episode collects all the state transitions from one day, midnight to midnight. After simulating a full episode, we iterate through transition pairs backwards through time - from hour 2400 to hour 0000 - using the TD update rules to augment the state-values. This Policy Evaluation algorithm is stated in Table 2. To speed up convergence on state-values and to induce explorative behavior, we initialize $V(s)$ from the normal distribution $\mathcal{N}(\mu, \sigma^2)$ $\forall s \in S$ instead of a set of zeros as prescribed in the Didi Chuxing paper. The function $N(s)$ tracks the number of times each state has been visited to dynamically adjust the learning rate.

The state-value function outputted from the Policy Evaluation algorithm is the *spatiotemporal quantization* described by Didi Chuxing's AI Labs. Using this state-value function, we can make better driver-rider pairings in the Planning Step.

4.3.2 *The Planning Step*

In Uber's Batched Matching, the platform waits a specified length of time - known as the *batch time* - to gather all the available drivers and riders. The driver-rider pairings are made for all of them simultaneously in one batch using combinatorial optimization. Didi Chuxing's Planning Step works exactly the same way, but instead of making pairs that minimize immediate pickup times, they seek to maximize profits. We visualize both pairing processes using the bipartite graph seen in Figure 8.

Learning Algorithm: Policy Evaluation for the MDP with dynamic programming[3]

```
Input: Historical state transitions $E = \{(s_i, a_i, r_i, s'_i)\}$; each state is composed of a
        space and time index: $s_i = (g_i, t_i)$
(1) Initialize $V(s) \sim \mathcal{N}(\mu, \sigma^2)$, $N(s) = 0, \forall s \in S$
(2) for $t = T - 1$ to $0$ do
(3)     Find a subset $E^{(t)}$ where $t_i = t$ in $s_i$.
(4)     for each sample $(s_i, a_i, r_i, s'_i)$ in $E^{(t)}$ do
(5)         $N(s_i) \leftarrow N(s_i) + 1$
(6)         $V(s_i) \leftarrow V(s_i) + \frac{1}{N(s_i)}[\gamma^{\Delta t(a_i)} V(s'_i) + R_\gamma(a_i) - V(s_i)]$.
(7)     end for
(8) end for
Return: State-Value Function $V(s)$ for all states
```

Table 2: Policy Evaluation[3]

Uber's algorithm falls into a trap of making excellent pairings now at the expense of pairings later. Didi's algorithm attempts to eliminate that trade-off by considering expected *future* reward in addition to immediate profits. They accomplish that using the Advantage function A_{ij}:

$$A_{ij} = \gamma^{\Delta t_j} V(s'_{ij}) - V(s_i) + R_\gamma(j),$$

where $V(s_{ij})$ and $V(s'_{ij})$ are the values of the current and future states computed in the Learning Step, $R_\gamma(j)$ is the discounted reward expected from the action, γ is the discount factor, and Δt_j is the number of timesteps the action takes. By calculating the advantage of every possible action in each batch, we can set new weights on our bipartite graph. We then solve for the driver-rider pairings by selecting the matching that maximizes the total weight. This process is equivalent to the following integer program:

$$\begin{aligned} \text{argmax}_{a_{ij}} \quad & \sum_{i \in D, j \in R} A_{ij} a_{ij} & \\ \text{subject to} \quad & \sum_{i \in D} a_{ij} \leqslant 1, \quad j \in R & (1) \\ & \sum_{j \in R} a_{ij} \leqslant 1, \quad i \in D & (2) \end{aligned}$$

where

$$a_{ij} = \begin{cases} 1, & \text{if rider } j \text{ is assigned to driver } i, \\ 0, & \text{if rider } j \text{ is not assigned to driver } i. \end{cases}$$

Constraint (1) ensures that no rider j may be paired to more than one driver i. Likewise, constraint (2) ensures that no driver i may be assigned to more than one rider j. The binary definition of a_{ij} ensures that there are no partial pairings - e.g., driver d_a may not be assigned 70% to r_b and 30% to r_c.

Solving the problem graphically, a flow of one across the bipartition from driver i to rider j represents the platform dispatching driver i to service rider j's ride request. The same is represented in the integer program with a value of $a_{ij} = 1$. A flow of 0 or a_{ij} in the IP means

Planning Algorithm: Order Dispatch for Real-Time Driver-Rider Matching[3]

```
Input: State-value function $V(s)$
(1) for every timestep in order dispatch do
(2)     Collect and create nodes for each available driver
        $i \in D$ and active ride request $j \in R$.
(3)     Create edge from each available driver $i$ to each
        ride request $j$ with weight $A_{ij}$.
(4)     Solve using Kuhn-Munkres algorithm.
(5)     Dispatch drivers to their matched ride request.
(6)     Unmatched drivers carry over into next timestep.
        First-time unmatched ride requests carry over into
        next timestep, but second-time unmatched ride
        requests expire.
(8) end for
```

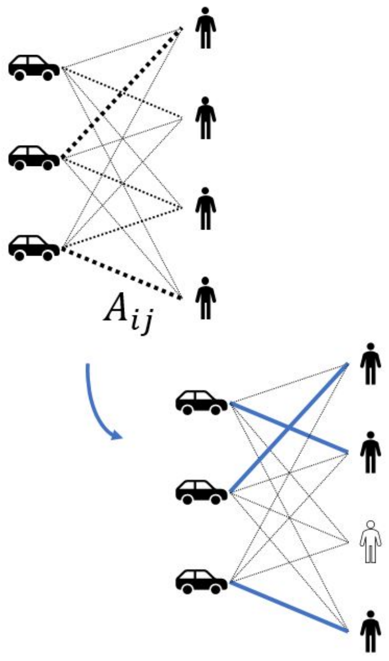

Table 4: Order Dispatch[3]

that driver i is not assigned to rider j. Didi Chuxing solves each planning stage using the Kuhn-Munkres algorithm. We choose to solve using the Python library NetworkX's minimum cost flow function.

4.3.3 *Reinforcement Learning: Combining Learning and Planning*

The paper discusses how the above practice may be framed within the context of *reinforcement learning* - an area of machine learning where an agent attempts to determine a policy π that will maximize the expected gain:

$$\mathbb{E}_\pi \left[G_t \,\middle|\, s \right],$$

where $s \in S$ is the state of the whole ride-hailing platform in timestep t. Our centralized agent takes action through the platform's drivers, so we may decompose this objective function into components for each driver i:

$$\mathbb{E}_\pi \left[\sum_{i \in D} G_t^i \,\middle|\, s = \{s_i\}, \forall\, i \in D \right],$$

where G_t^i is the gain of a particular driver i in timestep t and s_i is the the state of driver i. Due to the linearity of expectation, we can further decompose this into the following:

$$\sum_{i \in D} \mathbb{E}_{\pi_i} \left[G_t^i \,\middle|\, s_i \right],$$

which represents the sum of individual agents' expected gain given its own state s_i and policy π_i. Computing this for self-interested agents would be difficult, but because we make the decisions with one centralized decision-making meta-agent, we can direct agents to make decisions for the platform-level objective. Effectively, this means that all agents share a policy $\pi_i = \pi$. Maximizing this expected gain is equivalent to maximizing the Advantage function values and, because we assume the driver agents to be functionally identical, they share Advantage functions just as they share state-value functions:

$$\sum_{i \in D} \mathbb{E}_\pi \left[G_t^i \,\middle|\, s_i \right] = \sum_{i \in D} A_\pi^i(s_i, a_i) = \sum_{i \in D} A_\pi(s_i, a_i), \tag{1}$$

Because drivers all share the same policy π, for clarity, we frequently omit the policy subscript. For conciseness, we often frequently replace the input values (s_i, a_i) with subscripts i and j denoting the current state s_i and the next state s_j resulting from the action a_i: A_{ij}.

4.4 *Our Contribution: Incorporating Autos*

Both Uber's and Didi's algorithms described above directly match drivers to riders. Our contribution, described in the sections below, expands upon the previous algorithms to improve urban mobility by incorporating self-driving vehicles, and hence Garages for their storage and fueling. In fact, self-driving vehicles enable a system-wide optimization approach proposed in the paper which is a key point of departure from other research done in the literature, such as those in the mentioned works.

Ride-hailing platforms fight a constant battle to match supply with demand. Most resort to surge pricing that simultaneously incentivizes drivers to work during peak hours and incentivizes riders to avoid peak hours and wait for a cheaper ride later. Despite incentives, some riders still walk away without a ride while others contribute to traffic congestion.

The paper from AI Labs at Didi Chuxing took a step in the right direction to more adequately address supply-demand imbalance by framing the ride matching problem more broadly, both spatially and temporally. By doing this, they utilize driver-rider matching decisions as a marketplace balancing tool.

We hypothesized that we could improve efficiency and performance in ride-hailing and urban mobility in general - measured primarily in platform earnings, pickup times, and carbon emissions - by replacing conventionally driven vehicles (CDVs) with self-driving vehicles. With complete control over the number of vehicles on the streets at any given moment, we can more efficiently match supply with demand.

4.4.1 *Autos*

"Self-driving cars" can mean many things. For the tractability of the study, we will narrow down this definition. *Autos*, as we will refer to them now on, are fully self-driving taxis. This primary characterization means that:

1. An Auto may operate in any conditions with neither a human driver nor human occupants.
2. Autos are not personally owned. They are owned or contracted with a municipality for the purpose of on-demand point A to point B travel.

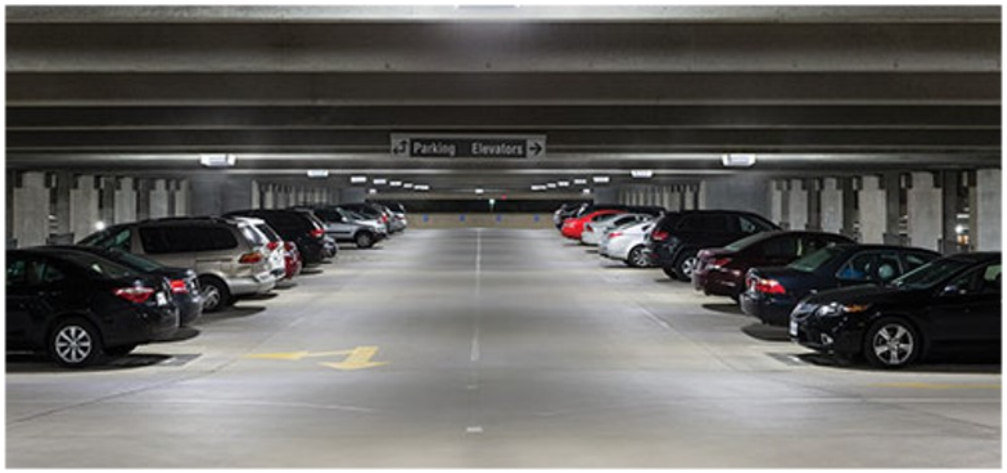

Figure 10: A normal parking garage, not the specialized hubs we refer to as a *Garages* in the model.[4]

Beyond these two simple points, we allow Autos to behave like CDVs in just about every other regard. Autos do not fuel more cheaply or more efficiently, they do not drive any faster or slower than CDVs, and we do not allow any inter-vehicle communication or anything else substantial.

Nonetheless, these two points change the ride-hailing game significantly. With no contracted drivers who sign in and out of the platform unexpectedly, the number of active drivers is known and controlled. This may be optimized to hourly fluctuations. However, all the operational costs - fuel, cleaning, maintenance, storage, etc. - that were once paid by drivers must now be paid by the platform.

We soon introduce *Garages* where Autos may fuel and store themselves. In the Batched Matching model, Autos automatically store themselves to fuel if necessary. In the Planning model, just like how drivers would pair to riders based on the advantage of each pairing, Autos store themselves, launch themselves from storage, and make rider pairings all based on that same advantage formulation. This entails marginal adaptations to both the learning and planning processes described in Didi Chuxing's paper.

4.4.2 *Garages*

We introduce *Garages* as facilities to house and refuel Autos. A reader might envision a typical parking garage like that in Figure 10, but a Garage is highly optimized. Parking garages are designed for personally owned vehicles. As seen in Figure 10, the accessibility of each individual vehicle by its owner demands a very low vehicle density - wide two-way lanes, large turn radii, oversized parking spots. With ownerless, driverless cars, there is no need for each vehicle to be accessible at any given time. This eliminates the need for the wide central lane. Without human drivers prone to error, vehicles can be parked much closer together. With a fleet of identical vehicles, parking spots need not be oversized and ceilings need not be so high to accommodate the largest vehicles. A Garage may have a much larger capacity than a parking garage of equal volume. Additionally, the use of identical vehicles and the absence of human drivers allows for highly specialized and automated fueling procedures.

By not offering fares and instead fueling vehicles for later activity, Garages present a meaningfully different destination. To account for this, each Garage represents its own spatial location in the Learning Step - i.e., it has its own state-values that fluctuate over the course of a day.

Likewise for the Planning Step, Garages provide a new destination for active Autos, so we add them to the demand side of the bipartition with ride requests to allow Autos to fuel themselves whenever advantageous. The number that may store themselves in any Garage is constrained by the capacity and current occupancy of each Garage.

Additionally, we allow Autos in storage to match with riders. Thus, we add Garages to the supply side of bipartition with active Autos so that stored vehicles may help meet demand when advantageous. The number that may launch themselves from storage is constrained by the current occupancy of each Garage.

4.4.3 *CDVs*

Conventionally driven vehicles, or *CDVs* as we will refer to them hereon, are vehicles operated by traditional human drivers without machine assistance. They represent today's standard Uber, Lyft, or Didi taxis. They are personally owned, fueled, maintained, and stored by their driver. In return for shuttling riders to their destinations, drivers earn the majority of each fare. As a commission for connecting drivers to riders, the platform accepts a percentage of each fare. Uber advertises that is takes only 25% from each fare, but when considering minimum fares and other booking fees, the cut can effectively exceed even 50% in some cases.[28] We set this platform cut percentage to 30% in our models.

The ideal of the gig economy, platforms allow drivers to work when they choose. Drivers sign in when they want to work and sign out when they want to stop. This presents a lack of certainty in the platform in knowing the supply-demand balance and hinders their ability to plan for future balance shifts. In the model, we represent this with frequently occurring small probabilities of each active driver signing out and for inactive drivers to sign in. Those probabilities fluctuate according to traffic, demand, and shift length of each individual driver.

4.4.4 *New Learning Step*

In terms of the MDP, the Learning Step only requires minor adaptations to accommodate Autos.

AGENT: Each individual Auto is an agent that performs actions in order to collect reward.

STATES: States $s = (g, t) \in S$ are still space-time pairs with spatial index $g \in G$ and time index $t \in T$, but in addition to discrete physical regions, G includes indices for each Garage.

ACTIONS: In addition to serving actions, the agent may now take several new actions.

1. **Launch** actions dispatch Autos in storage to serve active ride requests.
2. **Store** actions send active Autos to a Garage to refuel.
3. **Redistribute** actions, rarely advantageous but permitted, send Autos stored in one Garage to storage in another Garage.
4. Previously identified by inaction, the **Idle** action is now defined explicitly to give the solver flexibility in the number of supplementary actions taken.

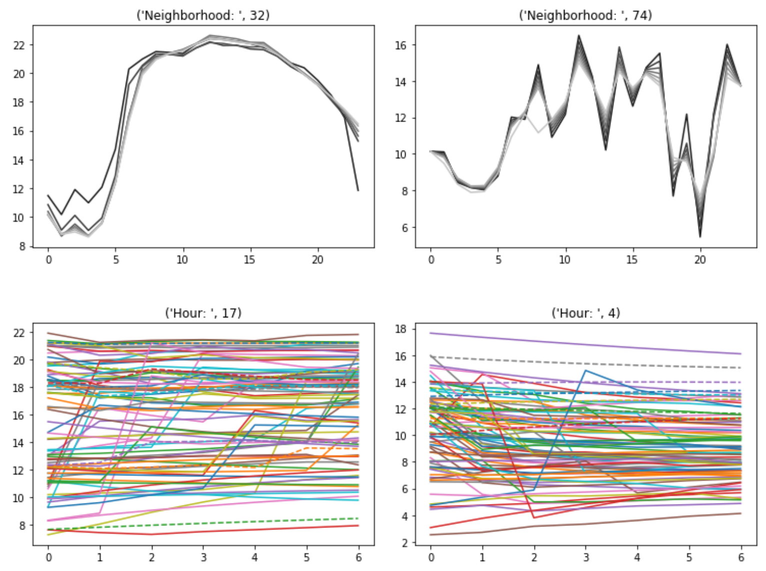

Figure 11: Learning state-values in (ABOVE) the Loop and Morgan Park and at (BELOW) 5:00pm and 5:00am

5. Even though not an action taken by a particular Auto, the solver may explicitly **Ignore** ride requests when it is disadvantageous to serve them.
6. Representing inaction in storage, the **Remain** action orders an Auto to stay in storage and to continue fueling.

REWARD: Reward is handled the same way for serving and idle actions except that Autos have higher per-mile costs. Launch actions are handled the same way as serving actions. Store and Redistribute actions earn no fare but still pay per-mile operating costs to reach their assigned Garage.

POLICY EVALUATION: The Policy Evaluation algorithm used to evaluate space-time pairs remains unchanged but now evaluates a greater number of states.

We visualize the learning process at the heart of Chicago with the Loop and less trafficked regions with Morgan Park in the top row of Figure 11. The dense supply-demand network in the Loop results in a smoother learned state-function across $V(33, t) \, \forall \, t \in T$ (Community Area 33 is indexed by 32 in the simulator as displayed in the figure). The sparser network in Morgan Park results in a choppier hourly state-value profile. In the bottom row of Figure 11, we visualize the learning process at 5:00pm (Left) where the general upward trend demonstrates learning the greater state-value and 5:00am (Right) where the general downward trend demonstrates the opposite. Each individual solid line on these bottom plots represent one Community Area while the dotted lines represent Garages.

4.4.5 *New Planning Step*

Given centrally owned and operated Autos, storage and fueling must be incorporated into the model. As opposed to Batched Matching, the Planning Step offers a unique opportunity to incorporate storage, fueling, and launch decisions into the same framework. It doesn't make sense

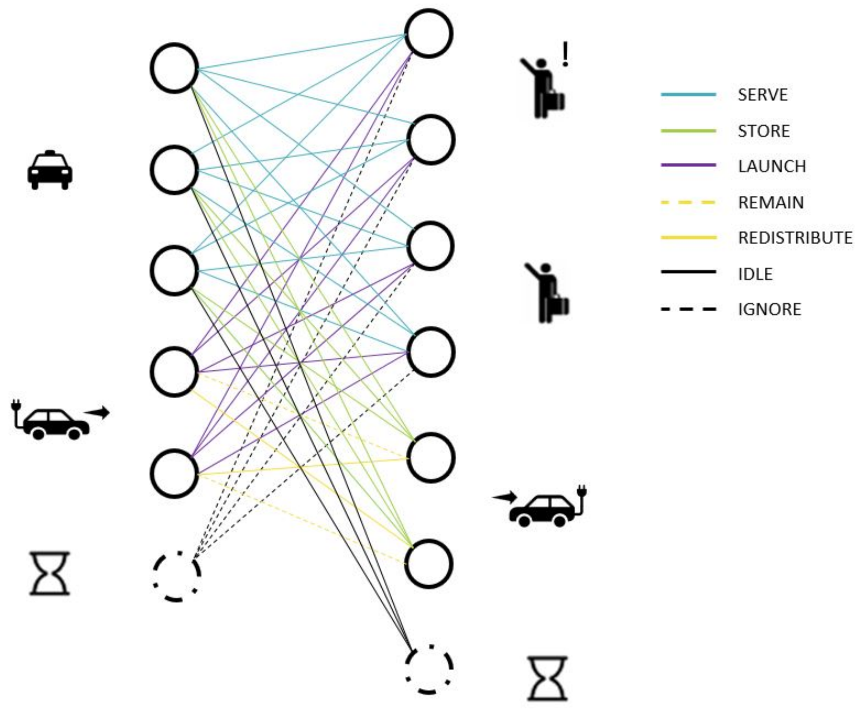

Figure 12: Incorporating Autos and Garages into the Planning Step creates several new types of actions. (Best viewed in color)

to evaluate actions beyond driver-rider matchings according to immediately minimized pickup times, but convenient storage can be just as advantageous as a good match when looking broadly through time.

Like before, for each timestep, we collect the batch of available drivers (now Autos) and unmatched ride requests and split these onto the supply and demand sides of the bipartition respectively. In addition, we add a node for each nonempty Garage to the supply side whose outflow is constrained by each Garage's current occupancy. Outflow from these nodes to a rider represents a Launch order. Outflow from these nodes to another Garage node on the demand side represents a Redistribute order if the Garages are different or a Remain order if the Garages are the same. Lastly, we add an Ignore node to the supply side.

Next, we add a node for each unfilled Garage to the demand side whose inflow is constrained by the Garage's capacity minus the number of already inbound Autos. Inflow through an active Auto node represents a Store order, but inflow through other Garage nodes represents a Redistribute or Remain order. Lastly, we add an Idle node to the demand side.

We may choose to differentiate ride requests between new requests and delayed requests by imposing a slight incentive for matching with delayed requests. New requests unmatched after one timestep become delayed requests in the next. We assume that unmatched delayed requests expire. By varying the imposed incentive, we can influence the likelihood of delayed requests matching to an Auto compared to a similar new request. All of the above changes to the planning algorithm are displayed in Figure 12.

Unlike the conventional Planning Step, not all supply side nodes connect to all demand side nodes with an outgoing edge. Like before, active Auto nodes connect to ride request nodes (Serve), but now they also connect to Garage nodes (Store) and to the Idle node (Idle). The Garage nodes connect to ride request nodes (Launch) and to other Garage nodes (Redistribute or Remain). The Ignore node connects only to ride request nodes (Ignore). Each of these order types are visualized in Figure 12. Just like the conventional Planning Step, the weight of each edge is computed using the Advantage function A_{ij}:

$$A_{ij} = \gamma^{\Delta t_j} V(s'_{ij}) - V(s_i) + R_\gamma(j),$$

where the reward is calculated just the same way by discounting fares received minus per-mile costs paid. Only Serve and Launch actions earn fares, but Serve, Launch, Store, and Redistribute actions all incur per-mile costs. The above flow problem is equivalent to the following integer program:

$$\begin{array}{llll}
\text{argmax}_{a_{ij}} & \sum\limits_{i\in\mathcal{S},j\in\mathcal{D}} A_{ij}a_{ij} & & \\
\text{subject to} & \sum\limits_{j\in\mathcal{D}} a_{ij} \leqslant 1, & i \in D & (1) \\
& \sum\limits_{i\in\mathcal{S}} a_{ij} \leqslant 1, & j \in R & (2) \\
& \sum\limits_{j\in\mathcal{D}\setminus\{I_1\}} a_{ij} \leqslant occ(i), & i \in G_\mathcal{S} & (3) \\
& \sum\limits_{i\in\mathcal{S}\setminus\{I_2\}} a_{ij} \leqslant cap(j) - in(j), & j \in G_\mathcal{D} & (4) \\
& a_{I_2,j} = 0, & j \in G_\mathcal{D} \cup I_1 & (5) \\
& a_{i,I_1} = 0, & i \in G_\mathcal{S} \cup I_2 & (6) \\
& a_{ij} \text{ integer} & i \in \mathcal{S}, i \in \mathcal{D} & (7),
\end{array}$$

where $\mathcal{S} = D \cup G_\mathcal{S} \cup I_2$ is the supply side of the bipartition, $\mathcal{D} = R \cup G_\mathcal{R} \cup I_1$ is the demand side of the bipartition, $G_\mathcal{S}$ and $G_\mathcal{D}$ are the Garage nodes on either side of the bipartition, I_1 is the Idle node, I_2 is the Ignore node, $occ(i)$ is the occupancy of Garage i, $cap(i)$ is the capacity of Garage i, and $in(i)$ is the number of Autos already dispatched to store in Garage i.

Constraint (1) ensures that no Auto is assigned to more than one task including serving a ride request, storing itself, or idling. Constraint (2) ensures that each rider is either matched to just one Auto or is ignored. Constraint (3) ensures that only as many Autos as a Garage's current occupancy may serve orders, redistribute, or remain. This constraint is always binding. Constraint (4) ensures that no Garage exceeds its capacity. Constraints (5) and (6) prohibit connections that don't have any functional meaning. Constraint (7) ensures that no partial orders are dispatched.

Like with the conventional Planning Step, each stage may be solved using the Kuhn-Munkres algorithm, but we choose to use NetworkX's minimum cost flow function in Python for its ease of use.

5 Numerical Analysis

With models formulated to compare Autos' and CDVs' impact on urban mobility, we trained and tested our models using real data. In the absence of private ride-hailing data, we chose a public

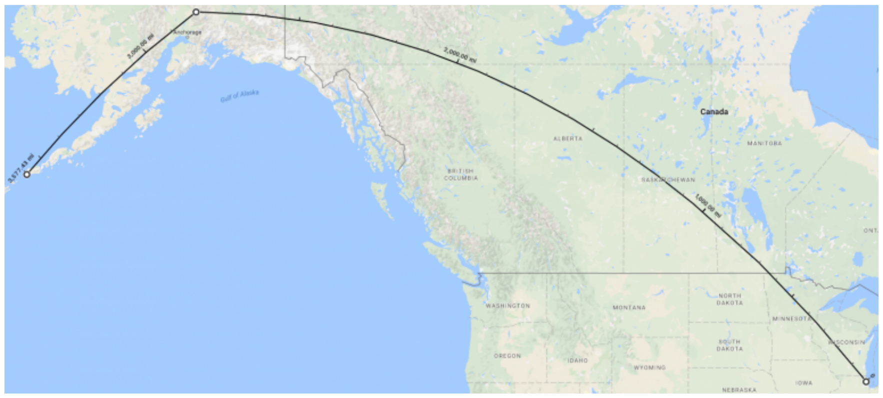

Figure 13: Some taxi trips are "plainly implausible".[5]

dataset of Chicago taxi trips to extract relevant information to spatially and temporally represent rider.

5.1 *Chicago Taxi Rides*

The City of Chicago anonymized and publicized all taxi trips in 2016 through their online Data Portal. With just under 20 million trips, it includes unique taxi IDs, start and end timestamps, pickup and drop-off Community Areas, latitude and longitude coordinates for pickup and drop-off, fares, trip distances, and more [29]. Although this represents data on hailed cabs, we treat it as a representation of demand for app-based on-demand rides.

5.1.1 *Cleaning the Data*

The City of Chicago reports that "open data can be messy data" before admitting that much of the data is "plainly implausible" [5]. Before publishing the data, they already removed trips exceeding 86,400 seconds, 3,500 miles, or $10,000 in fares. Given that 86,400 is the number of seconds in a day, there remain plenty of trips made implausible by duration. Likewise, as seen in Figure 13, 3,500 is "about the furthest from Chicago one can drive and end within the United States," so there remain many trips made implausible by length. Although $10,000 is obviously too much for a cab ride, the City admits that "$1,000 is not necessarily implausible for rare trips" [5]. So where do we draw the lines?

We concluded that instead of correcting for errata, we can report distances, times, and fares more consistently by generating them all within the simulation according to the bare bones necessities of pickup and drop-off locations and timestamps.

A preliminary search revealed that the dataset was missing significant amounts of information. Some information - including fares, trip times, and trip distances - are not necessary to input into the simulation, but other information - including timestamps and locations for both pickup and drop-off - are necessary. While almost no trips were missing timestamps, roughly 3 million entries (16.25%) were missing Community Area locations for either pickup or drop-off. When we observed that more than 99% of those entries with missing Community Areas were also missing their latitude and longitude coordinates, we knew that this portion of the dataset would not be salvageable.

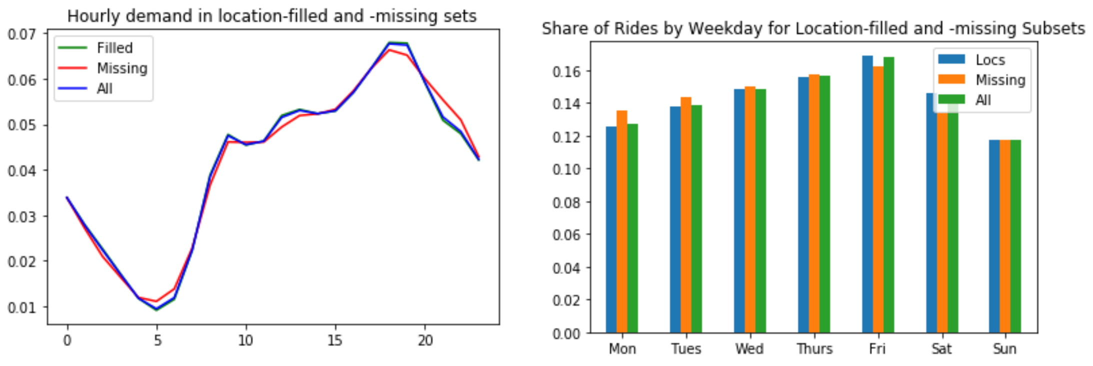

Figure 14: Data subsets with and without location data have the same profiles by weekday and hour.

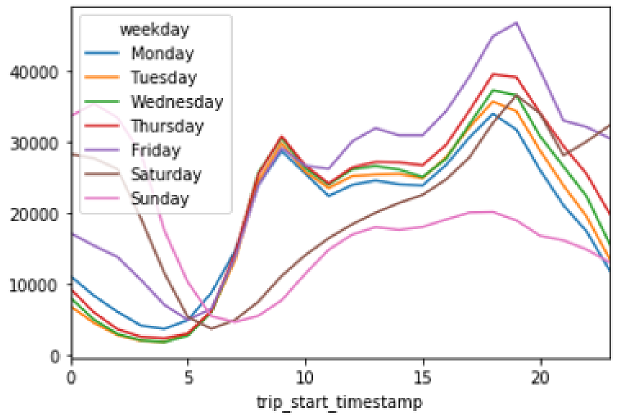

Figure 15: Demand Profile by Weekday

We hoped that the 16.25% of location-missing data was a representative sample of the entire dataset. Else - e.g., if it composed the longest trips or all the rides ending in Central Chicago - the abridged dataset would not adequately represent Chicago ride-hailing demand to our satisfaction. However, after comparing the location-missing and location-filled data subsets by weekday, timestamp, fare, and trip length, we determined that the location-filled subset of 16.6 million entries was representative enough of the whole set in order to move forward. The profiles of location data and location-missing data by hour and weekday are displayed in Figure 14.

In the exploratory search of the data, we also observed, as one would expect, that different weekdays displayed different hourly demand profiles as seen in Figures 15.

Because of these differences, we partitioned the dataset into four subsets: A, weekdays in the summer; B, weekends in the summer; C, weekdays in the winter; and D, weekends in the winter - where weekdays consist of Mondays through Fridays and summer exists between 2016's equinoxes on March 20th and September 22nd.

5.1.2 *Parameterizing Chicago*

First, we used Google Maps Directions API to calculate the time and distance matrices C_T and C_D for Chicago's 77 Community Areas.[24]

Chicago offers notably cheap Uber rides among American cities. In addition to a base fare of $1.70, Chicago residents and visitors pay $0.20 per minute and $0.90 per mile of the ride.[27]

5.2 *Cost Analysis*

5.2.1 *Operational Costs*

The Nunes & Hernandez study from the MIT Energy Initiative concluded that autonomous taxis were not economically viable due in large part to their prohibitive operational costs. To make a stronger argument, we took this as our starting point. Their study breaks down vehicle financing, licensing, insurance, maintenance, cleaning, fuel, and safety oversight in the San Francisco market into per-mile operational costs. We adjust their formulations with new inputs based on the Chicago market and more.

VEHICLE FINANCING: Nunes & Hernandez assume a low vehicle price of $15,000 in addition to a loan paid off over three years with a 7% annual interest rate. With this information, we can calculate the monthly loan payment with the following equation:

$$15000 \cdot \left(\frac{\frac{0.07}{12}(1+\frac{0.07}{12})^{36}}{((1+\frac{0.07}{12})^{36}-1)} \right) = 463.16.$$

Assuming an annual mileage of 90,000 miles and a lifespan of 5 years, they convert the $463.16 monthly payment of $36 \cdot \$463.16/(90,000 \cdot 5) = \$0.037$ per mile.

This per-mile financing cost is highly sensitive to input values. If we expect a car driving 90,000 miles per year might have a lifespan of 4 years instead of 5, then the cost jumps to $0.046 per mile. If we think $15,000 is an underestimation of the price, consider the popular-among-Uber-drivers Toyota Prius. A new 2020 Toyota Prius might go for $25,000 increasing the financing cost $0.062 per mile.[30] The vehicle financing cost we use in our base model is $0.077 per mile to account for both of these adjustments, but we vary the total per-mile cost in the sensitivity analysis to account for uncertainty.[21]

LICENSING: The Nunes & Hernandez study consider a taxi medallion, $200,000 in San Francisco. They consider a 20% down payment ($50,000), 25% of which is paid up front ($12,500). They finance this initial balance of $37,500 at a 5.4% interest rate over 7 years. With a similar calculation as the above, we get a monthly payment of $537.10 for 84 months:

$$37500 \cdot \left(\frac{\frac{0.054}{12}(1+\frac{0.054}{12})^{84}}{((1+\frac{0.054}{12})^{84}-1)} \right) = 537.10.$$

Likewise, financing the remaining balance $200,000 over 5 years, we have the following monthly payment calculation:

$$200000 \cdot \left(\frac{\frac{0.054}{12}(1+\frac{0.054}{12})^{60}}{((1+\frac{0.054}{12})^{60}-1)} \right) = 3811.01,$$

MODEL	MPG (City)	\$/mi	WEIGHT
Toyota Prius	58	0.0345	$\frac{137}{300}$
Honda Civic	32	0.0625	$\frac{77}{300}$
Toyota Camry	29	0.0670	$\frac{47}{300}$
Toyota Highlander (Hyb.)	36	0.0556	$\frac{9}{100}$
Chevrolet Suburban	15	0.1333	$\frac{1}{25}$

Table 5: Fuel costs per mile in the most popular vehicles among Uber drivers.[6]

yielding \$3,811.01 over 60 months. Assuming a medallion lasts 20 years, we find a licensing cost of $(\$537.10 \cdot 84 + \$3811.01 \cdot 60)/(90000 \cdot 20) = \$0.16$ per mile.

However, drivers on ride-hailing platforms like Uber and Lyft traditionally do not require taxi medallions to work, so there may be not per-mile licensing cost whatsoever. Anticipating the future of autonomous technology regulation is uncertain and difficult, and it seems soon to assume that driverless cars will be treated the same as taxi drivers.

Let's assume anyway that this is the case. In Chicago in March 2020, most taxi medallions cost below \$30,000, significantly less than the Nunes & Hernandez study.[31] By the same method, this results in a licensing cost of \$0.02 per mile.

INSURANCE: Assuming an \$800 monthly insurance premium, the Nunes & Hernandez study anticipates $\$800 \cdot 12/90,000 = \$0.11$ per mile based on the same expected mileage.[21] This rate matches what is expected in Illinois.[32]

MAINTENANCE: They attribute \$0.08 per mile for maintenance to the U.S. Bureau of Transportation statistics. This value has since grown to \$0.089.[33]

CLEANING: They cite a study conducted by Bosch, et al. which calculated an expected 3.00 CHF (Swiss Francs) per 100 km driven by a mid-size vehicle.[34] Thus, we calculate $(3.00\text{CHF}/100\text{km}) \cdot (1\text{km}/0.621371\text{miles}) \cdot (\$1.03/1\text{CHF}) = \$0.05$ per mile.[21] We don't expect this value has changed significantly.

FUELING: The Nunes & Hernandez study uses 43 mpg mileage and \$3.01 gas price to calculate a \$0.07 per mile fuel cost.[21]. Our calculations for Chicago in 2020 differ.

To get an idea of the Auto's fuel cost per mile, we compiled a list of Uber drivers' five most popular vehicles.[6] Using miles per gallon in the city by model and a gas price of \$2.00 per gallon - considering the Chicago gas price at the time of writing is \$1.908[35] and in 2016 at the time of data collection averaged \$2.237[36] - we calculated the cost of driving one mile in each vehicle based on fuel alone as seen in Table 5. Using the rank-order centroid weighting method to take a popularity weighted average, this results in a cost per mile of \$0.05.[37] This seems on the inexpensive side, so we vary this value in our sensitivity analysis to confirm that other values will not substantially alter the results.

However, given that vehicle purchasing decisions can be made centrally, this value could be reduced even further to as low as \$0.035 for a Toyota Prius fleet.

SAFETY OVERSIGHT: The Nunes & Hernandez study details Air Traffic Control-like supervision for clusters of up to fifty autonomous vehicles. For safety, we assign two supervisory monitors at a time to each cluster. Both work 12 hour shifts at minimum wage ($11/hour in California in 2018, but $14/hour in Chicago in 2020) which amounts to four monitors per day. We then consider two groups. The first group works Monday through Friday for 50 weeks in the year which amounts to 250 days in the year. The second group works weekends and the remaining two weeks, amounting to 116 days in the year.

Including an overhead rate of 1.59 attributed to one of the four monitors in the former group, we can compute the annual cost of one Auto cluster as $((250 \cdot 12 \cdot 14) \cdot 1.59 \cdot 1) + ((250 \cdot 12 \cdot 14) \cdot 3) + ((116 \cdot 12 \cdot 14) \cdot 4) = \$270,732$. The cost per mile depends on how large the clusters are - i.e., how many Autos each monitor is responsible for. A cluster size of 10 results in a cost of $(\$270,732/(90,000 \cdot 10)) = \$0.301$ per mile. With 50 Autos per cluster, the cost drops to $(\$270,732/(90,000 \cdot 50)) = \$0.060$ per mile.[21]

Certainly, other supervisory schemes are also possible. It seems that a redundant monitor on every cluster might not be the most efficient division of labor, but even with just one monitor per cluster, this framework still costs an annual $147,756. If we accept these two scenarios as bounds, we expect the cost of a 50-Auto cluster to exist between $0.033 and $0.060.

Given the above analysis, we expect the cost per mile of operating an Auto to be about $0.46 per mile in Chicago to account for vehicle financing, licensing, insurance, cleaning, maintenance, fuel, and oversight. In our sensitivity analysis, we vary the operational costs to determine how the results behave in other circumstances.

5.2.2 *Implementation Costs*

Having already accounted for vehicle financing, licensing, insurance, cleaning, maintenance, fuel, and oversight in the operational costs, we now consider the infrastructure costs entailed by a set of Garages.

Parking garage construction costs are typically broken down into costs per parking space and cost per square foot. In Chicago, typical garages cost a median of $21,797 per space and $65.23 per square foot.[38] Given these costs, the total cost of construction depends heavily on *parking efficiency*, square feet per parking space. The International Parking Institute reports that larger parking garages range from 300 to 360 square feet per space given their span, and smaller garages range from 360 to 400 square feet per space.[39]

A study by Nourinejad, Bahrami, and Roorda from the University of Toronto suggests that autonomous vehicles can park much more closely, decreasing parking space in a lot or deck by 62% to 87% for the same number of vehicles. Because they modeled for personally owned vehicles, they included space to allow reshuffling to retrieve vehicles in barricaded spots when recalled.[40] Thus, given that we consider a fleet of identical vehicles, this space is unnecessary, and space savings are likely even greater.

Our Garages are more than just storage locations. They also are also responsible for refueling the fleet. We already accounted for fuel in operational costs, but we will account for equipment here. A typical fuel pump costs $200 to $400.[41] There is room for plenty of specialization and optimization in the implementation, but to be generous, we allot an extra $1000 per space.

When considering a fleet of a particular size, we have a range of options from many Garages with small capacities to a few Garages with larger capacities. Let us first consider the former.

If we have a fleet of 1000 Autos, we could build ten small Garages with a capacity of 100 Autos each. Given a typical parking efficiency for a short-span Garage (360 square feet per space) and the savings for Autos (62%), we can expect to use 13680 square feet for 100 spaces in each of the ten Garages. Thus, we can compute a ballpark cost of $(\$22797 \cdot 100) + (\$65.23 \cdot 13680) = \$3,172,047$ per Garage. For ten Garages, this is $31.7 million.

Let us consider also 4 large Garages with a capacity of 250 Autos each. Given a typical parking efficiency for a long-span Garage (300 square feet per space) and the same Auto space savings, we can expect to use 28500 square feet for 250 spaces in each of the four Garages. Thus, we can compute a ballpark cost of $(\$22797 \cdot 250) + (\$65.23 \cdot 28500) = \$7,558,305$ per Garage. For four Garages, this is $30.2 million.

The many small Garages configuration costs over $1 million more, but different distributions of Garages have a significant impact on service and platform earnings. It could be that the former configuration could be paid off much faster. We explore the impact of different Garage distributions in the sensitivity analysis.

Running simulations with fleets of 1000 Autos, we expect platform earnings surpassing $170,000 daily after paying for operational costs. Let us consider financing a set of Garages costing $32 million. If we finance a 25% down payment ($8 million) with 25% of that paid up front ($2 million) at a 6% interest rate over 8 years, we get a monthly payment of $78848.58, or 1.55% of our monthly earnings.[42]

$$6000000 \cdot \left(\frac{\frac{0.06}{12}(1 + \frac{0.06}{12})^{96}}{((1 + \frac{0.06}{12})^{96} - 1)} \right) = 78848.58,$$

Likewise, if we finance the remaining $24 million at 6% over 8 years, we get a monthly payment of $315394.33, or 6.18% of our monthly earnings.[42]

$$24000000 \cdot \left(\frac{\frac{0.06}{12}(1 + \frac{0.06}{12})^{96}}{((1 + \frac{0.06}{12})^{96} - 1)} \right) = 315394.33,$$

Thus, by paying 7.73% of earnings over 8 years, a platform can construct a suite of Garages - costing a total of $39.8 million - to store and fuel 1000 Autos. Then, the platform may earn the full $170,000 per day for the rest of the Garages' lifespans of up to 40 years.[43]

5.3 *Methodology & Results*

In the following sections, we compare four models:

1. **Batched CDV,** which resembles Uber where human drivers are matched to riders using Batched Matching;
2. **Planned CDV,** which resembles Didi Chuxing where human drivers are matched to riders using Didi's original Learning and Planning approach;
3. **Batched Auto,** which resembles Uber except that human drivers are substituted for Autos and Garages; and
4. **Planned Auto,** which uses our adaptation Didi's Learning and Planning with Autos and Garages to make rider matching, storage, idling, and redistribution decisions all in one unified framework.

In the plots below, CDV models are visualized in orange, and Auto models (referred to as SDV in the plots) are visualized in blue. Planned models are darker, and Batched models are lighter.

Before simulating, we select days from our historical data to train and test on. These are randomly selected according to necessary criteria. For example, we may select random weekdays (Monday-Friday) in the summer, or we may select random weekend days in the winter.

Our typical simulations apply Batched Matching to 168 hours (about 350,000 rides) of historical data in 7 distinct episodes. Individual vehicle objects - representing drivers in CDV models or Autos in Auto models - collect data attributed to individual vehicles and to the whole platform both by ride and by timestep.

After collecting these results, the learning algorithm iterates through each episode's state transitions to compute our initial state-values. Then, we start training our Planned model with another 168 hours of data in another 7 episodes, but instead of using Batched Matching, the centralized agent uses the Planning Step to make all fleet decisions including driver-rider pairing, storage, redistribution, and more. After each episode, the Learning Step updated the state-value function. We then simulate one last 168 hours of data in 7 episodes to test the state-values learned throughout the training process. The simulator reports results including platform earnings, driver earnings, pickup times, ride acceptance rates, vehicle miles traveled, and vacant time and distance among others. These Planned model results are then compared with the previously observed Batched model results.

We repeat this process to compare Autos to CDVs, Garage distributions, municipal regulations, fleet sizes, and more. Our first results considered a fleet of 975 Autos housed in 13 small Garages spread throughout Chicago.

Among our results depicted in Figure 16, the most striking is platform earnings where Auto models are expected to earn between than 90.6% and 99.0% more in Planned models and 90.4% and 96.9% in Batched models, a difference of between \$66,427 and \$83,679 per day.

Additionally, rides acceptance rates were slightly higher in Autos than CDVs compared to their Planned or Batched counterpart. Pickup time differences were negligible in improving the passenger experience, improving at most by 21 seconds on average with the Planned Auto model on summer weekdays. The cost of those improved earnings, acceptance rates, and pickup times were slight upticks in Vehicle Miles Traveled (VMT) with the greatest difference being 44,703 more miles driven by Autos than CDVs in the Batched model on summer weekdays. However, given the ability of Autos to use high-level control to optimize braking and reduce fuel consumption by up to 20%, even that worse case in VMT could represent emissions savings of up to 15,672 conventionally driven miles by switching to Autos.

We also observed that Autos spend more time vacant, but the Planned Auto model travels fewer miles vacant. The greatest average difference in vacant time by Autos was 43 more hours vacant per winter weekend day. The greatest average savings of vacant distance was 6,778 miles per summer weekday by the Planned Auto model compared to its CDV counterpart.

In the next section, we vary our inputs to evaluate how results change in different circumstances.

5.4 *Sensitivity Analysis*

We determine the dependency of the models' outputs on the models' inputs by varying input parameters. In this section, we focus on the fleet size, operational costs including the fuel cost, an imposed cost on idling, different Garage configurations, and a "regulation" to keep inactive

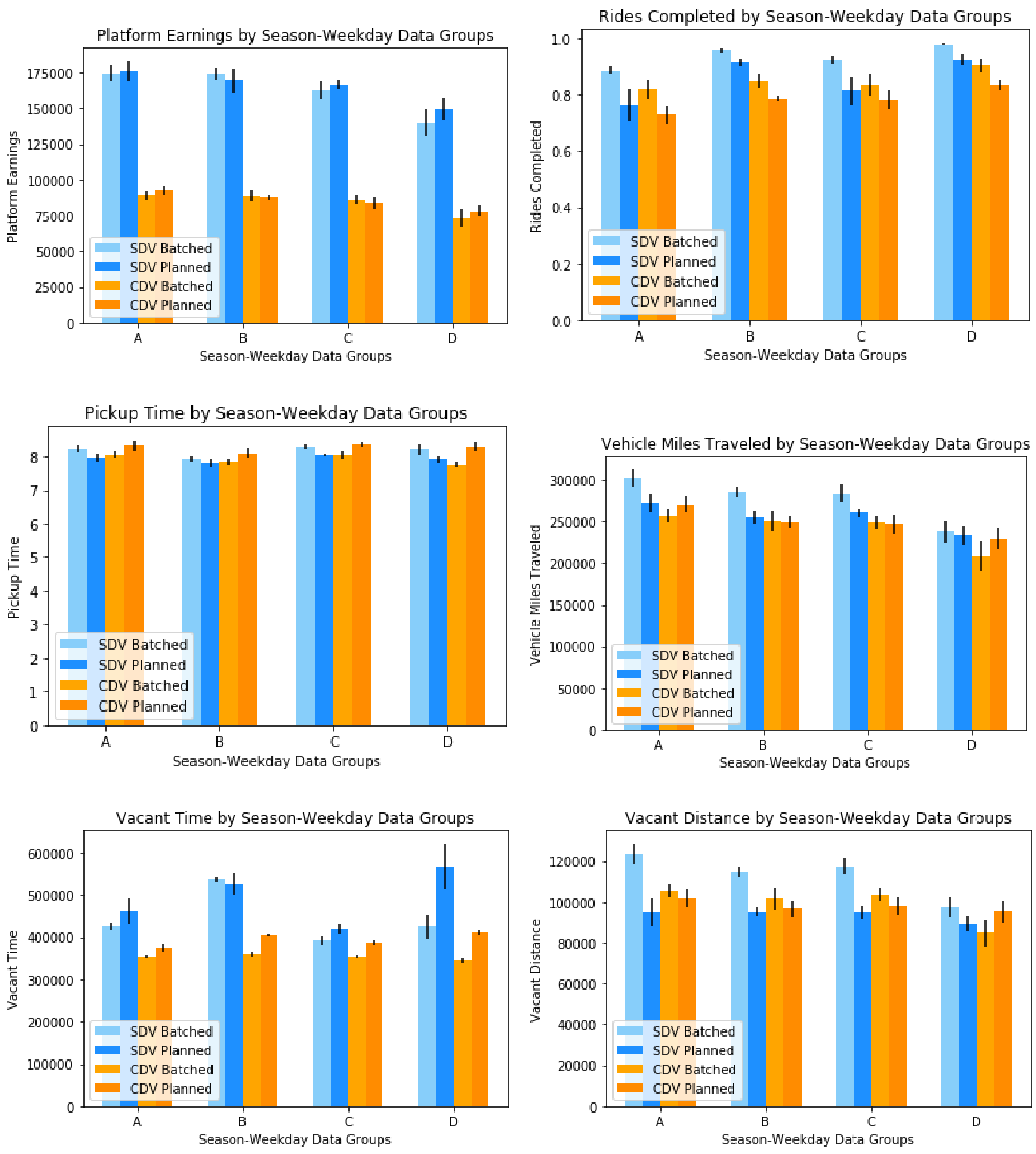

Figure 16: Auto and CDV Results by Weekday-Season Data Groups

Autos out of the streets. We will compare the four models - Planned Auto, Planned CDV, Batched Auto, and Batched CDV - by the average differences in key metrics as well as the likelihood of one model exceeding another in any metric. We arrive at those probabilities by treating the two compared models as normal random variables - e.g., $X, Y \sim \mathcal{N}(\mu, \sigma^2)$ - with their observed means and standard deviations and then calculating $P(X - Y > 0)$.

5.4.1 *Fleet Size*

In Auto models, "fleet size" refers to the number of Autos in the whole fleet, active and in storage. In CDV models, it refers to the maximum number of drivers allowed at any given time. Each key metric appears sensitive to the fleet size.

Auto platforms earn significantly more than CDV platforms. At 400 vehicles, this difference starts at around $45,000 daily varying slightly depending on which Auto model and which CDV model you compare. This difference increases significantly until both earning rates level off at 1600 vehicles where Auto platforms outperform CDV platforms by $85,000. The Planned Auto model consistently outperforms the Batched Auto model with probabilities as high as 99.997% certainty initially to as low as 51.19% at 800 vehicles. The Planned CDV model initially outperforms the Batched CDV model at 400 vehicles with 99.86% certainty, but the Batched model surpasses the Planned model after 800 vehicles and ultimately outperforms the Planned model 73.43% of the time. These trends are visible in the top left quadrant of Figure 17

Batched models accepted a higher percentage of ride requests than Planned models consistently between 90.17% to 99.91% of the time. The same was often true of Auto models to their CDV counterparts except as they converge to acceptance rates between 90% and 100%. As expected, all models accepted more rides with a greater fleet size capable of servicing more demand. Despite accepting fewer rides, we expect that the Planned Auto model was able to earn more than the Batched Auto model because it is more selective in which rides it accepts, focusing on the longer, more profitable rides.

Among pickup times, we see different behavior between Planned and Batched models. Batched models initially had lower pickup times at lower fleet sizes as Planned models are willing to travel farther to select more profitable rides. As fleet sizes grow in size and ability to service more demand, Planned models started to outcompete Batched models in pickup times even as they prioritize profits given their ability to proactively maintain supply-demand balance. Planned Auto models passed both Batched models around 900 vehicles, but Planned CDVs didn't accomplish the same until 1200 vehicles. The Planned Auto model earned the lowest average pickup time in the whole set at 7 minutes and 32 seconds.

With Vehicle Miles Traveled (VMT), we observed comparable behavior to pickup times where Planned models drove more miles than Batched models. At 400 vehicles, the Planned Auto model exceeded the Batched Auto model 99.97% of the time with an average difference of 12,936 miles daily. Likewise, the Planned CDV model exceeded the Batched CDV model an expected 100.00% of the time, by 23,842 miles on average. At 400, Batched CDV drove the fewest miles with an average of 122,901 miles daily compared to Planned Auto, the worst offender, with 153,629 miles. However, by 1600 vehicles, there is another complete reversal where Planned models drive more efficiently than Batched models. The Planned Auto model drove more efficiently than the Batched Auto model 63.41% of the time with an average difference of 11,715 miles. The worst offender at 1600 vehicles was the Batched CDV model which drove more miles than Planned CDV 71.94% of the time with an average difference of 14,972 miles.

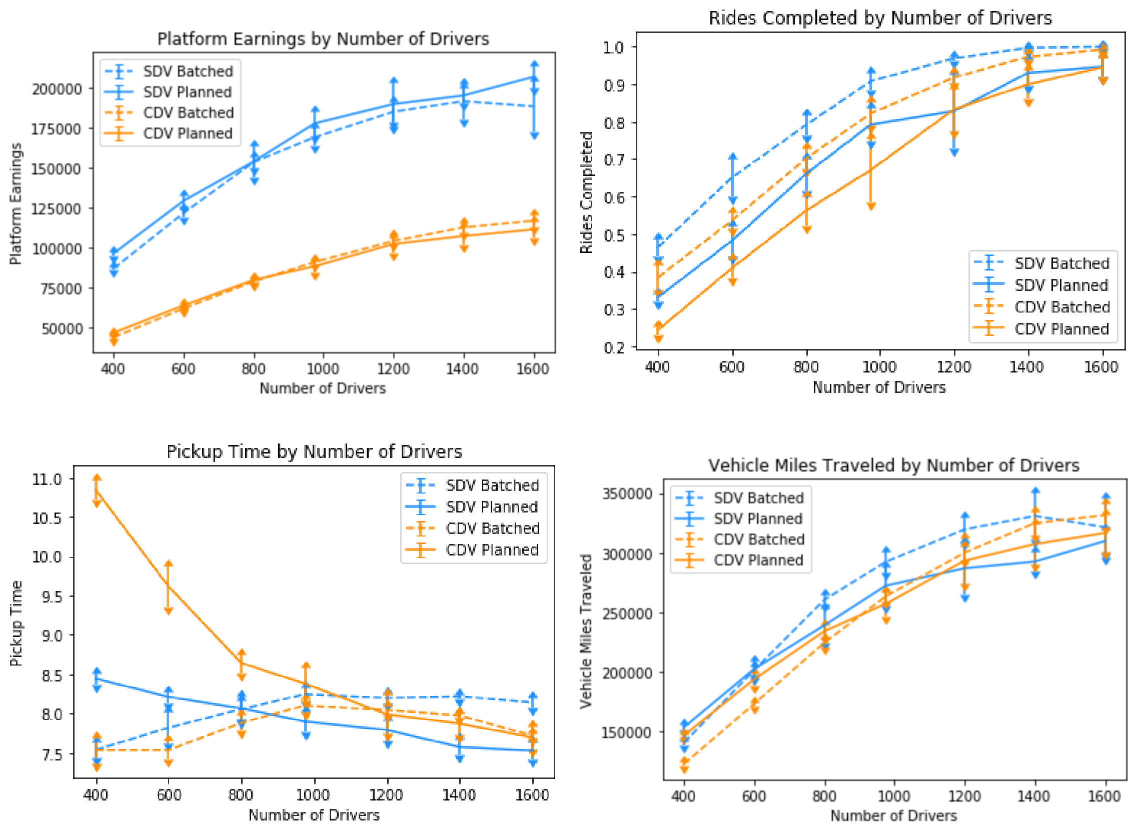

Figure 17: Impact of Number of Drivers or Autos on Key Metrics for Urban Mobility

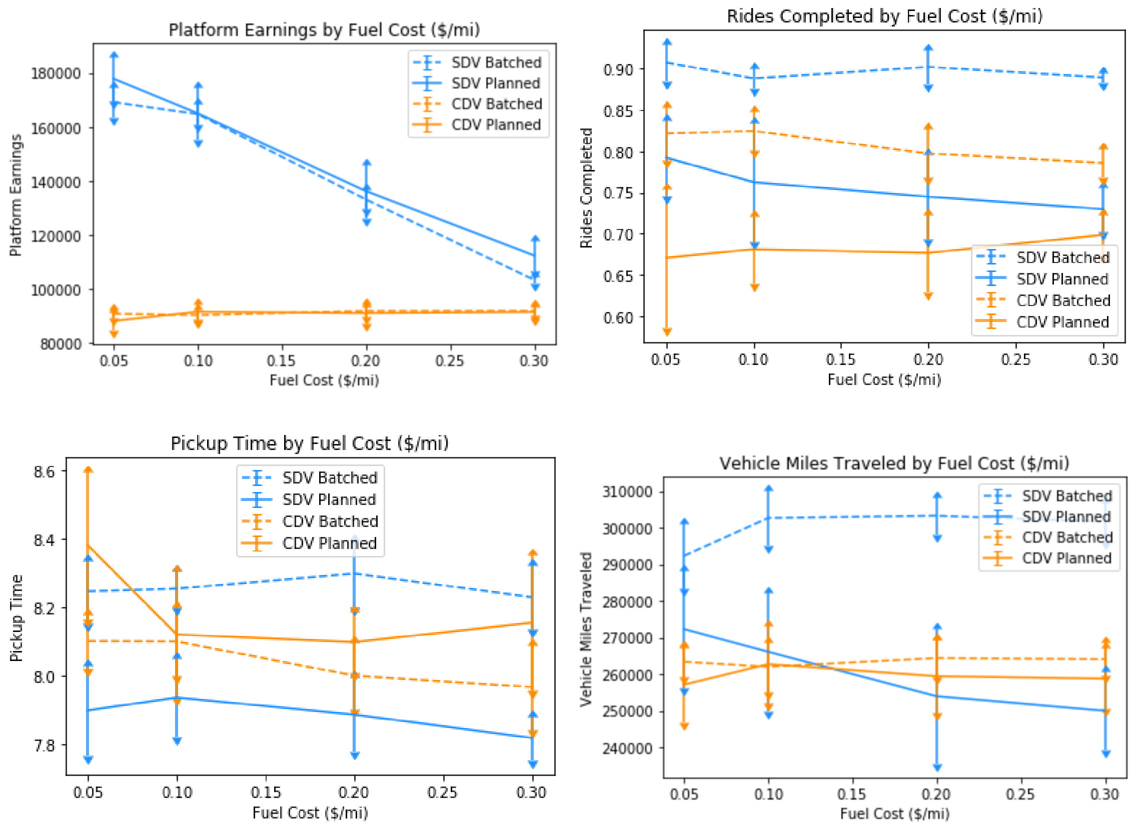

Figure 18: Impact of Fuel Cost per Mile on Key Metrics for Urban Mobility

With a fleet size at or above 1000 vehicles, the Planned Auto model may be expected to earn the most and pick up passengers in the least amount of time while also driving the fewest miles compared to alternative configurations.

5.4.2 *Operational Costs*

Among our operational costs, we first considered fuel cost per mile. The fuel cost we used in our first model results was $0.05 per mile as explained in the section on operational costs. This corresponds to a 40 MPG (City) vehicle paying $2.00 per gallon for gas as was typical in Chicago in 2016 at the time of data collection as well as now. Increasing the gas price to $4.00 per gallon for the same vehicle results in a fuel cost of $0.10. We increase the fuel cost up to $0.30 per mile - corresponding to a gas price of $12 per gallon - to demonstrate its impact on the model.

It primarily impacts platform earnings for Auto models. Unlike in the CDV models, the platform pays all operational costs in Auto models instead of contracted drivers. However, even at outrageous gas prices, Auto models earned significantly more than CDV models. Also noticeable was the Planned Auto model's tendency to accept fewer rides, drive fewer miles, and pick up nearer passengers in an effort to save on fuel. These trends are apparent in Figure 18.

After considering fuel costs alone, we considered Auto operational costs as a group. Planned models consider operational costs in determining driver-rider pairings and other actions, but in CDV models, the contracted drivers pay the costs, not the platform as is the case in Auto models.

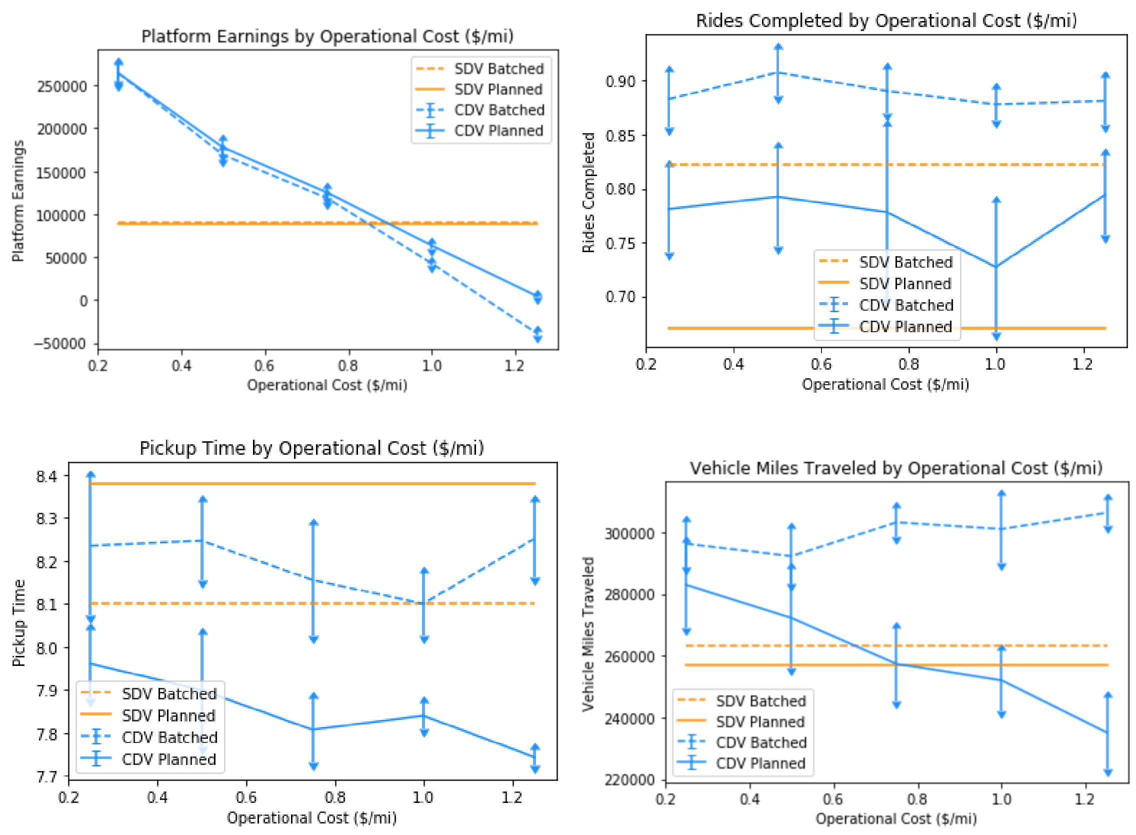

Figure 19: Impact of per-mile Operational Costs on Key Metrics for Urban Mobility

Thus, we only vary operational costs for Auto models but show CDV levels at a fuel cost of $0.05 as a comparison to the Auto models. The trends described below are visible in Figure 19.

Because Auto platforms pay operational costs, their earnings are very sensitive to cost adjustments. We anticipate total operational costs per mile to be $0.46. It is not until costs exceed $0.80 that CDV models begin to outperform Auto models in earnings.

Beyond earnings, the quality of the service does not change significantly. The percentage of rides accepted does not change drastically between $0.20 and $1.25 per mile. We do see the the Planned Auto model tends to drive more sustainably as costs increase. Between $0.20 and $1.25 per mile, this is apparent in slightly lower pickup times 98.7% of the time by an average of 13 seconds as well as lower VMT 99.9% of the time by an average of 47,830.77 miles.

5.4.3 *Garage Distribution*

We tested the Auto models using three different distributions of Garages using a fleet size of 600 while keeping the same total capacity across garages. They can be seen in Figure 24 in the Appendix.

1. **Distribution #1** has 13 small (Capacity 80) Garages spread evenly throughout the city of Chicago.

Metrics for the Planned Auto Model by Garage Distribution

	#1	#2	#3
Earnings ($)	129,193	126,126	120,641
Accept Rate	48.2%	52.8%	49.4%
Pickup Time	8m 13s	8m 1s	8m 18s
VMT (mi)	202,444	195,377	191,799

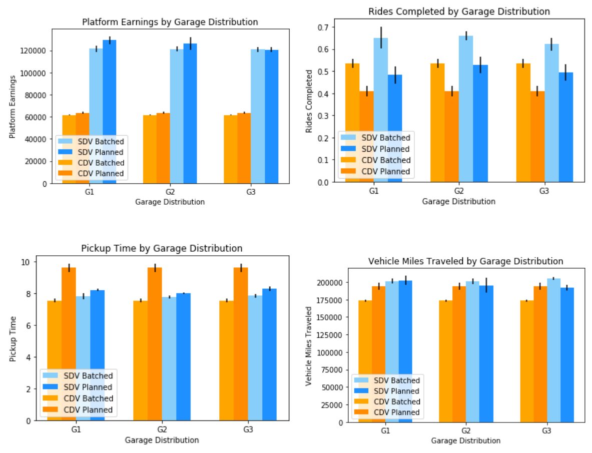

Table 7: Urban Mobility Metrics by Garage Distribution

2. **Distribution #2** has 8 larger Garages (Capacity 130) clustered towards the city center. This may be preferable if it is advantageous to have Garages in close proximity with large demand.

3. **Distribution #3** has 8 larger Garages (Capacity 130) primarily closer to the edges of the city. This may be preferable if it is advantageous for Garages to be closer to the least accessible demand.

We found that the results were not very sensitive to these Garage spatial distributions. For the Planned Auto model, Distribution #1 stands to earn an average of $3,067 and $8,552 more than #2 and #3 respectively. Distribution #2 accepts a slightly larger percentage of rides and can reduce pickup times by a more than 12 seconds on average compared to the others. Distribution #3 drove 10,645 miles fewer than #1.

By varying location, starting allotment, and capacity, there are countless ways to distribute Garages around the city. We expected a more significant result from just this small selection due to their geographical and capacity differences, but it may be necessary to try more extreme options to see drastic changes. With more time, we would investigate other distributions like a small Garage in every Community Area or just one or two very large Garages in the middle of the city. We would also investigate distributions where the total capacity is more constraining. Lastly, we would attempt these tests with greater fuel costs and fleets of varying sizes.

5.4.4 *Idle Cost*

We chose to impose a cost on idling Autos in an attempt to investigate its impact on ride acceptance as well as time and distance spent vacant. Because Batched models don't consider cost in matching decisions, they remain unchanged through idle cost differences. The Planned CDV model does start to accept slightly more rides, but the result is not positive. Earnings do not improve, pickup times get significantly longer - increasing on average by 29 seconds for CDV models and 36 seconds for Auto models - and also driving many more miles per day - increasing on average by 12,733 miles for CDV models and 25,839 miles for Auto models.

We also consider here two other metrics: vacant time and vacant distance. Vacant time is the amount of time that drivers our Autos spend active (i.e., not in storage) but without a passenger.

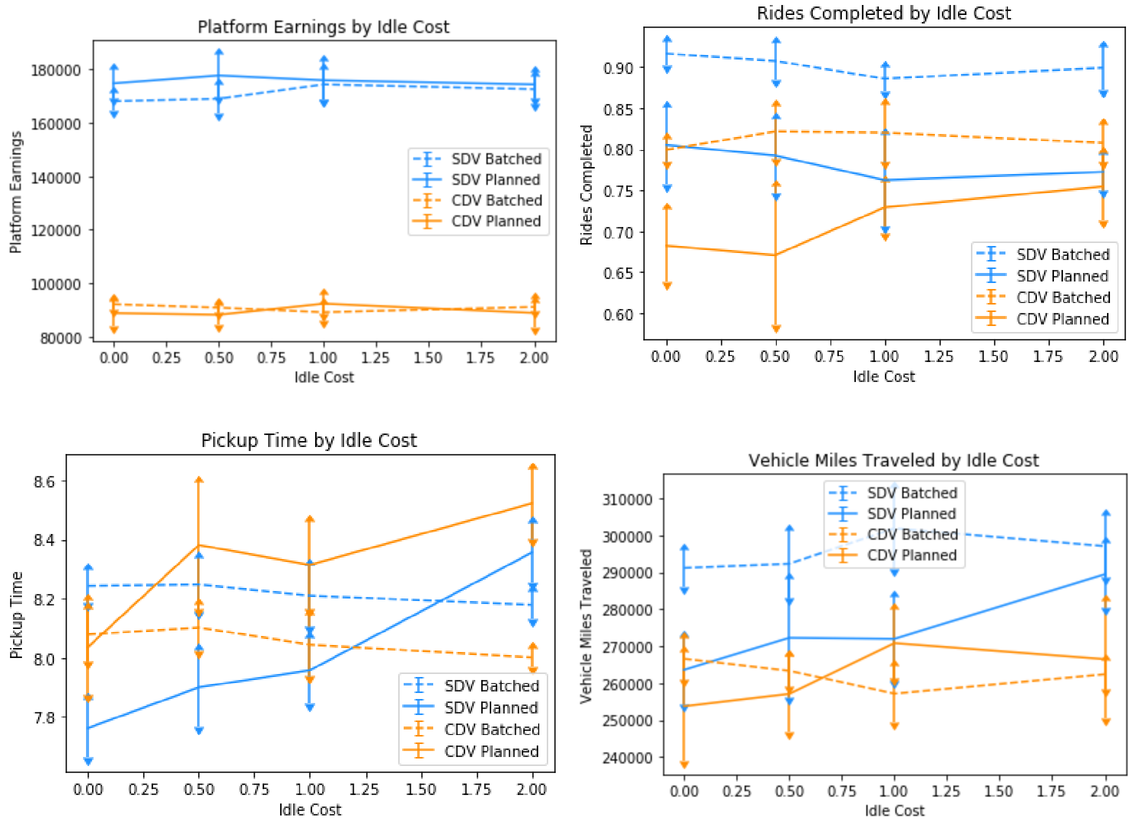

Figure 20: Impact of Idle Cost on Key Metrics for Urban Mobility

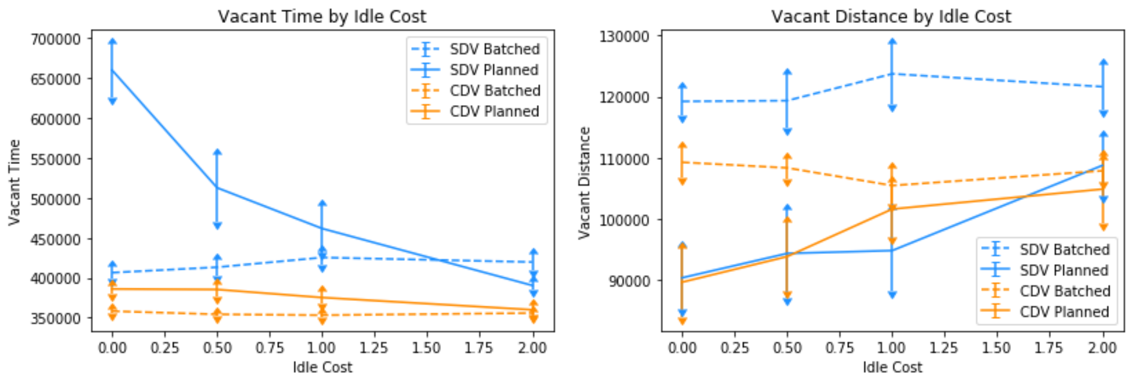

Figure 21: Impact of Idle Cost on Vacant Time and Vacant Distance

This time is spent either idling or driving to pick up a rider. Vacant distance is distance driven by drivers or Autos without a passenger in the vehicle. Idling vehicles do not move, so this only captures distance traveled to pickup locations and storage.

We see in Figure 21 that imposing an idle cost does drastically decrease vacant time for Planned Auto models by nearly half from 183.4 hours to 108.4 hours daily. It is apparent that without an imposed cost to idling, Planned Auto models spend dramatically more time idling. This manifests itself in Autos parked at the roadside awaiting a match. This could exacerbate already difficult parking situations in many cities. An idle cost would alleviate this.

We also see in Figure 21 that imposing an idle cost increases the vacant distance for Planned models, both increasing from about 90,000 miles vacant to over 100,000 miles vacant daily. It is apparent that Planned models spend significantly less distance vacant compared to Batched models - beyond 93.39% of the time for Auto models and beyond 65.92% for CDVs. Increased vacant distance manifests itself in increasingly congested streets with no economic benefit - contracted drivers driving with no passenger or Autos driving with no person at all in the vehicle. Imposing an idle cost could exacerbate already difficult traffic congestion problems in many cities.

This illustrates the trade-off between traffic and parking. Some implementation choices may lead to clear streets but expensive and impossible-to-find parking. Others might allow cities to reduce parking minimums and reclaim street parking space, but traffic congestion would worsen.

5.4.5 *Regulations*

In an effort to explore the trade-off posed above between traffic and parking, we implemented a hypothetical "regulation" by the city that demands that unmatched Autos never idle in or cruise the city streets awaiting a request. Instead, any idle Autos must return to a Garage to await orders. This could potentially remove empty Autos clogging up the streets and parking space.

The results for the Planned Auto in Figure 22 visualize this trend. As idling becomes more costly and when it becomes prohibited, vacant time decreases. This means that Autos won't park by the side of the road waiting for rides. Vacant time was greater with no cost to idling 100% of the time compared to the regulated option with an average difference of 74 hours daily. As idling became more difficult - with costs increases and with the regulation - vacant distance increased, meaning that empty Autos are filling up the streets more and worsening traffic. Vacant distance was greater with the regulated option 100% of the time compared to the no cost option with an average difference of 29,138 miles daily.

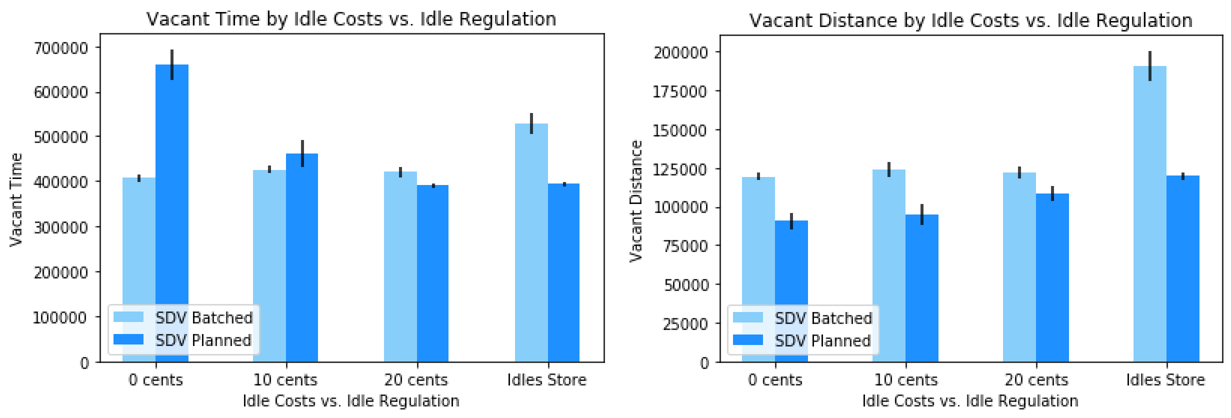

Figure 22: Easy idling leads to congested streets. Difficult idling leads to wasted miles.

Batched Matching does not consider costs in matching decisions, but the idle regulation increases vacant time by 31 hours daily and increases vacant distance by 71,670 miles daily on average. Without the ability to change its behavior like the Planned model, a disruptive regulation such as this significantly hinders the ride-hailing service.

The idle regulation does hinder the service, but if a platform or city government can find a desired balance of vacant time and vacant distance, imposing the regulation may be worth the cost. Profits decrease with the regulation 66% of the time with an average loss of $3,685 daily. Pickup times increase by 38 seconds on average.

5.4.6 *Training*

We found that our models were very insensitive to the time spent training. We trained our Planned Auto model for a **short** duration (200,000 rides over 96 simulated hours), a **medium** duration (450,000 rides over 216 hours), and a **long** duration (700,000 rides over 336 hours). Our results did not show any clear trends with greater training lengths.

When we did observe slight differences in performance in the Planned model, we also observed coincident similar differences in the Batched model. Because the Batched model does not rely on learned state-values, this indicates that the results are likely not due to changes in training duration but possibly due instead to the particular days selected to train on. Although we attempt to control for this using data groups based on weekday and season, there is still great variation in the number and distribution of rides even within days of the same weekday and season.

It is possible that the centralized agent in the Planned model learns state-values well enough within the short training duration to offer better service than the Batched model. It is possible also that the performance of the Planned model could improve significantly more if we considered much larger training durations. The improved performance of the Planned Auto model over the Batched Auto model, if not due to training, may instead be due to the structure of the Advantage function and the prioritization of expected future profits instead of pickup times. This apparent insensitivity to training does not falsify the above observations that the Planned model responds effectively to operational costs, fleet sizes, idle costs, and more.

With more time and resources, we would investigate other learning methods as well as smaller and greater training lengths.

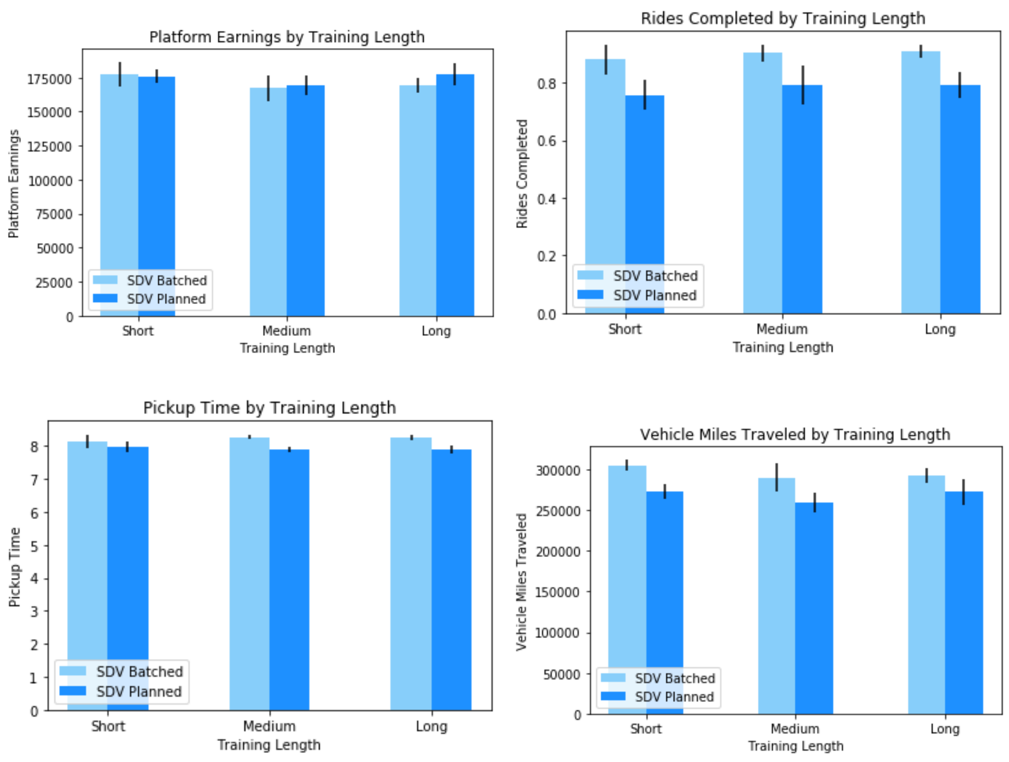

Figure 23: Performance Insensitive to Training Duration

5.5 *Error Analysis*

In this section, we analyze and provide explanations of what we expect were the main sources of inaccuracy in our models.

First, we believe that starting and ending rides in the center of Community Areas attributed nonneglible error to every ride. If we consider a ride from Lincoln Park to Uptown, the Google Maps API would inform us that the 4.4-mile ride will take 10 minutes or 20 timesteps, center to center. However, it's not true that each ride from Lincoln Park to Uptown is from the center of one to the center of the other. Rides between those areas could be as little as 1.7 miles in 7 minutes or as long as 6.3 miles in 16 minutes. A 58% swing in either direction is huge variability, and error like this exists on every one of approximately 50,000 rides per day in Chicago. We hope that because error deviates in both directions to a comparable degree that results may not be too affected, but this is not an adequate representation of the ride-hailing platform. In the future, we would explore using increasingly smaller zones.

Second, we typically use a 30 second batch time to reduce required computation. This is much longer than most platforms use and longer than most users would like to wait for a match. Additionally, by gathering more riders and drivers in each batch, it likely improves match results on average by increasing the number and variety of possible matchings in each timestep. Vehicles spend less time actively carrying out orders, but the orders carried out likely have improved pickup times, decreased vacant miles, and higher earnings per ride. In the future, we would investigate more deeply the impact of batch time on mobility metrics.

Third, reinforcement learning has a reputation for being intractable for most realistic applications because of the number of possible states. The agent must explore each state to a sufficient degree that it has enough knowledge about it to make decisions to successfully maneuver its way through states to maximize its reward. Our model considers 77 Chicago community areas as the spatial component to states which, as discussed above, contributes to error by being too few and too large. As its temporal component, our states consider each of a day's 24 hours as distinct. We recognize this as less than optimal in much the same way as our large, discrete Community Areas. Obviously, the traffic conditions downtown differ significantly at 5:01pm and 5:59pm. However, with 77 spatial components and 24 temporal components, we already have 1848 states for the agent to learn with sufficient accuracy to improve urban mobility over simpler models. If we included half hour states, that number would jump to 3696. If on top of that we wanted to include a simple 5-degree component representing the fuel level of a driver or Auto, we would have 18,480 states. So although there is more information we would like to capture in our model to make better decisions, adding detail quickly makes training intractable. Because of this, we hinder the ability of the Planned models to successfully direct agents to future states.

There are plenty of other adaptations we would have liked to consider whose absence could have contributed to model error. Our CDV models fluctuated in fleet size based on traffic conditions, demand, and shift lengths in order to mimic the decentralized decisions by drivers to log in and log out of the platform. We would have liked to alter this mechanism more to see how it impacts CDV model results and the ensuing comparisons. We would have liked to make rider decisions to renege after two timesteps more variable to account for an array of possible passenger behaviors. We would have liked to consider more breakdowns of the data than by weekday or weekend and summer or winter. Partitions like this help account for more information in the model without increasing the number of states. More considerations like these could have significantly improved model accuracy.

6 Conclusions

While there is much to be improved on this study, our results counter existing works that dismiss self-driving ride-hailing as an economically infeasible venture. Not only do our simulations demonstrate nearly double the earnings of conventionally driven ride-hailing fleets in Chicago in 2020, but our sensitivity analysis suggests that our Planned models are resilient to changing circumstances including outrageous gas prices, strict policy regulations, suboptimal supporting infrastructure, and more. This demonstrated economic competitiveness unlocks unknowable social and environmental potential - safer roads, cleaner air, and reduced emissions as well as quicker, cheaper, and more equitable transit. It is up to further research and proactive policy to realize this potential.

6.1 *Strengths*

- **Incorporating all decisions into the same framework:** When incorporating Autos into Didi Chuxing's Planning Step, we included all decisions - not just driver-rider matching - into the centralized decision-making network. We effectively adapted the framework to consider driver-rider matching, storage, refueling, idling, redistribution, and more all at once. This simultaneity allows for equal consideration of value throughout the decision-making process, enabling it to make the smartest possible choices across the board.

- **Considers many relevant metrics:** Our study does not focus only on the profitability of self-driving ride-hailing platforms. It considers pickup times and ride acceptance rates to capture the passenger's experience. It considers vacant time and distance to ensure that city streets are used efficiently and effectively. It considers vehicle miles traveled to ensure that changes to ride-hailing do not disregard emissions and air quality in dense urban areas.
- **No drivers and no other changes:** To make the self-driving platform as comparable as possible to existing platforms, we restrict Autos to drive the same as conventional drivers and we price each ride using the same per-mile costs, per-minute costs, and base costs that Uber uses in Chicago. Thus, the transition from conventional to self-driving rides should be minimally disruptive to the passenger, and our results do not reflect other possible gains from inter-vehicle communication, lower fares, or anything else.

6.2 *Weaknesses*

- **Large Community Areas:** Restricting pickup locations and destinations to the center of Community Areas likely attributed nonneglible error to every ride. Although this is the best granularity we had available, it does not adequately model the ride-hailing environment to a satisfactory degree.
- **Hour-long states:** Like the above, temporally-long states fail to capture shapes in the demand profile that allow the centralized agent to make the best possible decisions. With more resources to handle a greater number of states, we would reduce the time granularity.
- **Four data groups:** We considered four subsets of the dataset based on the intersection of weekdays or weekends with summer or winter. This helped account for characteristic demand profiles that shift by day and demand levels that shift by season, but with more data groupings, we could account for more information without drastically increasing the number of states. With more time, we would consider groups by single days of the week, months, holidays, and even weather.
- **Fuel level not considered in state:** To keep the number of states manageable, we could not include fuel information within each vehicle's state. Storage decisions were made knowing that vehicles do need to refuel regularly but not with the ideal specificity. With more resources, we would include fuel information in the vehicle states.
- **Insensitivity to training:** Our results were achieved without a clear picture of how state-values and ride-hailing performance depend on training. With more time and resources, we would investigate other learning mechanisms and training durations.

6.3 *Policy Recommendations*

Our study demonstrates the feasibility and significant profitability of self-driving ride-hailing fleets. Without prior action taken by policymakers and urban planners, platforms will deploy their self-driving services - likely without permission - without regard for the priorities of the community. If these services are unwelcome, cities must take action in advance to prevent their appearance. However, if these services are welcome, the city stands to gain by playing a part in the transition.

Cities should take a proactive role in transitioning to Auto-based ride-hailing services. There are implementation decisions - e.g., regulations, idle costs, and Garage distributions - that significantly impact the quality of self-driving ride-hailing services. If cities wish to improve urban mobility by maximizing the benefits and mitigating the harms of high-tech trends, they must make these decisions consciously and thoughtfully in advance.

If urban planners and policymakers want to reduce traffic congestion and parking minimums, maximize safety and public health, and minimize infrastructure costs, then they should put the first foot forwards in engagement with service providers, explore opening their city up to self-driving vehicle testing, and conducting further research.

It is possible that people will not readily accept self-driving cars and that fear and uncertainty may significantly reduce demand to an unprofitable and unsustainable level. If the social and economic benefits of self-driving ride-hailing are desired, cities must take steps to accustom people to this new and unfamiliar practice.

Our study uses historical data set in stone, but in reality, demand shifts as the service changes. If low-density transit becomes cheaper and more efficient, will people simply rely on it more causing traffic to worsen and air quality to suffer? Or will people use the service to connect them with high-density transit options and further engage the cities' most efficient options? It is possible that cities may employ incentive structures that favor one outcome over the other if transition decisions are made deliberately.

7 Future Considerations

Through the course of the research, other questions arose that warrant study of their own. With more time and resources, the following are among other questions we would like to explore.

TRANSITION MODELS: Our study compares completely autonomous fleets with completely traditional fleets of human drivers. There is no reason to assume that such a complete transition is desirable or even possible. We would like to explore how ride-hailing fleets might utilize an autonomous subcomponent to improve urban mobility and how might service improve as the autonomous proportion increases. Additionally, using a ramp-up deployment might help accustom people to autonomous cars and prevent the demand avoidance inevitable in an instantaneous and complete transition.

OTHER MARKETS: Our study takes Chicago as a case study, simulating on-demand ride-hailing among its 77 Community Areas using a dataset of Chicago taxi rides. Due to economic, social, and geographical differences, cities likely respond differently to the same service. Also, Chicago is notably one of the cheapest cities in America for ride-hailing services, and there is no obvious reason to accept on its face that urban mobility improvements would be representative of other urban environments across the country and the world.

BEHAVIOR INDUCTION AND DEMAND RESPONSE: The problem with using an historical dataset is that demand cannot respond to shifts in service. Different prices, pickup times, and even the lack of a driver don't impact the inflow of ride requests. Smart systems like our Planned Auto model are excellent at tailoring the service to meet demand, but enabling existing demand patterns may not be productive. Existing demand - e.g., the after work peak in traffic - is not necessarily sustainable, efficient, or conducive to good service. With more time and resources, we wish to explore designs of smart systems that use incentive structures to induce more sustainable behavior.

BEYOND THE CITY: Our model operates only within the city limits, but the commute - largely to and from the suburbs - makes up a large proportion of US transit-related emissions as well as a large proportion of the lives of workers. Managing a service based on connecting supply to demand becomes much more difficult in sparse environments and may require more creative thinking. Additionally, we are interested in exploring how autonomous ride-hailing might carry over to long-distance travel, between cities or even across countries.

References

[1] National Association of City Transportation Officials. Shared micromobility in the u.s.: 2018, 2018.

[2] Uber Marketplace. How does uber match riders with drivers?, 2020.

[3] Z. Xu, Z. Li, Q. Guan, D. Zhang, Q. Li, J. Nan, C. Liu, W. Bian, and J. Ye. Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 905–913, 2018.

[4] Jeff Ferenc. Led lighting improves safety, cuts energy in hospital parking garage, 2016.

[5] Chicago Digital. Chicago taxi data released, 2016.

[6] James Derek Sapienza. The most popular cars for uber drivers (and their passengers), 2018.

[7] United Nations Department of Economic and Social Affairs Populations Division. World urbanization prospects 2018: Highlights, 2019.

[8] Brian Martucci. What is bike sharing? – 10 best bike share programs in north america, 2015.

[9] Megan Rose Dickey. The electric scooter wars of 2018, 2018.

[10] Bruce Schaller. The new automobility: Lyft, uber and the future of american cities, 2018.

[11] Romic Aevaz. 2018 acs survey: While most americans' commuting trends are unchanged, teleworking continues to grow, and driving alone dips in some major cities, 2019.

[12] Environmental Protecton Agency. Fast facts on transportation greenhouse gas emissions, 2017.

[13] Tomio Geron. California becomes first state to regulate ridesharing services lyft, sidecar, uberx, 2013.

[14] American Public Transportation Association. 2019 public transportation fact book, 2019.

[15] Robert Hahn and Robert Metcalfe. The ridesharing revolution: Economic survey and synthesis, 2017.

[16] Yanbo Ge, Christopher R Knittel, Don MacKenzie, and Stephen Zoepf. Racial and gender discrimination in transportation network companies. Working Paper 22776, National Bureau of Economic Research, October 2016.

[17] Tim Adams. Self-driving cars: from 2020 you will become a permanent backseat driver, 2015.

[18] Business Insider. 10 million self-driving cars will be on the road by 2020, 2016.

[19] Uber Marketplace. Tesla's musk is overpromising again on self-driving cars, 2020.

[20] Kelsey Piper. It's 2020. where are our self-driving cars?, 2020.

[21] Ashley Nunes and Kristen D Hernandez. Autonomous vehicles and public health: High cost or high opportunity cost?, Apr 2019.

[22] Matthew Barth and Kanok Boriboonsomsin. Energy and emissions impacts of a freeway-based dynamic eco-driving system. *Transportation Research Part D: Transport and Environment*, 14:400–410, 08 2009.

[23] Andrei Zakhareuski. Uber driver requirements: 4 things to know to become an uber driver, 2019.

[24] Google Developers. Directions api: Developer guide, 2020.

[25] Michel Goemans. Lecture notes on bipartite matching, 2009.

[26] Philipp Afèche, Zhe Liu, and Costis Maglaras. Ride-hailing networks with strategic drivers: The impact of platform control capabilities on performance. *SSRN Electronic Journal*, 01 2018.

[27] Jim Dallke. Uber cheap: Chicago is the least expensive city in america for uberx rides, 2014.

[28] Brett Helling. Uber fees: How much does uber pay, actually?, 2020.

[29] City of Chicago. Taxi trips, 2020.

[30] Car and Driver. 2020 toyota prius, 2020.

[31] City of Chicago. Taxicab and medallion information, 2020.

[32] QuoteWizard. Compare auto insurance in illinois, 2018.

[33] Bureau of Transportation Statistics. Average cost of owning and operating an automobile, 2019.

[34] Patrick M. Bösch, Felix Becker, Henrik Becker, and Kay W. Axhausen. Cost-based analysis of autonomous mobility services. *Transport Policy*, 64:76 – 91, 2018.

[35] U.S. Energy Information Administration. Weekly retail gasoline and diesel prices, 2020.

[36] U.S. Energy Information Administration. Weekly chicago, il regular reformulated retail gasoline prices, 2020.

[37] JIA Jian-min, Jianmin Jia, Gregory W. Fischer, and James S. Dyer. Attribute weighting methods and decision quality in the presence of response error: A simulation study. *Journal of Behavioral Decision Making*, 04 1997.

[38] Gary Cudney. Parking structure cost outlook for 2015, 2015.

[39] Bill Kavanagh. Mixing it up: Financing and designing the most efficient and effective mixed-use projects., 2015.

[40] Mehdi Nourinejad, Sina Bahrami, and Matthew J. Roorda. Designing parking facilities for autonomous vehicles. *Transportation Research Part B: Methodological*, 109:110 – 127, 2018.

[41] CarInsurance.com. Here's what happens when you drive away from the pump with the gas hose attached, 2012.

[42] Tara Mastroeni. Everything you need to know about construction loans, 2018.

[43] K. Nam Shiu. Extending the service life of parking structures: A systematic repair approach, 2007.

8 Appendix

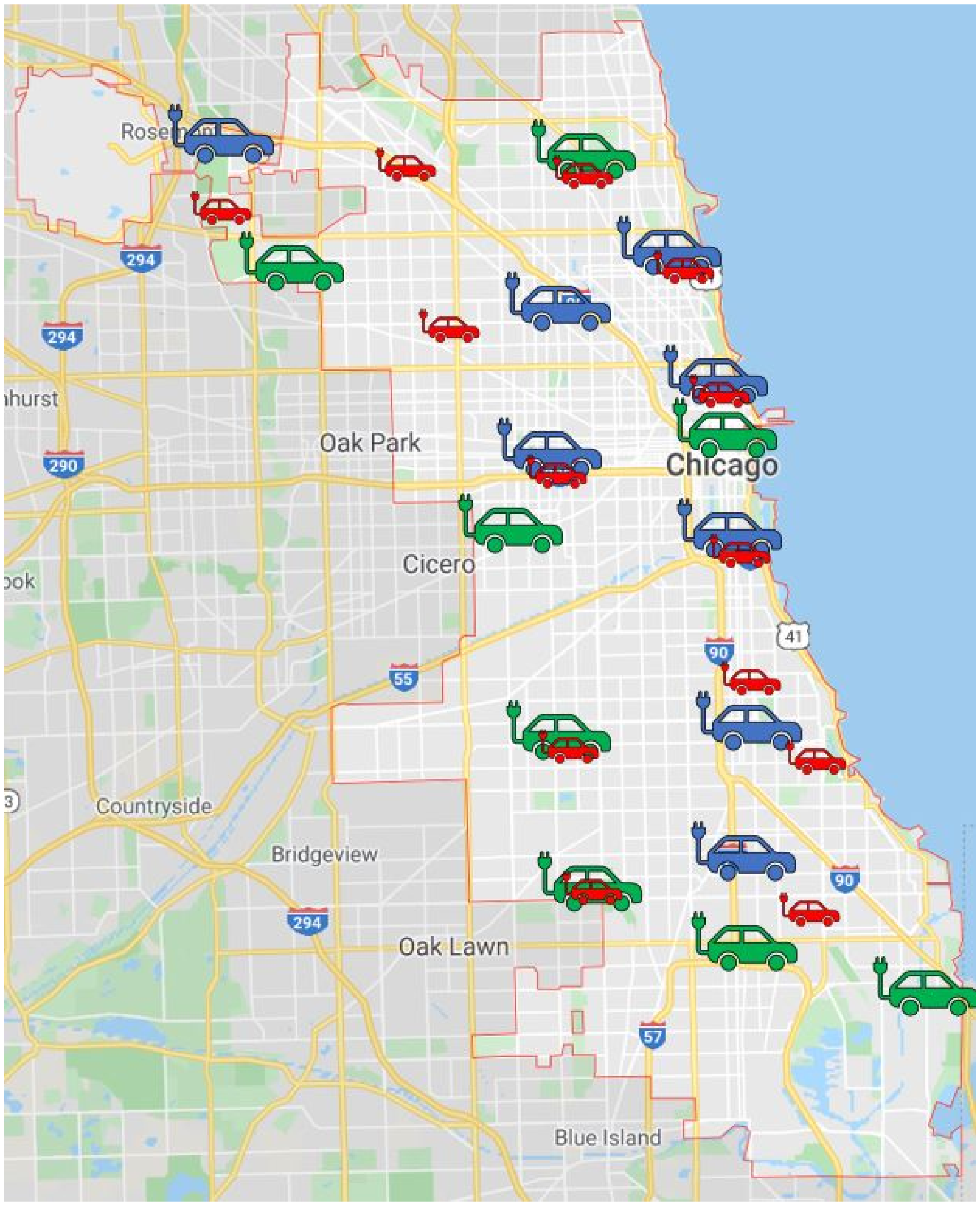

Figure 24: Garage Distribution #1 in Red, #2 in Blue, and #3 in Green