A Distributed Delivery-Fleet Management Framework using Deep Reinforcement Learning and Dynamic Multi-Hop Routing

Kaushik Manchella
Dept. of Industrial Eng.
Purdue University
kmanchel@purdue.edu

Marina Haliem
Dept. of Computer Sc.
Purdue University
mwadea@purdue.edu

Vaneet Aggarwal
Dept. of Industrial Eng.
Purdue University
vaneet@purdue.edu

Bharat Bhargava
Dept. of Computer Sc.
Purdue University
bbshail@purdue.edu

Abstract

With autonomous driving on the horizon, a crucial area of innovation is fleet management. The ubiquitous adoption of mobility-on-demand platforms and ridesharing is a scenario that will go hand-in-hand with fleet management specific to autonomous taxis and delivery vehicles. Alongside delivering passengers, ridesharing systems have shown to be a promising medium to address the growth in crowdsourced goods deliveries and last-mile deliveries. In addressing the problem of combined passenger and goods delivery through ridesharing and autonomous fleet management, intelligent transportation systems are being developed to maximize operational profitability, user convenience, and environmental sustainability. In such a scenario, it is challenging for vehicles to meet the expectation to fulfill dynamically arriving requests efficiently while satisfying existing delivery constraints in routing. Existing methods address this using static routing methods considering neither the demands of requests nor the transfer of customers between vehicles during route planning. In this paper, we present a dynamic and demand aware fleet management framework that is informed by a representation learned from Deep Reinforcement Learning. In our proposed model-free method, vehicles independently interact with the ridesharing environment and learn a spatio-temporal representation to anticipate regions of high reward on the map. We utilize this learned representation for various decision-making tasks including vehicle dispatching, route-planning, matching, and pricing. Our proposed model is deployable independently within each vehicle as this minimizes computation costs associated with the growth of distributed systems and democratizes decision-making to each individual. In a simulated ridesharing environment, our framework shows improved fleet utilization and vehicle profits in delivering passengers and goods over existing model-free methods.

Machine Learning for Autonomous Driving Workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

1 Introduction

1.1 Background

Mobility-on-Demand (MoD) platforms have grown to over 40 million users [1] and continue to grow. Alongside, crowd-sourced delivery has experienced over 300% growth in the past 2 years alone [2] [3] [4]. These platforms have three primary stakeholders: the ride-hailing offering company, the driver, and the customer. Customers seek convenience while drivers and the ride-hailing platforms seek profits [5]. Hence an ideal solution would be one that maximizes benefits for all these parties. Additionally, MoD services have proven potential to improve the efficiency and sustainability with vehicle sharing [6]. Urban traffic congestion, pollution, and energy expenditure are a few of the adverse effects of transportation systems that stand to be mitigated by the adoption of intelligent transportation systems [7]. Route planning for MoD services has also proven to be an opportunity for optimization [8] towards more profitable transportation models. Challenges that arise due to scaling the number of vehicles [9] also need to be addressed. Considering this, we present a distributed algorithm for fleet management where each vehicle can individually learn its policies and make decisions while contributing to the optimization of global fleet objectives.

Model-free approaches have been proven to be very effective in the space of urban mobility for vehicle dispatching as they are robust to erratic changes in the environment dynamics. MOVI [10] first introduced this distributed approach to vehicle dispatching using Deep Reinforcement Learning. The authors of [11, 12] expanded that to consider ridesharing and this significantly improved fleet performance in terms of acceptance rate, customer waiting time, and fleet utilization. Consequently, the authors of [13] presented an approach to extend this to vehicles delivering combined workloads of passengers and goods which improved fleet utilization and directly addressed the growing need for shared goods delivery along with passenger ridesharing. However, all of these approaches did not explicitly model pricing strategies which tend to be key drivers of the MoD marketplace. Additionally, the aforementioned approaches resort to naive matching approaches, leaving significant room for improvement in route-planning for vehicles. The authors of [14, 15] directly address these two aspects for passenger transportation but not for goods.

In this paper, we aim to build on all of these approaches by considering a combined workload of passengers and goods along with an improvement in route-planning with multi-hop considerations.

1.2 Contributions

The key contributions of the paper can be summarized as follows:

1. An entire decision-making framework for autonomous delivery vehicles that is distributed in nature to address demand workload from passengers and goods optimally
2. A DQN-based dispatch policy where vehicles learn a representation of regions of high anticipated reward. The dispatch policy re-balances idle vehicles to regions of high ancipated reward.
3. A demand-aware pricing strategy that leverages the DQN representation to allow vehicles to propose a suitable price to customers by considering regions of high anticipated reward.
4. A demand-aware route planning policy that leverages the DQN representation to inform vehicle's routing decisions to maximize potential future reward.
5. A multi-hop routing policy where the fleet vehicles iteratively move goods in chunks from the origin to destination, thereby utilizing excess capacity to meet goods demands along with passengers.

Using real-world taxi-trip, and goods-order datasets, we simulate experiments with our joint approach that shows improvements in (i) vehicle profits (ii) idle driving (iii) fleet utilization. We note that each of the components of our framework is crucial to optimizing the fleet objectives as a whole.

2 System Model

In this section, we will discuss the overall architecture and objective of the system.

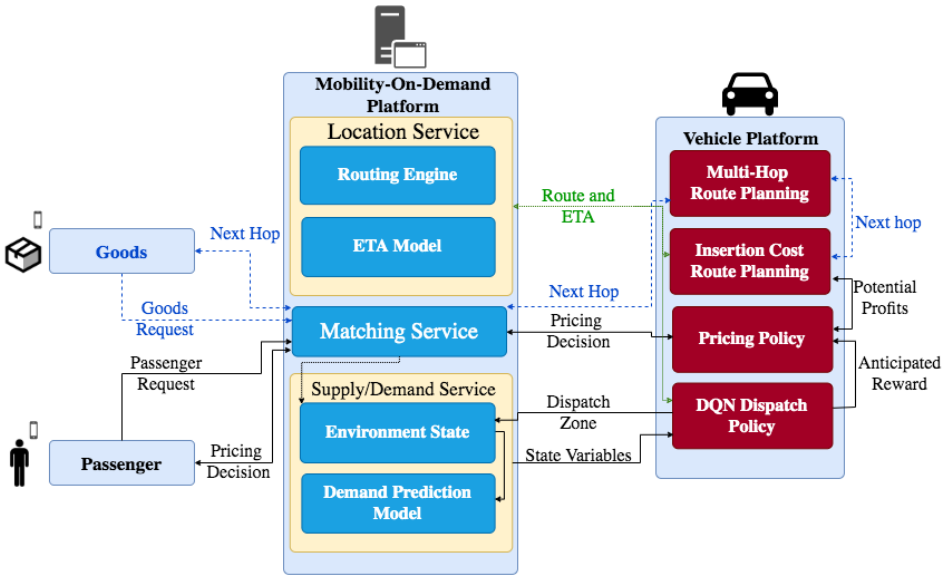

Figure 1: An illustration of the framework architecture. The image shows the various actors, services, and interactions.

2.1 Overview

As seen in Figure 1, the framework consists of 3 primary actors: customers (which may be in the form of passengers or goods delivery customers), vehicles, and the MoD (Mobility-On-Demand) Platform. Similar to a real-world scenario, the workflow involves the customer requesting the MoD platform for a pickup. This initiates an initial matching with a vehicle followed by a pricing decision based on the customer and vehicle's preferences. The MoD platform's main role is to act as an intermediary between customers and vehicles, thereby acting as a marketplace for delivery services. The services offered by the MoD platform include the matching service, location service, and supply/demand service. Each of these services exists to communicate information to customers and vehicles' respective decision-making tools. On the customer's end, if it is a passenger, we model a utility function to indicate the passengers' preferences on vehicle choice, urgency, and pricing. If the customer happens to be a goods delivery request, we proceed with a uniform preference and fixed pricing. The core innovation, however, happens on the vehicle end. Our proposed algorithm involves a set of components shown under the vehicle platform interacting with each other to maximize reward. As each vehicle seeks to maximize its reward, we optimize the overall fleet objective in a distributed manner. This way, the distributed decision-making algorithm also scales easily with the increasing number of vehicles. Leveraging the distributed computing power in vehicles, each vehicle is equipped with the following policies: 1. DQN Dispatch Policy 2. Pricing Policy 3. Insertion Cost Route Planning 4. Multi-hop Route Planning. In the vehicle model, each of these policies exchange information with the MoD Platform as well as amongst themselves to make optimal decisions on the following: 1. Which area of the map to dispatch to 2. How much to charge the passenger upon initial matching 3. Which routes to take in delivering passengers 4. Which next destination to delivery goods in their "hitchhiking" journey.

The DQN dispatch policy is the core component of the vehicle platform. The primary objective of this DQN is to learn the spatio-temporal distribution of supply/demand across the map and ultimately be able to predict which areas of the map to mobilize a given vehicle in order to gain the maximum possible rewards. As shown in Section 2.2, rewards are formulated by considering various components. On one hand, DQN Dispatch Policy utilizes this representation learned by the DQN to re-balance idle vehicles into regions of anticipated high rewards, while on the other hand, this DQN is used to inform other decision making components such as the Pricing Policy and the Insertion Cost Route Planning component. With the DQN equipping the vehicle with knowledge on which parts of the map yield the maximum reward, the vehicle is able to plan its route to pick-up and drop-off passengers in a manner to minimize the cost of moving away from regions of high reward. Likewise, the price proposed to the passenger may also consider this cost thereby balancing

the vehicle's incentives with the demand of the passenger. Finally, the Multi-Hop Route Planning is a component that is invoked when the vehicle encounters a goods delivery request. This route planning component aims to deliver the good to a "hopzone" to serve as a layover or directly to the destination depending on which is more convenient for the vehicle. At each iteration when a package is dropped to a hopzone, we ensure the hopzone that maximizes the package's distance towards destination. Section 3.2 discusses this process in further detail.

2.2 Objectives

In this section, we detail our system's global reward objective which allows efficient fleet dispatch in fulfilling service workloads of different kinds. This global reward is optimized by our proposed algorithm, in a distributed fashion as vehicles solve their own DQN (Deep Q-Network) to maximize rewards.

The goals of the system objective include: 1. satisfying the demand of pick-up orders, thereby minimizing the demand-supply mismatch, 2. minimizing the time taken to pick-up an order (aka pick-up wait time) in tandem to the dispatch time taken for a vehicle to move to a pick-up location 3. minimizing the additional travel time incurred by orders due to participating in a shared vehicle 4. minimizing the number of vehicles deployed to minimize fuel consumption and traffic congestion 5. maximizing vehicle earnings.

This overall objective is optimized at each vehicle in the distributed transportation network.

Algorithm 1 Framework Algorithm

Initialize vehicles' states X_0 and t_0
for $t \in T$ **do**
    Fetch all vehicles that entered the martket in time slot t
    Dispatch V_{new} to zones with anticipated high reward
    Fetch all ride requests at time slot t, D_t
    Fetch all available vehicles at time slot t, V_t
    for each vehicle $V_j \in V_t$ **do**
        Obtain initial matching A_j
        for each request $r_i \in A_j$ **do**
            if request is passenger **then** :
                Obtain initial pricing from MoD Platform
                Perform Insertion-cost Route Planning
                Propose final price to customer in negotiation
                Get customer i's final decision
            else if request is good **then**:
                if vehicle capacity is empty **then**
                    Obtain initial pricing from MoD Platform
                    Perform Insertion-cost Route Planning
                else
                    Assign next destination using multi-hop assignment policy
                    Perform Insertion-cost Route Planning
            Update State Vector s_t
        Retrieve next stop from route plan
        Drive to next stop
    Dispatch idle vehicles with Idle_duration $>$ 10 minutes to zones with anticipated high reward
    Update State Vector s_t

3 DQN Informed Decision-making

As each vehicle in the fleet is input the state of the ridesharing environment, including the spatio-temporal distribution of supply and demand, it learns a representation that maps this environment state to the regions of highest reward in terms of demand and earnings. In this section, we present how this representation is learned and how it is utilized for further decision-making on routing, matching,

and pricing. Please note that each vehicle is running the model locally resulting in a distributed decision-making algorithm across the entire transportation system.

As mentioned previously, the core component of our algorithm is the representation learned from the DQN Dispatch Policy which is described in Section 3.1. Subsequent steps in decision-making which use this representation include demand-aware pricing which is adopted from DPRS [16]. Here, the most optimal price for a given vehicle is proposed to a passenger based on the region where the passenger requests a drop-off. Regions of low-anticipated reward yield a higher proposed price as there is a greater cost to the vehicle. We use the DQN representation in route planning by adopting the DARM [14] insertion-cost based route planning strategy. Here, the path of the vehicle is planned in a manner to adhere to regions of high anticipated rewards. Finally, we further explain the core intuitions of the multi-hop route planning component in Section 3.2

3.1 DQN Dispatch Policy

The fleet of autonomous vehicles was trained in a virtual spatio-temporal environment that simulates urban traffic and routing. In our simulator, we used the road network of the New York City Metropolitan area along with a realistic simulation of taxi pick-ups and package delivery requests. This simulator hosts each deep reinforcement learning agent which acts as a delivery vehicle in the New York City area that is looking to maximize its reward.

Below, we describe the state, action, and reward formulated for the Q-Learning procedure of the DQN dispatch policy:

State: The state variables defined in this framework capture the environment status and thus influence the reward feedback to the agents' actions. We discretize the map of our urban area into a grid of length $\boldsymbol{x}$ and height $\boldsymbol{y}$; resulting in a total of $\boldsymbol{x} * \boldsymbol{y}$ zones. This discretization prevents our state and action space from exploding thereby making implementation feasible. The state at time t is captured by following tuple: $(X_t, V_{t:T}, D_{t:T})$. These elements are combined and represented in one vector denoted as s_t. When a set of new ride requests are generated, the simulator engine updates its own data to keep track of the environment status. The three-tuple state variables in s_t are passed as an input to the DQN input layer which consequently outputs the best action to be taken. 1. X_t will track vehicle seating capacity and trunk space for goods/packages. $C_{p,v}$ and $C_{k,v}$ respectively. As a result X_t will track: *current zone of vehicle v, available seats, available trunk space, pick-up time of delivery order, destination zone of each order*. 2. $V_{t:T}$ is a prediction of number of available vehicles at each zone for T time slots ahead. 3. $D_{t:T}$ has a term $\delta_{t:T}$ that predicts joint demand of passengers & goods delivery orders at each zone for T time slots ahead.

Action: The action of vehicle n is denoted by $a_{t,n}$ decides the zone it should be dispatched to at time slot t, which is given as $Z_{t,i}$.

If vehicle n is full it can not serve any additional customer. Alternatively, if a vehicle decides to serve current customers, the shortest route along the road network is used to pickup the assigned orders.

Reward: Below is the reward function $r_{t,n} = r(s_{t,n}, a_{t,n})$ which is used at each agent n of the distributed system at time t. The weights shown in equation (1) (below) are used only in instances when an agent is not fully occupied and is eligible to pickup additional orders. The reward function is formulated as a linear combination of the following terms along with there respective weights β_i: (1) total number of customer orders picked up $b_{t,n}$ denotes the number of passengers served by vehicle n at time t and $p_{t,n}$ denotes the number of packages being carried in the trunk of vehicle n at time t. (2) $c_{t,n}$ denotes the time taken by vehicle n at time t to hop or take detours to pick up extra orders. This term discourages the agent from picking up additional orders without considering the delay in current passengers/goods orders. (3) $\sum_{u=1}^{U_n} \omega_u \cdot \delta_{t,n,u}$ denotes the sum of additional time vehicle n is incurring at time t to serve additional passengers or packages. The ω_u term is an "urgency" weight for the u'th order. (4) $\mathbb{P}_{t,n}$ rewards the agent for its pricing strategy (5) $\max(e_{t,n} - e_{t-1,n}, 0)$ addresses the objective of minimizing the the number of vehicles at t to improve vehicle utilization.

$$\begin{aligned} r_{t,n} = \beta_1(b_{t,n} + p_{t,n}) - \beta_2 c_{t,n} - \beta_3 \sum_{u=1}^{U_n} \omega_u \delta_{t,n,u} \\ - \beta_4 \mathbb{P}_{t,n} + beta_5 \max(e_{t,n} - e_{t-1,n}, 0) \end{aligned} \tag{1}$$

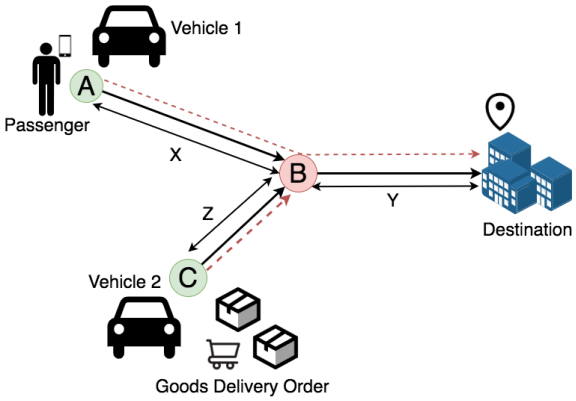

Figure 2: An illustration of multi-hop transportation scenario with hybrid delivery workload. Vehicle 1 and 2 are in zones A and C respectively. Zone B represents a hop-zone' where the packages part of the goods order packages transfer over to another vehicle

The DQN is trained using epsilon-greedy exploration. As mentioned previously, the Q-values for each specific action represent the potential reward of a vehicle mobilizing to a given region. These Q-values are key inputs to the Insertion Cost Route Planning component, and Pricing Policy as seen on Figure 1.

3.2 Multi-Hop Route Planning

In multi-hop routing, our algorithm specifically considers layovers for goods delivery requests in their journey to the destination.

In Figure 2, a good request is to go from zone C to the destination. The good can be transported by one vehicle from C to B, and another vehicle from B to the destination. Zone B will serve as a transit location referred to in this paper as a "hop-zone". This flexibility of changing vehicle is multi-hop transport of goods. Such multi-hop flexibility improves the packing of the goods, as was shown for the case of passengers in [12]. For passenger transfers, it was shown that the multi-hop transfers leads to 30% lower cost and 20% more efficient utilization of fleets, as compared to the ride-sharing algorithms. Even though such transfers may not be convenient for passengers, they are viable for goods transportation.

In transporting goods, we present a route planning framework that is analogous to hitch-hiking. A package is routed through layovers or "hop-zones" as we will refer to henceforth in this paper, in an iterative fashion such that at each iteration the system maximizes the distance towards the final destination. This way, at each leg of the package's journey, we allocate the locally optimal hopzone. We refer to this as Multi-hop routing.

Once the next destination of the request has been established for goods. The DARM framework presented in [14] is utilized for insertion-cost based route planning for both passenger and goods delivery requests. Using the cost function driven by the vehicle's DQN's expected discounted sum of rewards, perform this route planning.

4 Experiments

Given the objective of the transportation system, an optimal algorithm is effective in maximizing the number of passengers & goods delivered, maximizing vehicle profits, minimizing idle cruising time, and minimizing the waiting time of customers. Considering this objective, for each of the experiment models, we observe the performance of the fleet as a whole through key performance metrics. We observe the average "Profits" that fleet vehicles accumulated over the course of a day. Alongside, we observe the total "Cruising Time" of the fleet. This is the duration when a vehicle is neither occupied nor gaining profit but still expending fuel. We also observe the "Travel Time" which is the total distance during which fuel was being consumed. "Occupancy Rate" is also monitored as the percentage of time where vehicles are occupied out of their total working time.

The following baselines were compared to evaluate the effect of combined transportation and multi-hop routing relative to established model-free methods:

1. *Combined Load (DARM + DPRS + DMH)*: This is our proposed method where we extend DARM + DPRS with Dynamic Multi-hop (DMH) route planning for goods delivery. We consider a combined load where each vehicle transports a combination of goods and passengers.
2. *Combined Load (DARM + DPRS)*: This is the baseline with an insertion cost route planning and matching along with distributed pricing policy. We consider a combined load where each vehicle transports a combination of goods and passengers. Note that Dynamic Multi-hop routing has been omitted.
3. *Independent Load (DARM + DPRS + DMH)*: This is the baseline with an insertion cost and multi-hop route planning and matching along with distributed pricing policy. We consider an independent load where each vehicle transports either goods or passengers exclusively.
4. *Independent Load (DARM + DPRS)*: This is the baseline with an insertion cost route planning and matching along with distributed pricing policy as in [14]. We consider an independent load where each vehicle transports either goods or passengers exclusively. Note that Dynamic Multi-hop routing has been omitted.

We observe the plots shown in Figures 3, 4, 5, and 6 to evaluate model performance on the key fleet performance metrics. In comparing methods that consider multi-hop routing (Figure 3, Figure 5) to their direct routing counterparts (Figure 4, Figure 6), it can be observed that multi-hop routing significantly reduces cruising time and travel time. Consequently, we also see that the occupancy rate is also improved in this comparison. Our proposed method performs best in each of these efficiency metrics. Results also show that multi-hop routing improves the potential profit of vehicles. Comparing Figure 3 to Figure 4, a potential profit gain of approximately 20% can be seen. Through multi-hop routing, vehicles are able to deliver more requests to hop-zones which are conveniently located close to their current paths. This allows them to incur less cost and accumulate more profit in delivering requests. To evaluate the efficiency of combined load transportation, we compare Figure 3 and Figure 4 to Figure 5 and Figure 6 respectively. This comparison shows that profits and occupancy rates are also significantly improved with combined load transportation.

In summary, our proposed method combines two strategies that have demonstrated a more profitable and efficient fleet usage: multi-hop route planning, and combined load transportation.

5 Conclusions

In this paper, we proposed a framework to enable distributed decision-making for a fleet of vehicles with passenger and goods delivery tasks to maximize operational profitability, environmental sustainability and customer convenience. With each vehicle learning a DQN representation to map the environment state to the regions of high anticipated reward, we are able to utilize this accross various decision-making components that work in tandem to optimize the fleet objectives. The decision-making components we proposed include a DQN dispatch policy, a demand-aware pricing policy, a demand-aware route planning policy, and a dynamic multi-hop route planning policy for goods. In experimenting with adaptations of such policies in a realistic discrete-event simulator, we were able to demonstrate promising results for the inclusion of multi-hop considerations in route planning. Most notably, we observed a significant increase in vehicle utilization and profits. It is also worth noting that our approach can easily be scaled up to a large number of fleet vehicles and readily deployable locally within each vehicle to enable distributed inference.

In regards to future research directions, we propose developing systems that can incorporate deadline-based constraints to enable the transportation of highly urgent goods such as medicines or perishable items. Additionally, research towards developing systems for longer range deliveries would be essential.

References

[1] M. Iqbal, "Uber revenue and usage statistics (2020)." `https://www.businessofapps.com/data/uber-statistics/`, 2020. Accessed: 2020-10-05.

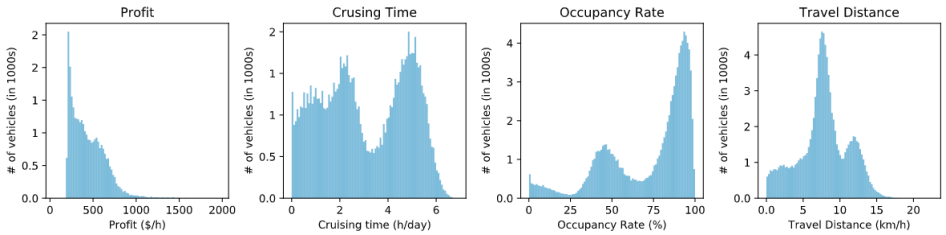

Figure 3: This figure plots the simulation metrics for the Combined Load (DARM + DPRS + DMH) proposed method.

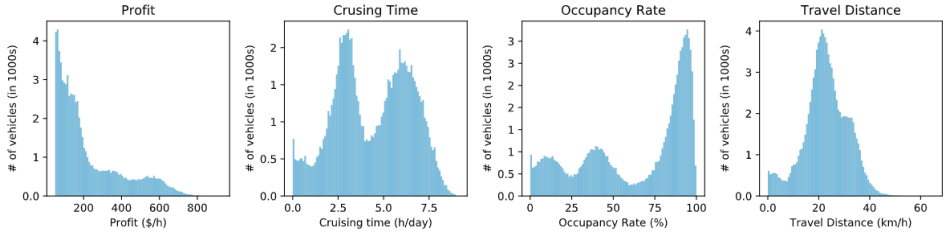

Figure 4: This figure plots the simulation metrics for the Combined Load (DARM + DPRS) baseline.

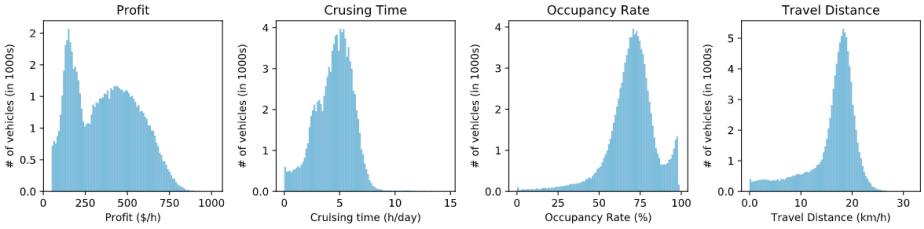

Figure 5: This figure plots the simulation metrics for the Independent Load (DARM + DPRS + DMH) baseline.

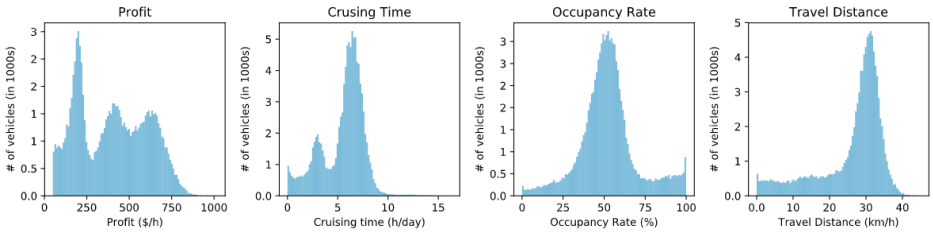

Figure 6: This figure plots the simulation metrics for the Independent Load (DARM + DPRS) baseline.

[2] D. Curry, "Instacart revenue and usage statistics (2020)." https://www.businessofapps.com/data/instacart-statistics/, 2020. Accessed: 2020-10-05.

[3] D. Curry, "Doordash revenue and usage statistics (2020)." https://www.businessofapps.com/data/doordash-statistics/, 2020. Accessed: 2020-10-05.

[4] D. Curry, "Grubhub revenue and usage statistics (2020)." https://www.businessofapps.com/data/grubhub-statistics/, 2020. Accessed: 2020-10-05.

[5] R. Hahn and R. Metcalfe, "The ridesharing revolution: Economic survey and synthesis," *More equal by design: economic design responses to inequality*, vol. 4, 2017.

[6] S. Shaheen, "Shared mobility: the potential of ridehailing and pooling," in *Three Revolutions*, pp. 55–76, Springer, 2018.

[7] D. Metz, "Developing policy for urban autonomous vehicles: Impact on congestion," *Urban Science*, vol. 2, no. 2, p. 33, 2018.

[8] X. Bei and S. Zhang, "Algorithms for trip-vehicle assignment in ride-sharing.," in *AAAI*, vol. 18, pp. 3–9, 2018.

[9] F. Jameel, Z. Chang, J. Huang, and T. Ristaniemi, "Internet of autonomous vehicles: Architecture, features, and socio-technological challenges," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 21–29, 2019.

[10] T. Oda and C. Joe-Wong, "Movi: A model-free approach to dynamic fleet management," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 2708–2716, IEEE, 2018.

[11] A. O. Al-Abbasi, A. Ghosh, and V. Aggarwal, "Deeppool: Distributed model-free algorithm for ride-sharing using deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 12, pp. 4714–4727, 2019.

[12] A. Singh, A. Alabbasi, and V. Aggarwal, "A distributed model-free algorithm for multi-hop ride-sharing using deep reinforcement learning," *arXiv preprint arXiv:1910.14002*, 2019.

[13] K. Manchella, A. K. Umrawal, and V. Aggarwal, "Flexpool: A distributed model-free deep reinforcement learning algorithm for joint passengers and goods transportation," in *Accepted to IEEE Transactions on Intelligent Transportation Systems Special Issue on Diversity in Transportation Systems for People and Goods*, Sep 2020.

[14] M. Haliem, G. Mani, V. Aggarwal, and B. Bhargava, "A distributed model-free ride-sharing approach for joint matching, pricing, and dispatching using deep reinforcement learning," *arXiv preprint arXiv:2010.01755*, 2020.

[15] M. Haliem, V. Aggarwal, and B. Bhargava, "Adapool: An adaptive model-free ride-sharing approach for dispatching using deep reinforcement learning," in *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pp. 304–305, 2020.

[16] M. Haliem, G. Mani, V. Aggarwal, and B. Bhargava, "A distributed model-free ride-sharing algorithm with pricing using deep reinforcement learning," in *The 4th ACM Computer Science in Cars Symposium (CSCS '20)*, ACM, 2020.