Optimizing Multi-class Fleet Compositions for Shared Mobility-as-a-Service

Alex Wallar[1], Wilko Schwarting[1], Javier Alonso-Mora[2], and Daniela Rus[1]

Abstract— **Mobility-as-a-Service (MaaS) systems are transforming the way society moves. The introduction and adoption of pooled ride-sharing has revolutionized urban transit with the potential of reducing vehicle congestion, improving accessibility and flexibility of a city's transportation infrastructure. Recently developed algorithms can compute routes for vehicles in real-time for a city-scale volume of requests, as well as optimize fleet sizes for MaaS systems that allow requests to share vehicles. Nonetheless, they are not capable of reasoning about the composition of a fleet and their varying capacity classes. In this paper, we present a method to not only optimize fleet sizes, but also their multi-class composition for MaaS systems that allow requests to share vehicles. We present an algorithm to determine how many vehicles of each class and capacity are needed, where they should be initialized, and how they should be routed to service all the travel demand for a given period of time. The algorithm maximizes utilization while reducing the total number of vehicles and incorporates constraints on wait-times and travel-delays. Finally, we evaluate the effectiveness of the algorithm for multi-class fleets with pooled ride-sharing using 426,908 historical taxi requests from Manhattan and 187,243 downtown Singapore. We show fleets comprised of vehicles with smaller capacities can reduce the total travel delay by 10% in Manhattan whereas larger capacity fleets in downtown Singapore contribute to a 9% reduction in the total waiting time.**

I. Introduction

Mobility-as-a-Service (MaaS) is important for the future of transportation by making transportation available anywhere at anytime. With autonomous vehicles on the horizon [1], transportation network companies are revolutionizing personal mobility and have the potential to provide faster and more efficient transportation using fleets of coordinated autonomous vehicles.

MaaS has to cope with the challenges of routing vehicles efficiently and determining the size and composition of fleets. State of the art algorithms are able to efficiently manage fleets of vehicles to service large volumes of requests as is needed in dense urban areas [2], as well as determining fleet sizes for fixed capacity vehicles [3]. This work not only determines routes and fleet sizes, but also the fleet composition, i.e. how many vehicles of which capacity the fleet should be composed of.

[1] The authors are at the Computer Science and Artificial Intelligence Laboratory of the Massachusetts Institute of Technology, 32 Vassar St, 02139 Cambridge MA, USA {wallar,wilkos,rus}@csail.mit.edu

[2] The author is at the Department of Cognitive Robotics, Delft University of Technology, Mekelweg 2, 2628 CD Delft, Netherlands J.AlonsoMora@tudelft.nl

This work was supported in part by the Netherlands Organisation for Scientific Research NWO Veni 15961, the Amsterdam Institute for Advanced Metropolitan Solutions, the MIT-Singapore Alliance on Research and Technology under the Future of Urban Mobility, an Amazon Research Award, NVIDIA, and the MIT Energy Initiative Shell-MIT Energy Fellowship.

Pooled ridesharing, where multiple passengers share the same vehicle, has allowed less vehicles to service more requests and therefore an increase in vehicle utilization and efficiency. This has been shown by services like UberPool and Lyft by extending our means of transportation within a city. In a similar manner, actively optimizing for the capacity and number of vehicles in a fleet is poised to increase efficiency.

Routing and assigning vehicles to pooled trips is computationally intractable for a large volume of requests and fleet sizes. While there has been recent developments in global on-demand vehicle dispatching algorithms that use constraints for the maximum allowable waiting time and travel delay to reduce the number of viable vehicle trips and make the pooled ridesharing problem more tractable [2], additionally assigning the number of vehicles and their respective capacities increases the problem complexity further.

When optimizing for fleet composition a trade-off between employing fewer cars and increased utility has to be made. When optimizing for fewer cars, one would intuitively increase the vehicle capacity, while if optimizing for vehicle utilization, more but smaller capacity cars would be the desired outcome. We introduce a tunable algorithm that allows a fleet operator to specify the desired operating point determining the tradeoff, driven by cost of vehicle versus cost of under-utilization. Our algorithm can also ensure that passengers do not need to wait too long to be picked up and that by sharing a ride, the passenger will not go too far out of their way. The approach can also guarantee that all requests are serviced and tells us how many vehicles of which capacity we need.

This paper contributes:

- Formulation of an optimization problem for ride-sharing fleet compositions, determining how many vehicles of which capacity are needed with objective of high utility and reduced number while incorporating bounds on maximum waiting time and maximum incurred delay.
- An algorithm for determining a computationally tractable approximate solution of the optimization problem.
- Case studies on downtown Singapore and Manhattan.

In the following we address the problem of determining how many vehicles of which capacity are needed and where they should be located for a MaaS fleet to service all the taxi demand at city-scale with a maximum waiting time and maximum incurred delay while allowing multiple requests to be serviced by the same vehicle. Our method runs offline and can inform a fleet operator of the distribution and size of their fleet needed to satisfy historical demand. We show that optimizing the number of vehicles, their distribution and their capacity, we can significantly improve the efficiency of a MaaS fleet. In the resulting case studies we also show that meaningful interpretations of fleet compositions can be made, such as whether investment in micro-mobility solutions with many 2 seat cars or fewer larger passenger cars with 6 seats additionally to traditional 4 seat vehicles are more suitable for increased utilization.

A. *Related Work*

While initial work on MaaS systems was bound to algorithms dealing with fleets of single occupancy vehicles [4]–[9], ride-sharing and the increased popularity of the sharing economy in general have shown to make fleets more efficient and affordable for more users.

Recently, algorithms that can compute and assign trips that allow multiple passengers to share the same vehicle for a large volume of requests have been developed. Santi et al. [10] showed that 80% of rides in Manhattan could pairwise shared without significantly impacting the quality of service to passengers. Tachet et al. [11] then extended this analysis of shared rides to multiple cities. Alonso-Mora et al. [2] developed a scalable algorithm to determine optimal assignments of transportation requests to vehicles in a fleet that allow more than two passengers to share the same vehicle. However, what makes these approaches so scalable are the hard constraints on the maximum waiting and delay a passenger can experience. This means that given an arbitrary vehicle distribution and fleet size, these approaches cannot guarantee that all the requests will be serviced.

There has been some research in determining what fleet sizes are needed to maintain a desired level of service. Boesch et al. [12] analyzed how different fleet sizes will perform to satisfy a given travel demand. Winter et al. [13] used a fixed start and end point model to determine how many vehicles would be needed. A recent breakthrough by Vazifeh et al. [14] can compute the minimum fleet size required to service historical demand data, but does not allow for pooled rides. Čáp and Alonso-Mora [15] used multi-objective analysis to estimate how many shared vehicles are needed to satisfy a set of requests, but the method was too computationally intensive and the authors only presented experiments using 1 minute worth of data.

The authors' recent previous work in [3] presented a scalable method to determine the fleet size given days worth of historical demand and allowed multiple passengers to share the same vehicle, but assumed that all vehicles had the same had the same capacity.

B. *Paper Structure*

The structure of this paper is as follows. Sec. II introduces many terms and defines many structures used throughout the paper. Sec. III presents the algorithm for determining the fleet size needed to service all the transportation requests. Sec. IV presents the experimental setup and evaluation on data from downtown Singapore and Manhattan. Lastly, Sec. V discusses the findings from the case studies and suggests a few future directions for research.

II. Problem Statement

Consider the operator of a pooled ridesharing system with vehicles of varying capacity. They are interested in optimizing the composition of their fleet without sacrificing their quality of service. Particularly, they want to know how many vehicles of each capacity are needed and where they should be located to satisfy all of the travel demand. The fleet operator provides historical travel request data to determine what fleet composition and vehicle distribution would have been needed to service these requests.

A. *Vehicles*

A vehicle's capacity is the maximum number of passengers it can carry at any time and can be one of $\mathcal{K}_{\text{avail}} = \{\kappa_1, \kappa_2, \ldots, \kappa_{|\mathcal{K}_{\text{avail}}|}\}$ where $1 \leq \kappa_i \leq \mathcal{K}_{\text{max}}$. Vehicles travel along a road-network, $G = (N, E)$, represented as a directed graph, and let's assume we have a function $\tau(n_i, n_j)$ for $n_i, n_j \in N$ that gives the shortest travel times between nodes in the graph. Vehicles are initialized on the road-network at locations called *vehicle deposits*. The vehicle deposits, $\mathcal{D} \subseteq N$, can be thought of as starting locations for vehicles before they have been assigned any requests.

B. *Requests*

For the time interval, $[0, t_{\text{max}}]$, the ridesharing system received a set of travel requests denoted as $\mathcal{R}$. A *travel request* is a tuple $r = (p_r, d_r, t_r^r)$, where $p_r \in N$ is the pickup location, $d_r \in N$ is the dropoff location, and t_r^r is the time the request was made. We consider a travel request to be successfully completed if the following quality of service constraints are satsified:

1) the waiting time, ω_r, given by the difference between the pickup time, t_r^p, and the request time, t_r^r is less than a specified maximum waiting time, Θ_{wait}
2) the travel delay for the request given by $\delta_r = t_r^d - t_r^*$ is less than a specified maximum travel delay, Θ_{delay}, where t_r^d is the time when the request is dropped off and $t_r^* = t_r^r + \tau(p_r, d_r)$ is the earliest possible time the destination could be reached.

C. *Trips*

Vehicles are assigned requests in batches we call *trips*. A trip, $T = \big((\ell_1, t_1), (\ell_2, t_2), \ldots, (\ell_{|T|}, t_{|T|})\big)$ is a sequence of pick up and drop off locations along with the time the pick up or drop off will occur. Let's define $\text{reqs}(T) \subseteq \mathcal{R}$ to be the requests serviced by T. Let's also define the *load* of T, $\mathcal{L}(T)$, as the maximum number of passengers in a vehicle at any one time while executing T. For convenience, let's use $\ell_{\text{start}}(T)$, $\ell_{\text{end}}(T)$, $t_{\text{start}}(T)$, and $t_{\text{end}}(T)$ to denote the locations and times of the first and last events of T. Let's also use $\ell_{\text{depo}}(T)$ to denote the vehicle deposit with the shortest travel time to $\ell_{\text{start}}(T)$

The cost of a trip is given by the sum of the delays experienced by the requests it services

$$\delta(T) = \sum_{r \in \text{reqs}(T)} \delta_r \tag{1}$$

TABLE I: Main symbols.

$\mathcal{K}_{\text{avail}} = \{\kappa_1, \ldots, \kappa_{\lvert\mathcal{K}_{\text{avail}}\rvert}\}$	Vehicle capacity, where $1 \leq \kappa_i \leq \mathcal{K}_{\max}$
$G = (N, E)$	Road network, directed graph
$\tau(n_i, n_j)$	Shortest travel time between $n_i, n_j \in N$
$\mathcal{D} \subseteq N$	Vehicle deposits, i.e. starting locations
$r = (p_r, d_r, t^r_r),\ r \in \mathcal{R}$	Travel request
$p_r \in N, d_r \in N, t^r_r$	pickup-, dropoff location, time of request
$\omega_r = t^p_r - t^r_r \leq \Theta_{\text{wait}}$	Wait time, diff. of pickup and request time
$\delta_r = t^d_r - t^*_r \leq \Theta_{\text{delay}}$	Travel delay, diff. of droppoff and earliest t.
$T = \big((\ell_1, t_1), \ldots\big)$	Trip, sequence of locations and times
$\text{reqs}(T) \subseteq \mathcal{R}$, $\mathcal{L}(T)$	Requests, and load of trip
$t_{\text{trans}}(T_i, T_j)$	Trip transition time
$t_{\text{idle}}(T_i, T_j) \leq \Theta_{\text{idle}}$	Tripe idle time
$\pi = \big(T_1, T_2, \ldots, T_{\lvert\pi\rvert}\big)$	Vehicle schedule
$\text{reqs}(\pi)$	Requests serviced by schedule
$\mathcal{T}^*$	Minimum trip cover of $\mathcal{R}$

For T to be valid, all the travel requests in the trip must be completed without violating the quality of service constraints. This implies that if an empty vehicle executed trip T, the vehicle would be empty upon completion. Additionally, for a vehicle to execute trip T, the load of T needs to be less than or equal to the capacity of the vehicle. We will use $\kappa(T) \in \mathcal{K}_{\text{avail}}$ to denote the capacity of the vehicle executing T.

D. Vehicle Schedules

We also consider which trips could be serviced by the same vehicle in sequence. We do so by finding ordered pairs of trips we call *trip transitions*. A trip transition is an ordered pair, (T_i, T_j), such that a vehicle is able to satisfy T_i then T_j without idling for longer than a given maximum vehicle idle time, Θ_{idle}, or violating the quality of service constraints. Let's define the transition time between trips as:

$$t_{\text{trans}}(T_i, T_j) = \tau(\ell_{\text{end}}(T_i), \ell_{\text{start}}(T_j)) \tag{2}$$

and the idle time between trips as:

$$t_{\text{idle}}(T_i, T_j) = t_{\text{start}}(T_j) - t_{\text{end}}(T_i) - t_{\text{trans}}(T_i, T_j) \tag{3}$$

We call the sequence of trips serviced by the same vehicle a *vehicle schedule* and it represents how a given vehicle will move to satisfy requests throughout the time interval. Specifically, a vehicle schedule, $\pi = \big(T_1, T_2, \ldots, T_{\lvert\pi\rvert}\big)$ is a trip sequence such that $\forall T_i \in \pi$, (T_i, T_{i+1}) is a valid trip transition.

Due to the similarity in structure of trip and schedules, we will reuse the convenience functions (i.e. $\text{reqs}(\pi)$ are the requests serviced by schedule π).

E. Fleet Optimization Criteria

We consider two main objectives when optimizing the composition of a pooled ridesharing fleet: 1) the total number of vehicles needed to satisfy the demand, 2) the utilization of vehicle capacity. The values of these objectives is fully defined by the set of schedules we compute. For a set of schedules, $\Pi = \{\pi_1, \pi_2, \ldots\}$, we can combine these objectives into a single cost function:

$$\mathcal{C}(\Pi) = K_{\text{size}} \cdot \lvert\Pi\rvert + K_{\text{util}} \cdot \sum_{\pi\in\Pi} \sum_{T\in\pi} \kappa(\pi) - \mathcal{L}(T) \tag{4}$$

where K_{size} and K_{util} are scalarization weights. K_{size} and K_{util} define a point along the pareto optimality forntier defining the tradeoff between smaller fleets and higher utilization of capacity of vehicles. They are defined by a fleet operators specifications of the cost tradeoff of fleets with many, smaller capacity vehicles and fewer, but larger capacity vehicles.

Using this cost function and the other structures provided in this section, we can formulate our fleet composition optimization problem as follows:

Problem 1 (Heterogeneous Fleet Composition)**.** Given a set of requests, $\mathcal{R}$, a set of available capacities classes, $\mathcal{K}_{\text{avail}}$, and vehicle deposit locations, $\mathcal{D}$, find the set of schedules Π that solves

$$\underset{\Pi}{\text{argmin}} \quad \mathcal{C}(\Pi) \quad \text{subject to}$$
$$\omega_r \leq \Theta_{\text{wait}} \wedge \delta_r \leq \Theta_{\text{delay}}, \quad \forall r \in \mathcal{R} \tag{5}$$
$$t_{\text{idle}}(T_i, T_{i+1}) \leq \Theta_{\text{idle}}, \quad \forall T_i \in \pi, \forall \pi \in \Pi \tag{6}$$
$$\mathcal{L}(\pi) \leq \kappa(\pi), \quad \forall \pi \in \Pi \tag{7}$$
$$\ell_{\text{start}}(\pi) \in \mathcal{D}, \quad \forall \pi \in \Pi \tag{8}$$
$$\cap_{\pi\in\Pi}\text{reqs}(\pi) = \emptyset \tag{9}$$
$$\cup_{\pi\in\Pi}\text{reqs}(\pi) = \mathcal{R} \tag{10}$$

A solution to the aforementioned problem is the set of vehicle schedules that would need to be executed to minimize the cost function, service all of the travel demand, and satisfy the quality of service constraints. The schedules directly define the number, the capacity classes, and the starting locations of all the vehicles.

III. Optimizing Multi-class Fleet Compositions

We optimize the composition and fleet size needed to service the given transportation demand in four steps. Firstly, using integer linear programming, we determine where and how many vehicle deposits are needed to dispatch vehicles in the fleet, cf. Sec. III-A. These deposits can be thought of as potential starting locations for the vehicles. Next, we generate a large candidate set of possible vehicle trips that service the transportation demand, see Sec. III-B. This set contains trips of different lengths and may contain multiple trips that service the same request. These candidate trips can be thought of as all the different ways the transportation requests can be serviced. Next, we select the minimum number of trips from the candidate set such that every request is serviced exactly once, Sec. III-C. We can think of this as minimizing the number of total trips needed to service all of the transportation demand. Finally, we determine how many and which types of vehicles are needed to complete these trips, cf. Sec. III-D, by considering which trips can be completed in sequence without violating the quality of service constraints or letting a vehicle idle for too long between trips.

A. Selecting Vehicle Deposits

Due to the maximum waiting time and maximum delay constraints for travel requests, it is not possible to guarantee a prescribed service rate for an arbitrary set of vehicle deposits. For instance, if the travel time from the closest vehicle deposit to a request's pick up location is larger than the maximum waiting time, that request may be impossible to service. Therefore, in order to provide a guaranteed service rate, we must intelligently select the locations for the vehicle deposits.

Using the road-network, $G = (N, A)$, we can select a set $D \subseteq N$ as vehicle deposit locations such that $\forall n \in N$, $\exists d \in D$ with $\tau(d, n) \leq \Theta_{\text{depos}}$ where $0 \leq \Theta_{\text{depos}} \leq \Theta_{\text{wait}}$. To reduce the computational overhead for computing trips, we select the minimum number of vehicle deposits needed. We can do this by solving an integer linear program. First, let's define a reachability matrix as follows,

$$H_{ij} = \begin{cases} 1, & \text{if } \tau(n_i, n_j) \leq \Theta_{\text{depos}} \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

This matrix describes which nodes are reachable from a given node within a specified amount of time. Let's also define a set of binary variables x where $x_i = 1$ if n_i is used as a deposit location and 0 otherwise. We can now solve an ILP to determine the minimum number of nodes to use as vehicle deposits such that the reachability constraint is satisfied:

$$\underset{x}{\text{argmin}} \quad \sum_{i=1}^{|N|} x_i, \quad \text{subject to} \tag{12}$$

$$\sum_{i=1}^{|N|} x_i \cdot H_{ij} \geq 1, \quad \forall j \in [1, |N|] \tag{13}$$

Eq. (13) guarantees that every node in N is reachable within Θ_{depos} travel time from at least one vehicle deposit. Now we can define the set of vehicle deposits as $\mathcal{D} = \{n_i : 1 \leq i \leq |N| \wedge x_i = 1\}$

B. Generating Candidate trips

The first part of our problem is to compute the set of candidate trips for a given set of requests. We first define a pairwise *shareability graph*, $\mathcal{Q} = (\mathcal{R} \cup \mathcal{D}, \mathcal{E})$, where $\mathcal{R}$ is our set of requests, $\mathcal{D}$ is our set of vehicle deposits, and $\mathcal{E}$ is our set of edges. There exists an edge between two requests, $r_1, r_2 \in \mathcal{R}$, if it is possible for a vehicle starting at the pick up location of one request to complete pick up and drop off of both requests while satisfying our maximum travel delay and maximum waiting time constraints and if $|t^r_{r_1} - t^r_{r_2}| \leq t_{\text{reqs}}$, that is the time between when the requests were made is less than some parameter, t_{reqs}. Since we are computing trips for an arbitrarily large set of requests, $\mathcal{R}$, this constraint limits the size of our shareability graph to make the computing trips more tractable.

Our shareability graph also has edges between vehicle deposits and requests. There exists an edge between $d \in \mathcal{D}$ and $r \in \mathcal{R}$ if a vehicle starting at the location of deposit d is able to reach r's pick up location under the maximum waiting time, Θ_{wait}. That is, $\tau(l_d, p_r) \leq \Theta_{\text{wait}}$. Let us also define a function $\zeta^*(d, R)$, where $d \in \mathcal{D}$ and $R \subseteq \mathcal{R}$ to return the minimum cost trip given by Eq. (1) or $\emptyset$ if no valid trip exists. The function $\zeta^*(d, R)$ can be determined using a variety of techniques such as solving an ILP, using constraint programming techniques, or tree search. For our implementation, we performed tree search over the possible trips and kept track of the minimum cost trip to return for a given set of requests. A valid trip must also ensure that the number of passengers in a vehicle from deposit d does not exceed the vehicle capacity, κ_d.

Algorithm 1 Generating Candidate trips

```
1:  $\mathcal{T} \leftarrow \{\}$
2:  for $d \in \mathcal{D}$ do
3:      $\mathcal{T}_1 \leftarrow \{\}$
4:      for $(d, r) \in \mathcal{E}$ do
5:          $T \leftarrow \zeta^*(d, \{r\})$
6:          $\mathcal{T}_1 \leftarrow \mathcal{T}_1 \cup \{T\}$
7:      end for
8:      for $i = 2$ to $\Gamma$ do
9:          $\mathcal{T}_i \leftarrow \{\}$
10:         for $T \in \mathcal{T}_{i-1}$ do
11:             $U \leftarrow \{r_1 : (r_1, r_2) \in \mathcal{E} \wedge r_2 \notin \text{reqs}(T) \wedge (d, r_1) \in \mathcal{E}\}$
12:             for $u \in U$ do
13:                 $T' \leftarrow \zeta^*(d, \text{reqs}(T) \cup \{u\})$
14:                 $\mathcal{T}_i \leftarrow \mathcal{T}_i \cup \{T'\}$
15:             end for
16:         end for
17:     end for
18:     $\mathcal{T} \leftarrow \mathcal{T} \cup \bigcup_{i=1}^{\Gamma} \mathcal{T}_i$
19: end for
```

We compute the trips for a given set of requests iteratively, starting with trips with one request, and adding requests to these trips to build larger ones. An overview of how trips are computed is shown in Algo. 1

Using the shareability graph, $\mathcal{Q}$, we compute all trips containing exactly one request for a given vehicle deposit, $d \in \mathcal{D}$, which we call $\mathcal{T}_1$. For all $T \in \mathcal{T}_1$, we expand our search tree with neighbors of requests in reqs(T) from $\mathcal{Q}$ that also have an edge to d. We then check if a trip exists and add it to our set of trips with two requests, $\mathcal{T}_2$. We repeat this process up until we have computed all trips routing up to Γ requests. We also repeat this process for each $d \in \mathcal{D}$. For more information on how these trips are computed, please refer to [2].

C. Trip Selection

Given the set of candidate trips, $\mathcal{T}$, we must determine which trips should be selected to feasibly satisfy all of the requests in $\mathcal{R}$. To compute these trips we propose the following problem:

Problem 2 (Trip Selection)**.** Given a set of candidate trips, $\mathcal{T}$, that service requests in $\mathcal{R}$, solve

$$\underset{\mathcal{T}^* \subseteq \mathcal{T}}{\text{argmin}} \quad |\mathcal{T}^*| \quad \text{subject to}$$

$$\bigcap_{T \in \mathcal{T}^*} \text{reqs}(T) = \emptyset$$

$$\bigcup_{T \in \mathcal{T}^*} \text{reqs}(T) = \mathcal{R}$$

A solution to the aforementioned problem selects the minimum number of trips from $\mathcal{T}^*$ that would satisfy all of the requests. We can formulate the trip selection problem as an ILP and solve it to find which trips will be used. Let's define a set of binary variables $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_{|\mathcal{T}|}\}$ where $\sigma_i = 1$ if the ith trip in $\mathcal{T}$ is selected and 0 otherwise. Let's also define a function $\mathcal{I} : \mathcal{R} \rightarrow 2^{\mathbb{N}}$ to be the set of indices of trips in $\mathcal{T}$ that service a given request, i.e. $\mathcal{I}(r)$ are the indices of trips in $\mathcal{T}$ that service request r. The whole ILP is formulated as follows:

$$\underset{\sigma}{\text{argmin}} \quad \sum_{i=1}^{|\mathcal{T}|} \sigma_i \quad \text{subject to} \tag{14}$$

$$\sum_{i \in \mathcal{I}(r)} \sigma_i = 1, \quad \forall r \in \mathcal{R} \tag{15}$$

The constraint in Eq. 15 ensures that each request is serviced exactly once and Eq. 14 computes the number of trips selected over the variables, σ.

We extract the minimum trip cover, $\mathcal{T}^*$, from the solution of the aforementioned ILP as follows:

$$\mathcal{T}^* = \{T_i : T_i \in \mathcal{T} \wedge \sigma_i = 1\} \tag{16}$$

We will call the $\mathcal{T}^*$, the *minimum trip cover* of $\mathcal{R}$, because it is the smallest subset from $\mathcal{T}$ to cover all of the requests in $\mathcal{R}$.

D. Determining the Fleet Size and Composition

The last step is to determine how many vehicles are needed and what capacity classes they should be to complete the trips selected from the previous step, $\mathcal{T}^*$. We do so by determining which trips can be executed in sequence by the same vehicle without violating the quality of service constraints or letting the vehicle idle for too long between trips, then we minimize the total vehicles needed. The composition and size of the fleet can be determined by solving the following problem:

Problem 3 (Fleet Size and Composition)**.** Given the minimum trip cover, $\mathcal{T}^*$, find the trip transition function, $\theta : \mathcal{T}^* \rightarrow \mathcal{T}^* \cup \{\emptyset\}$ and capacity class for each trip, $\kappa : \mathcal{T}^* \rightarrow \mathcal{K}_{\text{avail}}$ that solves

$$\underset{\kappa,\theta}{\text{argmin}}\ K_{\text{size}} \cdot |\mathcal{T}^* \setminus \bigcup_{T \in \mathcal{T}^*} \theta(T)| + K_{\text{util}} \cdot \sum_{T \in \mathcal{T}^*} \kappa(T) - \mathcal{L}(T)$$

subject to

$$t_{\text{end}}(T) + t_{\text{trans}}(T, \theta(T)) \le t_{\text{start}}(\theta(T)), \quad \forall T \in \mathcal{T}^*, \theta(T) \neq \emptyset \tag{17}$$

$$t_{\text{idle}}(T, \theta(T)) \le \Theta_{\text{idle}}, \quad \forall T \in \mathcal{T}^*, \theta(T) \neq \emptyset \tag{18}$$

$$\kappa(T) = \kappa(\theta(T)), \quad \forall T \in \mathcal{T}^*, \theta(T) \neq \emptyset \tag{19}$$

$$\kappa(T) \ge \mathcal{L}(T), \quad \forall T \in \mathcal{T}^* \tag{20}$$

$$\bigcap_{T \in \mathcal{T}^*} \theta(T) = \emptyset \tag{21}$$

Constraints (17) and (18) ensure that the a vehicle is able service trips T and $\theta(T)$ in sequence without violating the quality of service guarantees (i.e. all trip transitions are valid). Constraints (19) and (20) ensure that all trips in a schedule are assigned the same capacity and that the assigned capacity is at least the load of the trip respectively. Finally, the last constraint in Eq. (21) ensures a trip has only one outgoing and incoming transition.

We can formulate this problem as an ILP. First let's define a transition matrix. For all $T_i, T_j \in \mathcal{T}^*$, we define a binary transition as

$$\eta_{ij} = \begin{cases} 1, & \text{if } t_{\text{idle}}(T_i, T_j) \le \Theta_{\text{idle}} \\ & \text{and } t_{\text{end}}(T_i) + t_{\text{trans}}(T_i, T_j) \le t_{\text{start}}(T_j) \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

This transition matrix constrains the possible sequences of trips. The first clause of Eq. (22) ensures that a vehicle would not idle for longer than Θ_{idle} between trips. The second clause guarantees that a vehicle will be be able to reach the first pick up of the trip in time.

We define sets of variables to optimize over as follows:

- $\epsilon = \{\epsilon_{ij} : \forall i \in [1, |\mathcal{T}^*|], \forall j \in [1, |\mathcal{T}^*|]\}$, where $\epsilon_{ij} = 1$ if a vehicle should transition from the ith to jth trips $\mathcal{T}^*$ and 0 otherwise
- $\chi = \{\chi_i : \forall i \in [1, |\mathcal{T}^*|]\}$, where $\chi_i = 1$ if the ith trip in $\mathcal{T}^*$ is the first trip to be completed by a vehicle and 0 otherwise
- $\xi = \{\xi_i \in \mathcal{K}_{\text{avail}} : \forall i \in [1, |\mathcal{T}^*|]\}$, where ξ_i is the assigned capacity of the ith trip in $\mathcal{T}^*$

With these variables and our transition matrix, η, we can formulate the full ILP as:

$$\underset{\epsilon,\chi,\xi}{\text{argmin}} \quad \sum_{T_i \in \mathcal{T}^*} K_{\text{size}} \cdot \chi_i + K_{\text{util}} \cdot (\xi_i - \mathcal{L}(T_i)) \tag{23}$$

$$\text{subject to} \tag{24}$$

$$\chi_j + \sum_{i=1}^{|\mathcal{T}^*|} \epsilon_{ij} = 1, \quad \forall j \in [0, |\mathcal{T}^*|] \tag{25}$$

$$\sum_{j=1}^{|\mathcal{T}^*|} \epsilon_{ij} \le 1, \quad \forall i \in [1, |\mathcal{T}^*|] \tag{26}$$

$$\xi_i - \xi_j + \epsilon_{ij} \le 1, \quad \forall i, j \in [1, |\mathcal{T}^*|] \tag{27}$$

$$\xi_j - \xi_i + \epsilon_{ij} \le 1, \quad \forall i, j \in [1, |\mathcal{T}^*|] \tag{28}$$

$$\eta_{ij} - \epsilon_{ij} \ge 0, \quad \forall i, j \in [1, |\mathcal{T}^*|] \tag{29}$$

$$\xi_i \ge \mathcal{L}(T_i), \quad \forall T_i \in \mathcal{T}^* \tag{30}$$

A solution to the aforementioned MIP selects the set of transitions, initial trips, and capacities minimizes the cost function from Problem 3 such that all requests are satisfied. The objective function in Eq. (23) represents the weighted sum of the fleet size and the unused capacity of the vehicles. The constraints in Eq. (25) and (26) ensure that trips can have at most one incoming transition and one out going transition (i.e. that a trip can only be serviced by exactly one vehicle). Eq. (27) and (28) guarantee that all trips in sequence are serviced by a vehicle with the same capacity. Eq. (29) allows only valid transitions between trips to be considered. The last constraint in Eq. (30) ensures that the vehicle capacity class assigned to a given trip can accommodate the trip's load.

From the optimized variables, we construct the set of schedules, Π, that need to be executed by the vehicles in fleet. This is shown in Algo. 2. Finally, we determine the initial

Algorithm 2 Constructing Schedules

```
1: $\Pi \leftarrow \emptyset$
2: for all $T_i \in \mathcal{T}^*$ do
3:     $\pi \leftarrow \emptyset$
4:     if $\chi_i = 1$ then
5:         $\pi \leftarrow \pi \cup \{T_i\}$
6:         $k \leftarrow i$
7:         while $\exists j$, $\epsilon_{kj} = 1$ do
8:             $\pi \leftarrow \pi \cup \{\mathcal{T}_j^*\}$
9:             $k \leftarrow j$
10:        end while
11:    end if
12:    $\Pi \leftarrow \Pi \cup \{\pi\}$
13: end for
```

vehicle distribution using the starting locations of the initial trips in each vehicle's schedule. This is shown in Eq. (31)

$$\mathcal{L} = \{\ell_{\text{start}}(T_i) : T_i \in \mathcal{T}^* \wedge \chi_i = 1\} \tag{31}$$

IV. Evaluation

We evaluate the proposed algorithm using historical taxi request data from Manhattan [16] and Singapore[1]. We collect various metrics to show how a fleet operator can tune the composition of their fleet to be most effective.

A. Experimental Setup

We evaluated the algorithm in two operating areas, Manhattan and downtown Singapore. These regions are indicated by the green polygons in Fig. 1. For each operating area, we used one day of historical taxi data. For Manhattan we used May 1st, 2013; and in Singapore, we used June 4th, 2012. The historical data contains the origin, destination, pick up time, and drop off time for a set of taxi requests in Manhattan and Singapore. From this data, we use the reported pick up time as the request time since the request time was not provided. We extracted the road networks from OpenStreetMap [17] and queried travel times from Google Maps. For each operating area, we computed the shortest paths and travel times between every pair of nodes in the road network offline. We evaluated the algorithm independently each 30 minute time intervals for each day (i.e. 12am-12:30am, 12:30am-1am, etc).

We assess the performance of the proposed approach using available capacities of $\mathcal{K}_{\text{avail}} = \{2, 4\}$ and $\mathcal{K}_{\text{avail}} = \{4, 6\}$. For each capacity set, we use a fixed maximum waiting time, $\Theta_{\text{wait}} = 3$ minutes and a maximum travel delay, $\Theta_{\text{delay}} = 6$ minutes. We specified the maximum vehicle idle time as $\Theta_{\text{idle}} = 30$ seconds. Vehicle deposits were selected using a maximum travel time of $\Theta_{\text{depos}} = 1$ minute.

We compare the overall fleet sizes produced to the actual size of the taxi fleet used to service the requests. We also compare characteristics of fleet compositions produced given the available capacity classes.

[1]Data provided by Comfort Taxi through the Singapore-MIT Alliance for Research and Technology

B. Results

We collect several metrics to assess the performance of the proposed algorithm including the resultant fleet compositions for a given set of available capacity classes, the total fleet size, the vehicle capacity utilization, the capacity efficiency, the total waiting time, and the total incurred travel delay. The vehicle capacity utilization is load of each trip divided by the available capacity of the vehicle. We define the capacity efficiency to be the total number of requests for a time interval divided by the total capacity of the fleet. We can think of this metric as the average number of requests serviced per seat. The fleet sizes and compositions for Manhattan and Singapore are shown in Fig. 2 and 3. A comparison of the capacity utilization and efficiency between fleets produced for different sets of available capacities for Manhattan and Singapore is shown in Fig. 4. These metrics can help a fleet operator decide what types of vehicles should make up their fleet.

For both Manhattan and Singapore, we see a large reduction in fleet sizes needed to service all of the demand compared to the actual fleet size. For Manhattan, we see that for a $\{2, 4\}$-fleet, there are always more vehicles of capacity 4 than of 2, but in Singapore, we see the opposite trend. This is because Manhattan has a denser demand profile than Singapore making sharing more prevalent. For both Manhattan and Singapore, we see that for $\{4, 6\}$-fleets, there are always more capacity 4 vehicles than capacity 6 vehicles.

During rush hour (between 5am to 8am), there are large spikes in demand that the fleets need to accommodate. In Manhattan, for both fleet types, we large spikes in the number of capacity 4 vehicles. For $\{2, 4\}$-fleets, we see the number of capacity 4 vehicles increase much more quickly during rush hour than capacity 2 vehicles. This is likely due to the large density in demand in Manhattan during rush hour. For $\{4, 6\}$-fleets, we observe an interesting phenomenon; the number of capacity 6 vehicles plateaus during rush hour even as the number of capacity 4 vehicles continues to increase. This is due to the saturation of capacity 6 vehicles given the quality of service constraints and capacity utilization cost function for determining the fleet composition. One can only add so many capacity 6 vehicles to the fleet before the marginal benefit due to more sharing is diminished due to longer delays and waiting times along with less capacity utilization than capacity 4 vehicles.

We can see in Fig. 4 that better vehicle capacity utilization and capacity efficiency is obtained when using the $\{2, 4\}$-fleet rather than the $\{4, 6\}$-fleet. Particularly in Manhattan during lower demand times in the early morning, we see steeper drops in the vehicle capacity utilization for $\{4, 6\}$-fleets than for $\{2, 4\}$-fleets. In Manhattan, we observe a higher capacity efficiency than in Singapore. For fleet types, Manhattan had more requests per seat than Singapore. This is also due to the differences in demand density between the two areas. In Manhattan we see that for $\{2, 4\}$-fleets each individual seat services at least one request on average per 30 minute interval. Given that the vehicle capacity utilization is less than 100%, this means that vehicles are quickly picking up and dropping off passengers without reaching full

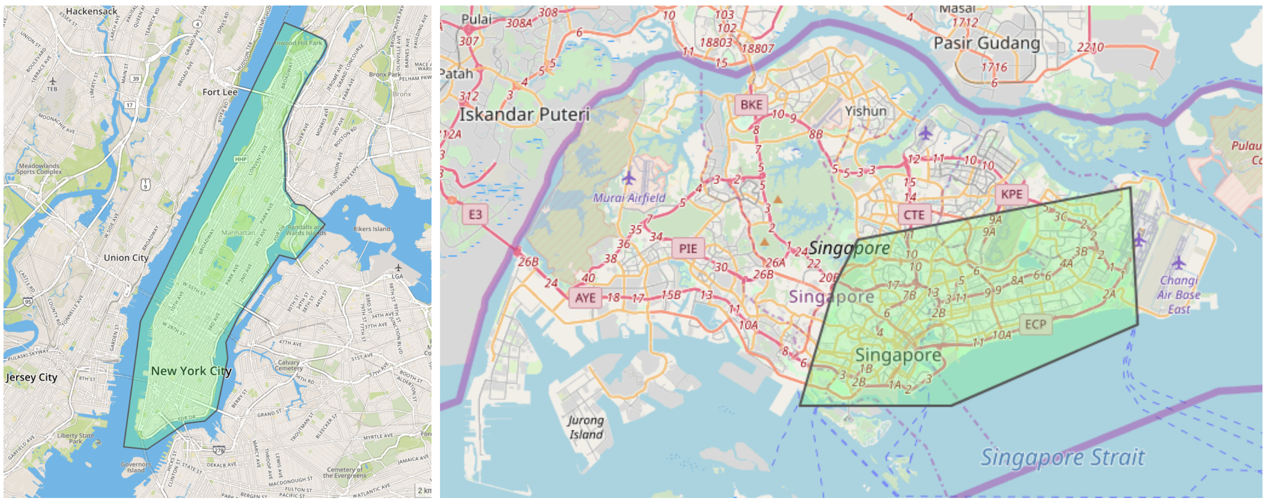
Fig. 1: Operating areas used for experimentation in Manhattan and Singapore

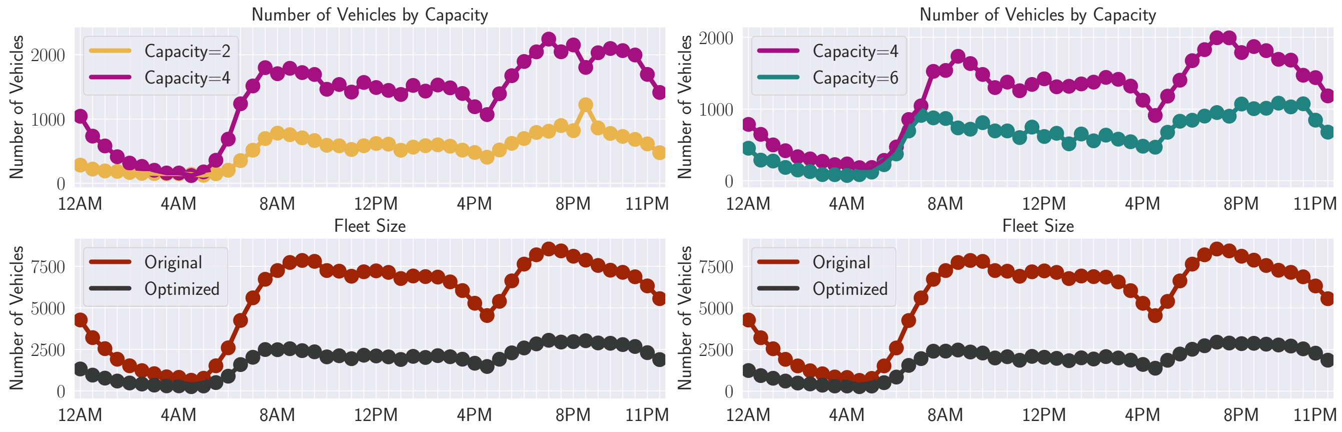

Fig. 2: Plots showing the compositions and sizes for fleets composed of vehicles with available capacities of $\{2, 4\}$ and $\{4, 6\}$ in Manhattan. Plots in the left column show results for $\{2, 4\}$-fleets and in the right column $\{4, 6\}$-fleets

capacity in order for the delay and waiting times to be under the specified maximum tolerances.

Since the overall fleet sizes for each fleet type are very similar, the more efficient utilization of the $\{2, 4\}$-fleet indicates that an introduction of lower capacity, micro-mobility platforms may be a more efficient addition to the mobility markets of Singapore and Manhattan than adding SUVs.

The total incurred travel delay and waiting time also differ between $\{2, 4\}$-fleets and $\{4, 6\}$-fleets. In Table. II we record the total delay and waiting time for Manhattan and Singapore for the different fleet types. In Manhattan we observe that the total delay for the $\{4, 6\}$-fleet is over 10% more than that of a $\{2, 4\}$-fleet while the total waiting time stays roughly the same. However in downtown Singapore we see the total delay and total waiting time drop almost 5% and 9% respectively. This result along with the the capacity utilization in Fig. 4 indicates that the additional capacity in Singapore is used to pick up passengers more quickly resulting to a drop in the waiting time. Due to the demand density in Manhattan, the additional capacity is quickly filled adding to total delays.

		Total Delay [hr]	Total Wait [hr]
Manhattan	$\{2, 4\}$-fleet	13477	5276
	$\{4, 6\}$-fleet	14855	5256
	% Difference	10.2%	-0.4%
Singapore	$\{2, 4\}$-fleet	5749	1720
	$\{4, 6\}$-fleet	5474	1569
	% Difference	-4.8%	-8.8%

TABLE II: Table showing differences in total travel delay and waiting time experienced by passengers for Manhattan and downtown Singapore for fleets composed of different available capacities

V. Conclusion

We have presented an algorithm to determine how many vehicles of each class and capacity are needed, where they should be initialized, and how they should be routed to service all the travel demand for a given period of time. The algorithm maximizes utilization while reducing the total number of vehicles and incorporates constraints on wait-times and travel-delays. We have shown that optimizing the number of vehicles, their distribution and their capacity, we can significantly improve the efficiency of MaaS fleets. In the future we would like run case studies in other cities and explore ways of applying the algorithm online.

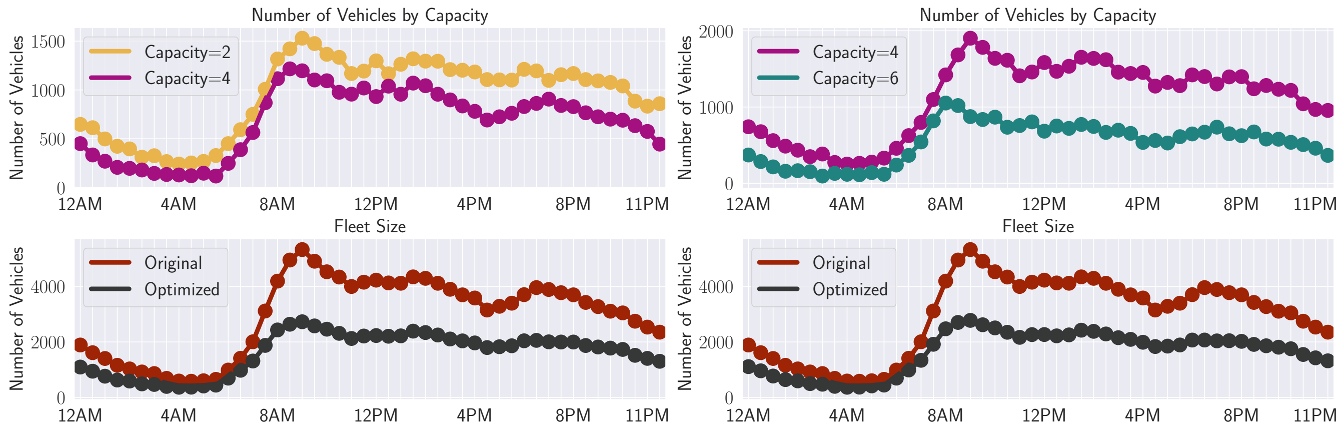

Fig. 3: Plots showing the compositions and sizes for fleets composed of vehicles with available capacities of $\{2, 4\}$ and $\{4, 6\}$ in downtown Singapore. Plots in the left column show results for $\{2, 4\}$-fleets and in the right column $\{4, 6\}$-fleets

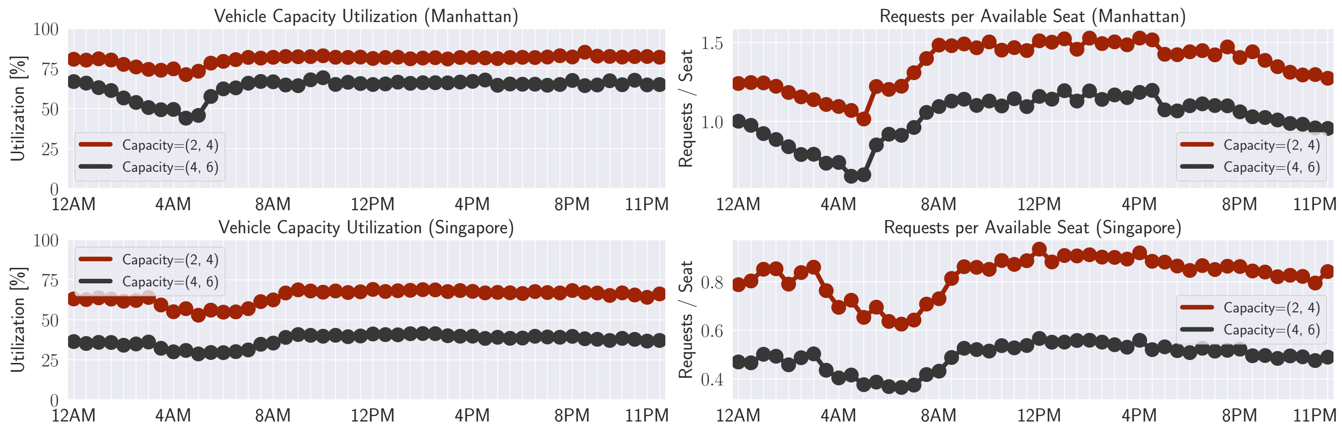

Fig. 4: Plots comparing the capacity utilization and efficiency of $\{2, 4\}$-fleets and $\{4, 6\}$-fleets in Manhattan (bottom row) and downtown Singapore (bottom row)

References

[1] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, 2018.

[2] J. Alonso-Mora, S. Samaranayake, A. Wallar, E. Frazzoli, and D. Rus, "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment," *Proceedings of the National Academy of Sciences*, vol. 114, no. 3, pp. 462–467, 2017.

[3] A. Wallar, J. Alonso-Mora, and D. Rus, "Optimizing vehicle distributions and fleet sizes for shared mobility-on-demand," in *International Conference on Robotics and Automation (ICRA)*, 2019.

[4] M. Pavone, S. L. Smith, E. Frazzoli, and D. Rus, "Robotic load balancing for mobility-on-demand systems," *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 839–854, 2012.

[5] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, and M. Pavone, "Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems: A case study in singapore," in *Road vehicle automation*, pp. 229–245, Springer, 2014.

[6] K. Treleaven, M. Pavone, and E. Frazzoli, "Asymptotically optimal algorithms for one-to-one pickup and delivery problems with applications to transportation systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 9, pp. 2261–2276, 2013.

[7] A. Prorok and V. Kumar, "Privacy-preserving vehicle assignment for mobility-on-demand systems," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pp. 1869–1876, IEEE, 2017.

[8] G. Clare and A. G. Richards, "Optimization of taxiway routing and runway scheduling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1000–1013, 2011.

[9] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: a queueing-theoretical perspective," *Proceedings of Robotics: Science and Systems Conference*, July 2014.

[10] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks.," *Proceedings of the National Academy of Sciences*, vol. 111, no. 37, pp. 13290–4, 2014.

[11] R. Tachet, O. Sagarra, P. Santi, G. Resta, M. Szell, S. Strogatz, and C. Ratti, "Scaling law of urban ride sharing," *Scientific reports*, vol. 7, p. 42868, 2017.

[12] P. M. Boesch, F. Ciari, and K. W. Axhausen, "Autonomous vehicle fleet sizes required to serve different levels of demand," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2542, pp. 111–119, 2016.

[13] K. Winter, O. Cats, G. H. d. A. Correia, and B. Van Arem, "Designing an automated demand-responsive transport system: Fleet size and performance analysis for a campus–train station service," *Transportation Research Record: Journal of the Transportation Research Board*, no. 2542, pp. 75–83, 2016.

[14] M. M. Vazifeh, P. Santi, G. Resta, S. H. Strogatz, and C. Ratti, "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, no. 7706, pp. 534–538, 2018.

[15] M. Čáp and J. Alonso-Mora, "Multi-objective analysis of ridesharing in automated mobility-on-demand," *Proceedings of Robotics: Science and Systems Conference*, 2018.

[16] B. Donovan and D. B. Work, "New York City Taxi Trip Data (2010-2013)." http://dx.doi.org/10.13012/J8PN93H8, 2014.

[17] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ." `https://www.openstreetmap.org`, 2017.