Balancing Efficiency and Fairness in On-Demand Ridesourcing

Nixie Lesmana^1 , **Xuan Zhang**^{2} , **Xiaohui Bei**^{1}
^{1}Nanyang Technological University
^{2}Shanghai Jiaotong University
nlesmana@ntu.edu.sg, zxjszy@126.com, xhbei@ntu.edu.sg

Abstract

We investigate the problem of assigning trip requests to available vehicles in on-demand ridesourcing. Much of the literature has focused on maximizing the total value of served requests, achieving efficiency on the passengers' side. However, such solutions may result in some drivers being assigned to insufficient or undesired trips, therefore losing fairness from the drivers' perspective.

In this paper, we focus on both the system efficiency and the fairness among drivers and quantitatively analyze the tradeoffs between these two objectives. In particular, we give an explicit answer to the question of whether there always exists an assignment that achieves any target efficiency and fairness. We also propose a simple reassignment algorithm that can achieve any selected tradeoff. Finally, we demonstrate the effectiveness of the algorithms through extensive experiments on a real-world dataset of an online ridesourcing platform.

1 Introduction

Ridesourcing refers to a mode of transportation that connects private car drivers with passengers via mobile devices and applications. Recent advances in technology provide the opportunity for ridesourcing platforms to dynamically match drivers and passengers in real time. This new generation of ridesourcing significantly increases the efficiency of urban transportation systems, consequently reducing congestion and pollution [Li *et al.*, 2018]. One key function of these platforms is to automatically assign potential passengers to active drivers. The development of an efficient real-time demand assignment algorithm is central to the concept and to the success of a ridesourcing enterprise.

Research into real-time ridesourcing has often focused on developing algorithms for optimal assignment of sets of requests to drivers [Alonso-Mora *et al.*, 2017; Xu *et al.*, 2018; Lin *et al.*, 2018]. In these studies, the common objective is to minimize the total waiting time for passengers and maximize the service rate, achieving efficiency on the passengers' side. Admittedly, customer satisfaction should be the main goal in any service industry. However, in the ridesourcing domain, the role of drivers is as important as that of passengers in terms of sustaining the business. Drivers have preferences that might not align with those of the passengers that are optimized by the algorithm. A centralized algorithm that only focuses on system efficiency will inevitably result in some drivers being assigned to insufficient or undesired trips. Leaving the system as it is would affect the sustainability of the ridesourcing business model in the long run, as unsatisfied drivers will not renew their memberships and new drivers will be deterred from signing up. Therefore, fairness on the drivers' side should be assessed more carefully and should receive more attention.

In this paper, we study the batch request-vehicle assignment problem with a focus on both efficiency and fairness and address the problem of assigning requests to vehicles that account for the natural tension between these two objectives. In the basic setting, we consider a fleet $\mathcal{V}$ of available vehicles and a set $\mathcal{R}$ of ride requests. The assignment constraints are captured by a bipartite graph $G = (\mathcal{V}, \mathcal{R}, E)$, such that there is an edge $(v, r) \in E$ if vehicle v can be assigned to serve request r. Our goal is to design an assignment algorithm that matches each vehicle to at most one request, such that both efficiency and fairness are optimized. For efficiency, we adopt the utilitarian criterion, defined as the sum of values of all requests served. For fairness, we adopt the max-min fairness criterion that emphasizes maximizing the least value that a vehicle obtains. This criterion is built on the Rawlsian egalitarian justice [Rawls, 2009] and is well-recognized in different application domains.

Efficiency and fairness are often competing objectives. In most cases, the optimum of both cannot be achieved simultaneously. In this paper, we are interested in the following generic question:

Given any problem instance and any required fairness threshold, how do we find a request-vehicle assignment that meets the fairness threshold and also has sufficiently good system efficiency?

1.1 Our Contributions

Our contributions can be summarized as follows.

- We answer the above generic question with an efficient algorithm REASSIGN. Our algorithm takes any desired fairness as a parameter, and through a surprisingly simple procedure, computes an assignment with desired fairness and provably good efficiency.

- We also show that the efficiency-fairness tradeoff that our algorithm guarantees are provably optimal. That is, we prove that for any target efficiency and fairness that go beyond our guarantee, there exist problem instances in which no assignment can achieve these objectives simultaneously.
- Finally, we demonstrate the performance of our algorithm in a case study that considers taxi assignment with real taxi data from New York City. Our experiment results show that in practical scenarios, algorithm REASSIGN is able to significantly improve the fairness of the assignment with almost no loss on the system efficiency.

1.2 Related Works

The problem of vehicle-request assignment in ridesourcing has been studied extensively. Several works have focused on real-time assignment using different approaches, such as greedy match [Lee *et al.*, 2004], collaborative dispatch [Seow *et al.*, 2010; Zhang and Pavone, 2016; Ma *et al.*, 2013], planning and learning framework [Xu *et al.*, 2018], and receding horizon control approach [Miao *et al.*, 2016].

When requests do not arrive in real-time but are given beforehand, the problem is known as the *Dial-a-Ride Problem (DARP)* [Cordeau and Laporte, 2007; Nedregård, 2015]. Many variants of the dial-a-ride problem were proposed depending on the specific applications [Cordeau, 2006; Kim, 2011; Santos and Xavier, 2015; Faye and Watel, 2016; Desaulniers *et al.*, 2016; Baldacci *et al.*, 2012; Chen and Xu, 2006].

2 Preliminaries

We consider the following bipartite matching problem that models the batch assignment of a set of requests to a set of available vehicles in on-demand ridesharing. Assume a bipartite graph $G = (\mathcal{V}, \mathcal{R}, E)$ where $\mathcal{V} = \{v_1, \ldots, v_n\}$ is the set of n available vehicles and $\mathcal{R} = \{r_1, \ldots, r_m\}$ is the set of m requests. There is an edge $e = (v, r) \in E$ if the request r can be served by the vehicle v. A *utility* w_{vr} is associated to each edge (v, r) that represents the profit the vehicle could obtain by serving this request.

We make no restrictions on the structure of edge set E and allow it to encode any physical or performance-related constraints, such as that the waiting time should be within some threshold, and that the vehicle type (regular, luxury) should match the request type, etc.

We further define Δ to be the maximum utility difference of different vehicles to the same request. That is

$$\Delta = \max_{r \in \mathcal{R}} \max_{(v,r),(v',r) \in E} |w_{vr} - w_{v'r}|.$$

This maximum difference turns out to be a critical parameter in the tradeoff analysis that we will show below. In practice, the incremented utility of a request to different vehicles is usually similar to each other, which means Δ is usually a small value compared to all request utilities.

Each vehicle v also has a historical utility h_v that represents the total utility it has already obtained in previous time periods.

We refer to a setting with graph $G = (\mathcal{V}, \mathcal{R}, E)$, edge weights $\{w_{vr}\}_{(v,r) \in E}$ and historical utilities $\{h_v\}_{v \in \mathcal{V}}$ as an *instance* $\mathcal{I}$.

Given an instance $\mathcal{I}$, our goal is to find an *assignment* M that assigns each vehicle v to at most one request $M(v)$, and each request r to at most one vehicle $M(r)$. That is, M is always a *matching* in bipartite graph G.

We focus on two main objectives:

- The *efficiency* of an assignment M is defined as $\mathcal{E}(M) = \sum_v (h_v + w_{v,M(v)})$. The optimal efficiency of an instance, denoted by $\mathcal{E}_{\text{opt}}$, is the maximum efficiency of all feasible assignments, i.e., $\mathcal{E}_{\text{opt}} = \max\{\mathcal{E}(M) \mid M \in \mathcal{M}\}$.
- The *fairness* of an assignment M is defined as $\mathcal{F}(M) = \min_v\{h_v + w_{v,M(v)}\}$. The optimal fairness, denoted by $\mathcal{F}_{\text{opt}}$, is the maximum fairness of all feasible assignments, i.e., $\mathcal{F}_{\text{opt}} = \max\{\mathcal{F}(M) \mid M \in \mathcal{M}\}$.

We will refer to the assignments that produce optimal efficiency and optimal fairness as *efficient assignment* and *fair assignment*, respectively.

Our model is flexible with respect to different features that might need to be added to the system. Below we discuss how to incorporate ridesharing (i.e., carpooling) to the model.

2.1 Ridesharing

Ridesharing refers to a ridesourcing mode in which a vehicle can serve multiple (usually no more than 2) requests simultaneously. Our model can be easily adapted to allow ridesharing. More specifically, the following changes need to be made: we define a *passenger* to be a past request that has already been assigned to or picked up by a vehicle and that is now en route to its destination. For each vehicle $v \in \mathcal{V}$, we additionally maintain a set of passengers S_v of v. Then for any current request r that is waiting to be assigned, we construct edge $(v, r) \in G$ only if request r can be served by v without violating any feasibility constraints of r or any of v's passengers, such as that the total number of occupied seats is within the vehicle's capacity, and that the delay caused by detour for each passenger or request wouldn't be too excessive. The utilities from passengers should all be included in the historical utility h_v of a vehicle. Then again $h_v + w_{v,r}$ will represent the total utility of vehicle v after assigning it to r. The definitions of efficiency and fairness remain unchanged. Note that these changes only affect the structure of graph G and values of $\{h_v\}$. The overall model remains the same. Hence all the results and algorithm in Section 3 directly apply to this setting. We will describe in more details on how to define the constraints for ridesharing in our case study in Section 4.

3 Efficiency-Fairness Tradeoff

In this section we analyze the efficiency and fairness tradeoff in ridesharing. Our main result is the following theorem.

Theorem 3.1. *Given any ridesharing problem instance $\mathcal{I}$ and any $0 \leq \lambda \leq 1$, there exists an assignment M with fairness $\mathcal{F}(M) \geq \lambda\mathcal{F}_{opt}$ and efficiency $\mathcal{E}(M) \geq \frac{2}{2+\lambda}(\mathcal{E}_{opt} - n\Delta)$ simultaneously.*

Our proof is constructive. In the following we present a simple reassignment algorithm that, starting from any existing assignment M_{old}, outputs a new assignment M_{new} with any desired fairness value and bounded efficiency loss from M_{old}.

Algorithm 1: REASSIGN (G, w, h, f)

```
Input  : Instance
         $\mathcal{I} = \{G(\mathcal{V}, \mathcal{R}, E), \{w_{vr}\}_{(v,r)\in E}, \{h_v\}_{v\in\mathcal{V}}\}$,
         current assignment $M_{\text{old}}$,
         fairness threshold $f \leq \mathcal{F}_{\text{opt}}$.
Output: A new vehicle-request assignment $M_{\text{new}}$
1  Compute a fair assignment $M_{\text{fair}}$
2  Set $M_{\text{new}} = M_{\text{old}}$
3  while there exists $v \in \mathcal{V}$ such that
   $h_v + w_{v,M_{new}(v)} < f$ do
4      $r \leftarrow M_{\text{new}}(v)$
5      $M_{\text{new}}(v) \leftarrow \emptyset$
6      while there exists $v' \in \mathcal{V}$ such that
       $M_{new}(v') = M_{fair}(v)$ do
7          $M_{\text{new}}(v') \leftarrow \emptyset$
8          $M_{\text{new}}(v) \leftarrow M_{\text{fair}}(v)$
9          $v \leftarrow v'$
10     end
11     $M_{\text{new}}(v) = M_{\text{fair}}(v)$
12 end
```

Intuitively, the algorithm repeatedly chooses a vehicle v whose total utility is lower than the fairness threshold, and swap its assigned request to the one given out by the *fair assignment*, i.e. assign v to $M_{\text{fair}}(v)$. Note that this new request $M_{\text{fair}}(v)$ may be assigned to another vehicle v' in M_{new} and thus, the swapping of solution continues until no such v' can be found, as described in line 6-10 of REASSIGN.

Compute a fair assignment M_{fair}. Line 1 of REASSIGN requires us to compute a fair assignment M_{fair}. This can be done efficiently using a simple variation of the standard bipartite matching algorithm: We add n *no-serve* requests $\overline{r}_1, \ldots, \overline{r}_n$ to set $\mathcal{R}$. Each $\overline{r}_i$ has only one vehicle v_i connected to it with $w_{v_i,\overline{r}_i} = 0$. It represents the option of not assigning vehicle v_i to any requests. Let the new request set be $\mathcal{R}^+$ and the new edge set be E^+. Then for any value f, define $G_f = (\mathcal{V}, \mathcal{R}^+, E_f = \{(v,r) \in E^+ \mid h_v + w_{v,r} \geq f\})$. It is now easy to see that the optimal fairness F_{opt} is the largest value f such that G_f still has a perfect matching. Such f can be found via a binary search on all possible fairness values. And M_{fair} is a perfect matching in $G_{F_{\text{opt}}}$.

To prove Theorem 3.1, we show a more general claim about the output of REASSIGN.

Lemma 3.2. *Given instance $\mathcal{I}$, current assignment M_{old} and any fairness value $f \leq \mathcal{F}_{opt}$, algorithm* REASSIGN$(\mathcal{I}, M_{old}, f)$ *always outputs an assignment M_{new} with fairness $\mathcal{F}(M_{new}) \geq f$ and efficiency $\mathcal{E}(M_{new}) \geq \frac{2\mathcal{F}_{opt}}{2\mathcal{F}_{opt}+f}(\mathcal{E}(M_{old}) - n\Delta)$.*

The proof is omitted due to space constraints.

Finally, Theorem 3.1 can be proved directly by replacing f with λF_{opt} in Lemma 3.2.

3.1 Lower Bound

In this section we focus on the theoretical lower bound for the efficiency-fairness tradeoff that any algorithm could achieve. In particular, we show that the tradeoff achieved in Theorem 3.1 is actually tight in this model.

Theorem 3.3. *For any $0 \leq \lambda \leq 1$ and any α **strictly larger** than $\frac{2}{2+\lambda}$, there always exists a problem instance $\mathcal{I}$, such that no assignment can achieve fairness $\mathcal{F}(M) \geq \lambda\mathcal{F}_{opt}$ and efficiency $\mathcal{E}(M) \geq \alpha(\mathcal{E}_{opt} - n\Delta)$ simultaneously.*

The proof uses a simple counter-example construction and is omitted due to space constraints.

Theorem 3.1 and 3.3 together show that among all possible algorithms that can achieve a certain fairness requirement, the efficiency achieved by our algorithm REASSIGN has the best theoretical guarantee.

4 Experiments

At a first glance, the theoretical guarantee obtained in Section 3 may not be enough to convince the decision maker of a ridesourcing platform to consider fairer solutions. Because the loss in efficiency, which directly translates to a revenue loss of the platform, might be too significant for fairness considerations. For example, if one wants to adopt the fairest solution, setting $\lambda = 1$ in Theorem 3.1 shows that in the worst case the platform needs to sacrifice more than 33% of efficiency. However, as we will demonstrate in this section, in practice such worst case scenario will almost never happen. Through extensive experiments on real-world datasets, we show that when moving towards fairer solutions, the incurred loss in efficiency is much smaller than the theoretical prediction and in many cases negligible.

4.1 Dataset

We use the publicly available dataset of taxi trips in New York City [Donovan, 2016], which contains for each day the time and location of all of the pickups and drop-offs executed by each of the active taxis. We choose a representative 2-hour horizon, 1700 - 1900, and extract all requests originating and finishing within Manhattan, happening in May 2013. We consider the recorded pickup time as the request arrival time and the recorded passenger count as the request size. There are between 31,694 to 56,743 extracted requests each day. To reflect real road conditions and traveling time, we construct a road network of Manhattan with 3,671 nodes and 7,674 edges. For the requests, we round their original pickup and drop-off location to these nodes, by finding the closest node to each recorded coordinate. Travel time on each road or edge of the network is estimated based on the daily mean travel time estimate following the method in [Santi *et al.*, 2014]. Shortest paths and travel times between all nodes are precomputed and stored in a look-up table.

4.2 Construction of Bipartite Graph

Following, we describe the relevant definitions and assumptions in the construction of $G = (\mathcal{V}, \mathcal{R}, E)$.

First, we set the following constraints in the construction of the edge set E in G[1]: some vehicle v and request r is connected by an edge (v, r) if and only if (i) for each request r, the difference between its pick-up time and request time should be within a threshold Ω; (ii) the total travel delay time, which is the difference between the actual drop-off time and the earliest possible drop-off time should be within a threshold Γ; (iii) if ridesharing is allowed, the total number of passengers on the vehicle at any time must not exceed the vehicle capacity χ. For simplification purposes, we also require that any vehicles can serve up to two requests at any time.

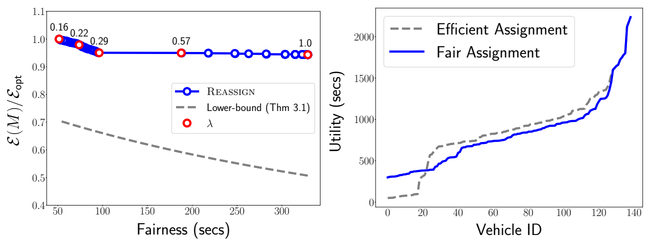
For each edge $(v, r) \in E$, we set its weight as $w_{vr} = c\tau_{vr} - \iota_{vr}$. Here τ_{vr} is the shortest time needed to travel from v to r, and c is a ratio constant. We use $c\tau_{vr}$ to approximate the value of the trip. ι_{vr} is the vehicle idle time associated with this trip, which is defined as the difference between the pick-up time of request r and the maximum of the assignment time of r and the last drop-off time of v. In the case of ridesharing and there are already passengers in v when picking up r, we set ι_{vr} to 0. Finally, we remove all edges whose weights are negative from the graph.

4.3 Experimental Setup and Data Preprocessing

To conduct our experiment, we create several cross-sectional scenarios when vehicles have been on the road for some time and are available to serve new request. We pick several days where there are more than 200 requests arriving in the first 30 seconds of its 1700 to 1900 period and consider all requests with trip length at least 400s such that we have $m = |\mathcal{R}| \in [105, 142]$. Upon initialization, we locate $n = 1.2m$ vehicles within a reasonable time-distance from the requests. We define two groups of vehicles, $\mathcal{V}_H$ and $\mathcal{V}_L$, to introduce some level of discrepancies in historical utilities and randomly generate h_v such that for every $v \in \mathcal{V}_H, h_v \sim U(200, 400)$ and for every $v \in \mathcal{V}_L, h_v \sim U(50, 100)$. Finally, we set the maximum waiting time $\Omega = 210$s, parameter $c = 1$, and capacity $\chi = 4$. We pick 10 different days and test our algorithm on each day. All data below are the average results from these 10 days.

4.4 Results

Figure 1(a) demonstrates the tradeoff between efficiency and fairness when applying our algorithm REASSIGN with different values of fairness threshold f. As one can see from the figure, when efficiency is the only concern (corresponding to the leftmost point), the resulting assignment may have the lowest utility of all drivers as low as 51. However, as we start applying REASSIGN with higher and higher fairness thresholds, this lowest utility value gradually improves, until it reaches the highest point $\mathcal{F}_{\text{opt}} = 328$ in the fair solution (corresponding to the rightmost point). For the worst-off driver, the utility improvement from efficient assignment to the fair assignment is over 6-fold.



(a) Efficiency-Fairness Tradeoff (b) Utilities

Figure 1: (a) The efficiency and fairness of the assignments output by REASSIGN with respect to different fairness thresholds, compared with the theoretical lower bound implied by Thm 3.1 (b) The utilities of all vehicles in the efficient assignment and the fair assignment, both sorted from smallest to largest.

In the meanwhile, although Theorem 3.1 and 3.3 claim that the maximum efficiency loss may be as high as 33% (as indicated by the dashed curve in Figure 1(a)) in the worst case. In reality, this loss is much smaller. In this example, the largest efficiency loss is less than 6% from the optimal efficiency. We also tested several other datasets from different days and with different parameters. While preserving similar magnitudes of fairness improvement, their efficiency losses are even smaller, with many under 1%. In summary, worst-case efficiency loss rarely arises from artificial examples; here we see real-world problem instances with much more benign behavior.

We also compare the final utilities of all vehicles in the *efficient* and *fair* matching. Figure 1(b) shows these two sets of utilities, both sorted from smallest to largest. It is evident that our algorithm manages to redistribute the utility increments to the vehicles with low utilities, without sacrificing too much on the efficiency.

Overall these results suggest that in practical scenarios, considering fair solutions should be a appealing option for every ridesourcing enterprise, since it is often possible to improve the fairness of the assignment significantly without too much sacrifice on the system efficiency.

5 Conclusion

In this paper, we deal with the problem of balancing efficiency and fairness in the context of ridesourcing request assignment. We present a simple reassignment algorithm that can compute an assignment with any desired fairness and provably good efficiency. Experiment results show that in practical scenarios, this algorithm is able to significantly improve the fairness of the assignment to drivers with very little loss on the system efficiency.

The theoretical bounds derived in our work are of independent interest and can be applied to a broader family of matching problems. How to find other suitable applications in which similar techniques or results can be applied is in our opinion a very interesting future working direction. Another question for future study is to consider different fairness criteria, such as proportional fairness [Kelly *et al.*, 1998], and measure their tradeoffs with efficiency.

[1]These are the same set of rules used in [Alonso-Mora *et al.*, 2017].

References

[Alonso-Mora *et al.*, 2017] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467, 2017.

[Baldacci *et al.*, 2012] Roberto Baldacci, Aristide Mingozzi, and Roberto Roberti. Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. *European Journal of Operational Research*, 218(1):1–6, 2012.

[Chen and Xu, 2006] Zhi-Long Chen and Hang Xu. Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science*, 40(1):74–88, 2006.

[Cordeau and Laporte, 2007] Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: models and algorithms. *Annals of operations research*, 153(1):29–46, 2007.

[Cordeau, 2006] Jean-François Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586, 2006.

[Desaulniers *et al.*, 2016] Guy Desaulniers, Fausto Errico, Stefan Irnich, and Michael Schneider. Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, 64(6):1388–1405, 2016.

[Donovan, 2016] Dan Donovan, Brian; Work. New york city taxi trip data (2010-2013), 2016.

[Faye and Watel, 2016] Alain Faye and Dimitri Watel. Static dial-a-ride problem with money as an incentive: Study of the cost constraint. 2016.

[Kelly *et al.*, 1998] Frank P Kelly, Aman K Maulloo, and David KH Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research society*, 49(3):237–252, 1998.

[Kim, 2011] Taehyeong Kim. *Model and algorithm for solving real time dial-a-ride problem*. PhD thesis, 2011.

[Lee *et al.*, 2004] Der-Horng Lee, Hao Wang, Ruey Cheu, and Siew Teo. Taxi dispatch system based on current demands and real-time traffic conditions. *Transportation Research Record: Journal of the Transportation Research Board*, (1882):193–200, 2004.

[Li *et al.*, 2018] Yanwei Li, Araz Taeihagh, and Martin de Jong. The governance of risks in ridesharing: A revelatory case from singapore. *Energies*, 11(5):1277, 2018.

[Lin *et al.*, 2018] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. Efficient large-scale fleet management via multi-agent deep reinforcement learning. *arXiv preprint arXiv:1802.06444*, 2018.

[Ma *et al.*, 2013] Shuo Ma, Yu Zheng, and Ouri Wolfson. T-share: A large-scale dynamic taxi ridesharing service. In *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, pages 410–421. IEEE, 2013.

[Miao *et al.*, 2016] Fei Miao, Shuo Han, Shan Lin, John A Stankovic, Desheng Zhang, Sirajum Munir, Hua Huang, Tian He, and George J Pappas. Taxi dispatch with real-time sensing data in metropolitan areas: A receding horizon control approach. *IEEE Transactions on Automation Science and Engineering*, 13(2):463–478, 2016.

[Nedregård, 2015] Ida Nedregård. The integrated dial-a-ride problem-balancing costs and convenience. Master's thesis, NTNU, 2015.

[Rawls, 2009] John Rawls. *A theory of justice*. Harvard university press, 2009.

[Santi *et al.*, 2014] Paolo Santi, Giovanni Resta, Michael Szell, Stanislav Sobolevsky, Steven H Strogatz, and Carlo Ratti. Quantifying the benefits of vehicle pooling with shareability networks. *Proceedings of the National Academy of Sciences*, 111(37):13290–13294, 2014.

[Santos and Xavier, 2015] Douglas O Santos and Eduardo C Xavier. Taxi and ride sharing: A dynamic dial-a-ride problem with money as an incentive. *Expert Systems with Applications*, 42(19):6728–6737, 2015.

[Seow *et al.*, 2010] Kiam Tian Seow, Nam Hai Dang, and Der-Horng Lee. A collaborative multiagent taxi-dispatch system. *IEEE Transactions on Automation Science and Engineering*, 7(3):607–616, 2010.

[Xu *et al.*, 2018] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping 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. ACM, 2018.

[Zhang and Pavone, 2016] Rick Zhang and Marco Pavone. Control of robotic mobility-on-demand systems: a queueing-theoretical perspective. *The International Journal of Robotics Research*, 35(1-3):186–203, 2016.