Intent Matters: Improving Zonal Fairness of Ridesharing Systems

Ashwin Kumar, Yevgeniy Vorobeychik, William Yeoh
Washington University in St. Louis

Abstract

Order dispatching algorithms, which match passenger requests with vehicles (agents) in ridesharing systems, are able to achieve high service rates (percentage of requests served) using deep reinforcement learning techniques to estimate the relative values of the different combinations of passenger-vehicle matches. While the goal of such algorithms is to maximize the service rate, this may lead to unintended fairness issues (e.g., high disparity between the service rates of zones in a city). To remedy this limitation, researchers have recently proposed deep reinforcement learning based techniques that incorporates fairness components in the value function approximated. However, this approach suffers from the need to retrain should one wish to tune the degree of fairness or optimize for a different fairness function, which can be computationally expensive. Towards this end, we propose a simpler online approach that uses state-of-art deep reinforcement learning techniques and augments their value functions with fairness components during the matching optimization step. As no additional training is needed, this approach can trivially be adapted to use any existing value function approximator (not limited to those by deep reinforcement learning techniques) and benefits from improved flexibility in evaluating different fairness objectives efficiently. In this paper, we describe several fairness functions that can be used by this approach and evaluate them against existing state-of-the-art deep RL based fairness techniques on standard ridesharing benchmarks. Our experiments show that our fairness functions outperform existing fairness techniques (i.e., it finds matching solutions that result in higher service rates *and* lower service rate disparity across zones), demonstrating the practical promise of this approach.

Introduction

On-demand ridesharing has been gaining traction over the past few years as a solution to the growing need for urban mobility. Providing low cost rides to passengers at the expense of small detours, higher earnings for drivers, and a way to reduce the number of vehicles on the streets, it is a solution that benefits everyone. Consequently, there has been work on finding efficient matches of passengers and vehicles to minimize delays and maximize system efficiency for such ecosystems. Recent approaches have used concepts from dynamic programming and deep reinforcement learning to learn policies for matching pools of passenger *requests* to available drivers, achieving real-time performance capabilities. Some approaches, such as that by Alonso-Mora et al. (2017), have also looked at this problem through the lens of managing fleets of high-capacity autonomous vehicles. These advances have led to significant improvements in the performance of these algorithms vis-a-vis the *service rate* (i.e., the fraction of passenger requests served out of all requests made) as well as the total delay passengers have to experience (Shah, Lowalekar, and Varakantham 2020; Lowalekar, Varakantham, and Jaillet 2019; Li et al. 2019).

Optimizing for such metrics, however, can have unintended effects on the fairness of such systems. The issue of fairness in ridesharing has been discussed in various contexts, ranging from passenger-side fairness in terms of delay and partner choice, driver-side fairness in terms of equitable earning, and even sub-group level fairness (Nanda et al. 2020; Xu and Xu 2020) for drivers and passengers. Sühr et al. (2019) outlines the three stakeholders in the ridesharing ecosystem: (i) the taxis/drivers, (ii) the customers, and (iii) the platform (Uber, Lyft, etc.). The platform takes on the job of mediating between customers and drivers, a problem that is often cast as an Integer-Linear Program (ILP) maximizing overall driver preferences over incoming requests. This central role situates ridesharing platforms to enforce fairness via balancing it with efficiency. Most fairness research in ridesharing aims to quantify this tradeoff in one way or the other as part of the objective function, by intervening in the matching process through a central mechanism.

In our approach, we view the problem in the context of a fleet of autonomous vehicles controlled by the platform. The optimization is formulated as a reinforcement learning problem, drawing from state-of-art techniques (Shah, Lowalekar, and Varakantham 2020; Nanda et al. 2020) to combine the best performing algorithms with fairness approaches. Using zonal fairness as an example of sub-group fairness for passengers, we propose an online approach to incorporate subgroup fairness into ILP-based matching algorithms, without needing to retrain the pre-learned function approximators. Further, we propose designating only a small fraction of the

vehicles or requests as "*fairness-aware*" and postulate that this solution is better at trading off efficiency and fairness compared to applying the same modifications to all vehicles. We perform experiments to demonstrate the efficacy of the approach, and perform a grid search over hyperparameters to qualitatively analyze the fairness-efficiency tradeoff. Our experimental results show that our methods outperforms existing techniques (i.e., it finds matching solutions that result in higher service rates *and* lower service rate disparity across zones), demonstrating the practical promise of this approach.

Specifically, our contributions are as follows:

- We develop an online framework that uses off-the-shelf value function approximators and user-defined fairness objectives to trade off efficiency for fairness using state-of-art matching algorithms.
- We introduce new metrics for zonal fairness that consider fairness across source-destination zone pairs.
- We provide real-time tuneable parameters for changing between different types and degrees of fairness, allowing systems to adapt on the fly, and compare the results to existing benchmarks.

Background and Related Work

Order Dispatching in Ridesharing

While there are many variants of *Ridesharing Matching Problems* (RMPs) (Huang et al. 2014; Alonso-Mora et al. 2017; Ma, Zheng, and Wolfson 2015; Simonetto, Monteil, and Gambella 2019; Shah, Lowalekar, and Varakantham 2020; Lowalekar, Varakantham, and Jaillet 2019; 2020; Wang et al. 2020), in a typical problem definition, a matching algorithm receives as inputs a continuous stream of batches of requests from passengers $\mathcal{R}$ and the current state of all the taxis $\mathcal{V}$, operating in a street network $\mathcal{G}$.

A request $r_i = \langle s_i, g_i, t_i \rangle$ contains the pickup location s_i, dropoff location g_i and the arrival time of the request t_i. The vehicle state $v_i = \langle l_i, p_i, c_i, r_i \rangle$ includes its location l_i, current path p_i, capacity c_i and the requests it is currently serving r_i. The street network $\mathcal{G} = \langle \mathcal{L}, \mathcal{E}, c(e) \rangle$ is a graph containing locations $\mathcal{L}$ connected by roads $\mathcal{E}$, with a cost function $c : \mathcal{E} \rightarrow R^+$ that defines the cost $c(e)$ of each edge $e \in \mathcal{E}$ in the graph. Intuitively, it corresponds to the time needed by a taxi to traverse that edge on the graph.

The matching algorithm then needs to match requests to vehicles given some time and capacity constraints $\mathcal{C}$, while optimizing some metric (e.g., maximizing the service rate). In some approaches, this process is repeated at regular intervals (e.g., every minute), and requests are accumulated during this time window. Each such matching iteration is called a *decision epoch*. An *assignment* $A = \{(r_i, v_j), \ldots\}$ to this RMP is a set of matches between request $r_i \in \mathcal{R}$ and taxi $v_j \in \mathcal{V}$ such that all the constraints in $\mathcal{C}$ are satisfied.

RMPs have been extensively studied, where researchers have introduced methods that improve the quality of the matches made in terms of increasing the number of requests matched (Lowalekar, Varakantham, and Jaillet 2019; 2020; Ma, Zheng, and Wolfson 2015), reducing the pickup and detour delays (Alonso-Mora et al. 2017; Huang et al. 2014; Ma, Zheng, and Wolfson 2015), and increasing the revenue of the drivers (Lesmana, Zhang, and Bei 2019). The complexity of RMP algorithms and, as a result, the time taken by algorithms to match drivers to passenger requests, increases with the increase in the number or vehicles and the capacity of each vehicle. As the runtime of real-time RMP algorithms need to be relatively small, most existing work has either considered assigning one request at a time (sequentially) to available drivers for high capacities (Ma, Zheng, and Wolfson 2015; Tong et al. 2018; Huang et al. 2014) or assigning all active requests together in a batch for a small capacity (Yu and Shen 2019; Zheng, Chen, and Ye 2018). The sequential solution is faster to compute but the solution quality is typically poor (Uber 2018). Alonso-Mora et al. (2017) proposes integer optimization approaches for assigning all active requests together for high-capacity ridesharing. Shah, Lowalekar, and Varakantham (2020) and Lowalekar, Varakantham, and Jaillet (2020) further improve these approaches by including information about anticipated future requests while matching current batch of requests to available drivers. Distributed approaches (Li et al. 2019) cast this as a multi-agent reinforcement learning problem and asynchronously execute matches for each vehicle, taking into account the demand-supply mismatch across zones. This involves some level of communication between nearby vehicles, as opposed to a single central agent that observes all information.

There have been multiple interpretations of the term "ridesharing" in the literature. In our work, we define ridesharing to be a system where multi-capacity vehicles allow passengers to share rides with others or be added onto existing trips, and a single central agent (the platform) aggregates all information and matches passenger requests to available vehicles dynamically. We build on the approach taken by Shah, Lowalekar, and Varakantham (2020), using integer optimization combined with deep reinforcement learning to rank matches and find the best assignment.

Fairness in Ridesharing

Researchers have evaluated ridesharing fairness from many viewpoints. One of the first papers to discuss this concept (Wolfson and Lin 2017) looks at passenger-side fairness, specifically addressing the lack of transparency in such systems. They propose a fair benefit-sharing approach based on passengers ranking their ridesharing partners. Foti, Lin, and Wolfson (2019) extend this by formulating an ILP for maximum matching and using a game-theoretic approach to fairness. They compare the fair and optimal solutions and discuss theoretical and practical bounds between the differences.

Gopalakrishnan, Mukherjee, and Tulabandhula (2016) approaches passenger-side fairness from the perspective of non-increasing disutility (i.e., the addition of new passengers onto the trip shouldn't decrease the utility of the existing passengers). They propose a method of sharing the benefits due to additional customers that solves this problem.

Driver-side fairness (Lesmana, Zhang, and Bei 2019; Sühr et al. 2019) has also been explored from the economic perspective. Lesmana, Zhang, and Bei (2019) designs a dual

objective, maximizing efficiency and fairness, and describe methods to balance between the two. They approach driver-side fairness by using a max-min approach, maximizing the worst-off vehicle. Another line of work (Sühr et al. 2019) aims to equalize driver income proportional to the number of hours spent on the platform, looking at fairness for both drivers and passengers over longer periods of time.

Fairness isn't restricted to monetary benefits, however. Motivated by demographic and geographic fairness concerns, Nanda et al. (2020) and Xu and Xu (2020) formulate a bipartite matching problem with parameters to trade profit for fairness. For passengers (Nanda et al. 2020), this unfairness might result from factors like start/end locations, race, gender, or age, which may lead to drivers canceling requests. Similarly, for drivers (Xu and Xu 2020), there might exist an income inequality due to discriminatory cancellations by passengers. Most of these approaches discussed above use a naive nearest-vehicle-first or myopic batched allocation approach towards efficiency. However, as discussed earlier, sophisticated methods have been developed that can increase the efficiency of the system.

One recent work (Raman, Shah, and Dickerson 2020) looks at disparate treatment of passengers and income disparity amongst drivers in a system that implements a state-of-the-art matching algorithm. This makes their work the closest to ours in terms of scope. While they also look at geographic zones to quantify fairness for passengers, their approach requires the training of a neural network based value function to include the fairness term in the objective, making it costly to change parameters for fairness. Our approach presents an online way to deal with this problem, without retraining existing value functions. This also allows for the tuning of fairness parameters in real time, in response to changing conditions. Further, our approach offers better tradeoffs between efficiency and fairness as compared to the existing approach, and we show this in our empirical evaluation.

Matching using Reinforcement Learning

As mentioned in the background, a *Ridesharing Matching Problem* is a tuple $RMP = \langle \mathcal{R}, \mathcal{V}, \mathcal{G}, \mathcal{C} \rangle$ consisting of the set of requests $\mathcal{R}$, vehicles $\mathcal{V}$, road network $\mathcal{G}$, and constraints $\mathcal{C}$. The solution to this problem is an assignment $\mathcal{A}$ that satisfies the constraints and provides a matching between vehicles and requests.

We now describe how NeurADP (Shah, Lowalekar, and Varakantham 2020), a state-of-the-art deep reinforcement learning based method, solves this problem. (Our approach is based on NeurADP.) NeurADP learns a *Value Function Approximator* (VFA) for a vehicle's state, which approximates the expected future value of being in that particular state, using a deep neural network. For each vehicle, it generates a set of feasible trips (Alonso-Mora et al. 2017) and score them using the VFA. Then, it uses an *Integer Linear Program* (ILP) to maximize the cumulative score over all vehicles.

In typical reinforcement learning fashion, there is an agent and an environment. The environment simulates the motion of vehicles and the arrival of requests with the passage of

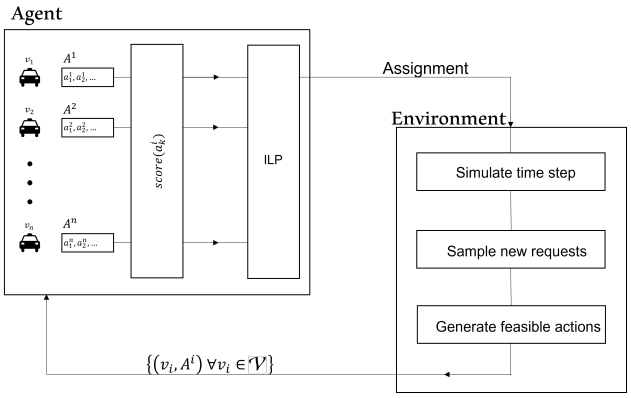

Figure 1: The RMP Pipeline

time. Each decision epoch, the agent takes the current state of vehicles and pending requests as input and solves the RMP, matching requests to vehicles. The agent in our setting involves the combination of a VFA for each vehicle, combined with the ILP that finds the optimal assignment based on the values. Using the VFA in this fashion to predict expected returns allows it to make non-myopic decisions that eventually improve the performance of the system.

The VFA is learned based on TD-learning (Sutton 1988) using experience replay, techniques popularly used in DQNs (Mnih et al. 2013). We observe transitions from the environment, which are stored in a buffer and later sampled in mini-batches to learn from. All vehicles share the same value function, allowing for efficient reuse of experiences. The training process and network architecture used for the VFA is identical to the one used by Shah, Lowalekar, and Varakantham (2020), and we refer the reader to that paper for more details.

We define an *action* as the matching of a set of requests to a vehicle. If vehicle v_i services request r_j, we denote the reward for the vehicle as $R(v_i, r_j)$. If the set of available (feasible) actions for vehicle v_i is A^i, then, for each action $a^i_k \in A^i$, the score is the corresponding immediate rewards obtained for servicing those requests plus the expected future value of the new vehicle state after being assigned those requests:

$$score(a^i_k) = V(s') + R(a^i_k) \tag{1}$$

where $V(\cdot)$ is the VFA and s' is the new state of the vehicle v_i after accepting all requests $r_j \in a^i_k$. As a shorthand, overloading some notation, we can write the reward for an action as:

$$R(a^i_k) = \sum_{r_j \in a^i_k} R(v_i, r_j) \tag{2}$$

Let o^i_k be an indicator variable for whether action a^i_k was selected for vehicle v_i. Then, we can write the objective function of the ILP as follows:

$$\max \sum_{i \in |\mathcal{V}|} \sum_{a^i_k \in A^i} o^i_k \times score(a^i_k) \tag{3}$$

subject to the constraints:

$$\sum_{a_k^i \in A^i} o_k^i = 1, \forall i \in |\mathcal{V}| \tag{4}$$

$$\sum_{i \in |\mathcal{V}|} \sum_{a_k^i \in A^i \text{ s.t. } r_j \in a_k^i} o_k^i \leq 1, \forall r_j \in \mathcal{R} \tag{5}$$

$$o_k^i \in \{0,1\}, \forall i \tag{6}$$

Intuitively, the constraints ensure that each vehicle is assigned exactly one action (Eq. 4) and no request is assigned to more than one vehicle (Eq. 5). In each vehicle's set of available actions, there is always the null action (i.e., accepting no new requests), so that there is always a solution. The final assignment $\mathcal{A}$ is a concatenation of all vehicle assignments.

The objective function in NeurADP (Shah, Lowalekar, and Varakantham 2020) is to maximize the number of requests served.As the reward for each request is uniform and the VFA's predictions are a discounted estimate of the number of requests the vehicle will pick up in the future, the efficiency can be measured using the *service rate*, which is the fraction of requests assigned out of all requests that were made over a given duration.

Zonal Fairness

As discussed in earlier sections, there has been interest in preserving fairness across sub-groups of the passenger and driver populations. In our approach, we focus on the passengers, looking at service rate fairness across *zone pairs* for pickup and dropoff locations. The motivation to select this metric is two-fold: (i) It is an intuitive metric and is consistent with other fairness objectives that aims for equity in efficiency across sub-groups; and (ii) It is difficult to get information about race, ethnicity, and other cultural factors from publicly-available ridesharing datasets.

Zonal fairness becomes a concern with algorithms like NeurADP (Shah, Lowalekar, and Varakantham 2020), which aim to maximize the number of passengers served. In such cases, if most of the demand is concentrated within a certain area, the model will learn to prefer requests to/from that area. Areas with lesser demand are ignored in favor of the overall maximization objective, resulting in the algorithm sending a large proportion of the vehicle population towards regions of high demand. As we discuss in the following sections, small steps towards fairness can go a long way, because serving only a small number of requests in the less-serviced areas can improve the inequity, possibly at only a marginal cost to the overall efficiency of the system.

We define m zones as disjoint subsets of the locations $\mathcal{L}$ in the city graph. Our sub-groups of interest are people moving between pairs of zones. Recent work (Raman, Shah, and Dickerson 2020) aims to maximize the minimum service rate by including variance in service rates across different zones in the optimization objective. Specifically, they look at the service rate by source zone, which we denote as z_i (i.e., the service rate for all requests originating in zone i).

However, this metric does not take into account the destination of the requests, and there may be a disparity between fairness based on source zones and fairness based on zone-pairs, where some source-destination pairs may be severely under-served. To address this limitation, we define zone-pair service rate z_{ij} as the service rate for requests originating in zone i and ending in zone j. This gives us a grid $\mathbf{Z}_p$ of zone-pair service rates of size $m \times m$, in addition to a vector $\mathbf{Z}_s$ of source-zone service rates of size m.

Given these two sub-group statistics, we can compute fairness metrics of interest, namely the minimum service rate (by source zone or zone-pair) and the Gini coefficient for the service rates (also by source zone or zone-pair). We treat each zone/zone-pair as an individual and fairness is defined by how high the minimum service rate is (Lesmana, Zhang, and Bei 2019) and by how low the Gini coefficient is. We would like to note that these metrics and definitions of fairness are just examples, guided by recent literature, and we do not claim that these are the only metrics that matter. Instead, fairness is nuanced and subjective, and we aim to show how one can improve the fairness for some particular definitions.

Optimizing Zonal Fairness

The key idea of our approach is that the score function in Equation 1 can be modified to encourage particular request-vehicle matches by adding a bonus to particular vehicles servicing requests that would improve fairness. This bonus can be tuned to suit the need for fairness, and the original score function can be used otherwise. This gives us the flexibility of using the best value functions for general operation to maintain efficiency, and using the bonus to guide the system towards fairer decisions. It is important to note that we do not retrain the value function, only using predictions from an existing one.

Specifically, we define two fairness scores for source-zone (f_s) and zone-pair (f_p) fairness.

$$f_s(r) = \frac{||\mathbf{Z}_s||_1}{m} - z_i \tag{7}$$

$$f_p(r) = \frac{||\mathbf{Z}_p||_1}{m^2} - z_{ij} \tag{8}$$

Here, r is the request of interest, and i and j are the source and destination zones of the request. Intuitively, this score captures how much worse the request's source zone (zone-pair) is, compared to the average score. A positive score means it is worse. For ease of representation, we also overload this function when used for an action a as:

$$f_s(a) = \sum_{r \in a} f_s(r) \tag{9}$$

$$f_p(a) = \sum_{r \in a} f_p(r) \tag{10}$$

We propose a few methods that use the fairness score in conjunction with the ILP in different ways to improve fairness. Each approach modifies the score function (Eq. 1), and can be used with either f_p or f_s, but we find that f_p is empirically stronger.

Fairness-aware Vehicles: In this method, we designate a fraction α of the vehicles as "fairness-aware," and these

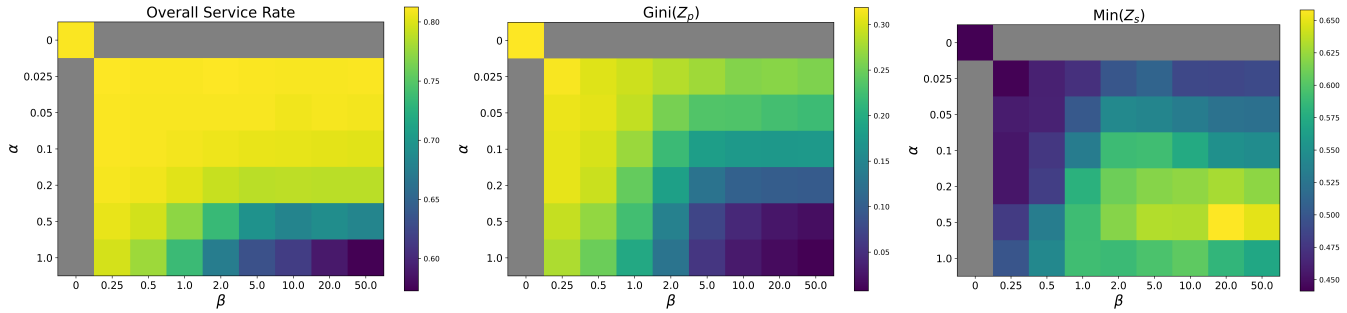

Figure 2: Trends for change in service rate, zone-pair Gini and minimum source-zone service rate with changing α and β for α-veh. The top left ($\alpha = 0, \beta = 0$) indicates the value for unmodified NeurADP.

vehicles receive the fairness bonus. Let $\mathcal{V}_\alpha$ denote the set of fairness-aware vehicles. We then have the following two variants:

- **α-fair vehicles (α-Veh):** The fairness bonus is added to the existing score function for $\mathcal{V}_\alpha$ with weight β. In other words, Equation 1 is replaced with:

$$score(a_k^i) = \begin{cases} V(s') + R(a_k^i) + \beta f_\cdot(a_k^i), & \text{if } v_i \in \mathcal{V}_\alpha \\ V(s') + R(a_k^i), & \text{otherwise} \end{cases} \tag{11}$$

- **α-exclusive-fair vehicles (Xα-Veh):** $\mathcal{V}_\alpha$ use only the reward and the fairness score, but not the estimated future value from the neural network. Thus, the fairness-aware vehicles are not concerned with maximizing the long-term value, just the fairness.

$$score(a_k^i) = \begin{cases} R(a_k^i) + \beta f_\cdot(a_k^i), & \text{if } v_i \in \mathcal{V}_\alpha \\ V(s') + R(a_k^i), & \text{otherwise} \end{cases} \tag{12}$$

Request-based Fairness: While having designated vehicles trying to enforce fairness is one solution, it is possible that those vehicles aren't in the right place at the right time and are thus unable to help improve the inequity. To combat this, we can instead look at fairness on a per-request basis, adding the fairness bonus to the rewards of certain requests. Thus, vehicles will be incentivized to pick up requests that have the fairness bonus, even if the value function gives it a lower score. For this approach, we modify the reward $R(a_k^i)$. Let $\mathcal{R}_f$ denote the set of requests with the fairness bonus. We then have the following new reward function to replace Equation 2:

$$R(a_k^i) = \begin{cases} \sum_{r_j \in a_k^i} \left(R(v_i, r_j) + \beta f_\cdot(r_j)\right), & r_j \in \mathcal{R}_f \\ \sum_{r_j \in a_k^i} R(v_i, r_j), & \text{otherwise} \end{cases} \tag{13}$$

To select which requests get the fairness bonus, we propose two methods:

- **α-subset of requests (α-Req)**: Using a pair of hyperparameters (α, β) similar to the two used for fairness-aware vehicles, this approach selects requests by ranking all requests by decreasing $f_\cdot(r)$ and selecting the top α fraction of requests.
- **Positive fairness score requests (+Req)**: In this approach, after ranking requests (as in α-**Req**), we select all the requests that have a positive $f_\cdot(r)$. It amounts to assigning a bonus to all requests going between zones whose service rate is worse than the average service rate.

We can vary α and β to change the degree of fairness we expect to see in the system. Each method may have a different response to the exact values, but in general, we expect larger values of α and β to improve the fairness objective, while causing a decrease in the overall service rate.

Experimental Setup

To evaluate the efficacy of the different approaches, we evaluate the performance and fairness metrics after running the matching algorithm over a 24-hour period on the island of Manhattan using demand data from the NY Yellow Taxi dataset (NYC Taxi & Limousine Commission 2020). The locations in the road network correspond to street intersections, with edges as roads connecting them. We define zones as all intersections falling within neighborhoods on Manhattan island.[1]

We use pre-trained NeurADP models (Shah, Lowalekar, and Varakantham 2020) as the base value function. We use the method proposed by Raman, Shah, and Dickerson (2020) as a baseline for fairness, using their passenger-side fairness implementation, which we call FairNN. FairNN uses one hyper-parameter (λ) to trade off profit and fairness. In the original paper, experiments were performed for 50 and 200 vehicles with a capacity of 4, for $\lambda \in \{10^8, 10^9, 10^{10}\}$. We use the same parameter settings for our experiments for a fair comparison to the baseline.

However, even with 200 vehicles, the demand from requests saturates the fleet's capacity. To simulate a more realistic scenario, we run the experiments with 1000 vehicles as well, similar to the work by (Shah, Lowalekar, and Varakantham 2020). For this setting, we also run FairNN with a wider range of λ ($10^3 - 10^{11}$), to more clearly see the trend. Note that each setting of λ requires retraining the value function for FairNN, which is costly.

[1] https://github.com/erikgregorywebb/nyc-housing/blob/master/Data/nyc-zip-codes.csv

For the methods discussed in this paper (α-Veh, Xα-Veh, α-Req, +Req), we perform a grid-search over the hyperparameters α and β. We run this search using both f_p and f_s as the fairness scores, independently. For each setting, we find the overall service rate, the Gini coefficient of the service rate by source zone (Gini($\mathbf{Z}_s$)) and by zone-pair (Gini($\mathbf{Z}_p$)), and the minimum service rate for any source zone (min($\mathbf{Z}_s$)) or zone-pair (min($\mathbf{Z}_P$)).

Experimental Results

In this section, we go over the key results from our experiments. The full set of experiments had over 900 runs;[2] we thus present selected results that illustrate the overall trend. We discuss the results from the analysis of the 1000 vehicle case here, though the trends hold for 200 and 50 vehicles as well.

Fairness-Efficiency Tradeoff

Figure 2 shows how the various metrics change with changing hyperparameters for α-Veh, using the zone-pair fairness score f_p. β controls the weight of the fairness score and α is the fraction of fairness-aware vehicles. We see that as α and β increase (stronger fairness), overall service rate decreases and the fairness objective used in the fairness score (here, Gini($\mathbf{Z}_p$)) improves. Other fairness metrics like the min($\mathbf{Z}_s$) also improve, but the change is not monotonic, as can be seen in the third heatmap. This trend is seen across all different methods and both fairness scores. This suggests that there is a direct tradeoff between efficiency and the selected fairness metric, and it is possible to use the other metrics to select a suitable value for the hyperparameters.

Further, this shows that there is merit in applying the fairness score to a subset of the vehicle population, as the change in service rate along the β axis is slower, while still giving gains for fairness. This is also shown by the fact that min($\mathbf{Z}_s$) is maximized when half the vehicles are fair. Specifically, it does not lie at $\alpha = 1$. We also observe that there are diminishing returns with respect to the fairness utility as α increases. As shown in Figure 2, doubling the number of fair vehicles does not double the fairness gain.

Source-Zone Fairness vs Zone-Pair Fairness

Figure 3 shows the Pareto frontiers showing the tradeoffs between service rate and different metrics of fairness for each method. Runs with the zone-pair fairness score f_p are shown with dotted lines, and runs with their source-zone f_s counterparts are shown with solid lines of the same color. In every case, the solid lines are Pareto-dominated by the dotted lines. This shows that zone-pair fairness is a stronger fairness objective, as the performance using f_p is better than f_s even when looking at fairness by source zone. This can be explained by the fact that f_p contains more information that is specific to the request, and looks at fairness at a higher granularity. f_s would be the same among nearby requests, and thus any vehicle having a choice between such requests wouldn't be influenced by the fairness score. The remaining analysis thus focuses on just the methods using f_p as the fairness score (the dotted lines in the Pareto curves).

[2]The complete results can be found at https://github.com/ashwinkwashu/TRASE22_Zonal-Fairness-Data.

Comparison to Baselines

All methods provide significant improvements over the base NeurADP algorithm in terms of fairness, while reducing the service rate by different amounts (Figure 3). This shows that our approaches perform as intended, empirically. We also observe that FairNN (Raman, Shah, and Dickerson 2020) is Pareto-dominated by most of our methods across different fairness metrics. Even within runs for FairNN with different λ, most points are Pareto-dominated by one (or a few) points. The best performance is at $\lambda = 10^7$, which is in the middle of the range we investigated, suggesting that the improvements are not monotonic with λ for this approach. This shows that our approach has a significant advantage, combined with the fact that FairNN requires retraining for each hyperparameter setting, while our method can be adjusted on the fly with no training required.

Relative Performance of Our Methods

Amongst all our methods, we find that α-Req and +Req with f_p perform the best across the board, while α-Veh and Xα-Veh have similar performance. +Req is a special case of α-Req, where the number of requests selected is not static. It is interesting to note that both have similar performance despite +Req having only one hyperparameter β compared to α-Req, which has two hyperparameters α and β. Thus, **+Req is our best approach**.

To explain the performance difference between the vehicle-based and request-based approaches, we can consider the fact that while there might be dedicated vehicles willing to improve fairness, they also need to be in the right place to serve requests that improve fairness. Request-based fairness methods do not face this issue as any request that improves fairness can be directly given a bonus, which indirectly gets transferred to the nearby vehicles. However, the advantage of vehicle-based approaches remains in the fact that their response to changing hyperparameters is more regular, allowing for better control without too many grid searches. We make this claim as a general observation without proof, and one can look at the full set of results to further support this.

While the general trend for each method can be seen through Figures 2 and 3, it is useful to contextualize the results from each of the methods by selecting a representative hyperparameter setting. To do this, we select the hyperparameter value for each method using f_p that maximizes min($\mathbf{Z}_s$). This emulates one way of selecting the best hyperparameter that could be used in practice, where we are selecting based on a third variable that is not being traded off directly. Table 1 shows results from these selected runs to allow for a qualitative comparison between the various methods.

Each method is able to improve the objective used in the fairness score in addition to performing better on the other metrics, and we also find some interesting and unforeseen

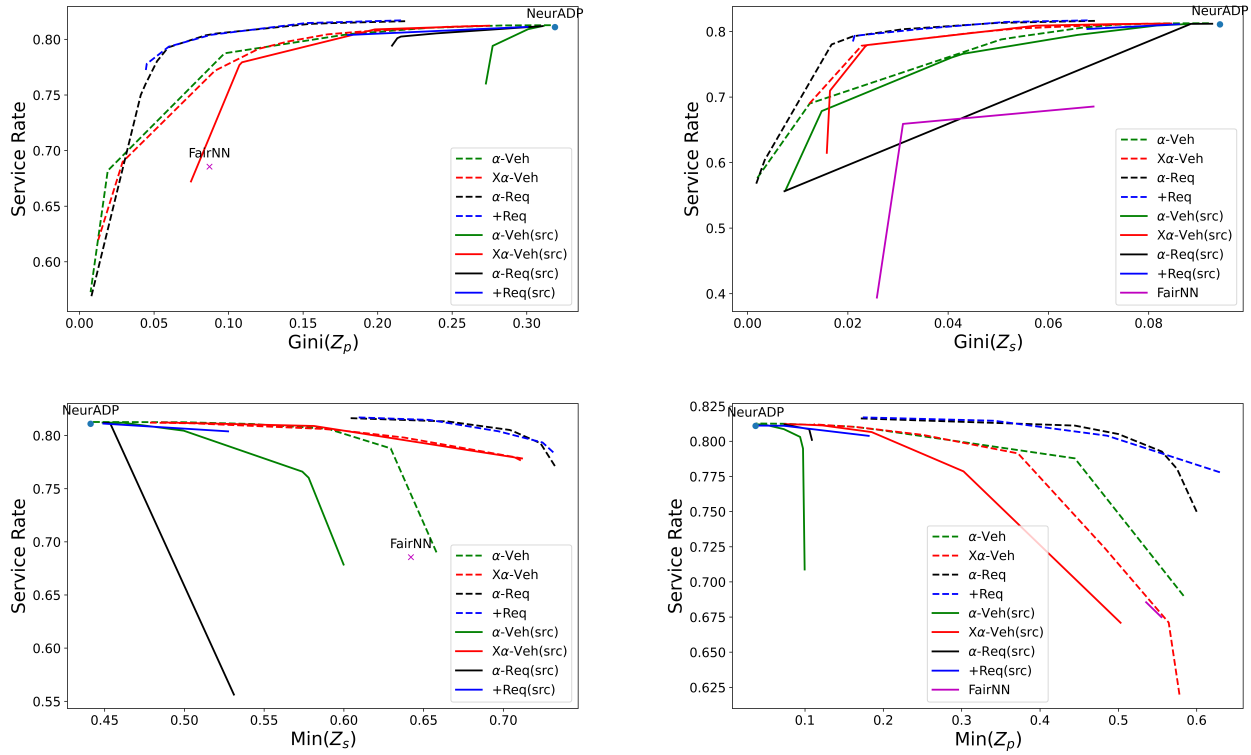

Figure 3: Pareto frontiers for all methods.

behavior. The request-based methods are able to slightly *improve* the overall service rate while also reducing the unfairness for a milder fairness setting. For example, +Req with $\beta = 15$ has an overall service rate of 81.7% compared to 81.1% for base NeurADP, while it has a Gini($\mathbf{Z}_p$) of 0.21 compared to 0.31 for NeurADP. Thus, depending on the need for fairness, it is possible to improve on the metrics without compromising on the efficiency. The last row in Table 1 shows one such example.

Driver-Fairness

While we do consider a fleet of autonomous vehicles, it is possible to instead look at our system as one where human drivers are part of the fleet and follow the central agent's decisions. In such a scenario, it is also of interest to look at how the methods discussed in this paper affect the income distribution among drivers. Figure 4 shows the distribution of the trips each driver served in one day for each of the cases discussed in Table 1. We consider the number of trips made as a proxy for the income, as in our setup, each request is worth the same amount, with the efficiency objective being maximization of requests served. We also assume that drivers work throughout the day for the purposes of this analysis.

The distribution is a tight one for NeurADP, with each driver getting on average 240 requests in the entire day. On the other hand, the distribution is very spread out for FairNN, suggesting that there is a large income disparity among drivers. For our approaches, the vehicle-based fairness approaches show an interesting behavior: We see two distinct clusters of drivers, where drivers of fairness-aware vehicles service significantly fewer requests compared to drivers of regular vehicles, which is to be expected. Therefore, our vehicle-based fairness approaches are unfair for drivers.

However, if the vehicle fleet includes a combination of human drivers and autonomous vehicles, where the fraction of autonomous vehicles is α, then assigning the autonomous vehicles as fairness-aware vehicles and human-driven vehicles as non-fairness-aware vehicles will *increase the number of trips served by the human drivers* compared to if all vehicles are fairness-aware. Alternatively, one can be view this as adding autonomous fairness-aware vehicles to the existing driver population to not only improve the overall service rate (because there are more vehicles serving requests now), but also improve zonal fairness and the income of human drivers! The financial viability of such systems is an open question, and we hope future research is able to answer this question in more detail.

For the request-based fairness methods, the distribution is very close to NeurADP. Using these methods to improve fairness, thus, does not affect the driver population much, beyond the reduction in average trips because of the lowering of the service rate. As shown by the last histogram, this approach can even improve the average trips per driver while keeping a similar income distribution and improving fairness.

Table 1: Comparison of different methods, selecting the hyperparameter value that maximizes min($\mathbf{Z}_p$) (using f_p). The values in bold are the best in their respective columns.

Matching Algorithm	Service Rate (SR)	Min($\mathbf{Z}_s$)	Gini($\mathbf{Z}_s$)	Min($\mathbf{Z}_p$)	Gini($\mathbf{Z}_p$)
NeurADP	0.8119	0.4595	0.0928	0.037	0.3102
FairNN (10^7)	0.6855	0.6421	0.069	0.5355	0.0872
α-Veh (0.5,20)	0.69	0.658	**0.0124**	**0.5833**	**0.0259**
Xα-Veh (0.5,0.5)	0.7782	0.7122	0.0227	0.329	0.102
α-Req (0.2,50)	0.7712	**0.7323**	0.0184	0.5555	0.0534
+Req (15)	0.7841	0.7313	0.0209	0.5416	0.0544
+Req (2)	**0.8146**	0.6536	0.0514	0.3414	0.1520

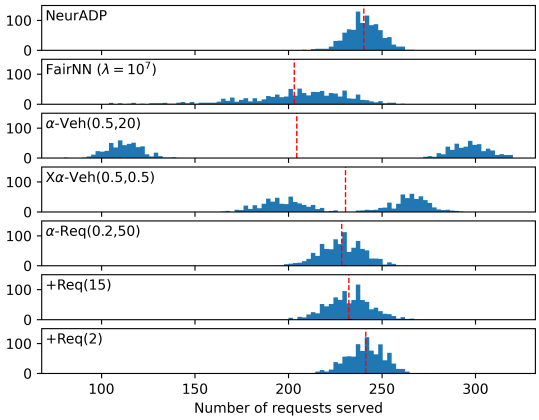

Figure 4: Histograms showing the distribution of trips per driver for each case in Table 1, with the y-axis showing the number of drivers in each bin. The red lines represent the average driver trips

In summary, we see that our simple approaches to improve zonal fairness are very effective across a variety of analyses. This works to further drive home the main idea of the paper: Intent matters. Most approaches are unfair only because they don't try to be fair, but oftentimes it is possible to achieve very similar results while being fairer.

Conclusions

As the demand for cutting edge algorithms for urban mobility increases, their effect on the underlying fairness of these systems needs to be studied, and measures taken to ensure that our algorithms do not inherit the implicit biases that result from pure optimization. In this work, we discussed the issue of zonal fairness in ridesharing systems, introducing new fairness metrics for zone-pair fairness. Through four simple methods that build on existing solutions for order dispatching, we showed how simple techniques can be used to trade off efficiency for fairness. Our methods outperform existing approaches to zonal fairness via an online modification to state-of-art techniques in ridesharing, with the added advantage of not needing any extra training and the ability to be used with any VFA. The tradeoffs proposed in this work can be dynamically adjusted to have increased fairness at minimal cost to efficiency. Our experiments showed that it is better to apply fairness incentives to a subset of the driver or request population for maximum results, as opposed to a blanket approach to fairness.

Our work leaves some interesting avenues open for future work:

- The generalizability of our approach needs to be evaluated. While it can theoretically be applied with any VFA, experiments are needed to verify that. Similarly, this approach needs to be tested with different sub-group fairness measures apart from zonal fairness.
- Our approach is limited by the need to find the right hyperparameter setting for a desired fairness level, but this problem is shared across other solutions for fairness. A potential solution to this problem may involve learning to adapt the hyperparameters to the current state of the world. Such an approach would automatically be able to tune the fairness based on the changes in demand and supply.
- It is of interest to prove theoretical bounds on the expected gains such an approach can provide, given a particular method to improve fairness. We provide general intuition on why we expect our approaches to work, and to explain the results we see. Future work may find mathematical guarantees that confirm our empirical results.

References

Alonso-Mora, J.; Samaranayake, S.; Wallar, A.; Frazzoli, E.; and Rus, D. 2017. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences* 114(3):462–467.

Foti, L.; Lin, J.; and Wolfson, O. 2019. Optimum versus nash-equilibrium in taxi ridesharing. *GeoInformatica*.

Gopalakrishnan, R.; Mukherjee, K.; and Tulabandhula, T. 2016. The costs and benefits of ridesharing: Sequential individual rationality and sequential fairness. *CoRR* abs/1607.07306.

Huang, Y.; Bastani, F.; Jin, R.; and Wang, X. S. 2014. Large scale real-time ridesharing with service guarantee on road networks. *Proceedings of the VLDB Endowment* 7(14):2017–2028.

Lesmana, N. S.; Zhang, X.; and Bei, X. 2019. Balancing efficiency and fairness in on-demand ridesourcing. In Wal-

lach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc. 5309–5319.

Li, M.; Qin, Z.; Jiao, Y.; Yang, Y.; Wang, J.; Wang, C.; Wu, G.; and Ye, J. 2019. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference*, WWW '19, 983–994. New York, NY, USA: Association for Computing Machinery.

Lowalekar, M.; Varakantham, P.; and Jaillet, P. 2019. Zac: A zone path construction approach for effective real-time ridesharing. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, 528–538.

Lowalekar, M.; Varakantham, P.; and Jaillet, P. 2020. Zone path construction (zac) based approaches for effective real-time ridesharing. *arXiv:2009.06051*.

Ma, S.; Zheng, Y.; and Wolfson, O. 2015. Real-time city-scale taxi ridesharing. *IEEE Transactions on Knowledge and Data Engineering* 27(7):1782–1795.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Nanda, V.; Xu, P.; Sankararaman, K. A.; Dickerson, J.; and Srinivasan, A. 2020. Balancing the tradeoff between profit and fairness in rideshare platforms during high-demand hours. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(02):2210–2217.

NYC Taxi & Limousine Commission. 2020. Tlc triprecord data - tlc.

Raman, N.; Shah, S.; and Dickerson, J. P. 2020. Data-driven methods for balancing fairness and efficiency in ride-pooling.

Shah, S.; Lowalekar, M.; and Varakantham, P. 2020. Neural approximate dynamic programming for on-demand ride-pooling. *Proceedings of the AAAI Conference on Artificial Intelligence* 34(01):507–515.

Simonetto, A.; Monteil, J.; and Gambella, C. 2019. Real-time city-scale ridesharing via linear assignment problems. *Transportation Research Part C: Emerging Technologies* 101:208–232.

Sutton, R. S. 1988. Learning to predict by the methods of temporal differences. *Machine learning* 3(1):9–44.

Sühr, T.; Biega, A. J.; Zehlike, M.; Gummadi, K. P.; and Chakraborty, A. 2019. Two-sided fairness for repeated matchings in two-sided markets. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM.

Tong, Y.; Zeng, Y.; Zhou, Z.; Chen, L.; Ye, J.; and Xu, K. 2018. A unified approach to route planning for shared mobility. *Proceedings of the VLDB Endowment* 11(11):1633–1646.

Uber. 2018. Uber matching solution. https://marketplace.uber.com/matching.

Wang, G.; Zhang, Y.; Fang, Z.; Wang, S.; Zhang, F.; and Zhang, D. 2020. Faircharge: A data-driven fairness-aware charging recommendation system for large-scale electric taxi fleets. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4(1):1–25.

Wolfson, O., and Lin, J. 2017. Fairness versus optimality in ridesharing. In *2017 18th IEEE International Conference on Mobile Data Management (MDM)*. IEEE.

Xu, Y., and Xu, P. 2020. Trade the system efficiency for the income equality of drivers in rideshare. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization.

Yu, X., and Shen, S. 2019. An integrated decomposition and approximate dynamic programming approach for on-demand ride pooling. *IEEE Transactions on Intelligent Transportation Systems*.

Zheng, L.; Chen, L.; and Ye, J. 2018. Order dispatch in price-aware ridesharing. *Proceedings of the VLDB Endowment* 11(8):853–865.