

Data-Driven Methods for Balancing Fairness and Efficiency in Ride-Pooling

Naveen Raman¹, Sanket Shah², John P. Dickerson¹

¹University of Maryland

²Harvard University

nraman1@umd.edu, sanketshah@g.harvard.edu, john@cs.umd.edu

Abstract

Rideshare and ride-pooling platforms use artificial intelligence-based matching algorithms to pair riders and drivers. However, these platforms can induce inequality either through an unequal income distribution or disparate treatment of riders. We investigate two methods to reduce forms of inequality in ride-pooling platforms: (1) incorporating fairness constraints into the objective function and (2) redistributing income to drivers to reduce income fluctuation and inequality. To evaluate our solutions, we use the New York City taxi data set. For the first method, we find that optimizing for driver-side fairness outperforms state-of-the-art models on the number of riders serviced, both in the worst-off neighborhood and overall, showing that optimizing for fairness can assist profitability in certain circumstances. For the second method, we explore income redistribution as a way to combat income inequality by having drivers keep an r fraction of their income, and contributing the rest to a redistribution pool. For certain values of r , most drivers earn near their Shapley value, while still incentivizing drivers to maximize value, thereby avoiding the free-rider problem and reducing income variability. The first method can be extended to many definitions of fairness and the second method probably improves fairness without affecting profitability.

1 Introduction

Ride-pooling platforms, such as UberPool and Lyft-Line, manage independent drivers who can service multiple riders concurrently. To match riders and drivers, they use artificial-intelligence-based matching algorithms [Turakhia, 2017]. While matching algorithms typically aim to maximize income, they can inadvertently have negative consequences on fairness, such as a gender wage gap [Cook *et al.*, 2018] or rider discrimination based on race [Brown, 2018].

While the nascent literature on fairness in ride-pooling is limited, prior research has looked at maximizing the minimum utility across riders and drivers by approximating the problem as an instance of the bipartite matching problem [Lesmana *et al.*, 2019]. There have also been past pa-

pers that have looked into the sub-problem of fairness in *rideshare*, where drivers can only service one rider request at a time (such as UberX), which simplifies the matching problem. Past work has used a bipartite matching problem framework to prove bounds on the trade-off between fairness and income [Nanda *et al.*, 2020; Xu and Xu, 2020]. Additionally, past research into rideshare fairness has considered equalizing utility across riders and drivers [Sühr *et al.*, 2019].

Recent work into the ride-pooling matching problem showed that using a Markov decision process (MDP) based approach, in combination with deep learning, maximizes the number of rider requests serviced because it makes non-myopic decisions [Shah *et al.*, 2020]. We build on prior ride-pooling work [Lesmana *et al.*, 2019; Shah *et al.*, 2020] to develop a simple yet robust method to improve fairness non-myopically, which can be generalized to different notions of fairness. Motivated by reports of wage inequality and rider discrimination [Brown, 2018; Cook *et al.*, 2018; Moody *et al.*, 2019], we develop two methods to reduce certain definitions of inequality within the MDP framework: (1) modifying the objective function of the MDP and (2) creating an income redistribution method.¹ Our contributions are the following:

1. We extend an MDP-based framework to non-myopically optimize for different definitions of fairness.
2. We propose new objective functions that aim to maintain profitability while minimizing inequality on the driver and rider-sides, and evaluate their effect on inequality. We show that certain objective functions can reduce inequality while maintaining or even improving profitability.
3. We implement an income redistribution scheme, where each driver contributes a certain percentage of income earned, which is then redistributed to other drivers to offset income fluctuation and reduce wage inequality. We show that, under certain levels of risk tolerance, we can use income redistribution to significantly reduce wage inequality while avoiding the free-rider problem. Additionally, we prove that our income redistribution scheme guarantees drivers a minimum wage.

We find that varying the objective function can positively

¹Our code and data is publicly available at <https://github.com/naveenr414/ijcai-rideshare>

impact rider-side fairness, and utilizing income redistribution can improve driver-side fairness. When used together, these methods can improve both rider and driver-side fairness, and could potentially be used not only for ride-pooling, but also for other matching and resource allocation problems.

2 Related Work

The complexity of the ride-pooling matching problem led to the need for algorithmic solutions. One attempt to solve the problem reduces it to a version of the bipartite matching [Zhao *et al.*, 2019]. Other papers model this problem as an MDP which takes into account future consequences of matching [Lin *et al.*, 2018; Li *et al.*, 2019]. Our work builds on previous work that uses offline-online learning and approximate dynamic programming to match drivers and riders non-myopically [Shah *et al.*, 2020].

While these algorithmic solutions serve more customers than traditional taxi services [Uber, 2015], recent literature has raised questions about the fairness of the matches generated by these algorithms. On the rider side, Brown *et al.* [Brown, 2018] highlight the disparate treatment of riders by ride-pooling companies, which results in higher rates of trip cancellation for black riders. Similarly, on the driver side, some ride-pooling drivers cannot achieve a living wage due to income inequality [Graham, 2017].

One approach to dealing with these issues is to reformulate the problem using bipartite matching and utilize a min-max objective function to maximize the minimum utility for drivers and riders [Lesmana *et al.*, 2019]. Within the sub-problem of rideshare matching, which is simpler because drivers cannot concurrently service multiple riders, past work has proven bounds on the trade-off between fairness and efficiency for a specific notion of fairness [Nanda *et al.*, 2020; Xu and Xu, 2020; Ma and Xu, 2020]. Past empirical research has looked into equalizing utility across drivers and riders [Sühr *et al.*, 2019], and improving the fairness of rideshare demand functions [Yan and Howe, 2020].

Our work builds on past ride-pooling work [Shah *et al.*, 2020; Lesmana *et al.*, 2019] to develop a general method that can be adapted for many definitions of fairness, while also matching non-myopically by using an MDP. We are the first study to look at fairness in MDP-based matching for ride-pooling or rideshare, and so our work applies to the broader class of non-bipartite matching problems. Additionally, we use income redistribution to reduce income fluctuation and inequality without affecting profitability, which has not been explored in previous work.

3 Problem Statement

In this section, we formally describe our problem, methodology, and evaluation strategy.

3.1 Problem Description

We consider the problem of matching a stream of rider requests to one of n drivers. All riders and drivers reside on a graph, which consists of locations, L , and edges, E , where $E_{i,j}$ represents the travel time in minutes between location $i \in L$ and location $j \in L$. A trip between location i and

location j is priced at $E_{i,j} + \delta$, where δ is a constant, capturing both the fixed and variable costs inherent in ride-pooling pricing. Although the stream of rider requests is continuous, we batch requests and match once a minute, which emulates what ride-pooling companies perform in reality [Uber, 2020]. We define rider requests and driver states as follows:

1. We define a rider request, which refers to a rider requesting a ride, to be the tuple $u_i = (g_i, e_i, t_i)$. In the tuple, $g_i, e_i \in L$ are the starting and ending location for the rider request, and t_i is the time when the rider request originated.
2. We define the state of each of the n drivers as $r_i = (m_i, c_i, d_i, p_i, s_i)$, where m_i is the capacity of driver i , and c_i is the number of riders currently driven by driver i . $d_i \in L$ is the location of the driver, p_i is the set of current requests that driver i is servicing, and s_i is the set of previously completed requests, making $|p_i| + |s_i|$ the total number of requests that will be serviced by driver i .

3.2 Matching Riders and Drivers

We describe how we match m rider requests, $U = u_1 \cdots u_m$, with n drivers, $R = r_1 \cdots r_n$, using an MDP framework. Because each driver can be matched to multiple riders, we reduce the number of rider-driver combinations by generating feasible matchings for each driver. We let F^i represent all feasible matchings for driver i , where each matching, $f \in F^i$ is a set of requests that can be served simultaneously by driver i [Alonso-Mora *et al.*, 2017]. We let $a^{i,f}$ denote an indicator value, determining whether the set of requests f is matched to driver i . Each driver is matched to one $f \in F^i$, which could be the empty set, where the driver gets no new rider requests.

The problem reduces to selecting feasible matchings that maximize an objective function, subject to constraints, which we solve through an integer linear program. Let the objective function be $o(R, W)$, which measures the benefit of having the drivers be in the state R , where W is the set of all previous unaccepted and accepted requests. If, after accepting a matching $f \in F^i$, the approximate new state of the drivers is R' , then the change in objective function is

$$\Delta o(R, W, f) = o(R', W \cup f) - o(R, W). \quad (1)$$

We approximate the post-acceptance state, R' , from prior work [Shah *et al.*, 2020]. To avoid matching myopically, we compute the value function, $V(R')$ for the state of drivers after matching. We approximate the value function through deep learning following past work, where the value function is an approximation of the objective function, $o(R, W)$, over a future state, R' , essentially stating the value of a driver residing in a state R' [Shah *et al.*, 2020]. To perform deep learning, we utilize a neural network that takes as inputs, the locations of the drivers and their current path, and uses that to learn the value function. The details of training, network structure, and configuring the deep network can be found in our code, and prior work [Shah *et al.*, 2020]. We then weight each feasible matching as $\Delta o(R, W, f) + \gamma V(R')$, where the discount factor $\gamma = 0.9$. Our aim is to maximize:

$$\sum_{i=1}^n \sum_{f \in F^i} a^{i,f} (\Delta o(R, W, f) + \gamma V(R')) \quad (2)$$

While o could be non-linear, we pre-compute $\Delta o(R, W, f)$ to avoid non-linearities. We use (2) as the objective function in an integer linear program subject to the following constraints:

1. **Driver-side feasibility.** One assigned action per vehicle: $\forall i, \sum_{f \in F^i} a^{i,f} = 1$.
2. **Rider-side feasibility.** At most one assigned action per request: $\forall j, \sum_{i=1}^N \sum_{f \in F^i, u_j \in f} a^{i,f} \leq 1$.

Through the integer linear program (ILP), we compute $a^{i,f}$, which determines which feasible matchings are selected. The ILP is run online using an offline-trained value function.

3.3 Evaluation Strategy

To evaluate our methods, we utilize the New York City taxi data set [New York City, 2016], a commonly used dataset [Lesmana *et al.*, 2019; Shah *et al.*, 2020; Alonso-Mora *et al.*, 2017] which contains pickup and drop-off locations and times for taxi passengers from March 23rd to April 1st and from April 4th to April 8th, 2016. Data from ride-pooling companies are generally proprietary, so we use taxi data to simulate ride-pooling under the assumption that the spatial and temporal distribution of rider requests are similar. The pickup and drop-off location for each rider request is reported in longitude-latitude coordinates; we discretize these into $|L|$ locations in New York City and compute travel time and paths between each pair of locations. The driver locations are initially randomized, but remain consistent between experiments and trials. The data from both experiments (Sections 4 and 5) are available online.²

4 Fairness-based Objective Functions

We develop fairness based objective functions to improve both driver-side and rider-side fairness. In Section 4.1, we discuss profitability metrics that prior work has optimized for. In Sections 4.2 and 4.3, we define our notions of driver and rider-side fairness, and then operationalize each of these by developing two objective functions. In Section 4.4 we discuss how we test each objective function, and in Sections 4.5 and 4.6, we discuss the results after running the experiments.

4.1 Profitability Metrics

We develop two different measures of profitability: the number of riders serviced and the total income accumulated by drivers. To define these two metrics, if the driver states are $R = r_1 \cdots r_n$, where $r_i = (m_i, c_i, d_i, p_i, s_i)$, then the total number of rides serviced by driver i is $|p_i| + |s_i|$, which represents ongoing requests (p_i), and completed requests (s_i). We develop an objective function based on the total number of rider requests serviced by all drivers:

$$o_1(R, W) = \sum_{i=1}^n |p_i| + |s_i|. \quad (3)$$

Similarly, the income for any request $u = (g, e, t)$ is $E_{g,e} + \delta$. So, if we let π_i denote the income for driver i , where

$$\pi_i = \sum_{u=(g,e,t) \in p_i \cup s_i} E_{g,e} + \delta, \quad (4)$$

then our income objective function is the total income:

$$o_2(R, W) = \sum_{i=1}^n \pi_i. \quad (5)$$

The state-of-the-art method [Shah *et al.*, 2020] matches based on maximizing the number of requests.

4.2 Rider-Side Fairness Metrics

We define fairness for both drivers and riders. We note that, as with all operationalizations of societally-relevant concepts such as fairness, the decision of *which* definition of fairness to use—if one is appropriate to use at all—is morally-laden, and one that should be made with the explicit input of stakeholders. Our definitions are drawn in part from recent reports of (primarily rider-side) unfairness in fielded ride-pooling applications [Dillahunt *et al.*, 2017; Brown, 2018; Pandey and Caliskan, 2020]; still, we acknowledge that these need not be one-size-fits-all formalizations of a complicated concept, and view them rather as illustrative examples of a class of fairness definitions that could be incorporated into automated matching algorithms used by ride-pooling platforms.

We base our conception of rider-side fairness on the differential treatment of riders based on membership in a protected group, such as race [Brown, 2018]. Because we lack access to any protected information in our data set, we instead measure fairness based on the distribution of neighborhoods serviced, though the method could be applied to any protected group.

We define a neighborhood as a set of locations in L , and so for any $l \in L$, we define the neighborhood function, $1 \leq N(l) \leq H$, as the function which maps locations to neighborhood labels. The neighborhood function is constructed through the k -means algorithm, which divides the $|L|$ locations into H neighborhoods based on their latitude and longitude. To determine the distribution of neighborhoods serviced, we define h_j as the number of requests serviced starting in neighborhood j , where h_j is

$$h_j = \sum_{i=1}^n \sum_{u=(g,e,t) \in (p_i \cup s_i), N(g)=j} 1, \quad (6)$$

The total number of requests, serviced or non-serviced, originating from neighborhood j is $k_j \geq h_j$ defined as

$$k_j = \sum_{u=(g,e,t) \in W, N(g)=j} 1. \quad (7)$$

Our aim is to equalize treatment across neighborhoods, and so we aim to equalize the percent of requests serviced, $\frac{h_j}{k_j}$, which we call the success rate. To avoid making all success rates 0, we include a profitability term, and regulate the size of the two terms with a hyperparameter λ . This makes our objective function for rider-side fairness:

$$o_3(R, W) = -\lambda \text{Var}\left(\frac{h_j}{k_j}\right) + \sum_{i=1}^n \pi_i, \quad (8)$$

where Var is the variance function. We develop two metrics of rider-side fairness which capture the spread and scale of disparate treatment: $\min\left(\frac{h_j}{k_j}\right)$ and $\text{Var}\left(\frac{h_j}{k_j}\right)$ respectively.

²<https://www.dropbox.com/s/y9pkmzmbjclfrx/test-data.zip?dl=0>

4.3 Driver-Side Fairness Metrics

We develop corresponding objective functions and metrics on the driver-side. Recent news has noted wage discrepancies among drivers [Graham, 2017; Bokányi and Hannák, 2020], so we aim to reduce income disparity. Similar to the rider-side, our objective function minimizes the spread of income while maintaining profitability, which is represented as

$$o_4(R, W) = -\lambda \text{Var}(\pi_i) + \sum_{i=1}^n \pi_i. \quad (9)$$

We measure the scale and spread of driver-side inequality through two metrics. The first measures the scale of income disparity by calculating the minimum income for any driver, which is $\min(\pi_i)$, and the second captures the spread using the variance function, which is $\text{Var}(\pi_i)$

4.4 Experiment Setup

We compare the four objective functions, which we call request, income, rider-side fairness, and driver-side fairness. We utilize each objective function with the matching algorithm, train the corresponding value function, and test each objective function on data from April 4th. The details of hyperparameters and value function training are in Appendix B.

4.5 Experiment Results

We run experiments comparing the different objective functions on both profitability and fairness metrics. We describe the major conclusions from our experiments below:

1. **Alignment of fairness and profitability metrics.** We find that certain fairness metrics are aligned with profitability metrics. In particular, objective functions that improve the overall success rate for riders also improve the success rate in the worst-off neighborhood (Figure 1). Similarly, the objective function which maximizes total income at 200 drivers, the request objective function, also maximizes income for those earning lower wages. However, efforts to decrease the spread of income do not result in higher wages for those at the bottom, and instead lower wages across the board (Figure 2).
2. **Driver-side fairness objective functions improve rider-side metrics.** We find that using an objective function that maximizes driver-side fairness manages to reduce the spread of income and also positively impacts rider-side inequality and profitability. In particular, at 50 drivers, optimizing for driver-side fairness improves the overall success rate and the success rate in the worst-off neighborhood, but at 200 drivers, the state-of-the-art objective function, which optimizes for the number of requests, maximizes both rider-side fairness and profitability metrics. This is probably because, when optimizing for driver-side fairness, drivers tend to service lower value and shorter-distance rides to reduce the spread of income, allowing for more rides to get serviced. However, when there are too many drivers, there are not sufficient short-distance requests.

3. **Inability to raise wages for lowest wage earners.** While objective functions that maximize driver-side fairness reduce the spread of income, they do so at a cost to profitability by lowering income across the board. This is a general problem, as no objective function was able to raise wages for the lowest-earning drivers when compared to the state-of-the-art objective function.

4.6 Discussion

Our experiments show that, by changing the objective function used in matching, it is possible to improve rider-side metrics. Additionally, we find that certain fairness metrics, such as the success rate (fraction of requests serviced) in the worst-off neighborhood, are aligned with profitability metrics. On the other hand, metrics that measure spread, such as the spread of income, are not aligned with profitability metrics. To tackle this issue, we introduce an income redistribution method (section 5) which tackles driver-side inequality without affecting profitability. The results from these experiments can be incorporated into other matching algorithms to improve fairness. Additionally, our methods can be extended to other definitions of fairness by varying the objective function, and so are robust to different definitions of fairness.

5 Income Redistribution

To reduce the spread and fluctuation of income, we propose an income redistribution method, where each driver takes home a certain percentage of their income, and the rest is redistributed at the end of each day. In Section 5.1, we formalize our income redistribution model, and we describe properties of the model in Section 5.2. In Section 5.3, we outline experiments to test out income redistribution method, and we detail the results in Section 5.4.

5.1 Model of Income Redistribution

Driver wages fluctuate day to day due to varying demand, making it difficult for drivers to earn a reliable wage. To counter this and reduce income inequality, we design an income redistribution scheme. To do this, we define the amount that drivers make before redistribution as π_i , $1 \leq i \leq n$, and the amount they make after redistribution as q_i , which is dependent on the risk tolerance, r .

More valuable drivers should have a higher q_i , as they might work longer hours or service higher need areas. To generate an objective measure of the value of a driver, we utilize a metric from game theory known as the Shapley value [Shapley, 1953], which measures the marginal contribution of an agent. The Shapley value helps assess the true contributions of each driver, and was previously used to determine the value of data points [Jia *et al.*, 2019] and decide how much ride-pooling riders should pay [Amano *et al.*, 2020]. In our context, we can formally define the Shapley value, v_i , as the marginal contribution of driver i to each subset of drivers; that is, if we let $D = \{1, 2, \dots, n\}$, and $\pi(s)$ be the income that a subset $s \in D$ would have made under the matching algorithm, for just those s drivers. Then

$$v_i = \sum_{s \subset D \setminus i} \frac{|s|!(n - |s| - 1)!}{n!} (\pi(s \cup i) - \pi(s)). \quad (10)$$

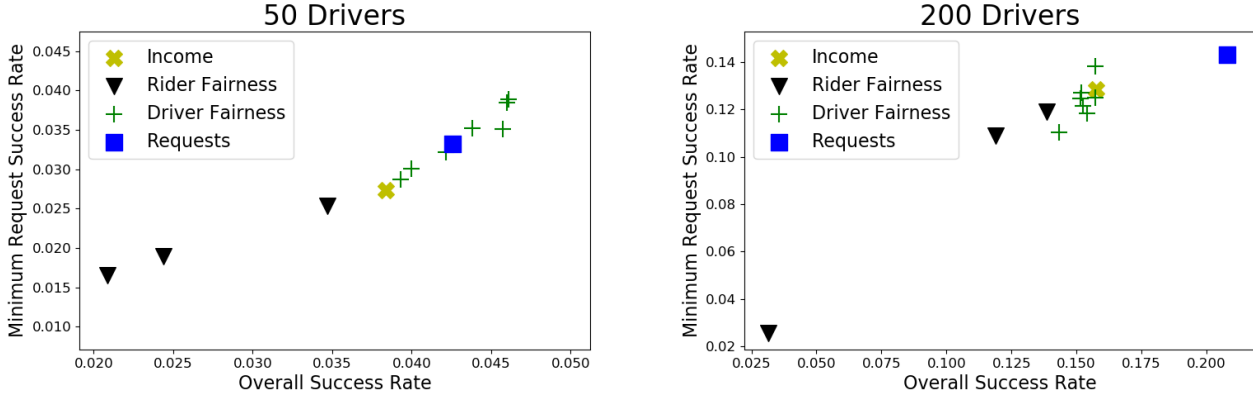


Figure 1: Each point represents one combination of hyperparameter and objective function. We find that with 50 total drivers, the objective function that minimizes the spread of income also achieves the highest request success rate both in the worst-off neighborhood and overall, outperforming the state-of-the-art objective function. However, at 200 drivers, the objective function which maximizes the number of requests serviced achieves the highest success rate overall and in the worst-off neighborhood. The success rates are small because the number of drivers is much smaller than the number of riders; as the number of drivers increases, the success rate will approach 1.

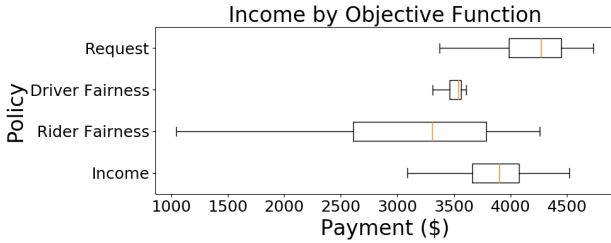


Figure 2: We compare the distribution of income for different objective functions, choosing a fixed value of λ , and set the number of drivers to 200. We find that optimizing for driver-side fairness reduces the income spread, however, at a cost to income for all drivers, as the income distribution shifts downwards. Optimizing for rider-side fairness does no better, as the whole distribution again shifts downwards compared to the state-of-the-art method.

Example 5.1. Consider three drivers, and two riders, with prices \$10 and \$5. Suppose drivers 1 and 2 can service the first request, and drivers 2 and 3 can service the second request, where each driver can service at most one request. This results in the following profits for each subsets of drivers:

Driver Subset	Total Income
$\{\}$	0
$\{1\}$	10
$\{2\}$	10
$\{3\}$	5
$\{1,2\}$	15
$\{1,3\}$	15
$\{2,3\}$	15
$\{1,2,3\}$	15

To compute the Shapley value of driver 1, consider the 4 subsets excluding driver 1: $\{\}$, $\{2\}$, $\{3\}$, $\{2,3\}$, which make 0, 10, 5, and 15 respectively. When adding driver 1 to each of these subsets, the total income increases by 10, 5, 10, and

0 respectively. We then average over these subsets, proportional to the number of permutations for each subset, to get

$$\frac{2!}{3!} \times 10 + \frac{1!}{3!} \times 5 + \frac{1!}{3!} \times 5 + \frac{2!}{3!} \times 0 = 5, \quad (11)$$

which is the Shapley value of the driver.

Due to Shapley value properties, $\sum_{i=1}^n v_i = \sum_{i=1}^n \pi_i$, or total value equals total income.

Calculating the Shapley value requires enumerating all $O(2^n)$ subsets of drivers. To reduce this, we approximate v_i in $O(n)$ evaluations through a Monte Carlo simulation [Ghorbani and Zou, 2019]. Using the Shapley value of a driver, we redistribute income to reduce the difference between a driver's pre-redistribution income, π_i , and their value, v_i . The redistribution amount depends on a risk parameter, $0 \leq r \leq 1$, which designates what fraction of their income is kept by drivers. We collect $\sum_{i=1}^n (1-r)\pi_i$ from all drivers, and redistribute it proportional to the difference between their value and earnings, which is $\max(0, v_i - r\pi_i)$. After redistribution, each driver earns q_i , defined by

$$q_i = rv_i + \frac{\max(0, v_i - r\pi_i)}{\sum_{j=1}^n \max(0, v_j - r\pi_j)} \sum_{j=1}^n (1-r)v_j. \quad (12)$$

This essentially allows drivers to keep some income, and redistributes the rest proportional to the difference between earnings and value, which allows drivers who earned less than their value to recoup some of their losses.

5.2 Redistribution Properties

Our approach to income redistribution begets two attractive theoretical properties.

Theorem 5.1. When π_i , $\sum_{j=1}^n \pi_j$, and r are constant, q_i is maximized when v_i is maximized.

Theorem 5.2. For a given v_i , r , and fixed $\sum_{j=1}^n \pi_j$, the minimum value of q_i is $\min(rv_i, (1-r)v_i)$.

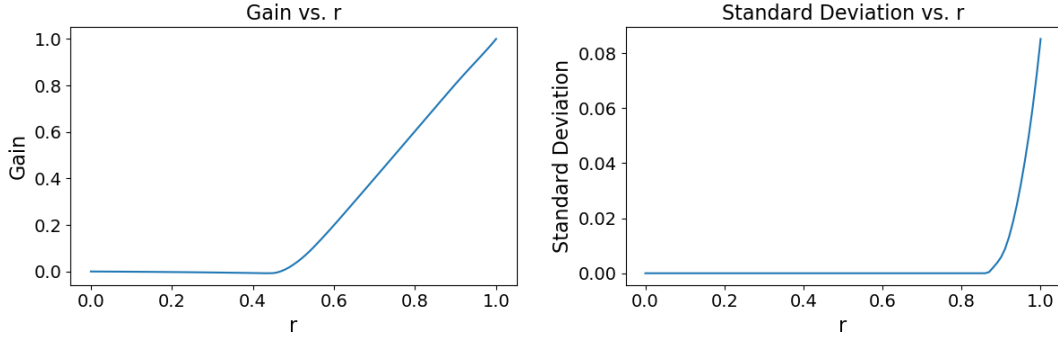


Figure 3: Comparing the gain (*left*) and standard deviation (*right*) of income to value ratio for different values of r . We find that when $0.5 \leq r \leq 0.9$, the gain is non-zero, while the standard deviation is small and non-increasing, meaning that drivers are incentivized to earn more without negatively affecting the income inequality.

Theorem 5.1 incentivizes drivers to maximize their value, meaning that drivers are not better off putting in less effort under income redistribution. Theorem 5.2 gives a guarantee on the minimum amount that drivers can make, which is helpful in light of the recent debates over minimum wage for ride-pooling drivers [Keeton, 2016].

5.3 Experiments

To test the effect of income redistribution, we vary the risk parameter and determine its effect on the income distribution. We run experiments using a variety of objective functions and number of drivers, though we focus on 200 drivers with the requests objective function when describing results. While we previously proved that drivers will aim to maximize v_i , making v_i positively correlated with q_i , we did not specify the magnitude of this correlation. To quantify the correlation, we define the gain metric, g_i , as the ratio of change in q_i to v_i when v_i is doubled. Because higher v_i leads to higher q_i , we know $g_i \geq 0$, and ideally, if v_i for a driver is doubled, then so would q_i , meaning $g_i = 1$. We calculate $g = \frac{1}{n} \sum_{i=1}^n g_i$ as the average gain metric over all drivers.

To determine the fairness of the income redistribution algorithm, we look at the distribution of the ratio of v_i to q_i . Ideally, each driver would earn their value, and so the spread of the distribution would be 0. However, as risk tolerance increases, less money will be redistributed, and so the spread will increase. To quantify this, we compute $\frac{q_i}{v_i}$ for all i , and then compute the standard deviation.

5.4 Experiment Results

We outline some of the findings from our experiments:

1. **Ability to fairly redistribute.** We find that, for $0.5 \leq r \leq 0.9$, we can keep the gain larger than 0, while keeping the spread of income to value ratio near 0 (Figure 3). In practice, this means when $r = 0.9$, which corresponds to $g = 0.8$, drivers receive, on average, an 80% raise in income if they doubled their value. At the same time, the standard deviation of $\frac{q_i}{v_i}$ is low, and so most drivers earn close to their value. This allows for an equitable distribution of income while avoiding the free-rider problem,

where income is spread out despite all the value being concentrated in one driver.

2. **Tightness of Theoretical Bounds.** While the bounds in section 5.2 give results on the minimum amount that drivers are guaranteed to make, in reality, the worst-off drivers end up making significantly more than that. For all values of r , the worst-off driver made at least \$2500, while the theoretical bounds peaked at \$1500. The bounds assume the worst-case situation where one driver gets all the pre-redistribution income, however in practice, this rarely occurs, and so the bounds are not tight.

6 Conclusions & Future Research

With an increasing ubiquity of ride-pooling services comes the need to critically evaluate fairness for both sides of the market – riders and drivers alike. By modifying state-of-the-art ride-pooling matching algorithms, we explored objective functions that increase the number of requests serviced for certain numbers of drivers, and also increased fairness across groups. We additionally proposed addressing income inequality by using income redistribution to allocate income and alleviate fluctuation in income between drivers. By using income redistribution, we found that it is possible to avoid free-riding while keeping income inequality low.

We discuss three potential avenues for future work:

1. By evaluating methods on other data sets, we can test the generalizability of our methods and determine the circumstances under which the methods perform optimally.
2. By developing provable guarantees for objective functions, we increase the robustness of our objective functions and guarantee that these objective functions work under different circumstances.
3. By exploring income redistribution in other resource allocation settings, we can tackle the fairness versus profitability trade-off for other problems.

Acknowledgements

NSF CAREER IIS-1846237, NIST MSE #20126334, DARPA #HR00112020007, DARPA #S4761, and DoD WHS #HQ003420F0035.

Ethical Impact

We address a specific application of the classic fairness-efficiency trade-off found in numerous economic systems. While we did derive our proposed definitions of fairness from reports of inequality in ride-pooling systems [Dillahunt *et al.*, 2017; Brown, 2018; Pandey and Caliskan, 2020], we acknowledge that may not be the proper solution for all settings where ride-pooling is deployed. Indeed, measuring, defining, and incorporating definitions of fairness into automated systems is an area of extremely active research within the FATE community. We confidently make the prescriptive statement that considerations of fairness at *both* the driver and rider level *should* be taken into account in some way in ride-pooling platforms. However, open dialogue with stakeholders—e.g., local governments, riders and drivers hailing from various backgrounds—is imperative to understanding the desires of those stakeholders.

References

- [Alonso-Mora *et al.*, 2017] Javier Alonso-Mora, Samitha Samaranyake, 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.
- [Amano *et al.*, 2020] Yuki Amano, Ayumi Igarashi, Yasushi Kawase, Kazuhisa Makino, and Hirotaka Ono. Fair ride allocation on a line. *arXiv preprint arXiv:2007.08045*, 2020.
- [Bokányi and Hannák, 2020] Eszter Bokányi and Anikó Hannák. Ride-share matching algorithms generate income inequality. *Scientific Reports*, 10(1):1–11, 2020.
- [Brown, 2018] Anne Elizabeth Brown. *Ridehail revolution: Ridehail travel and equity in Los Angeles*. PhD thesis, UCLA, 2018.
- [Cook *et al.*, 2018] Cody Cook, Rebecca Diamond, Jonathan Hall, John A List, and Paul Oyer. The gender earnings gap in the gig economy: Evidence from over a million rideshare drivers. Technical report, National Bureau of Economic Research, 2018.
- [Dillahunt *et al.*, 2017] Tawanna R Dillahunt, Vaishnav Kameswaran, Linfeng Li, and Tanya Rosenblat. Uncovering the values and constraints of real-time ridesharing for low-resource populations. In *Conference on Human Factors in Computing Systems (CHI)*, pages 2757–2769, 2017.
- [Ghorbani and Zou, 2019] Amirata Ghorbani and James Zou. Data Shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning (ICML)*, 2019. Full version: arXiv:1904.02868.
- [Graham, 2017] Mark Graham. *Towards a fairer gig economy*. Meatspace Press, 2017.
- [Jia *et al.*, 2019] Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gurel, Bo Li, Ce Zhang, Dawn Song, and Costas Spanos. Towards efficient data valuation based on the shapley value. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.
- [Keeton, 2016] Richard B Keeton. An Uber dilemma: The conflict between the Seattle rideshare ordinance, the NLRA, and for-hire driver worker classification. *Gonzaga Law Review*, 52:207, 2016.
- [Lesmana *et al.*, 2019] Nixie S Lesmana, Xuan Zhang, and Xiaohui Bei. Balancing efficiency and fairness in on-demand ridesourcing. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5309–5319, 2019.
- [Li *et al.*, 2019] Minne Li, Zhiwei Qin, Yan Jiao, Yaodong Yang, Jun Wang, Chenxi Wang, Guobin Wu, and Jieping Ye. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference*, pages 983–994, 2019.
- [Lin *et al.*, 2018] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In *Conference on Knowledge Discovery & Data Mining (KDD)*, pages 1774–1783. ACM, 2018.
- [Ma and Xu, 2020] Will Ma and Pan Xu. Group-level fairness maximization in online bipartite matching. *arXiv preprint arXiv:2011.13908*, 2020.
- [Moody *et al.*, 2019] Joanna Moody, Scott Middleton, and Jinhua Zhao. Rider-to-rider discriminatory attitudes and ridesharing behavior. *Transportation Research Part F: Traffic Psychology and Behaviour*, 62:258–273, 2019.
- [Nanda *et al.*, 2020] Vedant Nanda, Pan Xu, Karthik Abinav Sankararaman, John P Dickerson, and Aravind Srinivasan. Balancing the tradeoff between profit and fairness in rideshare platforms during high-demand hours. In *Conference on Artificial Intelligence (AAAI)*, 2020.
- [New York City, 2016] New York City. New York Yellow Taxi data, 2016.
- [Pandey and Caliskan, 2020] Akshat Pandey and Aylin Caliskan. Iterative effect-size bias in ridehailing: Measuring social bias in dynamic pricing of 100 million rides. *arXiv preprint arXiv:2006.04599*, 2020.
- [Shah *et al.*, 2020] Sanket Shah, Meghna Lowalekar, and Pradeep Varakantham. Neural approximate dynamic programming for on-demand ride-pooling. In *Conference on Artificial Intelligence (AAAI)*, 2020. Full version: arXiv:1911.08842.
- [Shapley, 1953] Lloyd S Shapley. A value for n -person games. *Contributions to the Theory of Games*, 2(28):307–317, 1953.
- [Sühr *et al.*, 2019] Tom Sühr, Asia J Biega, Meike Zehlike, Krishna P Gummedi, and Abhijnan Chakraborty. Two-sided fairness for repeated matchings in two-sided markets: A case study of a ride-hailing platform. In *Conference on Knowledge Discovery & Data Mining (KDD)*, pages 3082–3092. ACM, 2019.
- [Turakhia, 2017] Chintan Turakhia. Engineering more reliable transportation with machine learning and AI at Uber. <https://eng.uber.com/machine-learning/>, 2017. Accessed: 2021-01-20.
- [Uber, 2015] Uber. Chicago: An Uber case study. https://uber-static.s3.amazonaws.com/web-fresh/legal/Uber_Chicago_CaseStudy.pdf, 2015. Accessed: 2021-01-20.
- [Uber, 2020] Uber. Matching: How does Uber match riders with drivers? URL: <https://marketplace.uber.com/matching>, 2020.
- [Xu and Xu, 2020] Yifan Xu and Pan Xu. Trade the system efficiency for the income equality of drivers in rideshare. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2020.
- [Yan and Howe, 2020] An Yan and Bill Howe. Fairness-aware demand prediction for new mobility. In *Conference on Artificial Intelligence (AAAI)*, volume 34, pages 1079–1087, 2020.
- [Zhao *et al.*, 2019] Boming Zhao, Pan Xu, Yexuan Shi, Yongxin Tong, Zimu Zhou, and Yuxiang Zeng. Preference-aware task assignment in on-demand taxi dispatching: An online stable matching approach. In *Conference on Artificial Intelligence (AAAI)*, volume 33, pages 2245–2252, 2019.

A Matching Details

We generate feasible combinations where the pickup delay is less than 300 seconds and the drop-off delay is less than 60 seconds, which follows from previous work [Alonso-Mora *et al.*, 2017].

B Hyperparameters and Value Function Training

We compare based on both the profitability metrics and the fairness metrics, while varying the total number of riders at full, half, and quarter demand, and the number of drivers at 10, 50, 100, and 200. We test each of the fairness objective functions with a variety of hyperparameters, λ , and set $\delta = 5$, $|L| = 4461$, $m_i = 4$, and $H = 10$. We train using data from the week of March 26th, and we list hyperparameters for each objective function below:

1. **Request:** We run the request objective function by training for 3 days and testing for 1 day.
2. **Driver-side fairness:** We run the driver-side fairness objective function by training for 3 days and testing for 1 day. We use $\lambda = \{0, \frac{1}{6}, \frac{2}{6}, \frac{3}{6}, \frac{4}{6}, \frac{5}{6}, \frac{6}{6}\}$
3. **Rider-side fairness:** We run the rider-side fairness objective function by training for 2 days and testing for 1 day. We use $\lambda = \{10^8, 10^9, 10^{10}\}$
4. **Income:** We run the income objective function by training for 3 days and testing for 1 day.

For both the objective function and redistribution experiments, we used a Red Hat Enterprise 7.9 server with 16 CPUs on an Intel Xeon X5550 processor, with 36 GB of RAM shared across multiple users. Each simulation took no longer than 1 day, and many ran within a couple of hours. For figure 2, we set $\lambda = \frac{4}{6}$ for driver-side fairness, and $\lambda = 10^9$ for rider-side fairness.

C Theoretical Guarantees

In this section, we provide supporting proofs for our theoretical guarantees detailed in the income redistribution section. First, we recall Theorem 5.1, and provide its proof now.

Theorem 5.1. *When π_i , $\sum_{j=1}^n \pi_j$, and r are constant, q_i is maximized when v_i is maximized.*

Proof. For constant π_i and r , this is the same as maximizing

$$rv_i + \frac{\max(0, v_i - r\pi_i)}{\sum_{j=1}^n \max(0, v_j - r\pi_j)} \sum_{j=1}^n (1-r)v_j \quad (13)$$

As v_i increases, rv_i also increases. Similarly, increasing v_i increases $\frac{\min(0, v_i - r\pi_i)}{\sum_{i=1}^n \min(0, v_i - r\pi_i)}$ and brings it closer to 1. Because π_i is constant, then $\sum_{j=1}^n \pi_j = \sum_{j=1}^n v_j$ is also constant. Therefore, maximizing both the first and second terms is done through maximizing v_i , and so in order to maximize q_i , we need to maximize v_i . \square

Next, we recall Theorem 5.2, and provide its proof below.

Theorem 5.2. *For a given v_i , r , and fixed $\sum_{j=1}^n \pi_j$, the minimum value of q_i is $\min(rv_i, (1-r)v_i)$.*

Proof. Consider q_i as a function of π_i . Define $d = \sum_{i=1}^n \max(0, v_i - r\pi_i)$, and $T = \sum_{i=1}^n v_i$, $d \leq T$. We rewrite q_i as

$$rv_i + \frac{\max(0, v_i - r\pi_i)}{d + \max(0, v_i - r\pi_i)} * T * (1-r) \quad (14)$$

This function has no local minima when $0 \leq \pi_i \leq v_i$, and when $\pi_i \geq v_i$, $q_i \geq rv_i$. At $\pi_i = 0$, $q_i = \frac{v_i}{d} \times (1-r)T \geq \frac{v_i}{T} \times (1-r)T = (1-r)v_i$. Therefore, the minimum of q_i is the smaller of these two, which is $\min(rv_i, (1-r)v_i)$. \square

D Additional Experimental Results

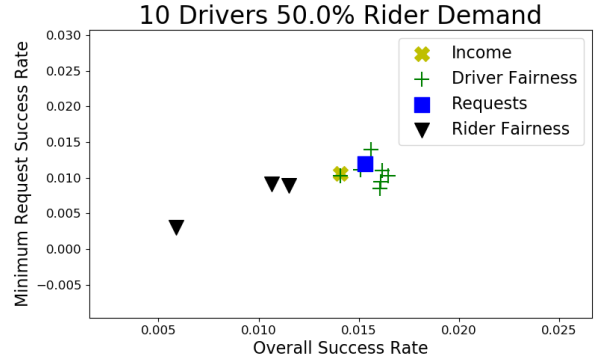


Figure 4: We compare the overall success rate and the success rate in the worst-off neighborhood for 10 drivers when rider demand is 50% of normal. We find that both requests and driver-fairness objective functions perform optimally.

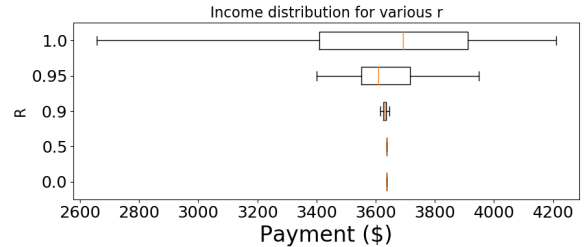


Figure 5: We ran an experiment where v_i was uniform for all drivers, with $n = 200$ using the requests objective function. Using this definition, even for $r = 0.9$, the spread of income is very small, and so is another way to increase earnings for those with lower wages.

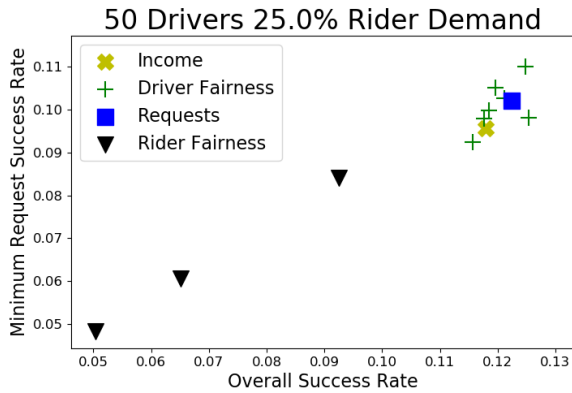


Figure 6: We compare the overall success rate and the success rate in the worst-off neighborhood for 50 drivers when rider demand is 25% of normal. We find that driver-fairness objective functions perform optimally, though the gap between the driver-fairness and requests objective functions tightens, when compared with figure 1. This is because driver-fairness policies perform better when the number of drivers is much larger than the number of riders, and so when the number of riders decreases, the driver-fairness objective function does worse relative to the requests objective function.

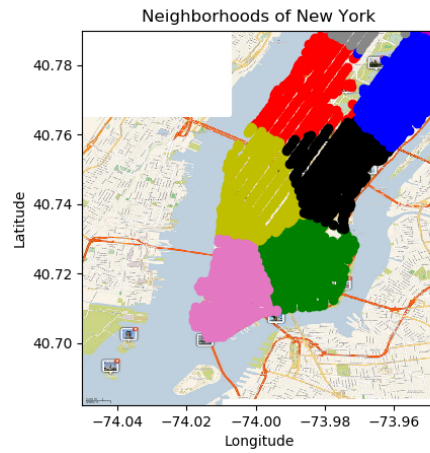


Figure 8: We show a map of New York City, with each neighborhood in a different color (note that all 10 neighborhoods are not shown).

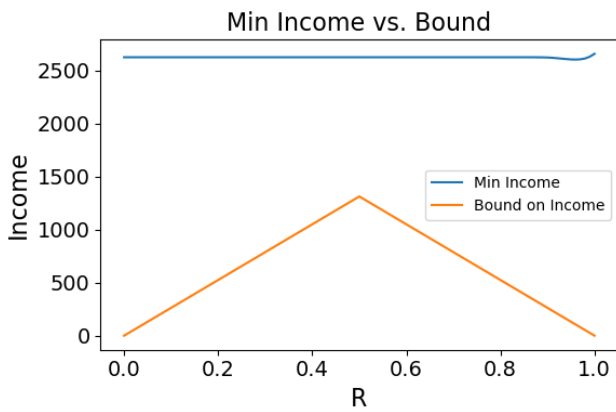


Figure 7: We compare the theoretical bound on the minimum wage against the lowest income for any driver, at $n = 200$ using the requests objective function. We find that the bound is very loose, as the worst-off driver makes nearly double the theoretical bound.