Balancing Fairness and Efficiency in 3D Repeated Matching in Ridesharing

Garima Shakya, Makoto Yokoo

Kyushu University, Fukuoka, Japan
garima@inf.kyushu-u.ac.jp, yokoo@inf.kyushu-u.ac.jp

Abstract

Ride-hailing services' main feature is mediating the assignment and transactions between drivers and passengers. Essentially, they decide on the quality of passengers' experience and the drivers' income distribution. To boost the company's profit, these matching platforms try to maximize the utility for the passengers by optimizing the matching, resulting in shorter waiting times and better service availability. Often, in the process of maximizing revenue, drivers' interests get sidelined. We focus on two objectives: efficiency (minimizing total cost of travel by the vehicles) and fairness (minimizing the maximum traveled distance by any vehicle) for the case when drivers can service at most two passengers simultaneously. We theoretically show the relation between the optimal solutions of both objectives and as the problem is computationally intractable, we propose a heuristic algorithm to achieve an approximately optimal solution. The experimental analysis for the proposed algorithm on real-world data shows that a bit of attention to fairness can bring significantly fair allocation for drivers while not losing much efficiency.

Introduction

Ride-hailing and food-delivery services such as Uber, Lyft, Ola, and Foodora have become essential components in increasing urban transportation's sustainability. These services are quickly changing the urban transportation ecosystem (Hall and Krueger 2018). Due to the flexible working hours, ride-hailing services are a popular alternative for people looking for a side job or a new career.

Ridesharing is a ride-sourcing mode in which a vehicle can simultaneously service more than one request. We investigate the ridesharing problem in the following setting: there are a set of vehicles in a city and a set of requests; each request has a pick-up location and drop-off location; between any two locations, there is a weight function representing the length or cost of the shortest route between them. The problem is to assign all requests to the vehicles such that each vehicle serves at most two requests simultaneously while *maximizing efficiency* (minimizing the sum of the cost of travel over all the vehicles) and/or while *maximizing fairness* (minimizing the maximum traveled distance by any vehicle). For efficiency, we adopt the 'minimizing total cost' criterion, defined as the sum of traveled distance by the vehicles. For fairness, we adopt the 'maximize the utility of least advantaged driver' criterion based on well-recognized Rawlsian egalitarian justice (Rawls 1971).

Motivation

By traveling in shared mode, the total distance traveled by the vehicles can be reduced than that in solo mode. *Sharing the ride* reduces the number of automobiles needed by travelers, which leads to long-term social and economic benefits such as 1) reductions in greenhouse gas emissions and energy consumption, 2) congestion mitigation, 3) reduced parking infrastructure demand, and others. Ridesharing can save fuel and reduce greenhouse gas emissions not only by ridesharing users but also by non-users by reducing the traffic congestion (Shaheen et al. 2018). Here are some case studies showing the environmental benefits of ridesharing in various cities (Caulfield 2009; Yin et al. 2018; Cai et al. 2019; Yan et al. 2020).

Ride-hailing services' main feature is mediating the assignment and transactions between drivers and passengers. Essentially, they decide on the quality of passengers' experience and the drivers' income distribution. To boost the company's profit, these matching platforms try to maximize the utility for the passengers by optimizing the matching that results in shorter waiting time and better service availability (Kedmey 2014). The main reason might be that the passengers contribute more directly to the platform's revenue. In the process of maximizing revenue, drivers' interests get sidelined (Jia, Xu, and Liu 2017). The solutions may result in undesirable social outcomes. Some drivers may be assigned to undesired or insufficient trips, resulting in a loss of fairness from the drivers' perspective (Lesmana, Zhang, and Bei 2019).

Recent studies on two-sided matching platforms raise concerns about the exploitation of employees, including unfair pay, working conditions and safety (Fairwork 2020). The distribution of drivers' income has yet to get much attention in the algorithm design domain. An investigation by Bokányi and Hannák (2020) shows the effect of algorithm design decisions on wage inequality in ride-hailing markets and how small changes in the system parameters can cause large deviations in the income distributions of identically performing drivers. Our concern is that the shortterm income differences may result in enforced and long-term wage gaps. Nonetheless, ensuring *fair income distribution* for drivers might also prove beneficial in sustaining the business in the long run. Otherwise, unsatisfied drivers may leave or remain inactive on the platform. Therefore, fair income distribution on the drivers' side should receive more attention.

The ridesharing platforms make request-to-vehicle assignments with a repeated set of vehicles and customers where the assignment for each individual is often for some limited duration in the day. Once that duration is over, the vehicle can be assigned to another available request in the following duration. In line with the current matching scenarios, we are specifically interested in investigating the repeated matching between drivers and requests that lead to a fair distribution of drivers' income at the end of the day while maintaining efficiency.

Related work

Ridesharing systems are widely studied in the literature. Finding an efficient ridesharing allocation is based on the 2-1 assignment problem reviewed by Goossens et al. (2012). The 2-1 assignment problem is as follows: given a set W of m white balls and a set B of $2m$ black balls; there is a cost function for every triple combination containing two balls from B and one ball from W; the objective is to find a collection of triple combinations such that the sum of costs of triple combinations is minimum, while each ball is in precisely one combination. In terms of the ridesharing problem, W and B can be considered as the set of vehicles and passengers' requests, respectively. Goossens et al. (2012) provide a $4/3$ approximation ratio algorithm for 2-1 matching. However, the expression for the cost function in (2012) differs from that in ridesharing settings.

Bei and Zhang (2018) studied the *maximizing efficiency* problem in ridesharing and proved that finding the most efficient 2-1 matching is NP-hard, and proposed a polynomial time 2.5 approximation ratio algorithm. Luo and Spieksma (2020) investigated the problem of *minimizing total latency* along with efficiency and provided two algorithms with approximation ratios 2 and $5/3$ for efficiency and total latency, respectively.

The idea of loss in efficiency in achieving a fair solution is well explored in many resource allocation settings. However, to the best of our knowledge, the problem needs to be more explored in the request-to-vehicle assignment domain. Mainly because the existing literature for other resource allocation problems can not be easily applied to the ridesharing settings. This is due to additional constraints unique to the ridesharing problem, such as the constraint of pick-up distances.

In line with fairness on ride-hailing platforms, Nanda et al. (2020) examined the rider's fairness due to drivers' discriminative cancellations. Xu and Xu (2020) construct a bi-objective linear program focused on profit and fairness on the platform and propose two LP-based parameterized online algorithms. Our objectives also differ from the literature mentioned above and traditional resource allocation settings as we study the problem of achieving a balance of fairness and efficiency in *3-dimensional* request-to-vehicle matching where two requests can be serviced by a vehicle simultaneously. The addition of one more dimension brings more challenges.

An exciting and closely related work by Lesmana et al. (2019) provides a reassignment algorithm to find a balance between the two objectives while finding one-to-one (2-D) request-to-vehicle matching. The algorithm in (Lesmana, Zhang, and Bei 2019) can also be applied to shared ride settings by assigning one passenger in each repetition of the algorithm. Repeating the algorithm provides a greedy solution by looking for the best solution in each repetition. We are designing an algorithm that finds a match for the combined input of two repeats for the algorithm in (Lesmana, Zhang, and Bei 2019). This allows the solution set concerning the capacity of the vehicles and opens up the opportunity to find a better solution. Along with the importance of finding the optimal solution for ridesharing, we also intend to look for the algorithmic question of finding the balanced solution for multiple objectives in 3-D matching.

Our contributions

In summary, this paper focuses on following research questions:

1. How the efficient and fair solutions for 2-1 matching, in ridesharing, are related?
2. How can we get a balance between efficiency and fairness in ridesharing?
3. What is the price, in terms of efficiency, for fair distribution of work and income among drivers?

To answer these questions, this paper contains the following:

1. We provide theoretical analysis for bound on the 'loss in efficiency' while achieving a fair solution, in terms of fairness (Theorems 1 and 2).
2. We propose a two-phase algorithm that account for the natural tension between these two objectives (Algorithm 1).
3. We experiment on a real world dataset and attempt to answer the question 3 and analyse the performance of the proposed algorithm (Figure 4).

Model and problem formulation

Consider the city as a weighted connected graph, $GC = (V, E, w)$, where V is the vertex set, $\mathcal{L}$ is a finite set of locations, E is the edge set s.t., $E = \{(l_1, l_2) | l_1, l_2 \in \mathcal{L}\}$ and $w : E \rightarrow \mathbb{R}^+$ is the edge weight function. The weight function w can be any metric satisfying (1) non-negativity $w(l_x, l_y) \geq 0$, (2) symmetry $w(l_x, l_y) = w(l_y, l_x)$. Typically, w can be considered as the distance function ℓ_2, ℓ_1, or distance on a road graph. The weight notation is extended for the path as well. With slight abuse of notations, the weight of a path $(l_1, l_2, \ldots, l_x)$ is defined as $w(l_1, l_2, \ldots, l_x) = \sum_{i=1}^{x-1} w(l_i, l_{i+1})$.

Denote the daily working hours of the ridesharing platform as T. The platform accumulates the requests entered during an accumulating time window λ and performs batch assignment of the accumulated requests to the available drivers. Let $(t_1, t_2, t_3, \dots, t_i, t_{i+1}, \dots, t_{|T|})$ be the sequence of time instants, such that $t_{i+1} - t_i = \lambda$. The set of accumulated requests at t_i are represented by $\mathcal{R}^{t_i} = \{r_1, r_2, r_3, \dots, r_m\}$. Each ride r_k consists of two elements, $r_k = (s_k, d_k)$, where, s_k and d_k denotes the source and destination of the request r_k, respectively.

The set of available drivers[1] at t_i are represented as $\mathcal{D}^{t_i} = \{v_1, v_2, v_3, \dots, v_n\}$. The attributes of each vehicle v_k is represented as a 6-tuple $(CL_{v_k}, DT_{v_k}, WD_{v_k}, TL_{v_k}, RC_{v_k}, CAP_{v_k})$ where, $CL_{v_k} \in L$ is the current location of v_k, $DT_{v_k} \in \mathbb{R}_{\geq 0}$ is total distance travelled by v_k since the beginning of the day, $WD_{v_k} \in \mathbb{Z}_{\geq 0}$ is the number of t_is the driver is been active on the platform, $TL_{v_k} \in \mathbb{Z}_{\geq 0}$ is time left to complete the ride currently being serviced by v_k, RC_{v_k} is the total number of requests serviced by v_k since the beginning of the day, $CAP_{v_k} \in \mathbb{Z}_{>0}$ is the seating capacity of v_k, which is the maximum number of passengers that can be serviced simultaneously by v_k. In the following part of the paper, we represent a time instance as t without mentioning the subscript i unless required.

We study the algorithmic question of allocating at most two requests to each vehicle. Hence, in this paper, we assume $CAP_{v_k} = 2$, $\forall v_k \in \mathcal{D}^t$. Let $\mathcal{M}$ denotes the set of all possible 2-to-1 matching for sets $\mathcal{R}^t$ and $\mathcal{D}^t$. More precisely, our task is to find a matching $M^t \in \mathcal{M}$ such that, $M^t = \{(v_k, \mathcal{R}_k) | v_k \in \mathcal{D}^t, \mathcal{R}_k \subseteq \mathcal{R}^t, |\mathcal{R}_k| \leq 2\}$, where, $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \dots, \mathcal{R}_n$ are mutually disjoint subset of $\mathcal{R}^t$. Each element $(v_k, \mathcal{R}_k)$ of M^t denotes that the vehicle v_k is assigned to a pair of requests $\mathcal{R}_k = (r_i, r_j)$.

We define a function $cost : M \rightarrow \mathbb{R}^+$ s.t., $cost(v_k, \mathcal{R}_k)$ is the minimum distance needs to be covered by v_k to service $\mathcal{R}_k$. There can be six possible routes, depending on the order of the pick-up and drop-off of the passengers in $\mathcal{R}^k$. More formally,

$$\begin{aligned} cost(v_k, \mathcal{R}_k) = \min(& w(CL_{v_k}, s_i, d_i, s_j, d_j), \\ & w(CL_{v_k}, s_i, s_j, d_i, d_j), \\ & w(CL_{v_k}, s_i, s_j, d_j, d_i), \\ & w(CL_{v_k}, s_j, s_i, d_j, d_i), \\ & w(CL_{v_k}, s_j, s_i, d_i, d_j), \\ & w(CL_{v_k}, s_j, d_j, s_i, d_i)) \end{aligned}$$

If $|\mathcal{R}_k| = 1$ then, $cost(v_k, \mathcal{R}_k) = w(CL_{v_k}, s_i) + w(s_i, d_i)$. The cost of a matching M is defined as,

$$cost(M) = \sum_{d_k \in \mathcal{D}} cost(v_k, \mathcal{R}_k)$$

Driver's per day income: The payment scheme to the driver might differ between service providers and cities. Sometimes, even the same service provider might rapidly change the pricing scheme within a city (Diakopoulos 2015). We omit those calculations and assume that the drivers get a fixed pre-decided daily wage. As there is some cost of travel between any two locations, such as fuel cost, the daily utility of a driver depends on the distance traveled on that day. In other words, the utility of a driver depends on the shortest distance covered by the vehicle to serve the assigned ride. The utility is more if the cost of the ride is less. Concerning a matching M, the utility of a driver v_k is defined as:

$$u_{d_k}(M) = -\kappa \, cost(v_k, M(k))$$

where κ is a proportionality constant that depends on the parameters, such as the fuel cost.

As most of the definitions and results are same for each interval $t \in T$, we write the notations without using the superscript t in the following part of the paper. We also represent the matching M among the rides and vehicles as a 3 dimensional matrix $A := [a_{i,j,k}]_{\forall (i,j) \in \mathcal{R} \times \mathcal{R}, \forall k \in \mathcal{D}}$ such that, $a_{i,j,k} = 1$ if $M(v_k) = \mathcal{R}_k$ and $\mathcal{R}_k = (r_i, r_j)$, otherwise, $a_{i,j,k} = 0$.

Desired properties

We focus on multiple objectives defined as follows.

Definition **1** (Maximum matching)**.** *Given $(\mathcal{R}, \mathcal{D})$, a matching $M \in \mathcal{M}$ is maximum matching if,*

$$M = \max \sum_{k \in \mathcal{D}} \sum_{(i,j) \in \mathcal{R} \times \mathcal{R}} a_{i,j,k}$$

In other words, given $(\mathcal{R}, \mathcal{D})$, a maximum matching M consists of a maximum number of 2-to-1 assignments among $\mathcal{R}$ and $\mathcal{D}$. This objective ensures that we assign a maximum number of requests to the drivers, which reduces the overall waiting time of the passengers.

Definition **2** (Efficient matching)**.** *Given $(\mathcal{R}, \mathcal{D})$, a matching $M \in \mathcal{M}$ is efficient matching if there does not exists a matching $M' \neq M$ such that,*

$$\sum_{v_k \in \mathcal{D}} cost(v_k, M'(k)) < \sum_{v_k \in \mathcal{D}} cost(v_k, M(k))$$

We desire to reduce the total traveled distance by vehicles that reduce the total emission of greenhouse gases and overall traffic congestion. Our objective is to get an efficient matching and hence to find an $M \in \mathcal{M}$ that has a minimum cost.

Based on the difference principle in the seminal work on the theory of justice by John Rawls in (Rawls 1971), we defined the *unfairness* (UF) of a matching M as,

$$UF(M) = \max_{v_k \in \mathcal{D}} cost(v_k, M(k))$$

Definition **3** (Fair matching)**.** *Given $(\mathcal{R}, \mathcal{D})$, a matching $M \in \mathcal{M}$ is a fair if there does not exist a matching $M' \neq M$ such that,*

$$\max_{v_k \in \mathcal{D}} cost(v_k, M'(k)) < \max_{v_k \in \mathcal{D}} cost(v_k, M(k))$$

We aim to maximize fairness and hence to find an M that minimizes the distance traveled by the driver who has traveled the most. In other words, we aim to *maximize the utility of the least advantaged agent.*

[1]We use the term 'driver' to denote a 'vehicle.' And, therefore, use both terms interchangeably.

Aiming for the *fairness* notion defined above may provide an unfair advantage to drivers with less active duration. The objective to increase fairness will imply allocating the least costed requests to the driver who traveled the most irrespective of their operational hours. To handle this issue, we also follow a fairness notion called 'proportional equality' (Sühr et al. 2019), which relies on the idea that, over time, the drivers should receive benefits proportional to the duration they remain active on the platform.

DEFINITION **4** (Proportionally equal matching)**.** *Given $(\mathcal{R}, \mathcal{D})$, a matching $M^* \in \mathcal{M}$ is called a proportionally equal if there does not exist a matching $M' \neq M$,*

$$\max_{v_k \in \mathcal{D}} \frac{DT_{v_k}(M')}{WD_{v_k}(M')} < \max_{v_k \in \mathcal{D}} \frac{DT_{v_k}(M^*)}{WD_{v_k}(M^*)}$$

where, $DT_{v_k}(M)$ and $WD_{v_k}(M)$ are the distance travelled by and working duration of v_k, after the implementation of M.

We aim to maximize proportional equality and find an M that minimizes the maximum distance traveled by any driver per unit of working duration.

We summarize our objectives as follows:

DEFINITION **5** (OBJ1: Efficient Maximum Matching)**.** *Given an instance $(\mathcal{R}, \mathcal{D})$, find a maximum matching, s.t., $\sum_{v_k \in \mathcal{D}} cost(v_k, M(k))$ is minimum. Mathematically,*

$$\begin{aligned}
&\underset{M \in \mathcal{M}}{\text{argmin}} \sum_{v_k \in \mathcal{D}} cost(v_k, M(k)) \\
&s.t. \sum_{r_j \in \mathcal{R}, v_k \in \mathcal{D}} a_{i,j,k} \leq 1, \quad \forall r_i \in \mathcal{R} \\
&\sum_{r_i \in \mathcal{R}, v_k \in \mathcal{D}} a_{i,j,k} \leq 1, \quad \forall r_j \in \mathcal{R} \\
&\sum_{(r_i, r_j) \in \mathcal{R} \times \mathcal{R}} a_{i,j,k} \leq 1, \quad \forall v_k \in \mathcal{D} \\
&a_{i,j,k} \in \{0, 1\} \quad \forall r_i, r_j \in \mathcal{R}, \forall v_k \in \mathcal{D}
\end{aligned}$$

DEFINITION **6** (OBJ2: Fair Maximum Matching)**.** *Given an instance $(\mathcal{R}, \mathcal{D})$, find the fair maximum matching, s.t., $\max_{v_k \in \mathcal{D}} DT_{v_k}$ is minimum. Mathematically,*

$$\underset{M \in \mathcal{M}}{\text{argmin}} \max_{v_k \in \mathcal{D}} DT_{v_k}$$

such that the sets of feasibility constraints defined in Definition 5 satisfy.

Relation between efficient and fair solutions

Any feasible solution of $OBJ1$ is also a feasible solution of $OBJ2$ and vice versa. However, they may not have common optimal solutions, as shown in the Example 1.

EXAMPLE **1.** *Consider two vehicles $\mathcal{D} = \{v_1, v_2\}$ with $CAP_{v_1} = CAP_{v_2} = 2$, $WD_{v_1} = WD_{v_2}$ and located at $A \in \mathcal{L}$. There are four requests $\mathcal{R} = \{r_1, r_2, r_3, r_4\}$ such that, $\forall r_i \in \mathcal{R}$, $s_i = A$. And the destination locations for r_1, r_2, r_3, and r_4 are B, C, D and E, respectively (Figure 1). The edges and weights in fig. 1 show the connectivity and the cost of travel between any two locations. One of the optimal solution for $OBJ1$ is $M_1^* = \{(v_1, (r_1, r_2)), (v_2, (r_3, r_4))\}$ with $OBJ1(M_1^*) = 3.5\epsilon + 8.5\epsilon = 12\epsilon$. However, M_1^* is not an optimal solution for $OBJ2$ as $OBJ2(M_1^*) = 8.5\epsilon$ and there exist another matching $M_2^* = \{(v_1, (r_1, r_3)), (v_2, (r_2, r_4))\}$ with increased fairness; $OBJ2(M_2^*) = 6.5\epsilon$, but also increased cost, $OBJ1(M_2^*) = 13\epsilon$.*

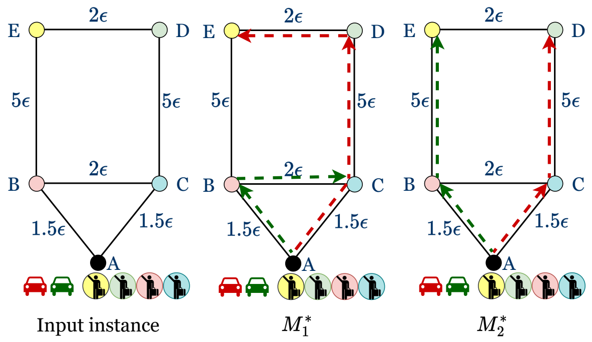


Figure 1: Example 1 with $\mathcal{D} = 2$ and $\mathcal{R} = 4$.

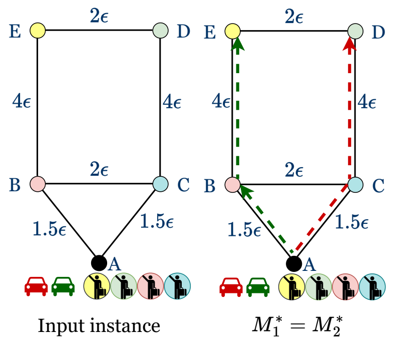


Figure 2: Example 2 with $\mathcal{D} = 2$ and $\mathcal{R} = 4$.

Represent the loss in efficiency while achieving a fair solution as Δ. Mathematically,

$$\Delta = OBJ1(M_2^*) - OBJ1(M_1^*)$$

Similarly, define the loss in fairness while achieving an efficient solution as Γ. Mathematically,

$$\Gamma = OBJ2(M_1^*) - OBJ2(M_2^*)$$

The lower bound for Δ and Γ is 0, as there exist problem instance where the efficient matching is also fair, as shown in the example below.

EXAMPLE **2.** *Consider the example 1 with a change in the edge weight as shown in Figure 2. One of the optimal solution for $OBJ1$ is $M_1^* = \{(v_1, (r_1, r_3)), (v_2, (r_2, r_4))\}$ with $OBJ1(M_1^*) = 11\epsilon$. Nevertheless, M_1^* is an optimal solution for $OBJ2$ as well with $OBJ2(M_1^*) = 5.5\epsilon$ and there exist no another maximum matching M_2' with increased fairness.*

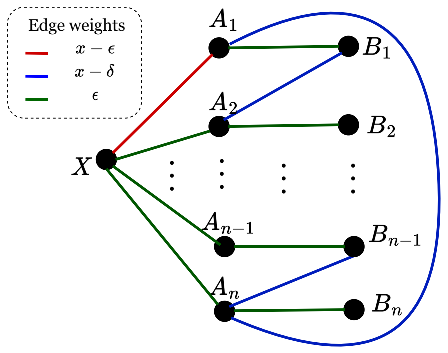


Figure 3: Problem instance for Theorem 2

THEOREM **1.** *If M_1^* and M_2^* are optimal solutions for $OBJ1$ and $OBJ2$, respectively, then* $\Delta = \mathcal{O}(nOBJ2(M_1^*))$.

Proof. Suppose the above statement is not true and there exist a scenario where $(OBJ1(M_2^*) - OBJ1(M_1^*)) > n(OBJ2(M_1^*)$. As $cost(.)$ is always non-negative, we get, $OBJ1(M_2^*) > n(OBJ2(M_1^*)$. This implies, $nOBJ2(M_2^*) > n(OBJ2(M_1^*)$, which leads to contradiction that M_2^* is an optimal solution for $OBJ2$. □

Next, we show a lower upper bound for Δ in terms of $OBJ2(M_1^*)$.

THEOREM **2.** *If M_1^* and M_2^* are optimal solutions for $OBJ1$ and $OBJ2$, respectively, then there exists an input instance, where $\Delta = o((n-1)OBJ2(M_1^*))$.*

Proof. We construct an input instance to prove the above statement. Consider n drivers $d_1, d_2, \ldots, d_n$ located at X, and $2n$ ride requests each with source location X. The destination location of half of the rides, say r_1 to r_n are $A_1, A_2, \ldots, A_n$ and for rides $r_{n+1}, r_{n+2}, \ldots, r_{2n}$ are $B_1, B_2, \ldots, B_n$, respectively. All these locations and their cost of travel between them is shown in Figure 3. The notations in edge weights are as follows: $x, \epsilon, \delta > 0$, $\epsilon < \delta$ and $\epsilon + \delta < x$. The most expensive edge is (X, A_1) with cost $x - \epsilon$.

One of the efficient maximum matching M_1^* is $\{(d_1, (r_1, r_{n+1})), (d_i, (r_i, r_{n+i}))$ for $1 < i \leq n\}$ with $OBJ1(M_1^*) = x + 2\epsilon(n-1)$. The matching M_1^* has $OBJ2(M_1^*) = x$. However, the efficiency is decreased in M_2^* as the $OBJ1(M_2^*) = n(x - \delta + \epsilon) + 2\epsilon$.

The difference in the efficiency in the two matching Δ

$$= n(x - \delta + \epsilon) + 2\epsilon - (x + 2\epsilon(n-1))$$
$$= (n-1)x + n(\epsilon - 2\epsilon) + 2\epsilon - \delta$$

with $\epsilon, \delta \ll x$,

$$= o(n-1)x$$

□

Remarks: The proof for Theorem 2 is valid when we consider *proportional equality* as $OBJ2$ instead of *fairness*. We can fix WD_{v_k} in Example 2 as equal for all $v_k \in \mathcal{D}^t$, and the same example will prove the statement of Theorem 2 for proportional equality as $OBJ2$.

Algorithm 1: **`Efficient-match Fair-assign Algorithm`** **(EMFAA)**

```
1: Procedure ($\mathcal{L}, \mathcal{R}^t, \mathcal{D}^t, t$)
2: Construct a weighted graph $G_1 := (\mathcal{R}^t, E^1, W^1)$ such that, $E^1 := \{(r_i, r_j) | \forall r_i, r_j \in \mathcal{R}^t, r_i \neq r_j\}$, $W^1(r_i, r_j) := \min\left(cost(s_i, (r_i, r_j)), cost(s_j, (r_i, r_j))\right) \ \forall (r_i, r_j) \in E^1$.
3: Find minimum weight matching $M^1$ for $G_1$.
4: $SM^1$ := Sorted request pairs in $M^1$ in increasing order of their cost.
5: $SD^t$ := Sorted $\mathcal{D}^t$ in decreasing order of $DT_{v_k}$.
6: for $i$ from 1 to $\min(|SM^1|, |SD^t|)$ do
       $M^2(SD^t(i)) = SM^1(i)$ // Find a matching $M^2$ by assigning the least weighted $(r_i, r_j)$ to the vehicle $v_k$ with the maximum value of $DT_{v_k}$, and so on. //
7: end for
8: return $M^2$ as the batch assignment for $t$.
```

Proposed algorithm

Finding the efficient maximum matching is equivalent to finding a 3-dimensional perfect matching (3DM), known to be NP-hard (Garey and Johnson 1990; Bei and Zhang 2018). We believe that finding an optimal solution for $OBJ2$ is also computationally intractable as the objective is similar to finding the optimal solution of minimum makespan scheduling, which is known to be NP-hard (Garey and Johnson 1990), with an additional constraint. The additional constraint is over the maximum number of jobs that can be allocated to a machine, where the processing time of a job on a machine depends also on the other jobs assigned to that machine.

We are looking for a computationally inexpensive algorithm that can provide close-to-optimal solutions. There are two major decisions to be made to achieve a 2-to-1 matching. First, to find the request pairs; second, to assign a driver to each request pair. To find a balance between the two objectives, we propose a two-phase heuristic algorithm (Algorithm 1) that focuses on the objectives individually. The main idea is to prioritize one of the objectives while making the first decision and another objective while making the second decision. The algorithm resembles the bi-level optimization where we partially optimize $OBJ1$ and $OBJ2$ at the lower and upper levels, respectively.

Algorithm 1 finds the assignment at the end of each accumulated window. At each time interval t, the algorithm goes in two phases: in the first phase, the algorithm matches the requests based on their combined travel cost, and in the second phase, the algorithm assigns the matched request pairs with the available drivers. The aim is to focus on efficiency in the first phase while matching the requests and fairness in the second phase while assigning the matched pairs to the drivers.

Remarks: (a) A minimum matching in a weighted graph of m vertices can be computed in time $O(m^3)$ (Gabow 1990). Therefore, the Algorithm 1 runs in $O(m^3)$ time.
(b) The step (6) results in the maximum possible assign-

ments by considering the number of available drivers and the number of request pairs found in step (3). Hence, the Algorithm 1 finds a *maximum* matching from the set of feasible matching.

More versions of EMFAA

We can change steps (5) and (6) of Algorithm 1 depending on various factors in the settings, such as the definition of fair allocation for drivers. Some examples are discussed here.

1. **According to the definition of fairness and payment scheme:**
   - ▷ For considering *proportional equality* instead of *fairness* as $OBJ2$, the sorting in step 5 is done based on $\frac{DT_{v_k}}{WD_{v_k}}$ instead of DT_{v_k}.
   - ▷ For the case when drivers get a fixed payment per request serviced by them. Then, the measure of proportional inequality for fixed payment (PEF) is

   $$PEF^t(M) = \max_{v_k \in \mathcal{D}} \frac{RC_{v_k}}{WD_{v_k}}$$

   where, RC_{v_k} is the total number of requests serviced by v_k since the beginning of the day. In that case, $OBJ2$ is to minimize PEF, therefore, sorting in step (5) is done based on the $\frac{RC_{v_k}}{WD_{v_k}}$ of drivers and step (6) changes accordingly.
2. **Flexible assignment:** In algorithm 1, we assume that all the vehicles available in $\mathcal{D}^t$ are empty and can be assigned to two rides. However, in reality, a vehicle v_k might be in the state where it is currently servicing only one passenger and has space for one more passenger. Hence, assigning a new passenger does not violate the capacity constraint. In such a case, we propose the following.

   Before constructing the weighted graph G_1, add the ride r_i, which is currently being serviced by v_k to set $\mathcal{R}^t$ with source s_i as the current location of the vehicle, and destination d_i as the original destination of r_i. Do not include v_k in $\mathcal{D}^t$ for steps (5-6). Assign v_k to the ride r_j matched with r_i in M_1, i.e., $M_2(v_k) = \{r_i, r_j\}$, where $M_1(r_i) = r_j$.

Experimental analysis

We experiment on a real-world dataset to analyze the behavior of Algorithm 1 concerning fairness and efficiency.

Framework

The dataset details and values of the parameters used for the experiments are as follows.

Trip Dataset: We use the publicly available dataset of taxi trips (solo rides) in Chicago city (Chicago Data Portal 2022). The dataset contains all the taxi trips, starting January 2013, reported to the City of Chicago. We extracted the dataset during the busy three hours in the morning (8 : 00-10 : 00) on 'April 4, 2022' (Monday). The dataset includes a unique identifier for each taxi and some essential attributes about trips, e.g., the source and destination locations and time of the trip. We choose this day arbitrarily, and we strongly believe the results will have a similar pattern if we experiment on another part of the dataset.

Accumulating window (λ): In the dataset, the trip start and end time are rounded to the nearest 15-minute interval to preserve the passengers' privacy. We consider λ=15 minutes and divide all the trips into 15-minute intervals depending on their start time. We randomly pick 20 requests in each 15-minute interval.

Drivers: The dataset does not contain information for the location of vehicles before the trips are assigned to them. Therefore, the data points for the drivers are generated synthetically. We fix the number of drivers m for a day as 30. Randomly chosen 30 locations on the Chicago city map are fixed as the drivers' initial or current locations CL_{v_k}. Initially, the attributes $DT_{v_k}, WD_{v_k}, TL_{v_k}, RC_{v_k}$ are equal to 0, and the capacity $CAP_{v_k} = 2$ for every $v_k \in \mathcal{D}$.

Procedure in every iteration

We assume each vehicles' average speed is 27 $miles/hr$. For each iteration, we pick a 15-minutes chunk of trips to be allocated as $\mathcal{R}^t$. Each driver $v_k \in \mathcal{D}^0$ is checked whether v_k is already busy in providing service to the ride assigned to it in previous iterations or the driver is free. This we do by comparing the time passed since the ride was given to v_k in previous iterations, assuming the speed of v_k is 27 $miles/hr$. If $v_k \in \mathcal{D}^0$ has already completed the service or is free, it is added to $\mathcal{D}^t$. Hence, we get the set $\mathcal{D}^t$ containing all the available drivers for the coming period.

With input $(\mathcal{R}^t, \mathcal{D}^t)$, we apply Algorithm 1. At the end of the ride, CL_{v_k} of v_k is assumed to be the location where the allocated ride ends. Similarly, all the other parameters in the tuple representing v_k are updated. Due to fewer vehicles, some requests from $\mathcal{R}^t$ may not be allocated in the period t. In that case, we count those remaining requests and add them to the next chunk considered in the next iteration. The procedure in every iteration is summarized in Algorithm 2.

Algorithm 2: Procedure in each iteration

```
1: Pick a 15-minutes chunk of trips to be allocated as $\mathcal{R}^t$.
2: Add the unmatched rides from $\mathcal{R}^{t-1}$ to $\mathcal{R}^t$.
3: for $\forall v_k \in D^0$ do
4:     Check whether $v_k$ is busy in providing service.
5:     if already completed the service then
6:         Add $v_k$ to $\mathcal{D}^t$.
7:     end if
8: end for
9: Apply Algorithm 1 for $(\mathcal{R}^t, \mathcal{D}^t)$.
10: Find the unmatched rides in $\mathcal{R}^t$.
11: $\forall v_k \in \mathcal{D}^0$, update elements in $(CL_{v_k}, DT_{v_k}, WD_{v_k}, TL_{v_k}, RC_{v_k}, CAP_{v_k})$.
```

State-of-the-art algorithm for comparison: We also apply the procedure on a state-of-the-art algorithm (we call, BZ) proposed by Bei and Zhang (2018) for $OBJ1$. BZ guarantees to achieve a 2.5 approximate solution for $OBJ1$ in polynomial time.

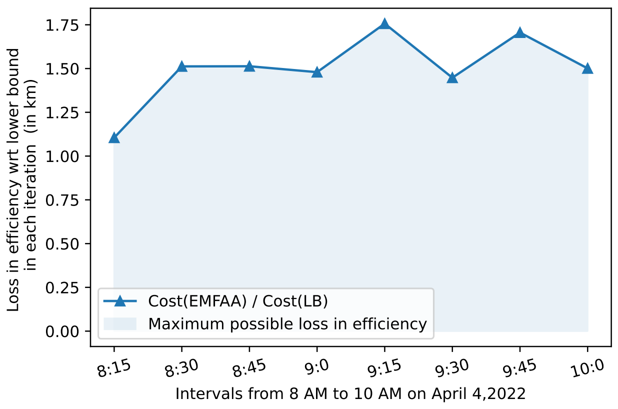
Comparison metrics

We compare the two algorithms, **BZ** and Algorithm 1, based on the following metrics:

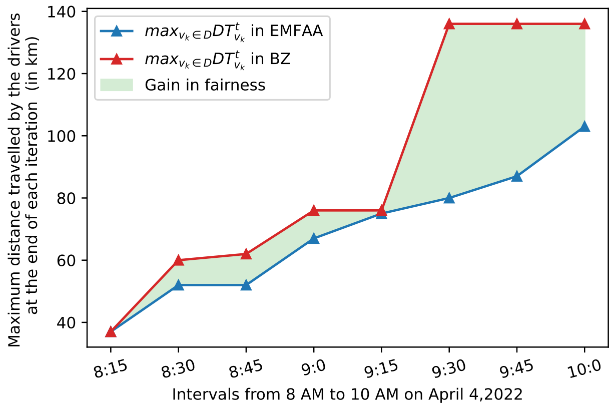
1. **Price of fairness:** Finding the efficient 3-D assignment is a computationally expensive problem, and therefore, comparing the efficiency generated by the algorithm with the optimally efficient solution is intractable. As $OBJ1$ is a minimization problem, for the analysis, we consider the *lower bound* (**LB**) for $OBJ1$ in each iteration. The **LB** is computed as the sum of the cost of minimum weighted matching among the nodes $\mathcal{R}^t$ and that of minimum bipartite matching between $\mathcal{R}^t$ and $\mathcal{D}^t$. We compare the efficiency by Algorithm 1 with the lower bound of efficiency in that iteration.
2. **Increase in fairness:** We compare the metrics *fairness* and *proportional fairness* after every iteration, corresponding to the allocations by Algorithm 1 and **BZ**.

Results

Figure 4a shows the 'loss in efficiency' in the matching given by **EMFAA** in each iteration. The approximation factor concerning the **LB** is at most 1.75. This hints that the matching provided by **EMFAA** doesn't lose much efficiency. Notice that **BZ** provides a 2.5 approximation factor for efficiency. Figure 4b shows the gain in fairness for the matching by **EMFAA** in comparison with that by **BZ**. The plot shows that fairness increases with up to 70% (for interval $9:30$). In summary, Figure 4 shows that a little attention to fairness can bring significantly fair allocation for drivers while not losing much efficiency.



(a) Loss in efficiency



(b) Gain in fairness

Figure 4: Results on datset for trips between 8-11 AM on April 4, 2022 in Chicago city

Conclusion

From the theoretical analysis in the paper, we conclude that simultaneously achieving both the objectives, namely, efficiency and fairness, is infeasible or challenging. However, from the experimental analysis, a bit of consideration towards fairness while aiming for efficiency can result in a fine balance between the two objectives. For both objectives, theoretical analysis of the approximation factor given by **EMFAA** is an exciting direction to follow.

The experiments done in this paper are on a small part of the available data set. In the future, we plan to run experiments on various two-phase algorithms, some of which are already mentioned in this paper, on a more extensive data set. We expect the experiment results for a more extensive data set will have a similar pattern. For the future avenue, some interesting directions are to study the balance of the two objectives problem for other fairness metrics with ride-sharing efficiency and heterogeneous capacity of the vehicles.

Acknowledgments

Shakya is supported by a project funded by Grants-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

References

Bei, X.; and Zhang, S. 2018. Algorithms for Trip-Vehicle Assignment in Ride-Sharing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Bokányi, E.; and Hannák, A. 2020. Understanding inequalities in ride-hailing services through simulations. *Scientific reports*, 10(1): 1–11.

Cai, H.; Wang, X.; Adriaens, P.; and Xu, M. 2019. Environmental benefits of taxi ride sharing in Beijing. *Energy*, 174: 503–508.

Caulfield, B. 2009. Estimating the environmental benefits of ride-sharing: A case study of Dublin. *Transportation Research Part D: Transport and Environment*, 14(7): 527–531.

Chicago Data Portal. 2022. Taxi Trips. https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psew. Accessed: 2022-09-20.

Diakopoulos, N. 2015. How Uber surge pricing really works. https://www.washingtonpost.com/news/wonk/wp/2015/04/17/how-uber-surge-pricing-really-works/?utm{_}term=.b16b8c1e885d. Accessed: 2022-10-15.

Fairwork. 2020. Fairwork India Ratings 2020: Labour Standards in the Platform Economy. https://fair.work/en/fw/publications/fairwork-2020-annual-report/. Accessed: 2022-10-13.

Gabow, H. N. 1990. Data structures for weighted matching and nearest common ancestors with linking. In *Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, 434–443.

Garey, M. R.; and Johnson, D. S. 1990. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co. ISBN 0716710455.

Goossens, D.; Polyakovskiy, S.; Spieksma, F. C.; and Woeginger, G. J. 2012. Between a rock and a hard place: the two-to-one assignment problem. *Mathematical methods of operations research*, 76(2): 223–237.

Hall, J. V.; and Krueger, A. B. 2018. An Analysis of the Labor Market for Uber's Driver-Partners in the United States. *ILR Review*, 71(3): 705–732.

Jia, Y.; Xu, W.; and Liu, X. 2017. An optimization framework for online ride-sharing markets. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*, 826–835. IEEE.

Kedmey, D. 2014. This Is How Uber's 'Surge Pricing' Works. https://time.com/3633469/uber-surge-pricing/. Accessed: 2022-10-13.

Lesmana, N. S.; Zhang, X.; and Bei, X. 2019. Balancing Efficiency and Fairness in On-Demand Ridesourcing. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Luo, K.; and Spieksma, F. C. R. 2020. Approximation Algorithms for Car-Sharing Problems. In Kim, D.; Uma, R. N.; Cai, Z.; and Lee, D. H., eds., *Computing and Combinatorics*, 262–273. Cham: Springer International Publishing. ISBN 978-3-030-58150-3.

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. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 2210–2217.

Rawls, J. 1971. *A Theory of Justice*. Cambridge, Massachussets: Belknap Press of Harvard University Press, 1 edition. ISBN 0-674-88014-5.

Shaheen, S.; Cohen, A. M.; Bayen, A.; et al. 2018. The Benefits of Carpooling. Technical report, Institute of Transportation Studies, UC Berkeley.

Sühr, T.; Biega, A. J.; Zehlike, M.; Gummadi, K. P.; and Chakraborty, A. 2019. Two-sided fairness for repeated matchings in two-sided markets: A case study of a ride-hailing platform. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 3082–3092.

Xu, Y.; and Xu, P. 2020. Trade the System Efficiency for the Income Equality of Drivers in Rideshare. In Bessiere, C., ed., *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 4199–4205. International Joint Conferences on Artificial Intelligence Organization. Main track.

Yan, L.; Luo, X.; Zhu, R.; Santi, P.; Wang, H.; Wang, D.; Zhang, S.; and Ratti, C. 2020. Quantifying and analyzing traffic emission reductions from ridesharing: A case study of Shanghai. *Transportation Research Part D: Transport and Environment*, 89: 102629.

Yin, B.; Liu, L.; Coulombel, N.; and Viguié, V. 2018. Appraising the environmental benefits of ride-sharing: The Paris region case study. *Journal of Cleaner Production*, 177: 888–898.