TESLA: A Centralized Taxi Dispatching Approach to Optimizing Revenue Efficiency with Global Fairness

Huigui Rong[1], Qun Zhang[1], Xun Zhou[2], Hongbo Jiang[1], Da Cao[1], Keqin Li[3]
{ronghg,xingfuzq,hongbojiang,caoda}@hnu.edu.cn,xun-zhou@iowa.edu,lik@newplatz.edu
[1]Hunan University, Changsha, China, 410082
[2]University of Iowa, Iowa City, IA 52242
[3]SUNY New Platz, New Platz, NY 12561

ABSTRACT

Taxi service plays an important part in urban transportation systems. Traditional taxi business model suffer from low efficiency. The development of smart phones and mobile computing has made centralized taxi dispatching possible. Existing taxi dispatching algorithms mostly focus on optimizing the overall profit of the entire fleet but ignore the income fairness issue of drivers. This may cause problem for driver engagement and drivers' working desire. In this paper, we study how to improve the overall taxi drivers' revenue in a fleet while addressing the fairness issue in a central dispatching model. We first identify the unfairness issue from the taxi dispatch process and propose a novel solution to match taxis and passengers in real time, namely cen**T**ralizE**d** di**S**patch with g**L**obal f**A**irness (TESLA). We design a dispatching system to improve driver's revenue efficiency while considering driver fairness, and propose a route recommendation algorithm to help drivers get dispatched faster. Experiment results on a real taxi dataset collected from 1400 taxis in Changsha, China for one year suggest that our TESLA approach outperforms the real taxi operation strategy and other approaches in terms of income fairness, and can improve taxis' revenue efficiency, reduce passengers' waiting time compared with the real data. TESLA also recommend better routes for waiting drivers to get next dispatch sooner and is computationally efficient.

CCS CONCEPTS

• **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

centralized dispatching, global fairness, revenue efficiency

ACM Reference Format:
Huigui Rong[1], Qun Zhang[1], Xun Zhou[2], Hongbo Jiang[1], Da Cao[1], Keqin Li[3]. 2019. TESLA: A Centralized Taxi Dispatching Approach to Optimizing Revenue Efficiency with Global Fairness. In *KDD Workshop on Urban Computing (UrbComp '20), August 24, 2020, San Diego.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

UrbComp '20, August 24, 2020, San Diego, CA

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In recent years, with the development of smart phones, centralized dispatching has emerged as a new model for the taxi and ride-sharing businesses. Ride requests from passengers are matched with nearby drivers. A recent study measured the efficiency of the traditional taxi service system and the centralized dispatching model[13]. All the taxis and passengers are matched using a minimum weight bipartite graph matching algorithm. The results showed that the traditional taxi service system is far from reaching the optimal efficiency. The main reason is the lack of a centralized taxi dispatching system to rationally allocate taxis.

There has been a number of related research on efficient matching techniques for centralized taxi dispatching. In industry, Uber, Lyft and DiDi Chuxing are successful examples of such new business models. However, most of the existing techniques and research focus on maximizing request response rate or revenue of the entire fleet in a city. While this objective is certainly important, we notice that in a centralized dispatching model taxi drivers have limited freedom of choice but to follow the direction. This leads to questions about whether these mechanisms can guarantee **"fairness"** among drivers. If a driver is not dispatched for profitable trips for a long period of time, the driver might take a hit in her income compared to other drivers. Also, sometimes experienced drivers might discover regions where profitable trips can be dispatched more easily (as a result of the passenger distribution and the dispatching algorithm), or even use malware apps to get more orders [2]. This will be unfair to new and honest drivers who have limited experiences and could have an adversarial impact on new driver engagement for businesses like Uber and DiDi.

Therefore, addressing the fairness of income is important for the dispatching algorithm. Unfortunately, this issue has rarely been discussed in research. In this paper, we evaluate the fairness of drivers' income under a centralized dispatching model and propose a solution to improve the fairness of the dispatching strategy while still maintain the revenue of the entire fleet.

1.1 Related Work

Related research on the optimization of taxi operation strategies can be roughly divided into two categories: (1) Recommend routes for taxis by analyzing historical data, (2) Match taxis and passengers on the road in real time.

Category 1: The first group of works recommend routes for taxis by analyzing historical data and detecting pick-up hotspots or high-margin areas in cities. For example, the main idea of some prior works [4, 12] was to probe the hotspots of the city by analyzing historical data, then recommend some hotspots for taxis. Another paper [9] proposed a framework (TaxiRec) for evaluating and discovering the passenger-finding potentials of road clusters and incorporated it into a recommender systems for taxi drivers to seek passengers. Ye et al. developed new methods that combined parallel computing and the simulated annealing with novel global and local searches to solve the mobile sequential recommendation (MSR) problem [11]. There were also some researches recommended for taxis by analyzing the average net profit and other indicators of each road segment [7, 17]. In addition, in order to reduce trip mileage, Bastani et al. proposed a flexible mini-shuttle like transportation system called flexi, with routes formed by analyzing passenger trip data from a large set of taxi trajectories [1]. Obviously, using the above strategies all taxis may be recommended to go to the same high-margin areas and the taxi gathering phenomenon may be aggravated. **These works do not perform passenger-driver matching, and do not consider driver fairness.**

Category 2: The second group of works, which are the most relevant to our work, obtain the information of taxis and passenger on the current road network in real time, and then reasonably match them. Zhang et al. proposed a taxi order dispatch model based on combinatorial optimization, they assigned passenger orders for each taxi by calculating the probability that a taxi will accept a passenger order [15]. Xu et al. modeled order dispatch as a large-scale sequential decision-making problem, then solved the probelm by a learning and planning manner [10]. Zheng et al. studied the O2O taxi scheduling problem using a stable marriage method. It aimed to balance the interests of taxi drivers, taxi companies and passengers [16]. Dai et al proposed a RRA-LSP algorithm to recommend for taxis by calculating distance between taxi and passengers [3]. Zhang et al. firstly predicted the passenger's ride service preferences based on passenger information, and then assigned the taxi according to the passenger's preferred service [14]. Others plan the taxi route through the needs of passengers and dispatched taxis via ridesharing [5, 8]. Most of the above strategies only considered overall benefits but rarely consider the fairness of income among drivers, which may reduce the enthusiasm of taxi drivers.

In Summary, almost none of the above related works consider drivers' income fairness while doing taxi dispatching, except for the paper [3]. It proposed a LSP algorithm, which tried to ensure the income fairness between taxis by constructing an EVA metric for each passenger and taxi pair. However, the calculation of the EVA metric is very simple, and it only attempted to ensure fairness through the accumulated income of taxis, neglecting the working time of taxis, the distribution of passengers in real time, and other factors. Therefore, in this paper we try to come up with a better solution for this issue while still improving the overall profit of the fleet.

1.2 Our Contributions

Improving dispatching fairness is **challenging** because **(1)** the revenue of the fleet is still the most important objective. Optimizing both driver fairness and overall revenue at the same time is very hard, if at all possible, and requires careful trade-offs. **(2)** Also it is non-trivial to define "fairness" in a fair way. Simply averaging total income does not value the working efficiency and service quality of top drivers and might create new unfairness.

In this paper, we propose a novel solution to match taxis and passengers in real time, namely TESLA (short for "cen**T**raliz**E**d di**S**patch with g**L**obal f**A**irness"). It aims to address the driver income fairness issue in centralized taxi dispatching while still achieving high revenue of the whole fleet. Our contributions are listed as follows:

- We formulate the taxi dispatching with global fairness problem and propose a TESLA approach to match the taxis and passengers. We design a strategy to dynamically calculate the priority of taxis for selecting taxis in the centralized dispatching to improve the fairness among taxis.
- We present a route recommendation algorithm to make recommendations for idling taxis so that they can be dispatched as quickly as possible.
- We conduct extensive experimental analysis and simulations to verify the effectiveness of our approach. Experimental results show that our algorithm can increase the revenue efficiency of taxi drivers by approximately 8.01%, reduce passengers' waiting time by approximately 20.6%, reduce taxis' seeking time by approximately 16.3% compared with real data and significantly improve the fairness of income.

Scope: This paper is focused on how to improve the fairness of the dispatching algorithms. There are other business options to address the fairness issue such as giving incentives to new drivers. However, they are not relevant to dispatching algorithm design therefore not discussed in this paper.

In this section we introduce concepts used to formulate the problem and then present the formal problem definition. First we introduce the concepts.

***Definition 1** (Road Network):* A road network is a undirected graph $G = (I, R)$, where R is a set of undirected edges representing road segments, and I is the set of intersections or joints between adjacent road segments.

***Definition 2** (Ride Request):* A passenger ride request is a tuple $q =< o_q, d_q, st_q, dt_q, p_q >$, where st_q is the time of the request, o_q is the origin location of the trip, d_q is the destination of the requested trip. dt_q is the expiration time of the request (that is, the passenger will cancel the request after dt_q if not picked up, it is calculated from the time st_q), and p_q is the fare of this ride request. o_q and d_q are both on road segments in R.

***Definition 3** (Vehicle Status):* The status of a taxi is defined as $v =< l_v, t_v, d_v >$, where t_v is the current time, l_v is the location of the taxi at t_v, and the destination of the vehicle (if occupied) is d_v. If the taxi is vacant, $d_v = \emptyset$.

***Definition 4** (Mileage distance between roads):* The Mileage Distance (MD) between two road segments $MD(r_i, r_j)$ is the length of shortest path on G from the centroid of r_i to the centroid of r_j.

The distances can be obtained in real time through routing services such as Baidu Map API. We use MD to represent the whole set of Mileage Distances between pairs of roads.

***Definition 5** (Time distance between roads):* The Time Distance (TD) between two road segments $TD(r_i, r_j)$ is the travel time from road r_i to r_j. We use TD to represent the Time Distance between all pairs of roads. Note TD is time dependant and can vary over time. However, this paper does not discuss how to estimate TD (a.k.a., ETA problem). We assume this information is available in real-time (e.g., via the Baidu Map API).

***Definition 6** (Trip Fare):* The total fare of a trip from road r_i to road r_j is calculated by the mileage distance. We use Eq. (1) to calculate it:

$$Fare(r_i, r_j) = \begin{cases} p_s, & \text{if } MD(r_i, r_j) \le d_s \\ p_s + \\ (MD(r_i, r_j) - d_s) \times p_{unit}, & \text{otherwise.} \end{cases} \tag{1}$$

where p_s is the minimum fare, d_s is the minimum distance, p_{unit} is the fare of unit mileage. If the trip length is below d_s, the minimum fare is charged. Otherwise additional fare is charged based on additional miles beyond d_s. This equation is based on the taxi fare rules in most Chinese cities.

***Definition 7** (Net Revenue Efficiency):* Net Revenue Efficiency represents the net revenue earned by the taxi driver per unit working time. Note, working time means that the driver is either taking the passenger(s) to the destination, or on the way to pick up the next passenger (in centralized mode) or actively seeking the next passenger (in the traditional mode). If the driver is off work (e.g., for lunch), the time is excluded from working time.

For simplicity, Net Revenue Efficiency is abbreviated as *Revenue Efficiency* or *RE*. The calculation method is as shown in Eq. (2):

$$RE = \frac{M - (T_{seek} + T_{drive}) \times fuelC}{T_{seek} + T_{drive}} \tag{2}$$

where M denotes the total revenue obtained by the taxi, T_{seek} denotes the taxi seeking/cruising time while vacant, T_{drive} denotes the taxi driving time while occupied, and $fuelC$ denotes the fuel consumption per unit working time.

***Definition 8** (Rating):* The rating level (denoted as rat) of a taxi is determined by various factors, such as driving skills, reputation, etc. Different taxi companies may have different methods for calculating the rating. In this paper, our focus is not on how to calculate and adjust the rating, but how to incorporate rating into our solution to try to ensure fairness between taxis. rat is an integer score, where $1 \le rat \le n$ and n is the highest level (best rating).

***Definition 9** (Fairness):* We use the variance of net revenue efficiency of each taxi as an indicator of fairness. Smaller variance indicates better fairness. Since the rating of individual taxis may be different, in order to measure fairness, we need calculate the variance of the net revenue efficiency of taxis at each rating level. Therefore, we quantify fairness as a value, expressed by F, it is measured by calculating the mean of sum of the variances of the net revenue efficiency of taxis at each rating stage. The smaller the value, the fairer the income between taxis. The calculation of F is shown in Eq. (3):

$$F = \frac{1}{n} \sum_{rat=1}^{n} \frac{\sum (RE_{i_{rat}} - \bar{RE})^2}{m_{rat}} \tag{3}$$

where rat is the rating of a taxi, ranging from 1 to n, m_{rat} is the total number of taxis rated as rat. If all taxis have the same rating, then F is the variance of the net revenue efficiency of all taxis.

Problem Statement: Based on the above definitions, our problem can be stated as follows:

Given:
- A time window T partitioned into small time slots t
- A passenger request set Q_t at each time t in T
- A taxi status set V_t at each time t in T

Find:
- A matching M_t for each time slot t between Q_t and V_t

Objectives:
(1) Maximize the average net revenue efficiency of all the taxis in the fleet over T.
(2) Minimize the value of F in the fleet over T.

Constraints: Each match $(q, v)_{q \in Q_t, v \in V_t}$ satisfies the following constraints:
(1) If $v.d_v = \emptyset$, then: $v.t_v + TD(v.l_v, q.o_q) \le q.dt_q$
(2) If $v.d_v \ne \emptyset$, then: $v.t_v + TD(v.l_v, v.d_v) + TD(v.d_v, q.o_q) \le q.dt_q$
(3) If no taxi can be found to match a request, the request will not be accepted

In the problem statement, we try to match each ride request with a taxi. In this paper, we make **two assumptions**: (a) Once a taxi has responded to a ride request, it cannot be changed until it completes the request; (b) The taxi cannot respond to other ride requests before completing current ride request.

Note constraint (2) above suggests that an occupied taxi can be matched as well, provided that the taxi will be completing the current trip soon enough to pick up the next passenger before the request expires. Otherwise the request will be cancelled.

2 A BASELINE ALGORITHM

In this section, we extend a recent work [13] for taxi-passenger matching and use it as a baseline algorithm. Then we demonstrate the importance of considering fairness in the process of centralized dispatching.

2.1 A Baseline Taxi Dispatching Algorithm

We can model ride requests and vacant taxis at each time t as vertices in a bipartite graph, and model the matches between the requests and the taxis as edges E_t in the graph. The cost of a taxi responding to a ride request can be modeled as the weight of the edge W_t. Then this matching process can be modeled as a minimum weight bipartite graph matching problem. Therefore, a centralized dispatching process at time t can be described as finding a matching set M_t between Q_t and V_t in the bipartite graph $G = (Q_t, V_t, E_t, W_t)$ that satisfies the following objective function:

$$\arg\min_{m_{ij}} \sum_{i=1}^{|V_t|} \sum_{j=1}^{|Q_t|} W_t(v_i, q_j) m_{ij}$$
$$s.t. \begin{cases} v_i \in V_t, & i = 1, 2, 3, ..., |V_t|; \\ q_j \in Q_t, & j = 1, 2, 3, ..., |Q_t|; \\ m_{ij} \in M_t; \\ \sum_{j=1}^{|Q_t|} m_{ij} \le 1, & i = 1, 2, 3, ..., |V_t|; \\ \sum_{i=1}^{|V_t|} m_{ij} \le 1, & j = 1, 2, 3, ..., |Q_t|. \end{cases} \tag{4}$$

where

$$m_{ij} = \begin{cases} 1, & \text{if ride request } q_j \text{ is assigned to taxi } v_i \\ 0, & \text{if ride request } q_j \text{ is not assigned to taxi } v_i \end{cases} \tag{5}$$

where $|Q_t|$ and $|V_t|$ respectively represent the number of ride requests and the number of taxis at time t. The last two of the constraints indicate that each taxi can only respond to one ride request (or does not respond to any request), and each ride request can only be assigned to one taxi (or not assigned to any taxi). We will employ the Kuhn-Munkres (KM) algorithm [6] to solve the problem. Note the nodes, edges, and weights change over time.

If a taxi is vacant or can finish the current trip before a potential request expires, then the taxi and the request can be matched. The cost for each mathcing consists of two parts: **fuel cost**, and **distance cost**. For fuel cost, in this paper we assume it is proportional to the working time rather than the distance driven since sometimes congestion or low traffic speed can cause high fuel consumption, although the distance is shorter. Therefore we amortize the cost of fuel to every working time unit as defined in Definition 7. For distance cost, we also convert it to cost in money. Ideally, we hope that the taxi is occupied in every single minute (100% driving time). That will give the highest net revenue efficiency. However, this is impossible. Therefore we calculate how much money the taxi would have earned had it been occupied rather than vacant on the way to pick up the next passenger. This can be calculated by Eq. (1) in Definition 6, assuming a passenger was on board while the taxi went to pick up the new passenger. This part of the cost will penalize long-distance pickups, although the fuel cost might not be very high. To sum up, the final weight function is defined in Eq. (6):

$$W_t(v, q) = Fare(v.l_v, q.o_q) + TD(v.l_v, q.o_q) \times fuelC \tag{6}$$

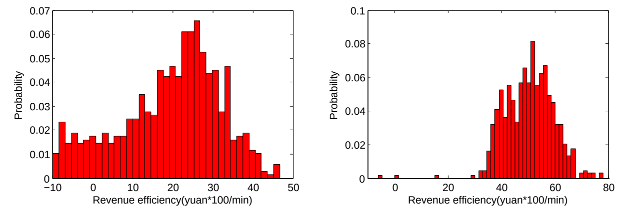
The above equation is for the case when v is a vacant taxi at the time of matching. If the taxi is occupied but it will soon arrive at the destination, the calculation of the weight will slightly change. The distance cost and the fuel cost will be calculated from the destination of its current trip rather than its current location. The weight in this case is calculated by Eq. (7):

$$W_t(v, q) = Fare(v.d_v, q.o_q) + TD(v.d_v, q.o_q) \times fuelC \tag{7}$$

After all the weights are calculated, we can use the **minimum weight** bipartite graph matching algorithm [13] to find the optimal matching M_t for each time t. The algorithm runs the matching for every t in time interval T. At each time t, we get taxi set and ride request set, and calculate the matching weight according to Eq. (6) and Eq. (7). Then we perform the minimum weight bipartite graph matching algorithm. Due to space limit we do not present the pseudo code of the algorithm, which can be found in [13].

2.2 Fairness Analysis on Real Data

Next, we analyze the fairness of the baseline algorithm. Here, the dataset we used for analysis is obtained from a taxi company in Changsha, China with 1400 taxis. The data contains real passenger pickup locations and drop-off event over a year, and the area covered by the dataset extends along longitude co-ordinates, 112.854452 to 113.083556 and the latitude co-ordinates, 28.149096 to 28.239767, which covers over 800 major roads and over 95% of all the taxi pickups and drop-offs, and each record has the latitude-longitude coordinates and timestamps of the pick-up and drop-off events, along with total traveled distance and the fare of the corresponding trip, and etc. We use 30-day historical data for this analysis. All the fare rate parameters are obtained from the taxi company. Specifically, p_s = 6 yuan, d_s = 2 km, p_{unit} = 1.8 yuan. In addition, since there is no rating in our dataset, we consider the fairness among all the drivers. Then after a complete centralized dispatching, we analyze the variance of the net revenue efficiency of all taxis.



(a) Distribution of real data (b) Distribution of baseline algorithm

Figure 1: Distribution of 30-day average revenue efficiency

In order to analyze fairness, we compare the revenue efficiency distribution of drivers under the traditional passenger-seeking model and the baseline dispatching algorithm. Our dataset was collected at the time when the taxis were still following the traditional model. So for the traditional model we directly use the real data to calculate the revenue efficiency based on their actual income and working time. If the gap between two consecutive trips of the same taxi is longer than 25 minutes, we consider the taxi to be off work and do not count this time towards the total working time. The net revenue efficiency is calculated using Definition 7. For the baseline algorithm, we use the pickup location and time of the passengers in the real data to simulate the requests. The expiration time length is set to a random number between 1 to 15 minutes. We also simulate the movement of each taxi based on the matching result of the baseline algorithm.

For the two business models, we calculate 30-day average revenue efficiency for each taxi from 7:00 am to 16:00 pm, which corresponds to the day-time drivers' working schedule. We draw the distribution of revenue efficiency of the two business models, respectively. Fig. 1 shows the distributions.

We can find from Fig. 1(a) that under the traditional business model, the difference in revenue efficiency between drivers is very high. The value of F is 157.27. Note that because we calculate the net revenue efficiency, some taxi drivers may not make up for the fuel consumption, which result in negative revenue efficiency. Similarly, we can observe from Fig. 1(b) that although the baseline algorithm improves the revenue efficiency of taxi drivers, the difference in revenue efficiency among taxi drivers is still large. The value of F is 84.07. And there are still several individual taxi drivers with negative revenue efficiency.

Through the above analysis, it can be summarized that although the baseline algorithm can increase the average revenue efficiency of taxi drivers, it does not explicitly guarantee the fairness of revenue. For example, the home/initial locations of the taxis might affect their chance to take profitable passengers. Sometimes a driver gets unlucky by being asked to take short and nonprofitable trips. However, the driver was dispatched by the central control and had no other choice. Therefore, we will improve the baseline algorithm in the next section by integrating fairness in the algorithm.

3 A CENTRALIZED DISPATCHING SCHEME WITH GLOBAL FAIRNESS

In this section we propose a novel solution to match taxis and passengers in real time with global fairness guarantee, namely the "cen**T**raliz**E**d di**S**patch with g**L**obal f**A**irness" (TESLA) approach. In order to implement the fairness of taxi dispatching as far as possible, we design two strategies: 1) Adjust the weight calculation in the bipartite graph by incorporating the expected fare of ride request to balance the income of both low-income taxi drivers and high-income ones. 2) Calculate the taxis priority dynamically, and then select taxis according to their priorities to participate in matching.

3.1 Weight Adjustment

In the matching step of time, we address the fairness by adjusting the weights W. Our expectation is to make the taxis with high revenue efficiency more likely to be matched with lower-fare requests, and vice versa. We sort the taxis in descending order of their accumulated revenue efficiency and divide them into three groups: high income (top 25%), medium income (25%-75%), and low income (bottom 25%). For each ranking category, we adjust the weights W as follows:

$$W_t(v,q) = \begin{cases} e^{W_t^{'}(v,q)+q.p_q}, & \text{if } v.rank = high \\ e^{W_t^{'}(v,q)}, & \text{if } v.rank = mid \\ e^{W_t^{'}(v,q)-q.p_q}, & \text{if } v.rank = low. \end{cases} \quad (8)$$

where $W_t^{'}(v,q)$ is the weight value calculated by Eq. (6) and Eq. (7), and $rank$ represents different revenue ranks. After the above changes, we can find that if the taxi revenue rank is high, the lower the passenger's fare $q.p_q$ is, the smaller the matching weight is, which makes it more inclined to match the passengers with lower fare; if the taxi revenue rank is low, the matching tendency is the opposite. If the taxi revenue rank is medium, there is no specific matching tendency.

3.2 Priority Exploration

Simply addressing revenue efficiency inequality might not be sufficient as there are other factors to consider. We also calculate a priority score for each taxi. Then we divide the taxis into batches based on the priority scores. Each batch participates in the matching sequentially. Unmatched taxis will participate in the matching together with the next batch. To design the priority score, we consider the following factors.

Revenue Efficiency (*RE*). Because the fairness evaluation in revenue efficiency is different among taxis drivers, we take the accumulated revenue efficiency in the current day of a taxi as an influencing factor of the priority. For a taxi with higher accumulated revenue efficiency, we should lower appropriately its priority in the passenger matching process. On the contrary, we enhance its priority to make it easier to match passengers.

Passenger density near a vehicle (ρ). Only using the accumulated revenue efficiency as the priority will create issues. When the total request volume is high in a certain region, most drivers nearby will get a chance to take passengers. In such cases, it is more important to address the requests than to prioritize low-income drivers. Therefore, we also take the passenger density near each taxi as a priority factor. We should consider appropriately increasing the priority of a taxi when the passenger density near it is relatively high. So we use the following Eq. (9) to calculate the passenger density ρ:

$$v.\rho = \sum_{i}^{k} TD(q_i.o_{q_i}, v.l_v) \quad (9)$$

where k represents the k-nearest ride requests from the taxi v. Then the above equation can be understood as: The passenger density ρ near the taxi v is equal to the sum of the time distance (TD) from the nearest k ride requests to v.

Vacant time of a vehicle (*emptyT*). In addition to the above two parameters, we also introduce the vacant time of a taxi. If a taxi has been vacant for a long time, the working desire and enthusiasm of the taxi driver will be greatly affected. In such a case we should gradually increase priority when one taxi is waiting for the next dispatch for a long time.

Rating of a vehicle (*rat*). We also need to consider the rating of each vehicle. Vehicle with higher rating may have higher priority, which is also an incentive for taxi drivers.

In general, we get the priority expression by fusing the weighted factors of *RE*, ρ, *emptyT* and *rat*, as shown in Eq. (10):

$$v.priority = w_1 \cdot v.RE + w_2 \cdot v.\rho + w_3 \cdot v.emptyT + w_4 \cdot v.rat \quad (10)$$

It should be noted that *RE*, ρ, *emptyT* and *rat* have different orders of magnitude, so we need to standardize them when calculating priorities. Standardization uses Eq. (11):

$$standardizedValue = \frac{originalValue - min}{max - min} \quad (11)$$

The priorities are used to determine the order in which the taxis are matched with ride requests. After matching each batch of taxis, the matched taxis and requests will be removed.

Now we need to consider how the four weight coefficients in Eq. (10) should be set. This parameters should be set such that the average revenue efficiency is maximized and the value of the F is minimized in each rating phase. To achieve this goal, we define an objective function as the sum of the average *RE* and *F* ratios of all taxis in each rating phase. The function is shown in Eq. (12):

$$Target = \frac{1}{n} \sum_{rat=1}^{n} \frac{avg_{rat}(RE)}{F_{rat}} \quad (12)$$

Therefore, we can convert the above problem of calculating weight coefficient into an optimization problem defined by Eq. (13):

$$\max_{w_i} Target, \quad i = 1, 2, 3, 4 \quad (13)$$

Since our objective function is not directly related to *priority*, but requires a complete centralized dispatching to calculate the *Target*, and our historical data does not include the *priority*, so an algorithm like gradient descent may not applicable, so we choose genetic algorithm to solve this optimization problem. It should be noted that there is no rating data in our dataset, so we first randomly set the rating of each taxi. In this paper, we assume that the range of ratings is an integer between 1 and 5. In combination with the actual situation, We set the 70% taxi rating to 3 or 4 and the remaining 30% taxi rating to 1, 2 or 5.

The coding format of chromosome in genetic algorithm is floating-point encoding, i.e., $f_1 \rightarrow f_2 \rightarrow f_3 \rightarrow f_4$, which means $w_1 = f_1, w_2 = f_2, w_3 = f_3, w_4 = f_4$. The fitness function is our objective function ($Target$). Then we set the population size to 60, the crossover probability to 0.85, the mutation probability to 0.01, the variable asynchronous length to 0.1 (that is, the mutation gene plus 0.1 or minus 0.1), the maximum algebra is 500, and if it does not produce a better individual after 200 iterations, the iteration is terminated. In the iterative process, the "elite retention mechanism" (that is, each generation retains the best individual in the

Algorithm 1: Centralized Dispatching algorithm with Global Fairness (*TESLA*)

Input: Time interval T, Mileage distance MD, Time distance TD
Output: A feasible allocation M

```
M = Φ;
for each t in T do
    M_t = Φ, candidateV_t = Φ;
    Get current taxi set V_t and ride request set Q_t;
    Initialize weight W_t with 0;
    for each taxi v in V_t do
        Calculate the priority of v;
    Sort V_t by their priority from big to small;
    if Num(V_t) ≤ Num(Q_t) then
        candidateV_t = candidateV_t ∪ V_t;
        Calculate W_t between candidateV_t and Q_t;
        M_t = M_t ∪ BipartiteGraph(candidateV_t, Q_t, W_t);
        Remove all v and q that have been successfully matched
          from candidateV_t and Q_t;
        if candidateV_t ≠ Φ then
            Recommend_routes(CandidateV_t);
    else
        index = 0;
        while index < Num(Q_t) do
            candidateV_t = candidateV_t ∪ V_t(index);
            index + +;
        Calculate W_t between candidateV_t and Q_t;
        M_t = M_t ∪ BipartiteGraph(candidateV_t, Q_t, W_t);
        Remove all v and q that have been successfully matched
          from candidateV_t and Q_t;
        Set a step size;
        while Q_t ≠ Φ and there are additional taxis in V_t left do
            Select additional size taxis according to their priority
              from V_t and add them to candidateV_t;
            Calculate W_t between candidateV_t and Q_t;
            M_t = M_t ∪ BipartiteGraph(candidateV_t, Q_t, W_t);
            Remove all v and q that have been successfully matched
              from candidateV_t and Q_t;
        if candidateV_t ≠ Φ then
            Recommend_routes(CandidateV_t);
    M = M ∪ M_t;
return M
```

previous generation) is used, and the "roulette" method is used in the selection operation.

In each iteration of the genetic algorithm, we simulate the operation of all the taxis for a whole month on the historical dataset (real request location and time) based on the current coefficients in the priority function and calculate the $Target$. The algorithm arrives at converged after 450 iterations.

The final objective function converged to 2.42, At this time, $w_1 = -5.2$, $w_2 = 2.0$, $w_3 = 1.5$ and $w_4 = 3.6$. Note these parameters only need to be learned once and can be directly used in the dispatching algorithms. They can be updated periodically as more data are collected (e.g., 3 months, 6 months or 1 year) but the time cost to calculate these parameter is not relevant to the performance of the system.

3.3 The TESLA Approach

Through the above two strategies, we improve the fairness of centralized dispatching. The new algorithm (TESLA) is shown as Algorithm 1.

Algorithm 1 runs the centralized dispatching algorithm in time interval T. At each time t, for each taxi, we calculate its priority and sort all taxis by their priorities. After that, we select candidate taxi set according to their priorities. If the number of V_t is less than the number of Q_t, all V_t are candidate taxis, otherwise, we select candidate taxis from V_t in batches according to the step $size$. Finally, algorithm performs the bipartite graph matching and recommends a road segment for the taxis not successfully being matched (discussed next). The algorithm returns all the matches.

The main difference between Algorithm 1 and Algorithm **??** is that Algorithm 1 selects candidate taxis in batches according to their priorities at each time t, and taxis that did not match successfully in the previous batch will participate in the matching process of the next batch, so each iteration requires multiple bipartite graph matching processes. Assuming that the number of V_t is n, the number of Q_t is m at each time t, and the step length of selecting taxi operation is $size$, then the number of taxis in each matching process is greater than or equal to $size$ and less than n, and the number of passengers is less than or equal to m, so the complexity of Algorithm 1 is $O((n+m)^3 \cdot \frac{n}{size} \cdot T)$. Similarly, at each time t, n and m are not large, so the time complexity is acceptable.

3.4 Route Recommendation

In Algorithm 1, if a taxi is not matched with any request, we can recommend the cruising routes based on historical data to increase the taxi's chance of being matched next time. This may also help reduce the response time to pick up the next passenger.

In order to make the taxi get matched more easily next time, we follow the idea of the weight design and priority scores. A taxi with high revenue rank may find a passenger more easily in a road with high passenger density but low expected fare according to the Eq. (8).

The passenger density of a road is measured by p_{find}(the probability of finding passengers in the road), p_{find} is calculated by the Eq. (14):

$$p_{find} = \frac{n_{pickup}}{n_{pickup} + n_{pass}} \tag{14}$$

where, n_{pickup} indicates the number of picking up on the road, and n_{pass} indicates the number of taxis passing through the road in the history data.

The average passenger fare of a road is measured by the average fare of passengers picked up on the road in the historical data.

The route recommendation process is shown as Algorithm 2. For each taxi that needs to be recommended, we obtain a set of nearby road segments and calculate the average probability of finding passengers of all nearby road segments. Then we take 2/3 of the probability as the threshold of the possible recommended roads for the taxi. We sort all the possible recommended roads according to the average passenger fare of each road from low to high (1-7). Then group the taxis to be recommended according to their revenue ranks (8-15). For the taxis with low revenue rank, we recommended

Algorithm 2: Route recommendation for unmatched taxi

```
   Input: All taxis that need to be recommended V
   Output: Cruise Route Recommended for Unmatched Taxis
1  Roads = Φ;
2  for each v in V do
3      Get the v.nearby_roads;
4      Calculate the average value avg of p_find for all
         v.nearby_roads;
5      v.p' = avg × 2/3;
6      Roads = Roads ∪ v.nearby_roads;
7  Sort Roads by their avgPay from small to big;
8  lowV = Φ, midV = Φ, highV = Φ;
9  for each v in V do
10     if v.rank = low then
11         lowV = lowV ∪ v;
12     if v.rank = mid then
13         midV = midV ∪ v;
14     if v.rank = high then
15         highV = highV ∪ v;
16 Sort lowV by their revenue efficiency from small to big;
17 r_index1 = Num(Roads) − 1;
18 for each v in lowV do
19     while Roads(r_index1) ∉ v.nearby_roads or
         Roads(r_index1).p_find < v.p' do
20         r_index1 − −;
21     v.recRoad = Roads(r_index1);
22     r_index1 − −;
23 for each v in midV do
24     index = random(0, Num(Roads));
25     while Roads(index) ∉ v.nearby_roads or
         Roads(index).p_find < v.p' do
26         index = random(0, Num(Roads));
27     v.recRoad = Roads(index);
28 Sort highV by their revenue efficiency from big to small;
29 r_index2 = 0;
30 for each v in V do
31     while Roads(r_index2) ∉ v.nearby_roads or
         Roads(r_index2).p_find < v.p' do
32         r_index2 + +;
33     v.recRoad = Roads(r_index2);
34     r_index2 + +;
35 return;
```

them to the nearby roads with the high average passenger fare and at the same time meeting the threshold (16-22); For the taxis with middle revenue rank, we randomly recommend a nearby road that meets the threshold (23-27); For the taxis with high revenue rank, the recommendation process is the opposite of that of the taxis with low revenue rank (29-34).

4 EXPERIMENTAL STUDY

In this section, we compare the performance of our TESLA approach with (i) the baseline algorithm (dispatching, no fairness), (ii) the LSP algorithm [3] (dispatching with fairness) and (iii) the real data (no dispatching nor fairness) by experimental simulation. **The dataset we use is the same as described in section 2.2**. We randomly set the rating of each taxi during the experimental where 70% of taxis are rated as 3 or 4, and the other 30% of taxis are rated as 1, 2 or 5. We use the pickup locations in the data as the request location of each order. The expiration time for each request is set to a random number between 1 to 15 minutes. The travel time of each trip is the same as in the real data. The matching is done for every 1 minute interval since our dataset has only 1400 taxis. However, for more taxis, the matching can be done more frequently. We simulate the order dispatching of all the taxis for a full month.

The experiments are done on a Lenovo QiTian M4350 Desktop Computer with 4GB RAM and Intel Core i3-3240 CPU running Windows 7. The experiments are implemented in Java in an Eclipse environment.

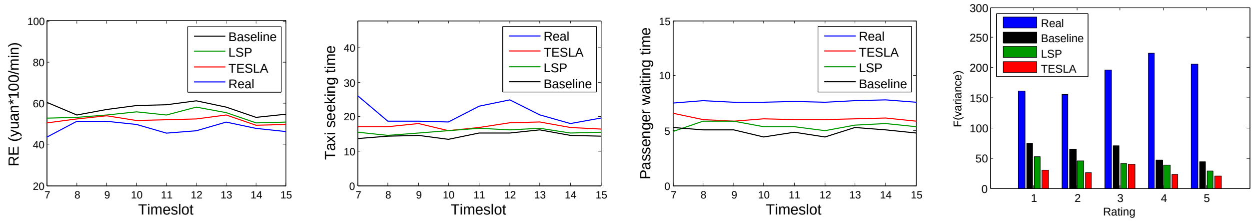
4.1 Experiment Results

Comparison with real dataset. First, we compare the revenue efficiency, taxi seeking time, and passenger waiting time among real historical data, baseline algorithm, LSP algorithm and our approach. The comparison is shown as Fig. 2. Fig. 2(a) shows a comparison of the average revenue efficiency. It can be found that our approach can increase the revenue efficiency of taxis by about 8.01% compared with real data; Fig. 2(b) is the comparison of taxi seeking time. Compared with real data, our approach can reduce taxi seeking time by about 16.3%; Fig. 2(c) is the comparison of passenger waiting time, and we can see that our approach can reduce the passenger waiting time by approximately 20.6% based on real data. In addition, because fairness and revenue efficiency are contradictory two indicators, compared with the baseline algorithm and LSP algorithm, our TESLA approach better guarantees fairness, so the three indicators of revenue efficiency, taxi seeking time and passenger waiting time will be inferior to the baseline algorithm and LSP algorithm.

Evaluation of Overall Fairness. We also compare fairness. We firstly randomly select a rating stage ($rat = 4$), then draw distribution maps of the revenue efficiency of all taxis in the rating phase based on real data, baseline algorithm, LSP algorithm and our TESLA approach, the result is shown in Fig. 3. We can find that the distribution of revenue efficiency based on our approach is more concentrated. In addition, we also calculated the fairness (F value) of taxis at each rating stage, as shown in Fig. 2(d). It can be seen that compared to other algorithms, our approach can greatly reduce the variance of revenue efficiency (F value), thus ensuring the income fairness.

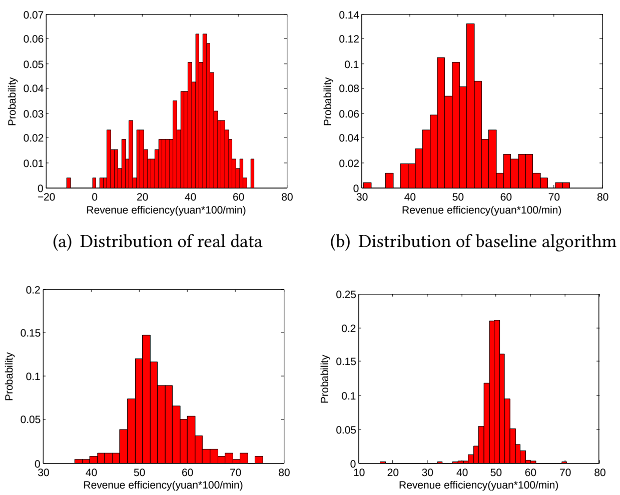
As mentioned above, fairness and revenue efficiency are two contradictory indicators, because our approach guarantees fairness, compared with the baseline algorithm, it also may lead to a higher overall cost. Due to the constraints of fairness, there may be a case that a taxi is closer to a passenger but it is not assigned, because it has lower priority. Fig. 4 shows two examples of this case.

Evaluation of route recommendation algorithm. Next, we also verify the effectiveness of our proposed route recommendation algorithm. We randomly select two-day data for simulation, and compare the average taxi waiting for dispatching time under two cases: TESLA with and without route recommendation. The results are shown in Fig. 5. We can see that through our route recommendation, taxi waiting for dispatching time is reduced by about 37.6%,



(a) Taxi revenue efficiency (b) Taxi seeking time (c) Passenger waiting time (d) Fairness at different rating levels

Figure 2: Comparison of revenue efficiency, taxi seeking time, passenger waiting time and fairness with baselines



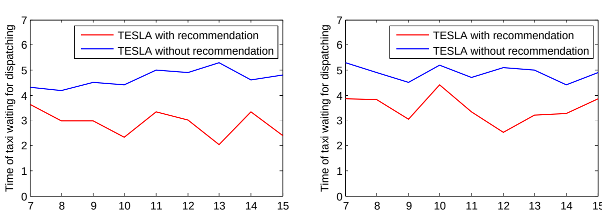
(a) Distribution of real data (b) Distribution of baseline algorithm

(c) Distribution of LSP algorithm (d) Distribution of TESLA approach

Figure 3: Distribution of revenue efficiency based on real data, baseline, LSP and TESLA approach

(a) (b)

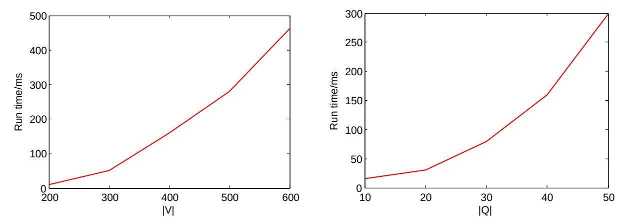
Figure 4: Examples of fairness constraints.



(a) Day 6 (b) Day 30

Figure 5: The comparison of taxi waiting time for dispatch

thus greatly improving the efficiency of centralized dispatching.
Running Time. We evaluate the running time of our approach for a round of dispatching by changing the number of taxis and ride requests. The result is shown in Fig. 6. Results show that with 600 vehicles and 50 requests at each round, the algorithm can finish the dispatching within 0.5 seconds. At the same time, it can be seen from Fig. **??** that the number of passengers per hour is about 2000-3000 in our dataset. Therefore technically our approach can do dispatching per second with such settings. The proposed solution is efficient and scalable for a single taxi company of this size.



(a) Runtime of varying $|V|$ (b) Runtime of varying $|Q|$

Figure 6: Runtime analysis

Empirical conclusion. Through the above experiments, it can be proved that the proposed TESLA approach can largely guarantee the revenue fairness among taxis compared with the baseline algorithm and LSP algorithm. And compared with the real taxi operation data, it can increase the revenue efficiency by about 8.01%, reduce the taxi seeking time by about 16.3%, and reduce the passenger's waiting time by about 20.6%. Moreover, our proposed route recommendation algorithm can reduce the time of taxi waiting for dispatching by about 37.6%, so as to ensure the efficient operation of the dispatch algorithm. In addition, in the case of a reasonable number of taxis and passengers, our approach can process the requests efficiently.

5 CONCLUSION

In this paper, we proposed a centralized taxi dispatch approach to optimizing revenue efficiency with global fairness (TESLA). In order to ensure the fairness of income between taxis, the approach dynamically calculates the priority of each taxi, and then selects the taxi to participate in the matching process according to the priority. In addition, for those taxis without matching passengers temporarily, we presented a recommendation strategy to make them match passengers more quickly as part of the TESLA approach. Finally, we conducted an extensive set of experiments and analysis, where the results suggest that our approach can improve the revenue efficiency of taxi drivers, reduce the waiting time for both taxi drivers and passengers, and guarantee revenue fairness of taxis drivers to a great extent. The approach is also computationally efficient.

REFERENCES

[1] Favyen Bastani, Yan Huang, Xing Xie, and Jason W. Powell. 2011. A greener transportation mode:flexible routes discovery from GPS trajectory data. In *ACM Sigspatial International Symposium on Advances in Geographic Information Systems, Acm-Gis 2011, November 1-4, 2011, Chicago, Il, Usa, Proceedings*. 405–408.

[2] C. Custer 2018. Unauthorized app allows Didi taxi drivers to cheat. https://www.techinasia.com/unauthorized-app-didi-drivers-cheat.

[3] Guang Dai, Jianbin Huang, Stephen Manko Wambura, and Heli Sun. 2017. A Balanced Assignment Mechanism for Online Taxi Recommendation. In *IEEE International Conference on Mobile Data Management*. 102–111.

[4] Yong Ge, Hui Xiong, Alexander Tuzhilin, Keli Xiao, Marco Gruteser, and Michael Pazzani. 2010. An energy-efficient mobile recommender system. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, Dc, Usa, July*. 899–908.

[5] S. Ma, Y. Zheng, and O. Wolfson. 2015. Real-Time City-Scale Taxi Ridesharing. *IEEE Transactions on Knowledge and Data Engineering* 27, 7 (July 2015), 1782–1795. https://doi.org/10.1109/TKDE.2014.2334313

[6] James Munkres. 1957. Algorithms for the Assignment and Transportation Problems. *J. Soc. Indust. Appl. Math.* 5, 1 (1957), 32–38.

[7] Huigui Rong, Xun Zhou, Chang Yang, Zubair Shafiq, and Alex Liu. 2016. The Rich and the Poor: A Markov Decision Process Approach to Optimizing Taxi Driver Revenue Efficiency. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16)*. 2329–2334.

[8] Yongxin Tong, Yuxiang Zeng, Zimu Zhou, Lei Chen, Jieping Ye, and Ke Xu. 2018. A unified approach to route planning for shared mobility. *Proceedings of the VLDB Endowment* 11, 11 (2018), 1633–1646.

[9] R. Wang, C. Chow, Y. Lyu, V. C. S. Lee, S. Kwong, Y. Li, and J. Zeng. 2018. TaxiRec: Recommending Road Clusters to Taxi Drivers Using Ranking-Based Extreme Learning Machines. *IEEE Transactions on Knowledge and Data Engineering* 30, 3 (March 2018), 585–598. https://doi.org/10.1109/TKDE.2017.2772907

[10] Zhe Xu, Zhixin Li, Qingwen Guan, Dingshui Zhang, Qiang Li, Junxiao Nan, Chunyang Liu, Wei Bian, and Jieping Ye. 2018. Large-Scale Order Dispatch in On-Demand Ride-Hailing Platforms: A Learning and Planning Approach. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. 905–913.

[11] Z. Ye, K. Xiao, Y. Ge, and Y. Deng. 2018. Applying Simulated Annealing and Parallel Computing to the Mobile Sequential Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2018), 1–1. https://doi.org/10.1109/TKDE.2018.2827047

[12] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie. 2013. T-Finder: A Recommender System for Finding Passengers and Vacant Taxis. *IEEE Transactions on Knowledge and Data Engineering* 25, 10 (Oct 2013), 2390–2403. https://doi.org/10.1109/TKDE.2012.153

[13] Xianyuan Zhan, Xinwu Qian, and Satish V. Ukkusuri. 2016. A Graph-Based Approach to Measuring the Efficiency of an Urban Taxi Service System. *IEEE Transactions on Intelligent Transportation Systems* 17, 9 (2016), 2479–2489.

[14] Lingyu Zhang, Wei Ai, Chuan Yuan, Yuhui Zhang, and Jieping Ye. 2018. Taxi or Hitchhiking: Predicting Passenger's Preferred Service on Ride Sharing Platforms. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 1041–1044.

[15] Lingyu Zhang, Tao Hu, Yue Min, Guobin Wu, Junying Zhang, Pengcheng Feng, Pinghua Gong, and Jieping Ye. 2017. A Taxi Order Dispatch Model based On Combinatorial Optimization. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2151–2159.

[16] Huanyang Zheng and Jie Wu. 2017. Online to Offline Business: Urban Taxi Dispatching with Passenger-Driver Matching Stability. In *IEEE International Conference on Distributed Computing Systems*. 816–825.

[17] Xun Zhou, Huigui Rong, Chang Yang, Qun Zhang, Amin Vahedian Khezerlou, Hui Zheng, M Zubair Shafiq, and Alex X Liu. 2018. Optimizing Taxi Driver Profit Efficiency: A Spatial Network-based Markov Decision Process Approach. *IEEE Transactions on Big Data* (2018).