FedSM: A Practical Federated Shared Mobility System

Shuyue Wei
SKLCCSE Lab, FBPC Beijing Advanced Innovation Center, Beihang University
weishuyue@buaa.edu.cn

Yuanyuan Zhang
North China Institute of Computing Technology
zyy-buaa@buaa.edu.cn

Zimu Zhou
School of Data Science, City University of Hong Kong
zimuzhou@cityu.edu.hk

Tianlong Zhang
SKLCCSE Lab, Beihang University
tianlong@buaa.edu.cn

Ke Xu
SKLCCSE Lab, Beihang University
kexu@buaa.edu.cn

ABSTRACT

Shared mobility leverages under-utilized vehicles to offer on-demand transport services by sharing vehicles among users. It strives to match supply with demand via a series of data-intensive operations such as supply prediction and task assignment. However, its full potential is often compromised in practice as most shared mobility platforms operate in isolation, leading to sub-optimal resource utilization. In this demonstration, we advocate a federated approach to shared mobility, which enhances its effectiveness by enabling optimizations across platforms while retaining their autonomy. We develop privacy-preserving operators and incentive mechanisms dedicated to supply prediction and task assignment in shared mobility and implement generic interfaces that support diverse prediction and assignment algorithms. We showcase the shared mobility system with real-world ride-hailing applications.

PVLDB Reference Format:
Shuyue Wei, Yuanyuan Zhang, Zimu Zhou, Tianlong Zhang, Ke Xu. FedSM: A Practical Federated Shared Mobility System. PVLDB, 17(12): 4445 - 4448, 2024.
doi:10.14778/3685800.3685896

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/AlexMufeng/Federated-Shared-Mobility.

1 INTRODUCTION

The shared use of a vehicle, bicycle, or other under-utilized transport resources for on-demand services, known as *shared mobility*, has stimulated various emerging urban services such as ride-hailing, food delivery, and parcel delivery [9]. The key is to maximize the matching between workers (*e.g.*, idle drivers) and tasks (*e.g.*, orders from passengers), which often arrive dynamically and must be handled in real-time, via a pipeline of optimizations, such as *supply prediction* [3], and *task assignment* [11]. Despite various existing shared mobility platforms (*e.g.*, Didi, Uber, and Lyft) within the same city or region, they typically operate in isolation without sharing the global information about their own workers and tasks, leading to sub-optimal resource utilization from a more global landscape, as illustrated by the toy example below.

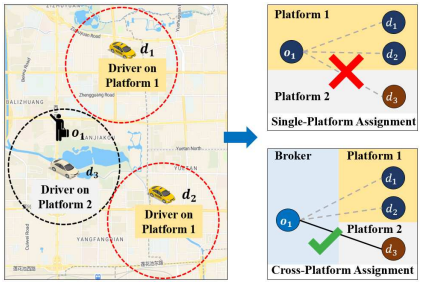

Figure 1: Motivating example for federated shared mobility.

Example 1. In ride-hailing depicted as Fig. 1, there is one task (riding order) o_1 from platform 1 and three workers (drivers) (d_1, d_2 from platform 1 and d_3 from platform 2), with task assignment modeled as bipartite matching. Consider two assignment settings. *(i) Single-Platform Assignment.* Platforms operate independently, meaning an order (o_1) placed on platform 1 can only be assigned to drivers on platform 1. In this example, drivers on platform 1 (d_1 and d_2), are too far away to pick up o_1 in time. Therefore, order o_1 is canceled once waiting time expires, despite being near the idle driver (d_3) on platform 2. *(ii) Cross-Platform Assignment.* Orders are placed through a unified interface, enabling assignments across multiple platforms. In this setting, the nearby idle driver (d_3) on platform 2 can respond to order o_1.

The example above illustrates the benefits of *cross-platform* optimization in shared mobility applications. However, realizing cross-platform services encounters three practical challenges. *(i) Privacy Protection*: Shared mobility platforms possess datasets with sensitive information, such as locations and travel histories. Unrestricted sharing of such data across platforms is prohibited by regulations like GDPR [10]. Privacy-preserving techniques dedicated to the core functionalities in shared mobility are crucial for cross-platform optimization without compromising user privacy. *(ii) Incentive Mechanisms*: Platforms may hesitate to share their valuable data without fair compensation based on their contributions. Establishing a mechanism to accurately quantify the contributions of various platforms is vital for fostering data sharing across platforms. *(iii) Transparent Interfaces*: Shared mobility platforms may adopt diverse algorithms for the core operations such as supply

Proceedings of the VLDB Endowment, Vol. 17, No. 12 ISSN 2150-8097.
doi:10.14778/3685800.3685896

prediction and task assignment, each catered for specific contexts. A practical cross-platform system should be agnostic to these core algorithms to accommodate diverse real-world applications.

We demonstrate the feasibility to fulfil the above requirements via a practical Federated Shared Mobility (FedSM) system. It has the following features. *(i)* FedSM provides privacy-preserving operators to support federated supply prediction and task assignment. Specifically, we employ a random masker to aggregate the supply predictions from individual shared mobility platforms, and develop a location encoder together with a utility obfuscator to protect the privacy of workers. *(ii)* FedSM adopts a Shapley Value [6, 7] based metric to measure the contributions of each shared mobility platform in both supply prediction and in task assignment. Various incentives and rewards schemes can be designed upon the contribution assessments to promote cross-platform collaboration. *(iii)* FedSM implements unified interfaces to accommodate diverse algorithms necessary for shared mobility.

We showcase FedSM through a cross-platform ride-hailing application. FedSM provides a unified ride-hailing interface for users to seamlessly place riding orders across platforms. It also implements multiple control panels for the core functionalities of cross-platform ride-hailing, including supply prediction, task assignment, and driver pre-dispatching. The demonstration is simulated with traces from real-world ride-hailing platforms, and offers diverse interactive visualizations and interfaces for the visitors.

2 FEDSM OVERVIEW

We first introduce the core functionalities, system architecture, and operational workflow of FedSM.

Functionalities. FedSM unites multiple shared mobility platforms $\mathcal{P}_1, \mathcal{P}_2, \cdots, \mathcal{P}_n$ to provide cross-platform services via a platform broker $\mathcal{P}_{broker}$. Each platform $\mathcal{P}_k$ holds its own workers $\mathcal{W}_k$, and the broker collects all tasks $\mathcal{T}$ through the unified user interface. Take ride-hailing as an example. Platforms such as Didi, Caocao Mobility, and Shouqi Yueche can form a federation, while a map company can act as the broker for cross-platform ride-hailing. The broker performs the following functionalities. ***(i)* Federated Supply Prediction**. This functionality calculates the global predicted worker supply from all shared mobility platforms. Providing essential guidance for task assignment, the supply prediction is an fundamental issue in shared mobility [3, 5]. Since workers are decentralized across multiple platforms within the federation, the prediction of worker supply relies on the prediction results by each local platform. To obtain a global supply $\mathcal{S}_{broker}$ in a specific area a, the broker needs to aggregate all local supplies $\mathcal{S}_k$ and such aggregation should be conducted in a federated manner due to privacy issues. As a result, with federated supply aggregation, the broker can make global supply predictions, as the task data are directly accessible. ***(ii)* Federated Task Assignment**. This functionality assigns workers from multiple shared mobility platforms to tasks collected from the unified user interface. Specifically, given a set of tasks $\mathcal{T} = \{t_1, \cdots, t_{|\mathcal{T}|}\}$, a set of workers $\mathcal{W} = \cup_k \mathcal{W}_k$, and a utility function u, it aims to find assignments across platforms such that total utility is maximized: $\max \sum_{t \in \mathcal{T}, w \in \mathcal{W}} u(t, w) \cdot M_{t,w}$ where $M_{t,w}$ is 1 if task t is matched to worker w and $M_{t,w} = 0$ otherwise. $u(t, w)$ denotes the utility when t is assigned to w. In federated task assignment, information of workers from individual platforms is shared with the broker in a privacy-preserving manner.

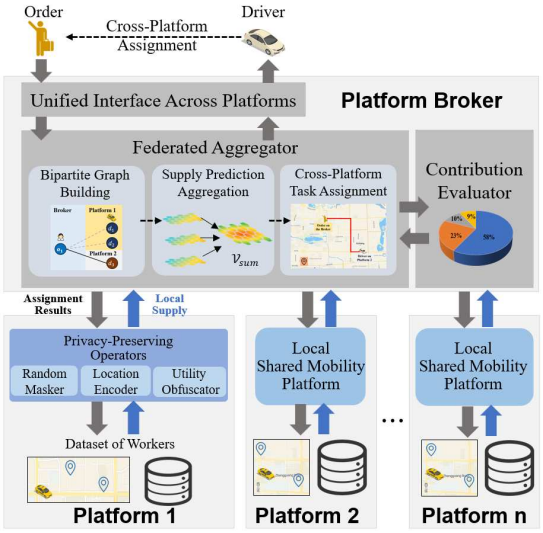

Figure 2: FedSM overview.

Architecture. Fig. 2 illustrates the system architecture of FedSM . We briefly explain the key components as follows. *(i) Local Platform Side*: Each local platform manages its own workers and provides three tailored privacy-preserving operators to support federated supply prediction and federated task assignment at the broker. *(ii) Platform Broker Side*: The broker has a unified user interface, a federated aggregator and a contribution evaluator. The user interface offers a unified way to collect tasks for all local platforms. The federated aggregator constructs a global bipartite graph using tasks from the unified interface and workers from local platforms. It then calculates the global worker supply to guide and execute cross-platform task assignments. The broker utilizes an evaluator to quantify the contributions of each local platform, and provides higher rewards to those with more significant contributions.

Workflow. The FedSM operates cross-platform transparently to users as if tasks were processed on a single platform. First, a user places a task via the unified interface. Then the broker inserts the task into a global bipartite graph, calculates the utilities between the task and workers from local platforms, and assigns the task to an appropriate worker. Finally, the broker evaluates and updates the contributions of each local platform based on the assignment results for fair compensation.

3 SYSTEM DESIGN

Next we explain the *privacy-preserving operators* and *contribution evaluator* in FedSM, the two key designs in FedSM, and how they enable federated supply prediction and task assignment.

3.1 Privacy-Preserving Operators

FedSM incorporates three privacy-preserving operators. Specifically, we employ a random masker for federated supply prediction

and develop a location encoder together with a utility obfuscator to protect the privacy of workers in federated task assignment.

Random Masker. Given n platforms where platform $\mathcal{P}_k$ holds a private value p_k, the random masker perturbs each by a random number, which allows computing $\sum_k p_k$ without revealing any private value. Specifically, the masker perturbs private value p_k as,

$$[p_k] = p_k + \sum_{k'<k} PRG(sd_{k,k'}) - \sum_{k'>k} PRG(sd_{k',k}), \tag{1}$$

where $PRG(\cdot)$ is the pseudo-random generator and $sd_{k,k'}$ is the corresponding random seed determined by negotiation between platform k and k'. As a result, the sum of all masked values equals that of the original values, *i.e.*, $\sum_k [p_k] = \sum_k p_k$.

Location Encoder. As we only use the connectivity in task assignment, we use a LSH based location encoder [4] to preserve the proximity relationships among some points without disclosing exact locations. Given a set of points $O = \{o_1, \cdots, o_n\}$, the location encoder maps each point to an array by LSH, such that points nearby are mapped to the same position with a high probability:

- $\forall o, o' \in O$ and $dist(o, o') \leq R$, $\Pr[\mathcal{E}(o) = \mathcal{E}(o')] \geq \delta$;
- $\forall o, o' \in O$ and $dist(o, o') \geq c \cdot R$, $\Pr[\mathcal{E}(o) = \mathcal{E}(o')] \leq \epsilon$,

where $\mathcal{E}(\cdot)$ is the encoding function, $dist(\cdot, \cdot)$ is distance function, and R, δ and ϵ are all thresholds with δ/ϵ expected to be large.

Laplacian Obfuscator. We adopt the Laplacian obfuscator to protects the private assignment utility in FedSM. It perturbs a private value by directly adding noise from a Laplacian distribution: $\widetilde{p} = p + Lap(\frac{\Delta p}{\beta})$, where Δp is the possible range of private value p and β denotes the privacy budget. The Laplacian obfuscator protects private scalar data that satisfies differential privacy [2].

3.2 Contribution Evaluators

FedSM utilizes a contribution evaluator to assess the contributions in a federation with n local platforms $\mathcal{P}_N = \{\mathcal{P}_1, \cdots, \mathcal{P}_n\}$. Specifically, we adopt the *Shapley Value* to measure the contribution:

$$\phi_k = \sum_{S \subseteq \mathcal{P}_N \setminus \{\mathcal{P}_k\}} \frac{U(\mathcal{P}_S \cup \{\mathcal{P}_k\}) - U(\mathcal{P}_S)}{n \cdot \binom{n-1}{|S|}} \tag{2}$$

where ϕ_k is the contribution of $\mathcal{P}_k$ and $U(\mathcal{P}_S)$ measures the rewards among a set of local platforms $\mathcal{P}_S$. We can accomplish task-specific contribution evaluation by defining specific reward function $U(\cdot)$. The contribution metric provides the basic fairness guarantee, such as: *(i)* A platform who has no impact on rewards in any federation has zero contribution; *(ii)* If two platforms can substitute for each other in any federation, they contribute equally. Thus, we can incentive platforms to cooperate actively through fair compensation.

3.3 Actions at Platform Broker

We introduce how our FedSM utilizes the above privacy-preserving operators and the contribution metric in its functionalities. *(i) Actions for Federated Supply Prediction*: As mentioned, the local platform $\mathcal{P}_k$ possesses a predicted local supply $\mathcal{S}_k$ for various areas, which can be perturbed using the random masker to $[\mathcal{S}_k]$. By summing these masked values from all platforms, we derive the global supply prediction $\mathcal{S}_{broker} = \sum_k [\mathcal{S}_k] = \sum_k \mathcal{S}_k$. The system employs the sum of local supplies from a subset of platforms $\mathcal{P}_S$ as the reward function $U(\mathcal{P}_S)$ and then we can assess contributions of each platform in federated supply prediction. *(ii) Actions for Federated Task Assignment*: Firstly, the broker employs the location encoder to determine the connectivity between tasks and workers without revealing their locations. Specifically, if a task and a worker are mapped to the same position by the location encoder, it indicates that the task can be assigned to that worker. As the assignment utility is a scalar value, it can be computed by the utility obfuscator in a privacy-preserving manner. After that, the system establishes a global bipartite graph where tasks from the broker and workers from multiple platforms can be matched through task assignment algorithms. For contribution evaluation, the broker uses the maximum total utility achieved with workers from different sets of platforms as the reward functions. Then, the evaluator can assess the contributions of each platform in federated task assignment.

4 DEMONSTRATION

We focus on cross-platform ride-hailing application and demonstrate the scenario of the FedSM from both the user and broker side. As next, we first introduce the used spatial-temporal dataset and the simulation environment and then describe our demonstration.

Simulation Environments. We take the real-world ride-hailing dataset from Didi's GAYA initiative [1], which holds the order record data and driver trajectory data from Chengdu China. We employ a batch-based assignment framework with a time interval of 2 seconds, which is commonly adopted in real-world applications. For drivers, there can be both online and offline. When online, they may have various behaviors including pick-up, delivery, idle, or random work. The local platforms independently generate local drivers, while the platform broker focuses on generating cross-platform orders. As next, we mainly demonstrate how a ride-hailing order is requested by the user from the interface of platform broker.

User Side: Cross-Platform Ride-Hailing. Fig. 3 shows the scenario in which a user requests a car through the user-friendly interface on the platform broker: ① In this panel, the user specifies the starting and ending points of his/her ride-hailing request. Initially, the system offers a recommended pickup point based on the user's current location. ② This panel provides multiple service options from various ride-hailing platforms and each has a visible price. Users can freely select their preferred taxi companies based on the price and service quality. ③ Finally, by pressing the call-to-action button, the user raises a ride request and the broker will assign an available driver across multiple platforms to fulfill the request.

Platform Side: Federated Order Dispatching. Our demonstration consists of three essential functions, including *federated supply prediction*, *federated task assignment* and *federated drivers predispatching*. We introduce each of them as follows.
(i) Federated Supply Prediction (in Fig. 4): ① The platform broker exhibits heatmaps for both predicted drivers across various areas on the real-world map, where the distribution of drivers can be viewed as guidance for task assignment and driver dispatching. ② This panel displays the percentage of drivers who are currently occupied by orders. ③ The platform broker evaluates the contributions of various local platforms during certain time interval, using

[1] https://outreach.didichuxing.com/research/opendata/

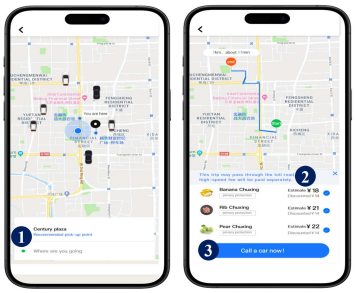

Figure 3: User interface for cross-platform ride-hailing.

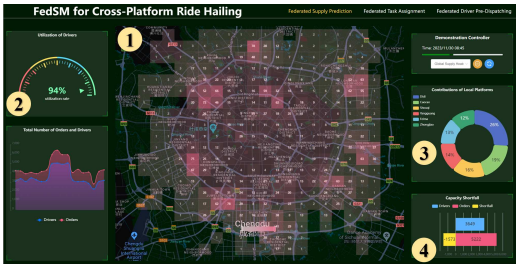

Figure 4: Federated supply prediction.

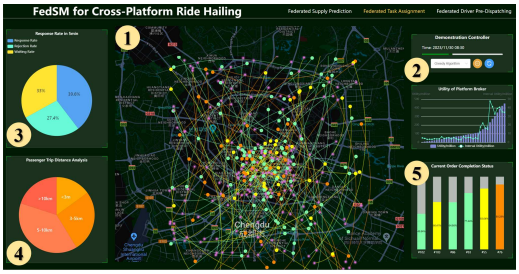

Figure 5: Federated task assignment.

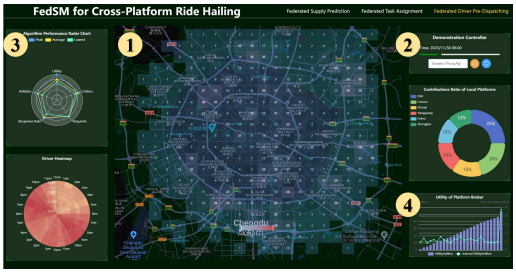

Figure 6: Federated driver pre-dispatching.

the Shapley value based metric for federated supply prediction as described in Section 3.3. ④ The FedSM also highlights the disparity between supply and demand for the cross-platform ride-hailing.

(ii) Federated Task Assignment (in Fig. 5): It assigns drivers to orders across multiple ride-hailing platforms. ① This panel visualizes the real-time assignment process between orders and drivers, where each line depicted denotes a route between starting and ending points, while distinct colors denote drivers affiliated with different taxi companies. ② This panel features a demonstration controller for simulation during different time intervals and provides various algorithms for task assignment including the greedy algorithm, the classical Kuhn-Munkres algorithm [1] and advanced learning to dispatch algorithm [8]. ③ There is a pie chart details response rate, rejection rate and waiting rate within a 5 minutes, which indicates the satisfaction levels of both passengers and drivers. ④ Similarly, the trip distance is analyzed and depicted in a pie chart, offering a snapshot of travel patterns. ⑤ In this panel, we can observe the real-time completion percentages of current orders, which is a valuable feedback to monitor the progress of the federated assignment.

(iii) Federated Driver Pre-Dispatching (in Fig. 6): Additionally, this module primarily prepares to dispatch drivers to areas where the ride-hailing demand exceeds supply after an assignment batch. ① It further displays the market imbalances in various areas using a heatmap, allowing us to observe whether an area has an oversupply or excess demand. ② This panel provides algorithms designed to motivate drivers to move from areas with an oversupply to those experiencing high demand. ③ There is a radar chart presenting the performance of the dispatch algorithm across multiple dimensions (*e.g.*, the response rate, total orders and the utility), which aids in understanding an algorithm and providing valuable insights for optimization. ④ The system also visualizes the improvements in utility gained through federated driver dispatching over time, including both the total utility and the interval utility.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their suggestions. This work was partially supported by Key Research and Development Program of China under Grant No. 2023YFF0725103, National Science Foundation of China (NSFC) (Grant Nos. U21A20516, 62336003) and Beijing Natural Science Foundation (Z230001), the Basic Research Funding in Beihang University No.YWF-22-L-531, and Didi Collaborative Research Program NO2231122-00047. Zimu Zhou's research is supported by Chow Sang Sang Group Research Fund No. 9229139. Yuanyuan Zhang is the corresponding author.

REFERENCES

[1] Thomas Cormen, Charles Leiserson, Ronald Rivest, et al. 2009. *Introduction to algorithms*. MIT press.
[2] Cynthia Dwork. 2006. Differential privacy. In *ICALP (Lecture Notes in Computer Science)*, Vol. 4052. Springer, 1–12.
[3] Xu Geng, Yaguang Li, Leye Wang, et al. 2019. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *AAAI*. 3656–3663.
[4] Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*. ACM, 604–613.
[5] Xiaochuan Lin, Kaimin Wei, Zhetao Li, et al. 2024. Aggregation-based dual heterogeneous task allocation in spatial crowdsourcing. *FCS* 18, 6 (2024).
[6] Lloyd S Shapley. 1953. A value for n-person games. (1953).
[7] Tianshu Song, Yongxin Tong, and Shuyue Wei. 2019. Profit allocation for federated learning. In *BigData*. IEEE, 2577–2586.
[8] Yongxin Tong, Dingyuan Shi, Yi Xu, et al. 2023. Combinatorial optimization meets reinforcement learning: effective taxi order dispatching at large-scale. *TKDE* 35, 10 (2023), 9812–9823.
[9] Yongxin Tong, Yuxiang Zeng, Zimu Zhou, et al. 2022. Unified route planning for shared mobility: an insertion-based framework. *TODS* 47, 1 (2022), 2:1–2:48.
[10] Paul Voigt et al. 2017. The EU general data protection regulation. *SIP* (2017).
[11] Yansheng Wang, Yongxin Tong, et al. 2022. Fed-LTD: Towards cross-platform ride hailing via federated learning to dispatch. In *KDD*. ACM, 4079–4089.