https://doi.org/10.1016/j.trb.2021.06.014

A MEAN-FIELD MARKOV DECISION PROCESS MODEL FOR SPATIAL-TEMPORAL SUBSIDIES IN RIDE-SOURCING MARKETS

Zheng Zhu, zhuzheng@ust.hk

Department of Civil and Environmental Engineering, Hong Kong University of Science and Technology, Hong Kong, China

Jintao Ke, jke@connect.ust.hk*

Department of Logistics and Maritime Studies, Hong Kong Polytechnic University, Hong Kong, China

Hai Wang, haiwang@cmu.edu

School of Computing and Information Systems, Singapore Management University, Singapore

Heinz College of Information Systems and Public Policy, Carnegie Mellon University, Pennsylvania, USA

**Corresponding Author*

ABSTRACT

Ride-sourcing services are increasingly popular because of their ability to accommodate on-demand travel needs. A critical issue faced by ride-sourcing platforms is the supply-demand imbalance, as a result of which drivers may spend substantial time on idle cruising and picking up remote passengers. Some platforms attempt to mitigate the imbalance by providing relocation guidance for idle drivers—who may have their own self-relocation strategies and decline to follow the suggestions. Platforms then seek to induce drivers to system-desirable locations by offering them subsidies. This paper proposes a mean-field Markov decision process (MF-MDP) model to depict the dynamics in ride-sourcing markets with mixed agents, whereby the platform aims to optimize some objectives from a system perspective using spatial-temporal subsidies with predefined subsidy rates, and a number of drivers aim to maximize their income by following certain self-relocation strategies. To solve the model more efficiently, we further develop a representative-agent reinforcement learning algorithm that uses a representative driver to model the decision-making process of multiple drivers. This approach is shown to achieve significant computational advantages, faster convergence, and better performance. Using case studies, we demonstrate that by providing some spatial-temporal subsidies, the platform is well able to balance a short-term objective of maximizing immediate revenue and a long-term objective of maximizing service rate, while drivers can earn higher income.

Keywords: ride-sourcing, subsidy, mean-field, Markov decision process, mixed agents

1. BACKGROUND

The emergence of advanced information technologies and the surge in smartphone users enable the fast development of ride-sourcing services. Provided by transportation network companies (TNCs), such as Uber, Lyft, DiDi, and Grab, ride-sourcing services address individuals' on-demand travel needs. A ride-sourcing market is analogous to a more efficient dial-hailing taxi market, in which passengers request services with a few clicks in smartphone apps. Unlike traditional taxi services with large meeting frictions due to street-hailing behaviors between drivers and passengers, ride-sourcing services enable passengers to be matched with drivers at a certain distance. Upon receiving a travel request from a passenger, the platform assigns the passenger to a near driver who then picks up and delivers the passenger. On one hand, the efficiency of supply-demand matching makes ride-sourcing systems indispensable in modern transportation systems. On the other hand, drivers may spend significant amounts of time on idle cruising[1] (IC; i.e., waiting for dispatches) and on the way to pick up passengers. A market failure, called a "wild-goose chase" (WGC), even occurs when drivers are always dispatched to far-away passengers, and waste substantial time on picking them up. These lead to low effective earning rates of drivers and cause negative social externalities, such as exacerbating traffic congestion and increasing carbon dioxide emissions.

The main cause of inefficient IC and the WGC phenomenon is the spatial-temporal supply-demand imbalance. If there is a lack of idle drivers in one region, the platform must call remote drivers to enter the region to mitigate the loss of passengers and revenue. However, these drivers may suffer from long pick-up time (i.e., as in WGC). By contrast, if there are insufficient passengers in one region, drivers may suffer from long idle time (i.e., as in IC). To tackle the issue of supply-demand imbalance, a number of approaches have been proposed, including but not limited to order dispatching (Xu et al., 2018; Yang et al., 2020a) and surge pricing (Zha et al., 2018). In particular, with the fast development of computational power and artificial intelligence technologies, researchers are paying increasing attention to the design and optimization of idle-vehicle relocation strategies for improving supply-demand balance (Rong et al., 2016; Yu et al., 2019; Lin et al., 2018).

In practice, based on actual or predicted spatial-temporal information on supply/demand (Ke et al., 2019) and traffic conditions (Zhu et al., 2019a), idle drivers are advised/incentivized to cruise to regions with higher potential rewards. These rewards could be reflected by the saving on waiting/matching time (Hwang et al., 2015); increase in trip fares and income (Rong et al., 2016; Shou et al., 2020a); increase in vehicle occupancy/utilization rate (Gao et al., 2018); and saving on idle-cruise distance and operational costs (Lin et al., 2018; Yu et al., 2019). These studies aim to generate optimal sequential movements for idle drivers to achieve some maximal system-wide total rewards over a time horizon. Dynamic gaming approaches, such as the Markov decision process (MDP), offer a convenient framework for formulating and solving these problems. In an MDP model, one or multiple players (also referred to as agents) interact with an environment. Each agent has a set of states and a set of actions. In each time slot, each agent chooses one action after it perceives the current state. Meanwhile, by taking an action, the agent receives a reward and their state will be updated by the current state, action, and the state transition law, moving to the next state. During a time horizon, agents attempt to seek out the optimal sequence of actions (determined by a policy that maps the current state to the action) that leads to maximal total rewards. In particular, an MDP model with multiple agents is referred to as a multi-agent MDP model.

[1] We use "idle cruising" because in ride-sourcing markets some vacant vehicles are en route to pick up passengers. To distinguish this from traditional taxi markets, we note that these vehicles are vacant but not idle.

Although MDP-based approaches for idle-vehicle relocation have been established in recent studies, research gaps remain. For instance, none of the previous studies have examined the designs and analysis of spatial-temporal subsidies for ride-sourcing drivers with their own relocation strategies using an MDP framework. To be more specific, each driver aims to maximize their own earning by relocating to profitable regions, while the platform tries to incentivize drivers' relocating behaviors to maximize overall system efficiency by subsidies. Clearly, the sequential decision-making of drivers and the platform interact with each other, resulting in a very complex multi-agent MDP with different (i.e., mixed) types of agents. To well formulate such a complex system, we propose a *mean-field (MF)-MDP model*, which can jointly analyze the platform's spatial-temporal subsidies and idle drivers' self-relocation strategies. We regard the platform as a major agent that pursues the subsidy to optimize some objectives from a system perspective—e.g., to maximize immediate revenue and/or the number of passengers served (service rate). A number of drivers are considered as minor agents who choose their self-relocation strategies to maximize their income. The decisions of the platform directly affect the income and decisions of the drivers, while the decisions of the drivers, in turn, collectively affect the platform's decisions via their average status (e.g., the spatial-temporal distribution of idle drivers), which is captured by the MF state. By using a simple stochastic process to approximate the MF state (instead of computing it based on each driver's state), we are able to reduce the standard multi-agent MF-MDP model to a simplified MF-MDP model with only the platform and one representative driver as agents. We then develop a representative-agent reinforcement learning algorithm to solve the simplified model. We conduct a set of numerical studies to examine the performance of the proposed representative-agent algorithm. By performing sensitivity analysis, we further investigate the impacts of spatial-temporal subsidies on drivers' self-relocation, drivers' income, the number of passengers served, and the platform's net revenue. The results suggest that by providing some spatial-temporal subsidies, the platform is able to achieve a higher total reward, while drivers can earn higher income.

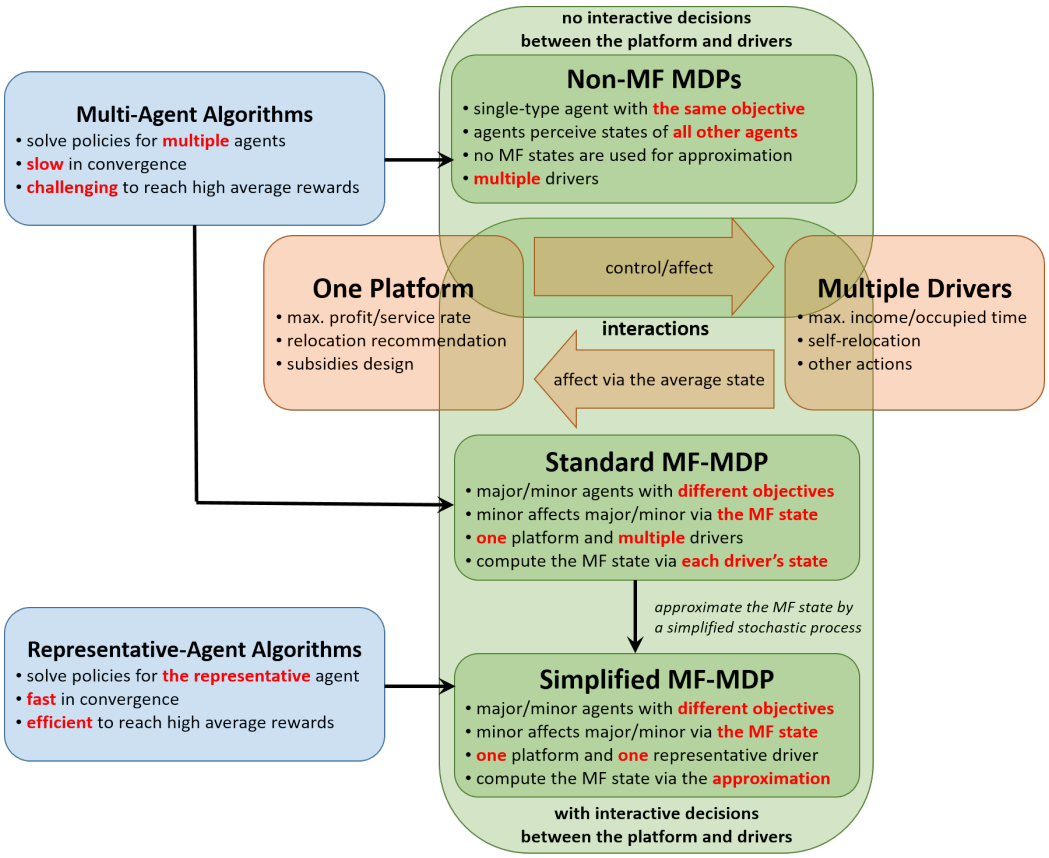


Figure 1. Summary of this paper's contributions

We use the term *non-MF-MDPs* to denote MDPs in which the MF state of minor agents is not required to compute the dynamics (e.g., transition laws and states of agents) in the environment. The main distinctions between the proposed MF-MDP model and other non-MF-MDP models, and the features of their targeting research problems are summarized in Figure 1.

In a non-MF-MDP model, each agent makes decisions by perceiving the states of all other agents, which may render the algorithm hard to converge due to the high stochasticity and instability of the environment. In an MF-MDP model, the states of agents are averaged in each zone and each time interval, and each agent makes its decisions according to the averaged (mean-field) state. This will help reduce the variance of the states and actions, and thus make the model easy to be trained. In describing a ride-sourcing system with one platform and multiple drivers, both non-MF-MDP models and standard MF-MDP models contain multiple agents (the platform and drivers). Naturally, these two models can be solved by multi-agent algorithms that treat each driver and the platform as an independent agent. The only difference is that agents in MF-MDP models could take the MF states as inputs for making actions, while non-MF-MDP models should be aware of the states of all other agents at each decision point.

Additionally, in a complex system with a large number of drivers as agents, multi-agent algorithms need to identify the optimal policy for each specific agent, and the underlying solution space (i.e., the Cartesian product of each agent's state-action set) could be so large that optimal strategies are hard to be identify. To address this critical issue, we then propose a simplified MF-MDP model that uses a representative driver to make decisions for all independent drivers. In other words, the simplified MF-MDP model only identify optimal policies for the representative driver (minor player) and the platform (major player), resulting in a much smaller solution space.

By developing representative-agent solution algorithms, it could be much easier for simplified MF-MDP models to fast convergence to high-rewarding policies.

The main contributions of this paper are:

- We propose a generalized MF-MDP model to capture the interactive decisions between the platform and a group of drivers with different objectives in ride-sourcing markets; in contrast, previous studies in this domain generally assume that the platform has full control of ride-sourcing drivers or that the platform and drivers have the same objective.
- We show theoretically that this generalized multi-agent MF-MDP model (also referred to as the standard MF-MDP) can be approximated by a simplified MF-MDP model that attempts to jointly identify the optimal policies of a platform and a representative driver. The simplified MF-MDP model offers computational advantages for solving multi-agent MDP models.
- We formulate a specific MF-MDP model to design spatial-temporal subsidies with predefined subsidy rates for drivers with self-relocation strategies. A representative-agent reinforcement learning algorithm is developed to solve the simplified MF-MDP model. Numerical studies demonstrate the effectiveness of the proposed algorithms in a small market, and examine the influences of spatial-temporal subsidies on a few key measures.

The rest of the paper is organized as follows. Section 2 reviews the literature on ride-sourcing markets and, in particular, idle-vehicle relocation. Section 3 presents the generalized ride-sourcing MF-MDP model with the platform and drivers as mixed agents. We discuss the approximation of the MF state and the simplified MF-MDP model with theoretical properties and a dynamic programming approach. In Section 4, we adopt the proposed MF-MDP model and formulate the spatial-temporal subsidy problem. A representative-agent reinforcement learning algorithm is developed. We conduct a set of numerical studies in Section 5 and demonstrate the advantage of the representative-agent algorithm over conventional multi-agent algorithms and the potential impacts of the subsidies on the platform and drivers. In Section 6, we discuss different potential subsidy schemes. Section 7 concludes.

2. LITERATURE REVIEW

With the development and deployment of smartphone and information technologies, ride-sourcing services have had substantial impacts on traditional taxis in terms of passengers' mode choices and mobility efficiency, and therefore have received intensive attention from researchers across fields. General research problems include optimal operating strategy designs in terms of the trip fares charged to passengers and wages paid to drivers (Cachon et al., 2015; Castillo et al., 2017; Zha et al., 2016; Bai et al., 2019; Taylor, 2018; Yang et al. 2020b); implications of governmental policies and regulations (Yu et al., 2019); examination of the elasticities of labor supply with respect to driver income level (Sun et al., 2019a; Sun et al., 2019b); on-demand matching and dispatching strategies (Xu et al., 2017; Zha et al., 2018; Zhang et al., 2017; Lyu et al. 2019; Yang et al. 2020a); forecasting real-time demand and supply (Ke et al., 2017; Ke et al., 2021; Yao et al., 2018; Zhu et al., 2021b); equilibrium in ride-pooling services (Ke et al., 2020); and the impact of ride-sourcing on public transit (Zhu et al., 2020). Readers may refer to Wang and Yang (2019) for a comprehensive review.

One critical problem faced by ride-sourcing platforms is how to mitigate supply-demand imbalance over space and time, which is commonly observed due to the stochastic arrivals and heterogeneous distributions of both drivers and passengers. The supply-demand imbalance can be alleviated with the help of approaches such as spatial-temporal demand prediction (Ke et al., 2021); fleet-size regulation (Yang et al., 2002; Lin et al., 2018); surge/spatial pricing and rewards

(Yang et al., 2010; Zha et al., 2018; Zuniga Garcia, 2019; Yang et al. 2020b); driver incentive/subsidy (Qian et al., 2017); efficient large-scale order dispatch (Xu et al., 2018; Li et al., 2019); and idle-vehicle relocation guidance (Rong et al., 2016; Yu et al., 2019).

Of these methods, idle-vehicle relocation, which guides or incentivizes idle vehicles from regions with extra supply to locate to regions with inadequate supply, is attracting substantial attention. Braverman et al. (2019) propose a fluid-based optimization approach that controls the flow of empty vehicles to optimize system-wide network utility, measured by the availability of idle vehicles upon passenger arrivals. They show that the optimal utility obtained from a fluid-based approach is an upper bound on the utility of a system with finite vehicles for any routing policy. Lin et al. (2018) propose a multi-agent deep reinforcement learning approach that controls the movements of idle vehicles. Using mobility data from DiDi, they show that their proposed multi-agent model significantly outperforms benchmark algorithms. In this study, the multi-agent advanced actor-critic (A2C) algorithm shows its ability to solve large-scale multi-agent reinforcement learning problems based on a simulator calibrated by actual mobility data. Most studies on idle-vehicle relocation assume that the platform has full control of drivers/vehicles (e.g., Rong et al., 2016; Lin et al., 2018; Shou et al., 2020b). In reality, however, the ride-sourcing platform and drivers have different objectives: Drivers aim to maximize their individual rewards (measured by income, vehicle occupancy rate, etc.) by following certain self-relocation strategies, while the platform aims to maximize overall system performances (measured by net revenue, saving on matching times, number of passengers served, etc.) by using spatial-temporal subsidy/guidance strategies. In this manner, incentives (e.g., subsidies or other rewards to drivers) are critical to motivate drivers to move from demand-cool locations (with more supply than demand) to demand-hot locations (with more demand than supply). Although subsidies/incentives strategies have been implemented in some ride-sourcing companies, such as DiDi, they have not been fully examined in the literature, particularly in a MDP framework. Shou and Di (2020a) propose a multi-agent reinforcement learning paradigm to approximate the system's equilibrating process in a routing game among atomic selfish agents on a network. Sous and Di (2020b) examine reward design scenarios with multiple drivers and a constant design across zones and time periods, in which the Bayesian optimization is adopted to find the optimal design strategy. Their model can help policymakers develop optimal operational and planning countermeasures under different environments. The two studies also consider mean-field approximation within the reinforcement learning algorithm only; in contrast, our model is a mean-field "oriented" that builds the ride-sourcing simulation based on mean-field information.

From a modeling perspective, the difficulty in mitigating the supply-demand imbalance in ride-sourcing markets lies in the complicated dynamic decision processes of the platform, drivers, and passengers, as well as endogenous relationships between decisions and scenarios. Specifically, the platform's strategies, such as order dispatching, idle-vehicle relocation, and dynamic pricing/subsidies, affect both supply and demand, which in turn affect the platform's decisions. A promising option for capturing the dynamics of ride-sourcing markets is the family of MDP models, which can well describe the sequential interactions between agents and environment. For example, Xu et al. (2018) formulate an order-dispatching process for a ride-sourcing system using an MDP model, with the order dispatch as action, the numbers of idle drivers and waiting passengers in each time/location as states, and the total gross merchandise volume (GMV) as reward. They propose a policy that simultaneously considers the immediate reward and long-term rewards, and demonstrate that the proposed policy based on the MDP model can substantially improve the per-day earnings of drivers. More recently, various MDP and reinforcement learning models (e.g., Wang et al., 2018; Li et al., 2019; Shou et al., 2020a; Jin et al., 2019) have been developed to enhance the supply-demand balance via better dispatching and idle-vehicle relocation strategies. However, as stated in Section 1, in a

complicated system with a huge number of drivers, it can be difficult to identify optimal policies for each specific driver. In addition, in most previous studies, drivers and the platform's objectives are not necessarily coincident with each other. While these studies assume the platform's reward is equal to the summation of the rewards of all drivers (which implies that the platform and drivers have the same objective), it is more interesting and realistic to ascertain the platform's and drivers' own policies in an environment where they mutually affect each other. To be more specific, drivers try to maximize their individual daily earning through self-relocation, while the platform attempts to maximize system-wide efficiency by paying subsidies to drivers based on location and time.

Inspired by the aforementioned studies and research gaps, we propose a generalized MF-MDP model to analyze the dynamics in ride-sourcing markets in which the platform and multiple drivers have different objectives and state-action sets. The MF-MDP model is new to transportation research questions which are solved by an MDP environment with interactive decisions between the mixed agents. According to the proposed model, we theoretically show that efficient algorithms can be developed by only considering the platform and a representative driver as agents in a simplified MF-MDP model. A specific MF-MDP model and a representative-agent reinforcement learning algorithm are developed to analyze the implications of spatial-temporal subsidies for drivers with self-relocation strategies. Our numerical results offer insights on the interactions between the platform's subsidy and idle drivers' self-relocation, as well as the influences of the intensity of subsidy on the platform's spatial-temporal subsidy strategy and idle drivers' self-relocation strategies.

3. Mean-field Markov decision process model for ride-sourcing markets

In this section, we present a generalized MF-MDP model for depicting the interactive decision processes of the platform and drivers in ride-sourcing markets. With generalized definitions and formulas for states, actions, the MF state, state transition laws, and rewards for mixed agents, we present some properties of the MF-MDP model. We also discuss simplification of the model to reduce the number of agents for computational advantages.

3.1 General concept of the MF-MDP model

The development of an MDP model should capture the particular feature of a research problem, which is depicted by the definition of states, actions, rewards of agents and the transition law (i.e., how the environment replies to agents' actions). In a practical problem, the number of states and actions for an agent can be large. For instance, in idle-vehicle relocation problems, a driver's state should include time and location and his/her actions may cover a list of locations/directions. Moreover, the transition law could involve complex computations that is executed based on spatial-temporal information of each agent and extra information of the environment. Given the large sets of states and actions and the complex transition for each agent, solving a multi-agent MDP model with a large number of agents results in a massive solution space and thus can be computationally prohibitive. The scenario can become more complicated when different types of agents (who may have distinct objectives) coexist in the environment, resulting in an MDP with mixed agents. To capture the interactions between a major agent and a number of minor agents who pursue their individual objectives, Huang et al. (2006) propose the concept of an MF-MDP model. In an MF-MDP model, the states and actions of the major agent can significantly affect the rewards and actions of minor agents. Meanwhile, each minor agent has a negligible impact on the rewards and actions of another minor agent or the major agent. Instead, the transitions, rewards, and actions of the major agent and a minor agent are

influenced by the mean-field (i.e., MF, average) state of all minor agents collectively (Gomes, 2014). In this manner, the major agent or a specific minor agent does not distinguish any individual minor agent in the MF-MDP model, but considers the MF state when taking actions.

In a standard MF-MDP (with one major agent and multiple minor agents), we need to compute the MF state via summarizing each minor agents' state so as to obtain the transitions and rewards. This can be computationally intractable when the number of minor agents is huge. To improve the efficiency, the standard MF-MDP can be simplified by approximating the MF state in a stochastic process and using a representative agent to determine actions for multiple minor agents with the same objective, states, and action sets (Huang et al., 2006; Huang et al., 2007). Once there are a large number of minor agents in the environment, the simplified MF-MDP can well approximate the dynamic nature of the standard MF-MDP. Also, it can significantly reduce computational complexity and achieve more efficient solution by optimizing only one policy for the representative minor agent instead of determining a group of independent policies for each of the minor agents. Correspondingly, we propose representative-agent dynamic programing/reinforcement learning algorithms to solve simplified MF-MDPs (see the next section), while conventional DP/RL algorithms are adopted to solve standard MF-MDPs and non-MF-MDPs.

Literature on the MF-MDP model (e.g., Huang et al., 2006; Huang et al., 2007) mainly focuses on the general conception, definitions, and mathematical propositions in a simple and stylized case; there is no discussion of how to configure and solve such a model when the environment is complicated. Inspired by the concept of the MF-MDP model, this paper aims to develop a MDP model that can well delineate the state-action transition laws in a system with one platform and a group of drivers whose actions mutually affect each other. At the beginning stage of MF-MDP studies, we develop a specific MF-MDP model for analyzing spatial-temporal subsidies for drivers with self-relocation strategies (see Section 4.1) and an efficient solution algorithm for the particular MF-MDP model (see Section 4.2). We demonstrate that the algorithm achieves significant computational advantages, faster convergence, and better performance on a small-scale market (see Section 5), and aim to examine the general performance on actual-size problems in near future.

3.2 Formulation of the ride-sourcing MF-MDP model

In a ride-sourcing market, the platform's operational strategies play important roles in affecting the performance (e.g., daily income, waiting time for order matches, and distances en route to pick up passengers) and decisions (e.g., self-relocation and working hours) of drivers. However, if the number of drivers is large, the impact of each individual driver's decisions and actions on the platform or other drivers is trivial and can be ignored without causing significant deviations in general. By contrast, the average (i.e., MF) state of all drivers collectively, which captures the spatial-temporal supply information, will significantly influence order matching/dispatching, performance (e.g., net revenue, vehicle occupied rate, and the number of passengers served), and other decisions (e.g., spatial-temporal pricing and subsidy) of the platform as well as those of each individual driver. Moreover, the platform sometimes chooses to display heat maps of its spatial-temporal surge pricing and/or subsidy and overall demand and supply to drivers on the app. In this manner, the state of the platform and the MF state of drivers are public information to drivers, who then process the information and take corresponding actions. Therefore, it is reasonable to describe the ride-sourcing market using an MF-MDP

model, in which the platform is regarded as the major agent and a number of drivers are treated as minor agents[2].

Suppose there is 1 platform and M homogeneous drivers (i.e., state sets, action sets, and objectives are the same for drivers), and the planning horizon consists of T time periods (i.e., time $t \in \{1,2,\dots,T\}$). Let $\boldsymbol{S}$ and $\boldsymbol{S}_d$ denote finite sets of the states of the platform and drivers, respectively. Let $y^t \in \boldsymbol{S}$ represent the state of the platform at time t; specifically, y^t can be a vector that contains time index t, the spatial-temporal pricing, subsidies, and number of waiting passengers across different regions in the market at time t. We use $y_{d,i}^t \in \boldsymbol{S}_d$ to represent the state of driver $i \in \{1,2,\dots,M\}$ at time t, which could include time index, their location, the number of loaded passengers, and the destination. Then the MF state of all drivers at any time period t can be represented as a vector $\boldsymbol{z}_d^t$ as follows:

$$\boldsymbol{z}_d^t = \left[z_{d,s_d}^t\right]_{1\times|\boldsymbol{S}_d|} \tag{1}$$

$$z_{d,s_d}^t = \frac{\sum_{i=1}^{M} \mathrm{I}\left(y_{d,i}^t = s_d\right)}{M} \tag{2}$$

where $\mathrm{I}(\cdot)$ denotes the identity function and we use $\boldsymbol{H}_d$ to denote the feasible domain of MF state $\boldsymbol{z}_d^t$, i.e., $\boldsymbol{z}_d^t \in \boldsymbol{H}_d$. Intuitively, the MF vector $\boldsymbol{z}_d^t$ represents the distribution of drivers' states. For instance, if a driver's state contains their current location and the occupancy of their vehicle, then the MF state captures the spatial distribution of all vacant vehicles and occupied vehicles.

Let $\boldsymbol{A}$ and $\boldsymbol{A}_d$ denote finite sets of the actions of the platform and drivers, respectively. We use $x^t \in \boldsymbol{A}$ and $x_{d,i}^t \in \boldsymbol{A}_d$, respectively, to denote the actions of the platform and driver i at time t. The actions of the platform can include pricing or subsidy strategies (e.g., 1 for subsidizing and 0 for not offering subsidy), and the actions of a driver are their self-relocation directions.

Following the conventions in discrete-time MDPs, the transition probability of a major or minor agent in the MF-MDP is determined by their current state and action and the MF state of the minor agents. Specifically, the state transition laws for the platform and a specific driver are denoted as $Q(\cdot \mid \cdot)$ and $Q_d(\cdot \mid \cdot)$ in Eqs. (3)–(4), where $\mathrm{P}(\cdot)$ denotes the probability operator[3].

$$Q(s'|s, \boldsymbol{h}_d, a) = \mathrm{P}(y^{t+1} = s' | y^t = s, \boldsymbol{z}_d^t = \boldsymbol{h}_d, x^t = a) \tag{3}$$

$$Q_d(s_d'|s_d, \boldsymbol{h}_d, a_d) = \mathrm{P}\left(y_{d,i}^{t+1} = s_d' | y_{d,i}^t = s_d, \boldsymbol{z}_d^t = \boldsymbol{h}_d, x_{d,i}^t = a_d\right) \tag{4}$$

The platform or a driver takes sequential actions to maximize their total rewards in T time periods, which can be measured by the net revenue, the number of passengers served, and so on. Let r denote the reward of the platform, which is a function of the platform's current state

[2] Note that in real ride-sourcing markets, market conditions have strong time-varying patterns with peak and off-peak hours, which indicate the nonstationary states and transitions in a day. However, if we consider a certain period of 2 to 3 hours, market conditions are more stable and thus can be approximately described using stationary states and transitions. Readers can refer to Figures 4 and 5 in Lyu et al. (2019) for demonstrations of daily temporal distributions of demand and supply in a real ride-sourcing market. If we consider a certain period—e.g., 8 am to 10 am during peak hours or 2 pm to 4 pm during off-peak hours—market conditions are quite stable and thus can be modeled as stationary MDP, with different transition matrices, respectively.

[3] In this paper, we use y^t and $y_{d,i}^t$ (also y_d^t) to represent random variables of states, x^t and $x_{d,i}^t$ (also x_d^t) random variables of actions, and $\boldsymbol{z}_d^t$ (also $\hat{\boldsymbol{z}}_d^t$) random variables of MF states in the MF-MDP model. We use s and s_d to represent values of random states, a and a_d values of random actions, and $\boldsymbol{h}_d$ values of random MF states.

and action and the MF state of drivers. For a particular driver, the reward r_d could be measured by their income, saving on idle-cruise distance, saving on operational costs, etc., and it is a function of their current state and action, the current state of the platform, and the current MF state[4]. The total rewards of the platform and a specific driver, which are also referred to as value functions, are given by

$$V^{\pi}(s, \boldsymbol{h}_d) = \mathrm{E}_{\pi}\left(\sum_{t=1}^{T}(\rho)^t r(y^t, \boldsymbol{z}_d^t, x^t) \,|y^1 = s, \boldsymbol{z}_d^1 = \boldsymbol{h}_d\right) \tag{5}$$

$$V_d^{\pi_{d,i}}(s, s_d, \boldsymbol{h}_d) = \mathrm{E}_{\pi_{d,i}}\left(\sum_{t=1}^{T}(\rho)^t r_d\left(y^t, y_{d,i}^t, \boldsymbol{z}_d^t, x_{d,i}^t\right) \middle| y^1 = s, y_{d,i}^1 = s_d, \boldsymbol{z}_d^1 = \boldsymbol{h}_d\right) \tag{6}$$

where V^{π} and $V_d^{\pi_{d,i}}$ denote the total rewards for the platform and driver i given some specific initial states (i.e., $y^1 = s$, $y_{d,i}^1 = s_d$, and $\boldsymbol{z}_d^1 = \boldsymbol{h}_d$), respectively; π and $\pi_{d,i}$ denote the policies (a mapping from states to actions) of the platform and the i-th driver respectively; $x^t = \pi(y^t, \boldsymbol{z}_d^t)$ and $x_{d,i}^t = \pi_{d,i}\left(y^t, y_{d,i}^t, \boldsymbol{z}_d^t\right)$ represent the actions following the corresponding policies; $\rho \in (0,1)$ is the discount factor that measures how the policy balances the trade-off between immediate reward and long-term rewards; and $\mathrm{E}_{\pi}(\cdot)$ and $\mathrm{E}_{\pi_{d,i}}(\cdot)$ are the expectation operators under policies π and $\pi_{d,i}$, respectively.

Given specific formulas for rewards and state transition laws, a straightforward approach to solving the ride-sourcing MF-MDP model is to regard the platform and each driver as an agent, then try to solve the problem with a decentralized multi-agent MDP approach. However, the decentralized multi-agent MDP is generally hard to solve, especially when there are many agents. In reality, we will have a large number of minor agents (drivers). The distinct objectives of the major agent (platform) and minor agents (drivers) also render the solution-seeking process more unstable and intractable. Alternatively, we approximate the random MF vector $\boldsymbol{z}_d^t$ as a stationary process and optimize an aggregate policy for all drivers. Namely, as $M \to \infty$, we have $\boldsymbol{z}_d^t \xrightarrow{a.s.} \hat{\boldsymbol{z}}_d^t$. Similar approximations of asymptotic processes of homogeneous decision-makers have been adopted in studies of day-to-day traffic dynamics (Hazelton and Watling, 2004; Zhu et al., 2019b; Zhu et al., 2020a). In the simplified MF-MDP model, the platform takes actions according to policy π (i.e., $x^t = \pi(y^t, \hat{\boldsymbol{z}}_d^t)$), and the decision processes of all drivers are determined by policy π_d (i.e., $x_d^t = \pi_d(y^t, y_d^t, \hat{\boldsymbol{z}}_d^t)$) of a representative driver[5]. The MF state at the next time period depends on the current MF state and the platform's state, which is simplified as an updating rule $\hat{\boldsymbol{z}}_d^{t+1} = l_d(y^t, \hat{\boldsymbol{z}}_d^t)$. Note that the updating rule also incorporates the policies (for action taking) of the platform and the representative driver. Therefore, the standard multi-agent MF-MDP model with $1 + M$ agents can be reduced to a simplified MF-MDP model with only 2 agents:

- The ride-sourcing platform that acts as a major agent to design the optimal policy to maximize its total rewards. The optimal value function is defined as $V^*(s, \boldsymbol{h}_d) = \max_{\pi} \mathrm{E}_{\pi}\left(\sum_{t=1}^{T} \rho^t r(y^t, \boldsymbol{z}_d^t, x^t) \,|y^1 = s, \boldsymbol{z}_d^1 = \boldsymbol{h}_d\right)$.
- A representative driver who acts as a representative minor agent to pursue the optimal policy and maximize their total rewards. The total reward is regarded as the average total

[4] With specific research problems in ride-sourcing markets, we sometimes need to incorporate the previous state (i.e., y^{t-1}, $y_{d,i}^{t-1}$, and $\boldsymbol{z}_d^{t-1}$) into the formulas for rewards (i.e., r and r_d). This is because the before-and-after changes in states may affect the reward. For instance, if a subsidy is offered to a driver upon a new match with a passenger, we must check the diver's previous state and include the subsidy in the reward only if the current state is "matched/dispatched" and the previous state is "idle".

[5] For convenience and clarity, we use notation without a driver index to denote the state (y_d^t), action (x_d^t), and policy (π_d) of the representative driver in the simplified MF-MDP model.

rewards of all drivers. The optimal value function is defined as $V_d^*(s, s_d, \boldsymbol{h}_d) = \max_{\pi_d} \mathrm{E}_{\pi_d}(\sum_{t=1}^{T} \rho^t r_d(y^t, y_d^t, \boldsymbol{z}_d^t, x_d^t) \,|y^1 = s, y_d^1 = s_d, \boldsymbol{z}_d^1 = \boldsymbol{h}_d)$.

The form of function $l_d(y^t, \hat{\boldsymbol{z}}_d^t)$ determines the consistency between the approximated MF state $\hat{\boldsymbol{z}}_d^t$ and the exact MF state $\boldsymbol{z}_d^t$ (i.e., Eqs. (1)–(2)). A consistent approximation of the MF states is a critical requirement, such that the simplified MF-MDP model is able to represent the complex state transition and decision dynamics characterized by the standard MF-MDP. We discuss the consistency requirement in Section 3.3.

3.3 Optimal policies and the consistency requirement

An MDP model is generally solved by Bellman equations. We first illustrate the Bellman equations of the simplified MF-MDP model. An arbitrary MF state updating rule $l_d(y^t, \hat{\boldsymbol{z}}_d^t)$ is adopted without checking the consistency between $\hat{\boldsymbol{z}}_d^t$ and $\boldsymbol{z}_d^t$. The following propositions are necessary to obtain the optimal policies with Bellman equations:

Proposition 1. $\boldsymbol{H}_d$ is a continuous and compact set.

Proposition 2. Given a continuous reward function $r(y^t, \hat{\boldsymbol{z}}_d^t, x^t)$ on $\boldsymbol{H}_d$, the value function $V^{\pi}(s, \boldsymbol{h}_d)$ is continuous on $\boldsymbol{H}_d$.

Proposition 3. Given a continuous reward function $r_d(y^t, y_d^t, \hat{\boldsymbol{z}}_d^t, x_d^t)$ on $\boldsymbol{H}_d$, the value function $V_d^{\pi_d}(s, s_d, \boldsymbol{h}_d)$ is continuous on $\boldsymbol{H}_d$.

where **Proposition 1** is straightforward because $\boldsymbol{z}_d^t$ is continuous as M goes to infinity, and given specific policies π and π_d, the reward functions (i.e., r and r_d) and the corresponding value functions (i.e., V^{π} and $V_d^{\pi_d}$) are continuous, leading to **Propositions 2** and **3**.

The optimal policy for the platform can be solved based on the following Bellman equation:

$$V^*(s, \boldsymbol{h}_d) = \max_{a \in \boldsymbol{A}} \left\{ r(s, \boldsymbol{h}_d, a) + \rho \sum_{s' \in \boldsymbol{S}} Q(s'|s, \boldsymbol{h}_d, a) V(s', \boldsymbol{h}_d') \right\} \tag{7}$$

where $\boldsymbol{h}' = l_d(s, \boldsymbol{h}_d)$. In light of **Proposition 2**, the existence of an optimal policy π^* for Eq. (7) is guaranteed. Suppose the optimal policy π^* has been implemented in the simplified MF-MDP model. The Bellman equation for the representative driver is given by

$$V_d^*(s, s_d, \boldsymbol{h}_d) = \max_{a_d \in \boldsymbol{A}_d} \left\{ \begin{array}{c} r_d(s, s_d, \boldsymbol{h}_d, a_d) + \\ \rho \sum_{\substack{s' \in \boldsymbol{S}, \\ s_d' \in \boldsymbol{S}_d}} Q(s'|s, \boldsymbol{h}_d, a) Q_d(s_d'|s_d, \boldsymbol{h}_d, a_d) V_d(s', s_d', \boldsymbol{h}_d') \end{array} \right\} \tag{8}$$

where $a = \pi^*(s, \boldsymbol{h}_d)$.

Similarly, based on **Proposition 3**, given π^*, the optimal policy π_d^* exists for Eq. (8). In other words, there is an optimal policy group (π^*, π_d^*) that simultaneously satisfies Eqs. (7)–(8).

Next, we seek the specific formula of $l_d(y^t, \hat{\boldsymbol{z}}_d^t)$ for a consistent approximation of the MF state. The basic idea is to identify some updating rule of the exact MF state $\boldsymbol{z}_d^t$ in the simplified MF-MDP model, then adapt this rule to the approximated MF state $\hat{\boldsymbol{z}}_d^t$. Based on Eq. (2), we obtain the asymptotic $\boldsymbol{z}_d^t$ as M goes to infinity:

$$\lim_{M\to\infty} z_{d,s_d}^t = \lim_{M\to\infty} \frac{\sum_{i=1}^{M} \mathrm{I}\left(y_{d,i}^t = s_d\right)}{M} \xrightarrow{a.s.} \mathrm{P}(y_d^t = s_d) \tag{9}$$

To examine the asymptotic property of $\boldsymbol{z}_d^t$ under the optimal policy group (π^*, π_d^*), we introduce the following theorem, which is valid for any function $\hat{\boldsymbol{z}}_d^{t+1} = l_d(y^t, \hat{\boldsymbol{z}}_d^t)$.

Theorem 1. Let policy group (π^*, π_d^*) denote the optimal policies of Eqs. (7)–(8); the underlying vector $(y^t, y_d^t, \hat{\boldsymbol{z}}_d^t)$ forms a stationary Markov process.

Proof. The policy group provides stationary mapping from states to actions, such that $x^t = \pi^*(y^t, \hat{\boldsymbol{z}}_d^t)$ and $x_d^t = \pi_d^*(y^t, y_d^t, \hat{\boldsymbol{z}}_d^t)$. The state transition probability from state $(s, s_d, \boldsymbol{h}_d)$ to state $(s', s_d', \boldsymbol{h}_d')$ is given by

$$\begin{aligned} &\mathrm{P}(y^{t+1} = s', y_d^{t+1} = s_d', \hat{\boldsymbol{z}}_d^{t+1} = \boldsymbol{h}_d' | y^t = s, y_d^t = s_d, \hat{\boldsymbol{z}}_d^t = \boldsymbol{h}_d) \\ &= Q\left(s' | s, \boldsymbol{h}_d, \pi^*(s, \boldsymbol{h}_d)\right) Q_d\left(s_d' | s_d, \boldsymbol{h}_d, \pi_d^*(s, s_d, \boldsymbol{h}_d)\right) \mathrm{I}\left(\boldsymbol{h}_d' = l_d(s, \boldsymbol{h}_d)\right) \end{aligned} \tag{10}$$

where the LHS only depends on the current state $(s, s_d, \boldsymbol{h}_d)$.

■

In light of **Theorem 1**, the asymptotic z_{d,s_d}^t and $\mathrm{P}(y_d^t = s_d)$ are also Markov processes. Based on the transition law, the formula of $\mathrm{P}(y_d^{t+1} = s_d')$ is given by

$$\mathrm{P}(y_d^{t+1} = s_d') = \sum_{s_d \in \boldsymbol{S}_d} \mathrm{P}(y_d^t = s_d) Q_d\left(s_d' | s_d, \hat{\boldsymbol{z}}_d^t, \pi_d^*(y^t, s_d, \hat{\boldsymbol{z}}_d^t)\right) \tag{11}$$

Eq. (11) is summarized as a matrix product form:

$$\boldsymbol{z}_d^{t+1} = \boldsymbol{z}_d^t \widehat{\boldsymbol{Q}}_d(y^t, \hat{\boldsymbol{z}}_d^t) \tag{12}$$

where $\widehat{\boldsymbol{Q}}_d(y^t, \hat{\boldsymbol{z}}_d^t) = \left[Q_d\left(s_d' | s_d, \hat{\boldsymbol{z}}_d^t, \pi_d^*(y^t, s_d, \hat{\boldsymbol{z}}_d^t)\right)\right]_{|\boldsymbol{S}_d| \times |\boldsymbol{S}_d|}$ is a probability transition matrix of the MF state. Let $\hat{\boldsymbol{z}}_d^1 = \boldsymbol{z}_d^1$ and $l_d(y^t, \hat{\boldsymbol{z}}_d^t) = \hat{\boldsymbol{z}}_d^t \widehat{\boldsymbol{Q}}_d(y^t, \hat{\boldsymbol{z}}_d^t)$; for any $t \in \boldsymbol{T}$, we can obtain the following equation by iteratively substituting Eq. (12) and $l_d(y^t, \hat{\boldsymbol{z}}_d^t)$.

$$\hat{\boldsymbol{z}}_d^{t+1} = \hat{\boldsymbol{z}}_d^1 \prod_{t'=1}^{t} \widehat{\boldsymbol{Q}}_d\left(y^{t'}, \hat{\boldsymbol{z}}_d^{t'}\right) = \boldsymbol{z}_d^1 \prod_{t'=1}^{t} \widehat{\boldsymbol{Q}}_d\left(y^{t'}, \hat{\boldsymbol{z}}_d^{t'}\right) = \boldsymbol{z}_d^{t+1} \tag{13}$$

Therefore, we conclude that $\mathrm{E}_{\pi^*, \pi_d^*}(\hat{\boldsymbol{z}}_d^t) = \boldsymbol{z}_d^t$ and the consistency requirement for the approximation of MF states reduces to the following updating rule:

$$\hat{\boldsymbol{z}}_d^{t+1} = l_d^{\#}(y^t, \hat{\boldsymbol{z}}_d^t) = \hat{\boldsymbol{z}}_d^t \widehat{\boldsymbol{Q}}_d(y^t, \hat{\boldsymbol{z}}_d^t) \tag{14}$$

where superscript # means that the updating rule is consistent.

We refer to the combination of the optimal policies for the platform and the representative driver and the consistent updating rule for MF states, i.e., $\left(\pi^*, \pi_d^*, l_d^{\#}(y^t, \hat{\boldsymbol{z}}_d^t)\right)$, as a consistent optimal solution of the simplified MF-MDP model. Note that $\left(\pi^*, \pi_d^*, l_d^{\#}(y^t, \hat{\boldsymbol{z}}_d^t)\right)$ satisfies Eqs. (7), (8), and (14) simultaneously. The consistency of the stochastic process depicted in Eq. (14)

requires a soft policy for the representative driver, i.e., $a_d|\pi_d \sim \mathrm{P}\left(x_d^{t+1} = a_d|\pi_d(y^t, y_d^t, \hat{\boldsymbol{z}}_d^t)\right)$. In contrast to a "hard" policy that selects a deterministic action given the observed state, a "soft" policy is a probabilistic distribution over the action set, and the agent stochastically selects an action according to the distribution given any observed state. The design of the soft policy enables the model to use a single policy to represent the aggregate actions of all drivers, rather than determining different policies for each driver.

The generalized MF-MDP model and its theoretical guarantees in terms of simplification and optimal solution seeking allow us to formulate and solve a variety of research questions in ride-sourcing markets. For instance, we can use this model to delineate the dynamics of a ride-sourcing market in which drivers (minor agents) have self-relocating behaviors for maximizing their individual earnings while a platform tries to achieve a more efficient system by imposing spatial-temporal pricing/subsidy strategies (see Section 4). As described in this subsection, the simplified MF-MDP model contributes to the solution algorithm of MDPs with multiple mixed agents. However, due to complex interactions between the platform and drivers, the formulas of $Q(\cdot \mid \cdot)$, $Q_d(\cdot \mid \cdot)$, r and r_d can be complicated. Moreover, the solution space with respect to states, actions, and time periods can be extremely large. Therefore, it is generally difficult to obtain exact optimal policies via solving the Bellman equations. A typical method is to use simulations to approximate the interactions between the environment and agents, and attempt to find close-optimal policies through reinforcement learning-based algorithms (Wang et al., 2018; Li et al., 2019; Jin et al., 2019; Shou et al., 2020b). The idea of a soft policy[6] for the representative driver and the consistent updating rule for the approximated MF state are valuable for designing computationally efficient simulation processes.

4 DESIGN AND ANALYZE SUBSIDIES FOR DRIVERS WITH SELF-RELOCATION

In this section, we substantialize the proposed generalized MF-MDP model in a specific research problem in which a platform tries to better allocate spatial-temporal subsidies for drivers, while drivers attempt to maximize their individual earnings by self-relocation. The formulation (in terms of states, actions, and rewards) of this model is introduced in Section 4.1, while a representative-agent reinforcement learning algorithm for solving the model is developed in Section 4.2.

4.1 FORMULATION OF THE SPECIFIC RIDE-SOURCING MF-MDP MODEL

In this subsection, we provide definitions and intuitive explanations of the states, actions, and rewards of the platform and drivers, and introduce a matching rule between passengers and drivers. Readers can refer to Appendix A for detailed mathematical formulations of the state transition laws, order-matching probabilities, and rewards.

In a ride-sourcing market with spatial-temporal imbalance between demand and supply. We use a hexagonal zone system, which has been used in some previous studies (Ke et al., 2019; Xu et al. 2018; Lin et al., 2018) and DiDi's ride-sourcing simulator (Xu et al. 2017). There are O hexagonal zones, passenger demand is exogenous, and driver supply can be characterized by the MF state $\boldsymbol{z}_d^t$ of M drivers (also by the approximate state $\hat{\boldsymbol{z}}_d^t$ in the simplified MF-MDP model). Other notation is the same as in the generalized model in Section 3. The platform could lose passengers in zones with high demand and insufficient idle vehicles. To increase net revenue

[6] The soft policy maps a state to a probability distribution over all possible actions. Given a MF state, the representative driver takes a stochastic action based on a probability distribution that is determined by the soft policy. In this way, we can approximate the collective behavior/decision of a group of drivers by using one representative driver at the expense of a small measuring error, especially when the number of drivers is large.

and the number of passengers served, the platform may offer spatial-temporal (time- and zone-based) subsidies to incentivize idle drivers to move from demand-cool zones to demand-hot zones. Meanwhile, drivers design their self-relocation strategies to increase their own income.

We begin with the states and actions of the platform. To model the subsidy strategy across different zones, the state of the platform is characterized by an $1 + O$ dimensional vector $\boldsymbol{s} = [t, s_1, \dots, s_O]$, where $s_o \in \{0, \beta\}$ denotes the subsidy in zone $o \in \{1,2, \dots, O\}$ such that 0 and β refer to "no subsidy" and "offering a subsidy," respectively, and β is a predefined amount of subsidy per ride (i.e., subsidy rate). In theory, we allow β to be non-positive values in the model, and $\beta = 0$ means a "non-subsidy" strategy and $\beta < 0$ indicates that drivers pay an extra "charge" rather than get subsidies. If zone o is subsidized at time t, drivers who are matched and dispatched to passengers originating from zone o at time t will be offered the same amount of subsidy (such a scheme is referred to as a uniform subsidy scheme). The platform's action with respect to subsidy is represented by a vector $\boldsymbol{a} = [a_1, \dots, a_O]$, where $a_o \in \{0, \beta\}$. Therefore, we have $\boldsymbol{S} = \{[t, s_1, \dots, s_O] | s_o \in \{0, \beta\}, o \in \{1,2, \dots, O\}, t \in \{1,2, \dots, T\}\}$ and $\boldsymbol{A} = \{[a_1, \dots, a_O] | a_o \in \{0, \beta\}, o \in \{1,2, \dots, O\}\}$. In this manner, both the state vector $\boldsymbol{s}$ and the action vector $\boldsymbol{a}$ represent the spatial distribution of subsidy, and the state of the platform for the next time period is identical to its current action. Eq (A.1) in Appendix A gives a mathematical formula of the state transition law $Q(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a})$.

Next, we introduce the states and actions of drivers. To comprehensively describe a driver's different status (e.g., idle, on the way to pick up passengers, delivering passengers), we formulate a driver's state as a six-dimensional vector $\boldsymbol{s}_d = [t, s_{d1}, s_{d2}, s_{d3}, s_{d4}, s_{d5}]$. Here, $s_{d1} \in \{0,1,2\}$ denotes the current task of the driver/vehicle with 0, 1, and 2 representing "idle," "picking up a passenger," and "delivering a passenger," respectively; $s_{d2} \in \{1,2, \dots, T\}$ denotes the remaining time periods before finishing the current status; $s_{d3} \in \boldsymbol{O}$ denotes the idling zone in which the driver is idle and waiting for a match; and $s_{d4} \in O$ and $s_{d5} \in O$ denote the origin and destination of the current passenger order, respectively. The value of s_{d2} depends on the travel time on the zone network. The action of the driver is the destination zone of self-relocation. Given the driver's current idling zone o (i.e., $s_{d3} = o$), the set of their actions (i.e., $\boldsymbol{A}_d|o$) is represented by set $\boldsymbol{J}_o$, which is the set of adjacent zones of o plus o itself. The action $a_d = o$ means the driver will stay in the current zone, and other actions $a_d \in \boldsymbol{J}_o/o$ indicate that the driver will relocate to an adjacent zone. A self-relocation action is only needed when the driver has no picking-up or delivering tasks and is not on the way of cruising to an adjacent zone, i.e., when $s_{d1} = 0$ and $s_{d2} = 0$ (referred to as a "purely idle" state). Therefore, one only optimizes policies in the "purely idle" state-action space (i.e., the Cartesian product of set $\{\boldsymbol{s}_d \in \boldsymbol{S}_d | s_{d1} = 0, s_{d2} = 0\}$ and the action set). Given a large number of drivers, drivers with the "purely idle" state are in different pairs of (t, s_{d3}) that could uniformly distribute across the spatial-temporal domain of the scenario, making the state-action-reward transitions in the learning process non-sparse. Furthermore, a state with $s_{d1} = 0$ and $s_{d2} > 0$ indicates a "self-relocating" state with a relocation destination such that no action is needed until he/she arrives at the destination and becomes purely idle. Following Eq. (4), the state transition law for a driver, i.e., $Q_d(\boldsymbol{s}_d'|\boldsymbol{s}_d, \boldsymbol{h}_d, a_d)$, depends on its current state and action as well as the MF state of drivers; detailed formulas are given by Eqs. (A.2)–(A.5) in Appendix A.

Next, we introduce the order-matching rule between drivers and passengers. If a driver is in a "purely idle" or "self-relocating" state, they have a chance to be matched with a passenger. Therefore, the state transition probability of the driver is substantially affected by the matching rule. Generally, the platform considers a maximal matching radius that only prevents passengers from being matched with far away drivers. A larger radius allows a lager flexibility in matching; namely, a larger pool of candidate idle drivers is generated for each passenger, and thus the

matching rate becomes larger; this also indicates a smaller passengers' expected waiting time. However, since some distant drivers may be matched to passengers, a larger matching radius will increase the average pick-up time. MDP-based models in the literature usually adopt a small matching radius so that drivers and passengers can be matched only if they are in the same zone (e.g., Shou et al., 2020b). Such matching rules ignore the cross-region dispatching and picking-up events that are commonly observed in ride-sourcing services, and thus are more suitable for taxi markets rather than for ride-sourcing markets. To allow cross-region matching between passengers and drivers, in this paper we propose an edge-based matching rule in calculating transition laws (Figure 2). Termination "edge-based" means that we allow drivers in zone o to be matched with passengers in zones $o' \in \boldsymbol{J}_o$; if the driver is matched with a passenger in zone $o' = o$, they immediately pick up the passenger and start the delivering task; if the driver is matched with a passenger in an adjacent zone $o' \in \boldsymbol{J}_o/o$, they must spend some time on the picking up task before delivering the passenger. For simplicity, we assume that drivers and passengers in each hexagonal zone is uniformly distributed (which does not mean that they are uniform across the network with many zones). The edge-based rule matches drivers and passengers near each common edge between zone o and its adjacent zones $o' \in \boldsymbol{J}_o/o$.

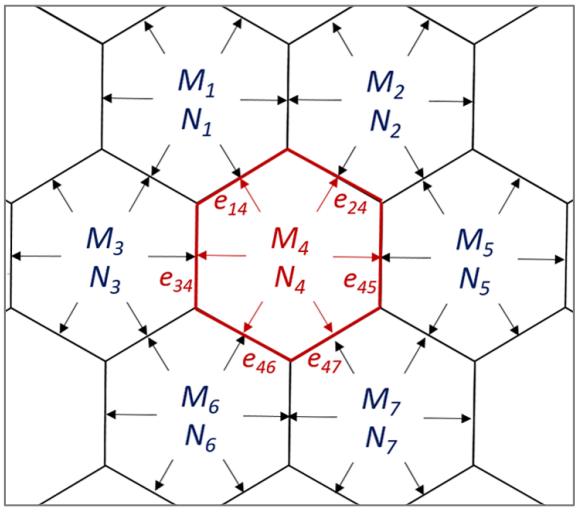


Figure 2. Edge-based matching.

Let M_o denote the number of idle drivers in zone o (i.e., with state $s_{d1} = 0$ and $s_{d3} = o$); N_o the number of passengers with origin in zone o; and $e_{oo'}$ the common edge between two hexagonal zones o and o'. At each time period, M_o is obtained from the MF state and N_o is observable and thus exogenously given. We illustrate the number of matches near edge $e_{oo'}$ using a simple example. Taking the zone indices in Figure 2, for instance, $\boldsymbol{J}_4 = \{2,5,7,6,3,1,4\}$, $\boldsymbol{J}_4/\{4\} = \{2,5,7,6,3,1\}$, and we match drivers and passengers near edge e_{14}. With uniformly distributed demand and supply, there are $\frac{M_4}{6}$ idle drivers and $\frac{N_4}{6}$ passengers near e_{14} in zone 4, and $\frac{M_1}{6}$ idle drivers and $\frac{N_1}{6}$ passengers near this edge in zone 1. Therefore, there are a total of $\frac{M_1+M_4}{6}$ idle drivers and $\frac{N_1+N_4}{6}$ passengers to be matched near edge e_{14}; for these passengers and drivers, we allow a driver/passenger in zone 4 to be matched with passengers/drivers in either zone 1 or zone 4. Based on a matching rule in Yu et al. (2019), the number of matches near e_{14} is approximated as $\min\left\{\frac{M_1+M_4}{6}, \frac{N_1+N_4}{6}\right\}$. Similar to this example, we can compute the number of matches near an arbitrary edge.

To approximate the matching probabilities of a driver, we assume that the numbers of matched passengers and drivers are proportional to the corresponding demand and supply near the common edge. To continue with the example above, if a driver is in zone 1, the probability

that they are near edge e_{14} equals $\frac{1}{6}$, and the probability that they get a passenger order near edge e_{14} is $\frac{1}{6}\frac{\min\left\{\frac{M_1+M_4}{6},\frac{N_1+N_4}{6}\right\}}{\frac{M_1}{6}+\frac{M_4}{6}}=\frac{\min\left\{\frac{M_1+M_4}{6},\frac{N_1+N_4}{6}\right\}}{M_1+M_4}$. Detailed formulas for calculating the number of matches and driver-side matching probabilities are given in Eqs. (A.6)–(A.9) in Appendix A. We need the MF state to compute M_o (also denoted as $M_o(\boldsymbol{h}_d)$ in Eq. (A.7)) and then the matching probabilities; this explains why the state transition law of a driver depends on the MF state, i.e., $Q_d(\boldsymbol{s}'_d|\boldsymbol{s}_d,\boldsymbol{h}_d,a_d)$. To our best knowledge, this paper is among the first idle vehicle relocation studies that consider cross-zone matches with the proposed edge-based matching rule.

Last, we discuss the rewards for the platform and drivers. We consider that the platform's objective is to maximize a weighted sum of the net revenue and the service rate, which is defined by the number of passengers served divided by the total passenger demand. Intuitively, the service rate reflects passengers' satisfaction, and a low service rate may cause a decrease in passenger demand in the long run and affect the platform's market share. Our motivation to set this objective structure is that the platform usually needs to make a trade-off between net revenue (short-term benefits) and customer service rate (long-term interests). To be more specific, the reward (also referred to as the objective value) of the platform is formulated by $r(\boldsymbol{y}^t=\boldsymbol{s},\boldsymbol{z}_d^t=\boldsymbol{h}_d,\boldsymbol{z}_d^{t-1}=\boldsymbol{h}'_d)=r_1(\boldsymbol{h}_d)-r_2(\boldsymbol{s},\boldsymbol{h}_d,\boldsymbol{h}'_d)+\mu r_3(\boldsymbol{s},\boldsymbol{h}_d,\boldsymbol{h}'_d)$, where r_1 refers to the commission withheld from trip fares by the platform; r_2 is the amount of subsidies offered to drivers; r_3 is the service rate; and μ denotes the weight of the service rate for the platform[7]. In addition to the MF state of drivers and the state of the platform, the calculation of r involves the following predefined variables: the ride-sourcing trip fare per time period (i.e., trip fare rate) α, commission rate for the platform η, and total passenger demand across the entire operational horizon N. Readers can refer to Eqs. (A.10)–(A.13) in Appendix A for detailed formulas for r_1, r_2, and r_3.

A driver's objective is to maximize the total income. The reward (referred to as income) for a particular driver is the sum of the trip fare and subsidy offered by the platform, i.e., $r_d\left(\boldsymbol{y}^t=\boldsymbol{s},\boldsymbol{y}_{d,i}^t=\boldsymbol{s}_d,\boldsymbol{y}_{d,i}^{t-1}=\boldsymbol{s}'_d\right)=r_{d1}(\boldsymbol{s}_d)+r_{d2}(\boldsymbol{s},\boldsymbol{s}_d,\boldsymbol{s}'_d)$, where r_{d1} and r_{d2} denote the income from the trip fare and the subsidy, respectively. The income from fare is provided gradually during the delivery task, while the subsidy is a one-time reward upon a match if the origin of the passenger is subsidized. Similar to r, we need α and η to compute r_d. Detailed formulas for r_{d1} and r_{d2} are given in Eqs. (A.14)–(A.16) in Appendix A.

To sum up, we provide the formulation of a specific standard MF-MDP model, in which the MF state of drivers is mainly used to compute the order-matching probabilities. As a result, the MF state directly determines drivers' state transition law and the platform's reward; it also affects the matching outcome of an individual driver and their reward. Approximation of the MF state and simplification of the MF-MDP model play an important role in solution-finding. For the simplified MF-MDP model, the states, actions, transition laws, and rewards for the representative driver are the same as for an arbitrary driver in the standard model. In light of Eq. (14), the consistent updating rule for the approximated MF state is $\widehat{\boldsymbol{Q}}_d(\boldsymbol{y}^t,\hat{\boldsymbol{z}}_d^t)=\widehat{\boldsymbol{Q}}_d(\hat{\boldsymbol{z}}_d^t)$, which can be summarized based on Eqs. (A.4)–(A.5).

4.2 SOLUTION ALGORITHMS

Given the state transition laws for the platform and drivers in Eqs. (A.1)–(A.5), the order-matching probabilities in Eqs. (A.6)–(A.9), and the rewards in Eqs. (A.10)–(A.16), we have a specific formulation of a standard MF-MDP model with multiple mixed agents. Although the

[7] As stated in footnote 3, we consider the previous state $\boldsymbol{z}_d^{t-1}$ in the formulation of r and consider $\boldsymbol{y}_{d,i}^{t-1}$ in r_d.

rewards of the platform and drivers and the state transition law of the platform have deterministic formulas, drivers' state changes are dependent on the stochastic order-matching process with numerous possible outcomes (e.g., different origins and destinations of passenger orders), making it difficult to obtain an exact solution via Bellman equations. As discussed in Section 3, reinforcement learning algorithms can be adopted to solve such a multi-agent MF-MDP model with large state and action sets, complex state transition laws, and reward formulas. We consider two solution-seeking approaches below.

- The multi-agent approach, which solves the standard MF-MDP model with 1 platform and M drivers. In a reinforcement learning algorithm, each of the $1 + M$ agents learns their own decision policy[8], which can be characterized by Q-tables, neural networks, etc. An agent takes actions based on their policy and updates the parameters of the policy via their experiences under the actions. The MF states $\boldsymbol{z}_d^t$ are summarized based on the states of all drivers (Eq. (2)).
- The representative-agent approach, which solves the simplified MF-MDP model with one platform and one representative driver. In a reinforcement learning algorithm, we create two decision policies: one for the platform and the other for the representative driver. The representative driver takes actions based on a soft policy and updates the parameters via experiences under the actions. We adopt the approximated MF state $\hat{\boldsymbol{z}}_d^t$, which is updated according to the previous approximated MF state ($\widehat{\boldsymbol{Q}}_d(\hat{\boldsymbol{z}}_d^t)$ summarized based on Eqs. (A.4)–(A.5)).

The multi-agent approach is proposed as a benchmark that solves the standard MF-MDP model. With a large M, the MF space is continuous and compact, and **Propositions 1** to **3** are valid for the representative-agent approach. Comparing with the benchmark, the representative-agent approach meets the consistency requirement with respect to the MF state and could be faster in terms of computation and identifying the optimal policies.

In this paper, the two approaches are implemented via the A2C algorithm, one of the most popular reinforcement learning algorithms (Mnih et al., 2016). For each agent, the A2C algorithm establishes two networks (also referred to as a group of networks): one policy network (or critic network) that observes the current states and generates policy, and one value network (or actor network) that evaluates the performance of the policy. Both networks are parameterized multi-layer neural networks, and their parameters (e.g., θ_p for the policy network and θ_v for the value network) are updated iteratively. The parameters of the value network θ_v can be updated by minimizing a loss function $L(\theta_v)$ defined as follows (Lin et al., 2018):

$$L(\theta_v) = \left[V_{\theta_v}(y^t) - \left(r(y^t, x^t) + \rho V_{\theta_v'}(y^t)\right)\right]^2 \tag{15}$$

where θ_v denote the parameters of the value network to be updated, θ_v' denote the parameters of the targeted value network, and $r(y^t, x^t)$ be the current reward. As for the policy network, parameters θ_p are updated using a gradient descent rule $\theta_p \leftarrow \theta_p + \delta \nabla_{\theta_p} G(\theta_p)$, where δ is the learning rate and $\nabla_{\theta_p} G(\theta_p)$ represents the gradient given below.

$$\nabla_{\theta_p} G(\theta_p) = \nabla_{\theta_p} \log \pi_{\theta_p}(x^t | y^t) \left[x^t + \rho V_{\theta_v'}(y^{t+1}) - V_{\theta_v'}(y^t)\right] \tag{17}$$

[8] Note that although the drivers are homogeneous, their optimal policies can differ. That is, idle drivers who are in the same zone might have different self-relocation destinations; otherwise, they would relocate to the same destination and compete with each other for passengers, which would result in a small matching probability and a low average income.

where $\pi_{\theta_p}(x^t|y^t)$ refers to the action taken by the agent given state y^t according to the policy π parameterized by θ_p, $x^t + \rho V_{\theta_v'}(y^{t+1}) - V_{\theta_v'}(y^t)$ is an advantage function used to reduce the variance of the value function and approximate the policy gradient[9].

The algorithm for the multi-agent approach is shown in Algorithm 1, which is referred to as the multi-agent actor-critic (MAC) algorithm (i.e., the benchmark algorithm). As mentioned previously, in the MAC algorithm each driver or the platform has a specific group of networks to characterize their own policy, which leads to a total of $1 + M$ groups of networks. The algorithm for the representative-agent approach is shown in Algorithm 2, which is referred to as the representative-agent actor-critic (RAC) algorithm. In the RAC algorithm, we propose two groups of networks: one for the platform and the other for the representative driver. In both MAC and RAC algorithms, N_E, N_S, and D denote the maximal number of learning epochs, the maximal number of learning samples, and the replay memory, respectively[10]. Note that in the MAC algorithm, index i denotes driver ID. We must simulate the transitions of each individual driver (i.e., steps 3.2 to 3.7, including states, actions, and rewards) based on the individual policy $\pi_{d,i}$; in the RAC algorithm, we simulate the transitions of the representative driver (i.e., steps 3.2 to 3.7) according to the soft policy π_d and use index j to represent the indices of the simulated transitions regardless of the ID of a specific driver who experiences the transition. In steps 4.4 to 4.6 in the MAC algorithm, each driver learns and updates the parameters of their own value network and policy network, via a mini-batch of samples extracted from the replay memory. In the RAC algorithm, the representative driver learns and updates the soft policy in steps 4.4 to 4.6.

Theoretically, given unlimited computational power, the MAC algorithm might learn the optimal self-relocation policy for each driver by conducting extensive simulations and sampling sufficient transitions over the huge solution space, which contains all feasible policies for each driver and the platform. However, computational resources are generally limited in practice, and thus it is nearly impossible to generate a massive number of samples for all possible transitions. In addition, one driver's self-relocation policy will affect other drivers' rewards, which is reflected by the impact of the MF state on the matching outcomes of each time period. Limited computational power and complicated competitive relationships between drivers make it difficult for the MAC to find the right pathway and obtain close-optimal policies for all drivers. By contrast, the RAC algorithm has a smaller solution space (the Cartesian product of the platform's and the representative driver's state-action set), and thus could identify a close-optimal solution more efficiently, which may provide solutions better than those obtained with the MAC.

Algorithm 1. Multi-Agent Actor-Critic (MAC) to solve the standard MF-MDP model

1. Initialize the value network with a fixed value table.
For $n_e = 1$ to N_E do:
  2. Reset simulator, get initial state $\boldsymbol{y}^1$, $\{\boldsymbol{y}_{d,i}^1\}_M$, and $\boldsymbol{z}_d^1$.
  3. Stage one: collecting experience.
  For $t = 1$ to T do:
    3.1. Decide action x^t based on policy π, and execute x^t.
    For $i = 1$ to M do:

[9] Note that Eqs. (15)–(18) give the basic formulas of the A2C algorithm. When adopting the A2C algorithm in the MF-MDP model, r, V_{θ_v}, V_{target} are calculated based on the definitions and formulas in Section 3 and Section 4.1.
[10] These are general terminations in reinforcement learning. A learning epoch refers to an iteration for the algorithm to simulate the transitions of the agents and update the parameters of their policies, and a replay memory is used to store and sample the simulated transitions.

3.2. Decide action $x_{d,i}^t$ based on policy $\pi_{d,i}$, and execute $x_{d,i}^t$.
End for.
3.3. Based on $\{\boldsymbol{y}_{d,i}^1\}_M$ and $\boldsymbol{z}_d^t$, compute $Q_d(\cdot \mid \cdot)$ for the simulator.
3.4 Run the simulator and observe next state $\boldsymbol{y}^{t+1}$ and $\{\boldsymbol{y}_{d,i}^{t+1}\}_M$.
3.5. Summarize $\boldsymbol{z}_d^{t+1}$ based on $\{\boldsymbol{y}_{d,i}^{t+1}\}_M$.
End for.
3.6. Observe reward $r^t(\boldsymbol{y}^t, \boldsymbol{z}_d^t, \boldsymbol{z}_d^{t-1})$, $\{r_d^t(\boldsymbol{y}^t, \boldsymbol{y}_{d,i}^t, \boldsymbol{y}_{d,i}^{t-1})\}_M$.
3.7. Store transitions $(\boldsymbol{y}^t, x^t, \boldsymbol{y}^{t+1}, r^t(\cdot))$ and $\left\{\left(\boldsymbol{y}_{d,i}^t, x_{d,i}^t, \boldsymbol{y}_{d,i}^{t+1}, r_d^t(\cdot)\right)\right\}_M$ to D.
End for.
4. Stage two: learning the experiences.
For $n_s = 1$ to N_S do:
4.1. Sample a mini-batch of transitions $(\boldsymbol{y}^t, x^t, \boldsymbol{y}^{t+1}, r^t(\cdot))$ from D.
4.2. Update the platform's value networks by minimizing $L(\theta_v)$.
4.3. Update the platform's policy networks as $\theta_p \leftarrow \theta_p + \delta \nabla_{\theta_p} G(\theta_p)$.
For $i = 1$ to M do:
4.4. Sample a mini-batch of transitions $\left(\boldsymbol{y}_{d,i}^t, x_{d,i}^t, \boldsymbol{y}_{d,i}^{t+1}, r_d^t(\cdot)\right)$ from D.
4.5. Update the ith driver's value networks by minimizing $L(\theta_v)$.
4.6. Update the ith driver's policy networks as $\theta_p \leftarrow \theta_p + \delta \nabla_{\theta_p} G(\theta_p)$.
End for.
End for.
End for.
5. Finish.

1 Algorithm 2. Representative-Agent Actor-Critic (RAC) to solve the simplified MF-MDP model

1. Initialize the value network with a fixed value table.
For $n_e = 1$ to N_E do:
2. Reset simulator, get initial state $\boldsymbol{y}^1$, $\{\boldsymbol{y}_{d,j}^1\}_M$, and $\hat{\boldsymbol{z}}_d^1 = \boldsymbol{z}_d^1$.
3. Stage one: collecting experience.
For $t = 1$ to T do:
3.1. Decide action x^t based on policy π, and execute x^t.
3.2. Decide actions $\{x_{d,j}^t\}_M$ based on soft policy π_d, and execute $\{x_{d,j}^t\}_M$.
3.3. Based on $\{\boldsymbol{y}_{d,j}^1\}_M$ and $\hat{\boldsymbol{z}}_d^t$, compute $Q_d(\cdot \mid \cdot)$ for the simulator.
3.4. Observe next state $\boldsymbol{y}^{t+1}$ and $\{\boldsymbol{y}_{d,j}^{t+1}\}_M$.
3.5. Calculate $\hat{\boldsymbol{z}}_d^{t+1}$ based on $Q_d(\cdot \mid \cdot)$ and $\hat{\boldsymbol{z}}_d^t$.
3.6. Observe reward $r^t(\boldsymbol{y}^t, \hat{\boldsymbol{z}}_d^t, \hat{\boldsymbol{z}}_d^{t-1})$, $\{r_d^t(\boldsymbol{y}^t, \boldsymbol{y}_{d,j}^t, \boldsymbol{y}_{d,j}^{t-1})\}_M$.
3.7. Store transitions $(\boldsymbol{y}^t, x^t, \boldsymbol{y}^{t+1}, r^t)$ and $\left\{\left(\boldsymbol{y}_{d,j}^t, x_{d,j}^t, \boldsymbol{y}_{d,j}^{t+1}, r_d^t(\cdot)\right)\right\}_M$ to D.
End for.
4. Stage two: learning the experiences.
For $n_s = 1$ to N_S do:
4.1. Sample a mini-batch of transitions $(\boldsymbol{y}^t, x^t, \boldsymbol{y}^{t+1}, r^t(\cdot))$ from D.
4.2. Update the platform's value networks by minimizing $L(\theta_v)$.
4.3. Update the platform's policy networks as $\theta_p \leftarrow \theta_p + \delta \nabla_{\theta_p} G(\theta_p)$.
4.4. Sample a mini-batch of transitions $\left\{\left(\boldsymbol{y}_{d,j}^t, x_{d,j}^t, \boldsymbol{y}_{d,j}^{t+1}, r_d^t(\cdot)\right)\right\}_M$ from D.

4.5. Update the representative driver's value network by minimizing $L(\theta_v)$.
4.6. Update the representative driver's policy network as $\theta_p \leftarrow \theta_p + \delta \nabla_{\theta_p} G(\theta_p)$.
End for.
End for.
5. Finish.

5. NUMERICAL STUDY

In this section, we conduct a set of numerical experiments with the MF-MDP model and algorithms developed in Section 4. We show (1) the computation time, converging speed for learning policies, and converged total rewards for the agents of the RAC algorithm compared with the benchmark MAC algorithm; (2) the impact of spatial-temporal subsidies on drivers' relocation strategies; and (3) the platform's different spatial-temporal subsidy strategies that balance the trade-offs between net revenue and service rate.

5.1 SCENARIO SETTINGS

The zone network of the ride-sourcing market is illustrated in Figure 3. Zone IDs are shown in the center of the hexagons, and travel times (number of time periods) between adjacent zones are shown via underlined numbers near edges. For instance, a driver needs 2 time periods to travel between zone 1 and zone 2. In this small town with 7 zones, we assume that zone 4 is a residential area, zone 7 is a business area, and zone 2 has a railway station. Due to the huge computational costs, numerical studies with a small network were usually adopted in reinforcement learning-related studies. For example, Mao et al. (2020) divide Manhattan into 8 zones and examine drivers' optimal repositioning among these zones. Braverman et al. (2019) use a nine-region network with parameters calibrated by DiDi data to evaluate their proposed empty-car routing policy. Moreover, by using a small network, we can observe clear patterns of drivers' sequential actions and better understand how self-relocation is affected by subsidies.

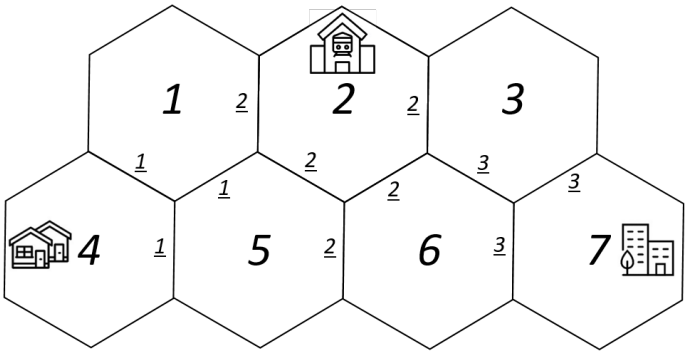
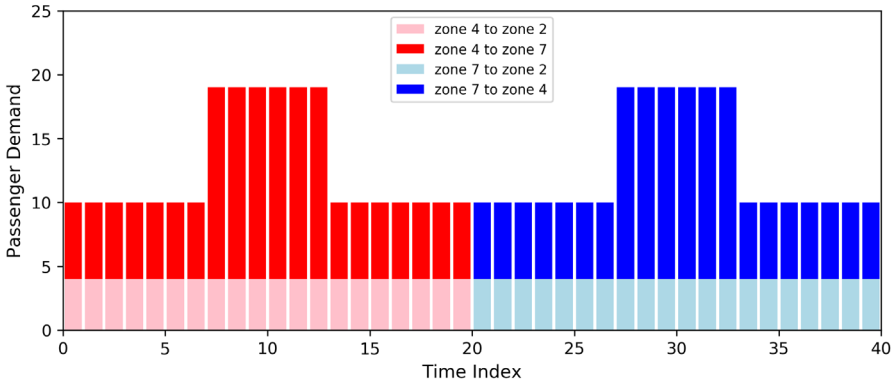


Figure 3. Network of the numerical study.

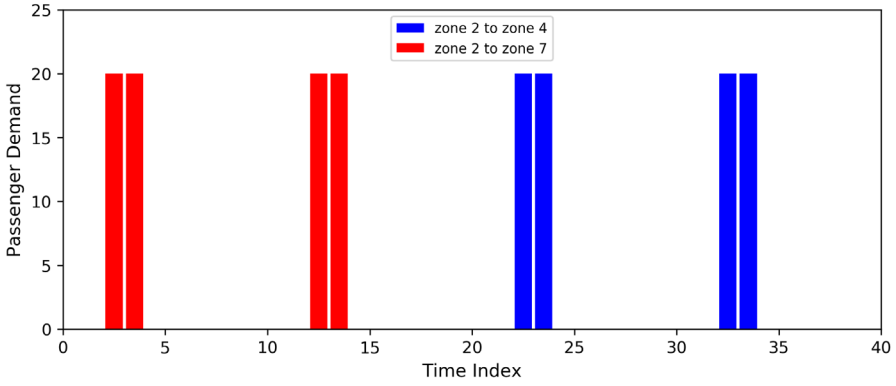
We consider a general ride-sourcing market that consists of both tidal and periodic passenger demand. The pattern of deterministic passenger demand is shown in Figure 4[11]. We consider a total of 40 time periods in the operational horizon, and each period represents 5 minutes. First, there is tidal demand between zone 4 and zone 7 (see Figure 4(a)), such that passengers go from the residential area to the business center at time periods 1–20 (red bars) and

[11] Note that the ticks on the horizontal axis, i.e., time index, refer to "time point," while demand is generated during time periods. Therefore, for instance, time period 15 refers to the period between time index 15 and time index 16. The same representations are adopted in Figure 6.

return at time periods 21–40 (blue bars). Each tidal demand has a peak period—i.e., during time periods 8–13 and 28–33. Second, also in Figure 4(a), we assume that since some passengers live in zone 4 but work outside the small town, there is ride-sourcing demand from zone 4 to zone 2 at time periods 1–20 (light pink bars); and since some passengers work in zone 7 but live out of town, there is ride-sourcing demand from zone 7 to zone 2 at time periods 21–40 (light blue bars). Third, in Figure 4(b), for every 10 time periods (50 minutes), a train arrives at zone 2, and passengers from the train either go to work (red bars before time index 20) or home (blue bars after time index 20). We use this demand setting because it reflects general scenarios with both demand-hot areas, demand-cold areas, tide demand, and periodic demand. Based on the edge-based matching rule, idle drivers in zones 1, 3, 5, 6 can also get passenger orders. However, the matching probability at zones 1, 3, 5, 6 would be much lower than that at zones 2, 4, and 7. In this case, there are trade-offs in the market with competing drivers: a driver can idly cruise to zone 4 to ensure a high matching probability; alternatively, the driver can stay in zones 1 or 5 such that he/she has a low probability of getting an order from zone 4 or zone 2 (at time periods when a train arrives).



(a) Tidal demand in zones 2, 4 and 7



(b) Periodic demand in zone 2

Figure 4. Passenger demand.

Other exogenous parameters are set as follows: commission rate $\eta = 0.20$, trip fare rate $\alpha = 10$ CNY per time period, and discount factor $\rho = 0.8$. Based on the numerical settings and definitions of states and actions, the cardinality of the platform's state-action set is 2,560; the

cardinality of the state-action set for a driver to take actions (i.e., in a purely idle state) is $1{,}160$[12]. With multiple drivers, since agents take actions independently, the cardinality of the solution space (containing the state-action sets for all the agents) can be as large as $2{,}560 \times 1{,}160^{M}$, which makes it difficult to solve via the MAC algorithm. In contrast, for the RAC algorithm, the cardinality of the solution space reduces to $2{,}560 \times 1{,}160$.

The hyperparameters of the algorithms for all subsequent numerical studies are as follows. In the RAC algorithm, for the platform agent, we establish a simple three-layer fully connected network with 24 neurons in the hidden layer for the value network and a three-layer fully connected network with 24 neurons in the hidden layer for the policy network. Similarly, for the representative driver agent, we use a three-layer fully connected network with 24 neurons in the hidden layer for both the value and policy networks. The activations of all hidden units are ReLu, while output layers of the value function approximation networks and policy networks use Linear and Softmax activations, respectively. The same policy and value network structures are used for each driver agent and the platform agent in the MAC algorithm. For both algorithms, the learning rate of the policy network is set at 0.001, and the learning rate of the value network is set at 0.01.

We consider two experiments. First, the platform implements a non-subsidy strategy (i.e., $\beta = 0$). We use different numbers of drivers (i.e., $M = 1, 10, 50,$ and 100) in the market to evaluate the performance (in terms of achieved total rewards) and efficiency (in terms of computation time) of the MAC and RAC algorithms. Second, we assume there are 100 drivers serving in the market and the platform must design the spatial-temporal subsidy strategy to maximize its total objective value, which is a weighted sum of net revenue and service rate (see Eq. (A.10)). A range of subsidy rates (i.e., $\beta = 0, 2, 4, 6,$ and 8 CNY per ride) and weights of the service rate (i.e., $\mu = 0$, $20{,}000$, and $40{,}000$ CNY) are tested[13]. This is to investigate how the spatial-temporal subsidy affects the self-relocation strategies of drivers, as well as the supply-demand situation, and examine how the platform's subsidy strategy varies with different weights for service rate. The execution programming codes for the two experiments are the same except for the settings of number of drivers and subsidy rates. Therefore, the specific amount of subsidies have no impact on the computational time and the results in Section 5.2 well support the performance of the proposed algorithms.

5.2 PERFORMANCE OF THE REPRESENTATIVE-AGENT ALGORITHM

In the first experiment, we test the computation time for the RAC and MAC algorithms for drivers to pursue high-rewarding self-relocation strategies without subsidies. Simulation of the environment (i.e., calculating the matching probabilities and sampling the matchings, rewards, and transitions) and reinforcement learning algorithms are conducted on an HP Z4G4 workstation with 12 Inter I7-7800 processors and four 16-GB rams.

---

[12] For the platform, since three zones (2, 4, and 7) have passenger demand, we can ignore zones without demand in set $\boldsymbol{S}$; therefore, there are 2^3 possible states and 2^3 possible actions, and the cardinality of the state-action set is $2^3 \times 2^3 \times 40 = 2560$. For a driver, we consider purely idle states such that a driver must take a relocation action. If the driver is in zone 4 or 7, then they have 3 relocation destinations; once they are in zone 1 or 3, then they have 4 relocation destinations; and if the driver is in zone 2, 5, or 6, they have 5 relocation destinations. Therefore, the cardinality of a driver's state-action set is $(3 \times 2 + 4 \times 2 + 5 \times 3) \times 40 = 1{,}160$.

[13] In both experiments, we assume that drivers are purely idle and uniformly distributed in zones at the beginning of the simulation. The platform would not allow a high subsidy rate that causes low net revenue for a single ride. Since the platform's net revenue for an order from zone 4 to zone 7 is $0.2 \times 10 \times 6 = 12$ CNY ($\alpha = 10$ CNY per time period and $\eta = 0.2$), the maximal subsidy rate is set as 8 CNY per ride and the net revenue after offering a subsidy is $12 - 8 = 4$ CNY per ride.

In Table 1, we present the computation time for one learning epoch, which consists of simulating the order matches, actions, and states in the environment; storing agents' transitions; and updating the knowledge and value networks of agents (i.e., steps 3 and 4 in Algorithms 1 and 2). We note that the RAC algorithm is slower than the MAC algorithm when $M = 1$; this is because the RAC algorithm must derive a comprehensive soft policy that covers all of the state-action sets of the representative driver; in contrast, the MAC algorithm updates 1 driver's policy based on their own experienced states and actions, ignoring policies that are conditional on unvisited states. As M increases, the time for computing the MF state and updating each agent's policy and value networks will get longer, so that the computation time notably increases for the MAC algorithm. By contrast, the RAC algorithm only computes an approximated MF state and updates the policy and value networks for the representative driver, and the computation time gets much longer as M increases. As a result, with $M = 50$ or 100, we note that the RAC algorithm is significantly faster than the MAC algorithm.

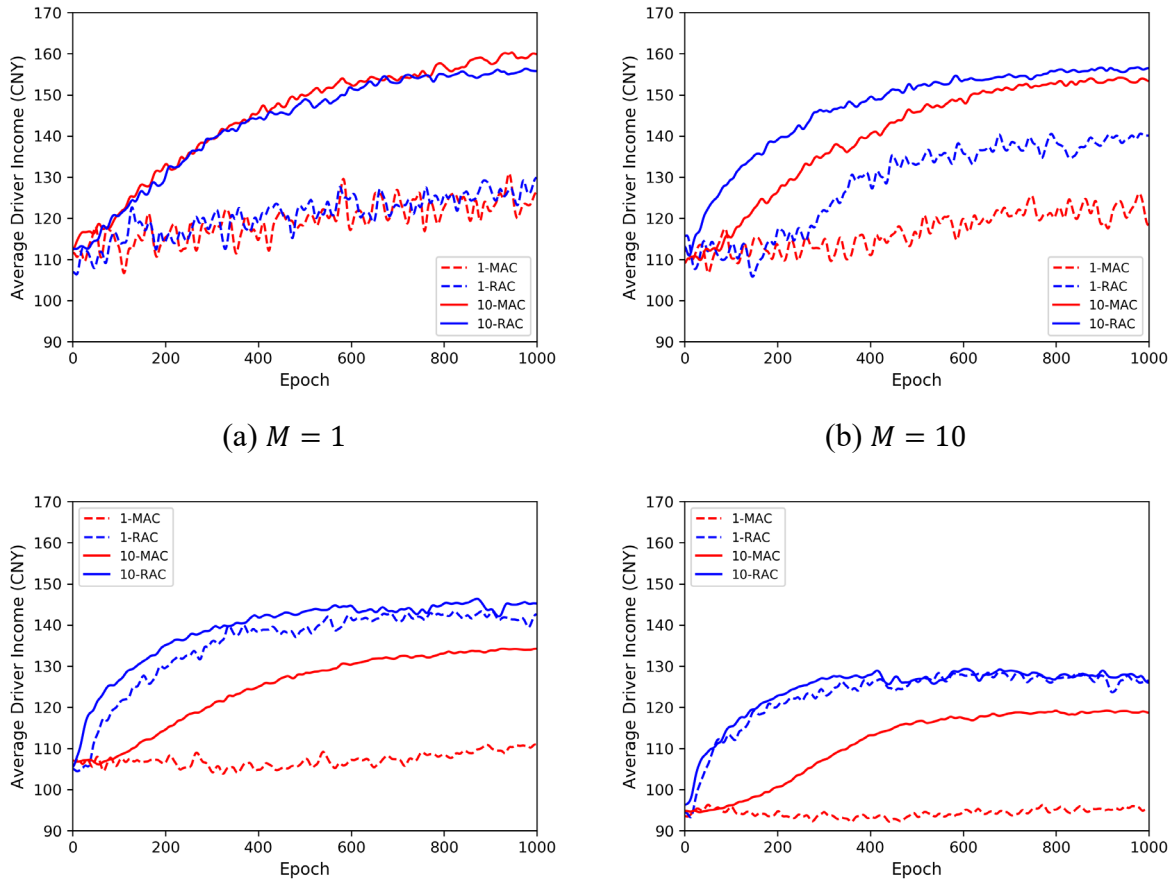
Table 1. Computation Time for One Learning Epoch (seconds)

	$M = 1$	$M = 10$	$M = 50$	$M = 100$
MAC	1.98	27.19	347.3	1654
RAC	2.59	8.17	35.5	85.9

The performance of the two algorithms can be measured by the increase in average driver income (i.e., the average value of the total income for M drivers) over learning epochs. In the experiment, the number of total epochs is $1{,}000$; within each epoch, we conduct either 1 simulation or 10 simulations (i.e., for each n_e, to repeat step 3 in Algorithms 1 and 2 for 10 times before going to step 4, such that more samples of transitions can be generated) to update the policy and value networks. Although global optimality is not guaranteed with reinforcement learning algorithms, the 10-simulation case provides much faster convergence and higher total rewards than the 1-simulation case. The disadvantage is that the computation time for each epoch will be longer as the number of simulations increases.

We illustrate the performance of the two algorithms with different numbers of drivers in Figure 5. When $M = 1$, the MAC algorithm with 10 simulations (referred to as 10-MAC) results in higher average driver income than the RAC algorithm with 10 simulations (10-RAC; see Figure 5(a)). This reflects the ineffectiveness of a soft policy in the simplified MF-MDP model when the number of drivers is small. In addition, average driver income grows slowly with the 1-MAC and 1-RAC algorithms (i.e., by conducting 1 simulation within each epoch), because it is difficult for the transitions observed in 1 simulation to cover a large state-action set of the driver (see Figure 5(a)). When $M = 10$, the RAC algorithm begins to demonstrate its advantages. In Figure 5(b), we see that the 10-RAC algorithm converges much faster and leads to higher average driver income than the 10-MAC algorithm. Unlike the $M = 1$ scenario, the 1-RAC algorithm under $M = 10$ can also achieve high average driver income because more transitions are sampled and used to update the policy and value networks of the representative driver. As M increases to 50, on one hand, the 10-MAC algorithm encounters its bottleneck (at around 132 CNY per driver) and can barely increase average income further (see Figure 5(c)). On the other hand, the 10-RAC algorithm still outperforms the other algorithms, but its gaps in both convergence speed and final average income from the 1-RAC algorithm become smaller. This is because with a large M, 1 simulation during an epoch can generate sufficient samples of transitions. Finally, in Figure 5(d) under $M = 100$, the 1-MAC algorithm never improves average driver income due to insufficient transitions sampled over the large solution space, and we still observe the bottleneck for the 10-MAC algorithm (at around 128 CNY per driver). Also, the advantage of the 10-RAC algorithm over the 1-RAC algorithm becomes negligible. Based

on these findings, we conclude that with a large number of drivers, the RAC algorithm is capable of identifying policies to further improve average total rewards compared with the MAC algorithm on a small-scale network. In addition, a small number of simulations within one epoch is sufficient for the RAC algorithm to quickly converge to a policy that leads to high average total rewards. We aim to examine the MAC and RAC algorithms on real-world networks in near future studies.



(a) $M = 1$ (b) $M = 10$

(c) $M = 50$ (d) $M = 100$

Figure 5. Performance of the MAC and RAC algorithms

5.3 Spatial-temporal subsidies and drivers' self-relocation

In the second experiment, we retain the ride-sourcing market that contains 100 drivers and let the platform to pursue rewardable spatial-temporal subsidy strategies with some predefined β. As described in Section 5.1, we assume the platform assigns weights to the service rate in the objective and adopts different subsidy rates. Namely, a zero weight (e.g., $\mu = 0$) for the service rate indicates that the platform is only concerned with net revenue; a medium weight (e.g., $\mu = 20{,}000$) implies a balance between net revenue and the service rate; and a large weight (e.g., $\mu = 40{,}000$) indicates that the platform mainly focuses on the service rate in the objective. As stated previously, the subsidy rates β range from 0 to 8 CNY per ride in steps of 2 CNY per ride. For each combination of parameters (i.e., μ and β), we pursue the platform's optimal subsidy strategy with respect to the total objective value in terms of drivers' self-relocation. We use the RAC algorithm to solve the simplified MF-MDP model for different combinations of μ and β. To balance computational cost and performance (i.e., for both the platform and the

representative driver), the number of epochs is set at 1,000 and 2 simulations are conducted within each epoch (i.e., a 2-RAC algorithm is adopted).

Table 2. Results with Spatial-temporal Subsidies for the Entire Horizon

(a) Total objective value (weighted sum of net revenue and service rate) for the platform

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	3,189	3,186	3,179	3,190	3,194
$\mu = 20{,}000$	13,616	13,735	13,859	13,930	13,689
$\mu = 40{,}000$	24,112	24,490	24,677	24,830	24,964

(b) Average driver income (CNY per driver)

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	127.55	127.49	127.35	127.72	127.87
$\mu = 20{,}000$	127.84	131.61	134.27	134.64	128.80
$\mu = 40{,}000$	127.68	132.43	136.55	136.16	136.10

(c) Total net revenue for the platform (CNY)

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	3,189	3,186	3,179	3,190	3,194
$\mu = 20{,}000$	3,196	3,095	3,159	3,110	3,189
$\mu = 40{,}000$	3,192	3,050	3,077	2,950	2,804

(d) Service rate in the market

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	51.5%	51.3%	50.8%	51.1%	51.3%
$\mu = 20{,}000$	52.1%	53.2%	53.5%	54.1%	52.5%
$\mu = 40{,}000$	52.3%	53.6%	54.0%	54.7%	55.4%

(e) Total subsidies offered by the platform (i.e., earned by all drivers) (CNY)

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	0.0	0.6	4.2	2.4	3.8
$\mu = 20{,}000$	0.0	117.7	158.4	210.0	29.6
$\mu = 40{,}000$	0.0	208.3	269.2	356.2	479.2

(f) Average subsidies per matched order (CNY)

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	0.00	0.00	0.01	0.01	0.01
$\mu = 20{,}000$	0.00	0.33	0.44	0.58	0.08
$\mu = 40{,}000$	0.00	0.58	0.75	0.97	1.29

(g) Average number of driver delivery/pickup time periods

	$\beta = 0$	$\beta = 2$	$\beta = 4$	$\beta = 6$	$\beta = 8$
$\mu = 0$	15.9/3.0	15.9/3.0	15.9/2.9	16.0/2.9	16.0/2.9
$\mu = 20{,}000$	16.0/3.0	16.3/3.0	16.6/3.1	16.6/3.1	16.1/3.0
$\mu = 40{,}000$	16.0/2.9	16.3/3.1	16.7/3.2	16.6/3.2	16.4/3.1

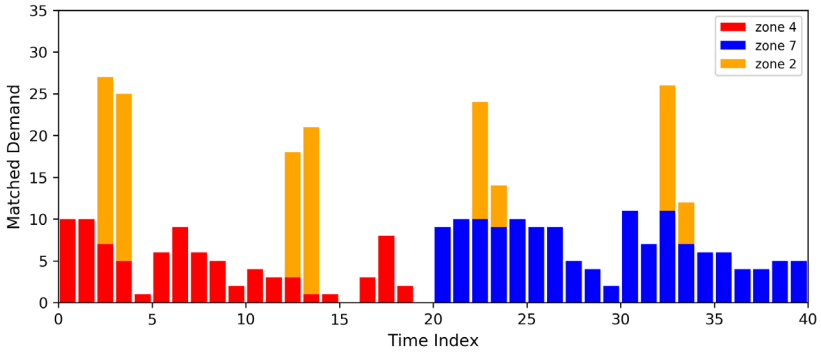
Results of this experience are illustrated in Table 2. From the $1{,}000$ epochs with fixed μ and β in the 2-RAC algorithm, we select 5 epochs with the highest platform objective values and summarize the average metrics[14]. The relation between β and the metrics with a fixed μ can be obtained; the total subsidies offered (i.e., Table 2(e)) reflect the platform's subsidy strategy.

When $\mu = 0$, the total amount of subsidies offered to drivers under different values of β is small (i.e., Table 2(e)); this indicates that the platform prefers not to provide subsidies, and drivers pursue high-rewarding self-relocation strategies without subsidies. As a result, the total objective value for the platform, total net revenue for the platform, average driver income, and service rate (i.e., Tables 2(a)–2(d)) across different values of β are more or less the same. Note that the reinforcement learning algorithm cannot guarantee the global optimal, and the small differences between the values are mainly due to simulation noise. The "non-subsidy" result under $\mu = 0$ is foreseeable for two reasons: (1) the commission rate is low, so that the platform, which only retains a positive but small net revenue for an order, might not afford a large subsidy rate; and (2) a small subsidy rate might not motivate enough drivers to change their self-relocation strategies in order to gain sufficient benefits from alleviating supply-demand imbalance. Therefore, the increased commission withheld by the platform cannot cover the subsidies offered to drivers, which leads to a loss in net revenue while implementing subsidy strategies.

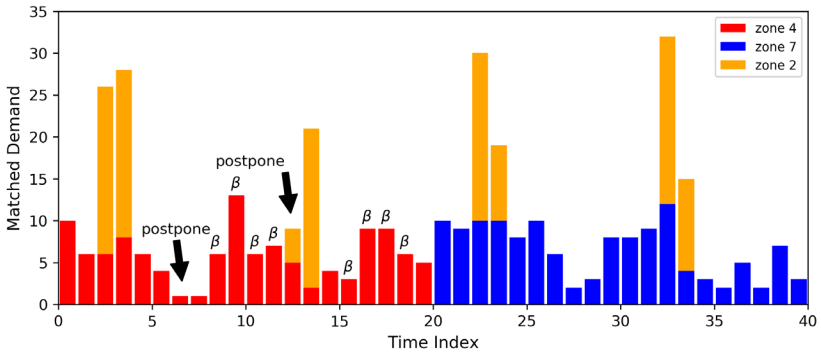
If the platform has a balanced weight with $\mu = 20{,}000$, its total objective value first increases and then decreases with β (see Table 2(a)). Based on Table 2(d) and Table 2(e), we note that as β increases from 0 CNY to 6 CNY per ride, the increased subsidy improves the service rate; however, once $\beta = 8$ CNY per ride, the subsidy becomes cost-ineffective because the benefits gained from the enhanced weighted service rate cannot offset the revenue loss caused by subsidy provisions, and thus the platform is inclined to adopt a non-subsidy strategy.

If the platform mainly prefers a high service rate (i.e., $\mu = 40{,}000$), a subsidy less than or equal to 8 CNY per ride is always cost-effective for improving the total objective value by reshaping drivers' self-relocation strategies and increasing the service rate (see Table 2(a), Table 2(d), and Table 2(e)).

[14] Table 2(a), 2(c) and 2(d) shows the total reward, net revenue, and service rate of the platform, respectively, 2(b) the average income (net revenue) of drivers (i.e., total income divided by 100 drivers), 2(g) the average time periods in delivery/pickup task for drivers (i.e., total number of delivery/pickup times divided by 100 drivers), 2(e) and 2(f) are subsidy related metrics. A value of 0.6 in Table 2(e) under $\mu = 0$ and $\beta = 2$ is obtained. This is because for the 5 epochs (i.e., a total of 10 simulations) with the highest platform's objective values, the total amount of subsidies offered by the platform to all the drivers in the entire horizon is 6 CNY; dividing 6 by 10 simulations, we get 0.6.



(a) Baseline scenario



(b) Subsidy scenario

Figure 6. Served Rides across the Time Horizon and Zones

We refer to $\mu = 0$ and $\beta = 0$ as the baseline scenario, in which the platform's objective consists of only net revenue, and refer to $\mu = 40{,}000$ CNY and $\beta = 8$ CNY per ride as the subsidy scenario in which the platform focuses more on the service rate. Comparing the subsidy scenario with the baseline scenario, spatial-temporal subsidies can lead to a 6.7% and 7.6% increase in the average driver income and the service rate, respectively (see Table 2(b) and Table 2(d)). We show the spatial-temporal number of passengers served (i.e., matched demand) in Figure 6. Time periods with subsidies are denoted using β in Figure 6(b): The platform provides subsidies at zone 4 at time periods 9–12 and 16–19. Motivated by the subsidy, some drivers "postpone" service by idle cruising before the target (i.e., demand-hot) zone is subsidized but relocating to (and thus arriving at) the target zone in time periods with subsidies. As denoted by "postpone" in Figure 6(b), fewer passengers are served at time periods 7–8 and 13 due to the idle cruising and postponing phenomenon; instead, more passengers are served during time periods with subsidies. There are 148 passengers served during time periods 1–15 under both scenarios. In contrast, due to the postponement of services, the number of passengers served after time index 15 notably increases from 195 to 223. These results imply that a platform with an emphasis on service rate has the foresight to mitigate the imbalance between driver supply and passenger demand.

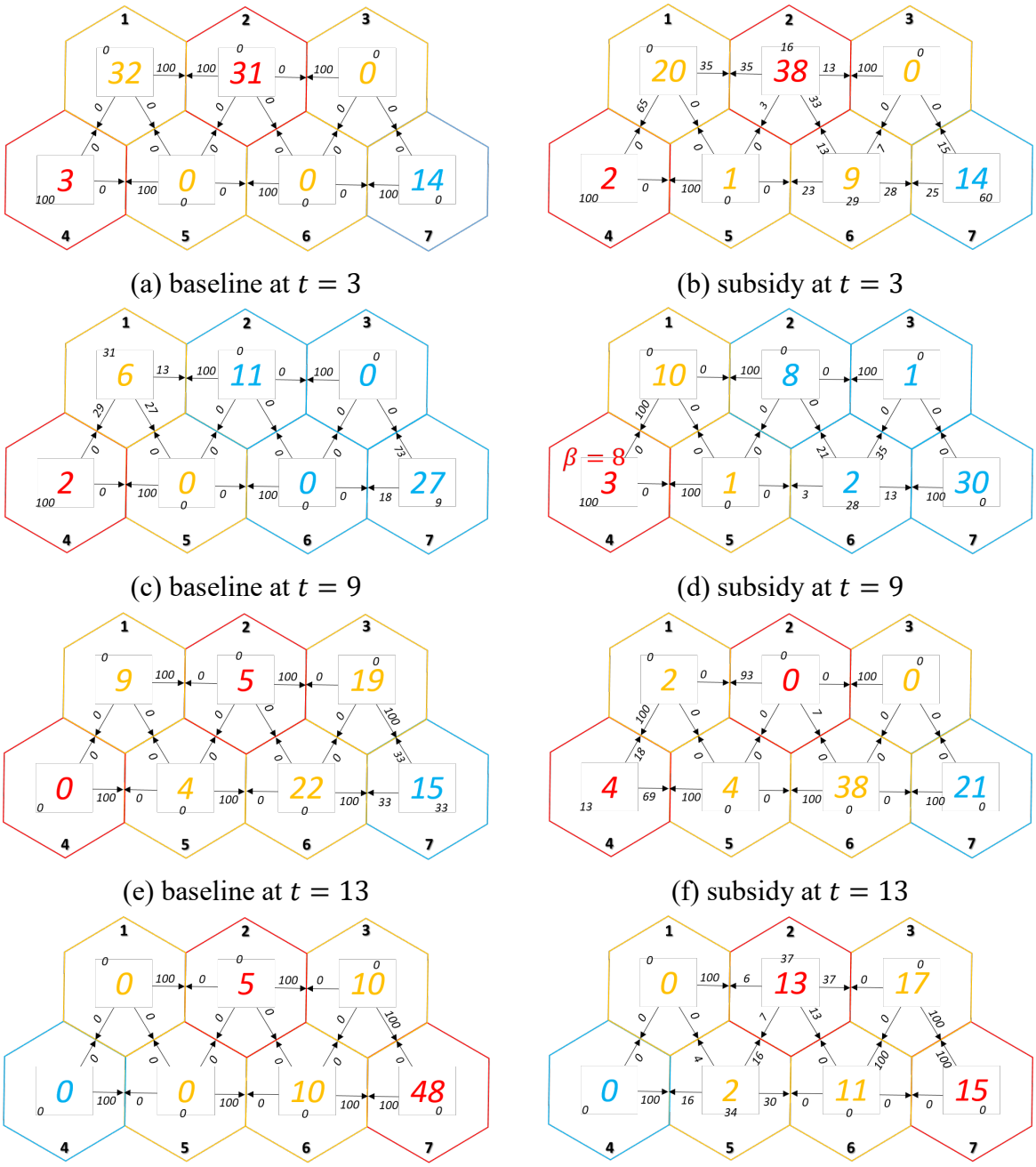


Figure 7. Spatial-temporal idle supply and self-relocation policies.

To better understand how drivers' self-relocation strategies are affected in the subsidy scenario, we provide the spatial-temporal idle driver supply and soft self-relocation policies in the simplified MF-MDP model in Figure 7. Black numbers at the top/bottom of the zones represent zone IDs; colored numbers at the center of each zone denote the numbers of idle drivers; and arrows with small underlined numbers denote relocation destinations and corresponding proportions in percentage (i.e., the soft policy). Blue, yellow, and red represent zones with a low matching probability (in demand-cold zones), a medium matching probability (in zones adjacent to demand-hot zones), and a high matching probability (in demand-hot zones), respectively. Note that the instances reported in Figure 7 are from a single simulation with the highest objective value for the platform. At $t = 3$, idle drivers in the baseline scenario move to either zone 2 or zone 4 to pick up passengers (see Figure 7(a)); in the subsidy scenario, some drivers in zones 6

and 7 have diverse relocation directions (e.g., drivers in zone 7 have a 60% chance of staying). One reason might be that they first cruise around and wait, then try to arrive at zone 4 at $t \in \{9,10,11,12\}$ to earn subsidies. Because of this phenomenon, at $t = 9$, the number of idle drivers adjacent to zone 4 in the subsidy scenario is notably higher than in the baseline scenario. The postponing phenomenon also happens at $t = 13$, when some drivers perceive the upcoming subsidies at $t \in \{16,17,18,19\}$ and decide not to immediately serve passengers in zone 2 (see Figure 7(f) and Figure 7(b)). The benefits of the postponing phenomenon can be partially observed in Figures 7(g)–7(h). In the baseline scenario, drivers in the first half of the operational horizon keep relocating to zones 2 and 4 to serve passengers. This results in a shortage of supply during time periods 9–11 and 14–17 (see Figure 6(a)), as well as peak arrivals of idle drivers at zone 7 at an early time in the second half of the operational horizon (e.g., time period 23 in Figure 7(g)). In contrast, in the subsidy scenario, some drivers are inclined to idly cruise and postpone their services so that the supply becomes smooth across zones and time periods, especially during the second half of the operational horizon. Consequently, the supply-demand imbalance in this case study is alleviated due to the implementation of spatial-temporal subsidies.

6 DISCUSSION OF SUBSIDY SCHEMES

The numerical studies in Section 5 offer in-depth insights for ride-sourcing platforms about the effectiveness of a uniform subsidy scheme in addressing supply-demand imbalance. In such a scheme, the platform predetermines the amount of subsidy per order (i.e., subsidy rate) and offers this amount of subsidy to drivers once they are matched with passengers whose origins are the subsidized zones. The strategy of spatial-temporal subsidies under the uniform scheme is largely affected by the objective of the platform. If the platform only cares about the immediate net revenue, the effectiveness of this subsidy scheme could be limited. This is because for a single order, the subsidy rate generally does not exceed the commission withheld by the platform (otherwise the platform earns negative net revenue for an order). Thus, the amount of subsidy offered to a driver is much smaller than they earn from the trip fare and is unattractive to drivers, who may not be motivated to move to the designated zones. In this case, offering subsidies could cause a loss in net revenue because the subsidy provision is higher than the commission gain; therefore, the platform would prefer a non-subsidy strategy. By contrast, if the platform pursues a high service rate (i.e., number of passengers served), it would like to offer a subsidy sufficient to stimulate drivers to demand-hot zones despite the reduction in immediate revenue. The latter case might occur when a platform expands its business and competes with others. An example is the price war between DiDi and Uber in mainland China in 2016 before they consolidated.

However, the uniform subsidy scheme is not superior in improving the service rate and net revenue of the platform simultaneously. This is because the platform provides the same amount of subsidies to different drivers: (1) those who already have desirable self-relocation strategies such that they could relocate to demand-hot zones even without subsidies, and (2) those who are incentivized by subsidies but would not relocate to demand-hot zones if no subsidies were provided. Therefore, subsidies offered to drivers belonging to the first type would not generate more net revenue for the platform, and only subsidies offered to drivers in the second type would help mitigate the supply-demand imbalance and improve the service rate.

To thoroughly examine the designs of spatial-temporal subsidies for drivers with certain self-relocation strategies, a number of tasks must be relayed left to the future studies. Different subsidy schemes must be examined and evaluated based on the ride-sourcing MF-MDP model. Below, we provide a few sample schemes with the potential advantages and feasibility:

- Surge subsidy (or zone-based) scheme, in which the platform provides higher subsidies at zones with greater supply-demand imbalance. Once there is super large passenger demand in a hot area, the platform could offer irresistible subsidies to drivers who relocate to and then serve passengers in the area. The revenue loss due to high subsidy provision could be offset by the improvement in the service rate, since sufficient drivers will be attracted to the hot area to accommodate the high service needs.
- Distance-based subsidy scheme, in which the subsidy rate is proportional to the travel distance of the ride order or a subsidy is applied only if the travel distance of the order exceeds some threshold. Such a scheme could be beneficial once there is an insufficient supply of long-distance passenger demand.
- Origin-destination-based subsidy scheme, in which the platform offers heterogeneous subsidies based on the origin and destination of the ride order. For instance, the platform could provide a high subsidy for trips that originate at a demand-cold area and terminate at a demand-hot area. Consequently, the supply-demand imbalance could be improved as the overall driver supply at demand-cold areas is incentivized to relocate to demand-hot areas. We can employ the MF-MDP model to determine the critical rules with respect to subsidy rates and the characteristics of origins/destinations.
- Performance-based (or driver-based) subsidy scheme, in which the platform offers subsidies according to drivers' performance and behaviors. For instance, the platform could only offer subsidies to drivers who would not relocate to demand-hot zones without incentives. Although this subsidy scheme can reduce the subsidy provided to drivers with high-rewarding self-relocation strategies, it could be controversial due to potential discrimination concerns.

All of these subsidy schemes merit analysis using the MF-MDP model to gain comprehensive insights into the pros and cons of diverse spatial-temporal subsidies in ride-sourcing markets. Furthermore, we would examine spatial-temporal subsidies in more realistic scenarios in terms of a large-scale zone network, passenger demand derived from actual data, and a flexible setting of subsidy levels. Real-world public datasets can be used to generate large-scale ride-sourcing scenarios. Although the current edge-based matching rules (Eqs. A.(6)–A.(9)) analytically capture the cross-zone matching feature of ride-sourcing markets, it can be computational inefficient for large-scale analyses when calculating the joint probability distribution of matching results. We aim to improve the efficiency of the edge-based matching rule for real-world scenarios in future studies. In addition, the predefined subsidy rates (i.e., either 0 or β) in this paper could underestimate the effectiveness of a subsidy due to the inflexibility of implementing heterogeneous subsidies at different locations and with different traveling distances. Instead, we can predefine a few subsidy levels and apply reinforcement learning (e.g., the RAC) algorithms to pursue the optimal subsidy level in each time period to maximize the total rewards.

7 CONCLUSIONS

In this paper, we propose a generalized MF-MDP model to capture sequential and interactive decision processes in a ride-sourcing environment with the platform as the major agent and multiple drivers as minor agents. The MF-MDP model is particularly suitable for research questions in which the major agent (platform) and minor agents (drivers) have distinct objectives. The decisions/actions of the platform can directly affect the drivers' states, while the drivers' actions can influence the platform's state and drivers' average state, which is referred to as the MF state. An approximation of the MF state is employed to simplify the model, such that we only need to optimize the policies for the platform and one representative driver instead of

the policies for the platform and all individual drivers (as in the standard MF-MDP model). Consequently, computational complexity can be notably reduced when there are a large number of drivers in the environment.

In particular, we adopt the MF-MDP model to design the platform's spatial-temporal subsidy strategies with a predefined subsidy rate for drivers who have self-relocation strategies. A representative-agent reinforcement learning algorithm is proposed to solve the MF-MDP model. Using numerical studies, we demonstrate that due to the significant reduction of the number of agents and solution space, the representative-agent algorithm demonstrates significant computational advantages and fast convergence and achieves higher rewards, compared with the conventional multi-agent algorithm. In addition, we investigate the potential impact of spatial-temporal subsidies on drivers' self-relocation strategies and the resulting platform's objective values and drivers' income. Based on a uniform subsidy scheme, our results suggest that subsidies can improve the service level (number of passengers served) by incentivizing idle drivers to locations with overfull passenger demand and insufficient driver supply. On one hand, if the platform only pursues net revenue (measured commission withheld from trip fares by the platform minus the amount of subsidies offered to drivers), a subsidy strategy with predefined subsidy levels is cost-ineffective due to a large reduction in net revenue from a single order versus a small increase in the number of passengers served. On the other hand, when the platform pays more attention to the number of passengers served (in order to improve the customer satisfaction rate), it is more willing to offer sufficient subsidies to stimulate drivers to demand-hot zones and achieve a better supply-demand balance. In this case, the spatial-temporal subsidy strategy leads to a win-win situation in which both average driver income and the platform's total objective value are notably improved.

This paper makes three major contributions to the literature. First, unlike previous ride-sourcing MDP models that assume the platform has full control of drivers, the proposed MF-MDP model considers interactive decision processes between the platform and drivers. Second, the simplified MF-MDP model with the representative-agent algorithm is shown to be a good alternative to multi-agent MDP models, which are generally computationally expensive. The mean-field approximation not only saves computational resources but also achieves higher rewards with a faster convergence in our research problem. This is mainly due to the special characteristic of a ride-sourcing market, in the sense that the number of drivers is so large that the platform can consider the mean-field state of all drivers without tracking the individual state of each driver. Third, our model describes the interactions between the platform's spatial-temporal subsidy strategies and drivers' self-relocation strategies; in contrast, most previous studies investigate either the impact of platform-side incentives or idle-vehicle relocation problems separately without considering their interplay.

There are several important directions for future research. First, some deep learning-based algorithms and MF simulation approaches can be developed to further enhance performance and reduce computational complexity. We are particularly interested in developing edge-based matching rules that are both capable of depicting cross-zone matching processes and powerful for large-scale multi-agent problems in practically relevant scenarios. Second, based on the generalized ride-sourcing MF-MDP model, we will examine the impacts of other subsidy schemes, such as surge subsidy schemes over time and zones, distance-based subsidy schemes, and origin-destination-based subsidy schemes. These subsidy schemes are expected to mitigate supply-demand imbalance more efficiently than the uniform subsidy scheme that offers the same amount of subsidy to drivers upon matches with passengers from subsidized regions. Third, the framework can be extended to investigate ride-sourcing markets coupled with public transit services, and identify optimal coordination between ride-sourcing drivers who aim to improve their earnings by self-relocation and public transit operators who attempt to design transit

schedules to improve transit usage. For example, the platform's knowledge of bus services' timeline could incentivize drivers to relocate to transit stations at the appropriate time, as a result of which the cooperation and substituting effect between ride-sourcing and public transit services could be enhanced.

ACKNOWLEDGEMENT

This work is partially supported by the Hong Kong Research Grants Council under projects HKUST16208920 and NHKUST627/18, and is partially supported by the Hong Kong University of Science and Technology–Didi Chuxing (HKUST-DiDi) Joint Laboratory. The third author gratefully acknowledges support by the Lee Kong Chian (LKC) Fellowship awarded by Singapore Management University. The opinions in this paper do not necessarily reflect the official views of the HKUST-DiDi Joint Laboratory. The authors are responsible for all statements. The authors would also like to acknowledge all of the reviewers for their constructive comments.

REFERENCES

Bai, J., So, K.C., Tang, C.S., Chen, X. and Wang, H., 2019. Coordinating supply and demand on an on-demand service platform with impatient customers. *Manufacturing & Service Operations Management*, *21*(3), pp. 556-570.

Braverman, A., Dai, J.G., Liu, X. and Ying, L., 2019. Empty-car routing in ridesharing systems. *Operations Research*, *67*(5), pp. 1437-1452.

Cachon, G.P., Daniels, K.M. and Lobel, R., 2017. The role of surge pricing on a service platform with self-scheduling capacity. *Manufacturing & Service Operations Management*, *19*(3), pp. 368-384.

Castillo, J.C., Knoepfle, D. and Weyl, G., 2017. Surge pricing solves the wild goose chase. In *Proceedings* of *the 2017 ACM Conference on Economics and Computation* (pp. 241-242). ACM.

Gao, Y., Jiang, D. and Xu, Y., 2018. Optimize taxi driving strategies based on reinforcement learning. *International Journal of Geographical Information Science*, *32*(8), pp. 1677-1696.

Gomes, D.A., 2014. Mean field games models—a brief survey. *Dynamic Games and Applications*, *4*(2), pp. 110-154.

Hazelton, M.L. and Watling, D.P., 2004. Computation of equilibrium distributions of Markov traffic-assignment models. *Transportation Science*, *38*(3), pp. 331-342.

Huang, M., 2010. Large-population LQG games involving a major player: the Nash certainty equivalence principle. *SIAM Journal on Control and Optimization*, *48*(5), pp. 3318-3353.

Huang, M., 2012, May. Mean field stochastic games with discrete states and mixed players. In *International Conference on Game Theory for Networks* (pp. 138-151). Springer, Berlin, Heidelberg.

Huang, M., Caines, P.E. and Malhamé, R.P., 2007. Large-population cost-coupled LQG problems with nonuniform agents: individual-mass behavior and decentralized ε-Nash equilibria. *IEEE Transactions on Automatic Control*, *52*(9), pp. 1560-1571.

Huang, M., Malhamé, R.P. and Caines, P.E., 2006. Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle. *Communications in Information & Systems*, *6*(3), pp. 221-252.

Hwang, R.H., Hsueh, Y.L. and Chen, Y.T., 2015. An effective taxi recommender system based on a spatio-temporal factor analysis model. *Information Sciences*, *314*, pp. 28-40.

Jin, J., Zhou, M., Zhang, W., Li, M., Guo, Z., Qin, Z., Jiao, Y., Tang, X., Wang, C., Wang, J. and Wu, G., 2019. CoRide: Joint order dispatching and fleet management for multi-scale ride-hailing platforms. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (pp. 1983-1992).

Ke, J., Qin, X., Yang, H., Zheng, Z., Zhu, Z. and Ye, J., 2021. Predicting origin-destination ride-sourcing demand with a spatio-temporal encoder-decoder residual multi-graph convolutional network. *Transportation Research Part C: Emerging Technologies, 122*, 102858.

Ke, J., Yang, H., Li, X., Wang, H. and Ye, J., 2020. Pricing and equilibrium in on-demand ride-pooling markets. *Transportation Research Part B: Methodological, 139,* pp. 411-431.

Ke, J., Yang, H., Zheng, H., Chen, X., Jia, Y., Gong, P. and Ye, J., 2019. Hexagon-based convolutional neural network for supply-demand forecasting of ride-sourcing services. *IEEE Transactions on Intelligent Transportation Systems*, *20*(11), pp. 4160-4173.

Ke, J., Zheng, H., Yang, H. and Chen, X., 2017. Short-term forecasting of passenger demand under on-demand ride services: a spatio-temporal deep learning approach. *Transportation Research Part C: Emerging Technologies*, *85*, pp. 591-608.

Li, M., Qin, Z., Jiao, Y., Yang, Y., Wang, J., Wang, C., Wu, G. and Ye, J., 2019, May. Efficient ridesharing order dispatching with mean field multi-agent reinforcement learning. In *The World Wide Web Conference* (pp. 983-994). ACM.

Lin, K., Zhao, R., Xu, Z. and Zhou, J., 2018, July. Efficient large-scale fleet management via multi-agent deep reinforcement learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 1774-1783). ACM.

Lyu, G., Cheung, W.C., Teo, C.P. and Wang, H., 2019. Multi-objective online ride-matching. *Available at SSRN 3356823*.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D and Kavukcuoglu, K. 2016, June. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning* (pp. 1928-1937).

Mao, C., Liu, Y., & Shen, Z. J. M., 2020. Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach. *Transportation Research Part C: Emerging Technologies*, 115, 102626.

Qian, X., Zhang, W., Ukkusuri, S.V. and Yang, C., 2017. Optimal assignment and incentive design in the taxi group ride problem. *Transportation Research Part B: Methodological*, *103*, pp. 208-226.

Rong, H., Zhou, X., Yang, C., Shafiq, Z. and Liu, A., 2016, October. 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* (pp. 2329-2334). ACM.

Shou, Z. and Di, X., 2020a. Multi-Agent Reinforcement Learning for Dynamic Routing Games: A Unified Paradigm. *arXiv preprint arXiv:2011.10915*.

Shou, Z. and Di, X., 2020b. Reward design for driver repositioning using multi-agent reinforcement learning. *Transportation research part C: Emerging Technologies*, *119*, 102738.

Shou, Z., Di, X., Ye, J., Zhu, H., Zhang, H., and Hampshire, R., 2020. Optimal passenger-seeking policies on E-hailing platforms using Markov decision process and imitation learning. *Transportation Research Part C: Emerging Technologies*, 111, 91-113.

Sun, H., Wang, H. and Wan, Z., 2019a. Model and analysis of labor supply for ride-sharing platforms in the presence of sample self-selection and endogeneity. *Transportation Research Part B: Methodological*, *125*, pp. 76-93.

Sun, H., Wang, H. and Wan, Z., 2019b. Flexible labor supply behavior on ride-sourcing platforms. Available at SSRN: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3357365.

Taylor, T.A., 2018. On-demand service platforms. *Manufacturing & Service Operations Management*, *20*(4), pp. 704-720.

Wang, H. and Yang H., 2019. Ride-sourcing systems: a framework and review. *Transportation Research Part B: Methodological*, *129*, pp. 122-155.

Wang, Z., Qin, Z., Tang, X., Ye, J. and Zhu, H., 2018, November. Deep reinforcement learning with knowledge transfer for online rides order dispatching. In *2018 IEEE International Conference on Data Mining (ICDM)* (pp. 617-626). IEEE.

Xu, Z., Li, Z., Guan, Q., Zhang, D., Li, Q., Nan, J., Liu, C., Bian, W. and Ye, J., 2018, July. 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* (pp. 905-913). ACM.

Xu, Z., Yin, Y. and Zha, L., 2017. Optimal parking provision for ride-sourcing services. *Transportation Research Part B: Methodological*, *105*, pp. 559-578.

Yang, H., Fung, C.S., Wong, K.I. and Wong, S.C., 2010. Nonlinear pricing of taxi services. *Transportation Research Part A: Policy and Practice*, *44*(5), pp. 337-348.

Yang, H., Qin, X., Ke, J. and Ye, J., 2020a. Optimizing matching time interval and matching radius in on-demand ride-sourcing markets. *Transportation Research Part B: Methodological*, *131*, pp. 84-105.

Yang, H., Shao, C., Wang, H. and Ye, J., 2020b. Integrated reward scheme and surge pricing in a ridesourcing market. *Transportation Research Part B: Methodological*, *134*, pp. 126-142.

Yang, H., Wong, S.C. and Wong, K.I., 2002. Demand–supply equilibrium of taxi services in a network under competition and regulation. *Transportation Research Part B: Methodological*, *36*(9), pp. 799-819.

Yao, H., Wu, F., Ke, J., Tang, X., Jia, Y., Lu, S., Gong, P., Ye J. and Li, Z., 2018. Deep multi-view spatial-temporal network for taxi demand prediction. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yu, X., Gao, S., Hu, X. and Park, H., 2019. A Markov decision process approach to vacant taxi routing with e-hailing. *Transportation Research Part B: Methodological*, *121*, pp. 114-134.

Zha, L., Yin, Y., Xu, Z., 2018. Geometric matching and spatial pricing in ride-sourcing markets. *Transportation Research Part C: Emerging Technologies*, *92*, pp. 58-75.

Zha, L., Yin, Y. and Yang, H., 2016. Economic analysis of ride-sourcing markets. *Transportation Research Part C: Emerging Technologies*, *71*, pp. 249-266.

Zhang, L., Hu, T., Min, Y., Wu, G., Zhang, J., Feng, P., Gong, P. and Ye, J., 2017. A taxi order dispatch model based on combinatorial optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 2151-2159). ACM.

Zhu, Z., Mardan, A., Zhu, S. and Yang, H., 2021a. Capturing the interaction between travel time reliability and route choice behavior based on the generalized Bayesian traffic model. *Transportation Research Part B: Methodological*, 143, 48-64.

Zhu, Z., Tang, L., Xiong, C., Chen, X. and Zhang, L., 2019a. The conditional probability of travel speed and its application to short-term prediction. *Transportmetrica B: Transport Dynamics*, *7*(1), pp. 684-706.

Zhu, Z., Sun, L., Chen, X. and Yang, H., 2021b. Integrating probabilistic tensor factorization with Bayesian supervised learning for dynamic ridesharing pattern analysis. Accepted by *Transportation Research Part C: Emerging Technologies*.

Zhu, Z., Qin X., Ke, J., Zheng, Z. and Yang, H., 2020. Analysis of multi-modal commute behavior with feeding and competing ridesplitting services. *Transportation Research Part A: Policy and Practice*, 132, 713-727.

Zhu, Z., Zhu, S., Zheng, Z. and Yang, H., 2019b. A generalized Bayesian traffic model. *Transportation Research Part C: Emerging Technologies*, *108*, pp. 182-206.

Zuniga Garcia, N., 2019. *Spatial pricing empirical evaluation of ride-sourcing trips using the graph-fussed lasso for total variation denoising* (doctoral dissertation).

APPENDIX A: FORMULAS FOR THE SPECIFIC MF-MDP MODEL

In this appendix we provide detailed formulas for state transition laws, matching probability, and rewards in the MF-MDP model developed in Section 4.

First, we illustrate the state transition laws of the platform and a driver in the standard MF-MDP model (or the representative driver in the simplified MF-MDF model). With the state vector $\boldsymbol{s} = [t, s_1, \dots, s_O]$ and the action vector $= [a_1, \dots, a_O]$, the state transition law for the platform is given by

$$Q(\boldsymbol{s}'|\boldsymbol{s}, \boldsymbol{a}) = \begin{cases} 1 & , \boldsymbol{s}' = [t+1, a_1, \dots, a_O], \forall \boldsymbol{a} \in \boldsymbol{A} \\ 0 & , otherwise \end{cases} \tag{A.1}$$

An intuitive explanation of Eq. (A.1) is that the state of the platform for the next time index equals its action vector.

With a state vector $\boldsymbol{s}_d = [t, s_{d1}, s_{d2}, s_{d3}, s_{d4}, s_{d5}]$ and an action $a_d \in \boldsymbol{J}_{s_{d3}}$, the state transition law for a driver has different formulas according to the current task s_{d1} and remaining time s_{d2}. Note that for the following formulas, we need $\boldsymbol{s}_d, \boldsymbol{s}'_d \in \boldsymbol{S}_d$, and $\boldsymbol{h}_d \in \boldsymbol{H}_d$.

- The driver is picking up a passenger, i.e., $\boldsymbol{s}_d = [t, 1, \tau, o, o', o'']$, $\tau > 0$, $o, o', o'' \in \{1,2, \dots, O\}$, and $o \neq o' \neq o''$

$$Q_d(\boldsymbol{s}'_d|\boldsymbol{s}_d, \boldsymbol{h}_d, a_d) = \begin{cases} 1 & , \boldsymbol{s}'_d = [t+1, 1, \tau - 1, o, o', o''], \tau > 1 \\ 1 & , \boldsymbol{s}'_d = [t+1, 2, \tau_{o'o''}, o', o', o''], \tau = 1 \\ 0 & , otherwise \end{cases} \quad (A.2)$$

where $\tau_{o'o''}$ denotes the average travel time from zone o' to zone o'', which is exogenous. The first line means that the driver still needs more than one time period to finish the current picking-up task; therefore the new remaining time s'_{d2} decreases by one, while the other dimensions of the state vector remain unchanged. The second line means that the driver is about to finish a picking-up task and will immediately change the task to delivering the passenger; then the new remaining time s'_{d2} becomes the average travel time between the origin zone and the destination zone, the new task s'_{d1} becomes 2, and the new idling zone becomes the current picking-up destination, which is identical to the origin of the passenger (i.e., $s'_{d3} = s_{d4} = o'$).

- The driver is delivering a passenger, i.e., $\boldsymbol{s}_d = [t, 2, \tau, o, o, o']$, $\tau > 0$, $o, o' \in \{1,2, \dots, O\}$, and $o \neq o'$.

$$Q_d(\boldsymbol{s}'_d|\boldsymbol{s}_d, \boldsymbol{h}_d, a_d) = \begin{cases} 1 & , \boldsymbol{s}'_d = [t+1, 2, \tau - 1, o, o, o'], \tau > 1 \\ 1 & , \boldsymbol{s}'_d = [t+1, 0, 0, o', o', o'], \tau = 1 \\ 0 & , otherwise \end{cases} \quad (A.3)$$

The first line means that the driver needs more than one time period to finish the current delivering task and the new remaining time s'_{d2} decreases by one, while the other dimensions of the state vector remain the same. The second line indicates that if the driver is about to finish a delivering task, the new state becomes a purely idle state (i.e., $s'_{d1} = 0$ and $s'_{d2} = 0$). Since there is no passenger order, we let $s'_{d3} = s'_{d4} = s'_{d5} = o'$ for convenience.

- The driver is purely idle, i.e., $\boldsymbol{s}_d = [t, 0, 0, o, o, o]$, and $o \in \{1,2, \dots, O\}$.

$$Q_d(\boldsymbol{s}'_d|\boldsymbol{s}_d, \boldsymbol{h}_d, a_d) = \begin{cases} m_{o,o',o''}(\boldsymbol{h}_d) & , \boldsymbol{s}'_d = [t+1, 1, \tau_{oo'}, o, o', o''] \\ m_{o,o,o''}(\boldsymbol{h}_d) & , \boldsymbol{s}'_d = [t+1, 2, \tau_{oo''}, o, o, o''] \\ u_o(\boldsymbol{h}_d) & , \boldsymbol{s}'_d = [t+1, 0, 0, o, o, o], a_d = o \\ u_o(\boldsymbol{h}_d) & , \boldsymbol{s}'_d = [t+1, 0, \tau_{oa_d} - 1, o, a_d, a_d], a_d \neq o \\ 0 & , otherwise \end{cases} \quad (A.4)$$

where $m_{o,o',o''}(\boldsymbol{h}_d)$ denotes the probability of getting a matched passenger order at zone o with the origin and destination of the passenger being o' and o'', respectively, and $u_o(\boldsymbol{h}_d)$ denotes the probability of not being matched at zone o. Generally, both $m_{o,o',o''}(\boldsymbol{h}_d)$ and $u_o(\boldsymbol{h}_d)$ depend on the MF state of drivers and the exogenous passenger demand. More specific formulas for the probabilities are given later in this appendix. The first line in the equation implies that the origin of the newly matched passenger is different from the driver's current idling zone (i.e., $s'_{d4} = o' \neq s_{d3} = o$); therefore, a picking-up task is needed and we have $s'_{d1} = 1$ and $s'_{d2} = \tau_{oo'}$. In the second line, the matched passenger and the driver are in the same zone (i.e., $s'_{d4} = s_{d3} = o$) and we assume the picking-up process can be ignored; therefore, the driver will directly start to

deliver the passenger (i.e., $s'_{d1} = 2$). The third line indicates that the driver is still not matched and their action is to stay in the current idling zone (i.e., $a_d = o$); therefore the state of the driver will remain unchanged. In the fourth line, the driver is not matched and will relocate to zone a_d; we let $s'_{d4} = s'_{d5} = a_d$ for convenience, and let $s'_{d2} = \tau_{oa_d} - 1$ because we assume the driver is already in the middle of the self-relocating state (i.e., no time is wasted by stopping the vehicle to load or drop off passengers).

- The driver is in a self-relocating state, i.e., $\boldsymbol{s}_d = [t, 0, \tau, o, o', o']$, $\tau > 0$, and $o \neq o'$.

$$Q_d(\boldsymbol{s}'_d|\boldsymbol{s}_d, \boldsymbol{h}_d, a_d) = \begin{cases} m_{o,o'',o'''}(\boldsymbol{h}_d) & ,\boldsymbol{s}'_d = [t+1,1,\tau_{oo''}, o, o'', o'''] \\ m_{o,o',o'''}(\boldsymbol{h}_d) & ,\boldsymbol{s}'_d = [t+1,1,\tau-1, o, o', o'''], \tau > 1 \\ m_{o,o',o'''}(\boldsymbol{h}_d) & ,\boldsymbol{s}'_d = [t+1,2,\tau_{o'o'''}, o', o', o'''], \tau = 1 \\ m_{o,o,o'''}(\boldsymbol{h}_d) & ,\boldsymbol{s}'_d = [t+1,2,\tau_{oo'''}, o, o, o'''] \\ u_o(\boldsymbol{h}_d) & ,\boldsymbol{s}'_d = [t+1,0,\tau-1, o, o', o'], \tau > 1 \\ u_o(\boldsymbol{h}_d) & ,\boldsymbol{s}'_d = [t+1,0,0, o', o', o'], \tau = 1 \\ 0 & ,otherwise \end{cases} \tag{A.5}$$

In the first line, the driver is matched with a passenger whose origin $s'_{d4} = o''$ is different from either the driver's current idling zone $s_{d3} = o$ or the self-relocation destination $s_{d4} = o'$; therefore, the driver begins a picking-up task and moves to zone o'' (i.e., $s'_{d1} = 1$ and $s'_{d4} = o''$). The second line indicates that if the self-relocation destination coincides with the origin zone of the matched passenger (i.e., $s'_{d4} = s_{d4} = o'$), the new remaining time s'_{d2} decreases by one because we regard the driver as already in the middle of the picking-up task (i.e., $s'_{d1} = 1$). To continue with the case $s'_{d4} = s_{d4} = o'$, the third line means that if drivers are leaving the self-relocating state (i.e., $s_{d2} = \tau = 1$), they immediately load the passenger and begin the delivering task (i.e., $s'_{d1} = 2$). In the fourth line, both the matched passenger and driver are in zone o (i.e., $s'_{d4} = s_{d3} = o$) and a delivering task starts. The fifth and sixth lines indicate that the driver is not matched with passengers, such that he/she either remains in the self-relocating state (the fifth line) or becomes purely idle in zone o' (i.e., $s'_{d1} = 0$, $s'_{d2} = 0$, and $s'_{d3} = o'$ for the sixth line).

Note that the MF vector $\boldsymbol{h}_d$ is used in $Q_d(\boldsymbol{s}'_d|\boldsymbol{s}_d, \boldsymbol{h}_d, a_d)$ to calculate the matching probabilities $m_{o,o',o''}(\boldsymbol{h}_d)$ and $u_o(\boldsymbol{h}_d)$. Next, we provide detailed formula related to the number of matches and matching probabilities. According to the edge-based matching rule in Section 4.1, the number of matches near an arbitrary edge $e_{oo'}$, which is denoted by $k_{e_{oo'}}$ is given by

$$k_{e_{oo'}}(\boldsymbol{h}_d) = \min\left\{\frac{M_o(\boldsymbol{h}_d)}{E_o} + \frac{M_{o'}(\boldsymbol{h}_d)}{E_{o'}}, \frac{N_o}{E_o} + \frac{N_{o'}}{E_{o'}}\right\} \tag{A.6}$$

$$M_o(\boldsymbol{h}_d) = M\left(h_{d,[t,0,0,o,o,o]} + \sum_{\tau>0, o' \in \boldsymbol{J}_o/o} h_{d,[t,0,\tau,o,o',o']}\right) \tag{A.7}$$

where E_o denote the number of edges of zone o, and assuming hexagonal zones, the value of E_o is 6 unless the zone is located at the boundary of the network; and h_{d,s_d} is the scalar value in $\boldsymbol{h}_d$ and represents the proportion of drivers in state s_d (i.e., $z^t_{d,s_d} = h_{d,s_d}$ in Eq. (2)), such that $h_{d,[t,0,0,o,o,o]}$ denotes the proportion of purely idle drivers and $h_{d,[t,0,\tau,o,o',o']}$ the proportion of

drivers in self-relocating states at time t (see Eq. (A.5)). Since $k_{e_{oo'}}$ is only used to calculate matching probabilities in this paper, we allow the value of $k_{e_{oo'}}$ to be a non-integer.

Since the numbers of matched passengers and drivers are proportional to the demand and supply near the common edge, and drivers and passengers are uniformly distributed in the zones (see Section 4.1), we have the formulas for $m_{o,o',o''}(\boldsymbol{h}_d)$ and $u_o(\boldsymbol{h})$ as follows:

$$m_{o,o',o''}(\boldsymbol{h}_d) = \begin{cases} \dfrac{1}{E_o}\dfrac{k_{e_{oo'}}(\boldsymbol{h}_d)}{\dfrac{M_o(\boldsymbol{h}_d)}{E_o}+\dfrac{M_{o'}(\boldsymbol{h}_d)}{E_{o'}}}\dfrac{N_{o'}}{N_o+N_{o'}}\dfrac{N_{o'o''}}{N_{o'}}, o' \in \boldsymbol{J}_o/o \\ \left(\sum\limits_{o'''\in \boldsymbol{J}_o/o}\dfrac{1}{E_o}\dfrac{k_{e_{oo'''}}(\boldsymbol{h}_d)}{\dfrac{M_o(\boldsymbol{h}_d)}{E_o}+\dfrac{M_{o'''}(\boldsymbol{h}_d)}{E_{o'''}}}\dfrac{N_o}{N_o+N_{o'''}}\right)\dfrac{N_{oo''}}{N_o}, o' = o \end{cases} \tag{A.8}$$

$$u_o(\boldsymbol{h}_d) = 1 - \sum_{o'\in \boldsymbol{J}_o, o''\in \boldsymbol{o}} m_{o,o',o''}(\boldsymbol{h}_d) \tag{A.9}$$

where $N_{o'o''}$ denotes exogenous passenger demand from zone o' to zone o''. In the first line we calculate the probability of matching an adjacent passenger in zone o (i.e., picking-up is needed); the term $\frac{1}{E_o}$ denotes the probability that the driver is near edge $e_{oo'}$; the term $\frac{k_{e_{oo'}}}{M_o/E_o+M_{o'}/E_{o'}}$ the chance of getting a match for drivers who are near edge $e_{oo'}$; the term $\frac{N_{o'}}{N_o+N_{o'}}$ the probability that the origin of the matched passenger is zone o'; and the term $\frac{N_{o'o''}}{N_{o'}}$ the chance that the matched passenger's destination is o''. The second line denotes the probability of matching a local passenger in zone o (i.e., direct delivery without picking-up), in which we sum the matching probabilities from all common edges between zone o and its adjacent zones (i.e., the summation term for $o''' \in \boldsymbol{J}_o/o$); the explanation for each term is similar to that for the first line. The unmatched probability equals one minus all matched probabilities in zone o (i.e., Eq. (A.9)).

Last, we show the detailed calculation of $r(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d)$ and $r_d(\boldsymbol{s}, \boldsymbol{s}_d, \boldsymbol{s}'_d)$. The decomposition of the platform reward $r(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d)$ is given by

$$r(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d) = r_1(\boldsymbol{h}_d) - r_2(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d) + \mu r_3(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d) \tag{A.10}$$

$$r_1(\boldsymbol{h}_d) = \eta\alpha M \sum_{\tau,o,o'} h_{d,[t,2,\tau,o,o,o']} \tag{A.11}$$

$$\begin{aligned} & r_2(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d) \\ = \beta M \Bigg(& \sum_{o,o',o''|s_o=\beta} h_{d,[t,1,\tau_{o'o'},o',o,o'']} + \sum_{o,o',o''|s_o=\beta} \left(h_{d,[t,2,\tau_{oo''},o,o,o'']} - h'_{d,[t-1,1,1,o',o,o'']}\right) \\ & + \sum_{o,o',o'',\tau>1|s_o=\beta} \left(h_{d,[t,1,\tau-1,o',o,o'']} - h'_{d,[t-1,1,\tau,o',o,o'']}\right) \Bigg) \end{aligned} \tag{A.12}$$

$$r_3(\boldsymbol{s}, \boldsymbol{h}_d, \boldsymbol{h}'_d)$$
$$= \frac{M}{N}\left(\sum_{o,o',o''} h_{d,[t,1,\tau_{o'o},o',o,o'']} + \sum_{o,o',o''} \left(h_{d,[t,2,\tau_{oo''},o,o,o'']} - h'_{d,[t-1,1,1,o',o,o'']} \right) \right. \tag{A.13}$$
$$\left. + \sum_{o,o',o'',\tau>1} \left(h_{d,[t,1,\tau-1,o',o,o'']} - h'_{d,[t-1,1,\tau,o',o,o'']} \right) \right)$$

In Eq. (A.11), the commission withheld during one time period is calculated based on the proportion of drivers who are performing delivery task $h_{d,[t,2,\tau,o,o,o']}$ (i.e., the proportion of drivers in state $[t, 2, \tau, o, o, o']$); the total number of drivers M; commission rate η; and trip fare rate α. In Eq. (A.12), the proportion of drivers who are offered subsidies consists of three terms: $h_{d,[t,1,\tau_{o'o},o',o,o'']}$ denotes newly matched/dispatched drivers who are currently neither in nor self-relocating to the subsidized zones but will pick up passengers there; $h_{d,[t,2,\tau_{oo''},o,o,o'']} - h'_{d,[t-1,1,1,o',o,o'']}$ denotes newly matched/dispatched drivers who are currently in the subsidized zones; and $h_{d,[t,1,\tau-1,o',o,o'']} - h'_{d,[t-1,1,\tau,o',o,o'']}$ ($\tau > 1$) denotes newly matched/dispatched drivers who are coincidentally in the process of self-relocating to the subsidized zones. In Eq. (A.13), the service rate is calculated via the number of drivers M, the proportion of drivers who are newly matched, and the total number of passenger demand N. Note that in Eqs. (A.12)–(A.13), the term $h_{d,[t,2,\tau_{oo''},o,o,o'']}$ also includes previously dispatched drivers who just finished picking-up tasks at subsidized zone o (i.e., the term $h'_{d,[t-1,1,1,o',o,o'']}$); therefore, we need a subtraction, $h_{d,[t,2,\tau_{oo''},o,o,o'']} - h'_{d,[t-1,1,1,o',o,o'']}$, to only count newly matched drivers. Similarly, the term $h_{d,[t,1,\tau-1,o',o,o'']}$ also includes previously dispatched drivers who are on the way to pick up passengers in zone o (i.e., the term $h'_{d,[t-1,1,\tau,o',o,o'']}$, $\tau > 1$), and we need a subtraction to exclude these drivers[15].

The decomposition of a driver's one-step reward $r_d(\boldsymbol{s}, \boldsymbol{s}_d, \boldsymbol{s}'_d)$ is as follows:

$$r_d(\boldsymbol{s}, \boldsymbol{s}_d, \boldsymbol{s}'_d) = r_{d1}(\boldsymbol{s}_d) + r_{d2}(\boldsymbol{s}, \boldsymbol{s}_d, \boldsymbol{s}'_d). \tag{A.14}$$

$$r_{d1}(\boldsymbol{s}_d) = \begin{cases} (1-\eta)\alpha & , s_{d1} = 2 \\ 0 & , otherwise \end{cases} \tag{A.15}$$

$$r_{d2}(\boldsymbol{s}, \boldsymbol{s}_d, \boldsymbol{s}'_d) = \begin{cases} \beta & , s_{d1} = 2, s'_{d1} = 0, s_{d3} = o, s_o = \beta \\ \beta & , s_{d1} = 1, s'_{d1} = 0, s_{d3} = o, s_o = \beta \\ 0 & , otherwise \end{cases} \tag{A.16}$$

In the first line of Eq. (A.15), we assume the fare is uniformly collected when the driver is delivering the passenger, i.e., $s_{d1} = 2$. For instance, if a driver delivers a passenger from time t_1 to t_2 (i.e., $s_{d1} = 2$ for $t \in \{t_1, \dots, t_2\}$), the driver will receive an income of $(1-\eta)\alpha$ for each time period during the delivering task, and the total income from trip fare equals $(t_2 - t_1 + 1)(1-\eta)\alpha$. In Eq. (A.16), a subsidy β for drivers is executed immediately after the

[15] Based on Eq. (A.2), for drivers who are in state $[t-1,1,1,o',o,o'']$ at time $t-1$, their states become $[t, 2, \tau_{oo''}, o, o, o'']$ at time t. Therefore, these drivers are counted in the term $h_{d,[t,2,\tau_{oo''},o,o,o'']}$. Still based on Eq. (A.2), for drivers who are in state $[t-1,1,\tau,o',o,o'']$ ($\tau > 1$) at time $t-1$, their states become $[t, 1, \tau - 1, o', o, o'']$, and these drivers are counted in the term $h_{d,[t,1,\tau-1,o',o,o'']}$.

task switches from "idle" to "picking-up" or "delivering" (i.e., $s'_{d1} = 0$ and $s_{d1} \neq 0$) and the origin of the matched passenger is subsidized (i.e., $s_{d3} = o$ and $s_o = \beta$); the first line indicates that the matched passenger is local (i.e., $s_{d3} = s_{d4}$); and the second line that the matched passenger is in an adjacent zone (i.e., $s_{d3} \neq s_{d4}$).