Multi-Agent Reinforcement Learning for Order-Dispatching via Order-Vehicle Distribution Matching

Ming Zhou, Jiarui Jin, Weinan Zhang, Zhiwei Qin, Yan Jiao, Chenxi Wang, Guobin Wu, Yong Yu and Jieping Ye

上海交通大學 SHANGHAI JIAO TONG UNIVERSITY

APEX 数据与知识管理实验室 Data & Knowledge Management Lab

Abstract

Improving the efficiency of dispatching orders to vehicles is a research hotspot in online ride-hailing systems. In this paper, we propose a decentralized execution order-dispatching method based on multi-agent reinforcement learning without explicitly coordination, to address the large-scale order-dispatching problem. Furthermore, we use KL-divergence optimization at each time step to speed up the learning process and to balance the vehicles (supply) and orders (demand). Besides, with the support of the online platform of Didi Chuxing, we designed a hybrid system to deploy our model.

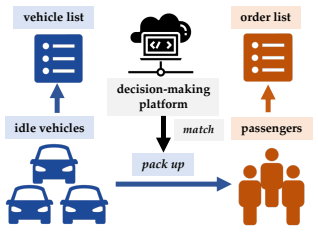


Fig 1. Order dispatching

Formulated as A Markov Game

We regard the order-dispatching task as a sequential decision task, where the goal is to maximize the long-term ADI and ORR per day. According to the characters of the practical environment, each vehicle can only serve the surrounding orders, thus we model the order-dispatching task as a Markov Game.

Agent: We regard a vehicle as an agent.
State: A tuple <G, N, M, D>. Elements in the tuple represent the grid index, the number of idle vehicles, the number of valid orders and the distribution of orders' destinations respectively. The distribution of order's destination is a mean over the destination vectors of orders in grid G.

Action: We regard an order as a action selection, which expressed as a tuple <G_source, G_dest, T, C>, where G_source represents the source grid, G_dest represents the destination grid, T is the order duration and C is the price.

Reward: Because of the goal of learning is to find a solution which maximizes the ADI with high ORR, so we design a reward function which is proportional to the price of each order.

Methodology

Non-Stationary Action Space: There is a fact that for the grid j, the orders produced at time t are always different from the orders produced at other moments. It cannot ensure that the action space is consistent along with the whole episode, so it is problematical to regard the orders as an action while ignoring the distribution of the variant action space. In our proposed method, we use the tuple <S, A> as an input of Q-learning network.

Action Selection Q-learning: From the perspective of agent i, we suppose that s_t denotes the state at time t, a_t denotes the set of orders, then the Bellman Equation in our settings can be expressed as

$$Q(s_t, a_t) = \alpha Q(s_t, a_t) + (1-\alpha)\Big[r_t + \gamma \cdot \mathbb{E}_{a_{t+1}\sim\pi(s_{t+1})}[Q(s_{t+1}, a_{t+1})]\Big]$$

To balance the exploitation and exploration, the Q values related to the same orders set are converted into a biased strategy Boltzmann exploration

$$\pi(a_t^j \mid s_t) = \frac{e^{Q(s_t, a_t^j)/\tau}}{\sum_{a_t^j \in \mathcal{A}_i} e^{Q(s_t, a_t^j)/\tau}}$$

Order-Vehicle Distribution Matching: Notice that we have two goals need to achieve in our proposed method: (1) maximize the long horizontal ADI; (2) optimize the order response rate. If there are always enough vehicles in the dispatching grid, it is easy to decrease the rate of idle vehicles and improve the order response rate, also the long horizontal ADI, while there is a fact that we cannot control the distribution of orders. So we want to make the order and vehicle distribution as similar as possible through finding feasible order-vehicle matches. We do not require explicit cooperation or communication between agents, but an independent learning process with centralized KL divergence optimization.

$$\Pi^* = \arg_\Pi \min D_{KL}(\mathcal{D}^o_{t+1} \parallel \mathcal{D}^v_{t+1}(\Pi))$$

$$\min_\theta \mathcal{L} = \parallel Q_\theta(s,a) - Q^* \parallel_2$$

$$s.t. \quad D_{KL} \le \beta\,,$$

$$\begin{aligned}\nabla_{\theta_j} D_{KL} &= \nabla_{\pi_j} D_{KL} \cdot \nabla_{\theta_j} \pi_j \\ &= -\Big(\sum_{i=1}^{N} p^i_{t+1} \nabla_{\pi_j} \log \frac{q^i_{t+1}}{p^i_{t+1}}\Big) \cdot \nabla_{\theta_i} \pi_j \\ &= c^j_t \sum_{i=1}^{N} p^i_{t+1} \Big[\frac{1}{N_{vehicle}} - \frac{1}{n^i_{t+1}}\Big] \cdot \nabla_{\theta_j} \pi_j\end{aligned}$$

Experiments

We examine the correctness of our model in a toy grid-based order-dispatching environment and the practicality of our model using real-world data from three cities.

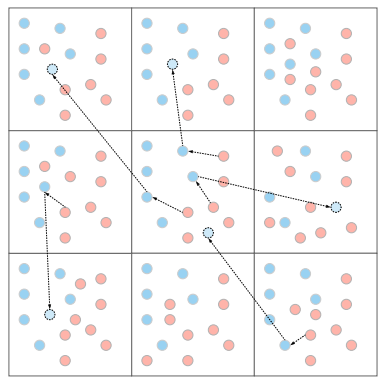


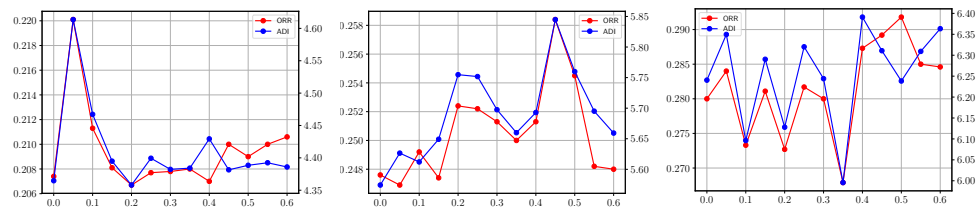
Fig 2. Toy grid-based order-dispatching environment. Blue particle: agent, Red particle: order

Order Distribution Divergence	Low		Medium		High	
Metrics	ADI	ORR	ADI	ORR	ADI	ORR
IL	+12.5%	+6.94%	+11.5%	+6.3%	+6.68%	+2.32%
MDP	+14.5%	+8.94%	+13.3%	+6.69%	+7.28%	+3.42%
KL-Based	**+25.12%**	**+13.40%**	**+20.94%**	**+7.89%**	**+13.47%**	**+4.61%**

Table 1. Performance comparison in terms of ADI and ORR with respect to NOD. We compare against with baselines in three different order distribution changes degree, namely, low, medium and high. KL-Based is our proposed method.

City	City A		City B		City C	
Metrics	ADI	ORR	ADI	ORR	ADI	ORR
IL	+4.69%	+1.68%	+2.96%	+1.11%	+4.72%	+2.05%
MDP	+5.80%	+1.89%	+3.69%	+2.63%	+5.98%	+2.14%
KL-Based	**+6.46%**	**+3.07%**	**+4.94%**	**+3.30%**	**+6.12%**	**+3.01%**

Table 2. Performance comparison in terms of ADI and ORR with respect to NOD. We compare against baselines using different datasets from three cities. KL-Based is our proposed method, which outperforms all baselines on all metrics.

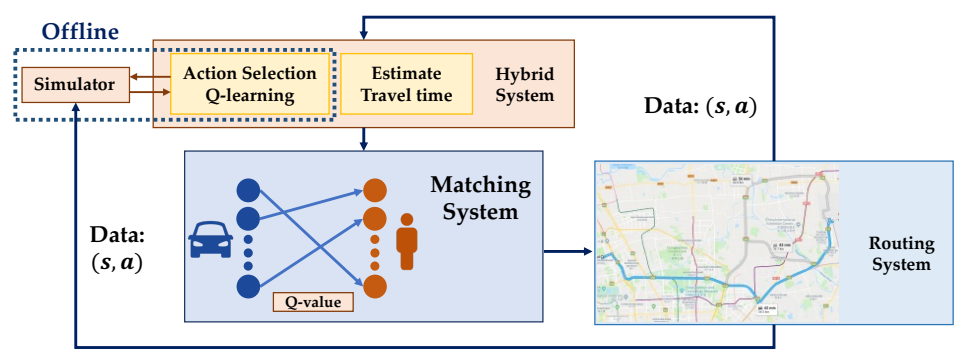


(a) margin=1 (b) margin=2 (c) margin=4

Fig 3. ORR and ADI performance under different λ settings. The horizontal axis represents different λ, the left and right vertical axis represent ORR and ADI respectively.

Deployment

Illustration of deployment. The hybrid system consists of two modules, namely, Action Selection Q-learning (ASQ) and Estimate Travel time modules. The ASQ will interact with simulator periodically, and it will be trained offline in the simulator. Matching System accepts value estimation and outputs <vehicle, order> matches to Routing System.



Once the matching pairs of orders and vehicles has been selected from the matching system, we then deliver these pairs with co- ordinate information to the routing system. The routing system equipped with route planning techniques [32] allows drivers to serve the order. This process will give feedback, i.e. reward to the hybrid system and help the whole system training to achieve better performance.