Real-time and Large-scale Fleet Allocation of Autonomous Taxis: A Case Study in New York Manhattan Island

Yue Yang

Co-authors: Wencang Bao, Mohsen Ramezani, Zhe Xu

Supervisor: Mohsen Ramezani

Overview

1. Introduction
2. Related works
3. Real-time Allocation Framework
4. Solving Approaches for Large-scale Setting
5. Case Study of Manhattan
6. Conclusions and Future Works
7. Acknowledgments

Introduction

Introduction

Autonomous vehicle industry is going through a rapid development right now. GM CRUISE, Tesla, Waymo and Daimler have tested their autonomous taxis on road, and more than ten global automakers have launched or planned programs regarding self-driving cars, which brought huge benefits for:

- Alleviating the burden of city congestion and increases road capacity;
- Reducing traffic accidents by preventing human errors.;
- Hindering the spread of the COVID-19 for passengers;

Congestions

Accidents

Pandemic

Figure 1. Plenty of issues benefited from Autonomous vehicle

Introduction

Developing optimal control algorithms plays an important role for these self-driving platforms to balance the demands and supplies.

1. One representative is order dispatching:

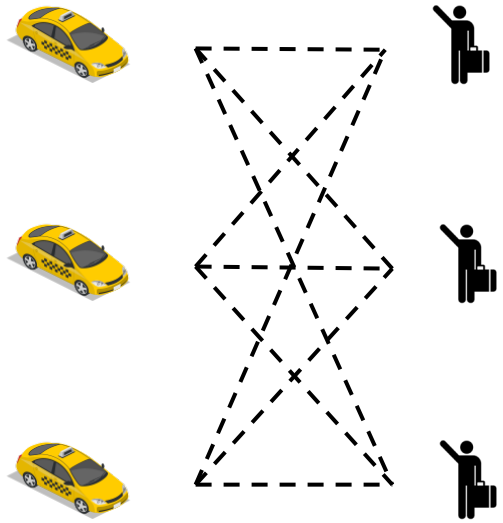
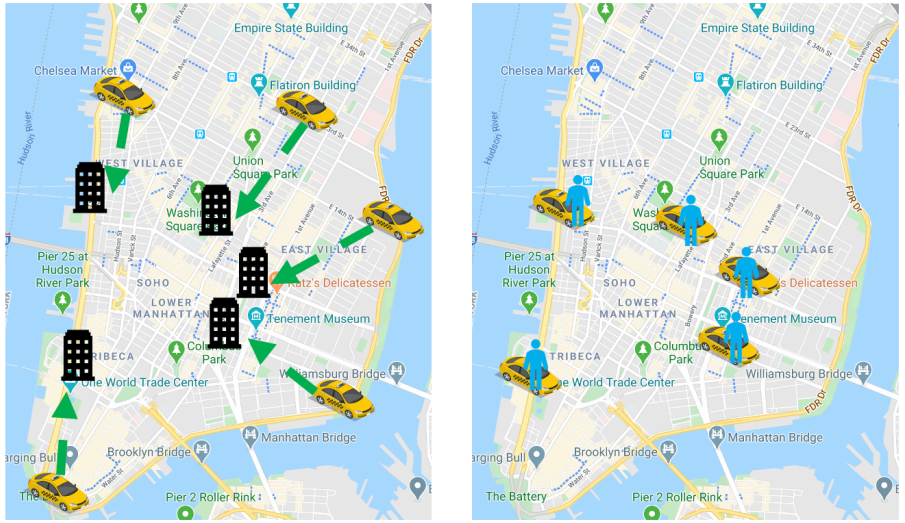


Figure 2. Order dispatching methods

2. Another key application is redistributing available vehicles:



(A) Current Time Step **(B) Next Time Step**

Figure 3. Fleet allocation problems

Introduction

Even though abundant historical demands and digital trajectories can be collected by some mobility on-demand platforms (i.e. Uber, Lyft and DiDi), how to allocate their available fleets is not an easy task:

1. Immediate Allocation will exert significant influence on the future. (**Long-term optimization strategy is needed!**)

2. Dramatical changes of demand with regard to hours in a day. (**Pre-calculating the fleet size!**)

3. Great challenge to the computational efficiency and optimality. (**Large-scale Exact optimization methods**).

Introduction

To address these challenges, the contributions of this work can be summarized as:

1. We formulate the large-scale fleet allocation problem in a Constrained Multi-agent Markov Decision Process (CMMDP) setting, which **considers the long-term payoffs and the immediate resource constraints simultaneously**.

2. To accommodate the real-time requirements, the CMMDP can be further split into dynamic assignment subproblems, and each decision is **endowed with a combinatorial function of instant and learned future rewards**.

3. Facing the complexity of large-scale allocation problem, we employ the **Column Generation algorithm** to achieve a computational efficiency and optimality, which shows a great potential to be deployed in industrial application.

4. Last but not least, the proposed model significantly outperforms the state-of-the-art fleet allocation methods with regard to individual's efficiency and platform's profit.

Related works

Related works

Resource Allocation

The autonomous taxi fleet allocation plan essentially is a resource allocation problem, which is similar to other research topics like e-hailing repositioning and idle taxi routing. These techniques could be grouped into two categories: spot recommendation and route recommendation:

Spot-based allocation

A hotspot is a specific position with a high likelihood of finding a customer. Recommending one or multiple pick-up spots at the immediate next step has been widely discussed.

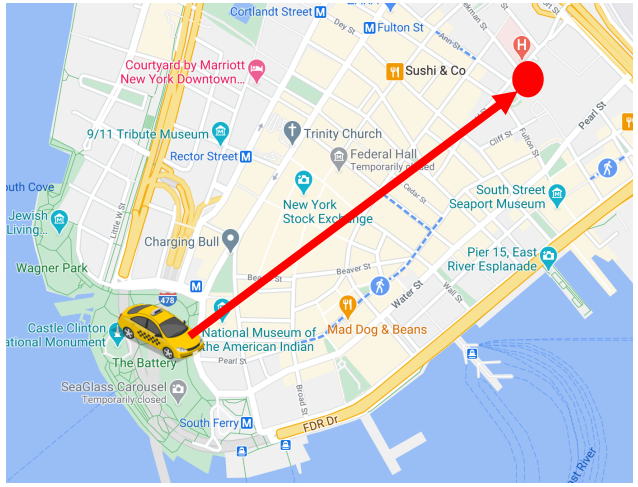


Figure 4. Hot-spot example

Ge et al. 2010, developed a recommender system to offer taxi drivers a sequence of pick-up points and potential parking positions.

Powell et al. 2011, presented a method, so-called Spatio-temporal Profitability (STP) map, to provide vacant taxicabs with the next pickup and profitable locations.

By considering distance, waiting time and expected fare, Hwang et al. 2015, also proposed a taxi recommender system to determine the next cruising location.

Related works

Route-based allocation

Rather than a single spot, cruising routes comprised of a sequence of pick-up locations may be more practical and effective for taxis. In particular, *Markov Decision Process* (MDP) and *Reinforcement Learning* (RL) have been becoming increasingly popular methods to solve these route recommendation problems.

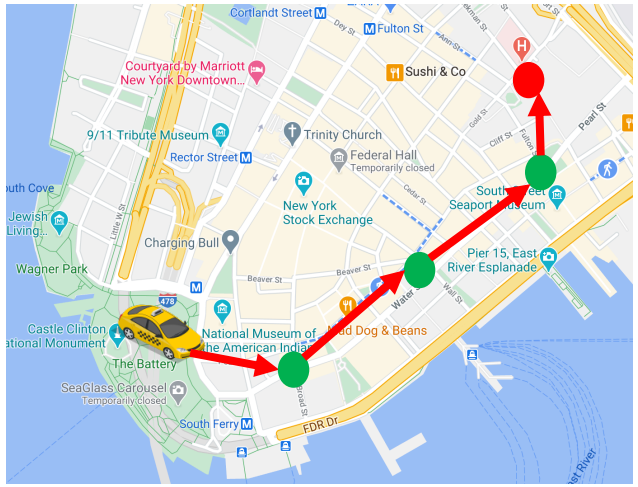


Figure 5. Route allocation example

Garg et al. 2018, modeled taxi route recommendations as a Monte Carlo Tree Search problem whose objective is to minimize the distance to the next anticipated customer. (*KDD2018*)

Yu et al. 2018, introduced temporal Poisson arrivals of passengers and spatial Poisson distributions of competing vacant taxis to calculate state transition probabilities into the MDP models. (*TR PartB 2018*).

Shou et al. 2020, considered the competitions among taxis and incorporated e-hailing drivers' new characteristic features to formulate MDP models. . (*TR PartC 2020*)

Still, all these studies have been poor at considering competitions among multiple taxis and not solved efficiently in a large-scale.

Related works

Constrained Multi-agent Markov Decision Process

Constrained multi-agent Markov Decision Process refers to the multi-agent MDP with resource constraints which always affect agents' decisions in practical use.

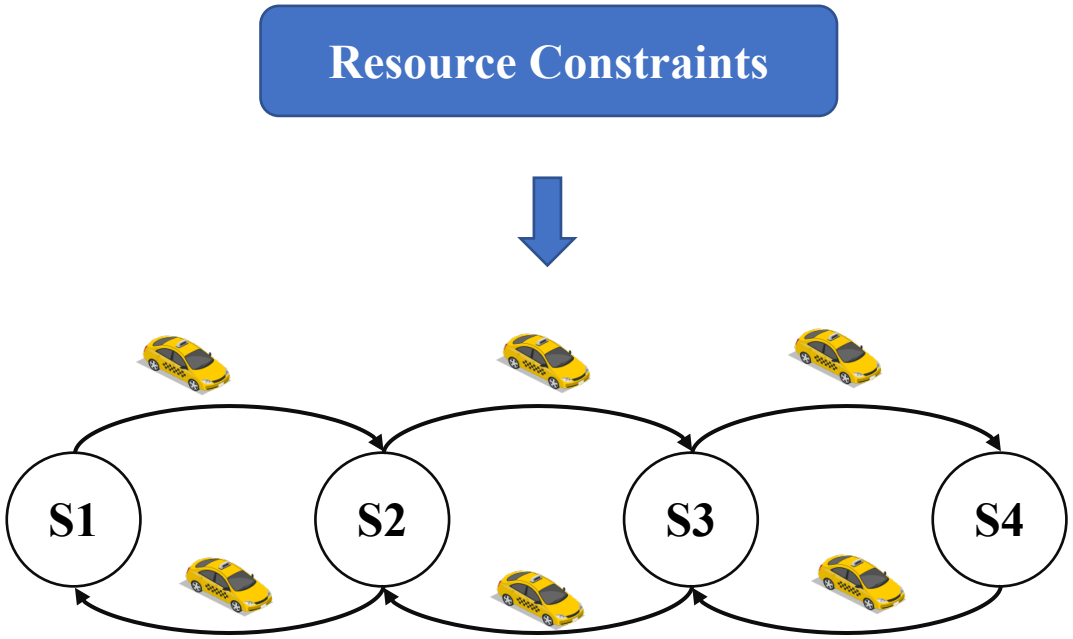


Figure 6. CMMDP example

Beynier et al. 2006, assessed the interdependence of agents under resource and temporal constraints. (*AAAI2006*)

Matignon er al. 2012, utilize a sequential decision making of multirobot collaboration under communication constraints. (*AAAI2012*).

Nijs et al. 2017, measure the probability of resource constraint violation. (*AAAI2017*).

Fowler et al. 2018, also discuss the performance of MDP with action, reward and probabilistic satisfaction constraints.

Related works

Constrained Multi-agent Markov Decision Process

The main challenge in CMMDP is how to lower the computational complexity.

1. Decomposing multi-agent to single agent.

Beynier et al. 2005, allowed the independence of decision making for each agent and consider the lost value stimulated by the local decision on other agents.

2. Harnessing column generation algorithm.

Erwin Walraven et al. 2018, illustrated the column generation algorithm for CMMDP.

De Nijs et al. 2020, combined column generation and CMMDP to investigate the balance of efficiency and safety in policy relaxation.

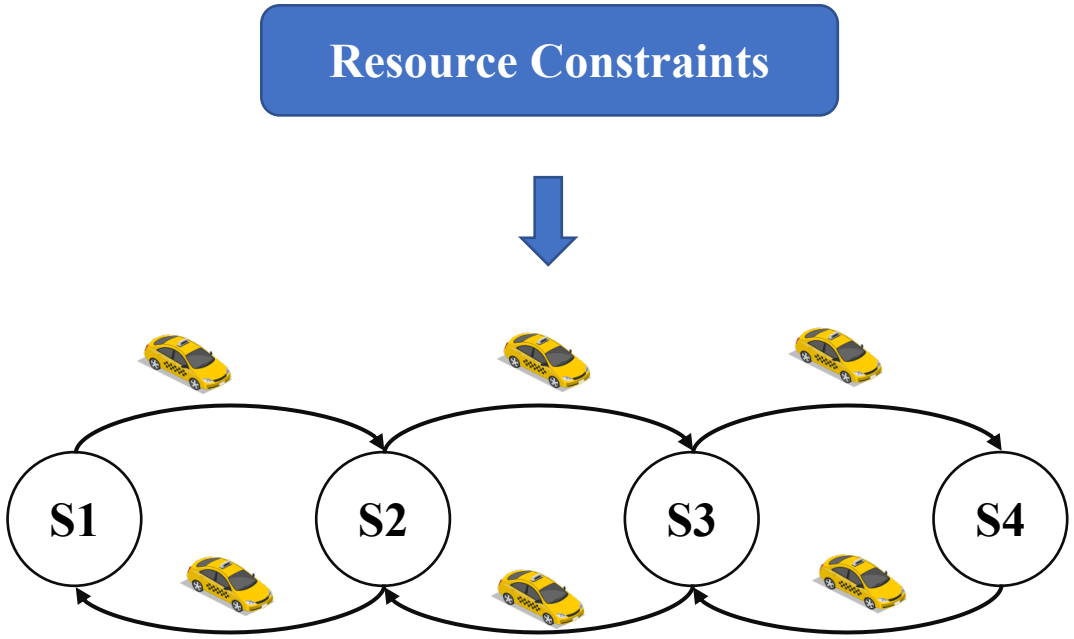


Figure 6. CMMDP example

However, few of CMMDP methods precisely contemplate real-time decision-making which is of high importance for optimization efficiency, profit and customer satisfaction.

Real-time Allocation Framework

Real-time Allocation Framework

Single-agent Setting

As a stepping stone, a multi-agent MDP consisting of N independent agents can be defined through the single agent formulation $\langle S, A, P, R, \mathbb{T} \rangle$ where only one agent interacts with the environment:

Time horizon $\mathbb{T}$, operating time steps set $\{1, \dots, T\}$.

State space S, a finite set of local states, and each state $s, \forall s \in S$ is the current location of the agent.

Action subspace $A(s)$, candidate action set for agents at location s.

$$A(s) = \{s' | \tau(s, s') \leq \delta, \forall\, s' \in S\}$$

Where $\tau(s, s')$ denotes the shortest travel time from s to s', which can be solved directly through Dijkstra's algorithm (Dijkstra et al. 1959) in the network. δ is the travel time threshold limiting the searching space of candidates.

Global action space A, the joint space for all $A(s), \forall s \in S$.

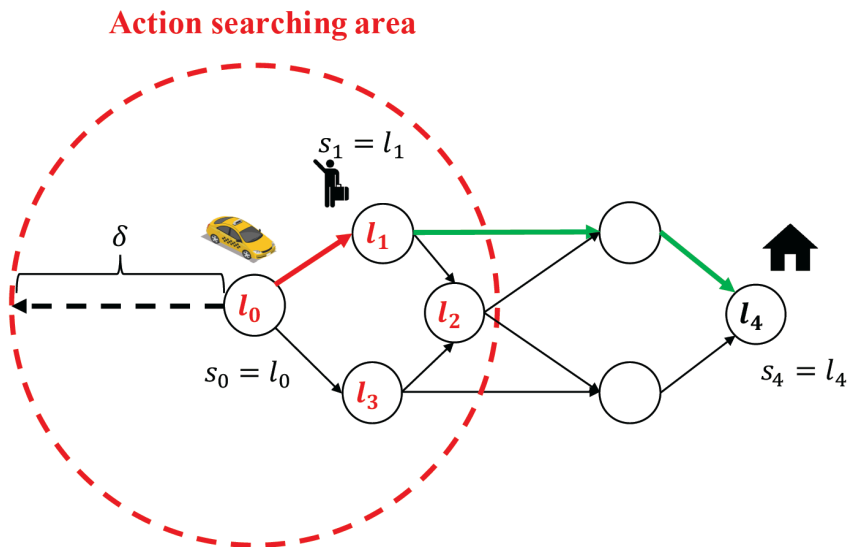


Figure 7. Single-agent example

Real-time Allocation Framework

Single-agent Setting

Matching probability, $p_m(t,s)$ estimates the probability that the vacant taxi is matched to a trip. To simplify, we assume a taxi can only pick an order whose starting node is same with the taxi's current location. According to others' work (Buchholz 2018; Yu et al. 2019),

$$p_m(t,s) = 1 - e^{-\theta \frac{n_{trip}(t,s)}{n_{taxi}(t,s)}}$$

where $n_{trip}(t,s)$, $n_{taxi}(t,s)$ denote the number of trips and taxis in state s of step t, respectively.

Destination probability, $p_d(t,s,s')$, measures the likelihood of the destination of the trip being the state s' when the passenger is picked up at state s of step t.

$$p_d(t,s,s') = \frac{n_{dest}(t,s,s')}{n_{trip}(t,s)}$$

where $n_{dest}(t,s,s')$ represents the number of trips starting from s to s' at step t.

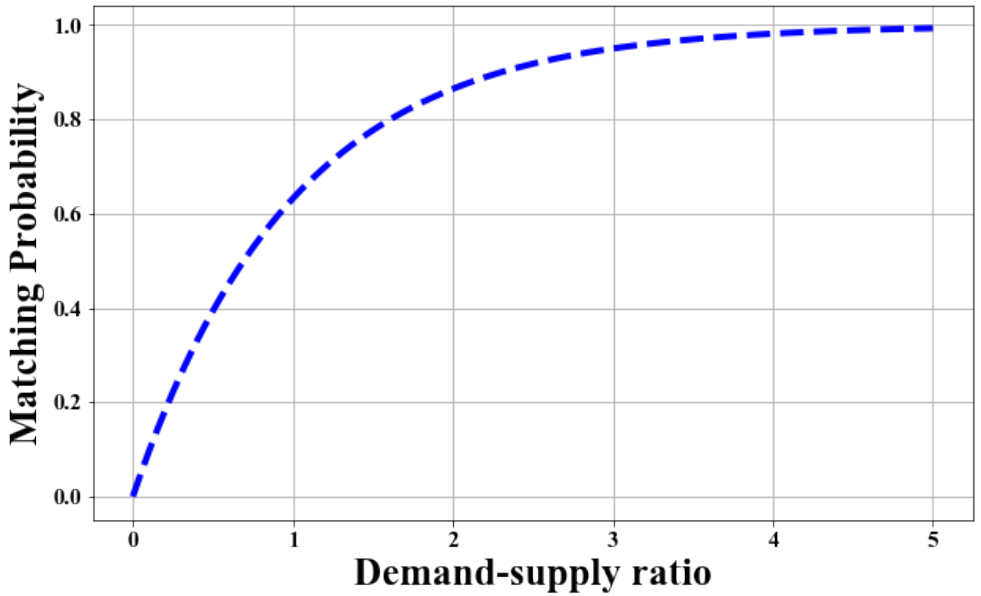


Figure 8. Example of matching function when $\theta = 1$

Real-time Allocation Framework

Single-agent Setting

State transition probability, $P(t, s, a, s')$ is the transition probability model that state s' will be reached when action a is taken in state s of step t.

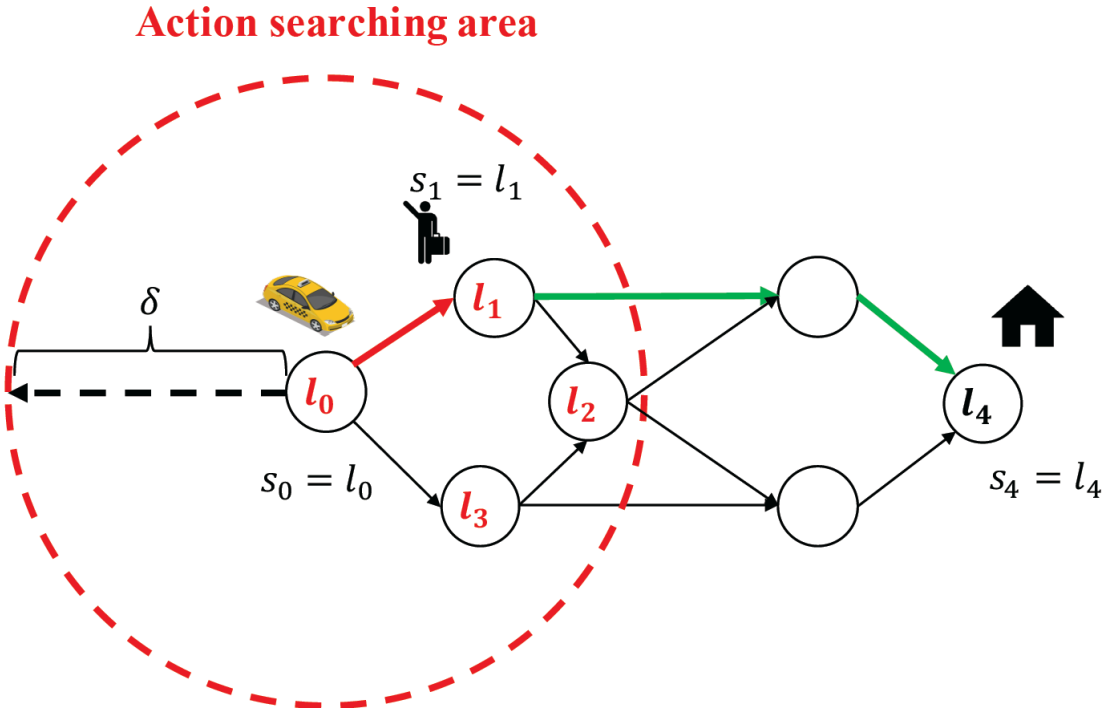


Figure 7. Single-agent example

- If it successfully finds a trip from s_1 to s_4, it will transfer from s_1 to s_4 , then, the transition probability:

$$P(t_0, s_0, a = s_1, s_4) = \boldsymbol{p_m(t_0 + \tau(s_0, s_1), s_1) \cdot p_d(t_0 + \tau(s_0, s_1), s_1, s_4)}$$

Matching **Transition**

- If unfortunately it fails to find a trip, it will stay in s_1 after repositioning.

$$P(t_0, s_0, a = s_1, s_1) = \boldsymbol{1 - p_m(t_0 + \tau(s_0, s_1), s_1)}$$

Failing to find a match

Real-time Allocation Framework

Single-agent Setting

Immediate reward $R(t, s, a)$ is a reward model that maps steps t, states s and actions a to rewards of the agent. Intuitively, the immediate reward can be set as the payment of a trip, by which the goal of the system is to maximize the Gross Merchandise Volume (GMV) of the platform.

Optimal expected reward $\mathrm{V}^*(t, s)$ is the maximal collective expected reward of the agent after T time steps.

$$V^*(t, s) = \max_{a \in A(s)} \{Q(t, s, a)\}$$

where $Q(t, s, a)$, in the Bellman form (Bellman 1957), is the expected reward that

$$Q(t, s, a) = R(t, s, a) + \sum_{s'} \gamma^{\tau(s,s')} [P(t, s, a, s') \cdot V^*(t + \tau(s, s'), s')]$$

Optimal policy π^* of a single-agent is to simply act greedily:

$$\pi^*(t, s) = \underset{a \in A(s)}{arg\max}\{Q(t, s, a)\}$$

Real-time Allocation Framework

Constrained Multi-agent Planning

So far we discussed MDP from the angle of one individual agent without any constraints. However, the single-agent model falls short of the competition among multiple agents or the resource constraint on each action. To model such a problem:

In the first place, We construct a *Multi-agent MDP (MMDP)* problem $\langle \mathbb{N}, \mathbb{S}, \mathbb{A}, \mathbb{P}, \mathbb{R}, \mathbb{T} \rangle$ by extending the single-agent framework to a set of N agents:

Agent $\mathbb{N}, \mathbb{N} = \{1, \dots, N\}$ is a set of agents.

State $\mathbb{S}, \mathbb{S} = S^1 \times S^2 \times \dots \times S^N$ is the Cartesian product of N local factorized states S^i per agent.

Action $\mathbb{A}, \mathbb{A} = A^1 \times A^2 \times \dots \times A^N$ is the Cartesian product of N local factorized actions A^i per agent.

State transition function $\mathbb{P}, \mathbb{P} = \prod P^i$ is the product of the independent transition functions P^i of agent i.

Immediate reward $\mathbb{R}$ is the total immediate reward:

$$\mathbb{R}(t, s, a) = \sum_{i=1}^{N} R^i(t, s, a)$$

Real-time Allocation Framework

Constrained Multi-agent Planning

Resource constraints (Altman 1999) force the agents to coordinate their decisions, that is, the policies used by agents should stay below the resource limits. In our work, we achieve this by computing the **maximum capacity** $\Delta_{t,a}$ accommodating taxis of action(location) a at step t.

$$\Delta_{t,a} = -\frac{-\hat{\theta} \cdot \hat{n}_{trip}(t + \tau(s,a), a)}{\ln(1 - \check{p}_m)}$$

where $\hat{\theta}$ represents the estimated parameter from historical data. . Given an **estimated trip quantity** $\hat{n}_{trip}(t + \tau(s,a), a)$ and a **lower bound matching probability** $\check{p}_m \in [0, 1)$, which guarantees that taking action a at step t will not result in a worse matching probability than the lower-bound, we can derived that:

$$p_m(t + \tau(s,a), a) = 1 - e^{-\hat{\theta}\frac{n_{trip}(t+\tau(s,a),a)}{n_{taxi}(t+\tau(s,a),a)}} \geq \check{p}_m$$

Thus,

$$n_{taxi}(t + \tau(s,a), a) \leq \frac{-\hat{\theta} \cdot \hat{n}_{trip}(t + \tau(s,a), a)}{\ln(1 - \check{p}_m)} = \Delta_{t,a}$$

Real-time Allocation Framework

Constrained Multi-agent Planning

The Constrained Multi-agent Markov decision processes(CMMDP) formulation corresponds to the linear program is presented in Eq. 11, which maximizes the sum of expected rewards of all agents while ensuring that the expected resource consumption respects resource limitations.

- The flow of probability is made consistent with the transition function through the first and second constraints;
- The resource usage restrictions are satisfied in expectation through the third constraint.

$$
\begin{aligned}
&\max \sum_{i}^{\mathbb{N}} \sum_{t}^{\mathbb{T}} \sum_{s}^{S^i} \sum_{a}^{A^i} x(i,t,s,a) \cdot R^i(t,s,a) \\
&s.t. \sum_{a'}^{A^i} x^i(t+\tau(s,s'),s',a') = Z(i,t,s'), \forall i,t,s' \\
&Z(i,t,s') = \sum_{s}^{S^i} \sum_{a}^{A^i} P^i(t,s,a,s') x(i,t,s,a), \forall i,t,s' \\
&\sum_{i}^{\mathbb{N}} \sum_{s}^{S^i} x(i,t,s,a) \leq \Delta_{t,a}, \forall t,a; \\
&0 \leq x(i,t,s,a) \leq 1
\end{aligned}
\tag{11}
$$

Nevertheless,

- $O(N \cdot T \cdot |S^i| \cdot |A^i|)$ variables
- Meeting the resource constraints only in expectation

$x(i,t,s,a)$ represents the probability that agent i at state s executes action a of step t.

Real-time Allocation Framework

On-line Assignment

From the perspective of the platform, real-time situations and on-line implementation should be considered in the optimization framework, which naturally falls into the category of 'Dynamic assignment problem'(Spivey and Powell 2004).

Therefore, the LP of Eq. 11 will be spilt into subproblems as a 'General assignment problem' (Fisher and Jaikumar 1981) in every time step t to downscale the variable size and adapt to the on-line implementation.

In particular, $y(i,t,s,a) \in \{0,1\}$, which indicates whether or not agent i takes a from s in t.

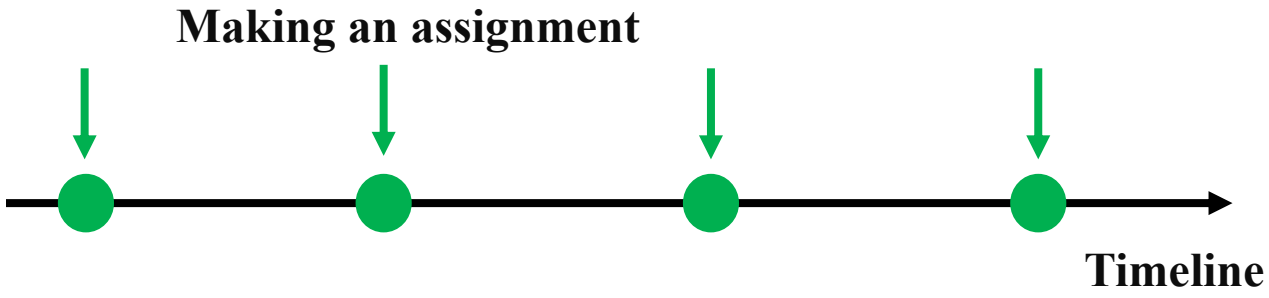


Figure 9. Dynamic assignment problem

$$\begin{aligned} \max & \sum_{i}^{\mathbb{N}} \sum_{a}^{A} y(i,t,s^i,a) \cdot Q_{\pi_i^*}(t,s^i,a) \\ s.t. & \sum_{i}^{\mathbb{N}} y(i,t,s^i,a) \leq \Delta_{t,a}, \forall a \\ & \sum_{a}^{A} y(i,t,s^i,a) \leq 1, \forall i; \\ & y(i,t,s,a) \in \{0,1\} \end{aligned} \tag{12}$$

Figure 10. General assignment problem in our work

Real-time Allocation Framework

On-line Assignment

Discussion of Objective function

In each time step t, We need to determine the best matching between agents and candidate actions to maximize the sum of expected rewards $Q_{\pi_i^*}(t, s^i, a)$.

Note that $Q_{\pi_i^*}(t, s^i, a)$ represents an expected reward in multi-agent system with each agent having its unique policy π_i .

However, **since all autonomous taxis in our setting are entirely subject to the centralized platform**, we can restrict all agents to have a common-interest in maximizing the global gain, resulting in them sharing the same policy π^* of single-agent setting.

$$Q_{\pi_i^*}(t, s^i, a) = Q_{\pi^*}(t, s^i, a)$$

$$\begin{aligned} &\max \sum_{i}^{\mathbb{N}} \sum_{a}^{A} y(i, t, s^i, a) \cdot Q_{\pi_i^*}(t, s^i, a) \\ &s.t. \sum_{i}^{\mathbb{N}} y(i, t, s^i, a) \leq \Delta_{t,a}, \forall a \\ &\sum_{a}^{A} y(i, t, s^i, a) \leq 1, \forall i; \\ &y(i, t, s, a) \in \{0, 1\} \end{aligned} \tag{12}$$

Figure 10. General assignment problem in our work

Real-time Allocation Framework

On-line Assignment

Discussion of Objective function

On the other hand, in many previous researches on MDP-based methods (Yu et al. 2019 (*TR PartB 2018*); Shou et al. 2020(*TR PartC 2020*)), the optimal policy derived from historical episodes is directly used to guide taxis in the current episode, which naturally misses and overlooks the particularities and characteristics of the real-time data.

To capture more information in real-time planning, we design a modified reward value

$$U(i,t,s^i,a) = \hat{R}(t,s^i,a) + Q_{\pi^*}(t,s^i,a)$$

Instant reward **Learned reward**

where $\hat{R}(t,s^i,a)$is the instant reward estimated at step t, equivalent to average payment earned by each taxi from action a.

$$\max \sum_{i}^{\mathbb{N}} \sum_{a}^{A} y(i,t,s^i,a) \cdot Q_{\pi_i^*}(t,s^i,a)$$

$$s.t. \sum_{i}^{\mathbb{N}} y(i,t,s^i,a) \le \Delta_{t,a}, \forall a \qquad (12)$$

$$\sum_{a}^{A} y(i,t,s^i,a) \le 1, \forall i;$$

$$y(i,t,s,a) \in \{0,1\}$$

Figure 10. General assignment problem in our work

Real-time Allocation Framework

On-line Assignment

Figure 11 illustrates the proposed procedure combining off-line learning step with on-line assignment step. Therefore, the objective of our problem can be further modified as:

$$max \sum_{i}^{\mathbb{N}} \sum_{a}^{A} y(i,t,s^i,a) \cdot U(i,t,s^i,a)$$

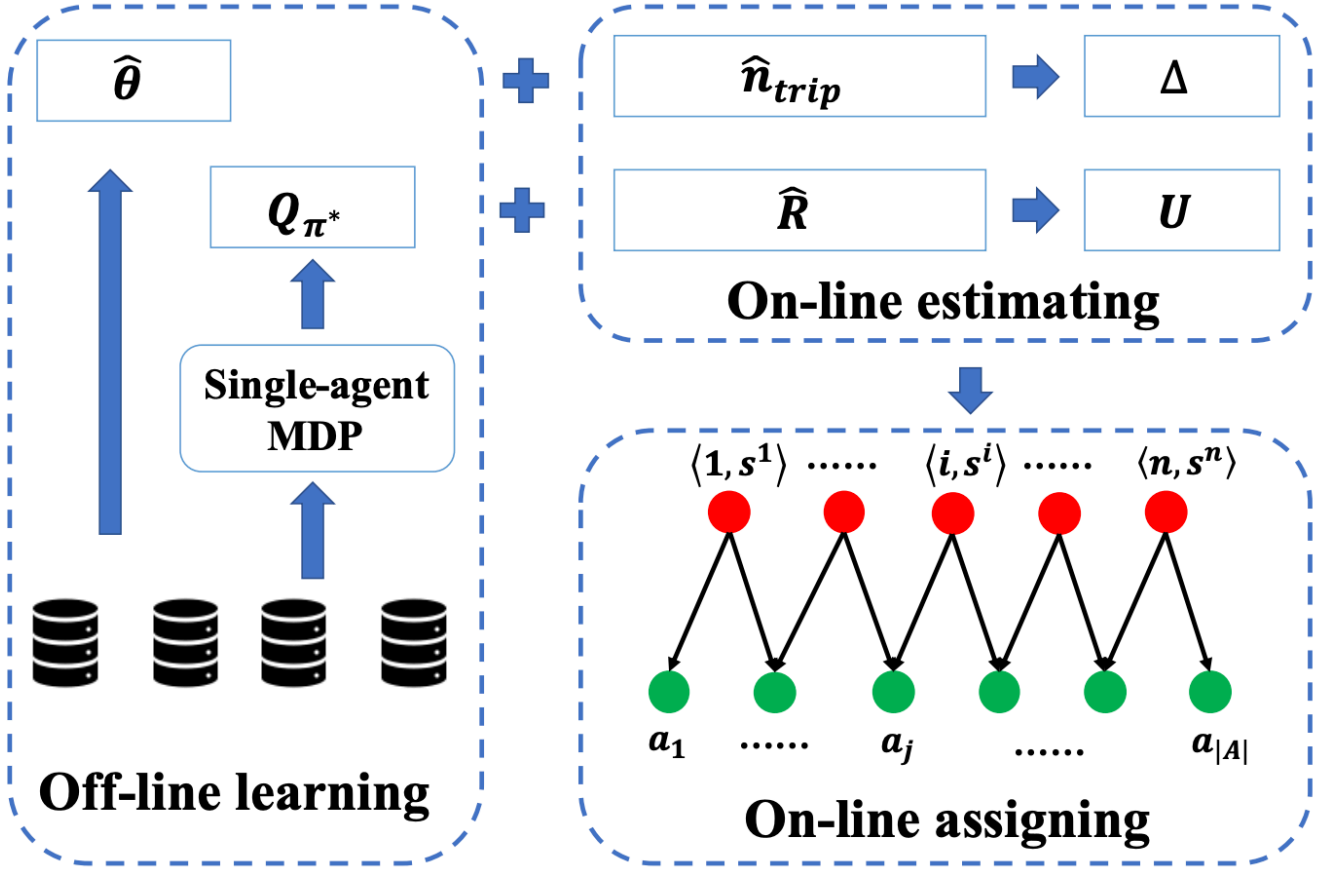


Figure 11. Illustration of deployment of online assignment framework

Solving Approaches for Large-scale Setting

Solving Approaches for Large-scale Setting

To solve larger-scale problems efficiently, we develop a column generation procedure that can be easily adapted to real-world cases.

Column generation approaches

Column Generation is an effective technique for decomposing combinatorial optimization problems, such as stock cut problem (Gilmore and Gomory 1961), graph coloring problem (Mehrotra and Trick 1996), or the vehicle routing problem (Desrosiers, Soumis, and Desrochers 1984).

The basic framework is to reformulate the assignment problem as a *Restricted master problem (RMP)*, then choose entering columns by solving the pricing subproblems. Let

$$\mathbb{Y} = \{Y_{a_1}^1, \dots, Y_{a_1}^{K_{a_1}}, Y_{a_2}^1, \dots, Y_{a_2}^{K_{a_2}}, \dots,$$

$$Y_{a_j}^1, \dots, Y_{a_j}^{K_{a_j}}, \dots, Y_{a_{|A|}}^1, \dots, Y_{a_{|A|}}^{K_{a_{|A|}}}, \}$$

be the set of all feasible assignments(columns) to all actions(locations), where K_{a_j} is the number of feasible solutions of action a_j . For any $Y_{a_j}^k$, $1 \leq k \leq K_{a_j}$, we assume

$$Y_{a_j}^k = \{y^k(1, t, s^1, a_j), \dots y^k(N, t, s^N, a_j)\}$$

to be a group of feasible solutions satisfying the resource constraint of a_j.

Solving Approaches for Large-scale Setting

Restricted master problem (RMP)

1. The first set of constraints enforce that each agent takes at most one action

2. The second set of constraints enforce that at most one assignment is selected for each action.

RMP can not be solved directly due to the exponential number of columns, so we start from a small subset of the columns. **Whether other columns for the RMP join the subset is decided by solving the pricing problem** of each action a_j , which can be reformulated as a knapsack problem.

$$
\begin{aligned}
&\max \sum_{a_j}^{A} \sum_{k}^{K_{a_j}} [\sum_{i}^{\mathbb{N}} y^k(i,t,s^i,a_j) \cdot U(i,t,s^i,a)] \cdot \lambda_{a_j}^k \\
&s.t. \sum_{a_j}^{A} \sum_{k}^{K_{a_j}} y^k(i,t,s^i,a_j) \lambda_{a_j}^k \leq 1, \forall i \in \mathbb{N} \\
&\sum_{k}^{K_{a_j}} \lambda_{a_j}^k \leq 1, \forall a_j \in A; \\
&\lambda_{a_j}^k \in \{0,1\}, \forall a_j \in A, k \in K_{a_j};
\end{aligned}
\tag{17}
$$

Let $\boldsymbol{\lambda_{a_j}^k} \in \{0,1\}$ be a binary variable indicating whether an assignment $\boldsymbol{Y_{a_j}^k}$ is selected for action $\boldsymbol{a_j}$.

Solving Approaches for Large-scale Setting

Pricing problem (Knapsack problem)

Given μ_i , ν_j being the optimal dual price of the first and second constraints in the RMP, respectively, the pricing problem of a specific action a_j , $\forall a_j \in A$ can be defined as right.

$$\begin{aligned} &\max \sum_{i}^{\mathbb{N}} y(i,t,s^i,a_j) \cdot (U(i,t,s^i,a_j) - \mu_i) - \nu_j \\ &s.t. \sum_{i}^{\mathbb{N}} y(i,t,s^i,a_j) \leq \Delta_{t,a_j} \\ &y(i,t,s^i,a_j) \in \{0,1\} \end{aligned} \tag{18}$$

Consequently, if the objective values of any pricing problems are less or equal to zero, the current optimal solution for RMP is also optimal for the unrestricted master problem (Savelsbergh 1997).

Case Study of Manhattan

Case Study of Manhattan

Basic Setting

The data used in this work were collected from taxi data-sets on Manhattan island, New York from November 1st-20st, 2019 on Yellow Cab's website. The details of data can be summarized as:

- The study area is comprised of 67 Locations, 164 Edges.
- 3000 autonomous taxis are randomly generated at 00:00 am per day.
- Each day is treated as an episode segmented into 10-minute windows($T = 144$).

Additionally, in this phase we set some hyper-parameters:

- The action searching threshold $\delta = 10$ min ,
- The matching parameter $\hat{\theta} = 0.94$ by logistic regression on real-world data
- The lower-bound probability $\check{p}_m$ is presented as time-dependent forms in Table 1 empirically.

Hours	00:00am -02:00am	02:00am -07:00am	07:00am -06:00pm	06:00pm -12:00pm
$\check{p}_m$	0.5	0.4	0.7	0.8

Table 1. Setting of $\check{p}_m$

Case Study of Manhattan

Basic Setting

We use the first 15 episodes for learning and conduct evaluation on the following 5 episodes. All the quantitative results of fleet allocation presented in this section are averaged over ten runs.

We adopt three evaluation metrics below:

- **Income Per Hour** (IPH). The average IPH represents the efficiency of taxis allocation as a whole.
- **Occupied rate** (OR). The occupied rate of a taxi is defined as the ratio of the time spent on carrying a trip to the total operating time of the taxi.
- **Gross Merchandise Volume** (GMV). GMV records the total payment of all trips served by taxis in the system.

Case Study of Manhattan

Evaluation

We use the first 15 episodes for learning and conduct evaluation on the following 5 episodes. All the quantitative results of fleet allocation presented in this section are averaged over ten runs.

To evaluate the performance of the optimal policy from our real-time allocation model, there are two baseline heuristics:

Hotspot walk, essentially defines a stochastic policy π^i for each agent i, where the actions are chosen according to the probability $\Pr^i\left(t, s^i, \pi^i\left(t, s^i\right) = a\right)$computed by trips quantity at each step t:

$$\Pr^i\left(t, s^i, \pi^i\left(t, s^i\right) = a\right) = \frac{n_{trip}(t, a)}{\sum_{a'}^{A(s^i)} n_{trip}(t, a')}$$

DiDi repositioning (Xu et al. 2020), employs a centralized optimization framework similar to our assignment decisions, while the reward function $U\left(i, t, s^i, a\right)$ will be set to the partial derivative of p_m with respect to n_{taxi} :

$$U\left(i, t, s^i, a\right) = \frac{\partial\, p_m(t + \tau\left(s^i, a\right), a)}{\partial\, n_{taxi}(t + \tau(s^i, a), a)}$$

Case Study of Manhattan

Evaluation

Table 2 compares average IPH, OR and GMV of Hotspot Walk, DiDi Repositioning and Optimal policy from our approach. In summary, the proposed method showed better performance on all evaluation metrics, indicating amelioration on both the individual's efficiency and the platform's profits.

Heuristics	Avg. IPH ($/hour)	Avg. OR (%)	GMV (M. $)
Hotspot walk (benchmark)	22.74	38.82	1.153
Didi repositioning	23.94 (+5.27%)	40.01 (+3.07%)	1.194 (+3.56%)
Optimal policy	25.56 (+12.40%)	41.36 (+6.54%)	1.206 (+4.59%)

Table 2. Quantitative results.

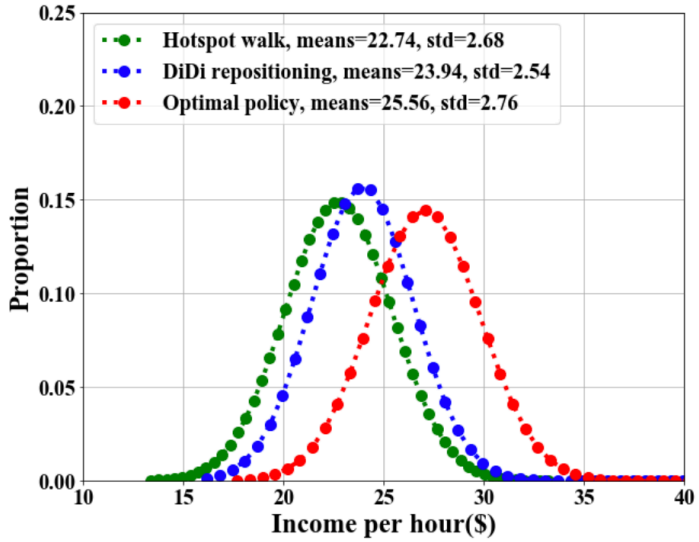


Figure 12. Distribution of the Income Per Hour

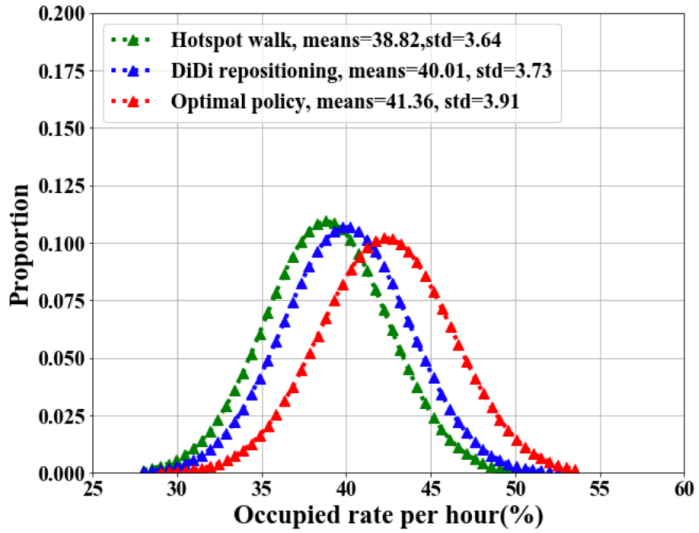


Figure 13. Distribution of the Occupied rate

Case Study of Manhattan

Dynamic Fleet Adjustment

- It's essential for the platform to dynamically adjust the fleet size to avoid waste.
- Our framework provide the platform with a guideline to control the fleet size in real-time.

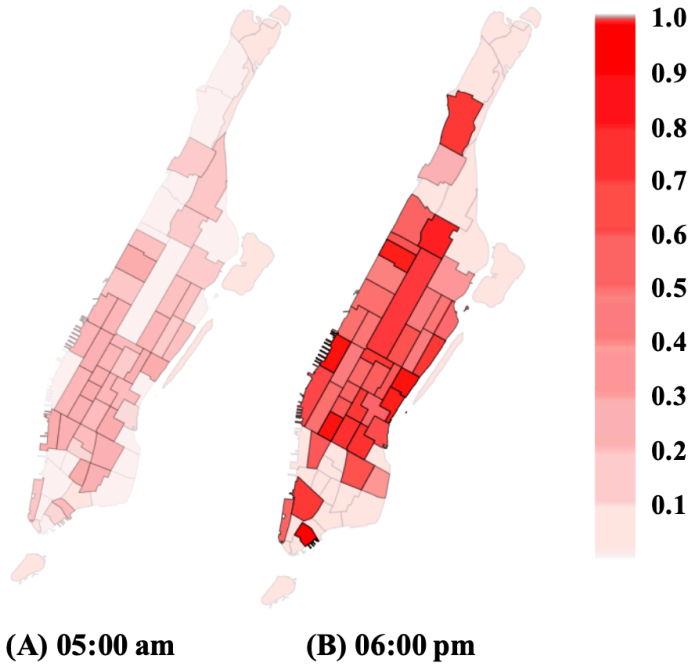


Figure 14. Heat-map examples of the agents' matching probability

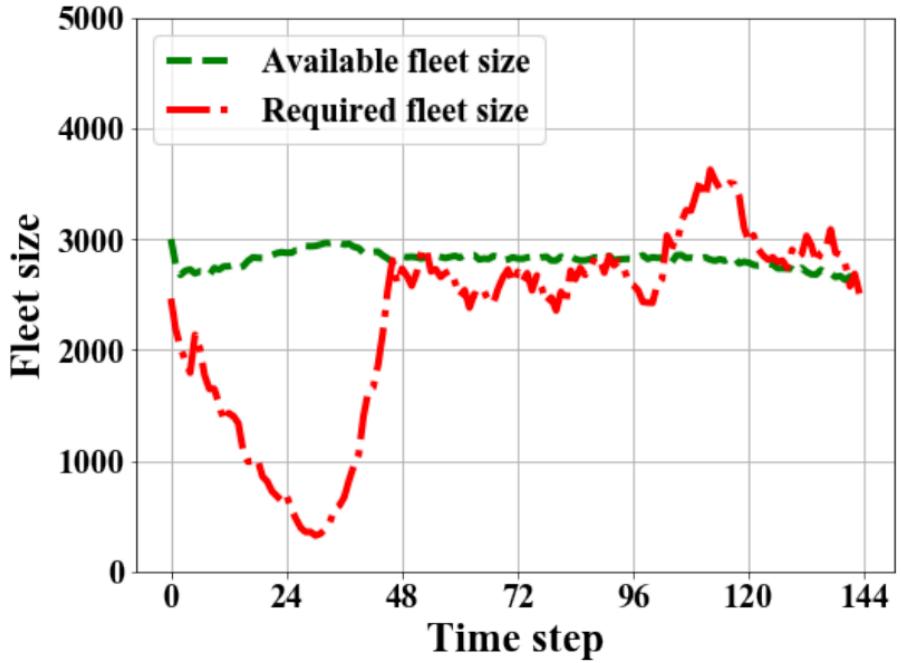


Figure 15. Changes of fleet size in a day

Conclusions and Future Works

Conclusions and Future Works

Conclusions

In this paper, we propose a novel framework of a real-time fleet allocation approach providing an effective redistribution for autonomous taxis on self-driving platforms. To do so,

1. we model the allocation problem as a CMMDP and then divide it into multiple assignment problems by on-line implementation, in which the value of assignment pair is computed as the sum of an instant reward and a long-term expected value learned from historical data.
2. Furthermore, the real-world large-scale assignment is solved more efficiently with the column generation algorithm.
3. Through extensive experiments on the simulator, the proposed algorithm remarkably improves the individual's efficiency and the platform's profit, and also reveals a time-varying fleet adjustment policy to minimize the operation cost.

Future works

In this work, we assume taxis transfer to next-step destinations immediately, whereas the routing paths are ignored. Investigating path decisions possesses superior appliance value(Yu et al. 2019; Tang et al. 2020).

Also, in the fleet adjust experiment, it appears that dynamic fleet size delivers lower cost and higher utilization, while dynamic fleet size adjustment models such as determining when, where and which taxis should join and exit the operation(Ye et al. 2020) are missed in this work.

Acknowledgments

Acknowledgments

The author Yue Yang would like to appreciate his precious and impressive experience of being an intern in Didi Chuxing, and all the data and codes of this work were shared on the GitHub.

Thanks

Reference

Reference

[1] Yong Ge, Hui Xiong, Alexander Tuzhilin, Keli Xiao, Marco Gruteser, and Michael Pazzani. 2010. An energy-efficient mobile recommender system. In Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 899–908.

[2] Jason W Powell, Yan Huang, Favyen Bastani, and Minhe Ji. 2011. Towards reducing taxicab cruising time using spatio-temporal profitability maps. In International Symposium on Spatial and Temporal Databases. Springer, 242–260.

[3] Ren-Hung Hwang, Yu-Ling Hsueh, and Yu-Ting Chen. 2015. An effective taxi recommender system based on a spatio-temporal factor analysis model. Information Sciences 314 (2015), 28–40.

[4] Xinlian Yu, Song Gao, Xianbiao Hu, and Hyoshin Park. 2019. A Markov decision process approach to vacant taxi routing with e-hailing. Transportation Research Part B: Methodological 121 (2019), 114–134.

[5] Shou, Zhenyu, Xuan Di, Jieping Ye, Hongtu Zhu, Hua Zhang, and Robert Hampshire. 2020. Optimal passenger-seeking policies on E-hailing platforms using Markov decision process and imitation learning. Transportation Research Part C: Emerging Technologies 111 (2020). 91-113.

[6] Nandani Garg and Sayan Ranu. 2018. Route recommendations for idle taxi drivers: Find me the shortest route to a customer!. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, 1425–1434.

Reference

[7] Beynier, A.; and Mouaddib, A.-I. 2006. An iterative algorithm for solving constrained decentralized Markov decision processes. In AAAI, volume 6, 1089–1094.

[8] Matignon, L.; Jeanpierre, L.; and Mouaddib, A.-I. 2012. Coordinated Multi-Robot Exploration Under Communication Constraints Using Decentralized Markov Decision Processes. In AAAI 2012, p2017–2023. Toronto, Canada. URL https://hal.archives-ouvertes.fr/hal-00971744.

[9] Nijs, F. D.; Walraven, E.; de Weerdt, M. M.; and Spaan, M. T. J. 2017. Bounding the Probability of Resource Constraint Violations in Multi-Agent MDPs. In Proceedings of the 31st AAAI Conference on Artificial Intelligence, 3562–3568. AAAI.

[10] Fowler, M.; Tokekar, P.; Charles Clancy, T.; and Williams, R. K. 2018. Constrained-Action POMDPs for Multi-Agent Intelligent Knowledge Distribution. In 2018 IEEE International Conference on Robotics and Automation (ICRA), 3701–3708.

[11] Beynier, A.; and Mouaddib, A.-I. 2005. A polynomial algorithm for decentralized Markov decision processes with temporal constraints. In Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, 963–969.

[12] Erwin Walraven, M. T. J. S. 2018. Column Generation Algorithms for Constrained POMDPs. Journal of Artificial Intelligence Research 62: 489–533.

Reference

[13] De Nijs, F.; and Stuckey, P. J. 2020. Risk-Aware Conditional Replanning for Globally Constrained Multi-Agent Sequential Decision Making. In Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, 303–311.

[14] Dijkstra, E. W.; et al. 1959. A note on two problems in connexion with graphs. Numerische mathematik 1(1): 269–271.

[15] Bellman, R. 1957. A Markovian decision process. Journal of mathematics and mechanics 679–684.

[16] Altman, E. 1999. Constrained Markov decision processes, volume 7. CRC Press.

[17] Spivey, M. Z.; and Powell, W. B. 2004. The dynamic assignment problem. Transportation Science 38(4): 399–419.

[18] Fisher, M. L.; and Jaikumar, R. 1981. A generalized assignment heuristic for vehicle routing. Networks 11(2): 109– 124.

[19] Bang-Jensen, J.; and Gutin, G. Z. 2008. Digraphs: theory, algorithms and applications. Springer Science & Business Media.

[20] Gilmore, P. C.; and Gomory, R. E. 1961. A linear programming approach to the cutting-stock problem. Operations research 9(6): 849–859.

[21] Mehrotra, A.; and Trick, M. A. 1996. A column generation approach for graph coloring. informs Journal on Computing 8(4): 344–354.

Reference

[22] Desrosiers, J.; Soumis, F.; and Desrochers, M. 1984. Routing with time windows by column generation. Networks 14(4): 545–565.

[23] Savelsbergh, M. 1997. A branch-and-price algorithm for the generalized assignment problem. Operations research 45(6): 831–841.

[24] Xu, Z.; Men, C.; Li, P.; Jin, B.; Li, G.; Yang, Y.; Liu, C.; Wang, B.; and Qie, X. 2020. When Recommender Systems Meet Fleet Management: Practical Study in Online Driver Repositioning System. In Proceedings of The Web Conference 2020, 2220–2229.

[25] Tang, J.; Wang, Y.; Hao, W.; Liu, F.; Huang, H.; and Wang, Y. 2020. A Mixed Path Size Logit-Based Taxi Customer-Search Model Considering Spatio-Temporal Factors in Route Choice. IEEE Transactions on Intelligent Transportation Systems 21(4): 1347–1358.

[26] Ye, X.; Li, M.; Yang, Z.; Yan, X.; and Chen, J. 2020. A Dynamic Adjustment Model of Cruising Taxicab Fleet Size Combined the Operating and Flied Survey Data. Sustainability 12(7): 2776.