2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)
October 8-12, 2022, Macau, China

SMART-eFlo: An Integrated SUMO-Gym Framework for Multi-Agent Reinforcement Learning in Electric Fleet Management Problem

Shuo Liu*, Yunhao Wang*, Xu Chen, Yongjie Fu, Xuan Di *Member, IEEE*

***Abstract*— Electric vehicles (EVs) have been used in the ride-hailing system in recent years, which brings the electric fleet management problem (EFMP) critical. This paper aims to leverage multi-agent reinforcement learning (MARL) in EFMP. In particular, we focus on how EVs learn to manage battery charging, pick up and drop off passengers. We propose an integrated SUMO-Gym framework based on the SUMO simulator to capture EVs' asynchronous decision-making regarding charging and ride-hailing in complex traffic environments. We adopt a hierarchical reinforcement learning (HRL) scheme, where each EV decides to get charged or pick up a passenger on the upper level and chooses a charging station or passenger on the lower level. We develop a learning algorithm for the HRL scheme to solve EFMP and present numerical results about the efficiency of our algorithm and policies EVs have learned in EFMP. Our codes are available at https://github.com/LovelyBuggies/SUMO-Gym, which provides an open-source environment for researchers to design traffic scenarios and test RL algorithms for EFMP.**

I. INTRODUCTION

EVs are anticipated to reduce emissions, save energy and improve mobility. However, the increasing use of EVs in the ride-hailing system raises many challenges: The detour time of idle EVs when searching for passengers in the system can lead to more frequent charging needs. The lack of charging stations and slow charging speed can heavily affect the ride-hailing service. This paper leverages MARL to solve the EFMP.

A. Related Work

Recent years have seen a growing trend of applying MARL in fleet management problems, including vehicle dispatching and repositioning [1], order dispatching [2], [3], online matching [4] and reward design [5]. There are a few studies focusing on EFMP. [6], [7], [8], [9] apply deep reinforcement learning to vehicle routing for EVs in the ride-hailing system. [10], [11], [12] study charging station recommendation. However, these studies do not fully capture factors that can affect EVs' decision-making, including the battery status, the spatial distribution of passengers, the waiting time in charging stations and road congestion. Therefore, we propose an integrated SUMO-Gym framework based on the SUMO simulator [13], which incorporates EVs, passengers, charging stations, traffic signals and background vehicles into traffic environments.

Existing literature has used the SUMO simulator to create complex traffic environments for MARL, including multi-agent routing game [14] and traffic signal control [15]. However, these studies cannot provide a reliable framework that allows researchers to implement various traffic scenarios for MARL. Therefore, we resort to Gym [16], a flexible paradigm to test RL algorithms, and combine it with SUMO in this work. In particular, we adopt the multi-agent Gym, i.e., Petting-Zoo [17], to develop the integrated framework SUMO-Gym for MARL in EFMP. Compared to some existing work like Flow [18] and SMARTS [19], SUMO-Gym allows users to flexibly create a wide range of multi-agent problem scenarios and its environments are highly configurable. By leveraging SUMO-Gym, users can easily implement different scenarios and embed RL algorithms.

B. Contributions

Contributions of this paper include:

1) We propose an integrated SUMO-Gym framework, which incorporates traffic environments from the SUMO simulator into Petting-Zoo for MARL research.
2) Based on SUMO-Gym framework, we adopt an HRL scheme to solve EFMP using deep Q-learning algorithm.
3) We implement the HRL scheme for EFMP on two road networks, i.e., Jumbo and COSMOS [20].

The rest of the paper is organized as follows. Sec. II describes the EFMP we study in this paper. Sec. III introduces the workflow of our proposed SUMO-Gym framework. Sec. IV introduces the HRL scheme to solve EFMP. Sec. V presents numerical results. Sec. VI concludes this study.

II. EFMP

In this section, we use a small road network to demonstrate the EFMP. In Fig. 1, the road network is represented by a grid world. There are three EVs, two charging stations, and four passengers who need to be picked up. Flags represent the destinations of passengers. EVs, denoted by green, blue, and red cars, make decisions regarding charging and picking up passengers. Arrows represent the trajectories of EVs. For example, the dashed red line marked by ① shows that an idle EV (red car) first picks up a passenger and drops the passenger off at the destination (the solid red line). As

(Corresponding author: Xuan Di.)
Shuo Liu and Yunhao Wang are with the Department of Computer Science, Columbia University, New York City, NY 10027 USA (e-mail: sl4921@columbia.edu, yw3736@columbia.edu). *(*: Shuo Liu and Yunhao Wang equally contributed to this work.)*
Xu Chen and Yongjie Fu are with the Department of Civil Engineering and Engineering Mechanics, Columbia University, New York City, NY 10027 USA (e-mail: xc2412@columbia.edu, yf2578@columbia.edu).
Xuan Di is with the Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY, 10027 USA, and also with the Data Science Institute, Columbia University, New York, NY, 10027 USA (e-mail: sharon.di@columbia.edu).

2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC) | 978-1-6654-6880-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITSC55140.2022.9922047

demonstrated by red line ② and ③, this EV then goes to a charging station and then picks up another passenger after its battery is charged. In this work, the goal of EFMP is to find the optimal strategy in EVs' sequential decision making about charging and picking up passengers.

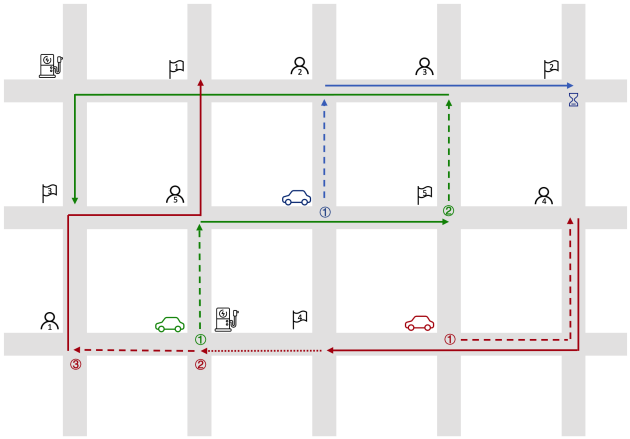


Fig. 1: EFMP.

III. SUMO-Gym Framework

In this section, we introduce the details of the integrated SUMO-Gym framework (See Fig. 2).

A. SUMO

We first demonstrate the settings of traffic environments configured in the SUMO simulator as follows:

- Travel demand: The spatio-temporal distribution of passengers, including origins and destinations.
- Road network: Users can design road networks for simulation. In this work, we use Jumbo and COSMOS networks. The COSMOS network is created based on the road information extracted by OpenStreetMap API.
- EV settings: The set-up of each EV includes its current location, battery status, battery capacity and maximum driving speed.
- Charging station settings: The set-up of each charging station includes its location, charging speed and charging delay.

SUMO-Gym obtains the settings of traffic environments from SUMO initial xml configurations and through TraCI at run time.

B. Multi-Agent Gym

We then introduce the set-up of multi-agent Gym:

- Reset: The initial locations of fully-charged EVs are randomized at the beginning of each episode. All charging stations are not occupied.
- Step: In each step, agents who need to make decisions will select actions, draw their private observations and get rewards. Their experiences are stored in replay buffers. Agents who do not need to make decisions only update their own states through the move function.
- Close: The terminal states are when the EVs pick up all passengers or all of them run out of power.

C. MARL Algorithm

The MARL algorithm used to study EFMP in the SUMO-Gym framework will be introduced in Sec. IV.

IV. HRL Scheme

In this section, we introduce an HRL scheme to solve EFMP based on the proposed SUMO-Gym framework (See Fig. 3). The HRL scheme incorporates hierarchical decision-making of agents into the learning process [21]. [22] provides a more comprehensive review of HRL. We describe the details of our HRL scheme we apply in the following subsections.

A. Upper-Level

On the upper level, EVs need to make decisions about whether to pick up passengers or go to charging stations. We adopt a partially observable Markov decision process (POMDP), denoted by $\langle \bar{\mathcal{S}}, \bar{\mathcal{A}}, \bar{\mathcal{P}}, \bar{\mathcal{R}}, \bar{\mathcal{O}}, \bar{\gamma} \rangle$, to capture EVs' decision making because each EV can only obtain its neighborhood information with the environment state $\bar{s} \in \bar{\mathcal{S}}$ not fully observable to the agent. We specify each element in the POMDP as follows:

- m. There are m EVs in the environment, denoted by $\{1, 2, \cdots, m\}$.
- $\bar{s} \in \bar{\mathcal{S}}$. On the upper level, state $\bar{s}$ refers to the battery status of all EVs and the spatial distribution of all road users. Non-strategic background vehicles generated from SUMO are introduced in the environment.
- $\bar{a} \in \bar{\mathcal{A}}$. The action space is discrete. Joint action is $\bar{a} = (\bar{a}_1, \bar{a}_2, \cdots, \bar{a}_m)$ where $\bar{a}_i \in \bar{\mathcal{A}}_i = \{\text{charge}, \text{pick-up}\}$, $i = 1, 2, ..., m$.
- $\bar{p} \in \bar{\mathcal{P}}$. After taking action $\bar{a}$ in state $\bar{s}$, an agent arrives at a new state $\bar{s}'$ with conditional transitions probability $\bar{p}^h(\bar{s}'|\bar{s}, \bar{a})$. $\bar{p}$ is typically unknown to the agent, thus the agent needs to repeatedly interact with the environment to gain state transition experiences, i.e., $(\bar{s}, \bar{a}, \bar{s}')$.
- $\bar{r} \in \bar{\mathcal{R}}$. When state $\bar{s}$ transits to $\bar{s}'$ by taking action $\bar{a}$, the agent receives a reward $\bar{r}(\bar{s}, \bar{a}, \bar{s}')$. An EV will get a positive reward when it is charged successfully or it drops the passenger off at the destination. When the EV runs out of its power before reaching a charging station, it gets a negative reward.
- $\bar{o} \in \bar{\mathcal{O}}$. Each EV is able to draw a private observation $\bar{o}$. We assume each agent can only observe its neighborhood environment, which is part of the traffic environment $\bar{s}$. Joint observation is $\bar{o} = (\bar{o}_1, \bar{o}_2, \cdots, \bar{o}_m)$ and $\bar{o}_i, i = 1, .., m$ includes the battery status of agent i and locations of vehicles in the agent's neighborhood.
- $\bar{\gamma}$ is the discount factor for discounting future rewards. When $\bar{\gamma} = 1$, the agent does not differentiate future rewards from the immediate rewards. As $\bar{\gamma}$ gets smaller, the agent cares less about rewards received in the distant future, therefore the decision-making gets more myopic.

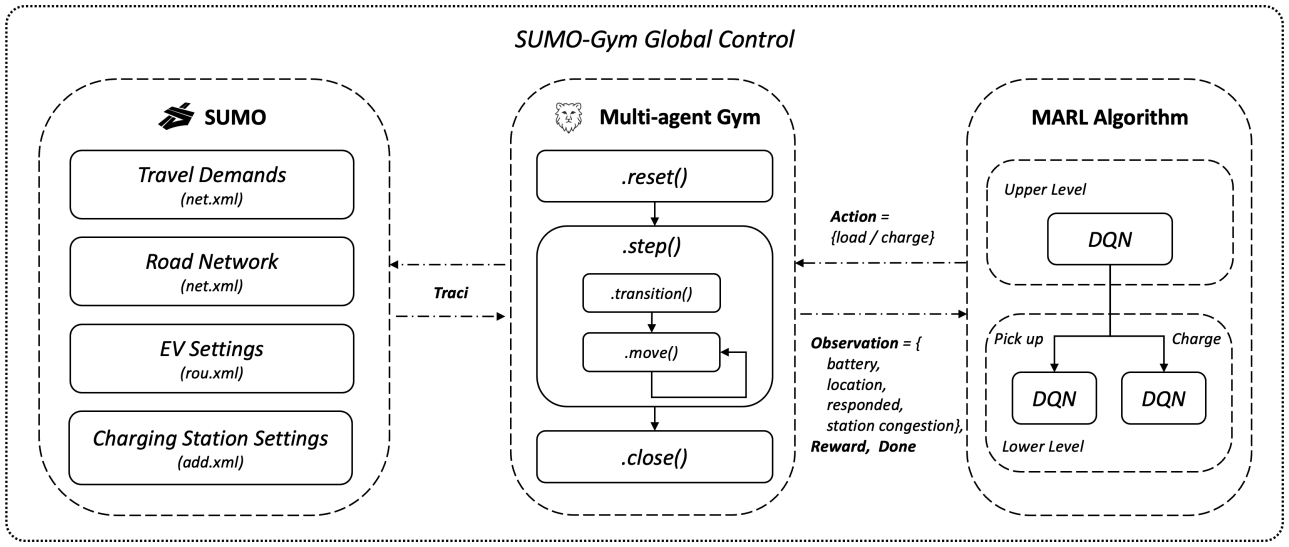


Fig. 2: SUMO-Gym framework with user-friendly interfaces in Petting-Zoo style.

The Q-value function $\bar{Q}_i$ for agent i is an estimate of expected future reward that can be obtained from observation $\bar{o}$ and joint action $\bar{a}$, namely:

$$\bar{Q}_i = \bar{Q}_i(\bar{o}_i, \bar{a}) \tag{1}$$

In this work, we use deep Q-networks (DQNs) [23] as a function approximator of the Q-value function. DQN $\bar{Q}_\theta$ updates its parameter θ by minimizing the loss function $\bar{\mathcal{L}}$.

$$\bar{\mathcal{L}}(\theta) = \mathbb{E}_{\bar{o},\bar{a},\bar{s}'}\left[(\bar{r}(\bar{o},\bar{a},\bar{o}') + \bar{\gamma}\max_{\bar{a}'}\bar{Q}_{\theta^-}(\bar{o}',\bar{a}') - \bar{Q}_\theta(\bar{o},\bar{a}))^2\right].$$

where $\bar{Q}_{\theta^-}$ is the target network parameterized by θ^-.

$$\nabla_\theta \underline{\mathcal{L}}^c(\theta) = \nabla_\theta \mathbb{E}_{\underline{o}^c,\underline{a}^c,\underline{o}^{c\prime}}\big[(-\underline{Q}^c_\theta(\underline{o}^c,\underline{a}^c) + \underline{r}^c(\underline{o}^c,\underline{a}^c,\underline{o}^{c\prime}) + \underline{\gamma}^c \max_{\underline{a}^{c\prime}} \underline{Q}^c_{\theta^-}(\underline{o}^{c\prime},\underline{a}^{c\prime})\ )^2\big].$$

B. Lower-Level

On the lower level, an EV needs to choose a charging station on the road network if the decision on the upper level is charging. The EV chooses a specific passenger if the decision on the upper level is to pick up. We specify elements on the lower level as follows:

Charge:

- $\underline{s}^c \in \underline{S}^c$. State $\underline{s}^c$ includes the status of charging stations (i.e., whether a charging station is occupied or not) and locations of EVs.
- $\underline{a}^c \in \underline{A}^c$. The action of an EV is to choose a charging station. Once EV chooses a charging station, it will follow the shortest path from the current location to the station.
- $\underline{r}^c \in \underline{R}^c$. $\underline{r}^c$ depends on the battery status and the waiting time of an EV to be charged successfully.
- $\underline{o}^c \in \underline{O}^c$. Each EV draws a private observation, including whether each charging station is occupied or not and its current location.

Algorithm 1 EFMP-HRL

Input: exploration parameter $\epsilon = \epsilon_0$, learning rate $\eta = \eta_0$, network update period k, target network update period τ. Initialize DQNs on the upper and lower level for each agent.
for $episode \leftarrow 1$ *to* M **do**
  while s is not terminal **do**
    On the upper level:
    Each agent selects action $\bar{a}_i$ (ϵ greedy method).
    On the lower level:
    Each agent selects action $\underline{a}_i$ ($\underline{a}^c_i$ or $\underline{a}^p_i$).
    Execute actions and update observations.
    Store $(\bar{o}_i, \bar{a}_i, \bar{r}_i, \bar{o}'_i)$ on the upper level.
    Store $(\underline{o}_i, \underline{a}_i, \underline{r}_i, \underline{o}'_i)$ on the lower level.
  end while
  Sample a batch of experience from replay buffers and Update parameter $\bar{\theta}_i$ of $\bar{Q}_i$, $\underline{\theta}_i$ of $\underline{Q}_i$ by minimizing the loss function every k episode.
  Decrease the exploration parameter ϵ.
  Decrease the learning rate η.
  Update target networks every τ episodes.
end for

The Q-value function is formulated as:

$$\underline{Q}^c_i = \underline{Q}^c_i(\underline{o}^c_i, \underline{a}^c) \tag{2}$$

Pick-up:

- $\underline{s}^p \in \underline{S}^p$. State $\underline{s}^p$ includes whether passengers have been picked up or not and locations of EVs.
- $\underline{a}^p \in \underline{A}^p$. The action of an EV is to choose a passenger to pick up. The EV will use the shortest path from its current location to the passenger.
- $\underline{r}^p \in \underline{R}^p$. $\underline{r}^p$ depends on the trip fare.
- $\underline{o}^p \in \underline{O}^p$. Each EV draws a private observation, including whether passengers have been picked up and its

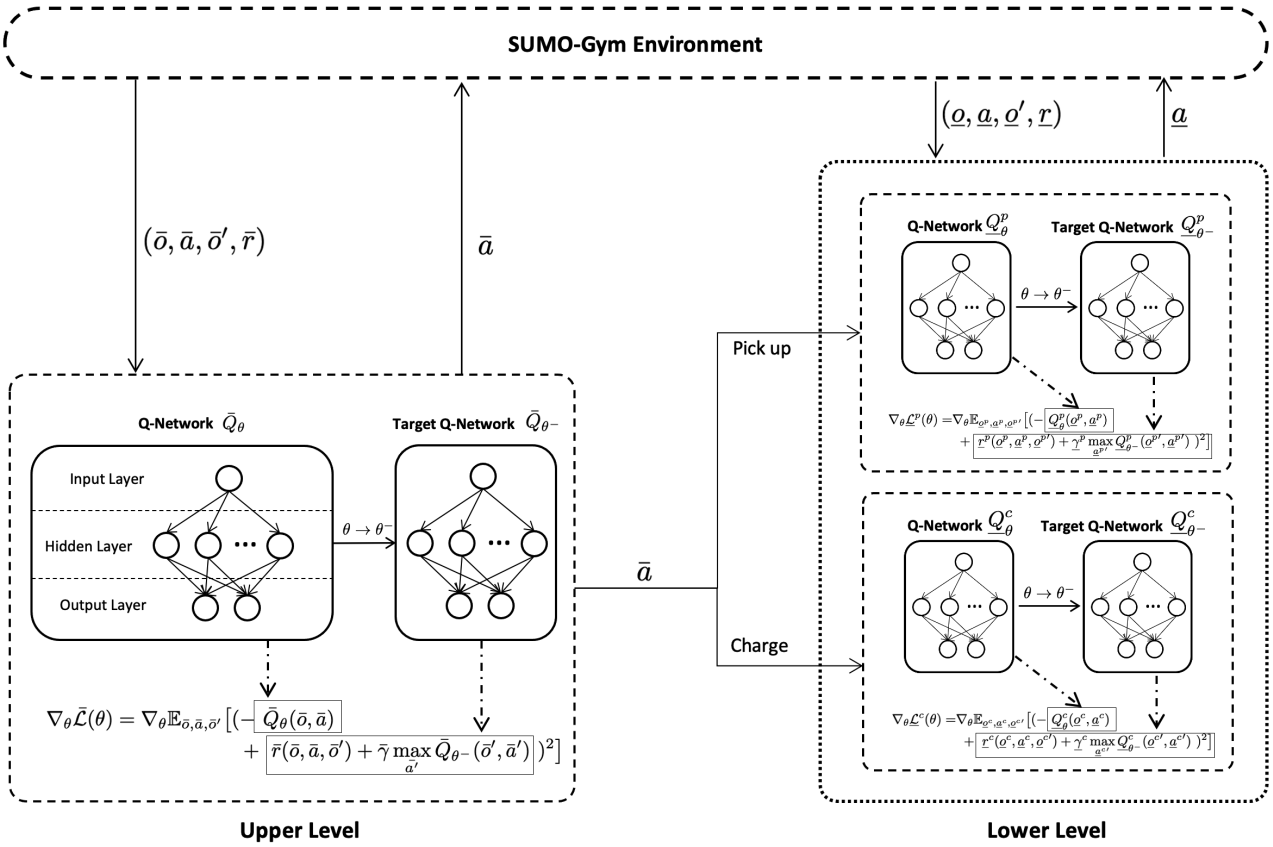


Fig. 3: HRL scheme to solve EFMP.

current location.

The Q-value function is formulated as:

$$\underline{Q}^p_i = \underline{Q}^p_i(\underline{o}^p_i, \underline{a}^p) \tag{3}$$

C. Algorithm

The algorithm to solve EFMP is summarized in Alg. 1. We first initialize DQNs and target DQNs on the upper and lower level for each agent. According to the widely used ϵ-greedy method, i.e., to choose action randomly with probability ϵ and from optimal policy with probability $1-\epsilon$, each agent first selection action $\bar{a}_i, i = 1, ..., m$ on the upper level and then selection action $\underline{a}_i$ on the lower level until reaching some terminal state. Joint action is executed in the environment to trigger the state transition $s \to s'$. Each agent keeps drawing private observations and getting their rewards. Their experience $(\bar{o}_i, \bar{a}_i, \bar{r}_i, \bar{o}'_i)$ and $(\underline{o}_i, \underline{a}_i, \underline{r}_i, \underline{o}'_i)$ are stored in replay buffers on the lower and upper level networks, respectively. Batching training is then used to update Q function by minimizing the loss function. Target networks are updated every τ periods.

V. Experiments

In this section, we apply the HRL scheme to two road networks: Jumbo and COSMOS. We first introduce the set-up of our traffic environments for EFMP and then present numerical results.

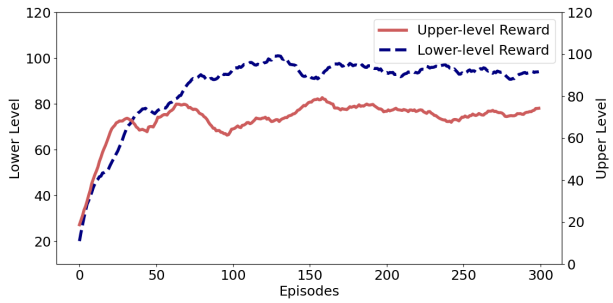
A. Set-up

The set-up of Jumbo and COSMOS networks is demonstrated in Table. I. On the upper level, we construct a DQN for each agent, using multi-layer perceptions with 5 fully connected layers. We use an adaptive learning rate starting from 0.003 and the learning rate decays 5% in every 25 episode. The Q-network gradient descents every 10 episode and the target Q-network is synchronized every 20 episode. On the lower level, we construct two DQNs for each agent with respect to the upper-level decisions: charging and pick-up. The learning rate starts from 0.02 and decays 5% in every 25 episode. The training batch size of DQNs on Jumbo and COSMOS are 8 and 32, respectively. Other hyperparameters on the lower level are the same as those on the upper level. We use the Adam optimizer for all DQNs in our study.

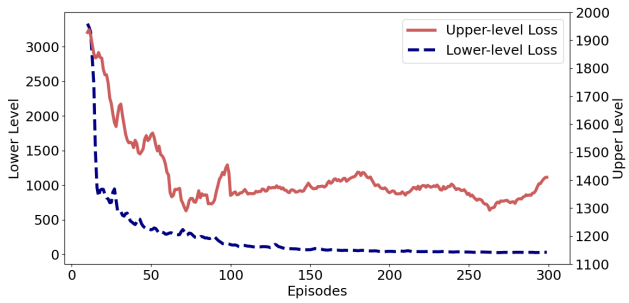
B. Results and Discussion

We first look into the algorithm performance. The convergence of the reward and loss function for an agent in the Jumbo network is illustrated in Fig. 4a and 4b, respectively. Fig. 4c and 4d demonstrate the algorithm performance in the COSMOS network. It takes around 200 and 80 episodes for EVs to stabilize their policy learning in the Jumbo and COSMOS networks, respectively.

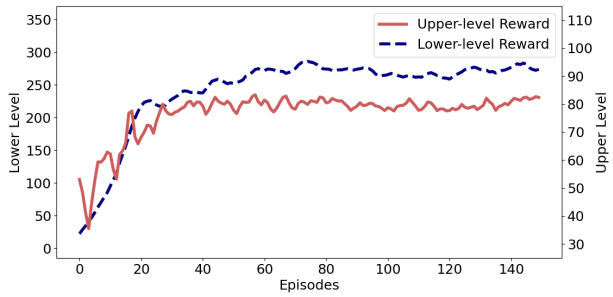
Fig. 5a, 5b and 5c demonstrate sample trajectories of an EV in the COSMOS network. The blue arrows in Fig. 5a



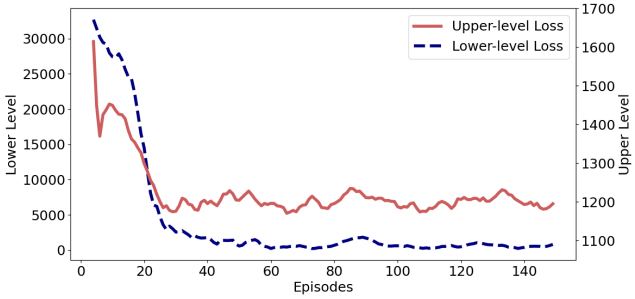
(a) Reward curves in Jumbo.



(b) Loss curves in Jumbo.



(c) Reward curves in COSMOS.



(d) Loss curves in COSMOS.

Fig. 4: Algorithm performance.

	Node	Link	Passenger	EV	Charging Station	Traffic Light	Background Vehicle
Jumbo	31	74	17	10	2	14	1000
COSMOS	75	202	49	12	2	56	5000

TABLE I: Set-up of Jumbo and COSMOS networks.

show that when an EV is in a safe battery status, it first picks up a passenger and drops off the passenger at the destination. The EV then picks up another passenger. The yellow arrow in Fig. 5b shows that when an EV is in an alert battery status, it first picks up a passenger. After dropping off the passenger, the EV goes to a charging station before picking up another passenger. The green arrows in Fig. 5c show that an EV chooses the charging station that is not fully loaded before picking up the second passenger.

We adopt the trained policy of EVs for 100 runs and look into several performance measures in the EFMP, which are defined as follows:

- TBC: Total battery consumption (TBC) refers to how much power EVs need to consume to finish all tasks.
- CWT: Charging waiting time (CWT) is the average waiting time for each EV to get charged.
- PWT: Passenger waiting time (PWT) refers to the average waiting time for a passenger to be picked up.
- PFR: Pick-up failure rate (PFR) refers to the proportion of cases in 100 runs that all EVs have run out of power before picking up all passengers.

We make a comparison of performance measures in the proposed HRL scheme and in the case when all EVs randomly select charging stations or pick up passengers (Table. II). It is shown that EVs in the HRL scheme complete tasks with lower energy consumption and less waiting time.

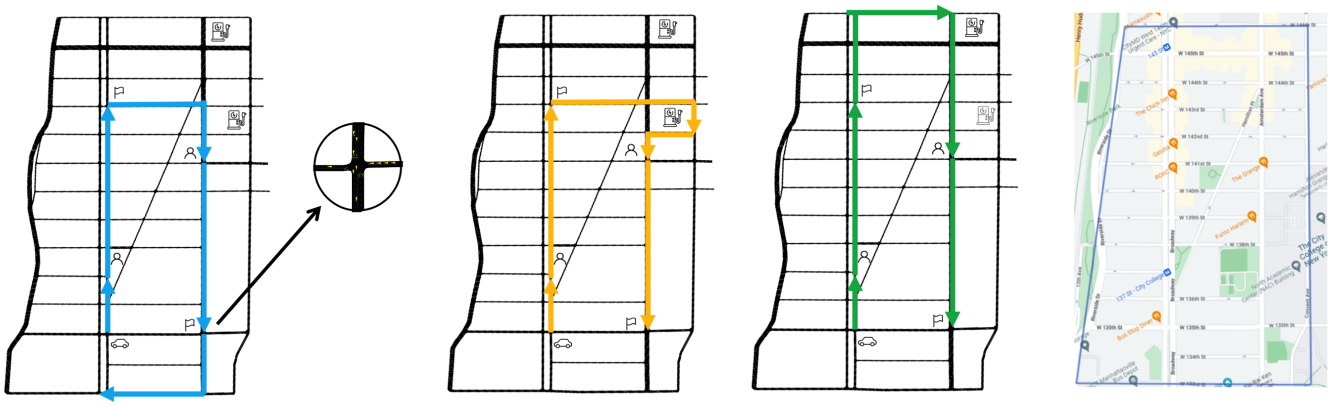
VI. CONCLUSIONS

In this paper, we propose an integrated SUMO-Gym framework for MARL studies. This framework incorporates the SUMO simulator into the multi-agent Gym, which allows researchers to easily implement different scenarios and embed RL algorithms. Based on the proposed SUMO-Gym framework, we adopt an HRL scheme to solve the EFMP. We implement numerical experiments to study EVs' policy learning on Jumbo and COSMOS networks. Results show that compared to the strategy that EVs randomly go to charging stations or pick up passengers, the trained policy in the HRL scheme allows EVs to complete tasks with lower energy consumption and less waiting time.

REFERENCES

[1] John Holler, Risto Vuorio, Zhiwei Qin, Xiaocheng Tang, Yan Jiao, Tiancheng Jin, Satinder Singh, Chenxi Wang, and Jieping Ye. Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1090–1095. IEEE, 2019.

[2] Zhiwei Qin, Xiaocheng Tang, Yan Jiao, Fan Zhang, Zhe Xu, Hongtu Zhu, and Jieping Ye. Ride-hailing order dispatching at didi via reinforcement learning. *INFORMS Journal on Applied Analytics*, 50(5):272–286, 2020.



(a) When an agent is in the safe battery status and charging stations are all vacant. (b) When an agent is in an alert battery status. (c) When a charging station (grey) is fully loaded. (d) A real-world map of the COSMOS network.

Fig. 5: Routing choice of an EV.

	TBC		CWT		PWT		PFR	
	HRL	Random	HRL	Random	HRL	Random	HRL	Random
Jumbo	273.35 kWh	334.28 kWh	2.53s	3.64s	213.42s	221.57s	32%	61%
COSMOS	279.61 kWh	427.67 kWh	2.98s	3.25s	106.98s	148.05s	18%	35%

TABLE II: Performance measures.

[3] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 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*, pages 1774–1783, 2018.

[4] Jintao Ke, Feng Xiao, Hai Yang, and Jieping Ye. Optimizing online matching for ride-sourcing services with multi-agent deep reinforcement learning. *arXiv preprint arXiv:1902.06228*, 2019.

[5] Zhenyu Shou and Xuan Di. Reward design for driver repositioning using multi-agent reinforcement learning. *Transportation Research Part C: Emerging Technologies*, 119:102738, 2020.

[6] Bo Lin, Bissan Ghaddar, and Jatin Nathwani. Deep reinforcement learning for the electric vehicle routing problem with time windows. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[7] Jie Shi, Yuanqi Gao, Wei Wang, Nanpeng Yu, and Petros A Ioannou. Operating electric vehicle fleet for ride-hailing services with reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4822–4834, 2019.

[8] Nicholas D Kullman, Martin Cousineau, Justin C Goodson, and Jorge E Mendoza. Dynamic ride-hailing with electric vehicles. *Transportation Science*, 2021.

[9] Xindi Tang, Meng Li, Xi Lin, and Fang He. Online operations of automated electric taxi fleets: An advisor-student reinforcement learning framework. *Transportation Research Part C: Emerging Technologies*, 121:102844, 2020.

[10] Weijia Zhang, Hao Liu, Fan Wang, Tong Xu, Haoran Xin, Dejing Dou, and Hui Xiong. Intelligent electric vehicle charging recommendation based on multi-agent reinforcement learning. In *Proceedings of the Web Conference 2021*, pages 1856–1867, 2021.

[11] Anyun Yang, Hongbin Sun, and Xiao Zhang. Deep reinforcement learning strategy for electric vehicle charging considering wind power fluctuation. *Journal of Engineering Science & Technology Review*, 14(3), 2021.

[12] Enshu Wang, Rong Ding, Zhaoxing Yang, Haiming Jin, Chenglin Miao, Lu Su, Fan Zhang, Chunming Qiao, and Xinbing Wang. Joint charging and relocation recommendation for e-taxi drivers via multi-agent mean field hierarchical reinforcement learning. *IEEE Transactions on Mobile Computing*, 2020.

[13] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[14] Zhenyu Shou, Xu Chen, Yongjie Fu, and Xuan Di. Multi-agent reinforcement learning for markov routing games: A new modeling paradigm for dynamic traffic assignment. *Transportation Research Part C: Emerging Technologies*, 137:103560, 2022.

[15] Wangzhi Li, Zhaobin Mo, Yongjie Fu, Kangrui Ruan, and Xuan Di. Cvlight: Decentralized learning for adaptive traffic signal control with connected vehicles. *arXiv preprint arXiv:2104.10340*, 2021.

[16] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[17] Justin K Terry, Benjamin Black, Nathaniel Grammel, Mario Jayakumar, Ananth Hari, Ryan Sullivan, Luis Santos, Clemens Dieffendahl, Caroline Horsch, Rodrigo Perez-Vicente, et al. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[18] Nishant Kheterpal, Kanaad Parvate, Cathy Wu, Aboudy Kreidieh, Eugene Vinitsky, and Alexandre Bayen. Flow: Deep reinforcement learning for control in sumo. *EPiC Series in Engineering*, 2:134–151, 2018.

[19] Zhou Ming and Luo Jun. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. 2021.

[20] PAWR NSF. Cosmos: Cloud enhanced open software defined mobile wireless testbed for city-scale deployment.

[21] Tejas D. Kulkarni, Karthik R. Narasimhan, Ardavan Saeedi, and Joshua B. Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3682–3690, Red Hook, NY, USA, 2016. Curran Associates Inc.

[22] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Comput. Surv.*, 54(5), 2021.

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.