EVS37 Symposium
COEX, Seoul, Korea, April 23-26, 2024

Balancing EV Demand at Charging Stations using Multi-Agent Reinforcement Learning

Rory Coulson, Elnaz Shafipour, Sebastian Stein, Jan Buermann, Suleiman M. Sharkh, Andrew Cruden

University of Southampton, University Road, Southampton, Hampshire, United Kingdom, rc3g20@soton.ac.uk

Abstract

This paper proposes a method for optimising the routing of electric vehicles (EVs) to charging stations via a multi-agent reinforcement learning (MARL) demand balancing system in order to reduce queuing time. This is achieved through simulations via the SUMO simulator to train and test agents to reduce demand by applying reinforcement learning algorithms. Q-learning, PPO and DQN experiments have been conducted to determine a suitable algorithm. The approaches were run on multiple test road networks and a real-world Berlin network with ten charging stations to validate the findings. Varied learning strategies are also explored to determine the appropriate behaviour patterns between the agents, including competitive and cooperative learning as well as a mix of the two. The results of the most promising DQN cooperative implementation applied to the Berlin network achieved an 88.09% reduction in the mean wait times when compared with a greedy approach. The findings of this paper demonstrate the potential for practical benefits of applying MARL systems to real-world environments.

Keywords: Multi-agent reinforcement learning, demand balancing, competitive and cooperative learning

1 Introduction

This paper aims to help address the growing number of electric vehicle (EV) drivers reporting inadequate charging experiences. A study [20] at the University of Southampton concluded from online and in-person surveys with 1,278 EV drivers that more than a third, 34.8% of EV drivers, are dissatisfied with the charging experience. This problem worsens with EV demand growth, with projections of UK EV stock rising "up to 36 million by 2040" [10]. This user dissatisfaction is mainly due to the lack of sufficient EV charging stations, with 68.6% of EV drivers saying they frequently experience out-of-service chargers and 78.4% experience fully occupied charging stations leading to queuing [20]. The queuing problem is exacerbated due to the time it takes to charge an EV, with even ultra-fast 400kW chargers taking 10-15 minutes [18] while slower 50kW chargers can take up to 30 minutes [22]. Additionally, there is demand from EV drivers to receive more reliable and detailed information on the availability of EV charging stations. This is backed up by a previous study [20], finding 69.4% of EV drivers noted specific information is lacking from current EV information apps, including "incomplete or unreliable information about charge points," with examples of limited information "regarding availability, service status and pricing." With a diversity of charging stations from different suppliers and a large variety of EV types, providing users with reliable real-time information is proving to be a challenging task.

Improvements to the EV driving experience have been addressed with multiple strategies, including implementing reinforcement learning (RL) and game-theoretic approaches [21, 1, 15, 30]. A paper [6] working on the routing of EVs to mitigate the queuing problem highlights how a stochastic time-dependent approach to the EV routing problem can accomplish over 80% waiting time improvements in some cases.

Wang et al. [23] introduce a "fairness-aware" recommendation system which formulates the problem as a "fairness constrained Pareto optimization problem," for large-scale taxi fleets. Furthermore, Yuan et al. [28] propose a proactive partial charging strategy for EV taxi fleets to more efficiently charge vehicles and finds the strategy to reduce the average waiting time of unserved passengers by up to 83.2%. Multi-agent reinforcement learning (MARL) approaches have also been shown to be effective particularly in complex interaction environments including transportation tasks, with implementations in the delivery-service space [8], operating traffic signals [24], and EV multi-critic recommendation systems [30]. MARL has been found to improve the model learning rates by utilizing the observations and behaviours of others in the environment through "experience sharing", which can optimise performance [3, 16]. MARL has also been used to recommend EV charging stations using real-world datasets [30] and models the agents as the charging stations due to issues of "ad-hoc" and "non-repetitive" actions making learning for vehicles challenging. This work additionally integrates a multi-critic system to handle multiple objectives including: charging wait time, price and failure rate. The approach uses centralized training and decentralized execution to allow for cooperative learning between agents and "guarantees efficiency and flexibility" [30]. This uses a strategy to delay access to information to allow agents to use "future knowledge" in the training phase while "taking actions immediately," [30]. The current paper takes a similar approach for integrating a centralized platform for multi-agent communication, but develops an additional custom cooperation learning strategy.

The main objective of this paper is to provide a proof of concept system to aid the management of the queues at EV charging stations. The paper focuses on queueing, as it has the largest impact on dissatisfaction and contributes to the lack of information concerns [20]. To improve the situation, a multi-agent reinforcement learning (MARL) approach with a custom localised cooperative learning implementation is proposed to reduce queuing times and learn policies that could run in real-world applications, utilising the benefits of agent collaboration and the advances in deep reinforcement learning algorithms.

2 Background

The paper defines a single-agent and multi-agent reinforcement learning system while additionally providing background information on the various training strategies within a multi-agent system.

2.1 Single-Agent Reinforcement Learning

Prior to delving into MARL, it may prove beneficial to first examine reinforcement learning (RL) for a single agent. This can be formalised as a Markov Decision Process (MDP) [3]. An MDP is a tuple (S, A, Pa, Ra) where:

- S is the state space,
- A is the action space,
- P_a represents $P : S \times A \to \Delta(S)$,
the transition probability of action a in state s leading to a new state s',
- R_a is the immediate reward after a transition with action a to the new state [3].

The agents interact with the environment and the goal of the agent is to maximise the reward received following an optimal policy learnt. This optimal policy is the goal of the MDP and maps the state space to the distribution of probabilities over the action space [29, 12]. The Bellman Optimality equation [12] for finding the optimal action function Q^* is useful:

$$Q^*(s, a) = E[R_{t+1} + \gamma max_{a'} Q^*(s', a')] \tag{1}$$

RL algorithms can find the maximum $Q^*(s, a)$ to determine the optimal policy for any state s,without knowing information about the model. When full information about the MDP is known, the optimal policy can be found from background induction approaches [29].

2.2 Multi-agent Reinforcement Learning

The multi-agent approach generalizes the MDP to a stochastic game case or as extensive-form games [3, 29]. For a stochastic game approach, we can represent it as a tuple $(X, U_1, .., U_n, f, p_1, ..., p_n)$ [3] where:

- n is the number of agents,
- X is the finite set of environment states,
- U is a finite set of actions available to the agents,
- f is the state transition probability function, and
- P is the set of reward functions for the agents.

In EV driving, U represents the different possible routing decisions and X will be limited by the road network and existing EVs. The different learning approaches consist of three main strategies which describe the agent exploration behaviour: fully cooperative, fully competitive, and a mix of the two [3].

2.3 Competitive and Cooperative Learning

There are typically two groups of agents to represent the different multi-agent learning strategies, *competitive* agents and *cooperative* agents [17]. To transition a competitive environment to a cooperative one, the design of the reward function is critical to ensure the performance of the system [11], and to prevent agents taking selfish actions through communication. A weighted sum of sub-rewards can be applied and coupled with suitable observations to allow agents to learn multiple objectives during training.

An effective case in splitting the agents is using the idea of "Friend-or-Foe" [9]. The concept could be well-suited to queuing optimisation by setting a particular radius function to determine if an agent is an adversarial equilibrium, "foe", or a coordination equilibrium, "friend." This would allow agents to learn more specialised geospatial policies by different sets of "friendly agents" focusing on different areas of the network [5]. Silva et al. [5] extend the idea to a "Selfish Optimization procedure," where the friendly agents work together to build a cooperative policy. However, if an agent's needs become more urgent, then it overrides the policy and chooses in its own interest. This could be used, for example, to allow vehicles with very low batteries to charge at the nearest station even if it is detrimental to the strategy. This approach more realistically models human behaviour with recommendation systems as users cannot be expected to follow a policy if it means they are worse off. This links to the idea of providing a dynamic user equilibria system where no individual has any reason to deviate from the policy [13].

3 System Architecture

A MARL system design is proposed to expose the benefits of utilising cooperation and competition between agents. This involves designing an environment that supports multiple charging station agents, actions, observations and rewards as well as communication support. The MARL design process shown in Figure 1 illustrates a single learning step in the environment. The communication is centralized to improve scalability, in which each agent can access other agents' current states. This design also allows agents to share the same policy as each agent is learning to optimise the same goal. The system is designed to run on multiple MARL algorithms and return reward performances over time and compare against a baseline greedy algorithm that will simply select the closest station when on low-charge.

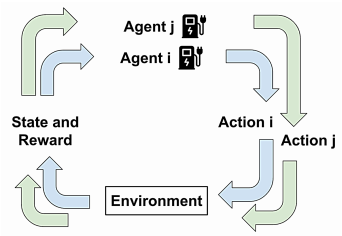


Figure 1: The MARL training loop

3.1 Competitive and Cooperative Agents

Cooperative learning can be achieved by updating the rewards effectively so that agents will instead be negatively rewarded by selfish actions. A combination of competitive and cooperative strategies can be achieved by allowing agents to cooperate with only N closest agents, where N is a parameter of the system. This "Friend-or-Foe" inspired strategy aids in the scalability of the system while aiming to improve the overall queuing balance of the environment.

3.2 Scaling the Simulation

By using other RL algorithms, rather than Q-learning which is limited by a Q-table, the system can be scaled more easily. For example, deep Q-learning networks use neural networks instead of a Q-table, providing greater capability for supporting larger and more complex simulations. OpenStreetMap (OSM) was used to export real-world networks as shown in 2a and Figure 2b, which are beneficial for presenting more realistic simulations.

(a) OpenStreetMap export preview (b) Converted network XML file in Netedit

Figure 2: Exporting and converting real-world maps to XML network files

4 Training

The open-source traffic simulation package, SUMO, was chosen to train the agents. SUMO provides extensive EV charging tools and flexible environment controls via the TRaCI API. The Netedit tool was used to create small test road networks to test the trained agents in different scenarios. A final training network was created from a snapshot of a real-world network in Berlin using OSM to train the agents to handle a variety of network scenarios.

4.1 Q-learning

The Q-learning algorithm discovers the optimal action-selection policy for a Markov Decision Process (MDP) without knowing the reward functions and what the transition is beforehand [25]. As the work of Wang and Silva suggests, [25], a typical Q-learning algorithm will "initialize the table entry $\hat{Q}(s_i, a_j)$ to zero, and initialize $\tau = 0.99$," then after checking the current state s the temporal difference algorithm will repeatedly "probabilistically select an action a_k with probability $P(a_k) = \frac{e^{\hat{Q}(s,a_k)/\tau}}{\Sigma_{l=1}^{m} e^{\hat{Q}(s,a_k)/\tau}}$, and execute it," this will allow the agent to immediately receive reward r and is then able to observe the new state s'. The table entry is updated:

$$\hat{Q}(s, a_k) = (1 - \epsilon)\hat{Q}(s, a_k) + \epsilon(r + \beta \max \hat{Q}[s', a']) \tag{2}$$

where, $0 \leq \epsilon < 1$ is the learning rate, $s \leftarrow s', \tau \leftarrow \tau^* 0.999$. After a sufficient number of iterations, the $\hat{Q}(s_i, a_j)$ will converge to the optimal value," [25].

The Q-learning off-policy and model-free algorithm learns the task directly through interactions with the environment without requiring explicit estimates of the system. The algorithm follows an epsilon-greedy strategy of sufficiently exploring the state space to learn optimal actions [4, 14, 14]. There is a trade-off between the exploration and exploitation of the system to maximise performance.

4.2 DQN

Deep Q-Networks (DQN) is a model-free RL algorithm which uses deep neural networks, unlike Q-learning which uses a Q-table, to approximate the action-value function, $Q(s, a)$ [26]. DQN stores the agent's experiences using a replay dataset, $\mathcal{D}$, used to "reduce correlations between observations" [7]. The dataset contains the state, action, reward and next transition state for each experience, represented as (s, u, r, s'). DQN uses a Q-network and Target network, with parameters θ and θ^-, that both take a random batch of training data from the experience replay dataset to predict the Q-value. The Predicted and Target Q-values from the networks combined with the reward are used to compute the loss, seen in the equation below, to train only the Q-Network following a temporal-difference approach. The Q-Network weights are then copied to the Target Network after a configured number of steps.

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')} \sim \mathcal{D}[(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i))^2][7]. \tag{3}$$

DQN is typically more data efficient than Q-learning since each step of experience can be used in multiple weight updates and a reduction in variance of the updates is achieved via randomizing the samples. Additionally, DQN avoids the problem of oscillations or divergence in parameters as it smooths the learning process by using an average distribution of experience replay behaviour over many past states [14]. The algorithm was implemented using RLlib to train the agents within the competitive and cooperative environments.

4.3 PPO

Proximal Policy Optimisation (PPO) is an on-policy RL algorithm that can be used within a multi-agent environment. PPO is a policy gradient method for RL.

Firstly a policy, π, is a function that given a state s will produce a viable action a. In the case of policy-based methods, this function can be established using adjustable parameters which can be modified to obtain a greater reward. For policy gradient methods, an estimator of the policy gradient is computed and used in a stochastic gradient ascent algorithm [19]. PPO employs two networks, one dedicated to representing the current policy to refine and the other serves to collect samples

Recent work [27, 2] has shown PPO and its variations to be effective in various MARL tasks. The PPO algorithm is implemented within the competitive and cooperative environments and the clipping range hyperparameter is set to constrain the policy update to be within 20% of the current policy.

4.4 Competitive Learning Environment

The initial custom SUMO environment developed supports solely competitive agent learning strategies. In this environment, the agents are said to be competitive as they are rewarded only for selfish actions, disregarding the other agents, and are competing to charge the limited vehicles to maximise their rewards. This environment was used to train agents on Q-learning, DQN, PPO and the greedy algorithm. The breakdown of this environment is as follows:

- **Agent:** Charging station agent has a corresponding lane and memory of actions.
- **Observation:** Current closest EV battery, wait time and lane density. The closest EV is the next closest, in driving distance, within a maximum range.

- **Action:** To charge or not charge an approaching EV.
- **Reward:** Determined by the battery, wait time and lane density values.

4.5 Cooperative Learning Environment

The previous environment is updated in various ways to enable cooperative agent behaviour. Firstly a busy value is introduced that takes the maximum of the normalized wait time and lane density values to reduce the observation space while still providing the necessary information for learning. The agent's observations are updated to include the busy values of the two closest agents that are within the range of the detected EV. This custom implementation of the "Friend-or-Foe" strategy adjusts the agent incentives to optimise the global reward of its friendly agents as the collective reward will be split. An example, shown in Figure 3, indicates the close friendly agents to station 1, indicated by red arrows, and the detected EV, indicated by the blue arrow.

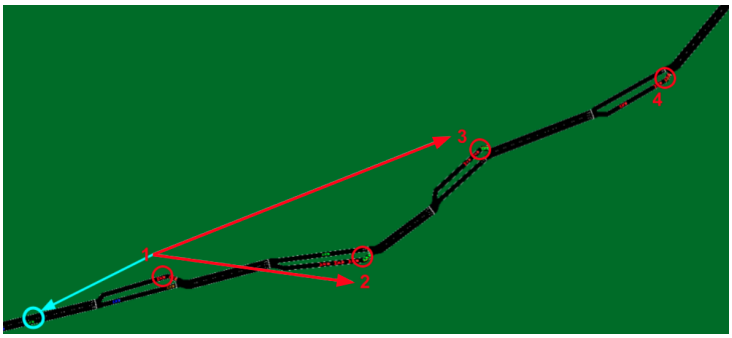
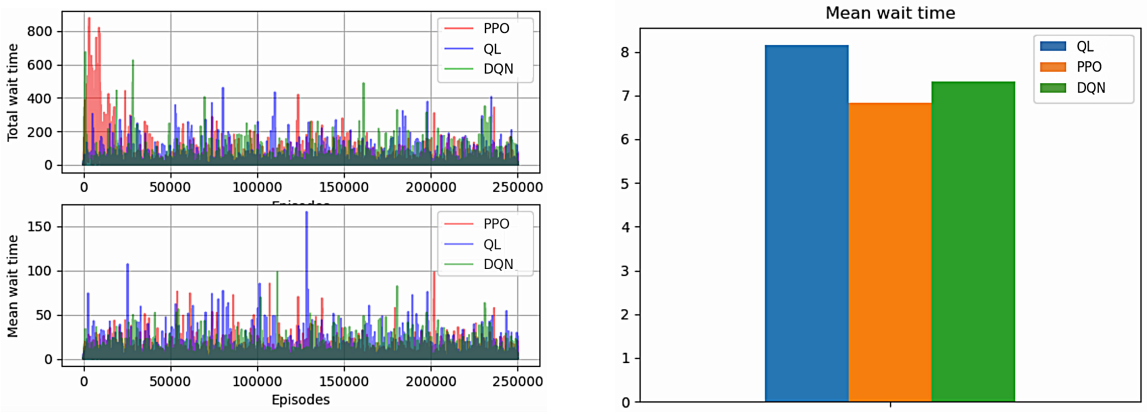


Figure 3: Friend-or-Foe Cooperative PPO on the 4-Station Strip network

5 Results

The agent behaviour during the training process with MARL algorithms was tested by analyzing the episode reward mean metric via TensorBoard performance logs in real-time, where episodes are the agent-environment interactions from start to end states. Additionally, the total wait times and penalties were recorded for each agent in the simulation.



(a) Waiting times

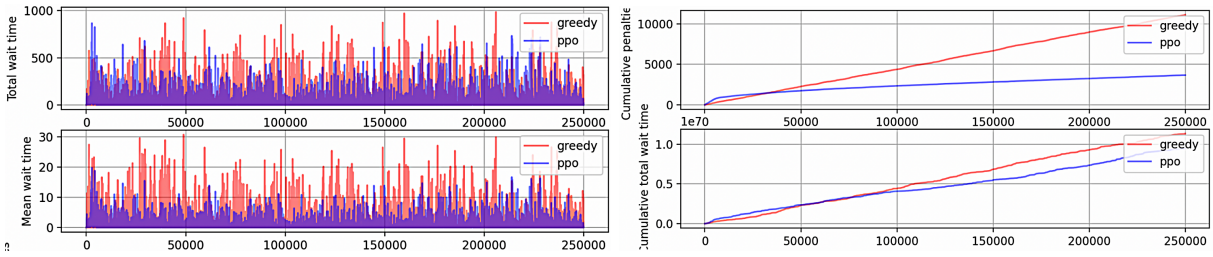
(b) Final run mean wait times

Figure 4: Competitive Environment PPO, Q-learning and DQN results

5.1 MARL in a Competitive Environment

The results, shown in Figure 4, of training on the 2-station strip network, shown in Figure 7, compares the different RL algorithms deployed in the multi-agent competitive system. Figure 4b displays the last 1,000 episodes performance for the different algorithms, finding PPO and DQN to show reduced waiting time over Q-learning, with PPO exceeding Q-learning by 16.32%.

PPO, the best performing algorithm in this experiment, was then compared against the greedy algorithm. PPO can be seen performing better in each metric recorded. The cumulative penalties initially for PPO is higher during training but after 10,000 episodes the agents appear to have learnt an improved policy, seen in Figure 5



(a) Greedy vs PPO wait times

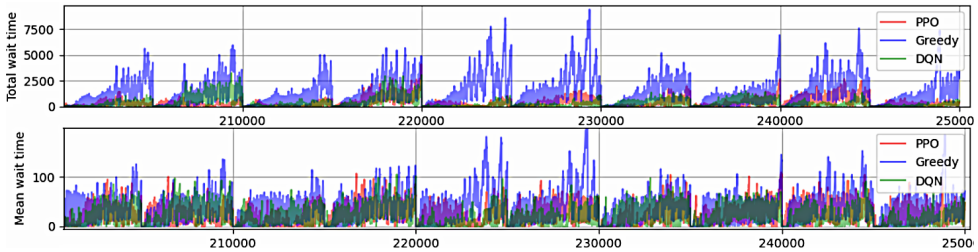
(b) Penalties and cumulative wait times

Figure 5: Competitive environment greedy and PPO training wait time results

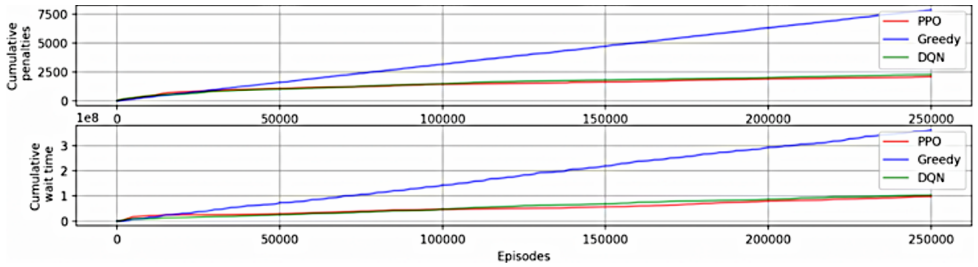
5.2 MARL in a Cooperative Environment

5.2.1 Test Network Scenario Results

A similar experiment is conducted for the cooperative environment with adjustments to the simulation parameters to ensure significant queuing. Q-learning algorithm is not used due to the limits of the Q-table with the environments continuous observation values. The 2-station strip results can be seen in Figure 6. The gradual spikes, seen in Figure 6a, demonstrate how each algorithm deals with the queuing of EVs throughout the training iterations. The results display the MARL algorithms to greatly outperform the greedy approach.



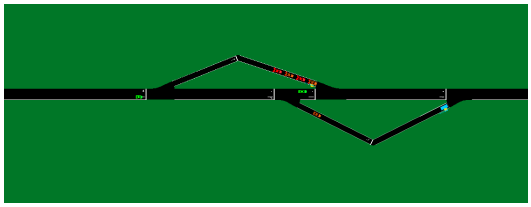
(a) Final episodes snapshot



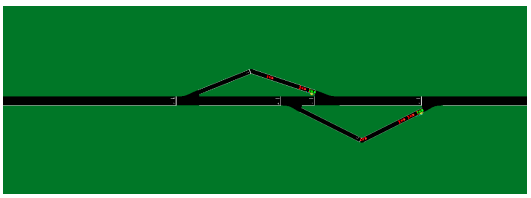
(b) 2-Station network wait times and penalties

Figure 6: 2-Station wait time training results

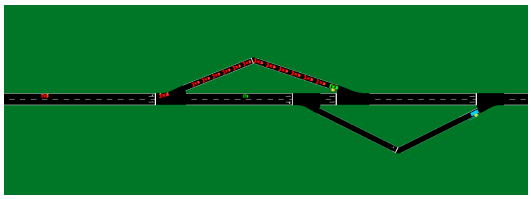
The cooperative strategy between agents has also improved the distribution between the agents and the overall demand balancing, as seen in Figure 7a and 7b. Figure 7c and Figure 7d display the best demand balancing algorithm, PPO, greatly outperforming the greedy method for this network since most of the EVs go to the first station for the greedy algorithm while PPO makes better use of both stations to distribute the low charge EVs. The EVs use colour-coding to visualise the battery levels, with more red for low charge, more green for full charge and blue for non-EVs. The light blue squares indicate a charging station.



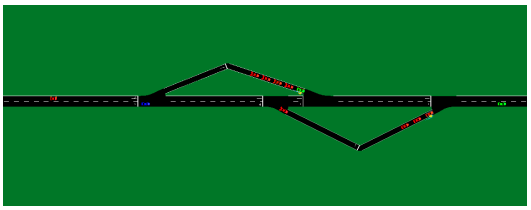
(a) Competitive DQN



(b) Cooperative DQN



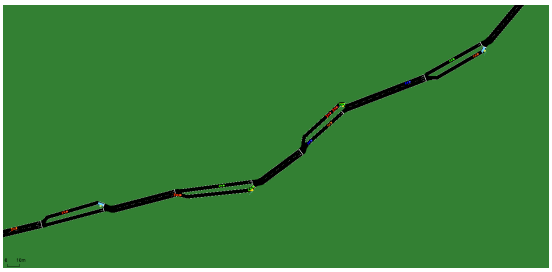
(c) Greedy



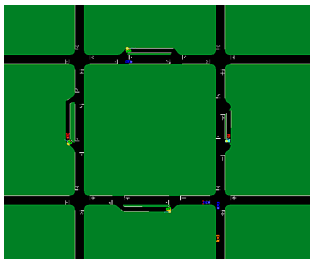
(d) Cooperative PPO

Figure 7: Trained DQN, PPO and greedy experiments comparing competitive and cooperative systems

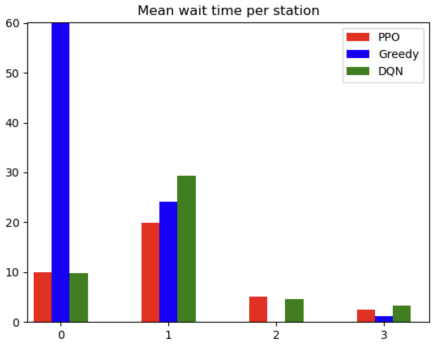
The other test networks were also run on this environment, including the 4-station strip and 4x4 grid networks. The agents were similarly able to learn to distribute the EVs between each other to best reduce the queues compared with the greedy method, seen in Figure 8.



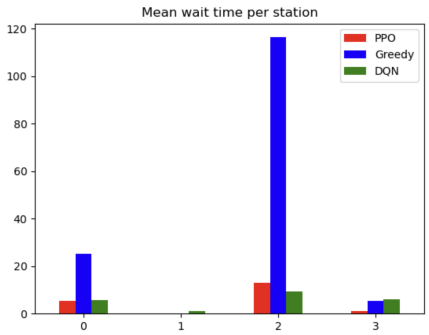
(a) Cooperative DQN 4-station strip



(b) Cooperative PPO 4x4 Grid



(c) 4-Station strip wait times

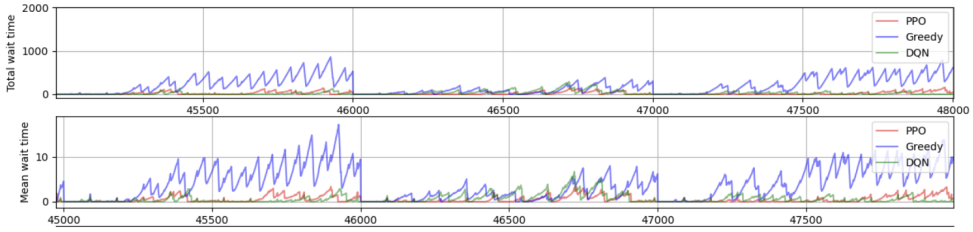


(d) 4x4 Grid wait times

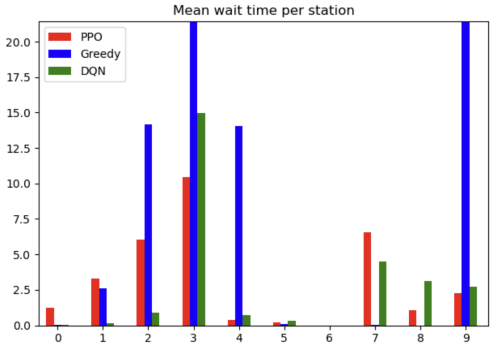
Figure 8: Cooperative MARL results and final episode agent wait times

5.2.2 Berlin Network

The results on the real-world Berlin network indicate the advantages of the MARL implementations with PPO and DQN[1] reducing the mean total wait time compared with the greedy baseline by 87.21% and 88.09% respectively, seen in Figure 9. Reflecting on these results, the MARL algorithms consistently present advantageous queueing outcomes for EVs. In both the competitive and cooperative environments, the PPO and DQN algorithms show similar optimisation results. Further hyperparameter tuning could be applied to these algorithms to further improve the results.



(a) Berlin final training episodes wait time results



(b) Agent wait times

(c) Cooperative PPO on a Berlin network snapshot

Figure 9: Berlin networks agent learning results

6 Conclusions

The reported findings expose several effective MARL implementations to reduce the EV queuing problem. PPO and DQN multi-agent approaches were shown to be the most promising RL algorithms and the integration of cooperative strategies within the reward function additionally indicated reduced collective wait time by taking other agents' observations and actions into account. The experiments overall suggest the potential practical application of MARL algorithms to improve the balance in demand at EV charging stations in real-world environments, which can be achieved through communicating the learnt policy suggestions to EV drivers.

7 Future Work

This paper focuses on reducing the wait time at EV charging stations, however, there are many other important features that drivers will be interested in that could be extended upon. This may include the price, total distance and out-of-service information of the chargers in the observations and rewards of the environment. The simulation could also be extended to increase the number of cooperative connections an agent shares to find an optimal cooperative-competition ratio. With greater computer resources the simulation could be scaled up to include more agents and EVs to closer match real-world scenarios. Furthermore, additional RL algorithms could be experimented on, for example, SAC, DDPG or QMIX

[1] https://github.com/RoryCoulson/sumo-ev-marl

could show improved results. Lastly, combining real traffic data into the simulation will provide an improved understanding of how these algorithms can perform in real-world applications to improve the justification of their use.

8 Acknowledgement

The authors acknowledge the financial support received from the Engineering and Physical Sciences Research Council (EPSRC) through a Turing AI Acceleration Fellowship on Citizen-Centric AI Systems (EP/V022067/1) and the Future Electric Vehicle Energy Networks supporting Renewables (FEVER) grant (EP/W005883/1).

References

[1] Meryem Abid, Mohammed Tabaa, Asmae Chakir, and Hanaa Hachimi. Routing and charging of electric vehicles: Literature review. *Energy Reports*, 8:556–578, 2022.

[2] James Ault and Guni Sharon. Reinforcement learning benchmarks for traffic signal control. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[3] Lucian Buşoniu, Robert Babuška, and Bart De Schutter. *Multi-agent Reinforcement Learning: An Overview*, pages 183–221. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[4] Jesse Clifton and Eric Laber. Q-learning: Theory and applications. *Annual Review of Statistics and Its Application*, 7:279–301, 2020.

[5] Felipe Leno Da Silva, Cyntia EH Nishida, Diederik M Roijers, and Anna H Reali Costa. Coordination of electric vehicle charging through multiagent reinforcement learning. *IEEE Transactions on Smart Grid*, 11(3):2347–2356, 2019.

[6] Mathijs M. de Weerdt, Sebastian Stein, Enrico H. Gerding, Valentin Robu, and Nicholas R. Jennings. Intention-aware routing of electric vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 17(5):1472–1482, 2016.

[7] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In Gita Sukthankar and Juan A. Rodriguez-Aguilar, editors, *Autonomous Agents and Multiagent Systems*, pages 66–83, Cham, 2017. Springer International Publishing.

[8] Yexin Li, Yu Zheng, and Qiang Yang. Efficient and effective express via contextual cooperative reinforcement learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 510–519, 2019.

[9] Michael L. Littman. Friend-or-foe q-learning in general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 322–328, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[10] Mostafa Mahdy, AbuBakr S. Bahaj, Philip Turner, Naomi Wise, Abdulsalam S. Alghamdi, and Hidab Hamwi. Multi criteria decision analysis to optimise siting of electric vehicle charging points—case study winchester district, uk. *Energies*, 15(7):2497, Mar 2022.

[11] Hangyu Mao, Zhibo Gong, and Zhen Xiao. Reward design in cooperative multi-agent reinforcement learning for packet routing. *CoRR*, abs/2003.03433, 2020.

[12] Deepanshu Mehta. State-of-the-art reinforcement learning algorithms. *International Journal of Engineering Research and Technology*, 8:717–722, 2020.

[13] Igal Milchtaich. The equilibrium existence problem in finite network congestion games. In Paul Spirakis, Marios Mavronicolas, and Spyros Kontogiannis, editors, *Internet and Network Economics*, pages 87–98, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[14] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[15] Tao Qian, Chengcheng Shao, Xuliang Li, Xiuli Wang, Zhiping Chen, and Mohammad Shahidehpour. Multi-agent deep reinforcement learning method for ev charging station game. *IEEE Transactions on Power Systems*, 37(3):1682–1694, 2021.

[16] NR Ravishankar and MV Vijayakumar. Reinforcement learning algorithms: survey and classification. *Indian J. Sci. Technol*, 10(1):1–8, 2017.

[17] Heechang Ryu, Hayong Shin, and Jinkyoo Park. Cooperative and competitive biases for multi-agent reinforcement learning. *CoRR*, abs/2101.06890, 2021.

[18] Md Safayatullah, Mohamed Tamasas Elrais, Sumana Ghosh, Reza Rezaii, and Issa Batarseh. A comprehensive review of power converter topologies and control methods for electric vehicle fast charging applications. *IEEE Access*, 10:40753–40793, 2022.

[19] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[20] Elnaz Shafipour and Sebastian Stein. Electric vehicle charging on long journeys: Current challenges and future opportunities, 2022.

[21] Sai Shao, Wei Guan, and Jun Bi. Electric vehicle-routing problem with charging demands and energy consumption. *IET Intelligent Transport Systems*, 12(3):202–212, 2018.

[22] Nitin Trivedi, Nikhil S. Gujar, Subrata Sarkar, and S.P.S. Pundir. Different fast charging methods and topologies for ev charging. In *2018 IEEMA Engineer Infinite Conference (eTechNxT)*, pages 1–5, 2018.

[23] Guang Wang, Yongfeng Zhang, Zhihan Fang, Shuai Wang, Fan Zhang, and Desheng Zhang. Faircharge: A data-driven fairness-aware charging recommendation system for large-scale electric taxi fleets. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(1):1–25, 2020.

[24] Yanan Wang, Tong Xu, Xin Niu, Chang Tan, Enhong Chen, and Hui Xiong. STMARL: A spatio-temporal multi-agent reinforcement learning approach for traffic light control. *CoRR*, abs/1908.10577, 2019.

[25] Ying Wang and Clarence W. De Silva. Multi-robot box-pushing: Single-agent q-learning vs. team q-learning. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3694–3699, 2006.

[26] Zhuoran Yang, Yuchen Xie, and Zhaoran Wang. A theoretical analysis of deep q-learning. *CoRR*, abs/1901.00137, 2019.

[27] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre M. Bayen, and Yi Wu. The surprising effectiveness of MAPPO in cooperative, multi-agent games. *CoRR*, abs/2103.01955, 2021.

[28] Yukun Yuan, Desheng Zhang, Fei Miao, Jimin Chen, Tian He, and Shan Lin. pˆ 2charging: Proactive partial charging for electric taxi systems. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pages 688–699. IEEE, 2019.

[29] Kaiqing Zhang, Zhuoran Yang, and Tamer Basar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *CoRR*, abs/1911.10635, 2019.

[30] 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*, WWW '21, page 1856–1867, New York, NY, USA, 2021. Association for Computing Machinery.

Authors

Rory Coulson is currently completing his MEng degree in computer science and artificial intelligence at the University of Southampton. His research interests include reinforcement learning, artificial intelligence and machine learning applied to real-world problems.

Elnaz Shafipour She is a Research Fellow at the School of Electronics and Computer Science at the University of Southampton as part of the Citizen-Centric AI Systems. She received her PhD from Lancaster University in September 2021. Her research focuses on AI and Multi-Agent Systems, as well as online planning and preference elicitation. She is also co-founder and director at EVtonomy Ltd (https://evtonomy.com). At EVtonomy she applies world-leading AI research to offer electric vehicle drivers ultra-personalised routes that balance factors like cost, charging speed, facilities and predicted queueing times.

Sebastian Stein completed an MEng at the University of Warwick in 2004 and then a PhD at the University of Southampton in 2008, both in Computer Science. He is a Professor of Artificial Intelligence within the School of Electronics and Computer Science at the University of Southampton. His research focuses on multi-agent systems, incentive engineering and citizen-centric AI. He has applied his work to challenges around smart energy and smart transportation, including electric vehicle charging. He currently holds an EPSRC Turing AI Acceleration Fellowship (https://ccais.ac.uk).

Jan Buermann obtained a BSc and MSc in computer science from the University of Paderborn (Germany) in 2015 and 2017, respectively, and completed his PhD in computer science at the University of Southampton in 2022. He is a Research Fellow in Citizen-Centric AI Systems at the School of Electronics and Computer Science at the University of Southampton. His research focuses on incentive engineering and resource optimisation in multi-agent systems under uncertainty. He applies this to address challenges in smart energy systems, transportation electrification and the wider acceleration of decarbonisation towards net zero.

Suleiman M. Sharkh obtained his BEng and MSc from the University of Southampton in 1990 and 1994, respectively. He is Professor of Power Electronics, Machines and Drives at the University of Southampton. His research interests include electric machines, power electronics and their applications in transport, renewable energy and microgrids. He is a Senior Member of the IEEE, a member of the IET and a Chartered Engineer.

Andrew Cruden (FIET) is Professor of Energy Technology within the Dept. of Mechanical Engineering at the University of Southampton. He is currently the Associate Dean (Infrastructure) for the Faculty of Engineering and Physical Sciences (FEPS) and he is the Principal Investigator (PI) for the FEVER project (https://www.fever-ev.ac.uk), an EPSRC Programme Grant. He is an electrical engineer by background, and his main area of research is in energy storage, specifically batteries, flow cells, fuels cells and alternative fuels, with a focus on applications within increasing electrification of all forms of transport.