Optimizing Urban Traffic Flow: From Traditional TSC to Multi-Agent Reinforcement Learning

Zihua Ding[1,*,†], Yanbin Hou[2,†], Yunfan Zhang[3,†]

{george211707@mail.ustc.edu.cn[1], houyanbin010@gmail.com[2], tony923923923@outlook.com[3]}

University of Science and Technology of China, Hefei, 230026, China[1]
The University of Sheffield, Sheffield, S10 2TN, UK[2]
North Eastern University, Shenyang, 110169, China[3]
[*]corresponding author
[†]This authors contributed equally to this work and should be considered co-first authors

Abstract. Traffic congestion in emerging megacities is exacerbated by rapid population growth and urbanization. Traditional Traffic Signal Control (TSC) methods struggle with dynamic and unpredictable conditions, facing computational and storage challenges. This paper explores modern TSC methods using advanced technologies, focusing on reinforcement learning (RL) and its variants like Deep Reinforcement Learning (DRL) and Deep Deterministic Policy Gradient (DDPG). Given the impracticality of centralized RL for large-scale Adaptive Traffic Signal Control (ATSC), we investigate Multi-Agent Reinforcement Learning (MARL) algorithms, specifically the Multi-Agent Advantage Actor-Critic (MA2C) algorithm, to address scalability and partial observability. Using SUMO for simulation, we compare various TSC algorithms, noting that RL-based algorithms, particularly DDPG and A2C, outperform traditional methods in terms of travel time.

Keywords: Traffic Signal Control, Reinforcement learning, Deep Q-Network, Advantage Actor-Critic Algorithm

1 Introduction

With the rapid growth of the global population and accelerated urbanization, “megacities” are emerging worldwide. While we enjoy the convenience and prosperity of urban development, we must not overlook the drawbacks of traffic congestion. Due to population growth, industrialization, economic expansion, and technological advancements, the number of vehicles on the roads is increasing, putting pressure on existing urban transportation infrastructure and leading to daily traffic jams. Additionally, congestion results in longer travel times, increased fuel consumption, higher costs, and environmental pollution. If traffic issues are not addressed, they will inevitably lead to a decline in the quality of social life and limit urban development.

Traffic signal control serves as a fundamental solution to urban traffic congestion. Since intersection delays constitute a significant portion of overall travel time, optimizing intersection signal planning has become a critical focus. Strategic coordination of signal changes aims to minimize vehicle waiting times, prevent traffic congestion, and enhance safety for pedestrians and vehicles [1]. Traditional Traffic Signal Control (TSC) methods can be classified into three distinct categories:

CONF-MLA 2024, November 21, Adana, Turkey

DOI 10.4108/eai.21-11-2024.2354610

a. Deterministic TSC implements fixed-time control systems derived from historical traffic patterns, utilizing the Webster formula to calculate optimal traffic phase durations.

b. Semi-dynamic TSC incorporates actuated control mechanisms that respond to real-time traffic conditions, such as triggering green lights only when detecting vehicle presence.

c. Fully dynamic TSC employs sophisticated actuated control systems that adapt signal timing based on comprehensive traffic metrics, including average vehicle waiting times and queue lengths over extended periods.

However, these methods have significant limitations [2]. Firstly, they are unable to adapt to dynamic and unpredictable traffic conditions. Traditional deterministic and semi-dynamic TSC methods cannot dynamically adjust according to real-time traffic situations, making them less effective in handling unexpected events and traffic changes. Secondly, there are computational complexity and storage capacity issues. Fully dynamic TSC methods require exploring many state-action pairs, leading to long learning times and high storage demands.

Modern traffic signal control methods leverage advanced technologies and algorithms for real-time optimization to address these limitations. The current mainstream approach combines reinforcement learning with traffic management, resulting in various reinforcement learning-based TSC algorithms, such as DRL and DQN, and the improved DDPG algorithm [3] based on DQN. However, because of the high dimensionality of the joint action space, centralized reinforcement learning (RL) is not feasible for large-scale ATSC [4]. Researchers have proposed MARL algorithms to overcome this issue by assigning control to local RL agents at each intersection. Nevertheless, MARL introduces new challenges, such as partial observability, where each local agent can only observe limited local information and cannot access the global traffic situation. The Multi-Agent Advantage Actor-Critic (MA2C) algorithm can be introduced to address these challenges in multi-agent systems.

In summary, each algorithm, from RL to A2C, has strengths and weaknesses. This paper will introduce the environment setup for TSC, the specific mechanisms of each algorithm, and their comparisons, using SUMO for simulation and modeling.

2 Related Work

Existing literature on traffic research identifies several causes of traffic congestion, including rapid urban population growth, increased vehicle ownership, inadequate road infrastructure development, and poor urban planning and management. An analysis of traffic control measures in Latin America highlights the importance of optimizing urban planning and implementing traffic demand management [5]. Artificial intelligence, big data analytics, and traffic simulation technologies are proposed to develop more effective solutions.

Early work has highlighted the limitations of traditional Traffic Signal Control (TSC) methods [6]. Reinforcement Learning (RL) has been applied to fully dynamic TSC, but it suffers from the “curse of dimensionality,” leading to high computational costs and storage requirements. Deep Reinforcement Learning (DRL), which combines deep learning and RL, shows promise in overcoming these limitations by enabling continuous state space representation, reducing learning time, and effectively approximating Q-values.

DRL offers several advantages for TSC, such as adapting to real-time traffic conditions, model-free learning, and considering multiple performance factors in the reward function.

In addition to traditional RL, existing literature compares various TSC algorithms to minimize wait times and queue lengths to improve traffic flow. The authors model intersections as Markov Decision Processes and explore various methods, including round-robin schedulers, feedback control mechanisms, and two RL techniques—Advantage Actor-Critic (A2C) algorithms and Deep Q-Network (DQN) [7]. These algorithms were tested in a simulated environment of a real intersection in Bangalore, India, using traffic data randomly generated according to a Weibull distribution.

A Multi-Agent Deep Reinforcement Learning (MARL) algorithm has been proposed for solving Adaptive Traffic Signal Control (ATSC) problems on large scales [8]. MARL has eliminated the scalability problem of centralized learning by assigning control to local RL agents at each intersection. However, MARL introduces new challenges, as the overall situation becomes limited accessible from the perspective of each local agent due to limited communication. Subsequent research [9] proposed a Multi-Agent Advantage Actor-Critic (MA2C) algorithm to address these challenges. Two methods were introduced to stabilize the learning process: integrating observations and footprints of neighboring agents into the state to improve observability and introducing a spatial discount factor to decrease the impact of neighboring agents' observations and reward signals. This approach was evaluated on a real large-scale traffic network in Monaco, demonstrating superior robustness, optimality, and sample efficiency compared with other decentralized MARL algorithms.

The Deep Deterministic Policy Gradient (DDPG) algorithm, developed by the DeepMind team, is an online deep reinforcement learning algorithm specifically designed for continuous control problems [10]. It borrows concepts from the Deep Q-Network (DQN) algorithm. When solving continuous action space problems, there are two main approaches: discretizing the continuous actions and then using RL algorithms (e.g., DQN) to solve them, or introducing Policy Gradient (PG) algorithms (e.g. Reinforce) directly. However, discretization can deviate from practical engineering applications, and PG algorithms often perform poorly in continuous control problems. The DDPG algorithm was proposed to address these issues and achieved remarkable results in many continuous control situations.

3 Experimental Setting

All simulations in this study were conducted using SUMO, with interactions facilitated through the Traffic Control Interface (TraCI) and Python. We focus on a multi-agent decision-making problem for network system control, where each agent's observability and communication are limited to its neighborhood. We used Python 3.11, TensorFlow 1.15, and SUMO 1.20.0 for our experiments. Figure 1 is an introduction to the algorithms:

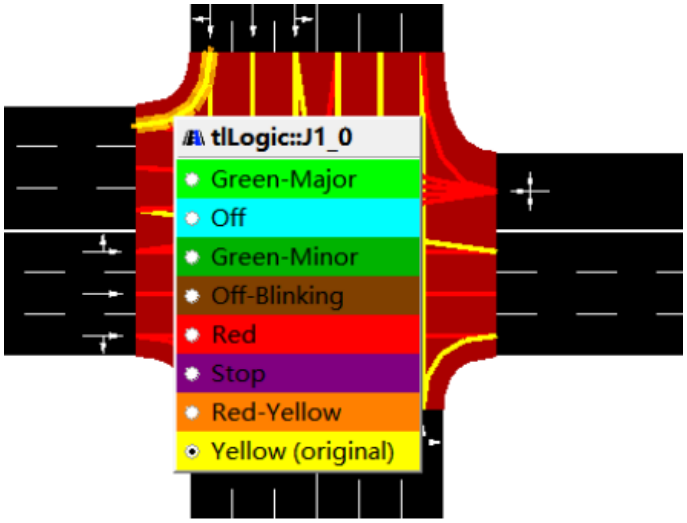

Fig. 1. Fixed Time TSC Control (Built-in SUMO)

3.1 RL and DRL

Reinforcement Learning (RL) allows the agent to exploit and explore variable state-action pairs to maximize positive rewards or minimize negative costs, thereby enhancing system performance. In each time step t of the RL algorithm, the agent observes its Markov decision factor (state st) in a dynamic and random operating environment, selects an action at, observes the next state $s_t + 1$, and receives an immediate reward or cost $r_t + 1(s_t + 1)$. Subsequently, the agent updates the Q value of the state-action pair (st, at), indicating the action in state st is appropriate. This process continues until convergence or a predetermined termination condition is achieved. Q-value updates are typically performed using the Bellman equation, with the action value function Q(s,a) expressed as:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_t + 1 + \gamma \max_a Q(s_t + 1, a) - Q(s_t, a_t)] \quad (1)$$

DRL (Deep Reinforcement Learning) is an advanced form of RL that leverages deep learning (DL) [11]. DRL utilizes deep neural networks (DNNs) to approximate the Q function, addressing RL's "curse of dimensionality" problem. It can be implemented as either a model-based or model-free method, allowing agents to autonomously learn without fully understanding their operating environment, such as traffic conditions and network dynamics. DRL represents system goals and performance indicators by designing a reward function considering multiple factors influencing system performance. Compared to other methods, DRL offers numerous advantages for solving traffic signal control problems [12].

3.2 Deep Q-Learning

Deep Q-Network (DQN) is a multi-agent method based on reinforcement learning that uses neural networks to approximate the Q-value function. In DQN, the traffic signal control for each arm is managed by an independent agent responsible for learning and decision-making. The agent observes the environment state s_t, selects an action a_t according to the Markov Decision

Process (MDP), and receives a corresponding reward $r_{\{t+1\}}$ The agent's goal is to choose a series of actions to maximize the cumulative reward. For DQN algorithm, deep neural networks are used to estimate the optimal Q-function $Q*(s,a)$.

The network's input is a state vector of length 36, and the output is the Q-values for four possible actions. The state representation follows a method similar to that in the literature, where each lane is divided into nine cells of different sizes, represented by a 36-length boolean vector indicating the presence of vehicles in each cell. The agent continuously updates the Q-function through experience replay and temporal difference (TD) learning to better estimate the optimal Q-function and make more optimal decisions.

Overall, DQN is a multi-agent method based on deep reinforcement learning in traffic signal control, capable of effectively learning optimal traffic signal control strategies. The function for Q-Learning updates using neural networks is:

$$Q(s,a;\theta) \leftarrow Q(s,a;\theta) + \alpha[r + \gamma max_{a'}\ Q(s'\ ,a'\ ;\theta) - Q(s,a;\theta)] \quad (2)$$

3.3 A2C or MA2C

A2C (Advantage Actor-Critic) is a synchronous Actor-Critic algorithm that improves training efficiency through multi-threaded parallelization (Figure 2).

In the Actor-Critic method, a neural network approximates the action-value function $Q\hat{}\pi(s,a)$. This neural network is called the "value network," denoted as $q(s,a;\ w)$, where w represents the trainable parameters of the neural network. The input to the value network is the state s, and the output is the value of each action. If the action space A has several actions, the output of the value network is a vector with each element corresponding to an action.

To address inconsistencies, the coordinator in A2C waits for all parallel actors to complete their tasks before renewing the global parameters. In the next iteration, the parallel actors use the same policy. Synchronous gradient updates make the training more cohesive and potentially faster in convergence. A2C has been shown to utilize GPUs more effectively and perform better at large scales. Its core idea is to combine the Actor and Critic networks [13]:

- **Actor**: Responsible for selecting actions based on the policy $\pi(a|s)$

$$\theta \leftarrow \theta + \alpha\nabla_\theta \log\pi_\theta(a|s)A(s,a) \quad (3)$$

- **Critic**: Responsible for evaluating the value of the actions selected by the Actor, based on the value function $V(s)$

$$L(\phi) = E[(r + \gamma V(s'\ ) - V(s))^2] \quad (4)$$

A2C improves training stability and efficiency by using the Advantage function

$$A(s,a)\ =\ Q(s,a)\ -\ V(s) \quad (5)$$

to reduce variance in the policy gradient. Specifically, it executes multiple environment instances in parallel, collects experiences, and then synchronously updates the Actor and Critic networks.

The MA2C (Multi-Agent Advantage Actor-Critic) algorithm is a distributed version of the A2C algorithm designed to address scalability and partial observability challenges in Adaptive Traffic Signal Control (ATSC) problems.

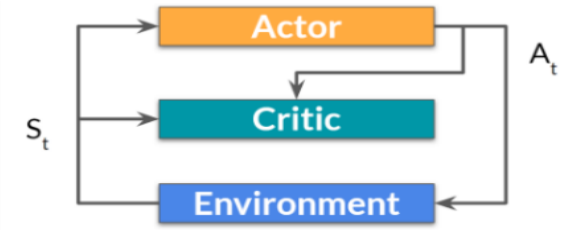

Fig. 2. A2C Algorithm

3.4 DDPG

The Deep Deterministic Policy Gradient (DDPG) algorithm simultaneously learns a Q-function and a policy. Specifically, it exploits off-policy data and the Bellman equation to learn the Q-function and then uses it to learn the policy. The DDPG algorithm architecture uses dual neural network model architecture for both policy function and value function, which makes the learning process of the algorithm more stable and the convergence speed speeds up. At the same time, the algorithm introduces the empirical playback mechanism, Actor interaction with the environment is stored in the empirical pool, and batch data samples are extracted for training, making the algorithm easier to converge. This approach is closely related to Q-learning and shares the same motivation: finding the best action-value function Q*(s,a) at any given state:

$$a^*(s) = arg\ max_a Q^*(s,a) \tag{6}$$

DDPG alternates between learning an approximation of Q*(s,a) and learning an approximation of a*(s), which is particularly well-suited for environments with continuous action spaces. In such environments, there are numerous actions to consider when computing $max_a Q*(s,a)$. While this is manageable with a finite number of discrete actions because we can compute each action's Q-value separately and compare them directly, it becomes challenging in continuous action spaces where exhaustively evaluating all possibilities is not feasible. Using a regular optimization algorithm would be prohibitively expensive due to the need for frequent computation of $max_a Q*(s,a)$ every time the agent takes an action in the environment. However, since we can assume that the function Q*(s,a) is differentiable wt ncerningthe action parameters in continuous action spaces, we can develop an efficient gradient-based learning rule for policy $\mu(s)$, exploit this fact, and approximate it with $max_a Q*(s,a) \approx Q(s,\mu(s))$ while still being able to find a near-optimal policy without incurring excessive computational costs.

3.5 Other Methods

For comparison, we also included the Webster method in our algorithm implementation; the traffic signal max-pressure algorithm based on network flow theory aims to reduce vehicle congestion at intersections. Fig 3 shows the max pressure control. This method allocates green light time by prioritizing the most pressured intersections to improve overall traffic efficiency. Additionally, we incorporated the adaptive signal control algorithm, which allows traffic lights to dynamically adjust control strategies based on real-time traffic conditions. These are more

traditional algorithms that do not use reinforcement learning. We included them in the comparison to demonstrate the superiority of reinforcement learning algorithms [14].

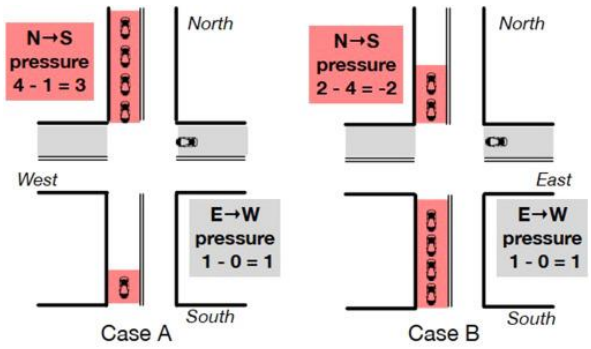

Fig. 3. Max pressure control

3.6 Sumo Introduction

In the past decade, SUMO has developed into a comprehensive traffic modeling utilities, including roads, capable of reading network importer formats, requirement generation, and routing utilities from different sources. They use multiple input sources, such as raw target matrices, traffic counts, etc., and can be used for high-performance simulation of individual intersections and entire cities. They include remote control interfaces to adapt to online simulations and a large number of additional tools and scripts.

3.7 Sumo Network Settings

The SUMO network consists of nodes and unidirectional edges representing streets, bike lanes, sidewalks, etc. Several line segments describe each edge and consists of one or more parallel running lanes. Width, speed limits, and access rights are set as parameters along the lane. The SUMO network includes detailed information about the intersection structure, traffic flow, and the corresponding road rules used to determine the simulated behavior. The SUMO road network represents the real-world network as a graph, where nodes are intersections and roads are represented by edges. The intersection is determined by location, shape, and right-of-way rules. An edge consists of two nodes and contains a fixed number of channels.

3.8 OpenStreetMap

We use OpenStreetMap to download road network environment information and extract real-world traffic road environments OpenStreetMap has a wide range of users from different regions, and due to its focus on collecting more local and on-site data, OpenStreetMap has rich and accurate map resources. The open street map includes both spatial and characteristic parameters. Spatial parameters mainly include three types: points, roads, and relationships, which make up the entire map image.

4 Results

We primarily focus on the total queueing time of vehicles under various algorithms. We replicated the algorithm comparison studied by a professor at Cornell University and used the metric library in Python to create comparison charts. The comparison of travel is as follows (Figure 4) [15]:

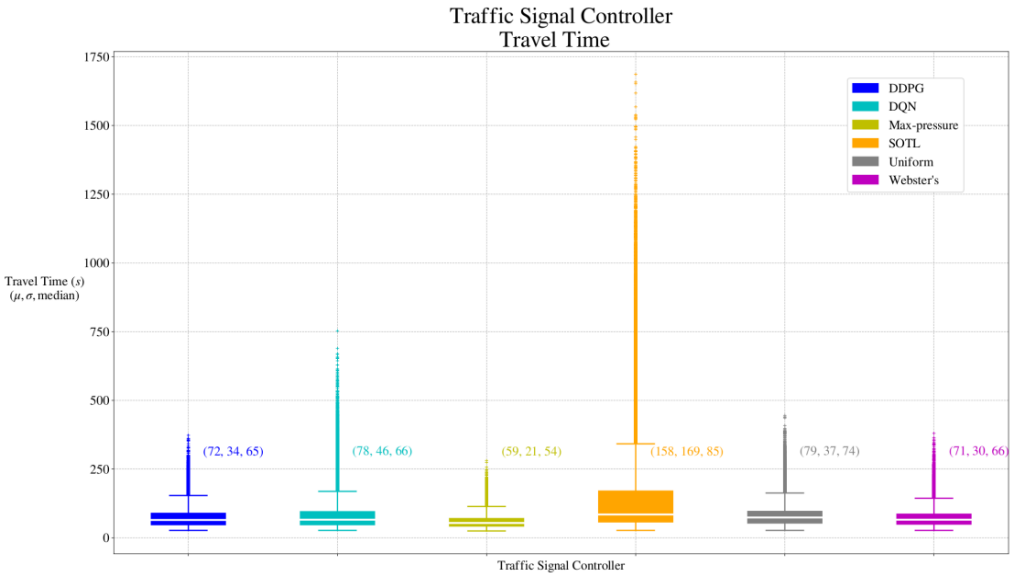

Fig. 4. Traffic Signal Controller Travel Time

The key observations are:

1. DDPG and DQN controllers have relatively lower median travel times, indicating better performance in reducing travel time.

2. Max Pressure and SOTL controllers show higher variability in travel times, with some outliers indicating occasional longer travel times.

3. Uniform and Webster's controllers have higher median travel times than IDQR and DQN, suggesting less efficient traffic flow management.

The chart highlights that DDPG and DQN controllers are more effective in minimizing travel times, while Max Pressure and SOTL show potential but with more variability. Uniform and Webster's are less effective in comparison.

We analyzed and reproduced the experiment and success of Tianshu Chu et al [16]. We created an agent using the TensorFlow instruction library and trained it to implement the A2C algorithm. We set the initial seed as a random number with a total step size of 100. After training, we tested and evaluated the model. By analyzing the results, we compared the optimization of ATSC achieved by other methods. We also constructed an MA2C model and evaluated the results after training MA2C in both synthetic high-traffic grids and real-world high-traffic networks, with

excellent special optimized traffic dynamics to ensure a certain difficulty level for MDP. Numerical experiments have confirmed that MA2C outperforms IA2C and state-of-the-art IQL algorithms regarding robustness and optimality. Although traditional ATSC algorithms have achieved good results in intersection optimization, they still have a slight disadvantage compared to RL based algorithms, especially in the collaborative optimization of intersections with large traffic changes and adjacent intersections.

In recent years, the implementation of RL in ATSC has been extensively studied. C. Cai et al. designed heuristic state features, S. Richter used an actor-critic algorithm to improve LR's fitting accuracy and optimization effect in ATSC, and T. Chu et al. verified the superior ability of Q-learning in simplifying traffic environment. On the one hand, our work comprehensively studied the practice of MARL in ATSC, and on the other hand, compared various proposed algorithms in the benchmark environment. We first extended the observation results of IQL to the actor-critic method to formulate IA2C. In addition, we experimented with two methods proposed by Tianshu Chu et al. for stabilizing IA2C to MA2C. In the constructed MA2C method, information on neighborhood policies is included to improve the observability of each local agent. Policy information sharing among intelligent agents helps them better coordinate their actions, thereby improving overall performance. By introducing spatial discount factors, we have improved the stability of the MA2C learning process. Under limited communication and local observation conditions, MA2C achieved stable convergence.MA2C exhibits the best and most robust learning ability, steadily increasing its training curve before becoming stable in narrow shadows.

5 Conclusion

This article introduces various algorithms of TSC and their expected optimization objectives, and their environmental settings, and uses SUMO to simulate and model them. In recent years, their respective advantages and limitations have been compared among the methods mentioned, including A2C and DQN. Although traditional TSC methods have limitations in dealing with dynamic traffic conditions, modern RL based methods provide the possibility of real-time optimization. However, these methods face computational and storage challenges when extended to large-scale ATSC. Researchers attempt to overcome local observability and other challenges in multi-agent environments to improve traffic signal control algorithms' optimization level and stability. We believe that by applying MARL, especially the MA2C algorithm, the performance and robustness of traffic signal control systems can be improved while maintaining localized control. This article demonstrates the application and effectiveness of these algorithms in TSC problems through simulation and comparative analysis. Table 1 includes a variety of reinforcement learning (RL) algorithms.

Table 1. The table includes a variety of reinforcement learning (RL) algorithms

Algorithm	Pros	Cons	Speed	Accuracy
Deep Q-Network (DQN)	1. Stable training 2. 2. Simple and efficient	1. Difficulty in handling continuous actions 2. Poor performance in high-dimensional state spaces	Low	Low

Deep Deterministic Policy Gradient (DDPG)	1. Better performance in high-dimensional action spaces 2. Suitable for continuous action spaces	1. Sensitive to hyperparameters	High	High
Advantage Actor-Critic (A2C)	1. Good for large-scale problems 2. Parallel training improves learning speed	1. Hard to tune and stabilize 2. It requires two networks to be trained simultaneously, which can be computationally expensive	High	High
Multi-Agent Advantage Actor-Critic (MA2C)	1. Scalability to Multi-Agent Settings 2. Stabilized Training By using the advantage function, MA2C reduces the variance in policy gradient estimates, leading to more stable and efficient learning compared to methods that rely purely on raw rewards. 3. Continuous and Discrete Action Spaces	1. Complexity in Training. Training multiple agents simultaneously can be computationally expensive and complex. The interactions between agents can lead to non-stationarity, making the learning process more challenging. 2. In multi-agent settings, it can be difficult to attribute rewards to individual agents. 3. Performance is highly sensitive	Medium to high	Medium to high
Q-Learning	1. Q-learning does not require a model of the environment, meaning it can learn optimal policies directly from interactions with the environment. 2. The algorithm is relatively simple to understand and implement compared to other reinforcement learning techniques.	1. Q-learning can struggle in environments with sparse or delayed rewards, as it might take a long time to propagate the reward information back to relevant states and actions. 2. The algorithm is relatively simple to understand and implement compared to other reinforcement learning techniques. 3. Time-consuming process.	Low	Low

We have conducted a basic comparison of various algorithms, and the advantages and disadvantages are illustrated in the figure. While simple algorithms such as DQN and Q-Learning are easier to implement, they lack performance and scalability compared to more advanced methods. On the other hand, algorithms like DDPG, A2C, and MA2C offer higher accuracy and are suitable for complex environments; however, they also increase the sensitivity of computing resources and require hyperparameter adjustment.

There is still much important work to be done in the future for the practical deployment of the proposed MARL algorithm. These measures include:

1. improving the realism of traffic simulators to provide training data on real-world traffic demand;
2. improve the robustness and tolerance of algorithms for road delay state measurement and sensor noise;
3. following practical engineering assumptions and combining appropriate learning and communication methods to enhance the superiority of existing MARL algorithms.

Acknowledgement. Zihua Ding, Yanbin Hou and Yunfan Zhang contributed equally to this work and should be considered co-first authors.

References

[1] T. Chu, S. Chinchali, and S. Katti, “Multi-agent Reinforcement Learning for Networked System Control,” arXiv:2004.01339 [cs, stat], Apr. 2020, Accessed: May 08, 2023. [Online]. Available: https://arxiv.org/abs/2004.01339

[2] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Başar, “Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents,” arXiv.org, 2018. https://arxiv.org/abs/1802.08757

[3] Li, Peng, Ding, Xiangcheng, Sun, Hongfang, Zhao, Shiquan, Cajo, Ricardo, Research on Dynamic Path Planning of Mobile Robot Based on Improved DDPG Algorithm, Mobile Information Systems, 2021, 5169460, 10 pages, 2021. https://doi.org/10.1155/2021/5169460

[4] F. Rasheed, K. -L. A. Yau, R. M. Noor, C. Wu and Y. -C. Low, "Deep Reinforcement Learning for Traffic Signal Control: A Review," in IEEE Access, vol. 8, pp. 208016-208044, 2020

[5] A. Bull 2004 Traffic Congestion - The Problem and How to Deal with it?

[6] Chaudhuri, H., Masti, V., Veerendranath, V., Natarajan, S. (2022). A Comparative Study of Algorithms for Intelligent Traffic Signal Control. In: Chen, J.IZ., Wang, H., Du, KL., Suma, V. (eds) Machine Learning and Autonomous Systems. Smart Innovation, Systems and Technologies, vol 269. Springer, Singapore. https://doi.org/10.1007/978-981-16-7996-4_19

[7] F. Rasheed, K. -L. A. Yau, R. M. Noor, C. Wu and Y. -C. Low, "Deep Reinforcement Learning for Traffic Signal Control: A Review," in IEEE Access, vol. 8, pp. 208016-208044, 2020

[8] G. Zhang, F. Chang, J. Jin, F. Yang, and H. Huang, “Multi-objective deep reinforcement learning approach for adaptive traffic signal control system with concurrent optimization of safety, efficiency, and decarbonization at intersections,” Accident analysis and prevention, vol. 199, pp. 107451–107451, May 2024, doi: https://doi.org/10.1016/j.aap.2023.107451.

[9] Tianshu Chu, Jie Wang, Lara Codecà, and Zhaojian Li, Multi-Agent Deep Reinforcement Learning for Large-scale Traffic Signal Control 2019, arXiv:1903.04527
[10] Josh Achiam 2020 Deep Deterministic Policy Gradient — Spinning Up documentation (openai.com)
[11] A. Haydari and Y. Yılmaz, "Deep Reinforcement Learning for Intelligent Transportation Systems: A Survey," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 1, pp. 11-32, Jan. 2022, doi: 10.1109/TITS.2020.3008612.
[12] P. Reyad and T. Sayed, “Real-Time multi-objective optimization of safety and mobility at signalized intersections,” Transportmetrica B: Transport Dynamics, pp. 1–22, Nov. 2022, doi: https://doi.org/10.1080/21680566.2022.2141911.
[13] S. S. Mousavi, M. Schukat, and E. Howley, “Traffic light control using deep policy-gradient and value-function-based reinforcement learning,” IET Intelligent Transport Systems, vol. 11, no. 7, pp. 417–423, Sep. 2017, doi: https://doi.org/10.1049/iet-its.2017.0153.
[14] Nada Faqir, Chakir Loqman, Jaouad Boumhidi, Combined extreme learning machine and max pressure algorithms for traffic signal control, Intelligent Systems with Applications, Volume 19,2023,200255,ISSN 2667-3053,https://doi.org/10.1016/j.iswa.2023.200255.
[15] Wade Genders, Saiedeh Razavi, 2019, An Open-Source Framework for Adaptive Traffic Signal Control Wade Genders and Saiedeh Razavi, arXiv:1909.00395
[16] Tianshu Chu, 2020, Multi-agent Reinforcement Learning for Network System Control