Transportation Research Part C 125 (2021) 103046

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Adaptive Traffic Signal Control for large-scale scenario with Cooperative Group-based Multi-agent reinforcement learning

Tong Wang, Jiahua Cao, Azhar Hussain *

College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China

ARTICLE INFO

Keywords:
Multi-agent reinforcement learning
Adaptive traffic signal control
Regional green wave control
CVIS

ABSTRACT

Recent research reveals that reinforcement learning can potentially perform optimal decision-making compared to traditional methods like Adaptive Traffic Signal Control (ATSC). With the development of knowledge through trial and error, the Deep Reinforcement Learning (DRL) technique shows its feasibility for the intelligent traffic lights control. However, the general DRL algorithms cannot meet the demands of agents for coordination within large complex road networks. In this article, we introduce a new Cooperative Group-Based Multi-Agent reinforcement learning-ATSC (CGB-MATSC) framework. It is based on Cooperative Vehicle Infrastructure System (CVIS) to realize effective control in the large-scale road network. We propose a CGB-MAQL algorithm that applies k-nearest-neighbor-based state representation, pheromone-based regional green-wave control mode, and spatial discounted reward to stabilize the learning convergence. Extensive experiments and ablation studies of the CGB-MAQL algorithm show its effectiveness and scalability in the synthetic road network, Monaco city and Harbin city scenarios. Results demonstrate that compared with a set of general control methods, our algorithm can better control multiple intersection cases on congestion alleviation and environmental protection.

1. Introduction

In recent years, with the increasing population and urgent needs for transportation efficiency, the demands for traveling have raised severe congestion problems, especially in urban areas. Solutions to congestion alleviation, such as optimizing road network construction and improving urban fundamental managing equipment, are proposed to handle the pressure caused by large traffic flow during rush hours. Among all possible methods, the combination of intelligence in traffic light control systems has proved to be both economical and efficient in relieving flow pressure at congested intersections. With the development of deep learning techniques, strategies for Adaptive Traffic Signal Control (ATSC) have shown great potential in integrating state-of-art intelligent methods. It includes automatic data feature mining and traffic flow mode learning, which alleviate uneven resource deployment problems at intersections compared to traditional systems like SCATS and SCOOT (Hunt et al., 1982).

The focus of this paper is to control the traffic lights by imitating an expert human-level intelligence. Existing traffic light control systems either deploy fixed programs without incorporating the real-time traffic or consider the traffic to a minimal degree (Cases, 2017). The selected programs either set the traffic signals at equal time duration in each cycle or vary the time duration depending on the historical information. Some of them take input information from the sensors underground, such as inductive loop detectors, to obtain vehicles near the traffic lights. However, such data is processed in a very coarse way to find the duration of green and red lights. Existing traffic light control systems work in some cases (at low efficiency); however, in events like sports, festivals, weather

* Corresponding author.
E-mail addresses: wangtong@hrbeu.edu.cn (T. Wang), caojiahua@hrbeu.edu.cn (J. Cao), engrazr@hrbeu.edu.cn (A. Hussain).

https://doi.org/10.1016/j.trc.2021.103046
Received 28 July 2020; Received in revised form 25 December 2020; Accepted 12 February 2021
Available online 22 February 2021

conditions, or a more typical high traffic hour scenario, the existing control systems become paralyzed. It is reported in the recent research work (Liang et al., 2019) that an experienced police officer can efficiently and directly manage the intersection by waving signals. Especially in high traffic scenarios, a human operator observes the real-time traffic condition in the intersection roads and determines the duration of allowed passing time smartly. Each direction uses his/her long-term experience and understanding of the intersection, and this approach is efficient. The proposed method's focus is motivated by the human operator's intelligence, which can take real-time input and learn how to manage the intersection. Such implementation requires eyes to watch the road conditions and a brain to process this information. For the eyes part, we employ V2X knowledge, and for the brain, we use deep reinforcement learning to solve this problem.

However, in the existing ATSC methods, the main focus is to vary the duration of green or red cycles based on real-time information. Such an approach has proven to be less efficient as described in Liang et al. (2019), and the main reason is neglecting the impact of the current set duration on the future upcoming traffic. In a fair comparison, we refer to Pandit et al. (2013), which presents comparison of existing ATSC (e.g., Varaiya, 2013), with various deep reinforcement learning techniques. The ATSC can achieve better results than the traditional fixed-duration method, but worse than the others because the optimal phases' time duration in the most recent cycles does not perform better in the future traffic considering the traffic scenarios which become very complicated.

With underlying features of dynamic traffic flow learned by the Deep Learning (DL) methods (serving as input for intersection controllers), the optimal design of basic controlling indexes such as green phase duration and phase sequence of the traffic light at an intersection can be seen as a procedure of optimal decisions making. Thus, reinforcement learning (RL) is applied to signal control to get the best control policy for a signalized intersection. To achieve immediate data collection and transmission with low latency, a Cooperative Vehicle Infrastructure System (CVIS) is available for supporting communication among vehicles and roadside infrastructure via vehicle-to-vehicle (V2V) and vehicle-to-roadside units (V2X). This communication paradigm is known as vehicle-to-everything (V2X). Compared with traditional sensing detectors, V2X can collect more accurate flow data to achieve comprehensive information for traffic control, leading to better RL decision making. With the RL method applied to tackle practical problems, Q-table-based learning was unable to handle a high-dimensional state in complex problems. Thus, DL was applied to strengthen an RL-based agent's comprehensive ability by replacing Q-table with Q-value networks. DQN, proposed by Deepmind in 2013 (Mnih et al., 2015), is the most typical value-based learning model of deep reinforcement learning (DRL). The major theories behind DRL consist of three categories, in which value-based RL focuses on the approximation of value function, such as Q-learning (Wiering et al., 2004) and SARSA algorithms (Thorpe, 1997). Policy-based RL (Peters and Schaal, 2008; Sutton et al., 2000) techniques are focused on the learning of parameterized policy. The above two methods assume that agents are learning by interacting with the real environment.

In Nicols and Carlos (2013), Nagabandi et al. (2017), the authors proposed a novel Model-based RL approach to model the reality with the prediction of reward and state transmission. Actor–critic (Mnih et al., 2016) method is a unique algorithm that combines value function approximation and policy-based method. With the actor choosing actions via the explore–exploit policy executed, the critic will evaluate the actor's current behavior with its immediate reward to guide the actor's next step. Continuous optimization techniques such as experience replay, target Q-network, and advantage value function generally apply to improve stability and faster convergence on a single agent case. The Multi-Agent Reinforcement Learning (MARL) system for a multi-agent system (MAS) has become a key research area (Grover et al., 2018). The impact of the agent's behavior on the environment is hard to predict in advance. Therefore, the literature emphasizes the realization of coordination among the agents to achieve better performance.

In preceding studies, green wave control on arterial roads (Ye et al., 2014) and agent-based DRL control on single intersection have proved their efficiency in congestion alleviation problems from different perspectives. However, it is still a complex problem to extend the DRL agent-based method directly to large-scale networks, which is quite time-consuming. It may lead to a significant requirement for computation resources and handling the Curse of Dimensionality (Friedman, 1997). To extend the DRL agent-based method to large-scale network control with reasonable computation cost, we propose a novel DRL-based framework. It combines DRL and pheromone-based green wave control with the support of CVIS. It is a hierarchical communication configuration comprising the Mobile Edge Computing (MEC) server with a fixed number of Road Side Units (RSUs), each agent covering about hundreds of meters to make the group-based regional agents. Compared to most cases where two agents achieve cooperation (with massive repetitive exchange of messages), CVIS extends the definition of coordination among multiple agents from a distributed point of view, making high-level coordination in MARL.

The key points of significant contributions to our study are listed below:

1. We devise a framework named CGB-MATSC with a dynamic group of traffic lights working under a regional agent to promote extensions of MARL algorithm, eliminating differences in state dimensions caused by various intersections when taking single intersection as an agent. It also slows down the expansion in the training data dimensions.
2. To work out an effective control strategy with great portability, we are the first to combine RL and pheromone-based regional green-wave control mode for ATSC by including it as one of the choices of action. After limited training episodes, the proposed CGB-MAQL shows superior scalability when migrated from a trained scene to a large-scale network scenarios.
3. We proposed k-nearest neighbor state representation and spatial factor-based discounted joint reward mode to boost coordination among agents in the CGB-MAQL algorithm. It shows excellent superiority on all evaluating metrics over CGB-IQL, which incorporates no coordination at all in ATSC.

This paper is structured as follows: Section 2 reports the related works on the MARL applied for the ATSC. Section 3 presents the background on deep reinforcement learning. Section 4 presents the CVIS-based three-layer hierarchical communication system and gives a detailed description of the proposed group-based CGB-MATSC framework. Section 5 presents results and analysis, and Appendix section is given at the end of the paper, and Section 6 concludes the article with future direction.

Table 1
Overview of various MARL-based ATSC methods.

Methods	Q-value representation	State	Action	Reward	Coordination
(Ge et al., 2019) QT-CDQN	Function approximation	Discrete traffic state encoding	Phase	Change of queue length	Q-value transfer
(Bakker et al., 2010) TC-SBC	Tabular Q-value	Congestion level ranged in [0, 1]	A subset of traffic lights to set green	Number of moving cars	CG based Max-plus
(Bakker et al., 2010) TC-GAC	Tabular Q-value	Sensed vehicle position	A subset of traffic lights to set green	Congestion factor based reward	CG based Max-plus
(Chu et al., 2020) MA2C	Function approximation	Delay of first car and total of approaching cars	Phase	Change of waiting time and queue length	Parameterized neighboring policy
(Jin and Ma, 2019) CMDP	Function approximation	Traffic flow	Change of green duration	Difference of throughput	No coordination

2. Related works

The research and development of the methods applied to ATSC summarize the evolution of traditional techniques, such as SCATS, SCOOT systems, and more intelligent technologies. A survey (Wang et al., 2019) shows how the revolution of traditional machine learning techniques intellectualizes the landscape of the Intelligent Transportation System (ITS). The classical methods for traffic signal control characterize as fixed periodic preset model-based control, which shows limited adaptiveness for the semi-stationary traffic flow situation. The research on combining state-of-the-art machine learning knowledge, such as DL and RL into the ATSC problem is emerging. The proposal of reinforced learning firstly shows its success over the human level on video games. Besides, it also provides a new perspective to solve Markov Decision Processes (MDP) problem. Most of the early works of RL on ATSC have proved their potential with iteratively updating the Bellman equation in a dynamic programming way, as well as the Monte Carlo method based on the complete sampling sequence. The application of the time series difference method based on the Q-table to store Q-value for action–state pairs has shown its ability to reduce road users' average waiting time and waiting for queue lengths in literature (Marco et al., 2004). Q-learning is applied to car-based agents voting for a traffic light decision with a lookup table as a model of state transition function. But above methods can only be used to small-scale problems with finite discrete state space, in contrast to the larger state space in complex problems driven by the Curse of Dimensionality. In 2013 (Mnih et al., 2015), the authors have proposed an RL framework that adapts to high-dimension model inputs. In that framework, the DRL with deep neural network applies to map original information to the corresponding value for optimum decision. In Qi et al. (2019), the authors have successfully incorporated the mechanism of RL technique into real-time optimum decision making in an energy management system.

Table 1 provides overview of various MARL-based ATSC methods. Since it takes significant time for an RL-based agent to learn from ample experience, most general RL methods are only tested on single-intersection cases, which is not the actual case in real life. With the increasing demands for RL applied in a MAS, some MARL algorithms are designed to find an effective and practical way in MAS. MADDPG (Lowe, 2017), proposed by Open AI in 2017, has developed a centralized critic (aware of information from all actors) to guide their distributed actions, which to some extent, release the stress on the unsuitability of experience replay as well as significant variance during the update. In Rashid et al. (2018), the QMIX algorithm (allowing local value function of each agent) is integrated to make joint action-value function via a deep network. Agents learn in a centralized way, while they take actions in a distributed manner. It shows a novel approach to extend single-agent RL to large-scale scenarios. For RL application with multiple intersections, several extensions of fundamental single-agent RL have been proposed from the perspective of a combination of Q-learning (QL) and Nash equilibrium and variants of DQN, aiming at improving convergence and effectiveness of all agents. However, the solution based on Nash equilibrium has always been a problem of equilibrium selection when the optimum strategies obtained by different agents are not unique. In Qu et al. (2020), the authors proposed a regional mixed strategy nash-equilibrium based multi-agent reinforcement learning method to perform distributed control for urban networks. Since the introduction of Natural DQN in 2015 (Mnih1 et al., 2015), various DQN-based variants have proved their effectiveness, and reinforcement learning has provided new ideas for solving MAS problems. Based on the stochastic game framework of Markov's decision, a series of MARL algorithms have been continuously improving in terms of agent collaboration, reward distribution, and algorithm efficiency.

One of the MARL applications in MAS is a centralized learning method that regards the entire system as a centralized agent. The research in Tan et al. (2019a) has proved to be sufficient to reduce congestion with a new regional agent-based framework named CODER, with the assumption that global Q-value can be decomposed into several local Q-value. The global Q-values are updated in a centralized way. However, centralized systems face problems such as low efficiency. Distributed learning methods have been improved in terms of robustness and scalability. Unlike the above two centralized methods, a set of decentralized methods are proposed in the practical viewpoint, such as the NAQL algorithm mentioned in Tan et al. (2018). It uses a decentralized system to avoid an exponential increase of state–action space size caused by the number of agents in large-scale scenarios. On the other hand, it boosts communication and cooperation among neighboring agents inside a single region.

From the perspective of awareness, adaptivity, and communication ability of agents, Rezzai et al. (2018) gives a general control architecture, offering a view on recent studies about the application of Temporal Difference (TD) methods in traffic control problem. The most common way to reduce the learning burden of collaborative agents is to use the discrete state method. It Vidhate and Kulkarni (2017) takes real-time traffic flow features as discrete states, of which transition delayed return is recorded to strengthen the

Table 2
Description of variables used in the configuration.

Parameter	Description	Parameter	Description
$agent_i$	An ith MEC server	$N_{w,i}$	Number of stopped vehicles under the region of $agent_i$
g	A traffic light intersection	a^i	Represents the index of previous action for $agent_i$
G_i	Set of all traffic lights under $agent_i$	x_i	Discrete level of congestion $\in \{0,1,2\}$ for $agent_i$
g_1, g_2	Indices of the two most congested traffic light intersections	c_1, c_2	Respective congestion levels formulated similarly as x_i
δ_0, δ_1	Thresholds for discrete state representation	s_i	Represents local state of $agent_i$
N_i	List of k nearest neighbors of $agent_i$	S_i	k-nearest-neighbor-based joint state of $agent_i$
$N^g_{i,t}$	Vehicles under traffic light g of $agent_i$	ρ^g_t	Pheromone belonging to the controlled edge of traffic light g at time t
$D_{green,t}$	A dictionary of green phases to set at time step t	$r_{i,local}$	Local reward of $agent_i$
T_e	Ending steps of an episode	$r_{i,joint}$	Spacial reward of $agent_i$
seq	A phase sequence	$\lvert d_{ij}\rvert$	Distance between $agent_i$ and $agent_j$
N^g_{lane}	The number of controlled lanes under traffic light g	f^i_t	Average flow of vehicles at time t in the region of an $agent_i$
v^g_t	Average speed of vehicles at time step t under the traffic light g	N_{G_i}	Total traffic lights under the region of $agent_i$
L_i	A set of edges controlled by each g under $agent_i$	Ph_i	A set of preset phase sequence dictionaries under $agent_i$
$D_{Ph,t}$	A preset phase sequence dictionary	$D_{i,t}$	A dictionary containing number of vehicles under $agent_i$ at time step t
$l^{g_i}_j$	A jth edge of ith traffic light	$D_{\rho,t}$	A pheromone dictionary at time step t
$D_{c,t}$	An edge congestion dictionary at time step t	$D_{l^g_{max},t}$	A dictionary containing edges with maximum pheromone at time step t

relationship among a control of agents and real-time improvement on waiting time and number of stopped cars. Three coordination models are applied to compare in a multi-agent framework.

The realization of cooperation among agents in MAS has always been a difficult problem. From the perspective of collaboration among agents, the degree of collaboration determines the entire system's performance to a large extent. The easiest way to extend to the MARL case is Independent Q Learning (IQL) (Schneider et al., 1999), with no coordination among intersection agents. To stabilize the learning process, with action and discretized state space, the authors in K.J.1 et al. (2015) take consideration of the adjacent neighbor's local impact through a feedback signal incorporated in the update of local Q-function, instead of taking them as part of environmental dynamics. Under limited communication, the coordination can also be achieved through messages exchanged via networks like Vehicular Ad-hoc NETwork (VANET). In Chu et al. (2020), cooperation mechanisms can assist independent RL methods in improving MARL via indirect communication and direct communication. VANET is mentioned to incorporate neighboring policies in the local policy update. The adaptivity of the practical application of VANET in city traffic is evaluated in Liu et al. (2012).

With communication available, deeper cooperation can be achieved via Q-value decomposition based on the graph method. In Kok and Vlassis (2005), a coordination graph (CG) based framework to support the exchange of payoff information between connected entities are proposed to access greedy joint actions in MAS, which outperforms variable elimination algorithms in both to solve quality and convergence efficiency. Bakker et al. (2010) has applied a coordination graph-based max-plus algorithm in a MARL traffic signal control system as a coordination method to achieve further cooperation among traffic lights in large networks, whose results show flexibility and scalability of the algorithm. Although max-plus can finally converge the overall agent's action space in the local region of the agent via message exchange, it seldom works on cyclic graphs. By decomposing global Q-value into a set of local values with two arguments incorporated in sparse RL, a clique-based cooperative MARL (Zhang and Zhao, 2014) is introduced that makes local Q-value function available to relate with more than two cases. Thus the message transmitting visualized by CG between adjacent agents has changed to message exchange among more agents with a max-plus algorithm to get greedy joint action.

3. Background on deep reinforcement learning

In this article, we extend the single-agent RL method to MARL in two ways: one is a direct extension method of general IQL, and the other uses the Advantage Actor–Critic (A2C) principle. The coordination among agents is an MDP; hence we will investigate the collaboration among agents of both algorithms. We describe the principle of agents' learning behind DQN and A2C and how the level of cooperation between agents affects agents' learning process. MARL can achieve intelligent collaboration among MEC servers (or agents). In the following subsections we present the fundamental RL algorithms concisely, and introduce ATSC application scenarios, such as problem assumptions, optimization goals, and decision variables.

3.1. Problem assumptions

Fig. 1 shows the illustration of an assumed ATSC application scenario for simplicity and brevity. A six by six road traffic grid is shown on the left, which has three MEC servers, each connected with several RSUs. Each MEC is responsible for the control of all the

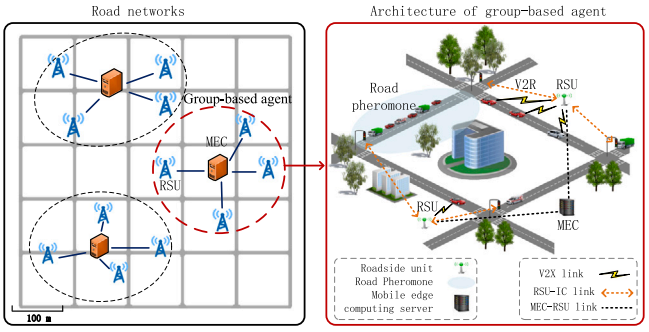

Fig. 1. The deployment of MEC–RSU–Vehicle based hierarchical communication architecture in large-scale ATSC application scenario.

traffic light intersections under its circular region. The road pheromone information (as shown on the right side) is received by the V2X communication links using vehicle-to-vehicle V2V and vehicle-to-roadside units V2R messages that are sent to the RSUs from the vehicles. The MEC–RSU link will provide the road pheromone information from the RSU to the MEC. Based on this information, the MEC will perform an algorithm and decide the optimal phase duration settings for its respective traffic lights. These settings will be sent back to the RSU using the MEC–RSU link. The RSU-intersection control link is shown by RSU-IC that provides the control commands (of phase duration settings) from RSU to the traffic lights as shown in Fig. 1. The MEC acts as a learning agent to perform its algorithm in discrete time steps.

At each time step t, an agent takes an action a_t (it can be the best phase duration of green or red traffic light signals) on the environment based on its policy $\pi(a_t|s_t)$, where s_t is the current observed state from environment. The agent then receives a reward (it can be the average flow of vehicles) r_{t+1} and a next observation state s_{t+1} from the environment. The agent's goal is to learn a policy $\pi(a_t|s_t)$ to maximize its sum of returns in an episode with T steps. In an ATSC perspective, an agent observes state s_t as input (it can be the real-time information about vehicles) and interacts with the environment to maximize traffic flow through optimizing behavior policy. RL's cornerstone is that the environment is assumed to be Markovian, and the RL is an MDP model. Setting signals at intersections to minimize the queue length and vehicle delay time is key to traffic management. Several research works (Yu and Stubberud, 1997) indicated that: to apply Markovian control to traffic systems a state-space and a probability measure must be defined. A threshold that is commonly the number of vehicles is chosen for each movement's queue at an intersection.

If the queue length is greater than the threshold value, this movement is defined as in its congestion mode; otherwise, it is in the non-congestion mode. These two modes, i.e., congestion or non-congestion, are defined as two states in the state space. The signal phasing can be considered as different alternatives in each state. We can have a simple example which assumes the traffic flow moves only in two directions; let us say north/south (as direction A) or east/west (As direction B) of a four branch intersection. There are four possible situations: (1) both directions are non-congested, (2) direction A is congested, but direction B is non-congested, (3) vice versa of situation 2, and (4) both directions are congested. These four different situations can be defined as the four states of the Markov process. Furthermore, if there are eight independent movements under 8-phase signal control, the traffic control problem can be formatted as a 256 state Markov process with eight alternatives in each state. In another research work, an approach to real-time control of a network of signalized intersections is proposed based on a discrete-time, stationary, Markov control model (Recker et al., 1995). This approach incorporates microscopic simulation of actuated controller output signals in response to probabilistic forecasts of individual vehicle actuation at downstream inductance loop detectors derived from a macroscopic link transfer function. In a research work (Zhao and Spall, 2018), a novel Markovian framework is presented for modeling dynamic network traffic. By relaxing the stationary assumption and allowing vehicles to enter and exit the network, this Markov-based dynamic network traffic model is more intuitive and realistic compared to previous studies (such as Crisostomi et al., 2011; Moosavi and Hovestadt, 2013; Schlote, 2014). Moreover, by applying the full-system/subsystem scheme to model the traffic network, they integrate information from link and route levels to estimate travel times for links. Furthermore, they also use the real-time traffic data resources, i.e., Google Maps, to populate the network model, providing a novel approach to use these new data sources to model the dynamic network traffic. Research work (Osorio and Wang, 2017), published in Transportation Research Part B, is motivated by recent results in designing signal plans for Manhattan that highlight the importance of providing signal control algorithms with an analytical description of between-link dependencies. Their paper models an urban road network as a finite space capacity Markovian queueing network. The model consists of a system of nonlinear equations with a linear dimension, instead of exponential, in the number of queues and that is independent of the space capacity of the individual queues. They used it to address an urban traffic control problem. They demonstrated the added value of accounting for higher-order spatial between-queue dependency information to control congested urban networks. Motivated by these recent research works we assume the problem of minimizing vehicle delay at intersections can be formulated as a Markov Decision Process (MDP).

3.2. Optimization goal and decision variables

Let $N_{w,l,i,t}$ be the number of waiting vehicles for an MEC server i on an lth edge at time step t. E is total number of evaluations and T_e represents the ending step of episode e, and t represents each time step. $Edges_i$ contains information of all edges under

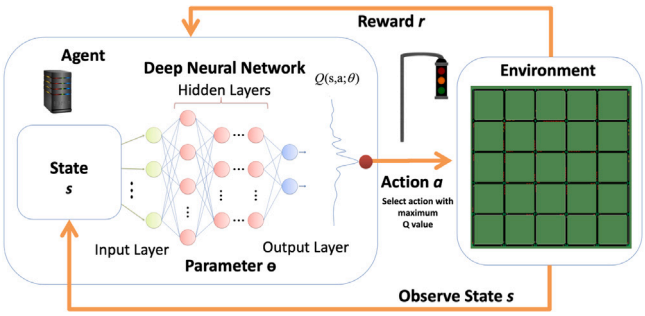

Fig. 2. An illustration of basic DQN learning scenario.

agent i. The final optimization goal is to minimize the average waiting time of all vehicles under N MEC agents, and given by:

$$minimize: \frac{\sum_{i=1}^{N}\sum_{e=1}^{E}\sum_{t=1}^{T_e}\sum_{l\in Edges_i} N_{w,l,i,t}}{N\times E\times T_e}. \tag{1}$$

Table 2 provides description of decision variables used in this article. We will concisely introduce the fundamental RL algorithms in the following subsections.

3.3. Q-learning

Q-Learning is a classical reinforcement learning algorithm based on the formulation of a Q-function, also known as state–action value function $Q^{\pi}(s,a)$ of a policy π. It determines the discounted sum of rewards obtained by taking action a in state s, and gradually following the optimal policy. The optimal Q-function obeys the Bellman optimality equation:

$$Q^{*}(s,a)=\mathbb{E}[r+\gamma \max_{a'} Q^{*}(s',a')], \tag{2}$$

where $\gamma \in [0,1)$ is a discount factor, that discounts the future rewards r relative to the immediate rewards. An iterative update $Q_{t+1}(s,a)\leftarrow \mathbb{E}[r+\gamma \max_{a'} Q_t(s',a')]$ leads towards convergence which means $Q_t \rightarrow Q^{*}$, as time $t\rightarrow\infty$.

3.4. Deep Q-learning

In general, an RL-based problem can be solved by following two steps: policy evaluation and policy iteration, to respectively access stable value function and optimum policy (Schulman et al., 2015). Q-learning (QL) is an off-line control mode of time series difference algorithm. It takes one behavior strategy, such as the ϵ-greedy strategy (Hausknecht and Stone, 2015), to select the current action while uses different policies like the greedy strategy to update the value function (Tsitsiklis and Van Roy, 1997). The update of the value function is:

$$Q(s,a)=Q(s,a)+\alpha(r+\gamma \max_{a} Q(s',a')-Q(s,a)), \tag{3}$$

where the (s,a) and (s',a') represent current and next state–action pair, respectively. α is the learning rate of agent. The r is an immediate reward, and γ is a discounted factor. The formula shows that the QL-based agent learns the optimum policy directly by corresponding to maximum Q-value. To solve problems with consecutive state space, DQN has improved the representation of value function in QL with a neural network to achieve function approximation method (Mnih1 et al., 2015). Fig. 2 gives the brief configuration of a general DQN network used to control traffic lights.

The input of the above deep neural network is the state characteristics of the environment, and it will output all corresponding value function of action–state pairs, which are represented as $Q(s,a,\theta)$. The term θ is a parameter of the network. There are two significant techniques applied to optimize the convergence:

(1) Experience replay helps eliminate the correlation among sampled experience.

(2) Target Q-network configuration, where the current Q network parameter will update and be copied to the target network periodically (Mnih1 et al., 2015), as:

$$y_j=\begin{cases} r_j & ;done\ is\ true \\ r_j+\gamma \max_a Q'(s_j,a_j,\theta) & ;done\ is\ false, \end{cases} \tag{4}$$

where y_j represents the target Q-value. While the agent interacts with the environment, the state transmission pair (s,s') and action executed as well as the immediate reward will be recorded as $(s,a,s',r,done)$, where $done$ tells whether the current episode ends or not. The target network will calculate a target Q-value for training the current Q network with sampled experience. Every fixed period, the target Q-value y_j will be used to update the parameters of the existing network. The current network will copy the

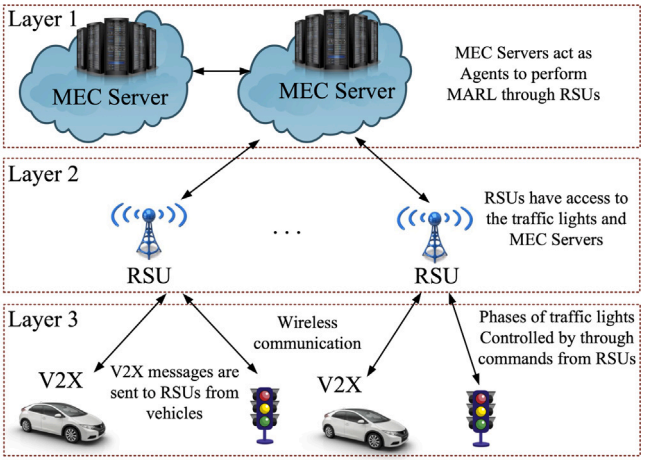

Fig. 3. Three-layer hierarchical MEC–RSU–Vehicle communication configuration.

updated parameters to the target network. The parameter update of the current network uses the Mean Square Error (MSE) as a loss function to update the Q-network parameters through backpropagation as:

$$loss = \frac{1}{m}\sum_{j=1}^{m}(y_j - Q(s_j, A_j, \theta)). \tag{5}$$

The purpose of backpropagation is to minimize the $loss$ and thus stabilize the agent's training process, also the experience replay is used to reduce the correlation between the target Q-value and the current Q-value.

3.5. Advantage actor–critic

The above DQN method focused on learning via Q-value function, which proves to do not apply to random strategy problems. The optimal strategy solved by the value-based way is deterministic. To improve this, in policy-gradient-based RL methods (Sutton et al., 1999), neural networks are used to represent parameterized strategy. By continuously updating the strategy parameters via neural networks training, the optimal strategy will finally be learned. The Actor–Critic (AC) algorithm is a combination of value-based and policy-gradient-based methods, where agent can also be called actor, which is responsible for actions generation (policy $\pi(s)$: which gives probabilities of all actions for a given state s) and execution (Mnih et al., 2016).

The role critic plays is equivalent to the value function (V(s): estimate of the value of a state a) part in DQN. It is responsible for the actor's performance evaluation and will also guide the action selection in the next state according to the environment's feedback. Actor and critic are each represented by a network. With θ represents the parameter of actor-network, the approximation of the strategy function can be expressed as:

$$\pi_\theta(s, a) = P(a|s, \theta). \tag{6}$$

When we select the advantage function as the evaluation point of the AC algorithm, we define it by representing the relative advantage of action value function over the value function for the current state:

$$A_{\pi_\theta}(s, a) = Q_{\pi_\theta}(s, a) - V_{\pi_\theta}(s). \tag{7}$$

After expressing the strategy as a continuous function with parameter θ, we can use the constant function optimization to find the optimal strategy. The gradient ascent method is used to find the optimal gradient. Here we define the optimized function objectively as $\bigtriangledown_\theta J(\theta)$. Thus the strategy parameter can be updated as:

$$\bigtriangledown_\theta J(\theta) = E_{\pi_\theta}[\bigtriangledown_\theta log\pi_\theta(s, a)A_{\pi_\theta}(s, a)]. \tag{8}$$

$$\theta_{t+1} = \theta_t + \alpha A_{\pi_\theta}(s, a)\bigtriangledown_\theta log\pi_\theta(s, a). \tag{9}$$

3.6. Extension of RL to multi-agent RL

The algorithm that applies RL to MAS is called MARL. According to the degree of collaboration among agents, MARL can be divided into three levels: (1) completely independent MARL, (2) partially cooperative MARL, and (3) fully cooperative MARL (Bakker et al., 2010). IQL is a typical algorithm of completely independent MARL. The idea is to train every single agent directly without any cooperation among agents. The disadvantage of this approach is that the behavioral impact of other agents is completely regarded as

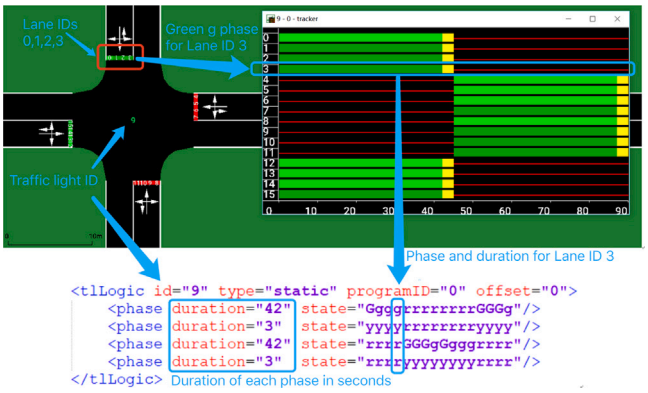

Fig. 4. A four branch traffic light intersection with its underlying phase sequence details. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

part of the environment. The dynamic randomness of other agents' strategies is not considered during the learning of a local agent. The partially coordinated MARL can be achieved by sharing state or introducing heuristic action decisions. In the local agent's observation of the environment, part of the neighbor agent's state is introduced to evaluate each action's value according to the state of the cooperation. The fully cooperative MARL solves the optimal joint action through the joint state.

We will discuss the three significant improvements mentioned above to stabilize the collaboration process among multiple agents in the MARL-ATSC algorithm. The state expression based on k-nearest neighbors makes the state transition of the environment more relevant to the characteristics of the agent's behavior, which significantly reduces the dimensions for the models' learning process. Based on ATSC's time and space characteristics, the distance-based discounted local return considers the agent's and neighbors' distances into the local reward's impact. This method achieves a trade-off between self-rewarding ways, which are prone to create selfish agents and bisected global reward methods that quickly lead to lazy agents (Riedmiller et al., 2018). Besides, the pheromone-based green wave control mode will balance road congestion with the road capacity. We adopt the above three significant optimizations to make the environmental information expression map with each agent in a more appropriate way.

4. CGB-MAQL: Cooperative group-based MAQL algorithm for ATSC problem

So far, we have reviewed the basic DQN and A2C techniques. With state space, action space, and system feedback clearly defined, the DRL algorithm can be applied to solve the optimal decision-making problems. However, when we try to extend general RL algorithms to MAS problems, such as ATSC for large road networks, we are faced with increasing complexity caused by vast joint state and action spaces and un-predefined cooperation. To design an applicable cooperative MARL algorithm for ATSC in the large-scale road network, CGB-MAQL, collaborative group-based multi-agent Q-learning algorithms for ATSC are proposed. Three techniques are applied to make the learning process of agent stable and convergent. These include state representation based on distance relationship among agents, regional green-wave action design based on pheromone, and local return design based on k-nearest-neighbors. In this paper, the acquisition of state, reward, and traffic lights' preset information are provided by a CVIS-based three-layer hierarchical communication architecture as shown in Fig. 3. Before proceeding to further details we introduce the concept of phase sequence and phase durations for a traffic light in general.

4.1. Phase sequences of a traffic light

To understand the concept of selectable phase indexes, we present a simple example for a traffic light (ID 9) with four edges as shown in Fig. 4. Each edge has four lanes. The ID of each lane counts from 0 to 15 (starting from top edge and going in clockwise direction). Annotations are added in blue to explain the phase and duration for each lane. For brevity we have marked Lane IDs 0, 1, 2, 3 in the top left corner of Fig. 4 with a red rectangle. The tracker window on the right hand side shows the phases (red light 'r', green light 'g', yellow light 'y', and lighter green light 'G') and their corresponding durations for all 16 lanes. The red light means vehicle must stop. The green light means vehicles may pass the junction if no vehicle uses a higher prioritized foe stream (e.g., ambulance), otherwise they decelerate for letting it pass. The lighter green light is for a signal priority i.e. vehicles may pass the junction. The yellow light means vehicles will start to decelerate if far away from the junction, otherwise they pass. For more details we refer the reader to Krajzewicz et al. (2012b). To further understand, we present the case of phase for the Lane ID 3. There are four phase sequences for this traffic light ID 9. The phase sequence $seq1$ is "GgggrrrrrrrrGGGg", phase sequence $seq2$ is "yyyyrrrrrrrryyyy", sequence $seq3$ is "rrrrGGGgGgggrrrr", and phase sequence $seq4$ is "rrrryyyyyyyyrrrr". The duration mentioned for each phase sequence as 42 s, 3 s, 42 s, and 3 s Each phase sequence represents the state in a form of (16 letters) string data structure, such as, in the first row we have state = "GgggrrrrrrrrGGGg". In this state, the index of phase (or string here) starts from 0 and ends at 15. The 0 corresponds to 'G' which is the phase with traffic light 'G' with duration 42 (s) for the Lane ID 0.

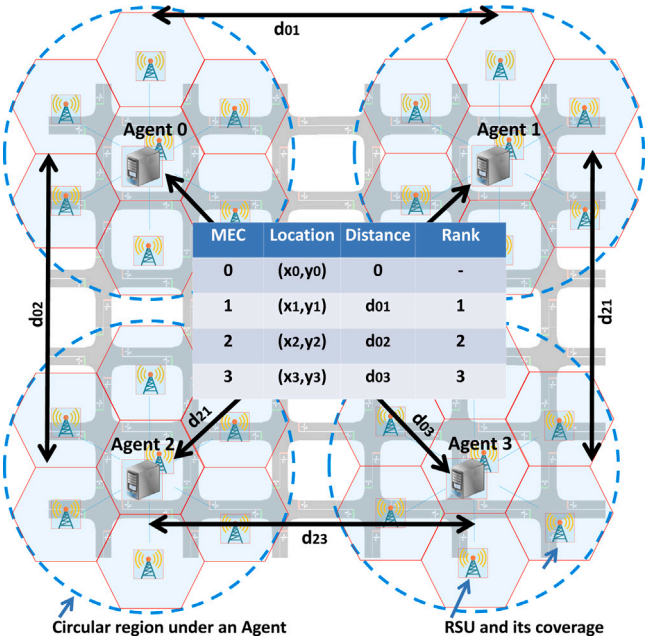

Fig. 5. An illustration of the proposed CGB-MATSC framework for large-scale ATSC problem consisting of multi-agents with their respective relative distances.

Similarly, if we take the case of Lane ID 3, the index is 3 in the state. Respectively, the 'g', 'y', 'r', and 'r' have durations as 42 s, 3 s, 42 s and 3 s for the Lane ID 3. The term list of selectable phase indexes refers to the possible choice of phases, because there are multiple combinations that are possible, which may lead to a choice where all lanes are set to green, which is intuitively not the choice to be made to avoid accidents and severe jam. The cycle of phases repeats according to phase durations, as mentioned in the state for the fixed traffic light control. However, in ATSC schemes the state and duration of the phases are altered based on the optimization algorithms. An edge may have any ID depending on the geographical street networks. The phase sequence information can be obtained from the specifications of the traffic light intersection in terms of its number of edges, lanes per edges, and the variety of the traffic light signals, as described in Krajzewicz et al. (2012b).

4.2. Proposed solution

To solve this problem, we design a joint return $r_{i,joint}$ based on the k-nearest neighbor algorithm, which incorporates global footprint sharing and a more realistic reward design.

4.2.1. CVIS-based three-layer hierarchical communication architecture

Fig. 3 shows the CVIS-based three-layer hierarchical communication architecture proposed in our work. In the bottom layer 3, each vehicle is equipped with an onboard unit that will inform the RSU of its essential information like vehicle identity, speed, and position through V2R communication in real-time. With the communication module embedded, each signal control facility can communicate with the RSU through a wireless or wired connection to report its position information, preset phase sequence information, and current phase. In the 2nd layer, RSU is supposed to summarize the necessary information sent by vehicles within its communication range. We assume that cars will not stop on the road if not congested. Vehicles with speed less than 0.05 m/s are counted as stopped. In layer 1, the MEC is connected with RSUs (layer 2) by the wired or wireless interface, and the MEC will calculate the global pheromone information. MEC communicates through RSUs: the information of busy roads, their mapped intersection ids, and the corresponding green wave phase index for all traffic lights.

4.2.2. CGB-MATSC: framework for combinations of RL algorithms for large-scale ATSC problem

Group-based agent: According to the hierarchical communication structure, we propose a group-based agent architecture, as shown in Fig. 1.

Environment: Our algorithm will be trained on a 6 × 6 network and tested in city scenarios, where various intersections such as two-lane, three-lane, and four-lane configurations are included. The acquisition of environmental information mainly depends on the preset phase information of intersection signal light, road length, RSU communication range setting, etc. For safety purposes, the time interval between two consecutive controls is 20 s. RSU can obtain the traffic pheromone information by comparing the ID information of the queue of vehicles. We will now present the description about state, action and reward formulations in the following section for each learning agent.

Algorithm 1: Green wave generation algorithm

Input : $\text{INFO}_{i,t}$ from RSUs of $agent_i$
Output: $D_{green,t}$ to RSUs for traffic lights

// Initialize time step
$t \leftarrow 0$
// Obtain set of traffic lights under an $agent_i$
$G_i \leftarrow \text{INFO}_{i,t}$
// Obtain set of controlled edges of each g traffic light
$L_i \leftarrow \text{INFO}_{i,t}$
// Obtain set of preset phases for each g
$Ph_i \leftarrow$ from $\text{INFO}_{i,t}$, G_i and L_i
repeat
  if $t\%\ \delta$=0 **then**
    repeat
      Update $N_{i,t}^{g} \leftarrow \text{INFO}_{i,t}$
      repeat
        //Get preset phase sequence dictionary
        $D_{Ph,t}[g][l] \leftarrow (start, end)$ from $\text{INFO}_{i,t}$
        Calculate ρ_t of each l using Eq. (12)
        Update $D_{\rho,t}[g]$ using Eqs. (19)-(20)
      until *Last edge l in L_i*;
      // Get the edge with maximum ρ_t of g
      $l_{\max}^{g} \leftarrow D_{c,t}[g]$ using Eq. (20)
      // Get $D_{l_{max}^{g},t}$ using $l_{\max}^{g}$ in $D_{Ph,t}$
      // Access phase sequences of g
      $seq_g \leftarrow D_{Ph,t}[g][l_{max}^{g}]$ using Eq. (21)
      // Iterate and check for all edges in $D_{l_{max}^{g},t}$
      if *seq contain no 'r' and 'y'* **then**
        $key = g$, $value = seq$
        $D_{c,t} \leftarrow$ (key, value) using Eq. (20)
      end
    until *Last g under G_i*;
    Set green phase of all g in the Dictionary
    $D_{green,t} \leftarrow D_{c,t}$ using Eq. (22)
  end
  $t \leftarrow t + 1$
until *Episode ends*;

4.2.3. State

The global footprint sharing is accomplished by a joint state using the relative distance relationship among agents. We represent agent's local state as $s_i = \{x_i, g_1, c_1, g_2, c_2, a^i\}$, where, $x_i \in \{0, 1, 2\}$ represents the congestion level to reflect the number of vehicles N_w waiting at all traffic light intersections under the region of ith agent. We set δ_0 and δ_1, as 5 and 10, respectively. Motivation behind this selection are the real-world traffic observations of stopped vehicles and the fact that respective three congestion levels significantly reduce the state dimensions. We formulate x_i as:

$$x_i = \left\{ \begin{array}{ll} 0 & \text{if } N_w \le \delta_0 \\ 1 & \text{if } \delta_0 < N_w \le \delta_1 \\ 2 & \text{if } N_w > \delta_1 \end{array} \right\}. \tag{10}$$

The g_1, g_2 are indices of the two most congested traffic light intersections and c_1, c_2 are their respective congestion levels formulated similarly as x_i. The a^i represents the index of previous action for agent i. The joint state for agent i can be represented as a $k \times 6$ matrix S_i:

$$S_i = \begin{bmatrix} s_i \\ s_1 \\ . \\ s_{k-1} \end{bmatrix} = \begin{bmatrix} x_i & g_1^i & c_1^i & g_2^i & c_2^i & a^i \\ x_1 & g_1^1 & c_1^1 & g_2^1 & c_2^1 & a^1 \\ & & \cdots & & & \\ x_{k-1} & g_1^{k-1} & c_1^{k-1} & g_2^{k-1} & c_2^{k-1} & a^{k-1} \end{bmatrix}, \tag{11}$$

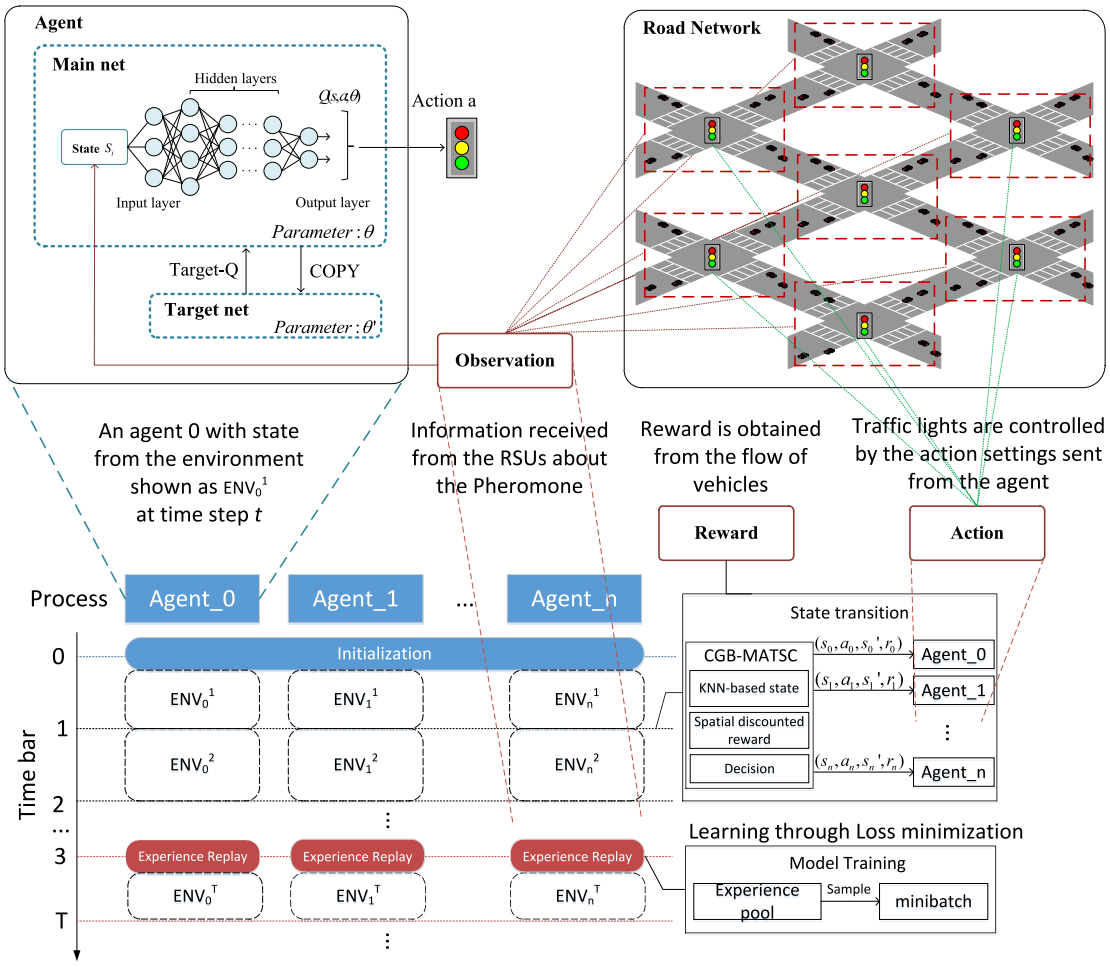

Fig. 6. Configuration of CGB-MATSC in the MARL framework.

where k is the number of k-nearest neighbors of agent i including itself. We will now present the selection mechanism of k nearest neighbors. In Fig. 5, the distances are shown for each MEC or Agent. For brevity we show the distances relative to an agent i with any other agent j as d_{ij}. In this way for the typical case of four agents, we will have these distances: d_{00}, d_{01}, d_{02}, d_{03}, d_{10}, d_{11}, d_{12}, d_{13}, d_{20}, d_{21}, d_{22}, d_{23}, d_{30}, d_{31}, d_{32}, d_{33} in all, and out of these distance, we will obtain the useful distances as: d_{01}, d_{02}, d_{03}, d_{12}, d_{13}, d_{22}, and d_{23}, (because other distances are either 0 or same $d_{ij} = d_{ji}$). After that select the k-nearest center points, in this case for each agent, there will be two more points. Assign the center point to the neighboring class to which majority of the k center points belong.

Action: In the circular regional areas as shown in Fig. 5, the number of vehicles under traffic light g of MEC i can be represented as $N_{i,t}^g$. The number of controlled lanes is N_{lane}^g. The pheromone belonging to the controlled edge of traffic light g at time t is given by:

$$\rho_t^g = \frac{N_{i,t}^g}{L_{length} \times N_{lane}^g}. \tag{12}$$

The road pheromone is directly proportional to the number of vehicles, and inversely to the length of the road L_{length}, as well as the number of lanes. There are two major advantages: (1) the final phase is chosen from the presetting phase configuration, which means the risk caused by random and unpredicted phase duration can be avoided for safety, (2) we extend the usual conception of general traffic pheromone by incorporating lane number and lane length, which ensures the objective measurement of the number of vehicles on road capacity. We have reduced the action space A of an agent to binary choices:

$$A : \left\{ \begin{array}{ll} \text{if } a^i = 0 & \text{perform green wave Algorithm 1} \\ \text{if } a^i = 1 & \text{perform periodic TLC} \end{array} \right\}, \tag{13}$$

the best response π_t^i of an ith agent is determined through iteration. In general, the average action $\bar{a}^i$ of all k-nearest neighbors of agent i is obtained by taking the average of their actions following the policies π_t^k governed by their previous mean actions $\bar{\boldsymbol{a}}^k$:

$$\bar{a}^i = \frac{1}{N^i}\sum_k a^k, a^k \sim \pi_t^k(.|s, \bar{\boldsymbol{a}}^k), \tag{14}$$

the policy π_t^i changes consequently because of the dependence on the current $\bar{a}^i$. The behavior policy can be written as Boltzmann policy:

$$\pi_t^i(a^i|s, \bar{a}^i) = \frac{\exp(-\beta Q_t^i(s, a^i, \bar{a}^i))}{\sum_{a^{i'}\in\mathcal{A}^i}\exp(-\beta Q_t^i(s, a^{i'}, \bar{a}^i))}, \tag{15}$$

where, the pairwise interaction $Q^i(s, a^i, \bar{a}^i)$ is approximated as described in Stanley (1971) and the fixed parameter β determines the exploration strategy. Iteratively, the mean actions $\bar{a}^i$ and the corresponding policies π_t^i improve in an alternate fashion. After several iterations the mean action $\bar{a}^i$ will be at an unique point. We keep the action space binary for each agent, and Appendix proves that with sufficiently large temperature, the convergence is guaranteed. We devise a regional pheromone-based green wave control mechanism to acquire the most appropriate phase sequence index using the latest intersection's state. MEC servers share real-time traffic information as $\text{INFO}_{i,t}$.

Fig. 6 describes the configuration of CGB-MATSC to illustrate the state transitions in the MARL framework. We run the agents in parallel, so that at each time step t, each of them observes a part of the environment ENV_{agent}^t, each agent has access to its neighbor's information. The road network illustrates the environment. The RSUs provide observation to the agent so that it can use it to generate the state, and then take an action to make a transition to the next state. The next state is provided by the environment. An agent will observe a state in each time step. This state may or may not be the same, depending on the variable s_i. As a result of each state transition, the spatial-factored reward is obtained in terms of $r_{j,joint}$ to the agent. The size of state, only depends on the number of agents used in the MARL, instead of, the number of intersections, which is a distinct factor of the proposed framework.

Algorithm 1 presents the regional green wave control for traffic lights. The $\text{INFO}_{i,t}$ consists of several dictionaries, which we will use to explain the regional green wave control mechanism:

Step 1: Based on the agent ID ($agent_i$), initialize the set of traffic lights $G_i = \{g_1^i, g_2^i, g_3^i, \ldots, g_{last}^i\}$ inside its circular region.

Step 2: Get the set $L_i = \{l_1^{g_1}, l_2^{g_1}, \ldots, l_m^{g_{last}}\}$ of all edges, controlled by every traffic light under the $agent_i$, where $l_m^{g_{last}}$ is the last edge of the last traffic light g_{last} (or g_{last}^i) of $agent_i$. Please note that a dictionary (or dict in short) means a kind of data structure which holds a value for a key.

As an example dict[k] = {key: value } represents a dictionary for the kth variable, and it has one key and its respective value. A dictionary data structure may have more keys and their respective values.

Step 3: From the above two steps use G_i and L_i to get the set $Ph_i = \{D_{Ph,t}[g_1], D_{Ph,t}[g_2].., D_{Ph,t}[g_{last}]\}$ containing preset phase sequence dictionary elements for every edge of every traffic light under $agent_i$. So, we can express the preset phase dictionary $D_{Ph,t}$ as:

$$D_{Ph,t} = \left\{ \begin{array}{l} g_1 : \{l_1^{g_1} : (start_1, end_1)^{seq1}, l_2^{g_1} : (start_2, end_2)^{seq1}, \ldots\} \\ g_2 : \{l_1^{g_2} : (start_1, end_1)^{seq1}, l_2^{g_2} : (start_2, end_2)^{seq1}, \ldots\} \\ \ldots \\ g_{last} : \{l_1^{g_{last}} : (start_1, end_1)^{seq1}, \ldots\} \end{array} \right\}. \tag{16}$$

A dictionary element $D_{Ph,t}[g_1]$ has keys as the edges, and values as their corresponding start and end indexes of phase sequences. To understand, we can write the expression for $D_{Ph,t}[g_1]$ as:

$$D_{Ph,t}[g_1] = \{l_1^{g_1} : (start_1, end_1)^{seq1}, l_2^{g_1} : (start_2, end_2)^{seq1}, \ldots\}, \tag{17}$$

where $(start_1, end_1)^{seq1}$ represents the start and end of the phase sequence with index 1 for the edge $l_1^{g_1}$. For example, in Fig. 4, the $D_{Ph,t}[9]$ will represent preset phase sequence dictionary for the traffic light 9 as $\{l_1^{g_9} : (0,3)^{seq1}, l_2^{g_9} : (5,7)^{seq1}, l_3^{g_9} : (8,11)^{seq1}, l_3^{g_9} : (12,15)^{seq1}, \ldots, \ldots, l_3^{g_9} : (12,15)^{seq4}\}$. The corresponding traffic signals for each of the edges will be: 'Gggg' for $l_1^{g_9}$, 'rrrr' for $l_2^{g_9}$, 'rrrr' for $l_3^{g_9}$, and 'GGGg' for $l_4^{g_9}$ for the phase sequence $seq1$ which is "GgggrrrrrrrrGGGg". Similarly, there are more keys and values for the remaining $seq2$, $seq3$, and $seq4$ in the $D_{Ph,t}[9]$.

Step 4: For each time step check if it is the δ = 20th time step, then move to step 5.

Step 5: Update the number of vehicles $N_{i,t}^g$ based on real-time information from RSUs and then make a dictionary $D_{i,t}$ that contains number of vehicles at time step t, so that for a traffic light g_1 the $D_{i,t}[g_1]$ can be expressed as:

$$D_{i,t}[g_1] = \{l_1^{g_1} : (N_{i,t}^1), l_2^{g_1} : (N_{i,t}^2), \ldots\}, \tag{18}$$

where edge $l_1^{g_1}$ is the key and the value $N_{i,t}^1$ is its corresponding number of vehicles at time step t and so on.

Step 6: Similarly, update the pheromone dictionary $D_{\rho,t}$ by applying Eq. (12) on $D_{w,t}[g_1]$, so that we can express it for the traffic light g_1 as:

$$D_{\rho,t}[g_1] = \{l_1^{g_1} : (\rho_t^{g_{1,1}}), l_2^{g_1} : (\rho_t^{g_{1,2}}), \ldots\}, \tag{19}$$

where $\rho_t^{g_{1,1}}$, and $\rho_t^{g_{1,2}}$ are respectively the pheromone values for the edges $l_1^{g_1}$ and $l_2^{g_1}$.

Step 7: Update an edge congestion dictionary $D_{c,t}$, and for each traffic light g, get the edge index l^g_{max} which has maximum pheromone value inside $D_{\rho,t}$, as:

$$D_{c,t} = \{g_1 : \arg\max_{l^{g_1}_{max}} D_{\rho,t}[g_1], g_2 : \arg\max_{l^{g_2}_{max}} D_{\rho,t}[g_2], \ldots\}. \tag{20}$$

Step 8: For every traffic light g and its corresponding each edge with maximum pheromone in $D_{c,t}$, get corresponding start and end indexes of the phase sequences $seq*$ (where * represents index of all sequences e.g., 1, 2, 3, and 4) using $D_{Ph,t}$ from Step 3, and make a dictionary $D_{l^g_{max},t}$ for them:

$$D_{l^g_{max},t} = \{l^{g_1}_{max} : (start_{max}, end_{max})^{seq*}, l^{g_2}_{max} :, \ldots\}. \tag{21}$$

Step 9: Access phase sequence for each edge inside $D_{l^g_{max},t}$ from dictionary $D_{Ph,t}$, where a sequence example for the traffic light g_1 will be $D_{Ph,t}[g_1][l^{g_1}_{max}]$, which is $(start_{max}, end_{max})^{seqj}$ representing jth preset phase of g_1.

Step 10: Update a dictionary $D_{green,t}$. Iterate and check for all edges in $D_{l^g_{max},t}$ to find corresponding phase sequence index for which all lanes of the edge are allowed to pass, e.g., 'Gggg'. Namely, if for example $seqj$ is $'green'$ (all elements for lanes are either 'g' or 'G') for its respective $(start_{max}, end_{max})^{seqj}$ then add index of corresponding phase $seqj$ in $D_{green,t}$ as a value for the key g_1 traffic light, similarly do the same for other traffic lights (for example $seqm$ as mth sequence for g_2) as:

$$D_{green,t} = \{g_1 : (seqj), g_2 : (seqm), \ldots\}. \tag{22}$$

Up till Step 10 we have accessed corresponding phase sequence indexes for edges with most pheromone for all traffic lights inside $agent_i$.

Step 11: Set green phase for the traffic lights based on $D_{green,t}$ at current time step t. This operation is perfectly safe, because a traffic light can only be assigned one of the preset phase sequences at a time. With time step keep counting, check if it is the last step of current episode, if yes then end the current episode and start the new one, otherwise move to Step 4.

Fig. 7 provides the process flow of the proposed pheromone-based regional green wave control algorithm. Fig. 8 presents preset phases of various heterogeneous intersections. We keep the preset signal phase information instead of changing the duration of each phase in a cycle.

The motivation behind this approach is to avoid frequent and uncertain phase rotation of signal lights. According to the feedback of the real-time pheromone list from each MEC server, the traffic lights corresponding to the most congested edges will get a list of selectable phase indexes. In this way, a repeated action to select green phase for the same edge is equivalent to extending the duration of the green phase.

Reward: Let f^i_t be the average flow of vehicles at time t in the region of an agent i and given by:

$$f^i_t = \frac{1}{N_{G_i}} \sum_{g \in G_i} \rho^g_t \times v^g_t, \tag{23}$$

where v^g_t is the average speed of vehicles at time step t under the traffic light g, and N_{G_i} is the total traffic lights under the region of agent i. The local return of an agent i is given by:

$$r_{i,local} = \left\{ \begin{array}{ll} 1 & \text{if } f^i_t > f^i_{t-1} \\ 0 & \text{if } f^i_t = f^i_{t-1} \\ -1 & \text{if } f^i_t < f^i_{t-1} \end{array} \right\}. \tag{24}$$

In real scenarios, the locations of agents and their circular regions may not be equally spaced. The nearby agents have higher correlation characteristics of pheromone. Therefore, we introduce a spatial discount factor $\frac{1}{|d_{ij}|}$, so that each agent incorporates the impact of its own as well as its nearest neighbors' actions on the environment. The joint return based on spatial discount factor is as follows:

$$r_{i,joint} = r_{i,local} + \sum_{j \in N_i} \frac{1}{|d_{ij}|} \times r_{j,local}, \tag{25}$$

where $|d_{ij}|$ is the distance between an agent i and agent j, and N_i is the k nearest neighbor list of agent i, $r_{j,local}$ is the local return of its jth neighbor agent.

Algorithm 2 provides the steps to achieve CGB-MAQL by using the state, action and reward values.

5. Experimental evaluations

To get credible experimental results, we build simulated road networks in SUMO as an environment for interaction. We evaluate the proposed algorithm by comparing it with other algorithms on a 6 × 6 intersection synthetic road network and two cities: (1) Monaco, and (2) Harbin with multiple heterogeneous intersections.

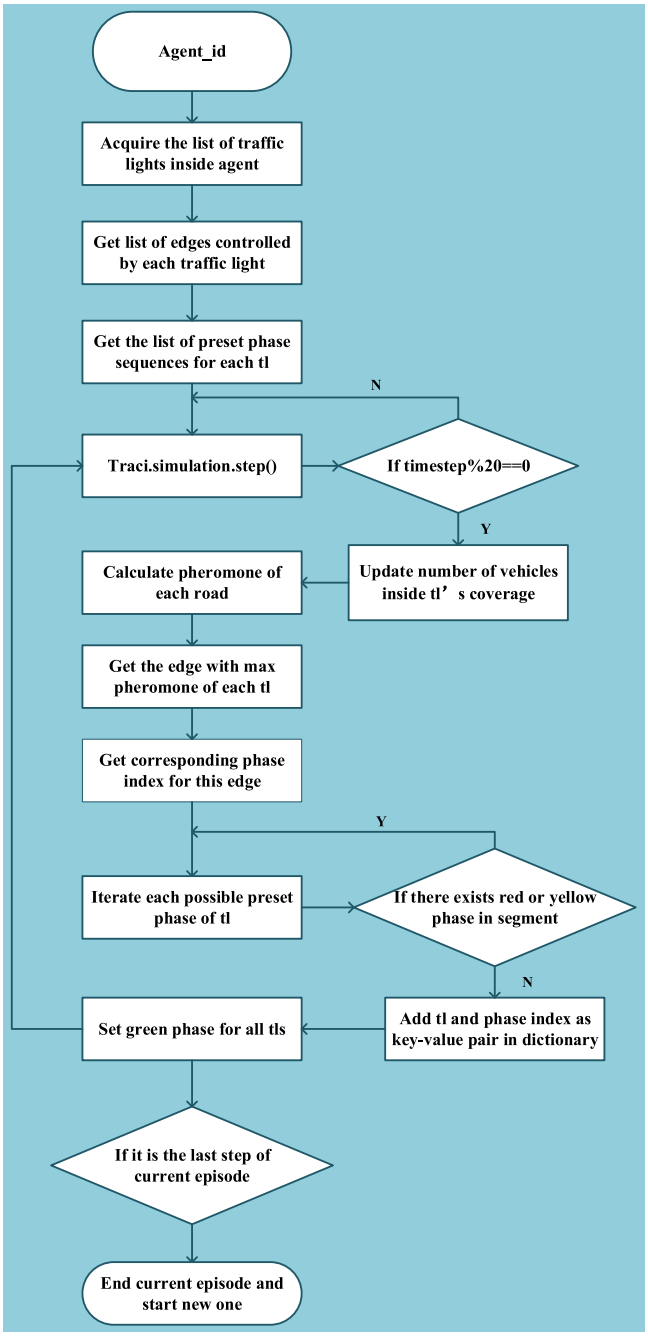

Fig. 7. Flow chart for pheromone-based regional green wave control mode.

5.1. General setups

Platform: SUMO is an open-source, highly portable, microscopic, and continuous traffic simulation package designed to handle large road networks. It is mainly developed by employees of the Institute of Transportation Systems at the German Aerospace Center (Krajzewicz et al., 2012a). SUMO builds the real road network structure through various road architecture files, giving detailed descriptions of nodes, edges, traffic light architecture, and so on. Routes of the vehicle can be designed manually or generated randomly via embedded tools. In SUMO, real-time acquisition of road information and implementation of management and control mainly depend on the interface named TraCI, by which the latest calculated results of the executed algorithm and intersection controllers are connected. Various kinds of control commands are provided via TraCI to conduct full control of intersections.

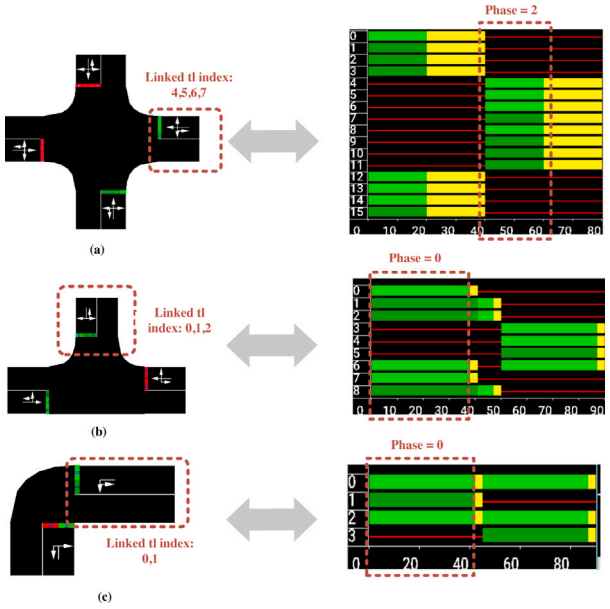

Fig. 8. Pheromone-based green phase setting for heterogeneous intersections inside synthetic scenario.

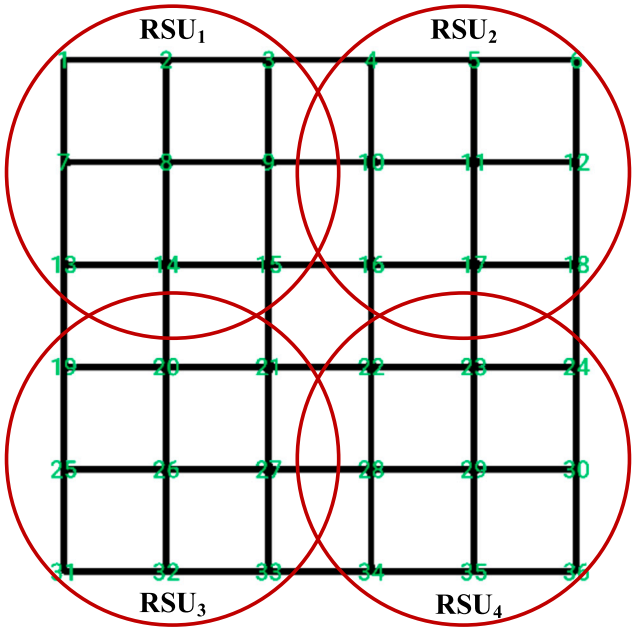

Fig. 9. Scenario of synthetic road network in SUMO with 4 RSU deployed.

Basic configuration for RL: The hyper-parameters for environment establishment and agent are in Tables 3 and 4, each with a short description of the basic setups.

(1) Environmental parameters: As far as settings for the environment, all roads in the scenario are 100 m long. There are various heterogeneous intersection controllers, which are two-lane intersection, three-lane intersection, and four-lane intersection in the synthetic networks, whose details are in Fig. 9. We set 5000 episodes for agents to learn from enough rounds of trail-and-errors. Taking actual conditions into consideration, we will execute real-time control every 20 s. In addition to the above, we set three discrete levels for the congestion description, corresponding to smooth, normal, and congested traffic flow conditions. In our case the congestion levels are decided by the number of waiting vehicles of less than 5, between 5 to 15, and above 15 cases.

(2) Hyper-parameters for an agent: In our proposed algorithms extending DQN, the Q-value approximation is based on DNN. The number of model layers and neurons should be appropriate for each specific question. In our case, there are fixed-period mode and

Algorithm 2: CGB-MAQL algorithm

Input : $|B|$, ϵ_{decay},ϵ,α,γ,A,δ, INFO_i
Output: Optimal Q values with trained weights
Initialize group as a dictionary dict
// Initialize time step
$t \leftarrow 0$
Initialize all Q-values randomly for all agents
Initialize $Memory$ for all agents
Calculate N_i based on distance
repeat
  For each step inside an episode
  if t % δ=0 **then**
    repeat
      // Obtain state information from RSUs
      $S_i \leftarrow \text{INFO}_i$ using Eq. (11)
      // Perform DQN on S_i using Eq. (3)
      // Generate a random float in range 0 to 1
      $rand \leftarrow$ random(0,1)
      // Choose action and execute it
      $a' = \begin{cases} \arg\max_{a'} Q(s', a') & \text{if } rand > 1 - \epsilon \\ \text{random } a' \in A & \text{otherwise} \end{cases}$
      // Send action information to RSUs
      Execute a' using Algorithm 1 and Eq. (13)
      Get latest $s'_{i,local}$ and $r_{i,local}$ using Eq. (24)
      Update $done$ flag on termination limits
    until *The last agent i*;
    repeat
      Append $s'_{j,local}$ to $s'_{i,joint}$
      // Calculate joint reward
      Obtain $r_{j,local}$ using Eq. (24)
      Get $r_{i,joint}$ using $r_{j,local}$, $r_{i,local}$ in Eq. (25)
    until *Each neighbor j in* N_i;
    Append (s', a', $s'_{i,joint}$, $r_{i,joint}$, $done$) in $Memory_i$
    if *length of* $Memory_i > |B|$ **then**
      Update Q-value function
      $\epsilon \leftarrow \epsilon \times \epsilon_{decay}$
    end
  end
  $t \leftarrow t + 1$
until *The last episode ends*;

Table 3
Configuration setups of environment.

Parameter	Description
Synthesized intersections	6 × 6
Geographic region of synthesized intersections	500 m × 500 m
Monaco city intersections	324
Geographic region of Monaco city	1000 m × 1000 m
Harbin city intersections	341
Geographic region of Harbin city	6000 m × 7000 m
Simulated platform	Sumo, keras
Training episodes	5000
Management time interval	20 s
Discrete congestion level	[0, 1, 2]

regional pheromone-based green-wave mode for each agent, making action space size 2. Thus we set the size of output layers of our neural networks as 2. In another extension of A2C based on CGB-MATSC, the size of the output layer for actor and critic networks

Table 4
Configuration hyper-parameters of agent.

Parameter	Description
Deep neural network	[24,24,24,2]
Actor–Critic network	[24,2],[24,24,1]
Activation	ReLU, Sigmoid
Learning rate α	0.001
Initial epsilon ϵ	1
Final epsilon	0.01
Epsilon decay ϵ_{decay}	0.99
Loss function	MSE
Optimizer	Adam
Batch size $\|B\|$	8

are 2 and 1 respectively, since the actor is supposed to calculate an action and critic is responsible for evaluating the behavior policy taking by the current actor. All agents will learn to follow the exploration–exploitation mode to have complete trails of all possible behavior policies. An agent under exploration will evaluate the new strategies with the probability of ϵ. Otherwise, it will behave as an optimum policy at that moment.

5.2. Benchmarks and evaluation index

Benchmark strategies: The primary control method is a periodic method, of which the phases and duration are all preset and fixed for all intersection controllers. The agent's learning of dynamic traffic flow characteristics is not involved. In addition to the periodic strategy (FIXED), we will also compare our proposed CGB-MAQL algorithm with traditional independent Q-learning strategy (Tesauro, 2003), greedy Q-learning, as well as the state-of-art DQN algorithms MA2C (Chu et al., 2020; Wang et al., 2016). All the algorithms are tested based on the proposed hierarchical three-layer communication architecture to guarantee a fair comparison. Here we named the extensions as: greedy policy, IQL, MA2C as CGB-Greedy, CGB-IQL, and CGB-MA2C. In the CGB-Greedy, all intersection controllers try to set the green phase for the most congested direction. In the CGB-IQL method, all agents are learning from their own local state and local reward. The behavior of other agents would be part of the environmental dynamics in this way. The only significant difference between CGB-MA2C and CGB-IQL is the network architecture for the agent. The motivation behind the selection of baseline algorithms was to investigate their performances under reduced action space. Moreover, these algorithms have already been investigated under the MARL, therefore, we they suit well for the evaluation of the proposed framework.

Evaluationindex : To make a more comprehensive comparison of the proposed algorithms, we carried out four rounds of tests. We recorded the road pheromone, joint return, waiting time in second as indicators of four agents. The histogram below shows the average level of the above signs in 4 rounds. Table 5 below gives a more detailed view of the above indicators in terms of the average and maximum levels for the synthetic scenario:

$$\hat{R}_{i,joint} = \frac{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\sum_{l\in Edges_i} r_{i,joint}}{E\times T_e}, \tag{26}$$

$$\hat{P}_i = \frac{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\sum_{l\in Edges_i} \rho_t^i}{E\times T_e}, \tag{27}$$

$$\hat{W}_i = \frac{\sum_{e=1}^{E}\sum_{t=1}^{T_e}\sum_{l\in Edges_i} N_{w,l,i,t}}{E\times T_e}. \tag{28}$$

Here E is total number of tests and T_e represents the ending step of episode e, and t represents each managing time step. $Edges_i$ contains information of all edges inside $agent_i$. In the same way, we calculated the local joint return level of each agent $\hat{R}_{j,i}$, and the average road pheromone level $\hat{P}_i$ as well as average vehicle waiting time of each edge $\hat{W}_i$.

5.3. Scenario A: Synthetic networks

Setups:
As shown in Fig. 9, we build a 6 × 6 road network for initial evaluations. We set length for all roads as 100 m. The speed limit on each street is 5m/s and limit on acceleration and deceleration is 1m/s^2. We set the sum of vehicles as 128 with spawn rate as 0.2 veh/s for the initial 20 s.

TrainingResults :
Fig. 10 shows how average accumulative rewards of an episode changes as exploration rate decreases. We simplify the design of each action as +1, 0 to prove the convergence of our algorithm. We record the accumulative return of each control moment when all vehicles in a scene end their trips. We take a heuristic training method to accelerate the agent's learning process, where the agent will terminate the current exploration of latest behavior strategy when total number of stopped vehicles at a certain intersection exceeds a threshold $\theta_{termination}$. Following by new episode of trail-and-error on new behavior strategy, unnecessary explorations

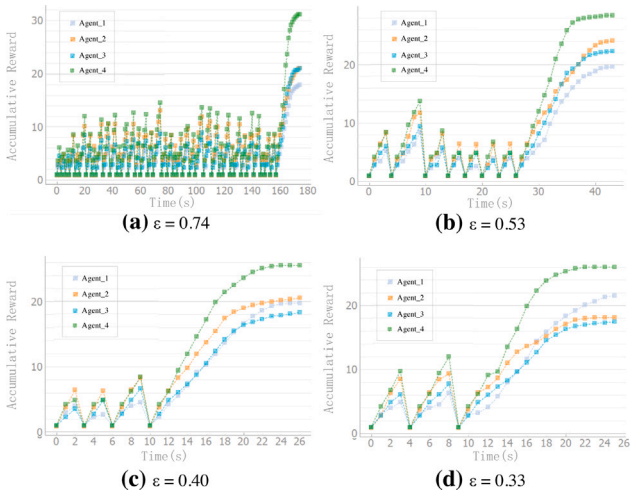

Fig. 10. Convergence of training process as exploration rate decreases for synthetic scenario.

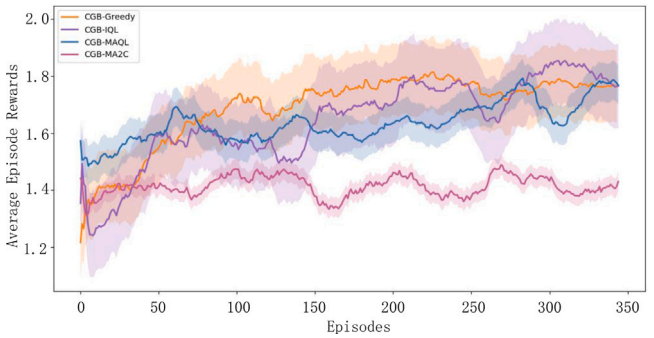

Fig. 11. Average reward during training process of synthetic scenario.

Table 5
Overview of average and maximum value of metrics for all policies for synthetic scenario.

Metrics	Avg.					Peak				
	FIXED	CGB-Greedy	CGB-IQL	**CGB-MAQL**	CGB-MA2C	FIXED	CGB-Greedy	CGB-IQL	**CGB-MAQL**	CGB-MA2C
Waiting Time (s)	326.19	140.33	216.45	**123.87**	158.46	423.39	339.40	334.34	**240.71**	191.63
Pheromone (veh/m)	0.21	0.17	0.20	**0.20**	0.19	0.23	0.24	0.23	**0.23**	0.22
Ending time	585	431	359	**332**	410	585	431	455	**356**	434

are efficiently avoided. Although the small value of $\theta_{termination}$ would accelerate the learning process theoretically, however it may result in frequent and early quit of current exploration, which increases the risk of failure for convergence caused by inadequate learning of agent. We have tried with several $\theta_{termination}$ and finally set it as 10 for all scenarios. The purpose of this is to eliminate unnecessary exploration of extreme traffic congestion cases during the model training, which reduces the computational cost of learning undesirable strategies.

To visualize the impact of heuristic training on the process of convergence of each agent, we recorded the changes in the cumulative return during the training process as the time step increased. Four representative training records are selected to illustrate the shift in reset times with the exploration rate decreasing. To give a more precise description of the positive return of action, we set immediate reward as 1 for any flow increasing case for each intersection inside an agent. Fig. 10 records the change in each agent's cumulative return value in each complete round as the exploration rate gradually decreases. The sudden drop point represents the reset of the scene caused by the heuristic termination condition, and the curve tends to be stable in the end. The time of breaking off via reaching terminated state shows a decreasing tendency to explore rate decreasing. It indicates that all agents are learning towards the target set initially, and are trying hard to converge towards a policy that decreases congestion at all intersections, boosting traffic flow. In reducing the exploration rate from 1 to close to 0, the agent will gradually adopt the behavior strategy of optimal mode selection.

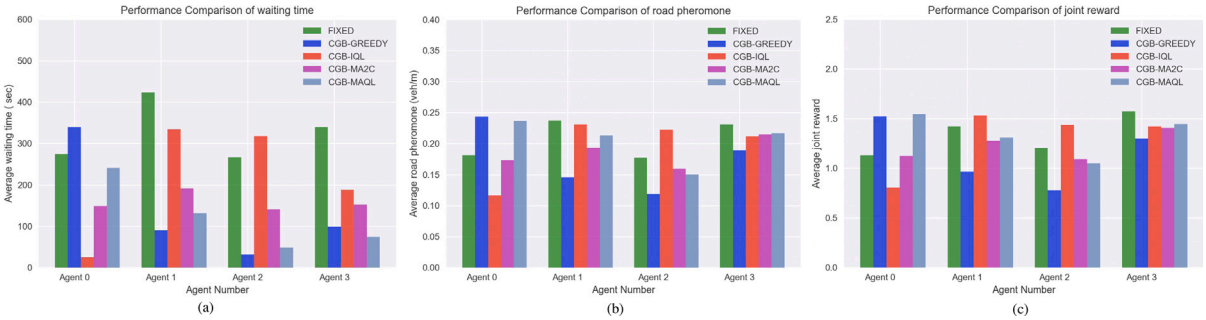

Fig. 12. Bar charts on average level of 3 metrics for all agents of synthetic scenario. (a) Average waiting time (lower is better), (b) Average road pheromone (lower is better), (c) Average joint reward (higher is better).

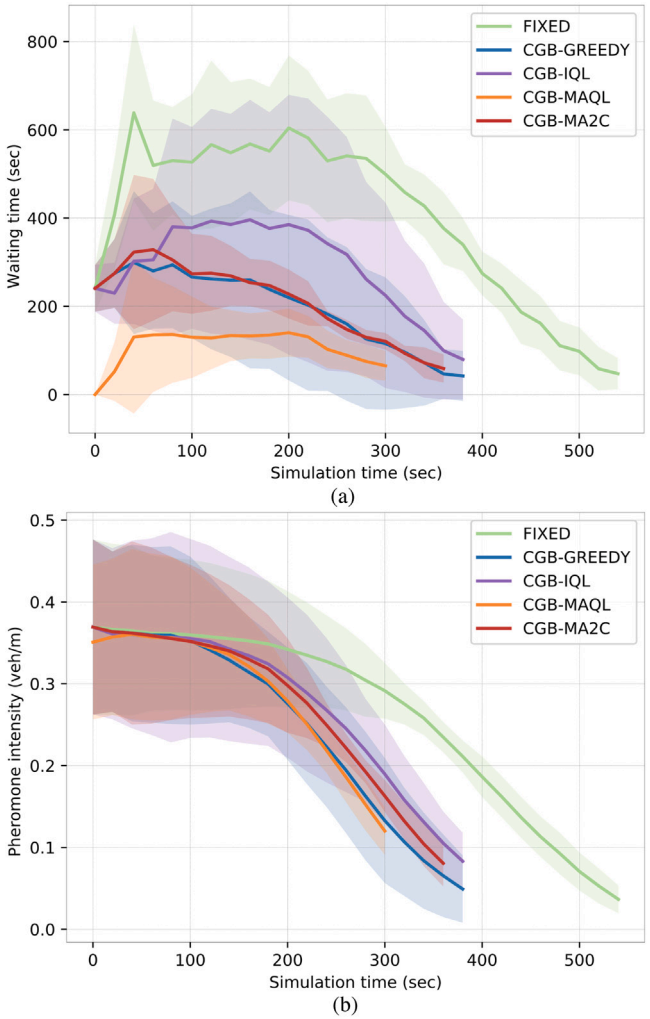

Fig. 13. Plots of all strategies of synthetic scenario. (a) Average waiting time (s). (b) Average pheromone intensity (veh/m).

Evaluation of test results:

To verify each strategy's convergence in the training process, we recorded the average cumulative return of each approach in the initial 350 rounds of the training phase. Fig. 11 presents the average return of each strategy from the beginning of training as the training progresses, with maximizing the traffic volume at the intersection as learning target. With fixed vehicular routes, the higher efficiency of vehicle circulation corresponds to the shorter transit time of all vehicles. Therefore, we define the average

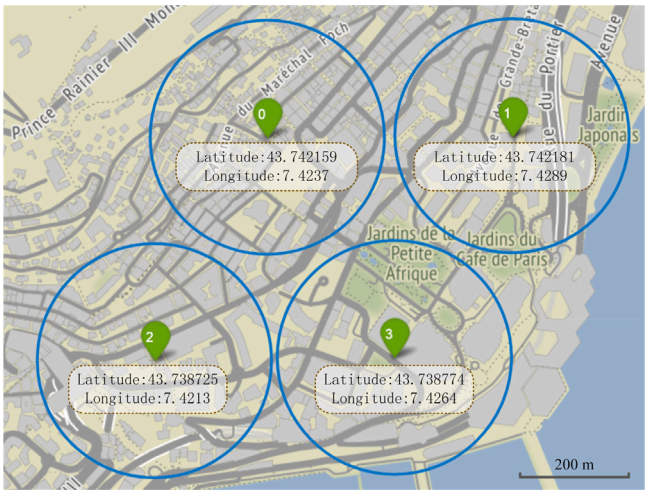

Fig. 14. Multi-Agent deployment in Monaco city.

cumulative return $\hat{R}_{episode}$ as the following formula:

$$\hat{R}_{episode} = \frac{E_{Fixed}}{E_{otherpolicies}}. \tag{29}$$

where $E_{otherpolicies}$ represents the end time of a round of testing. The E_{Fixed} for the fixed strategy is 585 in our synthetic case.

Fig. 11 shows that the overall trend of the average return curve of all strategies is rising, which indicated that the ending time tends to decrease with training. The time for the vehicle to complete the route is getting shorter and shorter, meaning that the agent under each strategy can move towards our specific goal of maximizing vehicle circulation. With the apparent upward trend of the curves, each RL-based approach shows a fast and robust learning ability and convergence speed. This preliminary experiment proves that our framework and algorithm can realize the practical MARL application in large-scale road network scenarios. The CGB-Greedy strategy achieves faster and more stable convergence than other approaches with a few and small fluctuations in its rising process. It is because the greedy nature of the behavior strategy of the algorithm determines that it starts to search and learn the optimal strategy at the beginning of training. When a new optimal strategy is found, the agent does not explore other possibilities but keeps going deeper directly. The advantage is that stable learning can be achieved at the beginning of training, but there is still a risk of convergence to sub-optimal strategy. As we can see, the greedy approach has shown a stable trend from about 150 rounds. The lack of attempts to explore better strategies fails to give higher average returns, as compared to other behavioral strategies with explorations like CGB-IQL and CGB-MAQL at around 270 rounds and 300 rounds, respectively.

Compared with the behavior strategy of CGB-Greedy with restricting exploration, other strategies all adopt the approach of exploration–exploitation behavior. The influence of exploration on agents' learning induces fluctuations in the average return curve. We find that even in the rising trend, there are still frequent drops, which means that agents are trying to use behavioral strategies other than the current optimal strategies to promote agents to explore more strategies. The probability of exploration is determined by the exploration rate of ϵ, which will decrease based on a decay rate to 0 with initial value as 1. It means that at the beginning of the training, the agents will invest most of the time on learning from trial and error, and in the later stage of training, strategies are gradually converging to the optimal. The shaded portion of the graph represents the standard deviation of the average return on 350 rounds of training. It can be seen that the standard deviation of CGB-MAQL and CGB-MA2C is the smallest, and the standard deviation of CGB-IQL is the largest. The instability of CGB-IQL is that when all agents are learning their optimal local strategies separately, the impact of other agents' behavior on the environment is regarded as a part of the dynamic randomness of the environment, which indicates no cooperation among agents at all. The zero cooperation between agents limits its application in large-scale problems.

Fig. 12 shows each agent's performance details, and the average and peak results are in Table 5. Compared to the FIXED strategy, CGB-MAQL shows superiority in all indicators. It reduces average waiting time by 62.03%, compared to CGB-IQL, CGB-MA2C, and CGB-Greedy with 42.77%, 21.83%, and 11.73%, respectively. As a result of the lack of cooperation among agents, CGB-IQL shows the apparent uneven distribution of agents' performance of CGB-IQL in bar charts as well as the large standard deviation in the figure. This result is consistent with our pre-set learning goal: to select the optimal signal strategy to maximize the traffic flow on the road. In terms of average road pheromone, compared with FIXED as 0.21 veh/m, CGB-MAQL achieves 4.76% reduction.

To compare each strategy's performance, we comprehensively evaluated the approach from road pheromone and vehicle waiting time. The average level of four rounds of tests is recorded in Fig. 13, to obtain credible results. It describes the mean level and standard deviation of each strategy's best series of tests on five indicators. Fig. 13(a)–(b) shows the average level and standard deviation of the high index for four agents in a round of tests. The time step corresponding to the end of the curve shows the time

Table 6
Overview of average and maximum value of metrics of 5 strategies for Monaco scenario.

Metrics	Avg.					Peak				
	FIXED	CGB-Greedy	CGB-IQL	**CGB-MAQL**	CGB-MA2C	FIXED	CGB-Greedy	CGB-IQL	**CGB-MAQL**	CGB-MA2C
Waiting Time (s)	32.13	17.74	39.62	**18.61**	28.41	40.87	45.83	69.08	**41.74**	40.61
Pheromone (veh/m)	0.23	0.22	0.25	**0.21**	0.23	0.32	0.36	0.43	**0.33**	0.31

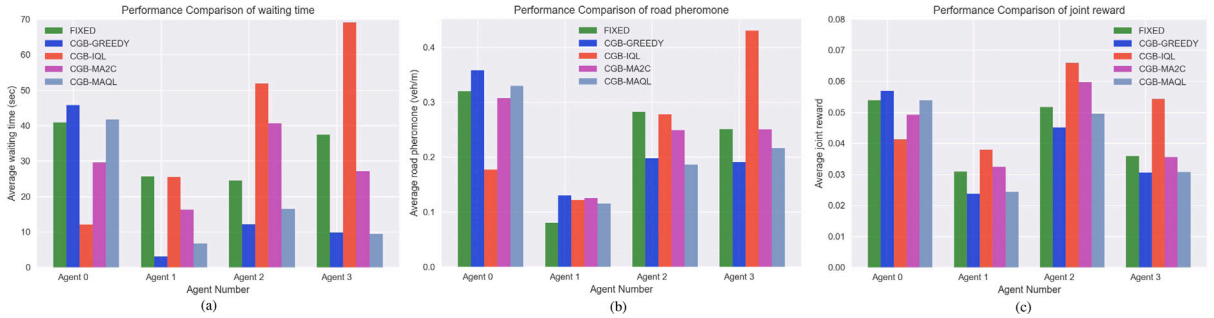

Fig. 15. Bar charts on average level of 3 metrics for all agents of Monaco scenario. (a) Average waiting time (lower is better), (b) Average road pheromone (lower is better), (c) Average joint reward (higher is better).

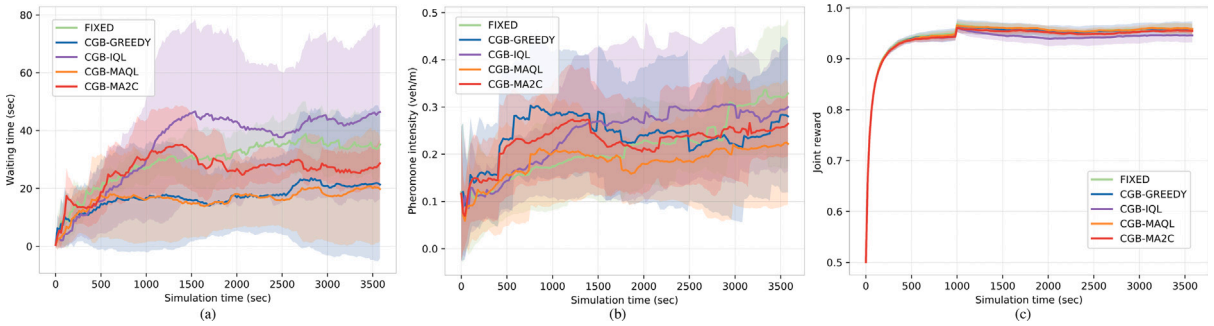

Fig. 16. Plots of average of various metrics of all four agents in all strategies of Monaco scenario. (a) Average waiting time (s). (b) Average pheromone intensity (veh/m). (c) Average joint reward.

taken for all vehicles to arrive at the destination. From Fig. 13(a), we can see that CGB-MAQL takes a shorter time to complete all routes of vehicles than other algorithms. CGB-MA2C and CGB-Greedy show better performance than CGB-IQL, and the worst is the Fixed Strategy. The poor performance of CGB-Greedy on the standard deviation is due to its incomplete knowledge of the environment due to lack of exploration throughout the agents' learning stage.

5.4. Monaco city scenario

As shown in Fig. 14, we built a Monaco urban road network structure with an area of 1000 × 1000 m^2 on the SUMO simulation platform, with a total of 329 traffic lights controlled intersections within the road network to test the feasibility and scalability of our algorithm. We have deployed four agents with a coverage of 200 m. The bar charts in Fig. 15 show each agent's performance details under each strategy in the four rounds of testing. Table 6 gives the specific average and maximum values under each indicator. In terms of average waiting time, the CGB-MAQL strategy performs 42.08% better than the Fixed approach as 32.13 s It shows 53.03% better than the CGB-IQL strategy, and 34.50% better than the CGB-MA2C strategy.

As shown in Fig. 15(a)–(c), both of CGB-MAQL and the CGB-Greedy algorithm can significantly reduce vehicles' average waiting time. Still, the pheromone index indicates that our algorithm is superior to the CGB-Greedy algorithm on road congestion. In road pheromones, CGB-MAQL achieves the lowest average level with 8.70% lower then FIXED strategy as 0.23 veh/m, with CGB-Greedy 4.35% lower than FIXED, CGB-IQL higher than FIXED and CGB-MA2C same as FIXED respectively. All results indicate that agents under the CGB-MAQL algorithm can effectively learn to promote intersection traffic flow. We conducted a round of 3600 steps of testing. During the simulation, the vehicle arrival rate was 520 veh/hour.

Results on all metrics are shown in Fig. 16(a)–(c). From the perspective of waiting time, the performance of CGB-MAQL and the CGB-Greedy are equal. The waiting time of the CGB-MA2C strategy is slightly higher than the previous two. However, as we can

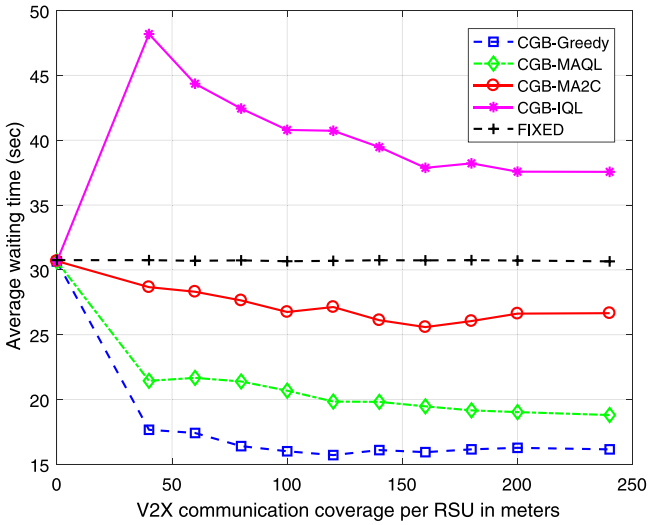

Fig. 17. V2X penetration results on the Monaco city scenario.

see from both bar charts and plots of curves, CGB-MA2C performs much better than CGB-IQL and FIXED strategies. Table 6 presents the overview of average and maximum value of metrics of 5 strategies for Monaco scenario.

Interestingly, the CGB-IQL strategy shows a significant standard deviation on all indicators, which is even worse than the FIXED strategy. The instability is because that each agent in CGB-IQL only concentrates on local optimal policy learning and ignores changes in the environmental status of other agents' behaviors, resulting in poor scalability caused by inaccurate records of executed actions. The difference between CGB-IQL and CGB-MAQL illustrates the importance of the collaboration in agents' learning process to the portability and scalability of a MARL system.

Fig. 17 presents the penetration results which show that with no communication coverage all schemes behave in the periodic or FIXED mode. The coverage of RSUs increases the penetration of connected vehicles, which results in an improved performance of the schemes, especially, the CGB-Greedy. We observed that although the performance of CGB-IQL was improved with the increase in the coverage range; but, it is still not better than the FIXED. These observations demonstrated the effectiveness of the proposed framework, which can utilize the cooperation of multiple agents to reduce the average waiting time of the vehicles.

In order to get a deeper insight on how explore–exploit mechanism impacts the MARL-based ATSC strategy, we have tested CGB-MAQL and CGB-MA2C algorithms with their pre-trained weights of relevant deep learning neural networks in Monaco scenario. The exploration rate or epsilon is set as: 0, 0.2, 0.4, 0.6 and 0.8, where 0 means total exploit mode, and 0.8 means the agents will start exploring for even better actions with an exploration rate of 80 percent. Fig. 18 shows the average waiting time during the whole episode. The performance of MAQL keeps on converging with the increase in epsilon, which can be described from these key insights: (1) Both CGB-MAQL and CGB-MA2C are deployed with well-trained models, and during the training process epsilon of each the algorithms keeps decreasing from 1 to 0.01. The continuous decay mode of exploration rate brings sufficient accumulation of an agent's experience. Based on well iterated weights of model, the impact of explore–exploit mode on control strategies can be accurately tested. (2) As shown in Fig. 18, the exploration rate still has an impact on the model control effect during test. Reinforcement learning can be seen as an online learning process. Even if in the test mode with exploration rate as fixed value, the experience replaying part is still enabled to update agents' experience. Based on sufficient experience and knowledge accumulation in the training stage, a greater exploration rate in the test means an attempt to expand the behavior space besides the current optimal decision-making with poor long-term reward. In addition, from the perspective of the impact of return mode on Q-value fitting, the integration of discount factor makes the behavior value fitting tend to be the overall long-term optimal. Therefore, even if each step of the behavior decision in the curve may not bring the current optimal result, as the bar chart shows, the test with the largest exploration rate achieves the smallest average waiting time in CGB-MAQL. However, it is observed that MA2C with its inherent actor and critic mechanism did not perform well with increasing exploration values, because the action selection policy highly depends on the number of actions in general A2C algorithms. It is also evident from the recent research works (Chu et al., 2020) that in the scenarios with larger action space the A2C based algorithms perform better as compare to the ones with reduced action space.

5.5. Harbin city scenario

As shown in Fig. 19, we built a Harbin urban road network structure with an area of 6000 × 7000 m^2 on the SUMO simulation platform, with a total of 341 traffic lights controlled intersections within the road network to test the feasibility and scalability of our algorithm. We have deployed four agents with a coverage of 500 m. We conducted a round of 3600 steps of testing. The bar

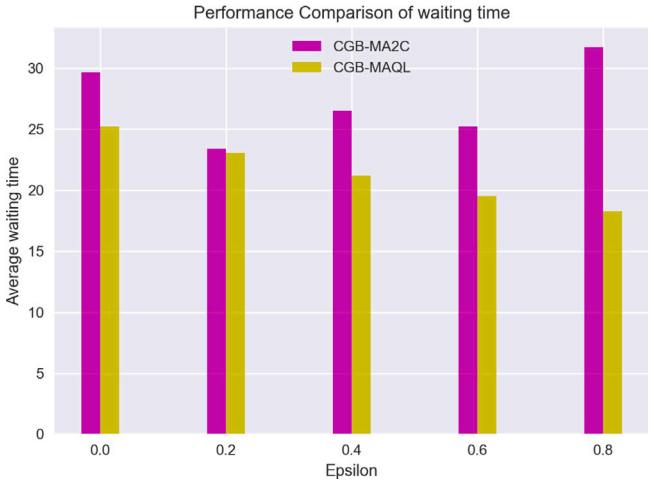

Fig. 18. Comparison of CGB-MAQL with CGB-MA2C under various Epsilon greedy values for the Monaco city scenario.

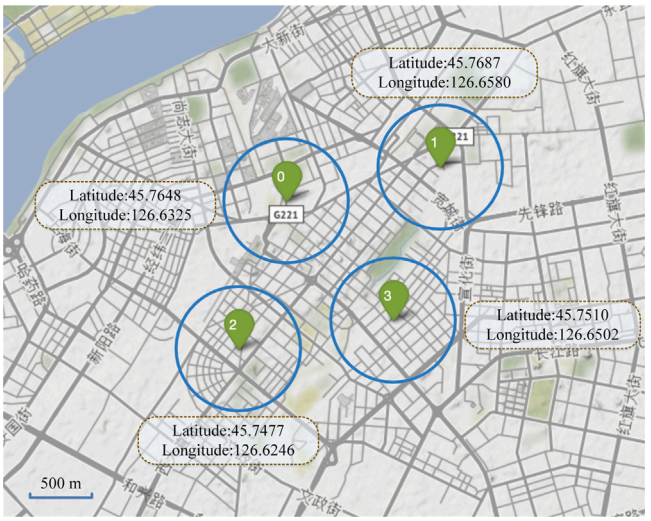

Fig. 19. Multi-agents deployment of Harbin city.

Table 7
Overview of average and maximum value of metrics for all policies for Harbin scenario.

Metrics	Avg.					Peak				
	FIXED	CGB-Greedy	CGB-IQL	**CGB-MAQL**	CGB-MA2C	FIXED	CGB-Greedy	CGB-IQL	**CGB-MAQL**	CGB-MA2C
Waiting Time (s)	24.2903	17.8750	119.0292	**5.1292**	38.3950	67.7778	62.1167	441.7833	**15.5667**	136.2611
Pheromone (veh/m)	0.0218	0.0181	0.049	**0.0115**	0.0278	0.0428	0.0396	0.1536	**0.0200**	0.0729

charts in Fig. 20(a)–(c) provides a more detailed view of each agent's behavior on average waiting time, average road pheromone and average joint reward during an episode respectively. As we can see all agents under control of CGB-MAQL perform the best with least average level on waiting time and road pheromone together with highest joint reward among all strategies under test. The CGB-GREEDY strategy also outperforms FIXED strategy on all metrics. CGB-IQL strategy shows an unfair performance among all agents with highest level of difference on waiting time, which tells the failure of traffic optimization over initial periodic control strategy.

The difference of CGB-MAQL and CGB-IQL shown in the above results proves the importance of coordination mechanism with MARL-ATSC problems. As we can see, independent learning mode is not suitable for individuals in an MAS because of uncertain impact on state transit caused by unpredictable neighboring agents' behaviors, which results in failure of efficacy of static local environment information at current moment. As can be seen from Table 7, in terms of average waiting time, the CGB-MAQL strategy performs 78.84% better than the Fixed approach as 24.2903 s It shows 95.69% better than the CGB-IQL strategy, and 86.64% better

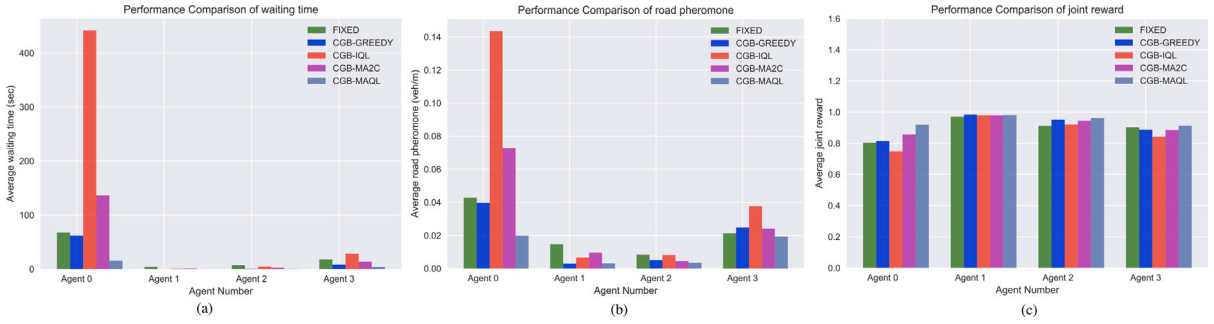

Fig. 20. Bar charts on average level of 3 metrics for all agents of Harbin scenario. (a) Average waiting time (lower is better), (b) Average road pheromone (lower is better), (c) Average joint reward (higher is better).

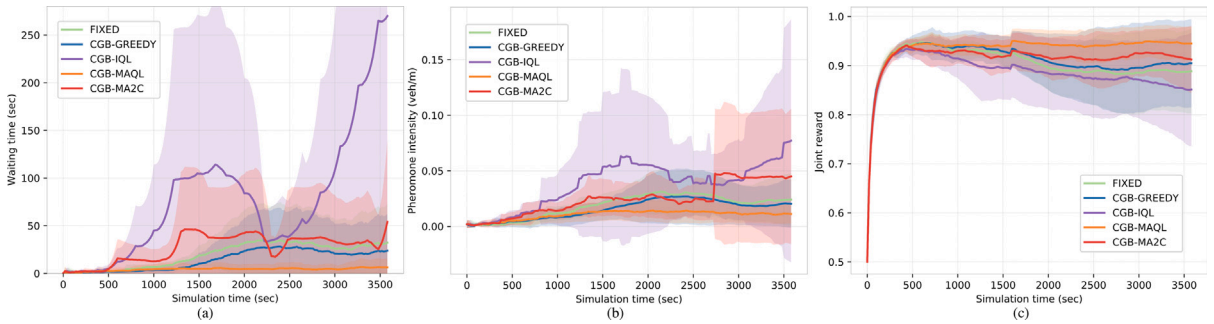

Fig. 21. Plots of average of various metrics of all four agents in all strategies of Harbin scenario. (a) Average waiting time (s) (b) Average pheromone intensity (veh/m). (c) Average joint reward.

than the CGB-MA2C strategy respectively. In terms of road pheromones, CGB-MAQL achieves the lowest average level as 0.0115 veh/m with 47.25% lower then FIXED strategy, 36.46% lower than CGB-Greedy and 58.63% lower than CGB-MA2C respectively. All results indicate that agents under the CGB-MAQL algorithm can effectively learn to promote intersection traffic flow. Both of CGB-MAQL and the CGB-Greedy algorithm can significantly reduce vehicles' average waiting time. Still, the pheromone index indicates that our algorithm is superior to the CGB-Greedy algorithm on road congestion.

Fig. 21(a)–(c) provides a general view of all agents' behavior on average waiting time, average road pheromone and joint reward during an episode. From Fig. 21(a), CGB-MAQL outperforms other policies with obvious decrease of average waiting time. Performance of CGB-GREEDY also shows effective control on four agents with greedy behavior mode. Although agents are autonomously trying to converge in CGB-IQL strategy, they finally fail to achieve initial target due to lack of effective observation of states' transit in neighboring agents. In addition the poor performance of CGB-MA2C proves the infeasibility of policy-based RL methods on discrete action space with low dimensions. Fig. 21(b) and (c) gives same rank among tested strategies, where CGB-MAQL outperforms others in both average road pheromone and average joint reward level among all agents, together with CGB-IQL performs the worst.

Interestingly, the CGB-IQL strategy shows a significant standard deviation on all indicators, which is even worse than the FIXED strategy. The instability is because that each agent in CGB-IQL only concentrates on local optimal policy learning and ignores changes in the environmental status of other agents' behaviors, resulting in poor scalability caused by inaccurate records of executed actions. The difference between CGB-IQL and CGB-MAQL illustrates the importance of the collaboration in agents' learning process to the portability and scalability of a MARL system. In addition, performance of CGB-GREEDY proves that even with greedy behavior mode, CGB-MAQL still can optimize traffic efficiency over other policies.

5.6. Evaluation

In summary, we trained all algorithms in a synthetic road network scenario. The test results show that our CGB-MAQL can reduce vehicle travel time, ease road congestion, and improve road network efficiency. The cumulative return and the stable performance of all agents under the synthetic road network scenario prove the stability and effectiveness of CGB-MAQL and its convergence. To verify our proposed algorithm's portability and scalability, we tested it in the Monaco and Harbin city scenarios. Compared with the config scenario, the number of intersections in the urban scenario is greatly increased. However, our algorithm still maintains its superiority in various indicators, with all agents keeping good adaptability and scalability for large-scale road networks. As a result, the proposed algorithm remains superior in improving the efficiency of the road network compared with other algorithms.

One of the advantages of our algorithm is improving agents' training efficiency via regional green wave control. The joint state and joint return design based on the three-layer hierarchical architecture help to deepen the degree of collaboration among agents in the MAS environment. Besides, with the expansion of the road network structure, the increase of the optimum behavior policy's computational cost will not bring the surge of the dimension for action space like that in the traditional MARL algorithm. As the A2C strategy always tends to perform better than DQN on other issues, it fails to over-perform the extension of DQN in the extended algorithm of the architecture proposed in this article. It is due to the characteristics of the problem. In this paper, to achieve a more scalable learning algorithm for the large-scale road network, the regional green wave control supported by the three-layer communication architecture based on CVIS and multi-agent RL is combined, which reduces the size of the action space. Also, to improve the agents' learning efficiency and reduce the computational cost of the training model, we make a discrete state of the large road network. However, the essence of A2C is the combination of RL based on value function and policy gradient, which tends to give better performance dealing with problems with large state space and continuous actions.

6. Conclusion

In this article, we have presented a framework named Cooperative Group-Based Adaptive Traffic Signal Control (CGB-MATSC) for the large-scale intelligent traffic light control in realistic urban scenarios. In previous research works the deep reinforcement learning (DRL) has shown its effectiveness in improving road efficiency compared to other traditional traffic light control methods in a single intersection. However, in the process of expanding DRL from a single agent to a multi-agent scenario, the degree of collaboration among agents significantly affect the training process. Therefore, we have also proposed a Cooperative Group-Based Multi-Agent Q Learning (CGB-MAQL) algorithm for large-scale intelligent traffic light control in realistic urban scenario. The proposed algorithm is based on the principle of multi-agent reinforcement learning (MARL). Supported by vehicle-to-everything (V2X) communication, a group-based configuration of agents is employed, where each agent is responsible for controlling the traffic lights in its region.

An agent communicates with the road side units (RSU), which are able to set the current phase sequences of the traffic lights. To perform coordination mechanism among neighboring agents, a k-nearest-neighbor-based joint state representation is proposed for each agent, by re-organizing local states received from all agents that are transmitted via Cooperative Vehicle Infrastructure System (CVIS). The CVIS is based on a hierarchical communication network. In order to take spatial relationship into consideration among agents, a distance-factored joint reward representation is designed to incorporate the impacts of other agents' behaviors, which helps to avoid unfair influence among all agents. In order to accelerate the training process, we particularly propose a heuristic training mechanism. With clear threshold values, agents perform autonomous targeted learning by terminating poor behavior strategies that may result in long waiting queue at an intersection in the exploring process.

To assess the effectiveness of proposed CGB-MAQL strategy, we integrate Multi-agent Advantage Actor–Critic (MA2C) method as CGB-MA2C, Independent Q-Learning (IQL) as CGB-IQL and Deep Q-Network (DQN) with greedy behavior policy as CGB-GREEDY (which is also the greedy mode of CGB-MAQL), in order to obtain a deep insight on various DRL-driven control strategies. Average waiting time and road pheromone under each strategy are analyzed on both grid road network and Monaco urban scenario, where the proposed CGB-MAQL shows effective coordination among agents by improving traffic efficiency. The comparative analysis shows the superiority of the proposed CGB-MAQL on CGB-IQL. It also outperforms CGB-GREEDY and CGB-MA2C with better exploring policies. Future work will focus on the application of MARL in large-scale road networks combined with the vehicle route guidance system. More in-depth research on taking advantage of green wave control mode with MARL will be conducted to access more scalable algorithms with low training costs and more flexibility on in large-scale city scenario.

CRediT authorship contribution statement

Tong Wang: Data curation, Writing - original draft, Supervision. **Jiahua Cao:** Conceptualization, Methodology, Software, Data curation, Writing - original draft, Validation, Writing - review & editing. **Azhar Hussain:** Conceptualization, Methodology, Software, Data curation, Writing - original draft, Visualization, Investigation, Validation.

Acknowledgments

This paper was supported by the National Natural Science Foundation, China (61102105, 51779050), the financial support from the National Key Research and Development Program of China (2016YFB0700100). The Harbin Science Fund for Young Reserve Talents, China (No. 2017RAQXJ036), Fundamental Research Funds for the Central Universities, China (HEUCFG201831).

Appendix

Definition 1. In a stochastic game with N agents, metric space $\mathbb{R}^N$ and sufficiently high temperature, if each agent has a binary action space, the procedure converges using pairwise interaction $Q^i(s, a^i, \vec{a}^i)$ following the Boltzmann policy.

Let $\boldsymbol{a} = [a^1, \ldots, a^N]$, $\boldsymbol{b} = [b^1, \ldots, b^N]$, and the distance $d(\boldsymbol{a}, \boldsymbol{b}) = \sum_i |a^i - b^i|$. Let the Q-function is K-Lipschitz continuous w.r.t. a^i, then the operator $\mathfrak{F}(a^i) \triangleq \pi^i(a^i|s, \bar{a}^i)$ in Eq. (28) under sufficiently low temperature β forms a contraction mapping. From the contraction mapping theorem (Kreyszig, 1978), the operator $\mathfrak{F}$ has to follow:

$$d(\mathfrak{F}(\boldsymbol{a}), \mathfrak{F}(\boldsymbol{b})) \leq \alpha d(\boldsymbol{a}, \boldsymbol{b}), \forall \boldsymbol{a}, \boldsymbol{b}. \tag{30}$$

where $0 \leq \alpha \leq 1$, and $\mathfrak{F}(a^i)$ for binomial case is given by:

$$\mathfrak{F}(a^i) = \frac{\exp(-\beta Q_i^i(s, a^i, \bar{a}^i))}{\exp(-\beta Q_i^i(s, a^i, \bar{a}^i)) + \exp(-\beta Q_i^i(s, a^{i'} \bar{a}^i))}. \tag{31}$$

Let $\Delta Q(s, a^i, \bar{a}) = Q(s, a^{i'}, \bar{a}) - Q(s, a^i, \bar{a})$, then $\mathfrak{F}(a^i)$ can be written as:

$$\mathfrak{F}(a^i) = \frac{1}{1 + \exp(-\beta \cdot \Delta Q(s, a^i, \bar{a}))}, \tag{32}$$

we express $\mathfrak{F}(a^i) - \mathfrak{F}(b^i)$ as $\Delta\mathfrak{F}$ and following the mean value theorem $x_0 \in [x_1, x_2]$, s.t., $f(x_1) - f(x_2) = f'(x_0)(x_1 - x_2)$, the $\Delta\mathfrak{F}$ is given by:

$$|\Delta\mathfrak{F}| = |\frac{\beta e^{-\beta \Delta Q_o}}{(1 + e^{-\beta \Delta Q_o})^2} \parallel \Delta Q(s, a^i, \bar{a}) - \Delta Q(s, b^i, \bar{a})|, \tag{33}$$

when $e^{-\beta \Delta Q_0} = 1$, i.e., $\Delta Q_0 = 0$ we get $1/4$ as the maximum value of $e^{-\beta \Delta Q_o}/(1 + e^{-\beta \Delta Q_o})^2$, and the expression becomes:

$$|\Delta\mathfrak{F}| \leq \frac{1}{4T} |Q(s, a^{i'}, \bar{a}) - Q(s, b^{i'}, \bar{a}) + Q(s, b^i, \bar{a}) - Q(s, a^i, \bar{a})|. \tag{34}$$

By applying the Lipschitz constraint, where constant $K \geq 0$, the expression simplifies as:

$$|\Delta\mathfrak{F}| \leq \frac{1}{4T} (K|1 - a^i - (1 - b^i)| + K|a^i - b^i|), \tag{35}$$

further reducing it:

$$|\Delta\mathfrak{F}| \leq \frac{1}{4T} 2K \sum_i |a^i - b^i|, \tag{36}$$

therefore, expression for $d(\mathfrak{F}(\boldsymbol{a}), \mathfrak{F}(\boldsymbol{b}))$ can be written as:

$$d(\mathfrak{F}(\boldsymbol{a}), \mathfrak{F}(\boldsymbol{b})) \leq \frac{K}{2T} \sum_i |a^i - b^i|, \tag{37}$$

as $\sum_i |a^i - b^i| = d(\boldsymbol{a}, \boldsymbol{b})$ the expression becomes:

$$d(\mathfrak{F}(\boldsymbol{a}), \mathfrak{F}(\boldsymbol{b})) = \frac{K}{2T} d(\boldsymbol{a}, \boldsymbol{b}) \quad \blacksquare \tag{38}$$

In order for the contraction to hold, the $T > K/2$, which proves that when the action space is binary for each agent, and the temperature is sufficiently large, and guarantees the convergence.

References

Bakker, B., Whiteson, S., Kester, L., 2010. Traffic light control by multiagent reinforcement learning systems. Comput. Sci.

Cases, N., 2017. Deep deterministic policy gradient for urban traffic light control, paper. Online. Available: https://arxiv.org/abs/1703.09035v1.

Chu, T., Wang, J., Lara, C., 2020. Multi-agent deep reinforcement learning for large-scale traffic signal control. IEEE Trans. Intell. Transp. Syst.

Crisostomi, E., Kirkl, S., Shorten, R., 2011. A Google-like model of road network dynamics and its application to regulation and control. Internat. J. Control 84 (3), 633–651.

Friedman, J.H., 1997. On bias, variance, 0/1-loss, and the curse-of-dimensionality. Min. Knowl. Discov.

Ge, H., Song, Y., Wu, C., 2019. Cooperative deep Q-learning with Q-value transfer for multi-intersection signal control. IEEE Access.

Grover, Aditya, Al-Shedivat, Maruan, Gupta, Jayesh K., Yura, Edwards, Harrison, 2018. Learning Policy Representations in Multiagent Systems. ICML.

Hausknecht, M., Stone, P., 2015. Deep recurrent Q-learning for partially observable MDPs. Comput. Sci.

Hunt, P.B., Robert, D.I., Bretherton, R.D., 1982. The SCOOT on-line traffic signal optimization technique. Traffic Eng. Control.

Jin, J., Ma, X., 2019. A multi-objective agent-based control approach with application in intelligent traffic signal system. IEEE Trans. Intell. Transp. Syst.

K.J.1, Prabuchandran, A.N1, Hemanth Kumar, Bhatnagarl, Shalabh, 2015. Decentralized learning for traffic signal control. In: Intelligent Transportation System Workshop.

Kok, Jelle R., Vlassis, Nikos, 2005. Using the max-plus algorithm for multiagent decision making in coordination graphs. In: The 9th International RoboCup Symposium.

Krajzewicz, Daniel, Erdmann, Jakob, Behrisch, Michael, Bieker, Laura, 2012a. Recent development and applications of SUMO - simulation of Urban MObility. Int. J. Adv. Syst. Meas.

Krajzewicz, D., Erdmann, J., Behrisch, M., Bieker, L., 2012b. In: Recent development and applications of SUMO - simulation of Urban mobil- ity. Int. J. Adv. Syst. Meas. 5 (3 and 4), 128–138.

Kreyszig, E., 1978. Introductory Functional Analysis with Applications, Vol. 1. Wiley New York.

Liang, X., Du, X., Wang, G., Han, Z., 2019. A deep reinforcement learning network for traffic light cycle control. IEEE Trans. Veh. Technol. 68 (2), 1243–1253.

Liu, Bojin, Khorashadi, Behrooz, Ghosal, Dipak, Chen-NeeChuah, H. MichaelZhang, 2012. Analysis of the information storage capability of VANET for highway and city traffic. Transp. Res. C.

Lowe, Ryan, 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. Comput. Sci.

Marco, W., Veenen, J.V., Jilles, V., 2004. Intelligent traffic light control. In: Proceedings of the National Academy of ences of the United States of America.

Mnih, V., Badia, Adri Puigdomnech, Mirza, M., 2016. Asynchronous Methods for Deep Reinforcement Learning. ICML.
Mnih, Volodymyr, Kavukcuoglu, Koray, Graves, Alex, 2015. Playing atari with deep reinforcement learning. Comput. Sci.
Mnih1, Volodymyr, Kavukcuoglu1, Koray, Silver1, David, 2015. Human-level control through deep reinforcement learning. Nature.
Moosavi, V., Hovestadt, L., 2013. Modeling urban traffic dynamics in coexistence with urban data streams. In: Proc. of the 2nd ACM SIGKDD International Workshop on Urban Computing.
Nagabandi, A., Kahn, G., Fearing, R.S., 2017. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. Comput. Sci.
Nicols, C., Carlos, A., 2013. Dyna-2.
Osorio, Carolina, Wang, Carter, 2017. On the analytical approximation of joint aggregate queue length distributions for traffic networks: A stationary finite capacity Markovian network approach. Transp. Res. B 95, 305–339.
Pandit, K., Ghosal, D., Zhang, H.M., Chuah, C.-N., 2013. Adaptive traffic signal control with vehicular ad hoc networks. IEEE Trans. Veh. Technol. 62 (4), 14591471.
Peters, J., Schaal, S., 2008. Reinforcement learning of motor skills with policy gradients. Neural Netw.
Qi, Xuewei, Luo, Yadan, Wu, Guoyuan, Boriboonsomsin, Kanok, Barth, Matthew, 2019. Deep reinforcement learning enabled self-learning control for energy efficient driving. Transp. Res. C.
Qu, Z., Pan, Z., Chen, Y., Wang, X., Li, H., 2020. A distributed control method for urban networks using multi-agent reinforcement learning based on regional mixed strategy nash-equilibrium. IEEE Access.
Rashid, T., Samvelyan, M., Witt, D., 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. ICML.
Recker, W.W., Ramanathan, B.V., Yu, X.-H., McNally, M.G., 1995. Markovian real-time adaptive control of signal systems. Math. Comput. Modelling 22 (47), 355–375.
Rezzai, M., Dachry, W., Moutaouakkil, F., 2018. Design and realization of a new architecture based on multi-agent systems and reinforcement learning for traffic signal control. In: International Conference on Multimedia Computing Systems.
Riedmiller, M., Hafner, R., Lampe, T., 2018. Learning by playing - solving sparse reward tasks from scratch. Comput. Sci.
Schlote, A.C., 2014. New perspectives on modelling and control for next generation intelligent transport systems.
Schneider, Jeff, Wong, Weng-Keen, Moore, Andrew, Riedmiller, Martin, 1999. Distributed value functions, conference paper. In: Proceedings of International Conference on Machine Learning.
Schulman, John, Moritz, Philipp, Levine, Sergey, Jordan, Michael, Abbeel, Pieter, 2015. High-dimensional continuous control using generalized advantage estimation. Comput. Sci.
Stanley, H.E., 1971. Phase Transitions and Critical Phenomena. Clarendon, Oxford, p. 9.
Sutton, R.S., Mcallester, D., Singh, S., 1999. Policy gradient methods for reinforcement learning with function approximation. Comput. Sci.
Sutton, R.S., Mcallester, D., Singh, S., 2000. Policy gradient methods for reinforcement learning with function approximation. Neural Inf. Process. Syst.
Tan, T., Bao, F., Deng Y, Y., 2019a. Cooperative deep reinforcement learning for large-scale traffic grid signal control. IEEE Trans. Cybern.
Tan, T., Chu, T.S., Peng, B., 2018. Large-scale traffic grid signal control using decentralized fuzzy reinforcement learning. In: Proceedings of SAI Intelligent Systems Conference.
Tesauro, G., 2003. Extending Q-learning to general adaptive multi-agent systems. Adv. Neural Inf. Process. Syst.
Thorpe, T.L., 1997. Vehicle traffic light control using SARSA. Available: citeseer.ist.psu.edu/thorpe97vehicle.html.
Tsitsiklis, J.N., Van Roy, B., 1997. An analysis of temporal-difference learning with function approximation. IEEE Trans. Automat. Control.
Varaiya, Pravin, 2013. Max pressure control of a network of signalized intersections. Transp. Res. C 36, 177–195.
Vidhate, D.A., Kulkarni, P., 2017. Cooperative multi-agent reinforcement learning models (CMRLM) for intelligent traffic control. In: 2017 1st International Conference on Intelligent Systems and Information Management. ICISIM.
Wang, Z., Bapst, V., Heess, N., 2016. Sample efficient actor-critic with experience replay. Comput. Sci.
Wang, Yuan, Zhang, Dongxiang, Liu, Ying, Dai, Bo, Lee, Loo Hay, 2019. Enhancing transportation systems via deep learning: A survey. Transp. Res. C.
Wiering, Marco, van Veenen, Jelle, Vreeken, Jilles, 2004. Intelligent traffic light control. In: Proceedings of the National Academy of ences of the United States of America.
Ye, B.L., Wu, W., Zhou, X., 2014. A green wave band based method for urban arterial signal control. In: IEEE International Conference on Networking.
Yu, X., Stubberud, A.R., 1997. Markovian decision control for traffic signal systems. In: Proceedings of the 36th Conference on Decision and Control San Diego, California USA.
Zhang, Z., Zhao, D., 2014. Clique-based cooperative multiagent reinforcement learning using factor graphs. IEEE/CAA J. Autom. Sin.
Zhao, X., Spall, J.C., 2018. A markovian framework for modeling dynamic network traffic. In: 2018 Annual American Control Conference. ACC. Milwaukee, WI. pp. 6616–6621.