Learning scalable multi-agent coordination by spatial differentiation for traffic signal control

Junjia Liu [a,1], Huimin Zhang [b,1], Zhuang Fu [a,*], Yao Wang [a]

[a] *State Key Laboratory of Mechanical System and Vibration, Shanghai Jiao Tong University, Shanghai, 200240, China*
[b] *School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China*

A R T I C L E   I N F O

A B S T R A C T

The intelligent control of the traffic signal is critical to the optimization of transportation systems. To achieve global optimal traffic efficiency in large-scale road networks, recent works have focused on coordination among intersections, which have shown promising results. However, existing studies paid more attention to observations sharing among intersections (both explicit and implicit) and did not care about the consequences after decisions. In this paper, we design a multi-agent coordination framework based on Deep Reinforcement Learning method for traffic signal control, defined as γ-*Reward* that includes both original γ-*Reward* and γ-*Attention-Reward*. Specifically, we propose the *Spatial Differentiation* method for coordination which uses the temporal–spatial information in the replay buffer to amend the reward of each action. A concise theoretical analysis that proves the proposed model can converge to Nash equilibrium is given. By extending the idea of Markov Chain to the dimension of space–time, this truly decentralized coordination mechanism replaces the graph attention method and realizes the decoupling of the road network, which is more scalable and more in line with practice. The simulation results show that the proposed model remains a state-of-the-art performance even not use a centralized setting. Code is available in https://github.com/Skylark0924/Gamma_Reward.

1. Introduction

Traffic congestion has been an increasingly critical matter for decades. It not only leads to an increase in commuting time but also exacerbates noise and environmental pollution issues due to frequent acceleration and deceleration. According to relevant researches, almost all collisions and delays in urban traffic are concentrated on intersections (To and Barker, 2001). Unreasonable signal control significantly leads to a waste of traffic resources. Therefore, the key to solve urban congestion is to keep the intersection clear.

Deep Reinforcement Learning (DRL) methods have been well applied in the traffic signal regulation of single-intersection and shown a better performance than traditional methods (Liang et al., 2018; Wei et al., 2018, 2019c), such as Max-pressure (Varaiya, 2013). Recent works began to try to apply DRL algorithms, especially multi-agent Reinforcement Learning (MARL) (Buşoniu et al., 2010), to multi-intersection and even large-scale road networks. Different from the single-intersection problem, the intelligent regulation of large-scale road networks needs to achieve synergistic control between various intersections which can be regarded as a Multi-objective Optimization Problem (MOP) and Markov/Stochastic Games (MGs) with Cooperative Setting (Zhang et al., 2019b). In other words, multiple agents need

to coordinate with each other. They need to keep their intersection open, and at the same time, pay attention to the traffic flow status of surrounding or even remote intersections, so that they can ultimately improve the efficiency of the overall road network. The latest research introduced the graph attention network (GAT) to share the observations of real-time traffic volume implicitly with each other, and get an inspired result (Wei et al., 2019b).

1.1. Related work and motivation

The existing traffic signal control (TSC) methods can be divided into two categories: rule-based methods and learning-based methods. The former transforms the problem into a rule-based optimization problem; the later one seeks control strategy from the traffic flow data.

For the rule-based methods, such as Webster (Koonce and Rodegerdts, 2008), GreenWave (Török and Kertész, 1996) and Max-pressure (Varaiya, 2013), a traffic signal optimization problem is usually solved under some assumptions like a preset period or fixed cycle-based phase sequence (Wei et al., 2019c). Webster is used for an isolated intersection and is a widely-used method in TSC. It assumes that the traffic flow is uniform during a certain period and constructs a

closed-form equation to generate the optimal cycle length and phase split for a single intersection that minimizes the travel time of all vehicles passing the intersection. GreenWave is a classical method in the transportation field to implement coordination, which aims to optimize the offsets to reduce the number of stops for vehicles traveling along one specific direction. Max-pressure aims to reduce the risk of over-saturation by balancing queue length between adjacent intersections and minimizing the "pressure" of the phases for an intersection. However, the unpractical assumptions in these methods might not lead to excellent performance.

Recently, the DRL technique, as a popular learning-based method, has been proposed to control traffic signals due to its capability of online optimization without prior knowledge about the given environment. At present, DRL has been successfully applied to the single-intersection regulation of traffic signal (Wei et al., 2018) by regarding the intersection as an agent. The results of various state-of-the-art DRL algorithms are compared in Mousavi et al. (2017), showing that Deep Q-Networks algorithm is more suitable for the solution of TSC tasks. However, the problem with multiple intersections is still a frontier.

Existing MARL algorithms focus on collaboration among agents and can be divided into centralized and decentralized setting according to various information structures (Zhang et al., 2019b). Independent reinforcement learning (IRL), as a fully decentralized setting, directly perform a DRL algorithm for each agent with neither explicit nor implicit information exchange with each other. This method has been applied in multi-intersection TSC problem (Xiong et al., 2019; Zheng et al., 2019). However, the environment is shared in MARL, and it changes with the policy and state of each agent (Foerster et al., 2016). For one of the agents, the environment is dynamic and non-stationary, leads to convergence problems. Tan (1993) compares IRL with Value Decomposition Networks (VDN) and illustrates the disadvantages of IRL. As a centralized method, there exists a central controller in VDN which integrates the value function of each agent to obtain a joint action–value function. The integration strategy is to add them directly. Moreover, QMIX (Rashid et al., 2018), as a extend of VDN, uses state information and integrates them in a nonlinear way and gets a stronger approximation ability than VDN. Both QMIX and VDN are typical centralized MARL algorithms with communication, and this joint-action idea has been already used in TSC (Van der Pol and Oliehoek, 2016).

Based on these centralized methods, recent TSC studies condense the global scope into a smaller neighborhood (Wei et al., 2019b; Nishi et al., 2018; Wei et al., 2019a) and use graph convolution network (GCN) to achieve coordination. Colight (Wei et al., 2019b) introduces the concept of attention mechanism and realized cooperation by integrating observations in a neighborhood implicitly into a hidden state. However, as mentioned in Colight, the neighborhood scope is a constant, so the traffic information among intersections cannot be utilized to determine the range of the neighborhood. Due to the usage of GCN and Multi-head Attention techniques, these methods still need to gather information for centralized computing and betray the intention of distribution.

In fact, such a central controller does not exist in many applications, apart from those that can easily have a global perspective, like video games (Zhang et al., 2019b). Considering the demand for the scalability in TSC, setting a central controller is impractical. Therefore, we need to seek a compromise solution which is both convergent and scalable. This setting is referred to as *a decentralized one with networked agents* (Zhang et al., 2019b).

1.2. Main contributions

In this paper, we first regard each intersection as a DRL agent and transform the TSC problem into a Markov Decision Process (MDP). Unlike existing work, this paper is aiming at improving method scalability while ensuring a SOTA performance. To achieve this goal, we introduce a structural prior about road networks as an inductive bias

and extend the Markov Chain theory to the temporal–spatial domain for the coordination.

The change of future states and rewards from distant intersections is multiplied by the *spatial discount rate* γ (Note that it represents the temporal–spatial discount on multi-agent information, not the ordinary meaning used in temporal difference, and it will be distinguished in detail later) and taken into account when learning the current intersection policy. This information is used as a penalty to correct the calculation of current rewards so that the agents have the ability to collaborate. Due to the various traffic volume of each road, the influence of surrounding intersections may be different. Therefore, the attention mechanism is introduced in this paper to correct the influence weight of surroundings on the current intersection.

To summarize, our main contributions are as follows:

- We propose a coordination framework, defined as *γ-Reward*, which can communicate with adjacent intersections and even further in a scalable way by sharing future states and rewards and achieve global optimal control of the TSC problem.
- Instead of Multi-head Attention, the *spatial differentiation* method is proposed to collect the temporal–spatial information in a decentralized way and amend the current reward by recursion.
- We just use attention in the neighborhood for distinguishing various significance, and update the attention score parameters in *spatial differentiation* formula by imitating the idea of the target network.
- It is found in the test results of various road networks that the *γ-Reward* series, including original *γ-Reward* and *γ-Attention-Reward*, maintain a SOTA performance while achieving a better scalability.

2. Problem formulation & proposed MARL method

In this section, we introduce the basic knowledge of the TSC problem and propose our MARL method.

2.1. Preliminary & formulation

- **Lane:** Lane is part of a roadway that is designated to be used by a single line of vehicles (Wikipedia contributors, 2019). There are two kinds of lanes: entering lane and exiting lane (Stevanovic, 2010). Each intersection consists of multiple lanes.
- **Phase:** A phase is a combination of movement signals (Wei et al., 2019c). Fig. 1(a) shows eight main directions of vehicles at the intersection. Note that the direction of turning right is usually ignored in these problems since it can execute every time without caring for the traffic signal. The directions in the same phase need not be a conflict which is shown in Fig. 1(b). Phase is the unit of TSC, and only one phase can turn green at a time.
- **Neighbor intersection:** The intersections which directly connect to the current intersection. In an informal road network, each intersection usually has at most four neighbors.
- **Waiting vehicle:** If a vehicle on an entering lane has a speed lower than a threshold, then we define it as a *waiting vehicle*, which means it is slowing down to wait for the red light.

By using DRL, we regard the TSC problem as MDP (see Fig. 2). An individual DRL agent controls one of the intersections. They need to observe part of the environmental state $\mathcal{O}$ and get actions $\mathcal{A}$ according to these observations to determine which phase in the intersection needs to turn green. The effect of control is fed back from the environment in the form of reward $\mathcal{R}$. The goal of the DRL agent is to maximize the reward function by continuously exploring and exploiting based on constant interaction with the environment. In this paper, the problem requests to reduce the length of the queue q_l or the travel time T_w in the road network. To make this problem more suitable for DRL, we can first abstract it into these parts $\langle \mathcal{O}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \pi, \gamma \rangle$:
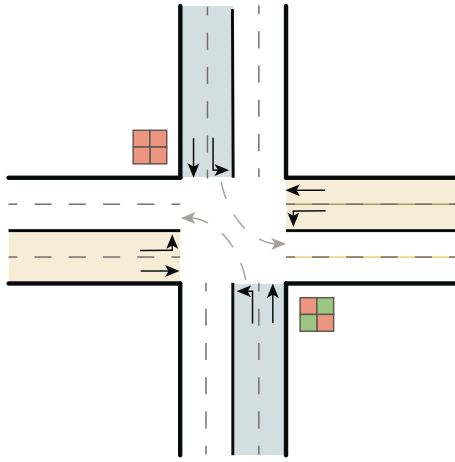
(a) Eight main directions in a single intersection
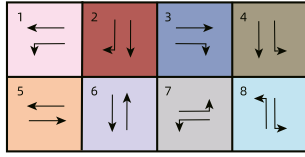
(b) Eight kinds of primary phases

Fig. 1. The foundation of the traffic problem; Phase is the fundamental unit of the TSC problem. The two directions in each phase never conflict.
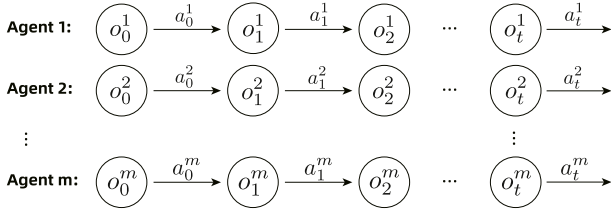

Fig. 2. The TSC problem is regarded as MDP. Each intersection is controlled by a unique agent that can implement a DRL algorithm and gain an optimal strategy of action decision. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- **Observation** o_t^i: $o_t^i = (o_1^i, \ldots, o_t^i)$, where $o_t^i \in \mathcal{O}_t^i$. Every agent observes the length of the vehicle queue on entering lanes of their intersection. Moreover, to cater to the design of the proposed γ-*Reward* algorithm, we also need to observe the number of vehicle on the exiting lanes which connect to neighbor intersections.
- **Action** a_t^i: $\boldsymbol{a}_t^i = (a_1^i, \ldots, a_t^i)$, where $\boldsymbol{a}_t^i \in \mathcal{A}_t^i$. Action can be easily set as the serial number of phase which is chosen to be green.
- **Transition probability** $\mathcal{P}$: $\mathcal{P}(o_{t+1}{}^i | o_t{}^i, a_t{}^i)$ describes the probability from state $o_t{}^i$ to the next potential state $o_{t+1}{}^i$.
- **Reward** r_t^i: After executing each action a_t^i, we can get a return information to judge whether a_t^i is good enough for o_t^i. We use the number of waiting vehicle on the entering lanes as a raw reward. For amendatory reward, we use R_t^i as a representation.
- **Policy** π: Policy is what agents need to learn in DRL. It represents the goal of reducing travel time and increasing average speed. For a single agent, $\pi^i : \mathcal{O}_t^i \mapsto \mathcal{A}_t^i$.
- **Discount rate** γ': This factor is the common meaning used in Temporal-Difference (see Appendix A.1.1). To avoid confusion with γ-*Reward* , we use γ' here to replace the original symbol γ.

By using the Bellman equation, the relationship among these parameters can be formulated. We can gain the optimal phases after iterating
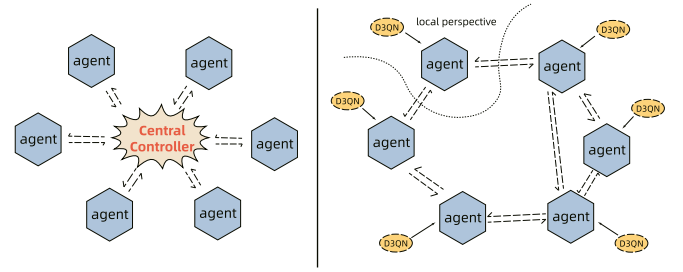

Fig. 3. Left: MARL algorithms with centralized setting. Right: γ-Reward framework which is a decentralized one with network agents. γ-Reward contains a coordination mechanism proposed based on D3QN.
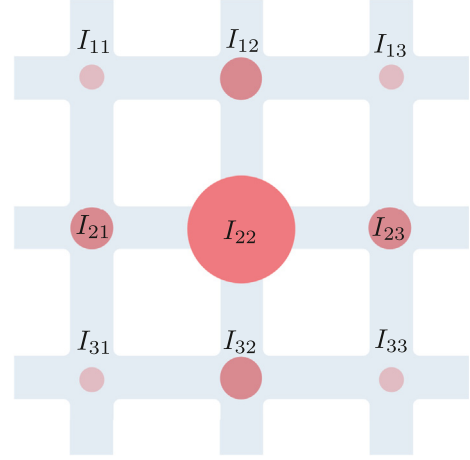

Fig. 4. Diagram for spatial differentiation.

these equations:

$$Q(o_t^i, a_t^i) = Q(o_t^i, a_t^i) + \alpha(r_t^i + \gamma' \max_{a_{t+1}^i} Q(o_{t+1}^i, a_{t+1}^i) - Q(o_t^i, a_t^i)) \quad (1)$$

$$a_t^i = \arg\max Q(o_t^i, a_t^i)$$

2.2. Proposed spatial differentiation function

Coordination among agents plays a critical role in MARL algorithms, either centralized or decentralized. In this paper, we propose a coordination mechanism among distributed agents. Each agent is based on Dueling-Double-Deep Q Network (D3QN) (Mnih et al., 2015; Van Hasselt et al., 2016; Wang et al., 2016), which is one of the best Q value-based model until now (Hessel et al., 2018). A detailed description of it can be found in Appendix A.1. We found that some studies already use D3QN directly on the TSC problem (Liang et al., 2019), but it can only be used as an independent Q-learning (IQL) method in the multi-intersection problem. For the TSC problem, the idea of distributed agents is wise and what we need is an appropriate decentralized coordination mechanism. Therefore, we propose γ-*Reward* framework for coordination (Fig. 3).

The basic theory of DRL inspires the main idea of this article. For the TSC problem, not only the temporal decision should be regarded as MDP, the road network itself is more like a Markov Chain since the decision of the current intersection will affect other intersections in the future. So it should consider not only the TD-error (see Appendix A.1.1) in time, but also a "TD-error" in space which is defined as the *spatial differentiation*.

Fig. 4 shows a diagram of multi-intersection road network with 3×3 intersections. Since the decision of intersection I_{22} at time t will affect the next intersection I_{23} at time $t + n$, the result of intersection I_{22} at time t should be affected not only by the current intersection reward

$r_t^{I_{22}}$, but also by the reward r_{t+n}^j of the surrounding intersections at time $t + n$, where $j \in I_{12}, I_{21}, I_{23}, I_{32}$. The formula of *spatial differentiation* is as follows:

$$R_t^{I_{22}} = r_t^{I_{22}} \cdot \left\{ 1 + \gamma \cdot \tanh \left[\sum_{j \in \mathcal{N}_i} \left(\frac{R_{t+n}^j}{r_t^j} - c \right) \right] \right\} \tag{2}$$

where $\mathcal{N}_{I_{22}} = \{I_{12}, I_{21}, I_{23}, I_{32}\}$ represents the four intersections around intersection I_{22}. The parameter n is called *delay span* which represents the time span of most vehicles reach the next intersection after current action a_t. It is related to the length of the road, the average velocity on the exiting lane and sample interval. The whole formula can also be treated as value function of reward, thus it is a *spatial differentiation* of *reward-value function*.

In Eq. (2), R_{t+n}^j / r_t^j shows the change of traffic capacity at intersection j between time t and $t + n$. If R_{t+n}^j / r_t^j is greater than threshold c, it indicates that the traffic capacity of the intersection j is deteriorated, in other words, the decision of the intersection I_{22} at the time t will cause the adjacent intersection j to be more congested; conversely, R_{t+n}^j / r_t^j less than c indicates that the capacity of intersection j is improved, and the decision of the intersection I_{22} at time t is good for j. Through some region transformations, the differential item is served as a penalty. It finally multiplies by a spatial discount factor γ as an amendment to reward. Since the training goal of DRL is to maximize the reward function, the existence of the penalty item forces the agent to pay attention to the situation around the intersection while improving its own policy.

This equation is also in line with the idea of MARL with networked agents which dates back to Varshavskaya et al. (2009). QD-learning (Kar et al., 2013) further gives a convergence proof of this kind of MARL algorithm. It incorporates the idea of *consensus + innovation* to the standard Q-learning algorithm and has the following Q-value update equation:

$$\begin{aligned} Q(o_t^i, a_t^i) \leftarrow {} & Q(o_t^i, a_t^i) \\ & + \alpha_{t,o,a} \left[r_t^i + \gamma' \max_{a_{t+1}^i \in \mathcal{A}} Q(o_{t+1}^i, a_{t+1}^i) - Q(o_t^i, a_t^i) \right] \\ & - \beta_{t,o,a} \sum_{j \in \mathcal{N}_t^i} \left[Q(o_t^i, a_t^i) - Q(o_t^j, a_t^j) \right] \end{aligned} \tag{3}$$

where the term after $\alpha_{t,o,a}$ denotes a local *innovation* potential that incorporates newly obtained observations, and the term after $\beta_{t,o,a}$ is a *consensus* potential (agent collaboration) which captures the difference of Q-value estimates from its neighbors.

Analogously, our Q-value update equation can be changed as follows:

$$\begin{aligned} Q(o_t^i, a_t^i) = {} & Q(o_t^i, a_t^i) + \alpha \{ r_t^i + \gamma \cdot r_t^i \cdot \tanh [\sum_{j \in \mathcal{N}_i} (\frac{R_{t+n}^j}{r_t^j} - c)] \\ & + \gamma' \max_{a_{t+1}^i \in \mathcal{A}} Q(o_{t+1}^i, a_{t+1}^i) - Q(o_t^i, a_t^i) \} \end{aligned} \tag{4}$$

The spatial differentiation term denotes the consensus like QD-learning and we further construct the communication in the temporal domain and the spatial domain at the same time. Although Eq. (2) uses future information which sounds contrary to the causality, it is achievable in the programming. Since we use Q-learning as a basic model which is off-policy, it saves the trajectory of state–action pair and the obtained reward into a replay buffer. In this way, the raw reward r_t^i is saved first and then corrected after n steps. Therefore the calculation process can be realized (see Appendix A.2 for pseudocode).

2.3. Attention mechanism for γ-reward

The *spatial differentiation* formula proposed in the previous section is based on the fact that each intersection in the road network has the same situation. In other words, the levels of them are equal. But in reality, the levels of the intersections in the road network are different, some have only one or two lanes, and others may have four to six. Imagine if the intersection I_{22} in Fig. 4 is a two-way road which has eight lanes, the intersection I_{23} is same as I_{22}, while on the other side, the intersection I_{32} is a two-way which only has two lanes. The decision of intersection I_{22} in $I_{22} \Rightarrow I_{23}$ and $I_{22} \Rightarrow I_{32}$ is inevitably different. The discharge of intersection I_{22} can easily lead to excessive congestion at intersection I_{32}, but it may not be very serious for intersection I_{23}. In addition to the number of lanes at the intersection, the length between each intersection is different, which means that the maximum congestion length each intersection can accept is also different. In summary, when using a *spatial differentiation* formula correction at an intersection, there are different influence weights for different intersections.

The problems mentioned above can indeed be solved by setting different thresholds to get weights, like (Liu et al., 2017). However, the actual road network situation is very complicated. There is no way to include all parameters into consideration. Therefore, we can learn the rules from the traffic data. In this respect, the attention mechanism gives us a good solution.

Attention mechanism (Bahdanau et al., 2014; Cho et al., 2014; Vaswani et al., 2017) is an algorithm first proposed to solve "seq2seq" translation problem in NLP. Attention can be interpreted broadly as a vector of importance weights: To predict an element, such as a word in a sentence, attention vector can be used to estimate how strongly it is related to other elements, and the sum of its values can be used as an approximation of the target. It breaks the limits of Euclidean distance between data, captures long-range dependencies in the sentences, and provides smoother translations.

In addition to sequence data, Attention can also be used for other types of problems. In the graphics world, the GCN (Kipf and Welling, 2016) tells us that combining local graph structures with node features can achieve good performances in node classification tasks. However, the way GCN combines the characteristics of neighboring nodes is closely related to the structure of the graph, which limits the generalization ability of the trained model on other graph structures. The GAT (Veličković et al., 2018) proposes a weighted summation of neighboring node features using the attention mechanism. The weights of the neighboring node feature depend entirely on the node characteristics and are independent of the graph structure.

Recent research has begun to introduce the idea of Attention to MARL algorithms. A Multi-Agent Actor–Critic (MAAC) algorithm has been proposed that combines attention mechanism (Iqbal and Sha, 2018). MAAC encodes the state of the surrounding agents and obtains the contribution value of the surrounding agents to the current agent through the attention network, together with (o, a) of the current agent as an input, the Q value is obtained through an MLP. While the Q network is updated in reverse, the attention network is updated, and the attention scores of the surrounding agents for the current agent are also corrected. Colight applied attention mechanism to the TSC problem of the large-scale road network, it encodes the state and directly obtains the Q value through the Multi-head Attention network.

2.3.1. Attention

Every agent can interact with their environment and get the observation on time. We first need to embed the observation from the environment by applying a layer of Multi-layer Perception (MLP):

$$z_i = W_I o_i + b \tag{5}$$

To get the weight of the intersection i to the adjacent intersection j, we need to combine their hidden variables z_i, z_j by following dot product:

$$e_{ij} = z_j^T W_k^T W_q z_i \tag{6}$$

e_{ij}, represents the influence of the adjacent intersection j on the current intersection i. It should be noted that the influence between
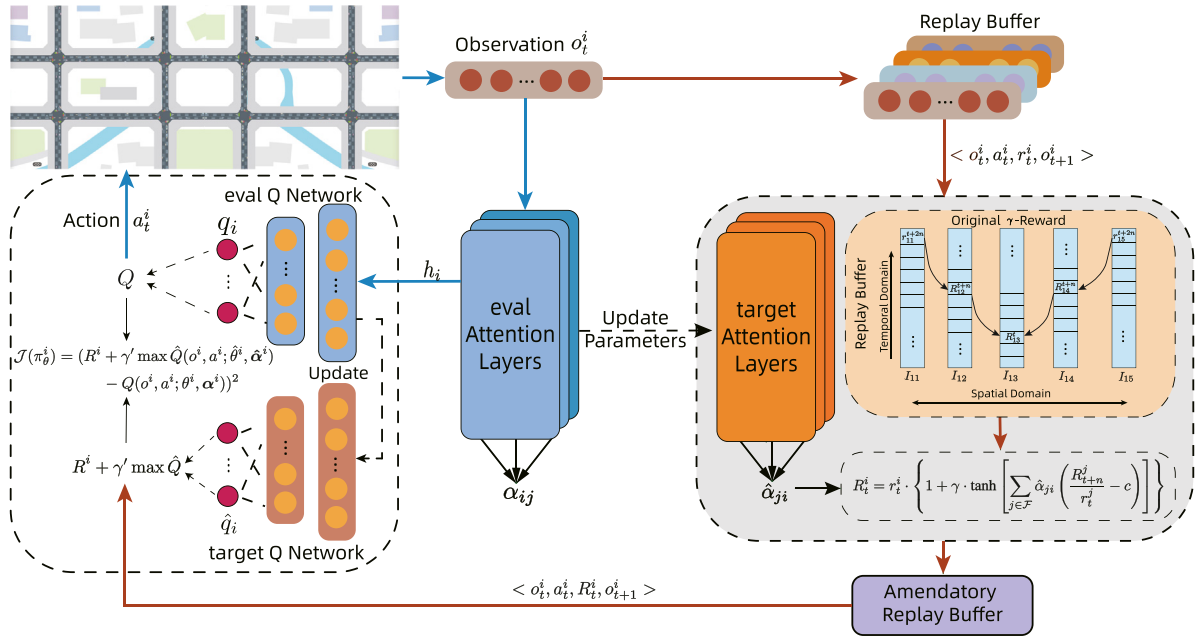
Fig. 5. Framework of the proposed γ-*Attention-Reward* model; In the inner cycle, eval Attention Layers and eval Q Network in blue are used to evaluate real-time Q value for the control. In the external cycle, target system in orange is used to predict long-term impact for improving the performance of Q network and updated periodically. At the right of this figure, we illustrate the message mechanism in original γ-*Reward* by the example of a $Arterial_{1\times5}$ road network. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

them may not be necessarily equal. Then we normalize them by softmax function:

$$\alpha_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(e_{ik}\right)} \tag{7}$$

Impact value v_{ij} can be calculate as $v_{ij} = \alpha_{ij}z_j$, which means the value i needs to consider from j. Finally, adding them together and passing the RELU activation function, the final characterization of the intersection i is obtained:

$$h_i = \sigma\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}z_j\right) \tag{8}$$

2.3.2. γ-attention-reward

We use the attention mechanism to make *spatial differentiation* function more perfect and interpretable by adding an attention score before the sum operation.

$$R_t^i = r_t^i \cdot \left\{1 + \gamma \cdot \tanh\left[\sum_{j\in\mathcal{N}_i}\hat{\alpha}_{ji}\left(\frac{R_{t+n}^j}{r_t^j} - c\right)\right]\right\} \tag{9}$$

Attention score is updated together with the policies:

$$\mathcal{J}(\pi_\theta^i) = (R^i + \gamma'\max\hat{Q}(o^i, a^i; \hat{\theta}^i, \hat{\alpha}^i) - Q(o^i, a^i; \theta^i, \alpha^i))^2 \tag{10}$$

It is worth emphasizing that α_{ij} represents the importance from j to i, but for γ-*Reward* we seek the influence from i to j. So in Eqs. (9) and (10), we need to use α_{ji} for amending rewards of agent i.

Since the attention score α_{ji} is a real-time updated value, this paper uses it as an evaluation metric to dynamically assess the impact of surrounding intersections based on dynamic traffic data. While the reward is also an evaluation indicator, which is timely feedback obtained after performing an action at a particular state, used to evaluate the quality of the state–action pair. If we introduce α_{ij} into the calculation of reward-value and update it in real-time, there is bound to be a problem. This causes the Attention layer to update the direction intentionally, which reduces the impact of essential neighbor intersections and increases those with less traffic flow to increase the reward-value. In this

way, the introduction of the attention score will be even worse than the original γ-*Reward*. To solve this problem, we can follow the Q-learning algorithm and use the off-policy idea to get target attention scores $\hat{\alpha}_{ij}$ in the reward calculation using target Attention layers. Its network parameters are updated together with target Q parameters $\hat{\theta}$. The whole framework of the proposed γ-*Attention-Reward* model is shown in Fig. 5.

In Colight, there is a section devoted to the selection of the hyperparameter neighbor scope. It is found through experiments that the larger the $|\mathcal{N}_i|$ is, the better the performance is. But when it is greater than 5, it takes more time to learn. This is because intersections within the scope of $|\mathcal{N}_i|$ needs to aggregate all the observations into one agent, and the total number of agents is still equal to the intersection, which will inevitably lead to an increase in the amount of calculation. However, unlike Colight, γ-*reward* does not need to consider the size setting of the neighbor number. We can merely consider the neighbor number as a constant 5, which contains the current intersection and the four intersections directly connected to it. Note that, we do not need to employ Multi-head Attention which leads to centralization. For the information of further intersections, recursiveness in the γ-*Reward* formula can work.

From the original γ-*Reward* part in Fig. 5, we can figure out its principle. The row in the figure represents the state information collected in time series and the column represents each intersection. To prevent too much confusion, only the γ-*Reward* process of intersection I_{13} is shown as a demonstration, and the interval of the scale variable n is also not shown here. The value of R_{13}^t is related to R_{12}^{t+n} and R_{14}^{t+n}. By analogy, these two rewards are related to R_{11}^{t+2n} and R_{15}^{t+2n} respectively. It is worth noting that only a two-dimensional replay buffer is drawn here, in fact, the real intersection has four or more surrounding intersections, so it should be a three-dimensional gradient.

3. Theoretical analysis in game theory

In this section, we establish theoretical results for the proposed algorithms. The key challenge of MARL algorithms is that the decision process may be non-stationary (Omidshafiei et al., 2017; Laurent et al., 2011). Since an agent is not aware of actions from other agents and lack of communication, the transition probabilities $\mathcal{P}(o_{t+1}^i | o_t^i \in \mathcal{O}_t^i, a_t^i \in \mathcal{A}_t^i)$

are not stationary and change as the other agents change. So we first need to demonstrate the decision process is stationary by using the proposed algorithms. Specifically, MARL can be regarded as a game model (Aragon-Gómez and Clempner, 2020), and we prove that they can converge to Nash equilibrium. The argument is started with the following definitions.

Definition 3.1. A decentralized MARL decision process is *stationary* (or homogeneous), *iff*, for each agent i and all $p, q \in \mathbb{N}$, $o^i \in \mathcal{O}^i$, $a^i \in \mathcal{A}^i$(Omidshafiei et al., 2017)

$$\sum_{a_p^{-i} \in A_p^{-i}} \mathcal{P}_i \left(o_{p+1}^i | o_p^i, \left\langle a_p^i, a_p^{-i} \right\rangle \right) =$$
$$\sum_{a_q^{-i} \in A_q^{-i}} \mathcal{P}_i \left(o_{q+1}^i | o_q^i, \left\langle a_q^i, a_q^{-i} \right\rangle \right) \tag{11}$$

and $\mathcal{P}$ for global state $s \in S$ must be stationary either.

$$\sum_{a_p^{-i} \in A_p^{-i}} \mathcal{P} \left(s_{p+1}^i | s_p^i, \left\langle a_p^i, a_p^{-i} \right\rangle \right) =$$
$$\sum_{a_q^{-i} \in A_q^{-i}} \mathcal{P} \left(s_{q+1}^i | s_q^i, \left\langle a_q^i, a_q^{-i} \right\rangle \right) \tag{12}$$

where $a^{-i} = a \backslash \left\{ a^i \right\}$.

Based on the definition of stationary MDP, we can define optimal global reward for proving the astringency of proposed methods by extending the definition by Nguyen et al. (2014).

Definition 3.2. For a stationary MDP, the global optimal reward can be decomposed into a sum of local optimal reward for each reward function $f_i \in \mathcal{F}$

$$\rho^* = \sum_{i=1}^m \rho_i^* \tag{13}$$

Proof Sketch. For a given stationary MDP, there must exists a stationary $\mathcal{P}_i^{\pi_*^i}$, where π_*^i is the optimal policy of agent i

$$\rho_i^{\pi_*^i} = \sum_{o^i \in \mathcal{O}^i} \mathcal{P}_i^{\pi_*^i}(o^i) f_i(o^i, a^i | a^i = \pi_*^i(o^i)) \quad \square \tag{14}$$

Definition 3.3. In stochastic game, a Nash equilibrium point is a tuple of m strategies $\left(\pi_*^1, \ldots, \pi_*^m \right)$ such that for all global state $s \in S$ and $i = 1, \ldots, m$ (Hu and Wellman, 2003)

$$v^i(s | \pi_*^1, \ldots, \pi_*^m) \leq v^i(s | \pi_*^1, \ldots, \pi^i, \pi_*^{i+1}, \ldots, \pi_*^m) \tag{15}$$

for all $\pi_*^i \in \Pi^i$, where Π^i is the set of policies of total m agents.

3.1. Stationarity of γ-Reward series

First, we give the proof of stationarity. Unless the MDP is stationary, the model cannot guarantee convergence to the optimal.

Assumption 3.1. The original reward function can be represented as

$$r_t^i = f(o_t^i, a_t^i) \tag{16}$$

where o^i, a^i include state–action pair from time step 1 to t. Assume that γ-*Reward* series are special reward function $f(o, a)$.

$$R_t^i = f \left(\left\langle o_t^i, o_k^{-i} \right\rangle, \left\langle a_t^i, a_k^{-i} \right\rangle \right) \tag{17}$$
$k \in [1, \mathbb{N}]$.

Assumption 3.2. As a continuing task without definite ending, the excepted reward $\mathcal{G}_t^i$ of TSC problem is defined as following with a discount rate γ'

$$\mathcal{G}_t \doteq r_{t+1} + \gamma' r_{t+2} + (\gamma')^2 r_{t+3} + \cdots = \sum_{k=0}^{\mathbb{N}-t} (\gamma')^k r_{t+k+1} \tag{18}$$

Assumption 3.3. The Q function is based on trajectory of expected return.

$$Q(o, a | \pi) = \sum \mathcal{P}(path_{o,a} | \pi) * \mathcal{G}(path_{o,a}) \tag{19}$$

$$a_t = argmax Q_{\max}(o, a | \pi) \tag{20}$$

Theorem 3.1. *With γ-Reward as a coordination mechanism, the decision process of distributed DQN algorithm ($\mathcal{RP}$) is stationary.*

Proof Sketch. According to Assumptions 3.2 and 3.3, Q value function with γ-*Reward* ($\mathcal{RQ}$) can be written as follow

$$(\mathcal{RQ})_t^i(o, a | \pi) = \sum_{t=0}^{\mathbb{N}} (\mathcal{RP})_t^i(path_{o^i, a^i} | \pi) * (\mathcal{RG})_i(path_{o^i, a^i})$$
$$= Q \left(\left\langle o_t^i, o_k^{-i} \right\rangle, \left\langle a_t^i, a_k^{-i} \right\rangle | \pi \right) \tag{21}$$

The calculation of ($\mathcal{RQ}$) is related to the except reward path. ($\mathcal{RG}$)$_t^i$ is a discounted sum of amendatory reward R_t^i, which is bound up with not only (o^i, a^i), but also (o^{-i}, a^{-i}) from the other agents. Since ($\mathcal{RG}$)$_t^i$ records the future trajectory of amendatory reward from time step t to the end of episode, it must contain the previous and posterior state–action pair as a vector, like Eq. (17) shown in Assumption 3.1. According to the above two properties, we can decompose ($\mathcal{RP}$) by using Eq. (20).

$$(\mathcal{RP})_t^i(o_{t+1}^i | o_t^i, a_t^i) = \mathcal{P}_t^i(o_{t+1}^i | o_t^i, \ argmax \ (\mathcal{RQ})_{t \max}^i(o, a))$$
$$= \mathcal{P}_t^i \left(o_{t+1}^i | o_t^i, \left\langle a_t^i, a_k^{-i} \right\rangle \right) \tag{22}$$

where $k \in [1, \mathbb{N}]$. Obviously, ($\mathcal{RP}$) satisfy the property in Definition 3.1. Thus, the process with γ-*Reward* series is a stationary process. $\square$

3.2. Convergence of γ-Reward series

Theorem 3.2. *Spatial differentiation formula in original γ-Reward can lead the reward value function to converge to local optimal reward.*

Proof Sketch. As mentioned in Section 2.B, our Q-value function shares a similar structure with $\mathcal{QD}$-learning. So the proof is inspired by the convergence analysis of $\mathcal{QD}$-learning. Since this article is not mainly about multi-agent theory, we would like to simply transform γ-Reward into the form of $\mathcal{QD}$-learning convergence proof. By following the rest of the proof process in $\mathcal{QD}$-learning, we can get the conclusion of convergence. Thus, we highly recommend readers to read the whole analysis in $\mathcal{QD}$-learning.

According to the *Spectral Graph Theory*, the multi-agent communication network is simplified to an undirected graph $G = (V, E)$, where V represents intersections and E denotes the communication links. From the adjacency matrix A ($A_{ij} = 1$, if $(i, j) \in E$, $A_{ij} = 0$, otherwise) and the degree matrix $D = diag\{d_1, \ldots, d_N\}$ ($d_i = |\mathcal{N}_i|$), we can get a positive definite graph Laplacian matrix $L = D - A$, where eigenvalues ordered as $0 = \lambda_1(L) \leq \lambda_2(L) \leq \cdots \leq \lambda_N(L)$. Let $\beta = -\alpha\gamma$ and $\beta\tanh[\sum_{j \in \mathcal{N}_i}(\frac{R_{i+n}^j}{r_t^j} - c)] = -\beta\tanh[\sum_{j \in \mathcal{N}_i}(c - \frac{R_{i+n}^j}{r_t^j})]$, we note that

$$Q_{o,a}(t+1) = (I_N - \beta L_t + \alpha I_N) r_t$$
$$+ \alpha(C_{o,a}(Q_t) - Q_{o,a}(t) + v_t) \tag{23}$$

where $Q_{o,a} = [Q_{o,a}^1(t), \ldots, Q_{o,a}^N(t)]^T$ and $C_{o,a}(Q_t) = [C_{o,a}^1(Q^1), \ldots, C_{o,a}^N(Q^N)]^T$. The local γ-Reward operator C^i of agent i is

$$C_{o,a}^i(Q) = \mathbb{E}(r_t^i) + \gamma' \sum_{j \in \mathcal{N}} p_{i,j}^a \max_{v \in \mathcal{A}} Q_{j,v} \tag{24}$$

The residual v_t^i for each agent i in Eq. (23) is

$$v_t^i(o, a) = (1 - \frac{1}{\alpha}) r_t^i + \gamma' \max_{v \in \mathcal{A}} Q_{o_{t+1}, v}^i(t) - C_{o,a}^i(Q) \tag{25}$$

By following the proof in Section 5 of $\mathcal{QD}$-learning, we can get the same main result that for each agent i, we have

$$\mathbb{P}(\lim_{t \to \infty} Q_t^i = Q^*) = 1 \tag{26}$$

Table 1
Situation of datasets.

DataSet	Intersections	Arrival rate (vehicles/300 s)			
		Mean	Std	Max	Min
$Arterial_{1\times6}$[a]	6	300	–	–	–
$Grid_{3\times3}uni$[a]	9	300	–	–	–
$Grid_{3\times3}bi$[a]	9	300	–	–	–
$NewYork_{16\times3}$	48	240.79	10.08	274	216
$Jinan_{3\times4}$	12	526.63	86.70	676	256
$Hangzhou_{4\times4}$	16	250.79	38.21	335	208

[a]Traffic flow from synthetic data are uniform, so there is no need to count another three values.

We can conclude that *spatial differentiation* formula can lead the γ-*Reward* to converge to local optimal. Obviously, γ-*Attention-Reward* has the same property. □

3.3. Optimality of γ-Reward model

For a stochastic game with multi-agent, the optimal point of the whole system is a Nash equilibrium point which is declared in Definition 3.3. $v^i(s|\pi_*^1, \ldots, \pi_*^m)$ can be interpreted as the discounted except reward $\mathcal{G}$. According to previous two theorems, we can draw the following conclusions.

Theorem 3.3. *With γ-Reward model, the multi-agent system can converge to a Nash equilibrium point.*

Proof Sketch. First we give the definition of optimal v^* with Definition 3.2

$$v^i(s|\pi_*^1, \ldots, \pi_*^m) = \rho^* = \sum_{i=1}^{m} \rho_i^* \tag{27}$$

From the process of convergence proof, we can figure that the local optimal reward of R^i depends on the local optimal of the other agents ρ_j^*. In other words, if there is an agent which does not converge to local optimal reward, the other will also not be optimal. From this, we can conclude that γ-*Reward* forces agents to care about the others and let the whole system finally converge to a Nash equilibrium point. □

4. Experiment

We conduct experiments on Cityflow, an open-source traffic simulator that supports large-scale traffic signal control (Zhang et al., 2019a), rather than the common used SUMO simulator (Krajzewicz, 2010), since it is more than twenty times faster than SUMO. Moreover, we use Ray (Moritz et al., 2018) framework, which is an open-source library for reinforcement learning that offers both high scalability and a unified API for a variety of applications for DRL algorithms.

4.1. Datasets

In the experiment, we use both synthetic data and real-world data. We share the same real-world dataset with Colight for convenience. The datasets mainly include two parts, roadnet and flow. Roadnet describes the number of intersections in the road network, the coordinates, and the number of lanes owned by each intersection. Flow is based on vehicles and lists thousands of vehicles, each vehicle has its own property, such as length, width, max of accuracy, max of speed and, most importantly, trajectory. The experiment used the real-world data of Hangzhou, Jinan in China, and also New York in the USA. Meanwhile, we used two kinds of synthetic data, arterial and grid type. We counted their characteristics and presented them in Table 1. $Grid_{3\times3}uni$ is one-way traffic, and $Grid_{3\times3}bi$ is two-way with the same road network.

4.2. Baseline methods

In Section 2, we have already introduced methods for TSC, including traditional rule-based methods and learning-based methods. The most primitive rule-based methods are still the most common methods nowadays. As a mature rule-based method, Max-Pressure can be used as a representative.

Learning-based methods have been prosperous under the development of deep learning and data science in recent years. They are characterized by the use of large-scale data to approximate optimal strategies through iterative learning. We have chosen several methods as the baseline:

- *IQL*: Since γ-*Reward* is based on the decoupling idea of IQL, and introduces a coordination mechanism so that it can demonstrate the impact of coordination mechanism compared to IQL. Here we use original D3QN, like (Liang et al., 2019), for comparison.
- *QMIX*: This is a sophisticated MARL algorithm, which integrates all agents into the same model and concentrates on the learning of joint action reward functions. As a typical one-model method, comparing it with γ-*Reward* can effectively observe the advantages and disadvantages of joint learning and independent learning for coordination.
- *Colight*: Unlike γ-*Reward*, Colight is more like QMIX, but not learns a joint action. It uses Attention layers to train the surrounding observation code to replace the observation of the current intersection. Due to its full collection of observation, it can apply Multi-head Attention. By this way, the coordination between agents is realized. γ-*Attention-Reward* has made some improvements on this basis. The method of Replay Buffer Amendment is employed to introduce the effect of the current intersection on the surrounding intersection, replacing the hyperparameter $|\mathcal{N}_i|$ in Colight and adding consideration of the impact of actions on both time and space.

4.3. Evaluation performance

Fig. 6 and Table 2 shows the performance comparison between γ-*Reward* and the more comprehensive γ-*Attention-Reward* and baselines. Each model has trained 100 iterations, and each iteration run 3600 time steps in the simulator. Each action in them lasts at least 10 s for avoiding rapid switching phase impractically. Delay span $n = 10$ s and threshold $c = 0.8$.

We use the average transit time of the vehicle to evaluate the performance of the model, which is the standard evaluation method in the field of TSC.

The performance of learning-based methods is significantly better than rule-based Max-Pressure (Table 2), which is widely proved in many researches.

Among the independent learning DRL model, the performance of γ-*Reward* series far exceeds the IQL model. This demonstrates the importance of coordination between agents for global performance improvement.

It is worth noting that, in all road network, the independent learning DRL model shows a stronger astringency than one-model QMIX. That is probably because for a single model, excessive dimensions can make the policy more difficult to learn. However, Colight does not show divergence while achieved excellent results. This may benefit from that it does not make joint decisions through the joint action function, but by sharing network parameters, so that all agents generate independent actions. Since the agents share the model parameters, they will undoubtedly ignore individual differences and sacrifice some performance. Compared to the proposed model, intense oscillations sometimes occur in Colight during training. This is also a result of sharing parameters. Once the model iterating in the wrong direction, it will mislead all agents and lead to horrible congestion in the whole road network. The performance of proposed methods is even better
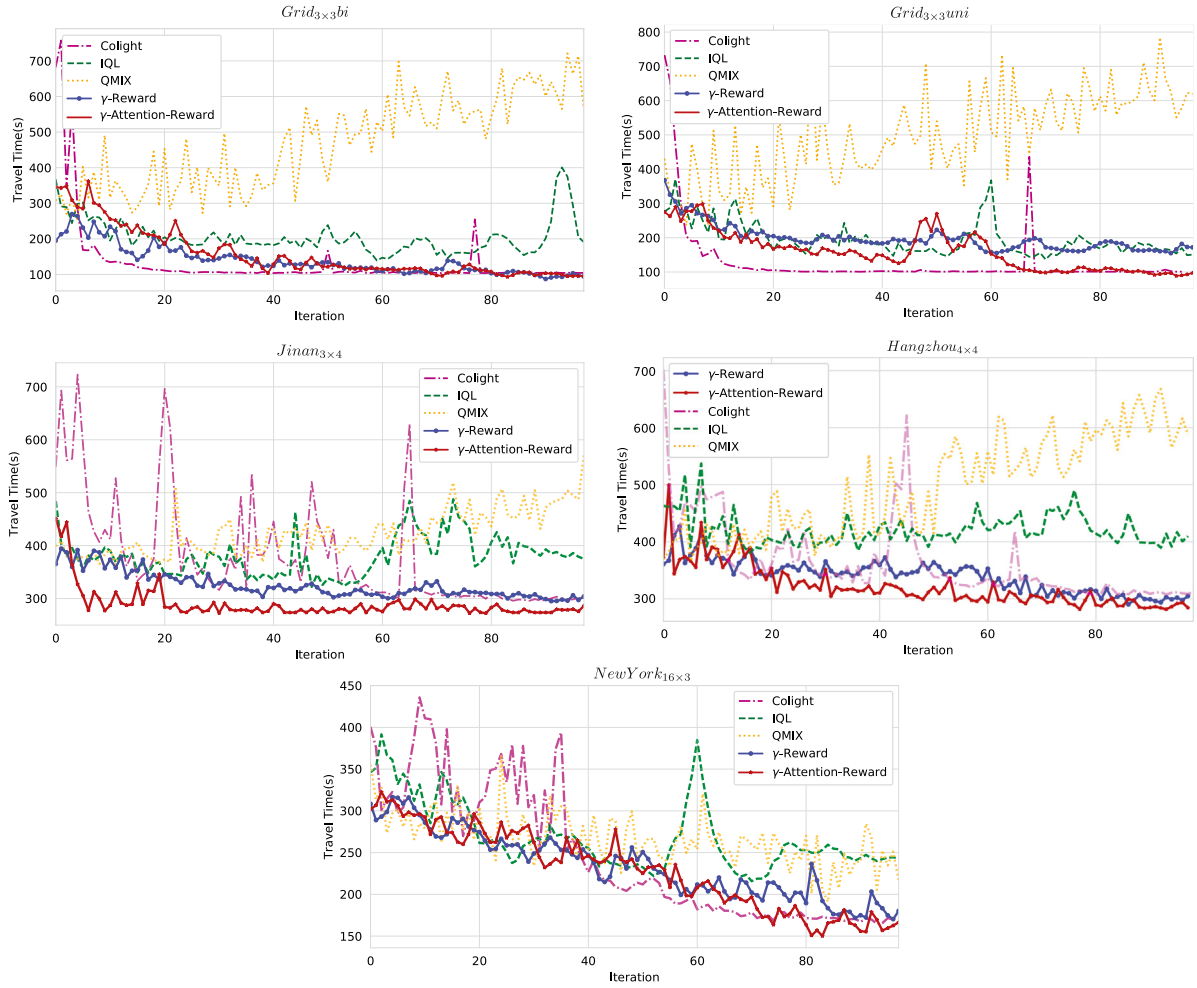
Fig. 6. Evaluation performance compared among baselines, proposed *γ-Reward* and *γ-Attention-Reward*.

Table 2
Performance comparison (average travel time).

Model	$Grid_{3\times3}bi$	$Grid_{3\times3}uni$	$NewYork_{16\times3}$	$Jinan_{3\times4}$	$Hangzhou_{4\times4}$
Max-pressure	204.72	186.06	405.69	359.81	431.53
IQL	191.05	157.51	248.46	371.74	406.27
QMIX	565.70	619.32	216.56	571.78	587.46
Colight	104.89	100.96	169.66	301.78	311.15
$γ$-Reward	96.44	175.38	162.18	303.97	304.90
$γ$-Attention-Reward	96.14	93.93	141.16	286.27	284.24

than Colight, which means sharing sensation is not the only way to realize coordination. Sharing results can also help to focus on the whole road network for a single intersection.

In real-world road networks, *γ-Reward* and *γ-Attention-Reward* do not show gigantic difference. The reason is that all real-world road networks we used are two-way road. We will introduce the study about the attention score later and can be shown in Figs. 9 and 10. It has shown its effect in specifically synthetic road networks. Compared $Grid_{6\times6}bi$ and $Grid_{6\times6}uni$ in Fig. 6, Attention layers distinguish one-way and two-way, and assist agents to achieve a better performance. This will also describe later by revealing the detail of Attention layers.

4.4. Study of hyperparameter γ

We use $Arterial_{1\times6}$ road network to study the impact of different $γ$ value. We have chosen [0.1, 0.3, 0.5, 0.7, 0.9] five $γ$ values and

compared the results. As shown in Fig. 7, 0.5 may be a balance point of the penalty item. So for the hyperparameter $γ$, we all set it to 0.5.

4.5. Visualization of proposed method

The core idea of the *γ-Reward* algorithm is to correct the reward in the replay buffer. In this section, we use the $Arterial_{1\times6}$ road network as an example to show how the reward values between different intersections affect each other. Compare the $Grid_{3\times3}uni$ and $Grid_{3\times3}bi$ road networks to demonstrate the role of the attention mechanism in the *γ-Reward* algorithm improvement.

4.5.1. Visualization of spatial differentiation formula

In Fig. 8, dashed lines represent the original reward, and solid lines represent the corrected reward. Since the linear road network is
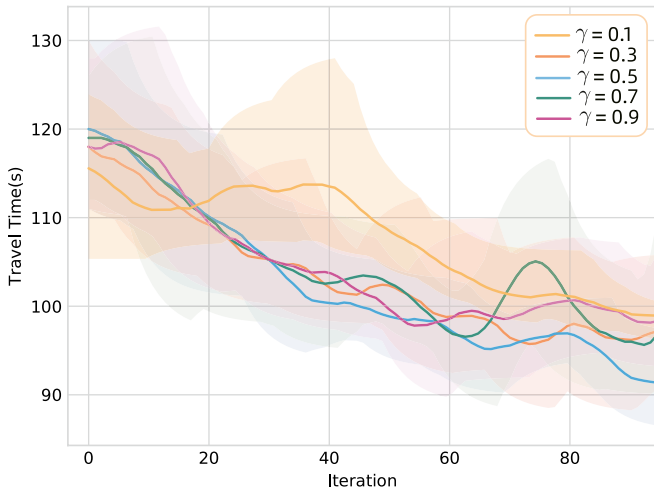
Fig. 7. Study of hyperparameter γ; dark lines represent results after smoothing, light lines represent the deviation of raw results.
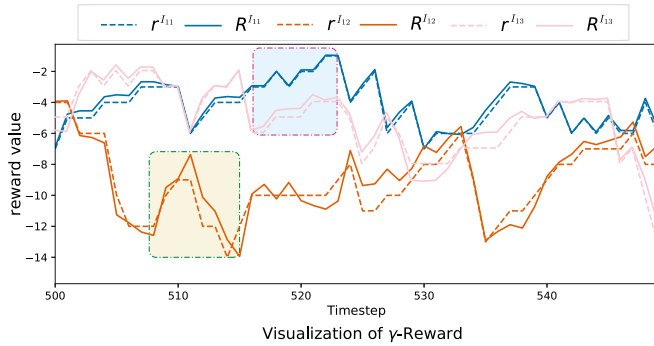

Fig. 8. Visualization of the reward changing by *spatial differentiation*; r^i represents raw reward from D3QN, R^i represents the reward after amendment. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Fig. 9. Attention score of intersection I_{22} selected from $Hangzhou_{4\times4}$ road network.

(a) Attention scores of intersection I_{22} selected from $Grid_{3\times3}bi$

(b) Attention scores of intersection I_{22} selected from $Grid_{3\times3}uni$

Fig. 10. Attention layers learn the different of surrounding from traffic flow; $Grid_{3\times3}bi$ and $Grid_{3\times3}uni$ road network are used for comparison to show the effect of attention layers.

selected, there are no more than two adjacent intersections, so it is easier to observe the influence between the intersections. We just select the first two intersections for clearness.

Observe the red line in the light yellow area, which represents the reward for the second intersection. It can be found that the solid line in this area is almost above the dashed line, meaning that after amendment, rewards become better. The reason is shown in the light blue area, dark blue and pink lines are getting a raise, which means the traffic situation is getting better both in intersection I_{11} and I_{13}. We believe that in the process of getting better at these two intersections, some of the efforts are contributed by intersection I_{12}. The lag between yellow and blue area is up to the delay span n.

From Fig. 8, we can see that in the actual training, the *γ-Reward* framework, as what we expected, introduces future changes in nearby intersections.

4.5.2. Effect of attention mechanism

Fig. 9 shows attention scores from Hangzhou road network. We can find that except current intersection I_{22}, others are declined and tending to the same value. Therefore, Attention does not play an important role in these real-world datasets. The reason could be that all of them are two-way road and have the same number of lanes. To highlight its effect, we need to compare the situation between one-way and two-way traffic in the same road network. Thus, we use $Grid_{3\times3}uni$ and $Grid_{3\times3}bi$ for visualization.
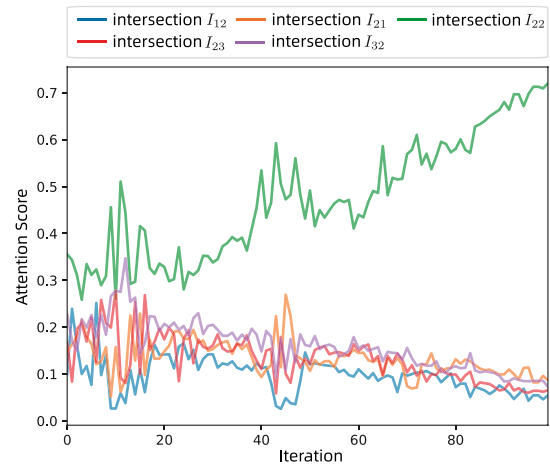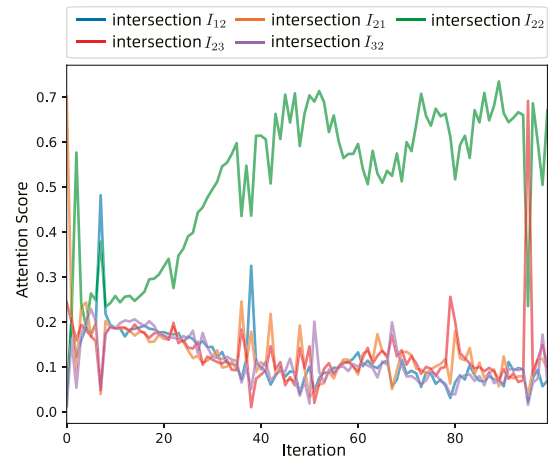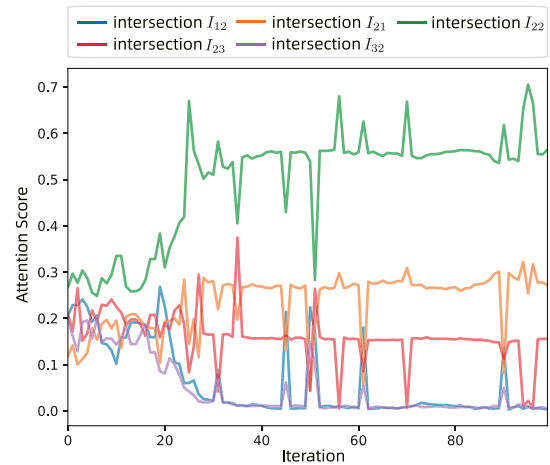
Fig. 10 shows the comparison between one-way and two-way 3×3 network. Comparing Fig. 10(a) with Fig. 10(b), the score change of intersection I_{22} in Fig. 10(a) is the equalization of surrounding
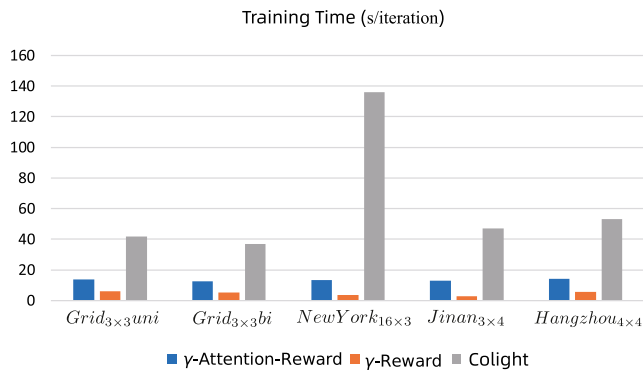
Fig. 11. Training time of *γ-Reward* series and Colight; Note that in order to show the difference between centralized and decentralized algorithms, we illustrate training time of one agent in *γ-Reward* series..

intersection I_{12}, I_{21}, I_{23} and I_{32}. While in Fig. 10(b), the intersection from the direction of exiting is obviously holding a commanding edge. This means that the attention mechanism does have a significant effect in understanding the structure of the road network. With the attention mechanism, the reward amendment is more concerned with the results of its actions, which is crucial in the revision of reward.

5. Discussion about real world application

TSC is an actual task running without the simulator, and the proposed control plan should aim at solving practical problems. Therefore, it is necessary to take the limitations that may exist in practice into account.

5.1. Scalability

Scalability is an important standard to evaluate whether a TSC method is meaningful. The proposed methods in this paper are both decentralized, each D3QN agent can be deployed on the embedding device in each intersection respectively. Either the training time or the inference time of our methods is not related to the number of intersections. Besides, the message transmission among neighboring intersections just takes tolerable time and all of the intersections communicate in parallel. Due to the limitation of computing capacity, experiments of larger scale are not able to conduct. Instead, we analyze the scalability by collecting the training time on datasets with different scales which is listed in Fig. 11. As the number of intersections increased, the time cost of Colight is significantly larger. In contrast, our methods remain a shorter training time due to the parallel computation. Another advantage is when the road network structure is changed, we only need to train the newly added intersections separately and there is a tiny impact on existing intersections. Therefore, the proposed methods are believed to be scalable.

5.2. Real-time

Real-time traffic communication may have problems like communication delay, information security, and risk of packet loss (Ezell et al., 2013). Therefore, it is necessary to decouple the calculation of the whole road network. Decentralization is becoming the trend of solutions in TSC problem (Liu et al., 2017; Chu et al., 2016). Colight, while using Multi-head Attention technology and sharing parameter, summarizes the relationship of the global road network and reduces the time complexity and space complexity of training. However, if we apply it to the practice, the global road network information in real-time is first needed to transmit to the central server for calculation, and then the server needs to dispense the resulting actions to each

intersection. Note that it is not just transmitting actions of each traffic signal, but also their sensor observations! In contrast, communication in our methods occurs in a small neighborhood, and the messages they need are only several bits. These attributes help the intersections keep in touch in real-time.

5.3. Further applications

The coordination mechanism in this paper can be expended to other multi-agent tasks which face the problem of scalability and real-time. For instance, the power dispatching of power networks can be described as the same as the TSC problem. Power transmission is similar to the traffic flow between two neighboring intersections. Furthermore, the control of a multi-joint tandem robot may be regarded as a multi-agent task. Actions from the joint which nears the base have a consequence on other joints and the end-effector. We can use the coordination mechanism to consider this effect on neighboring agents or joints and achieve a global optimal performance. There are already broad applications using this kind of coordination mechanism in engineering systems like sensor networks (Rabbat and Nowak, 2004), smart grid (Zhang et al., 2018) and robotics (Corke et al., 2005).

6. Conclusion

In this paper, we propose the *γ-Reward* method and its variant *γ-Attention-Reward* that introduces the attention mechanism to solve the problem of intelligent control of the traffic signal. By extending the Markov Chain to the space–time domain, the methods turn to be a scalable solution for TSC. Specially, we give a concise proof of the *spatial differentiation* formula which shows that the two frameworks can converge to Nash equilibrium. We conduct extensive experiments using both real-world and synthetic data. They confirm that our proposed methods have a superior performance over state-of-the-art methods. In the asymmetry road network, *γ-Attention-Reward* shows inspired results than *γ-Reward* by adding the attention mechanism. Moreover, we investigate the effect of the reward amendment and attention mechanism in achieving coordination thoroughly. Compared to the recently proposed Colight, *γ-Reward* series replaces the graph attention with recursion, decouples the calculation of the whole road networks, and is more suitable for practical applications.

CRediT authorship contribution statement

Junjia Liu: Conceptualization, Methodology, Software, Writing - original draft. **Huimin Zhang:** Data curation, Writing - original draft. **Zhuang Fu:** Supervision, Funding acquisition. **Yao Wang:** Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

---

[2] Our project of Deecamp 2019 is open-source in Github: https://github.com/multi-commander/Multi-Commander.

Appendix

A.1. Fundamental DRL algorithms

A.1.1. Temporal-Difference Learning

Temporal-Difference Learning (TD learning), proposed by Sutton (1988), combining with Dynamic Programming (DP) and Monte Carlo (MC) methods, becomes the core idea of DRL. Like the Monte Carlo algorithm, it does not need to know the specific environment model and can learn directly from experience. On the other hand, it also inherits bootstrapping from DP algorithm, which is the unique feature of TD learning: predictions are used as targets during learning (Sutton and Barto, 2018). Monte Carlo simulates (or experiences) an episode until it ends, then estimates the state value based on the value of each state. In contrast, TD learning simulates an episode with one step (or several steps) per action which based on the value of the new state, and then estimate the state value before execution.

The Q-learning algorithm (Watkins and Dayan, 1992) is a ground-breaking algorithm. TD learning is used here for off-policy learning.

$$\delta_t = r_{t+1} + \gamma' \max_a Q\left(o_{t+1}, a_{t+1}\right) - Q\left(o_t, a_t\right)$$

where δ_t represents TD-error.

A.1.2. Deep Q-network

Deep Q-network (DQN) is a powerful off-policy algorithm which has achieved excellent results in many fields since 2015 (Mnih et al., 2015). It uses a neural network to approximate the Q-value function instead of tabling.

$$Q\left(o_t, a_t\right) \leftarrow Q\left(o_t, a_t\right) + \alpha \left[y_j - Q\left(o_t, a_t\right)\right]$$

$$y_{t+1} = r_{t+1} + \gamma' \max_a Q\left(o_{t+1}, a_{t+1}\right)$$

A.1.3. Double deep Q-network

Q-learning uses *max* to select the best action, which causes a Maximization Bias problem. So (Van Hasselt et al., 2016) solved this by designing a Double Q-learning, it only differs in the calculation of the target Q value:

$$y_{t+1} = r_{t+1} + \gamma' Q'\left(\phi\left(o'\right), \arg\max_{a'} Q\left(\phi\left(o'\right), a, w\right), w'\right)$$

A.1.4. Dueling deep Q-network

Another improvement for DQN is Dueling DQN (Wang et al., 2016), which decomposes the Q network into two separate control streams, a value function $V(s)$, and a state-based action advantage function $A(s, a)$. These two control flows obtain an estimate value of the Q function through a special aggregating layer.

$$Q(o, a; \theta, \alpha, \beta) = V(o; \theta, \beta) +$$
$$\left(A(o, a; \theta, \alpha) - \frac{1}{|\mathcal{A}|} \sum_{a'} A\left(o, a'; \theta, \alpha\right) \right)$$

A.2. Pseudocode for γ-Reward series

See Algorithms 1 and 2.

Algorithm 1 γ-*Attention-Reward* Algorithm for MARL Traffic Lights Control

1: Initialize E parallel environments with N agents
2: Initialize replay memory D to capacity N_D
3: Initialize raw replay memory D_r to capacity $N_D + n$
4: Initialize action-value function Q with random weights θ
5: Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
6: Initialize attention scores $\alpha_{i,j}$
7: $T_{update} \leftarrow 0$
8: **for** $episode = 1, M$ **do**
9:     Reset environments, and get initial o_i for each agent i
10:     **for** $t = 1, T$ **do**
11:        Select actions $a_i \sim \pi_i(\cdot|o_i)$ for each agent i in each environment e
12:        Send actions a_i to all parallel environments and get o'_i, r_i for all agents
13:        Store (a_i, o_i, r_i, o'_i) in D_r
14:        $T_{update} = T_{update} + N$
15:        **if** $T_{update} \leq$ min steps per update $+ n$ **then**
16:           *Replay Buffer Amendment*$(D_r, \alpha_{i,j})$
17:           Update Policies:
             Calculate $a^B_{1...N} \sim \pi_i^{\bar{\theta}}\left(o'^B_i\right), i \in 1 \dots N$
             Calculate $Q_i^{\bar{\psi}}\left(o^B_{1...N}, a^B_{1...N}\right)$ for all i in parallel
             Update policies using $\nabla_{\theta,i} \mathcal{J}(\pi_\theta)$ and Adam
18:           Update target Q parameters: $\hat{\theta} \leftarrow \theta$
19:           Update target attention parameters: $\hat{\alpha} \leftarrow \alpha$
20:           $T_{update} \leftarrow 0$
21:        **end if**
22:     **end for**
23: **end for**

Algorithm 2 *Replay Buffer Amendment*

1: **function** REPLAYBUFFERAMENDMENT$(D_r, \alpha_{i,j})$
2:     **for** $i = 1, N$ **do**
3:        $index = len(D_r) - n$
4:        **while** $index > ex\_index$ **do**
5:           $(o_i, a_i, r_i, o'_i) = D_{r,i}(j)$
6:           $R_i = \gamma$-*Attention-Reward* function$(r_i, \alpha_{i,j})$
7:           Store (o_i, a_i, R_i, o'_i) in D
8:           $j = j - 1$
9:        **end while**
10:        $ex\_index = len(D_r) - n$
11:     **end for**
12: **end function**

References

Aragon-Gómez, R., Clempner, J.B., 2020. Traffic-signal control reinforcement learning approach for continuous-time Markov games. Eng. Appl. Artif. Intell. 89, 103415.

Bahdanau, D., Cho, K., Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

Buşoniu, L., Babuška, R., De Schutter, B., 2010. Multi-agent reinforcement learning: An overview. In: Innovations in Multi-Agent Systems and Applications-1. Springer, pp. 183–221.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

Chu, T., Qu, S., Wang, J., 2016. Large-scale traffic grid signal control with regional reinforcement learning. In: 2016 American Control Conference (ACC). IEEE, pp. 815–820.

Corke, P., Peterson, R., Rus, D., 2005. Networked robots: Flying robot navigation using a sensor net. In: Robotics Research. the Eleventh International Symposium. Springer, pp. 234–243.

Ezell, B.C., Robinson, R.M., Foytik, P., Jordan, C., Flanagan, D., 2013. Cyber risk to transportation, industrial control systems, and traffic signal controllers. Environ. Syst. Decis. 33 (4), 508–516.

Foerster, J., Assael, I.A., de Freitas, N., Whiteson, S., 2016. Learning to communicate with deep multi-agent reinforcement learning. In: Advances in Neural Information Processing Systems. pp. 2137–2145.

Hessel, M., Modayil, J., van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M.G., Silver, D., 2018. Rainbow: Combining improvements in deep reinforcement learning. In: AAAI.

Hu, J., Wellman, M.P., 2003. Nash Q-learning for general-sum stochastic games. J. Mach. Learn. Res. 4 (Nov), 1039–1069.

Iqbal, S., Sha, F., 2018. Actor-attention-critic for multi-agent reinforcement learning. arXiv preprint arXiv:1810.02912.

Kar, S., Moura, J.M., Poor, H.V., 2013. *QD*-Learning: A collaborative distributed strategy for multi-agent reinforcement learning through consensus + innovations. IEEE Trans. Signal Process. 61 (7), 1848–1862.

Kipf, T.N., Welling, M., 2016. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.

Koonce, P., Rodegerdts, L., 2008. Traffic Signal Timing Manual. Tech. Rep., United States. Federal Highway Administration.

Krajzewicz, D., 2010. Traffic simulation with sumo–simulation of urban mobility. In: Fundamentals of Traffic Simulation. Springer, pp. 269–293.

Laurent, G.J., Matignon, L., Fort-Piat, L., et al., 2011. The world of independent learners is not Markovian. Int. J. Knowl. Based Intell. Eng. Syst. 15 (1), 55–64.

Liang, X., Du, X., Wang, G., Han, Z., 2018. Deep reinforcement learning for traffic light control in vehicular networks. arXiv preprint arXiv:1803.11115.

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, W., Qin, G., He, Y., Jiang, F., 2017. Distributed cooperative reinforcement learning-based traffic signal control that integrates v2x networks' dynamic clustering. IEEE Trans. Veh. Technol. 66 (10), 8667–8681.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529.

Moritz, P., Nishihara, R., Wang, S., Tumanov, A., Liaw, R., Liang, E., Elibol, M., Yang, Z., Paul, W., Jordan, M.I., et al., 2018. Ray: A distributed framework for emerging AI applications. In: 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18). pp. 561–577.

Mousavi, S.S., Schukat, M., Howley, E., 2017. Traffic light control using deep policy-gradient and value-function-based reinforcement learning. IET Intell. Transp. Syst. 11 (7), 417–423.

Nguyen, D.T., Yeoh, W., Lau, H.C., Zilberstein, S., Zhang, C., 2014. Decentralized multi-agent reinforcement learning in average-reward dynamic DCOPs. In: Twenty-Eighth AAAI Conference on Artificial Intelligence.

Nishi, T., Otaki, K., Hayakawa, K., Yoshimura, T., 2018. Traffic signal control based on reinforcement learning with graph convolutional neural nets. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 877–883.

Omidshafiei, S., Pazis, J., Amato, C., How, J.P., Vian, J., 2017. Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, pp. 2681–2690.

Van der Pol, E., Oliehoek, F.A., 2016. Coordinated deep reinforcement learners for traffic light control. In: Proceedings of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016).

Rabbat, M., Nowak, R., 2004. Distributed optimization in sensor networks. In: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks. pp. 20–27.

Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., Whiteson, S., 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In: International Conference on Machine Learning. pp. 4295–4304.

Stevanovic, A., 2010. Adaptive Traffic Control Systems: Domestic and Foreign State of Practice. Project 20-5 (Topic 40-03).

Sutton, R.S., 1988. Learning to predict by the methods of temporal differences. Mach. Learn. 3 (1), 9–44.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: an Introduction. MIT press.

Tan, M., 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In: Proceedings of the Tenth International Conference on Machine Learning. pp. 330–337.

To, H.R., Barker, M.M., 2001. White Paper European Transport Policy for 2010: Time to Decide. Commission of the European Communities Brussels.

Török, J., Kertész, J., 1996. The green wave model of two-dimensional traffic: Transitions in the flow properties and in the geometry of the traffic jam. Physica A 231 (4), 515–533.

Van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double q-learning. In: Thirtieth AAAI Conference on Artificial Intelligence.

Varaiya, P., 2013. The max-pressure controller for arbitrary networks of signalized intersections. In: Advances in Dynamic Network Modeling in Complex Transportation Systems. Springer, pp. 27–66.

Varshavskaya, P., Kaelbling, L.P., Rus, D., 2009. Efficient distributed reinforcement learning through agreement. In: Distributed Autonomous Robotic Systems 8. Springer, pp. 367–378.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. In: Advances in Neural Information Processing Systems. pp. 5998–6008.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks. In: International Conference on Learning Representations.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., Freitas, N., 2016. Dueling network architectures for deep reinforcement learning. In: International Conference on Machine Learning. pp. 1995–2003.

Watkins, C.J., Dayan, P., 1992. Q-learning. Mach. Learn. 8 (3–4), 279–292.

Wei, H., Chen, C., Zheng, G., Wu, K., Gayah, V., Xu, K., Li, Z., 2019a. PressLight: Learning max pressure control to coordinate traffic signals in arterial network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1290–1298.

Wei, H., Xu, N., Zhang, H., Zheng, G., Zang, X., Chen, C., Zhang, W., Zhu, Y., Xu, K., Li, Z., 2019b. Colight: Learning network-level cooperation for traffic signal control. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 1913–1922.

Wei, H., Zheng, G., Gayah, V.V., Li, Z., 2019c. A survey on traffic signal control methods. CoRR vol. abs/1904.08117.

Wei, H., Zheng, G., Yao, H., Li, Z., 2018. Intellilight: A reinforcement learning approach for intelligent traffic light control. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. ACM, pp. 2496–2505.

Wikipedia contributors, 2019. Lane—Wikipedia, the free encyclopedia. [Online; accessed 1-November-2019].

Xiong, Y., Zheng, G., Xu, K., Li, Z., 2019. Learning traffic signal control from demonstrations. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 2289–2292.

Zhang, H., Feng, S., Liu, C., Ding, Y., Zhu, Y., Zhou, Z., Zhang, W., Yu, Y., Jin, H., Li, Z., 2019a. Cityflow: A multi-agent reinforcement learning environment for large scale city traffic scenario. In: The World Wide Web Conference. ACM, pp. 3620–3624.

Zhang, K., Shi, W., Zhu, H., Dall'Anese, E., Başar, T., 2018. Dynamic power distribution system management with a locally connected communication network. IEEE J. Sel. Top. Sign. Proces. 12 (4), 673–687.

Zhang, K., Yang, Z., Başar, T., 2019b. Multi-agent reinforcement learning: A selective overview of theories and algorithms. arXiv preprint arXiv:1911.10635.

Zheng, G., Xiong, Y., Zang, X., Feng, J., Wei, H., Zhang, H., Li, Y., Xu, K., Li, Z., 2019. Learning phase competition for traffic signal control. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 1963–1972.