Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6693636, 13 pages
https://doi.org/10.1155/2021/6693636

*Research Article*

# Coordinated Control of Distributed Traffic Signal Based on Multiagent Cooperative Game

**Zhenghua Zhang**,[1] **Jin Qian**,[1] **Chongxin Fang**,[1] **Guoshu Liu**,[2] **and Quan Su**[2]

[1]*College of Information Engineering, Yangzhou University, Yangzhou, China*
[2]*Yangzhou Guomai Communication Development Co. LTD., Yangzhou, China*

Correspondence should be addressed to Zhenghua Zhang; zhangzh@yzu.edu.cn

Received 7 November 2020; Revised 9 December 2020; Accepted 19 May 2021; Published 2 June 2021

Academic Editor: Zhipeng Cai



In the adaptive traffic signal control (ATSC), reinforcement learning (RL) is a frontier research hotspot, combined with deep neural networks to further enhance its learning ability. The distributed multiagent RL (MARL) can avoid this kind of problem by observing some areas of each local RL in the complex plane traffic area. However, due to the limited communication capabilities between each agent, the environment becomes partially visible. This paper proposes multiagent reinforcement learning based on cooperative game (CG-MARL) to design the intersection as an agent structure. The method considers not only the communication and coordination between agents but also the game between agents. Each agent observes its own area to learn the RL strategy and value function, then concentrates the $Q$ function from different agents through a hybrid network, and finally forms its own final $Q$ function in the entire large-scale transportation network. The results show that the proposed method is superior to the traditional control method.

## 1. Introduction

With the continuous growth of modern urban road traffic volume and road network density, traffic congestion has become a global common problem. In a region, with the passage of time, the congestion of intersection may gradually spread to several intersections in the surrounding area or even all intersections in the whole area. However, due to the limitation of urban space, it is difficult to realize the connection by expanding the road [1]. Therefore, on the premise of not changing the road network structure, the traffic signal adaptive control strategy can improve the traffic efficiency of the intersection and effectively reduce the emission pollution caused by vehicle starting and braking.

### 1.1. Related Work

*1.1.1. Nonreinforcement Learning Method.* The traditional traffic signal uses a fixed phase time control method [2], that is, the time and sequence of the traffic light switching at each intersection are set to a fixed mode. Although this method is simple and easy to implement, it is easy to cause a situation where one phase is unblocked and another phase is congested, which is not conducive to alleviating traffic conditions. Later, with the development of wireless sensors, the control method began to become intelligent. Heuristic signal control methods appeared; the signal control scheme can be formulated according to the traffic information at the intersection [3], such as Jiao et al. [4] switched phase sequence to achieve congested phase priority traffic. In actual application scenarios, this method greatly reduces the traffic congestion problem compared to the fixed mode control method. But its control range is limited to a single isolated intersection. It not only has no coordination mechanism with adjacent intersections but also is not suitable for complex traffic network and dynamic traffic flow. In recent years, the algorithm of traffic signal control is developing towards intelligence, for example, swarm intelligence algorithms (particle swarm optimization (PSO), ant colony optimization (ACO), etc.) [5], fuzzy logic reasoning (FLR) [6], and artificial neural network (ANN) [7]. Although the above methods can solve the optimization problem of traffic signals, most of

the methods are limited to relatively stable traffic conditions and cannot complete real-time strategy updates in actual complex and changeable traffic conditions.

*1.1.2. Single-Agent Reinforcement Learning.* Recently, the advantages of RL application in transportation are gradually reflected. RL has strong adaptive control capabilities as a branch of machine learning (ML). As one of the classic RL algorithms, Q-learning [8] was first used in traffic signal control algorithms [9]. It belongs to model-free, and it usually learns the optimal strategy directly through continuous attempts. In the beginning, Q-learning showed excellent ability in single intersection signal control [10–14]; then, it is combined with the multiagent model to apply to multi-intersection signal control. But as the number of intersections increases, the state of the intersection will increase, and the global/joint action space of the agent will increase exponentially. This brings the challenge of vector dimensionality explosion. In order to avoid or solve this problem, many scholars have done research. In [15], feedforward neural network and Q-learning value function to approximate. Kernel methods are introduced in [16] to continuously update the feature vector to adjust the $Q$ function. In [17], radial basis function, the A-CATs controller, acts as the $Q$ function approximator. Although these methods can solve the problem of dimensional explosion, they are limited to simple traffic environment, until Mnih et al. [18] proposed the deep Q network (DQN) algorithm, which uses a neural network to approximate the $Q$ value of all possible behaviours in each state. It is one of the widely used deep reinforcement learning (DRL) algorithms. This algorithm also has a very wide range of applications in the field of intelligent transportation [19–22]. Li et al. [19] use DQN pairs to take the sampled traffic state as input and the maximum intersection throughput as output. A cooperative DQN with $Q$ value transmission (QT-DQN) is proposed in [20], which discretized the traffic information and then entered the state encoding. The method has combined with the state of neighboring agents to search for the optimal control strategy. However, no matter how powerful a neural network is in actual use, without an excellent coordination mechanism, it will be difficult to cope with the large-scale, complex, and changeable traffic environment.

*1.1.3. Multiagent Reinforcement Learning.* The MARL framework is generally used in the coordinated control of multiple intersections. In the MARL framework, each agent transmits information to each other and cooperates with each other to obtain the overall optimal strategy. The max-plus algorithm is used in [23] to implement an explicit coordination mechanism in the agent's reward function so that the DQN of a single agent can be applied to the multiagent system (MAS). In [24], there is direction of traffic flow that each agent should emphasize during the learning process and improve the conflict between adjacent intersections in the max-plus algorithm, thereby enhancing the coordination between agents. Although the above algorithm can find the Nash equilibrium strategy of multiagent reinforcement learning to a certain extent, it cannot allow the agent to adjust and optimize its own strategy according to the opponent's strategy. The algorithm can only find the Nash equilibrium strategy of the random game. In addition, when further extended to actual traffic systems with more intersections, this method will quickly become difficult to calculate. For large-scale control tasks through DRL, the work in [25] considers the application of policy gradient methods to control traffic signal timing. The author describes the problem as a continuous control problem and applies the depth deterministic strategy gradient (DDPG) to the centralized control of traffic signals in the entire transportation system. However, in their experiments, this simple centralized method only achieved slightly better performance than ordinary Q-learning on the traffic grid at six intersections.

*1.2. Contributions.* Many studies have realized the optimization methods to solve the traffic control problems by offline searching control parameters according to historical traffic patterns. They have obvious limitations when the traffic flow changes significantly in a short period of time. In order to solve the related problems, this study is aimed at using agent-based framework and emerging intelligent control technology to achieve traffic adaptive control. A computational framework for reinforcement learning of cooperative game multiagent (CG-MARL) is proposed. The main contributions can be summarized as follows:

(1) The paper adopts an algorithm to make the monotonicity of the joint action value function and each local value function the same. Taking the maximum action for the local value function is to maximize the joint action value function

(2) Our work verifies the proposed algorithm from three evaluation indexes: average vehicle flow speed, average vehicle delay time, and average emission of $COx$

*1.3. Organization.* The other parts of this article are arranged as follows. Section 2 mainly introduces the multiagent system model. Section 3 elaborates on the algorithm implementation. Section 4 is the experimental results and evaluates the performance of the proposed method. Lastly, we make a conclusion in Section 5.

## 2. System Model

We aim to design a multiagent learning framework with cooperative learning. The framework can make full use of the state information of intersections and the influence of adjacent intersections. The multi-intersection traffic network in the region is modeled as a multi-intersection agent system. Each agent controls an intersection through a deep learning network. In this way, the method can optimize the overall traffic signal plan for regional traffic scenarios and balance the congestion traffic at each intersection.

In a multiagent control system, each agent improves its strategy by interacting with the environment to obtain reward values, as to obtain the optimal strategy in the environment. This section mainly describes the multiagent traffic signal control framework, as shown in Figure 1. This

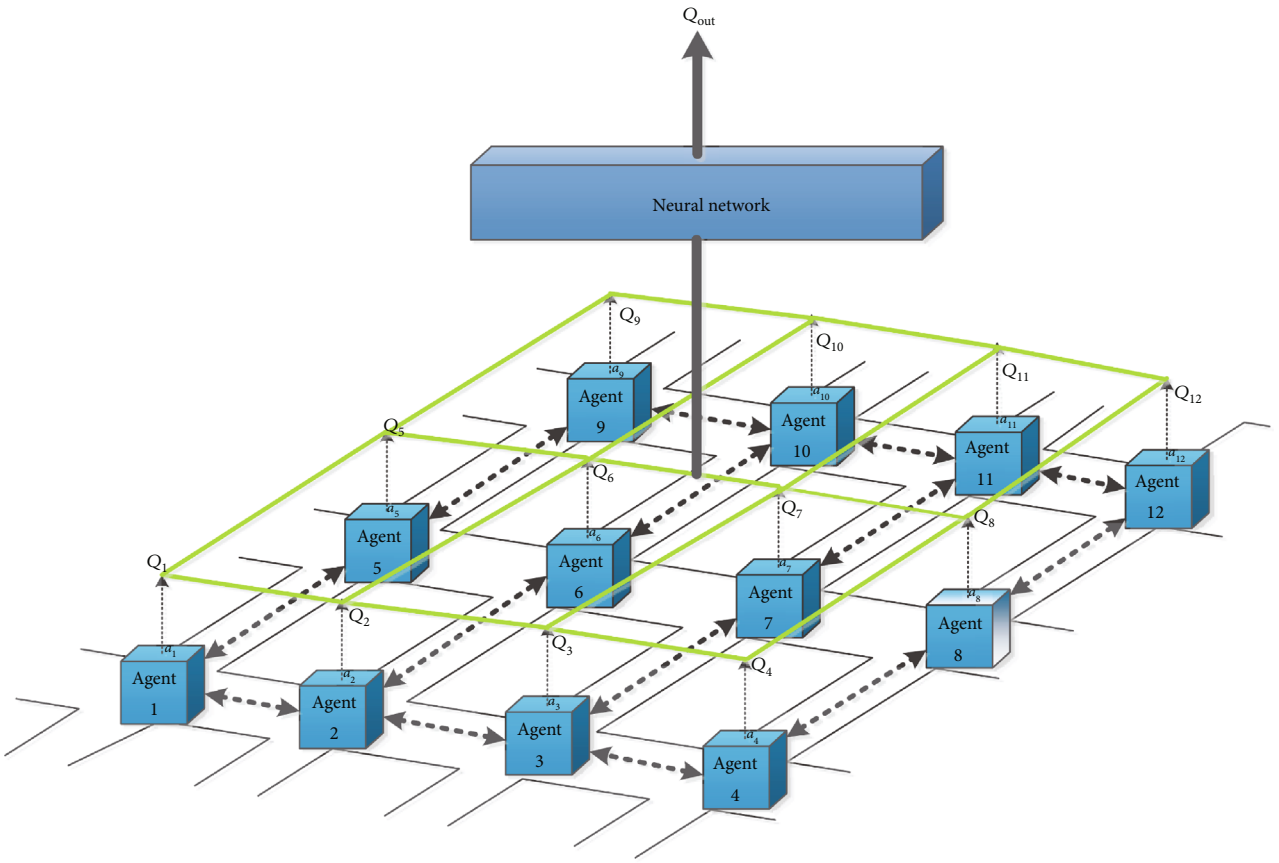


Figure 1: Multiagent reinforcement learning model.

framework consists of three parts. The bottom layer is the perception layer, that is, the agent perceives local traffic state information and makes corresponding decision actions to generate the $Q$ function value; the middle layer is the $Q$ network layer, which is the $Q$ value generated by each agent merging; the highest layer is the coordination layer, which judges the pros and cons of all $Q$ function values and coordinates and controls all agents in the lowest layer. In this framework, each intersection is equivalent to an agent structure, the intersection agent can perceive the surrounding traffic state information, and each agent makes decisions in discrete time intervals based on the perceived traffic state information. This framework is a distributed structure. Each agent can communicate with its neighbors to publish its current state information, ensuring that agents can coordinate with each other, as to achieve the goal of stabilizing the entire system.

## 3. An Algorithm for Multi-Intersection Signal Control

This part first introduces the related theoretical basis of reinforcement learning and then proposes a multiagent multi-intersection signal control reinforcement model and a collaborative reinforcement learning algorithm.

*3.1. Reinforcement Learning.* RL is a branch of machine learning, which is realized through interaction with the environment. This is a kind of goal-oriented learning. The learner does not know which behaviour is the best at first, and this is determined from the consequences of his actions. The basic model of RL, Markov Decision Process (MDP), provides a mathematical framework for modelling decision scenarios, which can be defined by 6 tuples:

$$<S, A, P^{a}_{ss'}, R^{a}_{ss'}, \gamma, V|s, s' \in S, a \in A>, \quad (1)$$

where $S$ represents a set of states the agent and $s, s'$ belong to the state at a certain time and the next time. $A$ represents the execution of the agent when it transitions from one state to another. $P^{a}_{ss'}$ is the transition probability, which is the probability that the agent performs a certain behaviour $a$ and transfers from one state $s$ to another state $s'$. $R^{a}_{ss'}$ is the probability that the agent will perform a certain behaviour $a$ and transfer from one state $s$ to another state $s'$. $\gamma$ is the discount factor, which controls the importance of instant rewards and future rewards. Since there is no final state in the continuous task, its return function can be redefined. The sum of the returns is infinity, which introduces a discount factor. The reward function is defined as

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+1} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}, \quad (2)$$

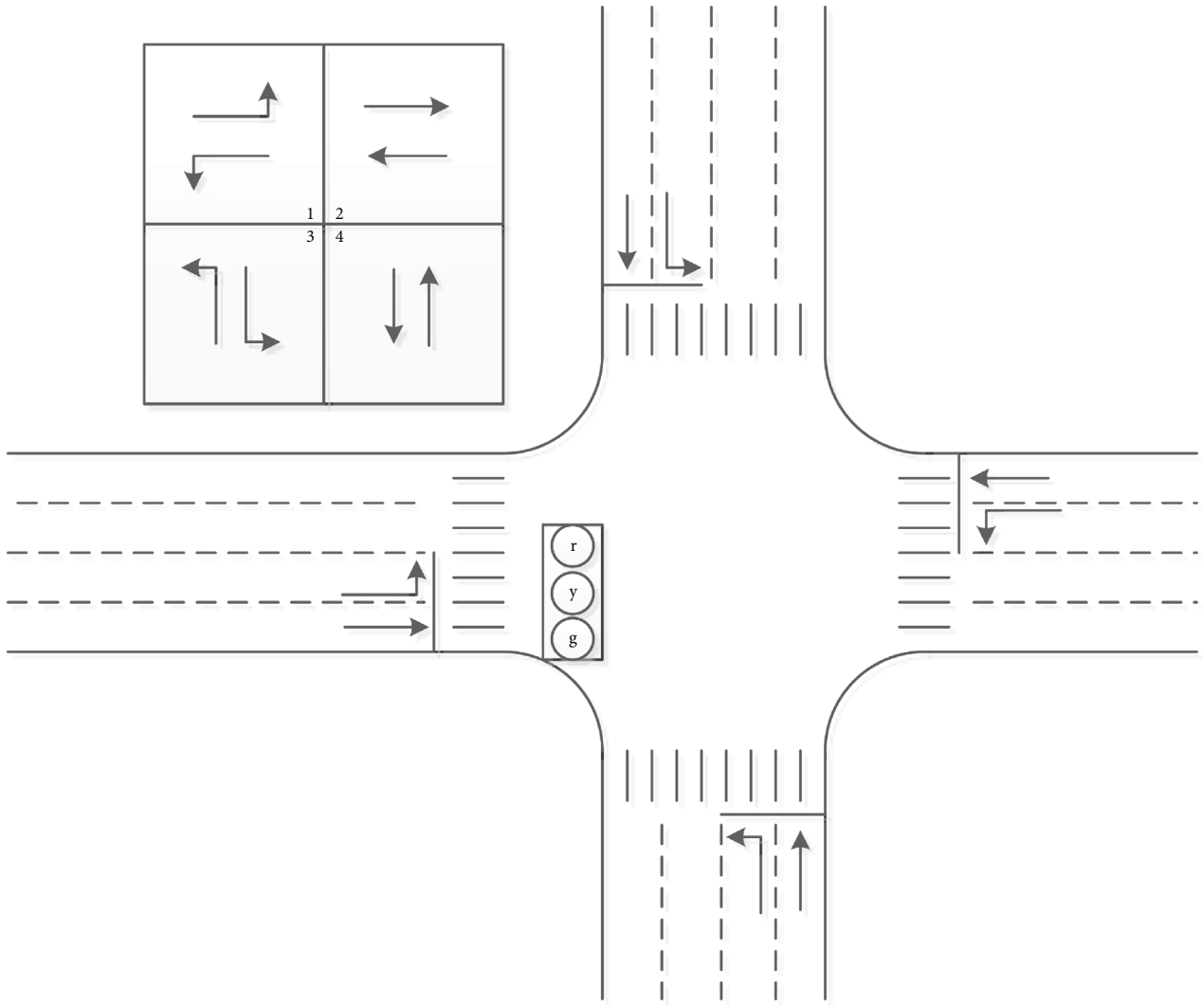


Figure 2: Schematic diagram of intersection.

where $R_t$ is the sum of all the rewards and $r_t$ is the reward value at each step.

MDP is essentially a probability model, which only depends on the current state to predict the next state, but has nothing to do with the previous state. Another way to think of this is that the future action state is independent of the past state and the selected action. However, it is obviously unrealistic to calculate the future discount reward for each state. Therefore, the state behaviour value function, also known as $Q$ function, is introduced to indicate the optimal degree of agent following the specific behaviour of policy $\pi$ in a certain state. The expression is

$$Q^{\pi}(s,a) = E_{\pi}\left[\sum_{k=0}^{\infty} \gamma^{k} r_{t+k+1} \mid s_t = s, a_t = a\right]. \quad (3)$$

Q-learning is one of the classical algorithms in RL. It is a time difference algorithm where the state behaviour value pair is considered, that is, the action of behaviour $a$ is executed in states. The expression is

$$Q(s,a) = Q(s,a) + \alpha\left(r + \gamma \max Q\left(s', a'\right) - Q(s,a)\right). \quad (4)$$

However, the traditional Q-learning algorithm is only suitable for an environment with a finite number of states and a finite number of behaviours. Later, with the development of artificial neural networks, deep reinforcement learning overcomes this point, which is to approximate the $Q$ function with a certain parameter: $Q(s,a;\theta) \approx Q^{*}(s,a)$. Use a weight for the neural network to approximate the values of all possible behaviours in each state, approximate the $Q$ function as a function approximator, and minimize the error through gradient descent, such as DQN which uses CNN network and DRQN which uses RNN network.

The agent's environment is stable in single-agent reinforcement learning. On the contrary, the environment is

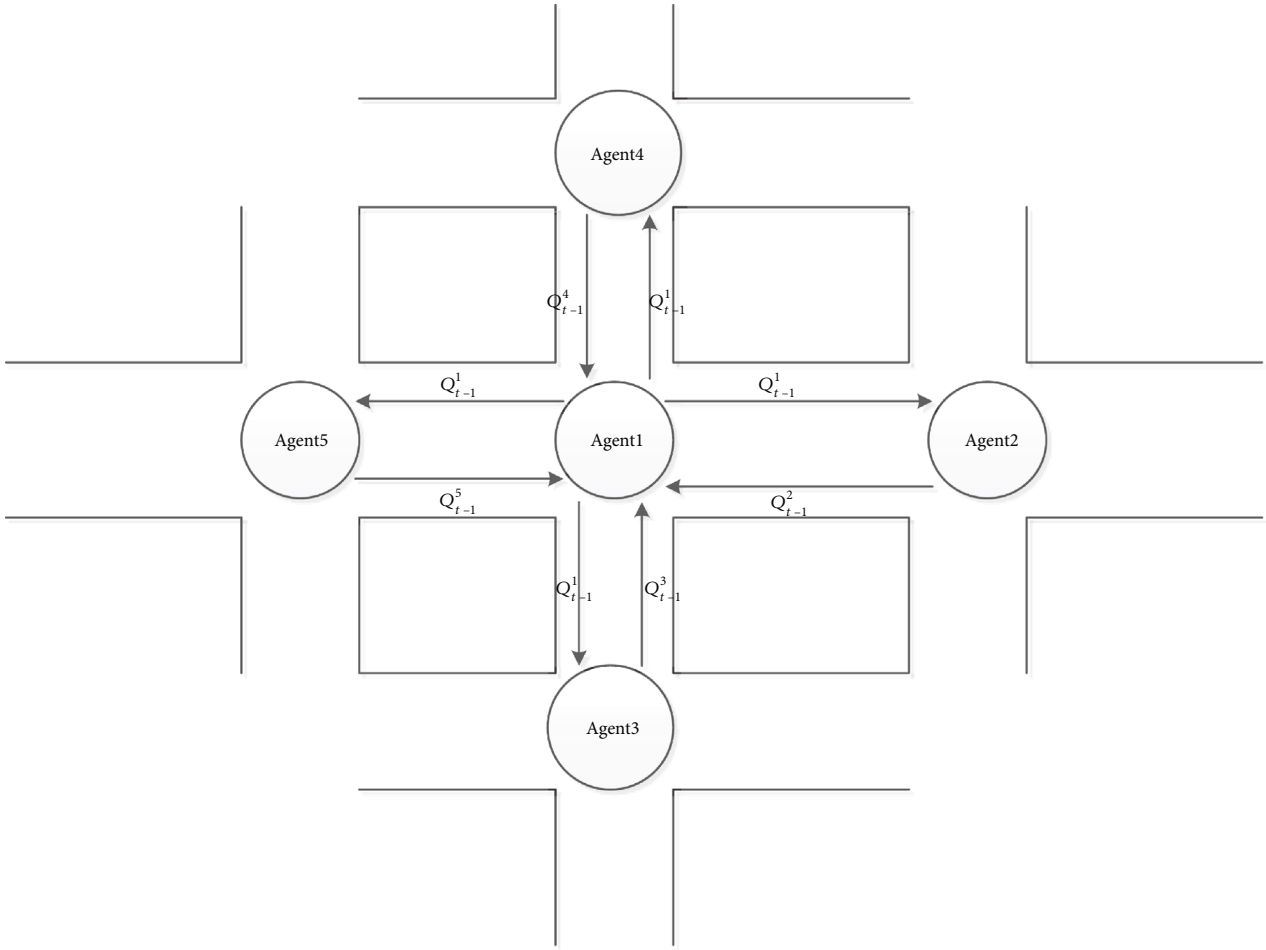


Figure 3: The structure of the CG-MARL.

complex and dynamic in multiagent reinforcement learning, which brings great difficulties to the learning process. Multiagent reinforcement learning is a random game that combines the Nash strategy of the stage game in each state to become an agent's strategy in a dynamic environment. It continuously interacts with the environment to update the $Q$ value function (game reward) in the stage game of each state.

For a random game, it can be written as $(n, S, A_1, \cdots, A_n, Tr, \gamma, R_1, \cdots, R_n)$, where $n$ represents the number of agents, $S$ represents the state space, $A_n$ represents the action space of the $n$-th agent, $Tr : S \times A_1 \times \cdots \times A_n \times S \longrightarrow [0, 1]$ represents the state transition probability, $R_i : s \times A_1 \times \cdots \times A_n \times S \longrightarrow R$ represents the return value obtained by the $n$-th agent under the current state and connected actions, and $\gamma$ represents the cumulative reward discount coefficient. Random games are also Markovian. The next state and reward are only related to the current state and the current connection action.

A multiagent reinforcement learning process is to find the Nash equilibrium strategy for each state and then combine these strategies. $\pi_i : S \longrightarrow A_i$ is an agent's strategy, which selects the best Nash strategy in each state. The optimal strategy of multiagent reinforcement learning (Nash equilibrium strategy of random game) can be written as $(\pi_1^*, \cdots, \pi_n^*)$, $\forall s \in S, i = 1, \cdots, n,$ satisfying

$$V_i(s, \pi_1^*, \cdots, \pi_i^*, \cdots, \pi_n^*) \geq V_i(s, \pi_1^*, \cdots, \pi_i^*, \cdots, \pi_n^*), \forall \pi_i \in \Pi_i, \tag{5}$$

where $V_i(s, \pi_1^*, \cdots, \pi_i^*, \cdots, \pi_n^*)$ is discount cumulative status value function. Abbreviate the above formula to $V_i^*(s)$. Use $Q_i^*(s, a_1, \cdots, a_n)$ to represent the action state discount cumulative value function. In the stage game of each fixed state, $Q_i^*$ is used as the reward of the game to solve the Nash equilibrium strategy. According to the Bellman formula in reinforcement learning, we can get

$$\begin{aligned} V_i^*(s) &= \sum_{a_1, \cdots, a_n \in A_1 \times \cdots \times A_n} Q_i^*(s, a_1, \cdots, a_n)\pi_1^* \\ &\quad \cdot (s, a_1) \cdots \pi_n^*(s, a) Q_i^*(s, a_1, \cdots, a_n) \\ &= \sum_{s'} Tr\left(s, a_1, \cdots, a_n, s'\right)\left[R_i\left(s, a_1, \cdots, a_n, s'\right) + \gamma V_i^*\left(s'\right)\right]. \end{aligned} \tag{6}$$

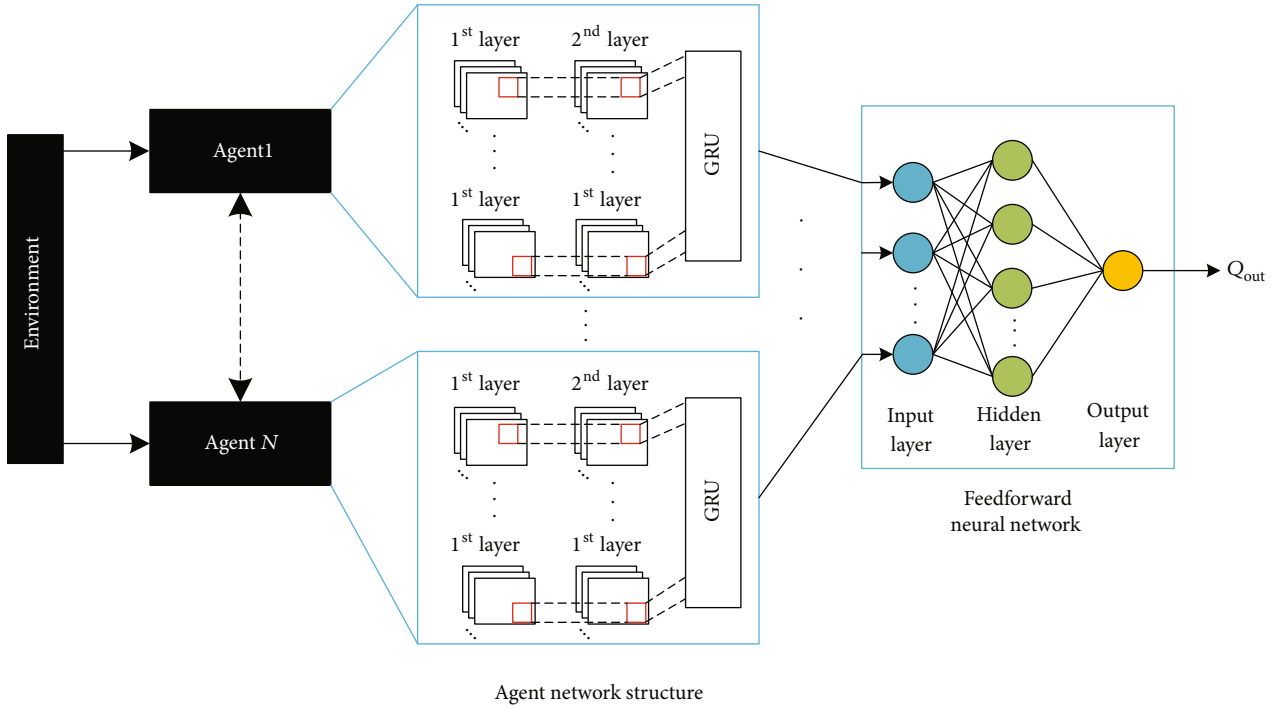


Figure 4: The network structure of CG-MARL.

Initialize $GRU_i$ with random weight $\theta_i > 0$
Initialize feedforward neural network with random weight $\vartheta_i > 0$
Initialize $\varepsilon$, $t_d$
Local_len (hyperparameter: the training time for each episode)
Initialize $Q_0^i = 0$ **for** episode = 1 to $N$ **do**
  **for** $t = 1$ to Local_len **do**
    Observe current intersection state $s_t^i$;
    The agent randomly selects an action with probability $\varepsilon$ and selects an action $a_t^i = \arg \max_{a' \in A} Q_t^i(s_t^i, a'; \theta_i)$ with probability $1 - \varepsilon$;
    Execute action $a_t^i$ and observe all agent reward $r_t^i$ and next state $s_{t+1}^i$;
    Use GRU neural network update $Q_i(s_t^i, a_t^i)$;
    $t = t + t_d$.
  **end**
The global argmax performed on $Q_{\text{out}}$ is the same as a single set of argmax operations performed on each agent. $Q_{\text{out}}$ is updated as (8).
**end**

Algorithm 1: Cooperative game multiagent reinforcement learning for intersections signal control.

The Nash strategy of MARL can be rewritten as

$$\sum_{a_{1,\cdots,a_n} \in A_1 \times \cdots \times A_n} Q_i^*(s, a_1, \cdots, a_n)\pi_1^*(s, a_1) \cdots \pi_n^*(s, a)Q_i^*(s, a_1, \cdots, a_n) \geq \sum_{a_{1,\cdots,a_n} \in A_1 \times \cdots \times A_n} Q_i^*(s, a_1, \cdots, a_n)\pi_1^*(s, a_1) \cdots \pi_i(s, a)Q_i^*(s, a_1, \cdots, a_n). \tag{7}$$

Random games can be classified according to the reward function of each agent. If the reward function of the agent is the same, it is called a fully cooperative game or a team game. If the reward function of the agent is reversed, it is called a perfect competition game or a zero-sum game. In order to solve the random game, it is necessary to solve the stage game of each state, and the reward value of each stage game is $Q(s, \bullet)$.

*3.2. Multiagent Reinforcement Learning Framework.* This paper will use the basic theory of game reinforcement learning to build a framework for multiagent reinforcement learning [26]. The value function of each agent is integrated to obtain a joint action value function. $\tau = (\tau_1, \cdots, \tau_n)$ denotes the joint action-observation history, where $\tau_i = (a_{i,0}, o_{i,1}, \cdots, a_{i,t-1}, o_{i,t})$ is the action-observation history, $a = (a_1, \cdots, a_n)$ is the joint action value function, $Q_i(\tau_i, a_i; \theta_i)$ is the local action value function of agent $i$, and the local value function only depends on the local observation of each agent. The

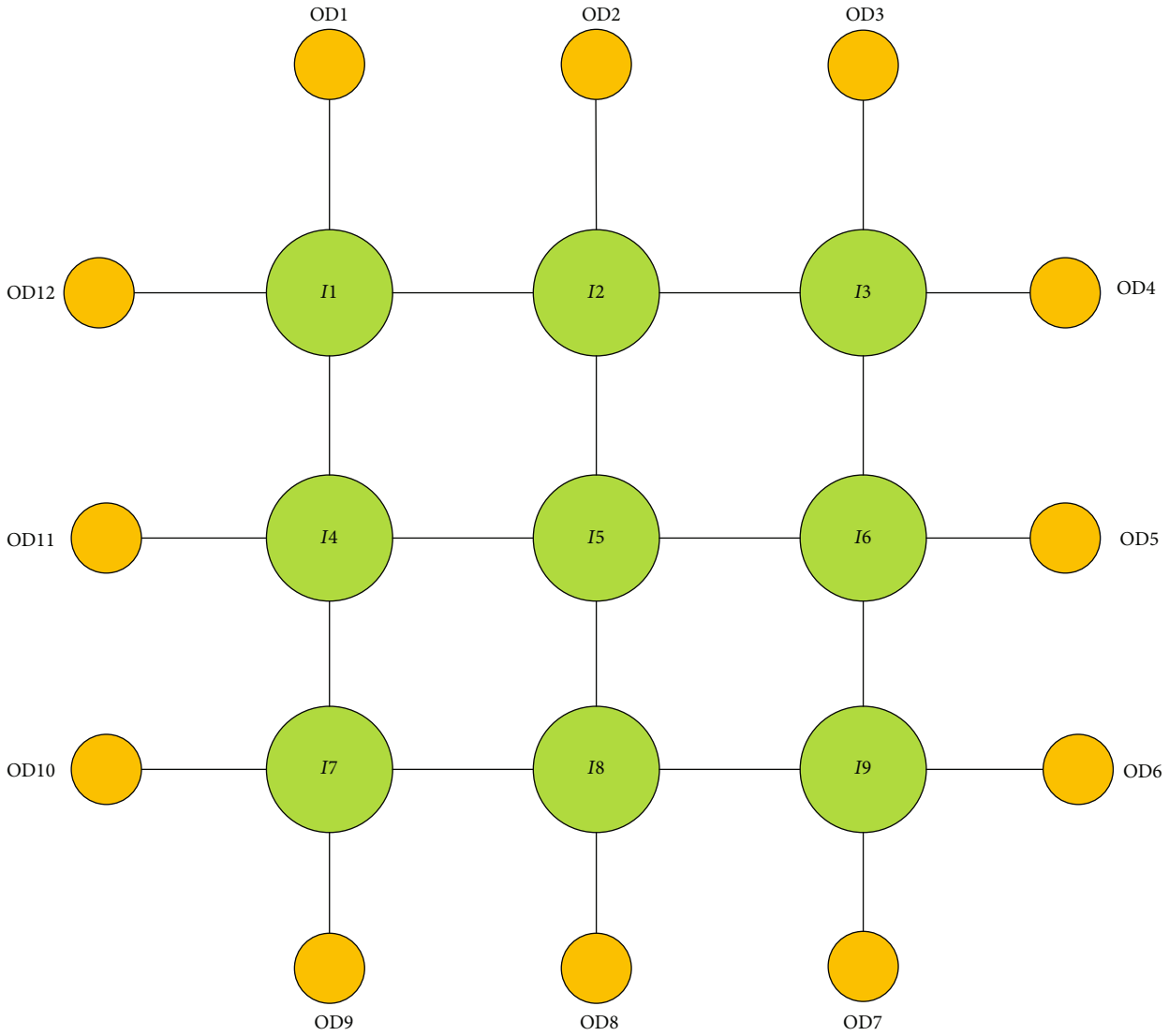


Figure 5: The structure of road network for the nine intersections.

joint action value function is equal to each local action value function, and its monotonicity is the same, as shown in the following formula:

$$\operatorname{argmax} Q_{\text{out}}(\tau, u) = \begin{pmatrix} \operatorname{argmax}_{u_1} Q_1(\tau_1, u_1) \\ . \\ . \\ . \\ \operatorname{argmax}_{u_n} Q_n(\tau_n, u_n) \end{pmatrix} \tag{8}$$

$$\text{s.t.} \quad \frac{\partial Q_{\text{out}}}{\partial Q_i} \geq 0, \quad \forall i \in \{1, 2, \cdots, n\}.$$

*3.3. Model Building.* In order to facilitate the description of signal control problems at multiple intersections, this article takes an intersection as an example. The structure of the intersection is shown in Figure 2, which contains four phases, each with two lanes. The intersection signal lights control and adjust the phase sequence switching to ensure the orderly passage of vehicles.

*3.3.1. State Space.* Traffic signal control mainly depends on the vehicle information at the intersection, that is, the size of the vehicles queued at an intersection. In MAS, the joint state space increases exponentially. If the joint state space is designed according to the number of queues for each phase at each intersection, the result of dimensionality explosion will obviously occur, so the state space of each intersection needs to be simplified. In this article, in order to better describe the traffic state of the intersection, a model is established according to the number of vehicles in the queue and the maximum vehicle capacity of the intersection, where $x$ is the state of the intersection, $l_{\text{que}}$ is the number of vehicles in the queue, $l_{\max}$ is the maximum number of vehicles that

Table 1: OD flow matrix.

| Input OD | Output OD | | | | | | | | | | | | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| 1 | | 229 | 255 | 186 | 136 | 57 | 400 | 308 | 176 | 61 | 61 | 178 | 2047 |
| 2 | 387 | | 310 | 240 | 335 | 119 | 138 | 251 | 162 | 95 | 334 | 159 | 2530 |
| 3 | 276 | 109 | | 333 | 218 | 256 | 208 | 218 | 114 | 59 | 98 | 219 | 2108 |
| 4 | 383 | 220 | 359 | | 146 | 280 | 77 | 217 | 398 | 157 | 326 | 108 | 2671 |
| 5 | 303 | 262 | 173 | 65 | | 271 | 219 | 247 | 233 | 145 | 182 | 199 | 2299 |
| 6 | 290 | 227 | 100 | 379 | 343 | | 387 | 203 | 188 | 301 | 57 | 358 | 2833 |
| 7 | 131 | 77 | 205 | 281 | 267 | 94 | | 75 | 264 | 161 | 244 | 186 | 1985 |
| 8 | 291 | 59 | 126 | 108 | 180 | 57 | 214 | | 67 | 211 | 303 | 366 | 1982 |
| 9 | 174 | 232 | 250 | 271 | 389 | 86 | 118 | 370 | | 221 | 256 | 69 | 2436 |
| 10 | 142 | 221 | 286 | 281 | 187 | 304 | 124 | 339 | 60 | | 217 | 285 | 2446 |
| 11 | 227 | 346 | 144 | 398 | 289 | 394 | 283 | 140 | 279 | 345 | | 179 | 3024 |
| 12 | 67 | 199 | 196 | 112 | 269 | 389 | 139 | 171 | 374 | 358 | 315 | | 2589 |
| Total | 2671 | 2181 | 2404 | 2654 | 2759 | 2307 | 2307 | 2539 | 2315 | 2114 | 2393 | 2306 | |

can be queued, and $\varphi$ is the queuing evaluation parameter (this article takes the value 0.5). Each intersection agent has four phases, and the combined state of the four phases is $X = [x^1, x^2, x^3, x^4]$, that is, the joint state space of the entire regional traffic can be expressed as $X = [X_1, X_2, \cdots, X_n]$.

$$x = \begin{cases} 0, & \dfrac{l_{\text{que}}}{l_{\max}} < \varphi, \\ 1, & \dfrac{l_{\text{que}}}{l_{\max}} \geq \varphi. \end{cases} \tag{9}$$

*3.3.2. Action Space.* Action space means that the agent selects an action $a_i \in A_i$ after observing the state of intersection $i$ at time step $t$. $A_i$ refers to the collection of all actions of the agent and then executes the selected action. In this article, the possible action is the traffic signal phase configuration. The intersection as shown in Figure 1 can be set up with four different actions according to the four phases, which are turn left from east to west, go straight from east to west, turn left from north to south, and go straight north-south. The time for each execution action is a fixed minimum unit time interval with a length of $\tau$. At time step $t + 1$, the agent observes the new state affected by the latest operation and selects the next operation. The agent can take the same action at time step $t + 1$ and $t$.

*3.3.3. Reward.* The reward function is to evaluate how well the actions performed by the agent interact with the environment affect the environment. The agent first observes the state of the external environment, selects, and executes a preset action. Then, the environment feedbacks the effect of the executed action on itself to the agent, so the reward function can give the agent scalar feedback information. The agent is looking for strategies in the direction of reward maximization. There are many reward mechanisms in traffic signal control, such as the cumulative delay time of the vehicle, the flow rate of the vehicle, and the throughput of the vehicle. In this article, the change of queued vehicles is used as the reward function, which is defined as follows:

$$r_i\left(s_t^i, a_t^i\right) = l_t^i - l_{t+1}^i, \tag{10}$$

where $l_t^i$ and $l_{t+1}^i$ are the average queue length at time $t$ and $t + 1$, respectively, and are the reward function at time $t$. It can be seen from the function expression that if the average long queue of vehicles at time $t + 1$ is less than that at time $t$, the function value is positive, which means that the currently executed action has a positive impact on the current traffic state.

*3.4. Cooperative Game Multiagent Reinforcement Learning (CG-MARL) Algorithm.* In this section, a multiagent distributed learning algorithm based on cooperative games is proposed to coordinate the signal control of multiple intersections. CG-MARL establishes an agent network model for all intersections in an area. Each agent has a DQN to maintain the control of each intersection and tries to find the optimal strategy solution in a dynamic environment. At the same time, it interacts with neighboring intersection agents to achieve the purpose of distributed and coordinated control of the entire area signal. The plane structure of CG-MARL is shown in Figure 3.

In CG-MARL, the $Q$ value of the neighboring agent will be transferred to the local agent for policy learning, and then, the $Q$ value of each agent's learning result will be uploaded to the hybrid network for further cooperative game solving so that the multiagent system can coordinate control of all intersections in the entire area. The action selection of an intersection not only considers its own $Q$ value but also depends on the $Q$ value of its neighbors. The overall system considers the $Q$ value of all agents in the area. This cooperation mechanism is conducive to balancing the traffic flow between intersections and improving the overall performance of the regional transportation network.


Figure 6: Game equilibrium between two intersection agents.

The input of the entire network is the discrete state coding of intersection traffic information, and the output is a vector formed by the estimated $Q$ values of all actions in the observation state. GRU estimation can use gradient-based training algorithms to automatically extract features from the original traffic state of the intersection and approximate the $Q$ function. Specifically, firstly, each agent has its own separate network representation value function. Through the deep $Q$ network, the input at each time step is the current observation result of the environment and the result of the last time step operation. Secondly, the output of each agent is integrated into a hybrid network. The hybrid network is a feedforward neural network, which takes the output of each agent as input and monotonically mixes it to produce the output of the entire area network. It is worth noting that in order to emphasize the monotonicity constraint, the weight of the hybrid network must be restricted to nonnegative numbers, which allows the hybrid network to approximate any monotonic function arbitrarily. The structure of the entire network is shown in Figure 4.

The loss function of the entire network is

$$\text{Loss} = \sum_{i=1}^{n}\left[\left(y_i^{\text{out}} - Q_{\text{out}}(s, a; \theta)\right)^2\right]. \tag{11}$$

The pseudocode is shown in Algorithm 1. At each time step $t$, the state observed by agent $i$ is input into the evaluation network. The agent $i$ chooses an action performed by the $\alpha$-greedy method according to the output $Q$ value. The agent gets the reward and enters the next state. For each agent $a$, there is an agent network representing its individual value function. We denote the proxy network as RQN, and they receive the input of the current personal observation result and the last operation at each time step, as shown in Figure 4.

The parameter of GRU is updated by the stochastic gradient descent algorithm. A schematic diagram of the training process is also shown in Figure 4. Since the cooperative training agent hybrid network, the weight is generated by a separate super network. Each super network takes the state as input and generates the weight of one layer of the hybrid network. It is composed of a single linear layer, followed by an absolute activation function to ensure that the weight of the hybrid network is nonnegative.

Then, the output of the super network is a vector, which is shaped into a matrix of appropriate size. The bias is generated in the same way, but is not limited to nonnegative. The final deviation is generated by the nonlinear 2-layer super network. The state is used by the super network instead of being passed directly to the hybrid network, because $Q_{\text{out}}$

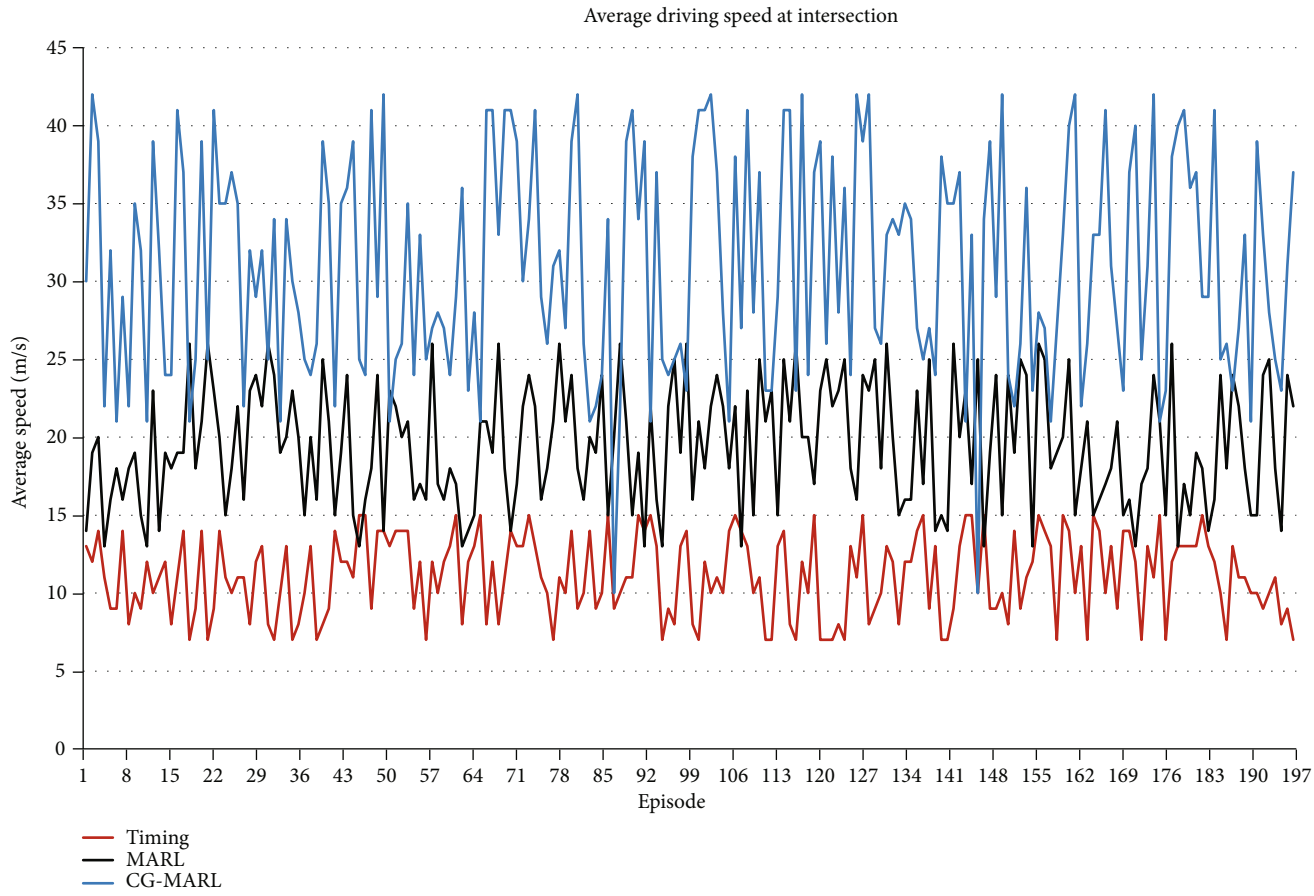


Figure 7: Average driving speed at intersection.

allows to rely on additional state information in a nonmonotonic way. Therefore, overconstraint passes certain function of $s$ along with the single-agent value through the monotonic network. On the contrary, the use of the super network can adjust the weight of the monotonic network in any way, merging the complete state into the joint action value estimate.

## 4. Results and Discussion

In order to verify the feasibility and superiority of the proposed algorithm in multi-intersection signal control, the experiment mainly establishes a traffic model and compares it with other algorithms. All traffic flow models are established based on the cell transmission model (CTM), which uses finite differences to design an approximate method for the macroscopic road traffic flow model. It has a huge advantage in response to the traffic flow characteristics with large fluctuations. Three evaluation indicators are used in the experiment: unit travel speed, that is, the speed of the vehicle in the lane (km/s); vehicle delay per unit time, that is, the average delay time of the vehicle in the lane (veh/s); vehicle emission per unit time, that is, the average $COx$ pollutants emitted by the vehicle in the lane.

This paper uses a $3 \times 3$ grid network as the experimental simulated road network environment. The specific structure is shown in Figure 5. The number $I$ represents the intersection node, the number OD represents the input node of the network, the connection between each node is represented as a two-way driving lane, the length is 800 m, the capacity is 1800 veh/h, and the average length of the vehicle is 4 m.

In the experiment, the arrival situation of the traffic flow is random. Each vehicle arrives at an intersection and turns to the next intersection at random. The turning probability of each intersection is set to 0.3~0.7. In order to simulate low, medium, and high flow rates, the flow rate in each OD direction in Table 1 is adjusted with a ratio of 1 to 2, and the control effect of CG-MARL under different flow rates is analyzed.

Figure 6 shows the game balance result diagram of two intersection agents. In this figure, we can see that the two intersection agents gradually reach a stable state. It illustrates the game balance is reached.

Figure 7 shows the average speed of vehicles at all intersections in the simulation step. Note that the traveling speed of the vehicle in Figure 7 represents the average speed of the vehicle traveling at all intersections due to the switching of traffic lights. If the length of the green band at the intersection can be long enough, the speed of the passing vehicle will not decrease, and the speed will be relatively faster. The CG-MARL algorithm can improve speed of the vehicle. Because the nature of reinforcement learning can optimize control actions, more vehicles are not waiting at intersections, thereby increasing the speed of the entire process. If we

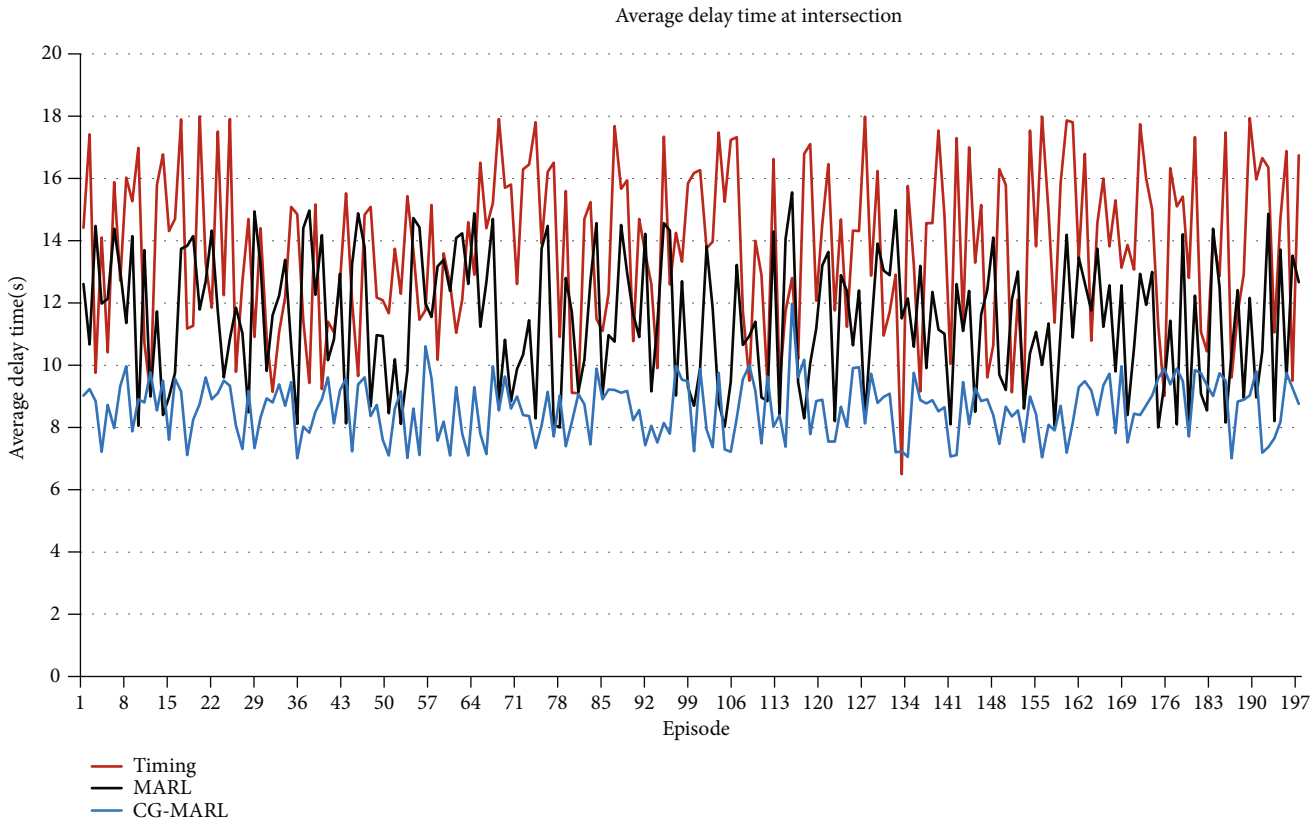


Figure 8: Average delay time at intersection.

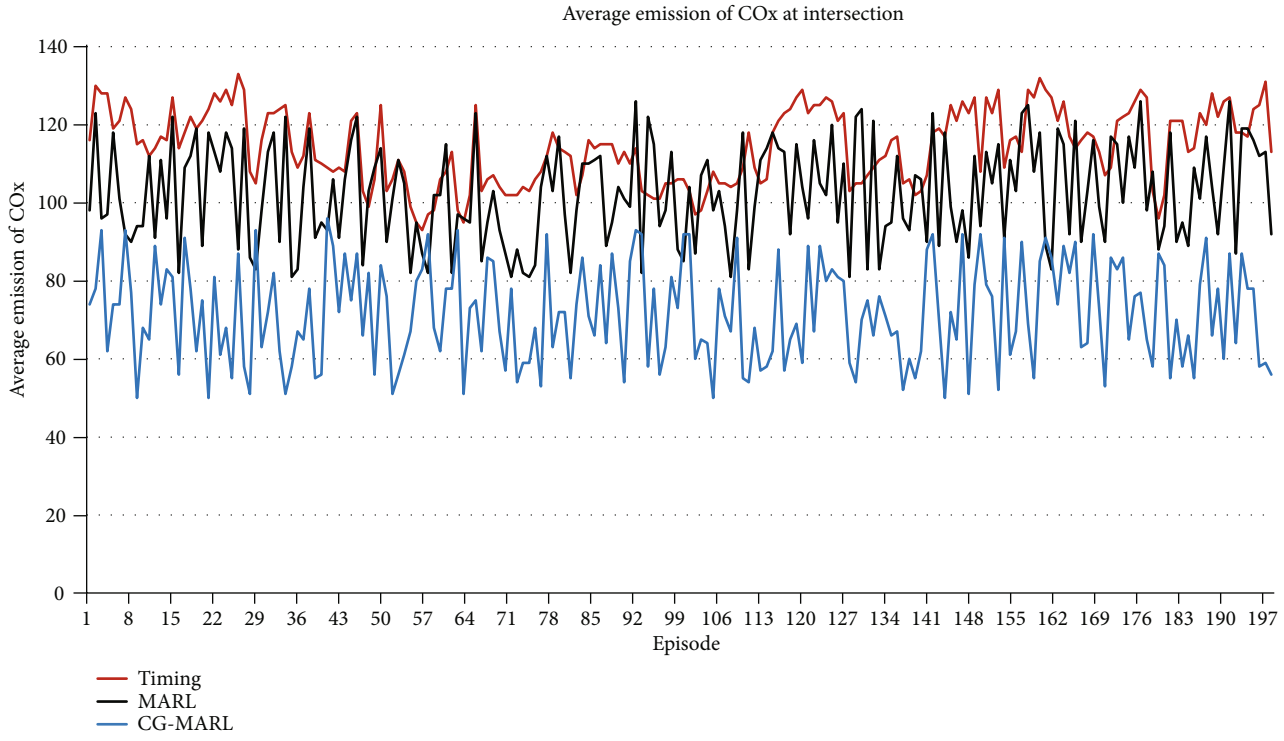


Figure 9: Average emission of *COx* at intersection.

compare the performance between the CG-MARL algorithm proposed in this paper and the traditional MARL algorithm, we can observe that our solution is better than MARL, and the average speed increased by 43.02%. The main reason is that our solution also considers that our solution exchanges information between multiple agents, which is very useful for realizing the optimization of the entire system. In the traditional MARL algorithm, the agent only obtains the information of its own environment and does not pay attention to the information interaction between the agents.

Figure 8 shows the comparative result of the average delay time. Likewise, our solution is superior to other solutions. In the results, the vehicle delay time is the most direct performance indicator, because the traffic light control algorithm should always reduce the vehicle delay time at the intersection. The average delay time is reduced by 26.59%. As the delay time is shortened, unnecessary parking and traffic congestion are reduced.

Figure 9 shows a comparison emission of *COx*. We can see that our solution is better than others. The result shows that when the delay time of vehicles at the intersection is reduced, the *COx* emission will also be relatively reduced, and the average *COx* emission is reduced by 32.18%. With the reduction of carbon dioxide emissions, it means that the waiting time for the red light at the intersection will also be reduced.

## 5. Conclusions

This paper proposes a cooperative game multiagent reinforcement learning (CG-MARL) for regional traffic control. CG-MARL can extract status information effectively at intersections and enable multiple intersections to control traffic conditions based on regional coordination signals. In the framework of learning, agents can exchange information with each other and then concentrate on learning to achieve the goal of regional coordination. We can train hierarchical architecture efficiently by using training models from simple tasks. Importantly, the proposed CG-MARL framework can be extended to have different intersection structures and numbers. The results show that the method proposed in this paper can greatly improve the operation efficiency of vehicles and reduce the delay time of vehicles and the emission of pollutants to a certain extent, respectively. At present, this algorithm is limited to traffic single objective optimization. In future work, the algorithm can be extended to multiobjective signal control and can be combined with traffic assignment algorithm.

## Data Availability

The data are laboratory data. They are generated by simulation software.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## References

[1] Y. S. Chang, Y. J. Lee, and S. S. B. Choi, "Is there more traffic congestion in larger cities? -scaling analysis of the 101 largest U.S. urban centers-," *Transport Policy*, vol. 59, pp. 54–63, 2017.

[2] A. J. Miller, "Settings for fixed-cycle traffic signals," *Operational Research Quarterly*, vol. 14, no. 4, pp. 373–386, 1963.

[3] B. De Schutter, "Optimal traffic light control for a single intersection," *European Journal of Control*, vol. 3, no. 10, pp. 2195–2199, 1999.

[4] P. Jiao, T. Sun, D. Li, H. Guo, R. Li, and Z. Hou, "Real-time traffic signal control for intersections based on dynamic O–D estimation and multi-objective optimisation: combined model and algorithm," *IET Intelligent Transport Systems*, vol. 12, no. 7, pp. 619–630, 2018.

[5] J. García-Nieto, A. C. Olivera, and E. Alba, "Optimal cycle program of traffic lights with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 6, pp. 823–839, 2013.

[6] Y. Bi, X. Lu, Z. Sun, D. Srinivasan, and Z. Sun, "Optimal type-2 fuzzy system for arterial traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 9, pp. 3009–3027, 2018.

[7] D. Srinivasan, M. C. Choy, and R. L. Chen, "Neural networks for real-time traffic signal control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 261–272, 2006.

[8] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[9] C. A. Helm, W. Knoll, and J. N. Israelachvili, "Intelligent traffic light control," *Proceedings of the National Academy of sciences of the United States of America*, vol. 88, no. 18, pp. 8169–8173, 1991.

[10] C. Jacob and B. Abdulhai, "Automated adaptive traffic corridor control using reinforcement learning: approach and case studies," *Transportation Research Record*, vol. 1, no. 19, pp. 1–8, 2006.

[11] J. Chen and X. Ma, "Adaptive group-based signal control by reinforcement learning," *Transportation Research Procedia*, vol. 10, no. 15, pp. 207–216, 2015.

[12] C. Wan and M. Hwang, "Value-based deep reinforcement learning for adaptive isolated intersection signal control," *IET Intelligent Transport Systems*, vol. 12, no. 9, pp. 1005–1010, 2018.

[13] P. Ha-li and D. Ke, "An intersection signal control method based on deep reinforcement learning," in *2017 10th International Conference on Intelligent Computation Technology and Automation (ICICTA)*, pp. 344–348, Changsha, China, 2017.

[14] Y. Liu, L. Liu, and W. Chen, "Intelligent traffic light control using distributed multi-agent Q learning," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–8, Yokohama, 2017.

[15] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intelligent Transport Systems*, vol. 4, no. 2, pp. 128–135, 2010.

[16] T. Chu, J. Wang, and J. Cao, "Kernel-based reinforcement learning for traffic signal control with adaptive feature selection," in *53rd IEEE Conference on Decision and Control*, pp. 1277–1282, Los Angeles, CA, 2014.

[17] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[18] M. Aslani, M. S. Mesgari, and M. Wiering, "Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events," *Transportation Research Part C: Emerging Technologies*, vol. 85, no. 10, pp. 732–752, 2017.

[19] L. Li, Y. Lv, and F. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 247–254, 2016.

[20] H. Ge, Y. Song, C. Wu, J. Ren, and G. Tan, "Cooperative deep Q-learning with Q-value transfer for multi-intersection signal control," *IEEE Access*, vol. 7, pp. 40797–40809, 2019.

[21] I. Lamouik, A. Yahyaouy, and M. A. Sabri, "Smart multi-agent traffic coordinator for autonomous vehicles at intersections," in *2017 International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, pp. 1–6, Fez, Morocco, 2017.

[22] T. Wu, P. Zhou, K. Liu et al., "Multi-agent deep reinforcement learning for urban traffic light control in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 8243–8256, 2020.

[23] J. C. Medina and R. F. Benekohal, "Traffic signal control using reinforcement learning and the max-plus algorithm as a coordinating strategy," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pp. 596–601, Anchorage, AK, USA, 2012.

[24] A. E. Ohazulike and T. Brands, "Multi-objective optimization of traffic externalities using tolls," in *2013 IEEE Congress on Evolutionary Computation*, pp. 2465–2472, 2013.

[25] A. Agliari, C. H. Hommes, and N. Pecora, "Path dependent coordination of expectations in asset pricing experiments: a behavioral explanation," *Journal of Economic Behavior & Organization*, vol. 121, pp. 15–28, 2016.

[26] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning," *International Conference of Machine Learning*, vol. 80, pp. 4295–4304, 2018.