Intelligent Traffic Signal Control Based on Reinforcement Learning with State Reduction for Smart Cities

LI KUANG, JIANBO ZHENG, and KEMU LI, Central South University, China
HONGHAO GAO, Shanghai University, China and Gachon University, South Korea

Efficient signal control at isolated intersections is vital for relieving congestion, accidents, and environmental pollution caused by increasing numbers of vehicles. However, most of the existing studies not only ignore the constraint of the limited computing resources available at isolated intersections but also the matching degree between the signal timing and the traffic demand, leading to high complexity and reduced learning efficiency. In this article, we propose a traffic signal control method based on reinforcement learning with state reduction. First, a reinforcement learning model is established based on historical traffic flow data, and we propose a dual-objective reward function that can reduce vehicle delay and improve the matching degree between signal time allocation and traffic demand, allowing the agent to learn the optimal signal timing strategy quickly. Second, the state and action spaces of the model are preliminarily reduced by selecting a proper control phase combination; then, the state space is further reduced by eliminating rare or nonexistent states based on the historical traffic flow. Finally, a simplified Q-table is generated and used to optimize the complexity of the control algorithm. The results of simulation experiments show that our proposed control algorithm effectively improves the capacity of isolated intersections while reducing the time and space costs of the signal control algorithm.

CCS Concepts: • **Computing methodologies** → **Reinforcement learning**; • **Applied computing** → *Transportation;*

Additional Key Words and Phrases: Traffic signal control, Q-learning

ACM Reference format:
Li Kuang, Jianbo Zheng, Kemu Li, and Honghao Gao. 2021. Intelligent Traffic Signal Control Based on Reinforcement Learning with State Reduction for Smart Cities. *ACM Trans. Internet Technol.* 21, 4, Article 102 (July 2021), 24 pages.
https://doi.org/10.1145/3418682

1 INTRODUCTION

As the economy and society develop, the number of vehicles in cities has increased rapidly [19]; consequently, traffic congestion has become a major bottleneck that restricts city economic

This work is supported by National Natural Science Foundation of China under Grant No. 61772560 and No. 61902236 and Natural Science Foundation of Hunan Province under Grant No. 2019JJ40388.
Authors' addresses: L. Kuang, J. Zheng, and K. Li, Central South University, School of Computer Science and Engineering, Changsha, China; emails: {kuangli, zhengjianbo}@csu.edu.cn, 674925312@qq.com; H. Gao, Shanghai University, School of Computer Engineering and Science, Shanghai, China, Gachon University, Gyeonggi-Do, South Korea; email: gaohonghao@shu.edu.cn.

1533-5399/2021/07-ART102 $15.00
https://doi.org/10.1145/3418682

development. In urban traffic situations with extremely limited road resources [15], the primary method for solving traffic congestion is to control traffic signals at intersections in the road network. Optimizing the signal timing plan at an intersection achieves higher traffic throughput, improves the traffic network capacity [14], and thus reduces road congestion. Therefore, traffic signal control has become an important research topic in modern traffic management and intelligent transportation, where the goal is to scientifically and reasonably plan signal timing at intersections.

The advent of artificial intelligence technology has made it possible to process a large volume of vehicular data and enabled support for intelligent services in smart cities, such as traffic control. At present, traffic control methods can be divided into four categories: fixed time control strategies, predictive control strategies (both based on historical traffic data) [13, 20], actuated signal control strategies [11, 50], and adaptive control strategies [18] based on real-time traffic flow data.

- **The fixed time control strategy** is currently the most commonly used. Under this strategy, to reduce vehicle delays, signal timing plans for different days and for different times are calculated based on historical traffic data via an optimization algorithm.
- **The predictive signal control strategy** predicts future traffic flow based on historical data using a neural network or another classification algorithm; then, the traffic lights are timed based on the predicted traffic flow to reduce vehicle delays at the intersection.
- **The actuated signal control strategy** uses sensors to acquire vehicle arrival information and extends the green light time of the current phase based on the real-time traffic situation to improve the traffic capacity of the intersection.
- **The adaptive signal control strategy** systematically and automatically adjusts the timing of traffic lights in real time. First, sensors acquire the current traffic demand at the intersection; then, control algorithms assign a reasonable green light time to each phase to improve the traffic capacity of the intersection.

Because real-life traffic flows are time-varying and random, establishing an accurate mathematical model is difficult. As the number of vehicles in cities increases, it becomes more difficult to achieve the desired signal control effect for single intersections using the traditional strategies. Consequently, experts and scholars have begun to apply artificial intelligence methods such as neural networks, fuzzy control algorithms, evolutionary algorithms, and reinforcement learning algorithms to the signal control process for isolated intersections. These approaches can improve the control effect to a certain extent.

As one of the adaptive control methods, signal control algorithms based on reinforcement learning can not only adjust the timing plan based on the current traffic demand but can also adapt to other changing factors in the environment. When one or more of these factors (such as the average speed of vehicles) changes significantly, the agent learns continuously from the new environment based on the optimal strategy of the previous environment to acquire the optimal strategy for the current environment guided by a reward function.

Under agent control technology, a signal at an isolated intersection is controlled by one agent while signals at regional intersections are coordinated by multiple agents. Isolated intersections are the basic building blocks of urban road networks and lie at the core of road control. However, the current approaches for signal control at single intersections based on reinforcement learning still have the following shortcomings:

(1) In real life, each isolated intersection is controlled by an independent agent, and many isolated intersections exist in the road network. Due to budget limitations, the computing resources available at each isolated intersection are very limited; therefore, the complexity of control algorithms must be tightly controlled, especially for distributed multiagent signal

coordination control. However, the existing studies generally focus on improving the control effect and ignore optimizing algorithm complexity, leading to excessive execution times and high storage consumption, which is not practical in real environments.

(2) Most of the existing reward functions in reinforcement learning-based signal control methods seek to minimize the delay [38, 45] or the queue length [5] of vehicles at intersections. However, the matching degree between signal timing and the traffic demand has not been fully considered, and the allocated green time often does not match that required by the current state; consequently, the optimal strategy is hard to learn, and the learning efficiency is relatively low.

In this article, we propose a traffic signal control method for the phase timing of isolated intersections based on reinforcement learning with state reduction. We discretized the state of the phase lane into K-types based on historical vehicle data; then, the reinforcement learning model is constructed, and its state, action, and double-goal reward function are defined. Next, we propose further improving the reinforcement learning model using two steps: the state and action spaces are initially reduced by selecting a proper control phase combination; then, the state space is further reduced by eliminating rare states from the historical traffic flow. Finally, simulation experiments verify that the proposed algorithm can reduce the space and time cost effectively while guaranteeing the signal control effect, and the training time of the algorithm can be greatly reduced.

The remainder of this article is organized as follows: In Section 2, we introduce the related works in the field of intersection traffic control. In Section 3, we introduce the proposed reinforcement learning model based on state reduction in detail. In Sections 4 and 5, we present the experimental settings and verify the proposed algorithm both through a theoretical proof and through experiments. Section 6 provides conclusions and suggestions for future work.

2 RELATED WORK

At present, research works related to traffic signal control systems are divided into three main types: regional signal control, trunk signal control, and isolated intersection signal control.

2.1 Regional Signal Control

The coordinated control of regional intersections [8] takes all intersections in the entire local road network as the control object and seeks to reduce the average vehicle delay in the region by coordinating the capacity of each intersection. Hong et al. proposed a multi-input and multi-output traffic signal control method that established the intersection control model in a globalized setting using MIMO linear control theory and high matrix formulation. The result of a simulation showed that the MIMO control method achieves a much shorter average travel delay compared with pretimed and actuated controls [35]. Prabuchandran et al. proposed a distributed reinforcement learning method that solved the dimensional explosion problem by discrete processing of the state space. Then, they constructed a joint function that acted as the reward function of the reinforcement model based on the feedback obtained from itself and other agents to guide each agent to learn the globally optimal strategy [28]. Chen et al. proposed an adaptive layer signal control method based on reinforcement learning. Based on traditional Q-learning, this method increased action adaptation to optimize the action space and reduced the cost of the control algorithm [7]. However, the current research on coordinated control of regional signals does not conduct a comprehensive and effective analysis of isolated intersections as basic control units; thus, the proposed regional control algorithm has a high space-time complexity that requires abundant computing resources and has lengthy execution times. Most of the existing research on regional signal control focuses on improving the effect of signal control but neglects the deep excavation of regional traffic flow data, while the optimization of control algorithms based on traffic flow theory has rarely

been studied. Common algorithms for regional signal control include multiagent algorithms [1, 25], evolutionary algorithms [4], and reinforcement learning algorithms, including Q-learning, Deep Q-learning [46], and so on.

2.2 Trunk Signal Control

Trunk intersection control takes all intersections on the city's main roads as the control object, coordinating the green light timing of each intersection to form a "green wave" to reduce the average delay of vehicles on the main line. Luo et al. proposed a multiobjective immune algorithm based on multiple groups to implement coordinated traffic signal control on urban arterial roads. They considered the traffic delays of vehicles on the main road and established an objective function for the total delay; then, the minimum delay was found by adjusting the phase difference to achieve the goal of reducing traffic delays [24]. Kazuhiro et al. adopted an unbalanced dual-channel control method to achieve a better green wave control effect on the main line. They reduced the total vehicle delay by choosing the appropriate travel route to balance the travel time [33]. Trunk control mines traffic flow laws in the trunk line and optimizes the control algorithm accordingly to minimize the wait time of the traffic flow on the main road when passing through each intersection, thus forming a "green wave control." Trunk control can be regarded as an upgraded version of timing control, but under this approach, it is still impossible to make changes to traffic changes and react in real time. At present, few studies have investigated intersection trunk control. Generally, evolutionary algorithms are used to determine green light timing and phase difference to implement trunk control based on historical traffic flow information.

2.3 Isolated Intersection Signal Control

Isolated intersections are not only the basic component of urban road networks but also the core of urban road network control. Efficient signal control for isolated intersections is the basis for effective trunk signal control and regional signal coordinated control. Some problems still exist in isolated intersection signal control, including vehicle intersection transit time or waiting time being too long due to inadequate use of intersection capacity. At present, to determine a signal timing scheme that conforms to the current state, many scholars are committed to making isolated intersections more aware of their own traffic conditions. By optimizing the traffic flow of the intersection, the traffic capacity of the isolated intersection can be improved, the average waiting time of vehicles can be reduced, and traffic congestion can be avoided or alleviated. To obtain an optimal signal timing scheme, the common algorithms for signal control at isolated intersections generally include timing control algorithms, actuated control algorithms [21, 30], optimal control algorithms [47, 49], fuzzy control algorithms [2, 26], and reinforcement learning algorithms [5, 9].

Timing control involves setting a fixed time-period signal timing scheme based on historical traffic flow data; however, this approach results in a poor signal control effect. To enhance the effect of signal control based on timing control, some proposed actuated control algorithms use sensors to sense vehicle arrival, such as SCOOT [17] and SCATS [23]. Due to the extremely low cost, timing control and actuated control are currently widely used worldwide. However, timing control and actuated control achieve relatively poor control effects and have gradually faded out of the main research trend. To improve the traffic control effect, some researchers introduced optimal control algorithms that optimize for the lowest delay or the shortest queue length into the signal control field. Alok et al. proposed a simulation-based optimization approach that can guarantee the result and improves algorithm scalability [27]. The optimal control algorithm attempts to establish an accurate mathematical model for the traffic flow of isolated intersections [34]. However, real-life traffic flow varies randomly over time, which makes it difficult to establish an accurate mathematical model. Moreover, the model solution time is too long to meet the changing

traffic demand at intersections in real-time, which causes the optimal control algorithm to become a bottleneck. To address the above problem, the researchers introduced a fuzzy control method into the field of signal control. Rahib et al. found that fuzzy control based on a fuzzy logical system is one of the most effective areas of fuzzy systems and applied it to simulated control of an isolated intersection [2]. Luo et al. proposed a self-adaptive timing model with a single-target constraint to reduce intersection delay. This model works via a fuzzy genetic algorithm [29]. Although fuzzy control algorithms avoid the need to establish accurate mathematical models, they lack learning ability and cannot adapt to environmental changes. When the traffic situation (such as vehicle speed) changes persistently, fuzzy control does not achieve a good control effect. Using fuzzy control technology in trunk signal control or regional signal control is difficult; therefore, researchers began to seek an adaptive traffic signal control system that does not require the establishment of accurate mathematical models but can sense changes in the environment and features self-learning and self-optimization abilities.

One type of adaptive computing and machine learning method is reinforcement learning, which is highly suitable for the signal control domain. In this approach, an agent learns and adapts to the current situation through constant interaction with the environment. Reinforcement learning algorithms do not need to establish an accurate mathematical model. There are two kinds of signal control based on reinforcement learning. One is based on the traditional reinforcement learning algorithms represented by the tabular Q-learning algorithm. Wang et al. proposed a single-intersection control algorithm based on Q-learning. The agent determines the optimal phase sequence based on real-time traffic information to improve intersection efficiency [39]. Araghi et al. proposed a reinforcement learning signal control algorithm that uses the current traffic information to plan the green light time. This study innovatively defined the reward function as the average difference between the traffic flow entering the intersection and the queue length of the corresponding road segment [3]. The other is signal control algorithm is based on a combination of deep learning and reinforcement learning, called **deep reinforcement learning (DRL)**, represented by deep Q-learning. Dunhao et al. proposed a DRL-based framework that uses multiple rewards for multiple objectives to address the difficulty of reinforcing desired behaviors using a single numeric reward [48]. Wade et al. proposed discrete traffic state encoding, which is information-dense. The discrete traffic state encoding is used as the input to a deep convolutional neural network, trained using Q-learning with experience replay [16].

At present, the primary goal of using a reinforcement learning algorithm in the field of signal control is to reduce vehicle delays and improve the signal control effect, which leads to the problems of poor learning efficiency and high resource consumption. In particular, training the deep reinforcement learning algorithm is exceptionally slow due to the large number of calculations in the neural network. In addition, the phenomena of "insufficient training" and "overfitting" can appear. In our method, the entire state space is reduced by mining the hidden-state laws of the traffic flow information. This approach effectively addresses the contradiction between the delay control effect and the control cost of the reinforcement learning algorithm. Compared with the traditional Q-learning algorithm, our method greatly reduces the storage resources and training time of the control algorithm when the control effects on the queue length and waiting time are similar.

3 Q-LEARING FOR TRAFFIC SIGNAL CONTROL

3.1 Preparatory Work

3.1.1 Building an 8-phase Isolated Intersection Model. The problem studied in this article is signal control for isolated intersections. First, we need to establish a standard 8-phase isolated

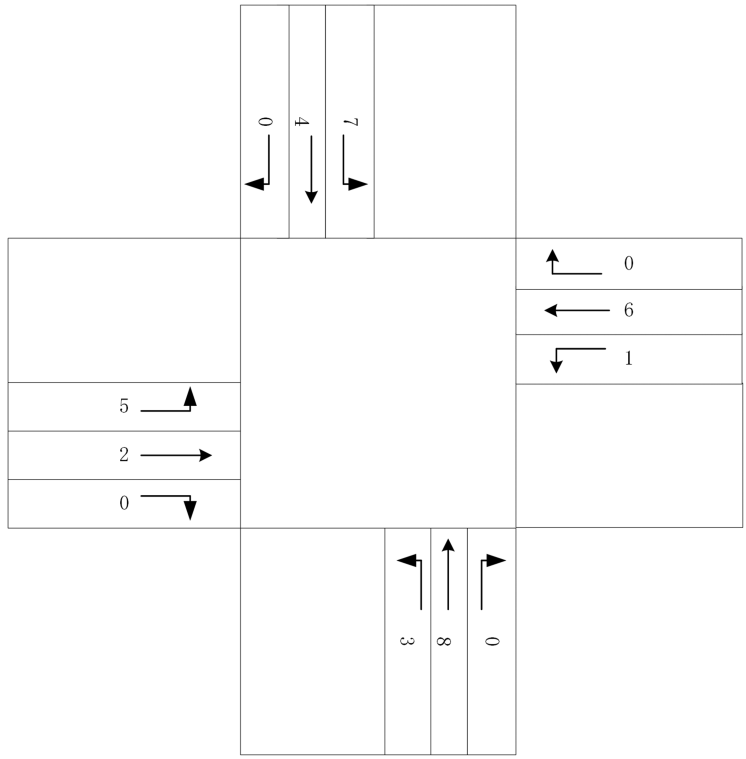

Fig. 1. An 8-phase isolated intersection model.

intersection model. To simplify this problem, we ignore the influence of vehicles in the right-turn lane, because the right-turn lane in every direction is not controlled by the signal lights in most cases. Therefore, we mark the right-turn lane as lane 0. The other eight lanes are controlled by the signal control algorithm and are numbered from 1 to 8, as shown in Figure 1.

A signal phase is the combination of different light colors displayed in the 8 lanes; here, we simply represent it as the lanes with a green light. As shown in Figure 1, there are eight phases for an isolated intersection: (1,5), (2,6), (3,7), (4,8), (1,6), (2,5), (3,8), and (4,7). In other existing studies, these eight phases are considered as a whole. However, when we analyze the phase sequence controlled by the intersection, we find that the eight phases can be divided into two groups of phase combinations, A for (1,5); (2,6); (3,7); (4,8) and B for (1,6); (2,5); (3,8); (4,7), as shown in Figure 2. Any phase combination in the two groups can be used in an agent control cycle to ensure that the traffic flows in each lane of the intersection do not interfere with each other, and each lane can transit the intersection normally. We define the selected phase combinations as the control phase.

3.1.2 Discretized Description of States and Actions. The state of a lane in this article is expressed by the number of vehicles in that lane, while the action is represented by the green light timing during the selected phase. If the number of vehicles and the green light time are used without discretization to represent the state and action, then dimensional explosion will occur. Due to the lack of comprehensive real data support, the state discretization in the existing studies is based largely on researchers' own real-world experiences, which is too subjective.

Therefore, we analyzed the history of the intersection in detail. Regarding vehicle data, the state of the phase lane is discretized into K-types, as shown in Table 1, according to the number

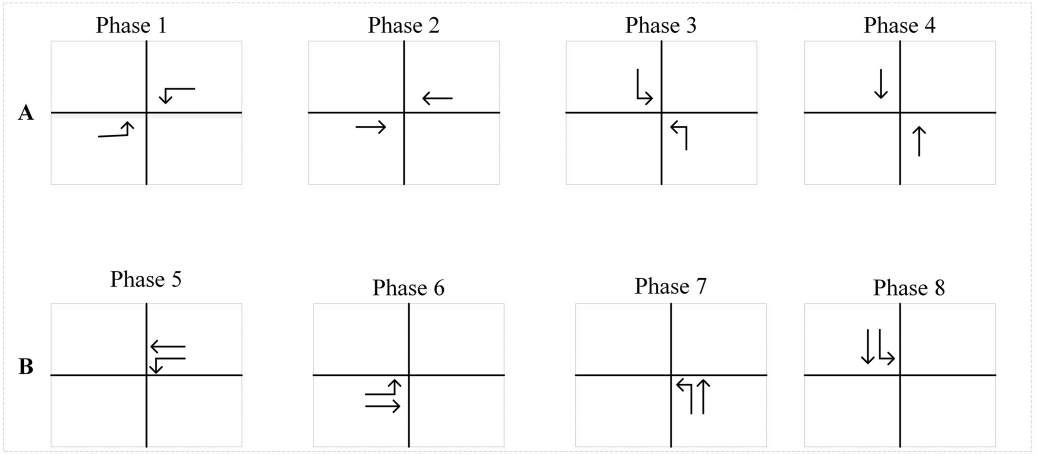

Fig. 2. Phase combinations.

Table 1. Traffic Flow Data Divided into K-classes

Class	1	2	...	k-1	k
Vehicle	$[0,n_1]$	$[n_1+1,n_2]$	...	$[n_{k-2}+1,n_{k-1}]$	$[n_{k-1}+1,n_k]$

of vehicles in a phase lane. In contrast, discretization of the actions is relatively simple. We simply use 1, 2, 3, 4, 5, 6 to represent the actions, which are associated with an assigned green light time of 10 s, 20 s, 30 s, 40 s, 50 s, and 60 s, respectively.

3.2 Modeling Based on Reinforcement Learning

In the traffic signal control scheme, an agent assigns different traffic capacities to different lanes in an intersection to prevent an increase of the total delay caused by mutual interference of the traffic flow in different lanes. Given the initial number of vehicles queued at a certain intersection, the speed of vehicles during the green light time, and the subsequent changes in traffic flow, the core problem of signal control at the intersection can be simplified to determine the green light timing of the phase for each traffic flow state. If the green light time allocated for a phase is too long, then some green light time will remain after the waiting vehicles have all passed through; consequently, the queuing vehicles in the other phases at the intersection must wait too long for the red light to change. Thus, no vehicles will pass through the intersection during this too-long green time, which wastes both traffic capacity and increases the total vehicle delay. Conversely, an allocated green time that is too short also increases the total delay of the vehicles at the intersection.

In the signal control field, there are two advantages to using the reinforcement learning algorithm. First, reinforcement learning can learn and adapt to dynamic changes in the traffic situation, such as traffic accidents. Second, the reinforcement learning algorithm does not need to establish an accurate mathematical model for the environment; instead, the learning process is guided through rewards. In this article, we model the scenario as follows: First, the agent uses the Q-learning algorithm [36] to obtain the optimal green light time for each phase of the intersection in the current state and implements the result as an action. The action leads to a change in the traffic flow, i.e., the current traffic state and an environmental evaluation are returned to the agent. Next, the agent selects the optimal phase green time based on the changed state and starts the next cycle. The state, action, and reward of the reinforcement learning model are defined below.

State. In real life, the state of an intersection may contain much information, such as traffic volume, time, and temperature. It is difficult to use a comprehensive state function to describe all the state changes at the intersection. In our study, we only use the number of vehicles to describe the state of the intersection. In a standard intersection with 8-phase control, the state is generally represented by a vector reflecting the number of vehicles in each of the 8 lanes, as shown in Equations (1) and (2).

$$L = \{l_i | i \in N\}, \tag{1}$$

$$l_i = \langle l_{i1}, l_{i2}, l_{i3}, l_{i4}, l_{i5}, l_{i6}, l_{i7}, l_{i8} \rangle, \tag{2}$$

where l_i indicates the state of the ith time period at the intersection, and l_{ij} indicates the number of vehicles in lane j in the ith time period. Because the intersection performs the A/B phase selection first, which will be explained in Section 3.3, the final state is expressed as shown in Equations (3) and (4):

$$S = \{s_i | i \in N\}, \tag{3}$$

$$s_i = \langle s_{i1}, s_{i2}, s_{i3}, s_{i4} \rangle, \tag{4}$$

where s_i indicates the state of the ith time period at the intersection, and s_{ij} indicates the number of vehicles on phase j in the ith time period.

Action. The traffic is controlled by the green light time allocated to each phase of the intersection. Therefore, we define the agent's action as the green light time allocated for the selected A/B phase combination. The phase order of the actions is consistent with that of the state. To make the action space discrete, we specify that the minimum allocation time for each phase is 10 s and the maximum time is 60 s, at 10-s intervals. Therefore, the choice of action can be represented by 1,2,3,4,5,6, which represent allocating green light times of 10, 20, 30, 40, 50, or 60 s, respectively. Finally, the action of an interaction in a given state can be represented by a vector that includes the green light times allocated for the selected A/B phase combination, where the phase sequence is the same as that of the state, and the A/B phase combination shares the same action space:

$$A = \{a_i | i \in N\}, \tag{5}$$

$$a_i = \langle a_{i1}, a_{i2}, a_{i3}, a_{i4} \rangle, a_{ij} \in \{1, 2, 3, 4, 5, 6\}, \tag{6}$$

where a_i indicates the action performed during the ith state of the intersection, and a_{ij} represents the green light time allocated to the jth phase in the ith time period.

Delay and control cost. There are many definitions of delay. In the ideal case, delay is composed of stable delay, random delay, and overload delay. However, it is difficult to collect all the data needed to calculate the time delay in a real environment; therefore, vehicle waiting time [22, 32] is used to represent the time delay. Based on the definitions of state and action, we then define the delay and control cost of an action a under the state s.

As defined above, the number of vehicles is represented by the state s_i, and the allocated green time is represented by the action a_i. Assume that at a certain time period i, the state of the intersection perceived by the agent is $s_i =< s_{i1}, s_{i2}, s_{i3}, s_{i4} >$. Based on the state s_i, the agent selects the action $a_i =< a_{i1}, a_{i2}, a_{i3}, a_{i4} >$; then, the delay of action a_i is defined as Equation (7):

$$delaycost(s_i, a_i) = \sum_{k=1}^{4} \left(s_{ik} \times \sum_{j=1}^{k-1} (a_{ij}) \right), \tag{7}$$

where $delaycost(s_i, a_i)$ is the action delay of taking the action a_i in the state of s_i, s_{ik} is the number of vehicles in the kth phase, a_{ij} is the green light time allocated to the jth phase, and j varies from

1 to $k-1$. The vehicle waiting time in phase k is the number of vehicles on phase k multiplied by the sum of the allocated green light times from phase 1 to phase $k-1$. The action delay of taking action a_i under state s_i is the sum of waiting times of four phases.

Let $st_i = s_i - a_i =< st_{i1}, st_{i2}, st_{i3}, st_{i4} >$; then, the action control cost $greencost(s_i, a_i)$ of action a_i under state s_i is defined as Equation (8):

$$greencost(s_i, a_i) = \|st_i\| \times \|a_i\|^2, \tag{8}$$

where st_i is the expected state after taking action a_i. When the allocated green light time is too large, causing wasted idle time, or the green light time is too small, preventing most of the waiting vehicles from transiting the intersection, the action control cost will increase.

Reward. The reward function [37, 40, 44] defines the learning target of the reinforcement learning algorithm and directly determines the convergence speed: It is the core part of the reinforcement learning algorithm [6, 41, 43]. The reward value obtained by the reward function is a scalar feedback signal that indicates the value of the agent's performance based on the resulting environment after taking a certain action in a certain state. A larger reward value indicates that the performed action is more accordant with the learning goal. The reinforcement learning algorithm guides the agent through the reward function to learn the action strategy that maximizes the cumulative rewards. A well-designed reward function not only improves the control effect of the agent but also substantially improves the agent's learning efficiency; consequently, the agent can quickly learn the "optimal" action that should be taken in the current state.

Signal control of isolated intersections is a sequence decision problem that is highly applicable to reinforcement learning algorithms [12, 31, 42]. Although the existing reward functions based on the reinforcement learning algorithm can improve the traffic capacity of the intersection, the signal timing often does not fully match the traffic demand. Therefore, we constructed a dual objective reward function. We took the optimization of traffic capacity as the overall goal of signal control and decomposed the overall goal into two parts: delay optimization and green light timing optimization. The first part minimizes the total delay of vehicles in the intersection, and the second part maximizes the matching degree between the signal timing and the traffic demand. The complete reward function R is defined as shown in Equation (9):

$$R_i = \mathrm{M}/\left(\beta \times \mathrm{delaycost}\left(s_i, a_i\right) + (1-\beta) \times \mathrm{greencost}\left(s_i, a_i\right)\right), \tag{9}$$

where R_i is the reward given to the agent at the ith timestep, delaycost(s_i, a_i) is the action delay cost when action a_i is taken in state s_i, and $greencost(s_i, a_i)$ represents the matching cost between the currently selected action ai and the action required by the state si. In this article, β is a weight factor, and M is a magnification factor to ensure that the reward function is not too small.

As the total delay becomes larger, the reward value R becomes smaller; therefore, this reward guides the agent to learn how to reduce the total delay at the intersection. When the allocated green light time matches the green light time required for the current state, the greencost(s_i,a_i) becomes smaller, and the reward value R becomes larger. This reward guides the agent to avoid too-large or too-small green time allocations caused by the pursuit of the shortest time delay.

An illustrating example of reward calculation. In this section, we use an example to explain the reward R calculation in detail. Assume that the current state is $s = [2, 6, 2, 8]$, the selected action is a = [1,2,1,2], and that $s\_$ is the expected new state after the state s takes action a, that is, $s\_ = [1, 4, 1, 6]$. Then, we have

delaycost(s,a)
$= s[2] \times a[1] + s[3] \times (a[1] + a[2]) + s[4] \times (a[1] + a[2] + a[3])$
$= 6 \times 2 + 2 \times 3 + 8 \times 4$

$= 50$
new state $s\_ = [1, 4, 1, 6]$; set $\beta = 0.5$, $M = 5$;
greencost(s,a)
$= (|s\_[1]| + |s\_[2]| + |s\_[3]| + |s\_[4]|) \times (a[1] + a[2] + a[3] + a[4])2 = 12 \times 62$
$= 432$
$R = 10 \div [delaycost(s, a) + greencost(s, a)]$
$= 0.021$

3.3 Optimization of Reinforcement Learning Model

3.3.1 Selecting Control Phase Based on State Delay. In this article, we use the state delay as the index to guide the agent's choice of the A or B phase combination. Assume at the ith time period, the state of the intersection perceived by the agent is $s_i =< s_1, s_2, s_3, s_4 >$, and the state delay $statedelay(s_i)$ of the state s_i is defined as Equation (10):

$$statedelay\,(s_i) = \sum_{j=1}^{4} \left(s_{ij} \sum_{k=1}^{j-1} t_{ik} \right), \tag{10}$$

where s_{ij} is the number of vehicles in the jth phase, and t_{ik} represents the required green timing that ensure the vehicles in the kth phase can completely transit the intersection. The waiting time of vehicles in phase j is the sum of the green time allocated from phase 1 to phase $j-1$ multiplied by the number of vehicles in phase j, and the state delay of state s_i is the sum of the vehicle waiting time in all 4 phases.

We then choose the phase combination A or B for the current state by comparing the state delays for A and B. When the state delay of A is greater than that of B, the total vehicle waiting time in the 4 phases when selecting combination A as the control phase is greater than that of combination B; therefore, phase combination B is selected as the control phase; otherwise, phase combination A is selected. After selecting one of the phase combinations, the original 8-phase control problem for an intersection is transformed into a 4-phase control problem, and the state and action representations transition from 8-tuples to 4-tuples; therefore, the state and action spaces are reduced primarily by selecting the control phase.

3.3.2 An Example of Selecting the Control Phase. Assume that the number of vehicles in the 8 lanes is L = <9, 19, 9, 19, 9, 19, 19, 29>, and each vehicle requires 2 seconds to transit the intersection.

If phase combination A (1,5), (2,6), (3,7), (4,8) is selected, then the number of vehicles for the four phases will be merged as <18 s, 38 s, 28 s, 48 s>, and after discretion, the state vector $SA =< 2, 4, 3, 5 >$. The required green time for the four phases are $< 18s, 38s, 38s, 58s >$, and after discretization $TA =< 2, 4, 4, 6 >$.
$statedelay\_SA$
$= SA[2] \times TA[1] + SA[3] \times (TA[1] + TA[2]) + SA[4] \times (TA[1] + TA[2] + TA[3])$
$= 8 + 18 + 50 = 76$

If phase combination B (1,6), (2,5), (3,8), (4,7) is selected, then the number of vehicles will be merged into <28, 28, 38, 38>, and after discretization, the state vector $SB =< 3, 3, 4, 4 >$. The required green times for the four phases are <38 s, 38 s, 58 s, 38 s>, and after discretization, $TB =< 4, 4, 6, 4 >$.
$statedelay\_SB$
$= SB[2] \times TB[1] + SB[3] \times (TB[1] + TB[2]) + SB[4] \times (TB[1] + TB[2] + TB[3])$
$= 12 + 32 + 56 = 100$

Table 2. Q-TABLE

S \ A	[1,1,1,1]	[1,1,1,2]	···	[6,6,6,5]	[6,6,6,6]
[1,1,1,1]	1.7037	0.3396	···	0.0010	0.0009
[1,1,1,2]	0.4169	1.1503	···	0.0011	0.0009
···	···	···	···	···	···
[5,5,5,4]	0.0390	0.0275	···	0.0044	0.0032
[5,5,5,5]	0.0368	0.0260	···	0.0058	0.0041

Because $statedelay\_SA < statedelay\_SB$, the total delay for selecting phase combination A at the intersection will be lower than that of phase combination B. When the current number of vehicles is L, the agent will choose the A phase combination with low delay as the control phase of the model.

3.3.3 State Space Reduction Based on Real Traffic Flow State. Q-learning is a simple way for agents to learn how to act optimally in traffic signal control domains; it is an unsupervised learning method and is a fundamental reinforcement learning algorithm. It works by establishing a Q-table that contains the Q-values of all state-action combinations and by continuously improving its quality evaluation of a specific action at a specific state. Finally, the Q-learning agent selects the action with the highest Q value in a given state as its action strategy, and the selected strategy guarantees that the agent will accrue the maximum long-term reward.

However, because the training time of the Q-learning algorithm increases exponentially with the number of state-action pairs, simplifying the state space for the Q-learning agent will reduce the training time. The existing research on traffic signal control based on reinforcement learning generally uses all possible discrete state descriptions to define the state space. However, many states may never appear or cannot appear in real traffic flow data, leading to a waste of storage resources.

A Q-table with complete state descriptions is shown in Table 2, where the columns represent the action space, and the lines represent the state space. Each cell in the table represents the Q value for an action under a certain state. To reduce the state space of the Q-learning algorithm, we analyzed the historical traffic state information of an intersection and recorded the occurrences of each state in the state space. The collected results are shown in Figure 3.

As shown in Figure 3, there are a total of 625 possible states. Among these, 402 states never appear in real traffic flow states; therefore, the state space can be further reduced by removing those states that never appear (i.e., states with zero occurrences) from the statistical results. In this way, the state space is reduced to a smaller set of 224 states that can be used to guarantee the signal control effect. However, in practical applications of the Q-learning agent, new states will occur that do not exist in the reduced state space; therefore, we use the nearest neighbor algorithm based on Euclidean distance to map them to the nearest state.

Reducing the state space has three advantages. First, the training time of the algorithm diminishes: Due to the large frequency gaps among real-world traffic flow states, traffic states that never appear in the real dataset require more training time to complete the training process than do the high-frequency states. Second, because the size of the Q-table is closely related to the number of states and actions, it requires fewer storage and computing resources. By reasonably reducing the state space, the storage and computing resources costs of the intersection controller can be effectively reduced while still ensuring the intersection control effect. Third, the algorithm has strong scalability: It reduces the hardware cost for large-scale applications through the state reduction.

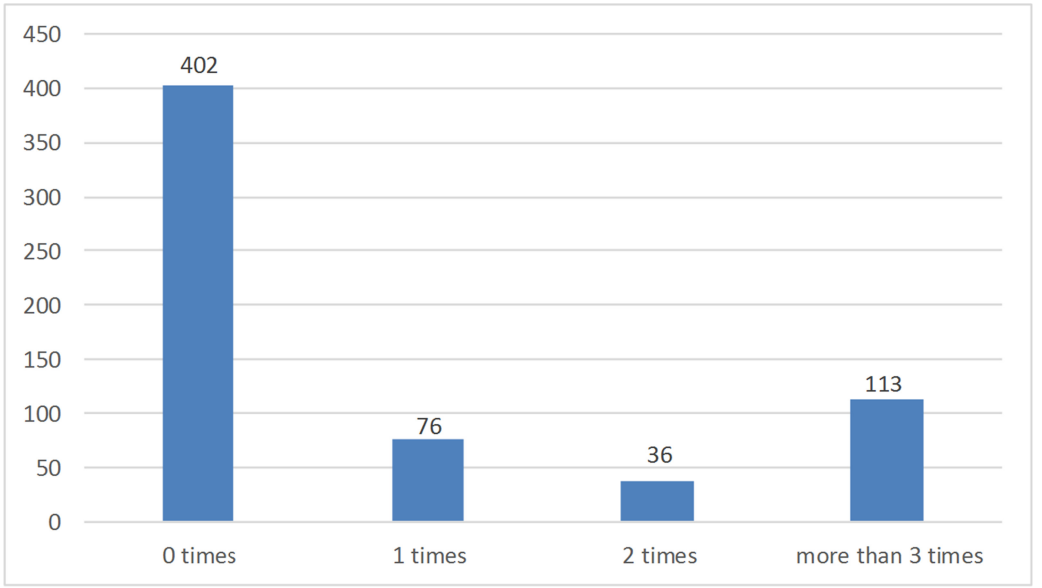

Fig. 3. Statistics of the number of states in state space.

3.4 Q-learning Signal Control Algorithm Based on State Reduction

At each timestep, the Q-learning agent updates the environment by reading the real traffic flow data of the interaction; then, the agent observes the current state and takes action to maximize the reward. Next, to obtain the highest cumulative rewards over time, the agent determines which action (green time for each phase) should be selected based on the Q-table, which contains the Q-values of the state-action pairs. Finally, the environment returns an evaluation r to guide future agent learning, and the Q-value at each timestep is updated according to Equation (11).

$$Q_{i+1}(s_i, a_i) = Q_i(s_i, a_i) + \alpha \left[r_i + \gamma \max_a \mathrm{Q}(s_{i+1}, a) - Q_i(s_i, a_i) \right], \tag{11}$$

where a_i is the action executed under state s_i, after the action a_i, the state is changed to s_{i+1}, and a reward r_i is generated. Here, α is the learning rate, γ is the discounting factor, and $\max_a Q(s_{i+1}, a)$ is the maximum Q value for all possible actions in the state s_{i+1}.

The Q-learning signal control algorithm based on state reduction proposed is shown in Algorithm 1.

4 SIMULATION SETUP

In this section, we describe the simulation experimental setup, which involves data collection and processing, configuring the parameters and methods of the simulation, reducing the state space based on traffic flow data, selecting the performance metrics, and determining the value of β for the reward function.

4.1 Dataset and Preprocessing

First, we collected and processed the traffic flow information at an intersection in Longgang, Shenzhen, from a public dataset. The dataset includes a total of 21 days of data from 1/12 to 2/2. However, due to the large number of missing data points on 1/26 and 1/27, we collated only the other 19 days of data. The collected data include the number of vehicles turning left and going straight in each

Table 3. Data Format

Time \ Lane	1	2	3	4	5	6	7	8
0:00	0	7	3	8	2	10	3	10
0:05	1	4	1	6	0	2	5	2
…	…	…	…	…	…	…	…	…
12:00	16	15	10	27	9	21	24	12
…	…	…	…	…	…	…	…	…
23:55	3	3	2	11	0	3	5	5

Table 4. Dataset Partition Results

Data subset 1	Traffic flow data from 1/12–1/18
Data subset 2	Traffic flow data from 1/19–1/25
Data subset 3	Traffic flow data from 1/12–1/25
Data subset 4	Traffic flow data from 1/28–2/2

ALGORITHM 1: Q-learning Based on State Reduction

Input:
Historical traffic data
Output:
Trained Q-table for simulation control

Selection of control phase based on state delay;
Establish a reinforcement learning model;
Initialize the Q-table, s, α, γ;
Initialize the algorithm variables $step = 0$, $episode = 0$,max_step and $max\_episode$;
repeat
  repeat
    Choose a from s using policy derived from ϵ-greed;
    Take action a, observe r, s'
    Update $Q(s, a)$
    Set $s = s'$
    Set $step = step + 1$
  until $step > max\_step$
  Set $episode = episode + 1$
until $episode > max\_episode$

direction every five minutes for 19 days at an isolated intersection. These collected data were processed into the format shown in Table 3, which reflects the trend of traffic lane flow for the isolated intersection over time and can be used to verify the control scheme performance. For example, at 0:00, the vehicles in the eight lanes are <0, 7, 4, 8, 2, 10, 3, 10>.

Second, commuters, as the main force of vehicle traffic, usually tend to travel regularly in one week. Therefore, we divided the open dataset into data subsets 1, 2, and 3 by week and used the remaining 5 days of data as subset 4. The four data subsets are shown in Table 4.

Third, the control phase was selected for the collected data according to Equation (10) and combined into the number of vehicles in the four-phase lanes. To discretize the number of vehicles, our study divided the vehicles into five categories, as shown in Table 5.

Table 5. State Discrete Results

Class	1	2	3	4	5
Vehicle	[0,9]	[10,19]	[20,29]	[30,39]	40 and above

Table 6. Simulation Parameters

Parameter	Value
Assumed traffic speed	2 s/veh
Initial queue number	[0,0,0,0,0,0,0,0]
α	0.3
γ	0.05
Steps per iteration	2,000
Green light optional time	{10,20,30,40,50,60}

Table 7. Results of State Reduction

State space	Data subset	Number of states that appear	Adopted number of states in reduced state space
Space1	Data subset 1	224	224
Space2	Data subset 2	201	201
Space3	Data subset 3	264	264

4.2 Parameter Setting

We assumed that each vehicle could pass through the intersection in 2 s. The number of vehicle queues was initialized to [0, 0, 0, 0, 0, 0, 0, 0], and these values changed dynamically as vehicles arrived and transited the intersection. The learning rate α, the discount factor γ, and the β reward setting were defined as 0.3, 0.05, and 0.5, respectively. All the parameters involved in the training algorithm are shown in Table 6. The simulations were completed on a computer with a 2.7 GHz Intel Core(TM) i5 processor and MATLAB R2017a software.

Next, we counted the number of states that appeared in data subset 1, data subset 2, and data subset 3 after selecting the control phase. There are 224 states in data subset 1, 201 states in data subset 2, and 264 states in data subset 3. We adopted the states that appeared as the reduced state space. Finally, based on data subsets 1, 2, and 3, we obtained three reduced state spaces: (Space1, Space2, and Space3). The details are shown in Table 7. As an example, when using data subset 1 to reduce the state space, there are only 224 states in Space1.

4.3 Reward Function Objectives and Comparison with Methods

Our experiments are designed to verify the effectiveness of the proposed Q-learning signal control schemes based on state reduction. The experimental part includes the following two main goals:

- Verify the effectiveness of the Q-learning signal control algorithm using the dual objective reward function to control the total waiting time and queue length.
- Verify the rationality and validity of the state space reduction in the Q-learning signal control algorithm based on the state reduction.

It is typically difficult to find a benchmark for traffic signal control problems given that, for commercial reasons, the operational details of most traffic control systems are not easily available [10]. Therefore, we compared Q-learning algorithms based on state reduction (QC) with the timing control method (QA) and the Q-learning control method without state reduction (QB). Next, the three signal control schemes are described in detail below.

Table 8. Fixed Time Control Scheme

Scheme Number	Time	Green time allocated
Plan 1:low peak	23 : 00 ~ 7 : 00	{10s,20s,10s,20s}
Plan 2:flat peak	7 : 00 ~ 8 : 00	{30s,30s,30s,50s}
Plan 3:peak	8 : 00 ~ 13 : 00	{40s,40s,60s,50s}
Plan 2:flat peak	13 : 00 ~ 23 : 00	{30s,30s,30s,50s}

QA is based on fixed time control. In this article, the vehicle transit time in each lane is 2 s/vehicle. First, we divided the intersection traffic flow into three periods: low peak, flat peak, and peak. Then, we set up a fixed time control scheme that causes the intersection to maintain the minimum waiting time on data subset 3. The optimal timing control scheme based on vehicle speed and historical traffic volume data for the isolated intersection is shown in Table 8.

QB is a traditional Q-learning control scheme that uses a double objective reward function but no state reduction. Thus, compared with our proposed method, the QB state space has 625 states that are never reduced.

QC is our traditional Q-learning control scheme using both a double objective reward function and state reduction. To explore the impact of the state space reduction based on the traffic flow data of different periods on the performance of the algorithm, we set up three variations of QC that used different state spaces according to the state space reduction results shown in Table 7 named QC_space1, QC_space2, and QC_space3, respectively. These variants differ only in the size of the state space. The details are as follows:

- QC_space1: There are 224 states in state space reduction based on Data subset 1.
- QC_space2: There are 201 states in state space reduction based on Data subset 2.
- QC_space3: There are 264 states in state space reduction based on Data subset 3.

4.4 Performance Metrics

The performances achieved by most of the extended **reinforcement learning (RL)** approaches were compared with the traditional RL approach, as well as the fixed-time, actuated, and deep RL control systems. The performances of the extended RL models and algorithms were validated from the following aspects: lower average delay, lower waiting time, smaller queue size, higher throughput, lower number of stops per vehicle, and higher average speed. In our study, waiting time and queue size are used to evaluate the performance of each control system. We first defined the concept of the error rate to evaluate the learning rate of the agent using different RL approaches.

- **Waiting time:** Average daily waiting time is the first evaluation index, because it functions as a reward during the training. The less time vehicles spend waiting at the intersection, the better the performance of the scheme is. The waiting time accumulates based on Equation (7).
- **Queue length:** The second evaluation index is queue length. During the peak period, a smaller average queue length indicates better performance. The accumulation of queue length in this article is defined simply as the sum of the discrete vehicles.
- **Error rate:** The third evaluation index is the error rate. Under the same number of iterations, the lower the error rate of the control scheme is, the higher the learning efficiency of the agent using the control scheme is. When the error rate is reduced to 0, the agent has learned the optimal control strategy. The error indicates the control ability of the control algorithm

Table 9. Learning Efficiency with Different β Values

Error \ Times / Beta	2,000	2,500	3,000	3,500	4,000	4,500	5,000
0.0	0.890	0.881	0.885	0.880	0.873	0.873	0.868
0.1	0.109	0.076	0.057	0.041	0.036	0.029	0.022
0.2	0.107	0.064	0.051	0.038	0.031	0.026	0.020
0.3	0.092	0.062	0.044	0.031	0.026	0.023	0.019
0.4	0.119	0.088	0.060	0.051	0.041	0.031	0.024
0.5	0.125	0.094	0.067	0.056	0.044	0.040	0.037
0.6	0.297	0.266	0.230	0.202	0.189	0.179	0.173
0.7	0.689	0.682	0.672	0.658	0.647	0.639	0.624
0.8	0.841	0.836	0.828	0.824	0.822	0.821	0.813
0.9	0.927	0.924	0.920	0.921	0.921	0.921	0.921
1.0	0.996	0.996	0.996	0.996	0.996	0.996	0.996

and is calculated as the ratio of the number of states for which the current control algorithm does not make the optimal decision to the total number of states. We defined the concept of error rate as shown in Equation (12):

$$error = count/length(s), \tag{12}$$

where *count* represents the number of agent-selected actions that are not optimal; $length(s)$ represents the total number of model states.

4.5 Determination of β Value for the Reward Function

To verify the effectiveness of the proposed double goal reward function and study the learning efficiency of β, we recorded the changes in error rate and the training iterations when $\beta \in [0, 1]$ with a step size of 0.1 over repeated runs. The average error rate results are shown in Table 9. We can conclude the following two points:

- When $\beta = 0$, the reward=$M/greencost(s, a)$, and the algorithm focuses only on matching the green light timing with the current state. When there are 5,000 training iterations, the error rate is 0.868. When $\beta = 1$, the reward =$M/delaycost(s, a)$, and the algorithm only on matching the green light timing with the current state. The error rate after 5,000 times of training is 0.996. In this case, the error rate after 5,000 training iterations is 0.996. We can clearly see that the error rate of both extremes is much higher than when $\beta \in [0.1, 0.4]$.
- When $\beta \in [0.1, 0.4]$, the reward =$M/delaycost(s, a) + delaycost(s, a)$, and the algorithm not only focuses on reducing the delay but also on matching the action and state. When the training iterations reach 5,000, the error rate is approximately 0.02, and the learning effect of the algorithm is best when $\beta = 0.3$.

Therefore, we set $\beta = 0.3$ for our reinforcement learning model.

5 EXPERIMENTAL RESULTS AND DISCUSSION

This section discusses the two groups of experiments. All of the comparative experiments were conducted several times to eliminate influences from accidental factors. First, the simulation results were reported for QA, QB, and the three QC variants with different state spaces. Second, we analyzed the waiting time and queue length performances to evaluate the validity of our proposed

Table 10. Daily Waiting Time

Scheme	Data subset 1	Data subset 2	Data subset 4	Average
QA	44,315.00	30,815.00	23,834.80	33,951.79
QB	8,920.6	8,296.30	6,648.20	8,092.60
QC_space1	8,941.22	8,422.44	6,737.44	8,170.15
QC_space2	9,257.28	8,320.88	6,905.76	8,293.47
QC_space3	8,938.76	8,382.48	6,752.36	8,158.45

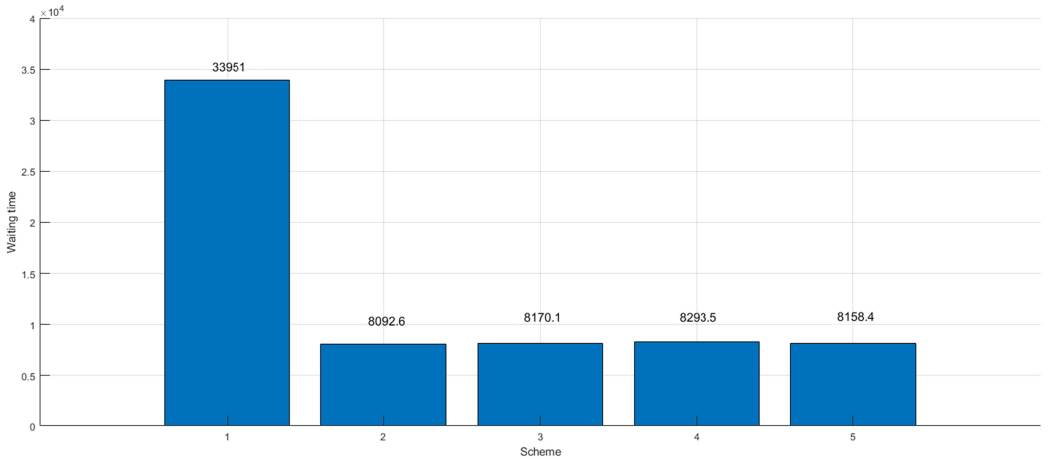

Fig. 4. Waiting time of three control scheme.

Q-learning model. Third, we analyze the relationship between the number of iterations and the error rate to explain the efficiency of state reduction in training time.

5.1 Performance Comparison of Control Schemes on Dataset

In this group of experiments, data subset 3 is a combination of data subsets 1 and 2. Therefore, we compared the control effect of QA, QB and the three QC variants with different state spaces on data subsets 1, 2, and 4. For each experiment, we obtained the average daily waiting time and average daily queue on the given dataset. Finally, we use the average value of several experiments to demonstrate the effectiveness of our proposed method.

5.1.1 Comparison of Waiting Time. The daily waiting time of the three control schemes on the four data subsets is shown in Table 10, and the average daily waiting time is shown in Figure 4. As Figure 4 shows, the average daily vehicle waiting time using the QA scheme is 33,951.79, which is the worst control effect. The next worst result was that of the QC_space2 scheme, in which there are 201 states in the state space, and the average waiting time is 8,293.47. Next was the QC_space1 scheme, in which there are 224 states in the state space and the average daily waiting time is 8,170.15. The QC_space3, in which there are 264 states in the state space, achieves an average daily waiting time of 8,158.45. The best control effect was obtained by the QB method using our reward function without state reduction, which achieved an average daily waiting time of 8,092.60. We can conclude the following:

- The average waiting times of QB and of QC with different state spaces are very similar, and all of them have average waiting times that are less than 30% of the waiting time of the

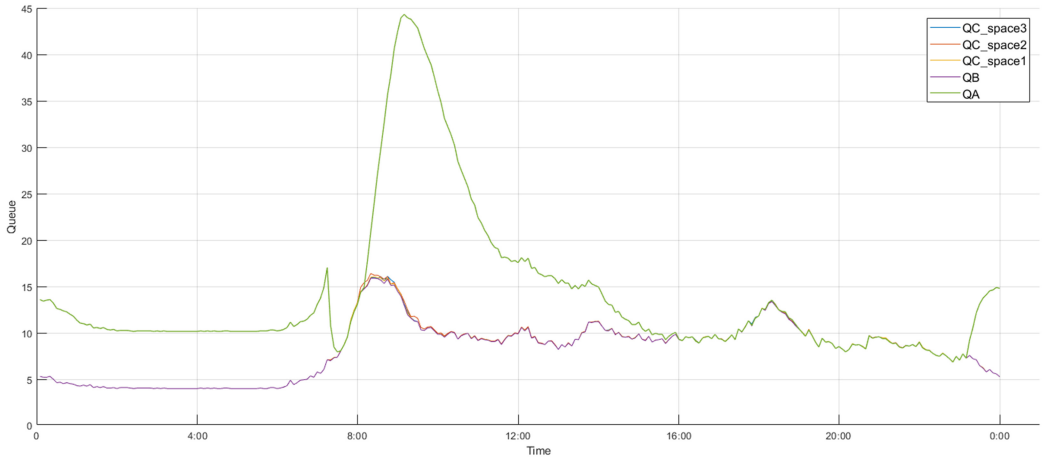

Fig. 5. Average queue length of three control scheme.

QA scheme. This result shows that signal control schemes based on Q-Learning are better than the optimal timing control scheme for controlling vehicle waiting time. The reason is that the Q-learning algorithm induces the agent to change the green light timing plan as the traffic flow changes, while the timing control cannot make a fast response to a change in traffic flow; therefore, its average vehicle waiting time is extremely high.

- The average waiting times of QC_space1, QC_space2, and QC_space3 are higher than the average waiting time of the QB scheme. This shows that state reduction reduces the effect of intersection signal control. The reason for this poor effect is that a small part of the traffic flow state is not reflected in the reduced state space during the actual control of the traffic flow. As a result, the green light timing selected based on state proximity does not match the current traffic flow state, which causes the control effect of the three QC variants to be slightly worse than that of the QB model. However, we can see that the average waiting times are not much different among the three QC variants, which shows that state space reduction based on one week of traffic flow data is similar to the state space reduction based on two weeks of traffic flow data, and all the variants achieve good traffic control. Thus, the rationality of state space reduction is proven.

5.1.2 Comparison of Queue Length. Similar to the average waiting time, the average daily queue length of the QA, QB, and three methods of QC with different state spaces on four data subsets are shown in Figure 5.

Figure 5 shows that the average queue length fluctuates dramatically under the QA scheme and that it increases rapidly during the peak period, with the maximum queue length reaching approximately 45. In contrast, the queue lengths of the QB method and the three QC variants remain fairly stable during the flat-peak and low-peak periods, and the queue lengths of these schemes are all below 20. We can reach the following two conclusions from Figure 5:

- A signal control scheme based on Q-learning is better able coordinate the traffic capacity at intersections and avoid congestion. According to our model, an intersection is controlled by the control phase, which consists of four subphases, and the maximum green time that can be selected for each phase is 6 (the green time before discretization is 60 s). Therefore, the maximum number of vehicles that can transit the intersection during a control cycle is 24. However, the maximum queue length of the QA control scheme during peak hours is 45,

which is far beyond the capacity of 24; thus, traffic congestion occurs. In contrast, the other methods do not exceed the traffic capacity of the intersection even during the peak period of traffic flow, and the queue lengths of QB, QC_space1, QC_space2, and QC_space3 are quite stable and remain below 24.

- The average queue lengths of QC_space1, QC_space2, and QC_space3 variants are similar to the average queue of the QB scheme, which shows that control effect of the QC schemes based on state reduction are highly similar to that of the traditional QB scheme. Therefore, the state space reduction does not result in a significant reduction of the signal control effect.

From the waiting time and queue comparison experimental results, it can be seen that the Q-learning algorithm with state reduction does significantly reduce the signal control effect—its control effect is only slightly lower than that of the traditional Q-learning algorithm that uses the full-state description. In addition, both of model types function much better than does the optimized timing control method.

5.2 Error Rate Comparison of Control Schemes on Dataset

In this group of experiments, we compared the learning speed of QB and the three QC variants on the same data subset. Before showing the experimental results, we first perform a simple analysis of the complexity of the above algorithms and theoretically explain the feasibility of our proposed method. Then, we introduce experimental results of the error rate comparison to prove the superiority of our method. The reasons for these results are analyzed in detail.

5.2.1 Theoretical Analysis of Complexity. A complexity analysis of the reinforcement learning algorithm applied to intersection signal control generally considers three aspects: computational complexity, message complexity, and storage complexity. For an isolated intersection signal control, we only need to analyze and compare the computational and storage complexities.

Computational complexity is the maximum number of times a Q-learning algorithm must execute to update the Q-values for all agent actions. For step-by-step computational complexity, this is considered in a specific state. As an example, in the traditional Q-learning algorithm, an agent updates its Q-value when it receives a delayed reward. Therefore, the step-by-step state computational complexity is O(|A|), because there are |A| actions for each state. The overall step-by-step complexity is calculated as O(|S||A|), since there are |S| states.

Storage complexity reflects the amount of memory required to store the Q-values. As an example, in the traditional Q-learning algorithm, the agents store a Q-value for each state-action pair,; thus, the step-by-step storage complexity has a value of 1, and the complexity of the agent is calculated as |S||A|, because an agent maintains |S||A| state-action pairs in a Q-table.

If we divide the traffic flow data of 8 lanes into 5 categories, then the green time allocated for the 8 phases is simply divided into 10 s, 20 s, 30 s, 40 s, 50 s, 60 s six categories. Then, the size of the state space is $|S| = 5^8$ and the action space is $|A| = 6^8$. After selecting the control phase, we can combine the traffic flow data of 8 lanes into the traffic flow data of 4 phase lanes and divide the traffic flow data of the 4 phase lanes into [0,10], [10,20], [20,30], [30,40], [40,+ ∞)—five categories according to the standard shown in Table 5. At the same time, the green timing of the 8 phases is transformed into 4 control phases. The state space and action space of the model are thus reduced to $|S| = 5^4$ and $|A| = 6^4$. Then, based on the historical traffic flow data collected in the dataset, those states that do not appear are removed from the initially reduced state space, and the remaining states are adopted as the reduced state space proposed in this article.

Suppose that we reduce the state space based on data subset 1, leaving states in the reduced state space. The changes in the computational and storage complexities of Q-learning are shown in Table 11. Obviously, selecting the control phase and the state space reduction based on the

Table 11. Change in Complexity

Operation	Size of Q-table	Complexity
Discretization of traffic flow data	$5^8 \times 6^8$	$\|S\| = 5^8$ and $\|A\| = 6^8$
Selection of control phase	$5^4 \times 6^4$	$\|S\| = 5^4$ and $\|A\| = 6^4$
State space reduced again	224×6^4	$\|S\| = 224$ and $\|A\| = 6^4$

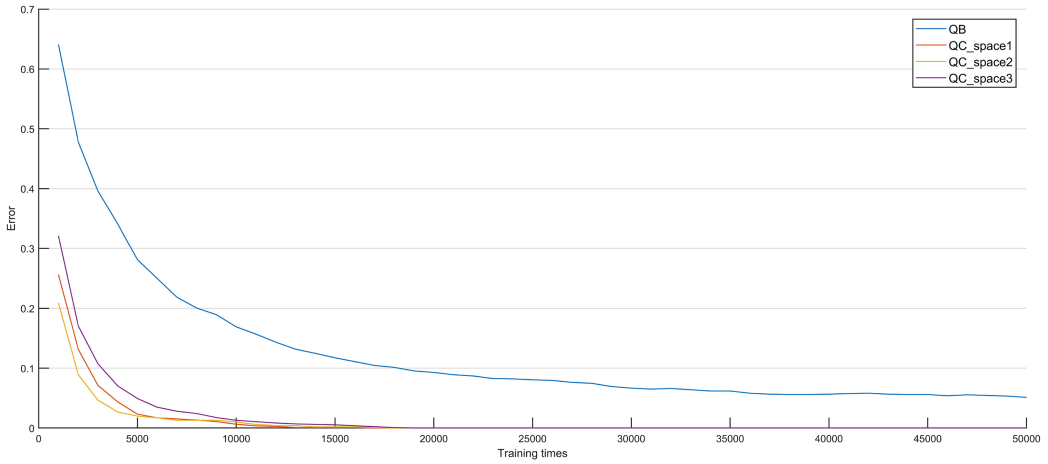

Fig. 6. Tendency of error rates change.

real traffic flow, the computational complexity, and storage complexity of Q-learning are greatly reduced. Therefore, it is theoretically feasible to optimize the complexity of the Q-learning algorithm by reducing the state space.

5.2.2 Comparison of Error Rate. Each phase contains two lanes, assuming that the traffic efficiency has remained unchanged. Therefore, the green time of a phase of 10 s is ideal to pass 10 vehicles. Under such circumstances, the optimal strategy that the agent can learn is determined. As an example, when the current state known to the agent is [2, 3, 4, 5], the optimal green light times that the agent should adopt are also [2, 3, 4, 5]. We can use the error rate to evaluate the training efficiency of five control schemes. As the iterations increase, the timing strategy of the agent will not change, and the error rate of QA is maintained at 1. Therefore, we only need to compare the changes in the error rate of the three QC variants and the QB scheme when comparing the learning efficiency. Under the same training time, the lower the error rate is, the higher the learning efficiency is.

Because of randomness in the training process of the Q-learning algorithm, this group of experiments was conducted several times. In each experiment, QB and three QC variants were trained on data subset 1, and the error rates of the schemes were recorded. Finally, the average value is taken as the result of the comparison. The trends of the average error rates are shown in Figure 6.

When the error rate of the agent is reduced to 0, it means that the agent using QB or QC has learned the optimal timing strategy. In other words, when the error rate is reduced to 0, the training can be considered complete. As Figure 6 shows, as the number of iterations increases, the error rates of both the QB scheme and three methods of QC are gradually reduced, which shows that the agents are learning the optimal strategy of green light timing from experience.

The error rate of the QB scheme declines the slowest, and its error rate remains at approximately 0.5% even after 50,000 iterations, which shows that many states without an optimal timing strategy

still exist. Thus, the learning efficiency of the QB scheme is the worst. The declining trends of the three QC methods for different state spaces are similar, and the error rates are reduced to 0 at approximately 18,000 iterations. However, at the beginning of training, the magnitudes of the error rate declines have the following sequence: QC_space2 > QC_space1 > QC_space3. Finally, we extract and analyze the following two conclusions from Figure 6:

- When the training iterations reach approximately 16,000, the average error rates of the three QC schemes with state reduction are reduced to approximately 0. When the number of iterations reaches 18,000, the error rate becomes 0. However, the average error rate of the QB scheme without state reduction is still approximately 0.1 even after training 18,000 iterations. Even though its average error rate after 50,000 training iterations is still approximately 0.05, from the declining trend of the QB scheme error rate, it would require many additional training iterations to reduce the error rate to 0. In contrast to the declining trend of the average error rate of the QB scheme, the relationship among the error rates of the three QC methods remains highly stable as QC_space2 < QC_space1 < QC_space3 < QB, while the relationship among the size of the state spaces is QC_space2 < QC_space1 < QC_space3 < QB. The above phenomena show that the smaller the state space is, the fewer training iterations the agent needs to learn the optimal strategy. Therefore, the state space reduction stably and effectively shortens the training time of the Q-learning algorithm.
- From the beginning of training to the point at which the average error rate of the three QC schemes is reduced to 0, the error rates of both QB and QC fall increasingly slowly with the number of iterations. The state reduction proposed in this article is based on real traffic data; therefore, the states in the reduced state space appear more frequently in real traffic data. To reduce the error rate to 0, the agent using QC requires fewer training times. The full state space includes many states that rarely or never appear in the real traffic flow, which causes the agent using QB to require many additional iterations to reduce its error rate. When the error rate is low, there some states that rarely or never appear in the real traffic flow that cannot learn the optimal strategy, which is one of the reasons why the training time of the QB scheme is long compared with those of the three QC schemes.

The results of this group of experiments show that there are two main reasons why the training time of the control algorithm can be reduced by reducing the state space. One reason is that the state space reduction optimizes the size of the state space of the algorithm, which effectively reduces the required training time. The other reason is that it eliminates states that rarely or never appear in the real traffic flow from the state space, also substantially reducing the training time of the control algorithm. Considering the performances of the QB scheme and those of the three QC variants regarding both waiting time and queue length control, the three QC schemes use a simplified Q-table and achieve lower training times while still ensuring a similar control effect as that of the QB scheme.

6 CONCLUSION

In this article, we propose a Q-learning signal control algorithm based on state reduction to improve the performance of the existing signal control algorithm based on reinforcement learning, which has the shortcomings of mismatches between the signal timing and traffic demand and high complexity. First, to match the signal timing with the traffic demand, we built a reinforcement learning model based on actual traffic flow data and the control rules and constructed a double objective reward function that not only reduces vehicle delays but also improves the matching degree between the signal timing and the traffic demand. Second, to reduce the complexity of the signal control algorithm, we first reduced the state and action spaces of the model by selecting a

control phase combination. Then, the state space of the model was further reduced based by eliminating states based on historical traffic flow information. Finally, the complexity of the control algorithm was optimized by using the simplified Q-table.

In the future, we plan to investigate a signal-coordination control mechanism for multiple intersections based on the signal control for isolated intersections developed in this study—especially the construction of a reward function for the coordinated control of multiple intersections. Eventually, we hope to propose a fast and effective signal-coordination control scheme for multiple intersections.

REFERENCES

[1] Monireh Abdoos, Nasser Mozayani, and Ana L. C. Bazzan. 2011. Traffic light control in non-stationary environments based on multi agent Q-learning. In *14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1580–1585.

[2] Rahib H. Abiyev, Mohammad Ma'aitah, and Bengi Sonyel. 2017. Fuzzy logic traffic lights control (FLTLC). In *9th International Conference on Education Technology and Computers*. 233–238.

[3] Sahar Araghi, Abbas Khosravi, Michael Johnstone, and Doug Creighton. 2013. Q-learning method for controlling traffic signal phase time in a single intersection. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1261–1265.

[4] Bhushan S. Atote, Mangesh Bedekar, and Suja S. Panicker. 2016. Centralized approach towards intelligent traffic signal control. In *2nd International Conference on Information and Communication Technology for Competitive Strategies (ICTCS'16)*. Association for Computing Machinery, New York, NY. DOI: https://doi.org/10.1145/2905055.2905121

[5] Saif Islam Bouderba and Najem Moussa. 2019. Reinforcement learning (Q-LEARNING) traffic light controller within intersection traffic system. In *4th International Conference on Big Data and Internet of Things*. 1–6.

[6] D. S. Broomhead and D. Lowe. 1988. Multivariable functional interpolation and adaptive networks, complex systems. *Complex Systems* 2 (1998), 321–355. https://www.bibsonomy.org/bibtex/24ef3a0adaabe7e13dcdeee339068f840/mcdiaz.

[7] Wentao Chen, Tehuan Chen, and Guang Lin. 2019. Reinforcement learning for traffic control with adaptive horizon. *arXiv preprint arXiv:1903.12348* (2019).

[8] Lucas Barcelos De Oliveira and Eduardo Camponogara. 2010. Multi-agent model predictive control of signaling split in urban traffic networks. *Transport. Res. Part C: Emerg. Technol.* 18, 1 (2010), 120–139.

[9] Kai Dou, Bin Guo, and Li Kuang. 2019. A privacy-preserving multimedia recommendation in the context of social network based on weighted noise injection. *Multimedia Tools Applic.* 78, 19 (2019), 26907–26926.

[10] Samah El-Tantawy, Baher Abdulhai, and Hossam Abdelgawad. 2013. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto. *IEEE Trans. Intell. Transport. Syst.* 14, 3 (2013), 1140–1150.

[11] Sébastien Faye, Claude Chaudet, and Isabelle Demeure. 2012. A distributed algorithm for multiple intersections adaptive traffic lights control using a wireless sensor networks. In *1st Workshop on Urban Networking*. 13–18.

[12] Honghao Gao, Yucong Duan, Lixu Shao, and Xiaobing Sun. 2019. Transformation-based processing of typed resources for multimedia sources in the IoT environment. *Wireless Networks* (2019), 1–17.

[13] Honghao Gao, Wanqiu Huang, and Xiaoxian Yang. 2019. Applying probabilistic model checking to path planning in an intelligent transportation system using mobility trajectories and their statistical data. *Intell. Automat. Soft Comput.* 25, 3 (2019), 547–559.

[14] H. Gao, C. Liu, Y. Li, and X. Yang. 2021. V2VR: Reliable Hybrid-Network-Oriented V2V Data Transmission and Routing Considering RSUs and Connectivity Probability, *IEEE Trans. Intell. Transport. Syst*. 22, 6 (2021), 3533–3546. DOI: 10.1109/TITS.2020.2983835

[15] H. Gao, Y. Xu, Y. Yin, W. Zhang, R. Li, and X. Wang. 2020. Context-aware QoS prediction with neural collaborative filtering for Internet-of-Things services. *IEEE Internet Things J.* 7, 5 (2020), 4532–4542.

[16] Wade Genders and Saiedeh Razavi. 2016. Using a deep reinforcement learning agent for traffic signal control. *arXiv preprint arXiv:1611.01142* (2016).

[17] P. B. Hunt, D. I. Robertson, R. D. Bretherton, and M. Royle. 1982. The SCOOT on-line traffic signal optimisation technique. *Traff. Eng. Contr.* 23, 4 (1982).

[18] Celine Jacob and Baher Abdulhai. 2006. Automated adaptive traffic corridor control using reinforcement learning: Approach and case studies. *Transport. Res. Rec.* 1959, 1 (2006), 1–8.

[19] Li Kuang, Chunbo Hua, Jiagui Wu, Yuyu Yin, and Honghao Gao. 2020. Traffic volume prediction based on multi-sources GPS trajectory data by temporal convolutional network. *Mob. Netw. Applic.* 25, 4 (2020), 1405–1417.

[20] Li Kuang, Xuejin Yan, Xianhan Tan, Shuqi Li, and Xiaoxian Yang. 2019. Predicting taxi demand based on 3D convolutional neural network and multi-task learning. *Rem. Sens.* 11, 11 (2019), 1265.

[21] Daniyar Kurmankhojayev, Gulnur Tolebi, and Nurlan S. Dairbekov. 2019. Road traffic demand estimation and traffic signal control. In *5th International Conference on Engineering and MIS (ICEMIS'19)*. Association for Computing Machinery, New York, NY. DPO: https://doi.org/10.1145/3330431.3330433

[22] Zhifang Liao, Dayu He, Zhijie Chen, Xiaoping Fan, Yan Zhang, and Shengzong Liu. 2018. Exploring the characteristics of issue-related behaviors in github using visualization techniques. *IEEE Access* 6 (2018), 24003–24015.

[23] P. R. Lowrie. 1990. *Scats, Sydney Co-ordinated Adaptive Traffic System: A Traffic Responsive Method Of Controlling Urban Traffic*. Technical report.

[24] Pei Luo, Qian Ma, and Hui-xian Huang. 2009. Urban trunk road traffic signal coordinated control based on multi-objective immune algorithm. In *International Asia Conference on Informatics in Control, Automation and Robotics*. IEEE, 72–76.

[25] Jinming Ma and Feng Wu. 2020. Feudal multi-agent deep reinforcement learning for traffic signal control. In *19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS'20)*. International Foundation for Autonomous Agents and Multiagent Systems, 816–824.

[26] Sandeep Mehan and Vandana Sharma. 2011. Development of traffic light control system based on fuzzy logic. In *International Conference on Advances in Computing and Artificial Intelligence (ACAI'11)*. Association for Computing Machinery, New York, NY, 162–165. DOI: https://doi.org/10.1145/2007052.2007085

[27] Alok Patel, Jayendran Venkateswaran, and Tom V. Mathew. 2015. Optimal signal control for pre-timed signalized junctions with uncertain traffic: Simulation based optimization approach. In *Winter Simulation Conference (WSC'15)*. IEEE Press, 3168–3169.

[28] K. J. Prabuchandran, Hemanth Kumar A. N., and Shalabh Bhatnagar. 2014. Multi-agent reinforcement learning for traffic signal control. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC'14)*. IEEE, 2529–2534.

[29] Luo Qin, Hou Yufei, and Wang Zhuoqun. 2018. Signal timing simulation of single intersection based on fuzzy-genetic algorithm. In *10th International Conference on Computer Modeling and Simulation (ICCMS'18)*. Association for Computing Machinery, New York, NY, 28–32. DOI: https://doi.org/10.1145/3177457.3177496

[30] Nouha Rida and Aberrahim Hasbi. 2018. Traffic lights control using wireless sensors networks. In *3rd International Conference on Smart City Applications (SCA'18)*. Association for Computing Machinery, New York, NY. DOI: https://doi.org/10.1145/3286606.3286791

[31] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. The MIT Press.

[32] Thomas L. Thorpe and Charles W. Anderson. 1996. *Traffic Light Control Using Sarsa with Three State Representations*. Technical Report. IBM corporation.

[33] Kazuhiro Tobita and Takashi Nagatani. 2013. Green-wave control of an unbalanced two-route traffic system with signals. *Phys. A: Statist. Mech. Applic.* 392, 21 (2013), 5422–5430.

[34] Elise Van der Pol and Frans A. Oliehoek. 2016. Coordinated deep reinforcement learners for traffic light control. In *Workshop on Learning, Inference and Control of Multi-agent Systems (at NIPS 2016)*.

[35] Hong Wang, Chieh Ross Wang, Meixin Zhu, and Wanshi Hong. 2019. Globalized modeling and signal timing control for large-scale networked intersections. In *2nd ACM/EIGSCC Symposium on Smart Cities and Communities (SCC'19)*. Association for Computing Machinery, New York, NY. DOI: https://doi.org/10.1145/3357492.3358635

[36] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. *Mach. Learn.* 8, 3-4 (1992), 279–292.

[37] Christopher John Cornish Hellaby Watkins. 1989. *Learning from Delayed Rewards*. PhD Thesis, University of Cambridge, England.

[38] Brian Wolshon and William C. Taylor. 1999. Analysis of intersection delay under real-time adaptive signal control. *Transport. Res. Part C: Emerg. Technol.* 7, 1 (1999), 53–72.

[39] Wang Yaping and Zhang Zheng. 2011. A method of reinforcement learning based automatic traffic signal control. In *3rd International Conference on Measuring Technology and Mechatronics Automation*, Vol. 1. IEEE, 119–122.

[40] Yuyu Yin, Lu Chen, Yueshen Xu, Jian Wan, He Zhang, and Zhida Mai. 2019. QoS prediction for service recommendation with deep feature learning in edge computing environment. *Mob. Netw. Applic.* 25 (2019), 391–401. https://doi.org/10.1007/s11036-019-01241-7

[41] Yuyu Yin, Jing Xia, Yu Li, Wenjian Xu, Lifeng Yu, et al. 2019. Group-wise itinerary planning in temporary mobile social network. *IEEE Access* 7 (2019), 83682–83693.

[42] Jun Yu, Jing Li, Zhou Yu, and Qingming Huang. 2019. Multimodal transformer with multi-view visual representation for image captioning. *IEEE Trans. Circ. Syst. Vid. Technol.* 30, 12 (2019), 4467–4480.

[43] Jun Yu, Min Tan, Hongyuan Zhang, Dacheng Tao, and Yong Rui. 2019. Hierarchical deep click feature prediction for fine-grained image recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* (2019).

[44] Jun Yu, Chaoyang Zhu, Jian Zhang, Qingming Huang, and Dacheng Tao. 2019. Spatial pyramid-enhanced NetVLAD with weighted triplet loss for place recognition. *IEEE Transactions on Neural Networks and Learning Systems* (2019).

[45] Bowen Zheng, Chung-Wei Lin, Shinichi Shiraishi, and Qi Zhu. 2019. Design and analysis of delay-tolerant intelligent intersection management. *ACM Transactions on Cyber-Physical Systems* 4, 1 (2019), 1–27.

[46] Guanjie Zheng, Yuanhao Xiong, Xinshi Zang, Jie Feng, Hua Wei, Huichu Zhang, Yong Li, Kai Xu, and Zhenhui Li. 2019. Learning phase competition for traffic signal control. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM'19)*. Association for Computing Machinery, New York, NY, USA, 1963–1972. https://doi.org/10.1145/3357384.3357900

[47] Nantao Zheng, Kairou Wang, Weihua Zhan, and Lei Deng. 2019. Targeting virus-host protein interactions: Feature extraction and machine learning approaches. *Current Drug Metabolism* 20, 3 (2019), 177–184.

[48] Dunhao Zhong and Azzedine Boukerche. 2019. Traffic signal control using deep reinforcement learning with multiple resources of rewards. In *Proceedings of the 16th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, & Ubiquitous Networks*. 23–28.

[49] Hongge Zhu. 2019. A platoon-based rolling optimization algorithm at isolated intersections in a connected and autonomous vehicle environment. In *Proceedings of the 2019 5th International Conference on Computing and Data Engineering (ICCDE'19)*. Association for Computing Machinery, New York, NY, USA, 41–46. https://doi.org/10.1145/3330530.3330545

[50] Yonghua Zhu, Weilin Zhang, Yihai Chen, and Honghao Gao. 2019. A novel approach to workload prediction using attention-based LSTM encoder-decoder network in cloud environment. *EURASIP Journal on Wireless Communications and Networking* 2019, 1 (2019), 274.

Received March 2020; revised July 2020; accepted August 2020