A Survey on Reinforcement Learning Models and Algorithms for Traffic Signal Control

KOK-LIM ALVIN YAU, Sunway University
JUNAID QADIR, Information Technology University, Punjab, Pakistan
HOOI LING KHOO, Universiti Tunku Abdul Rahman
MEE HONG LING, Sunway University
PETER KOMISARCZUK, Royal Holloway University of London

Traffic congestion has become a vexing and complex issue in many urban areas. Of particular interest are the intersections where traffic bottlenecks are known to occur despite being traditionally signalized. Reinforcement learning (RL), which is an artificial intelligence approach, has been adopted in traffic signal control for monitoring and ameliorating traffic congestion. RL enables autonomous decision makers (e.g., traffic signal controllers) to observe, learn, and select the optimal action (e.g., determining the appropriate traffic phase and its timing) to manage traffic such that system performance is improved. This article reviews various RL models and algorithms applied to traffic signal control in the aspects of the representations of the RL model (i.e., state, action, and reward), performance measures, and complexity to establish a foundation for further investigation in this research field. Open issues are presented toward the end of this article to discover new research areas with the objective to spark new interest in this research field.

CCS Concepts: • **General and reference** → **Surveys and overviews**; • **Computing methodologies** → **Reinforcement learning;**

Additional Key Words and Phrases: Applied artificial intelligence, multiagent system, traffic signal control, intelligent transportation system

ACM Reference Format:
Kok-Lim Alvin Yau, Junaid Qadir, Hooi Ling Khoo, Mee Hong Ling, and Peter Komisarczuk. 2017. A survey on reinforcement learning models and algorithms for traffic signal control. ACM Comput. Surv. 50, 3, Article 34 (June 2017), 38 pages.
DOI: http://dx.doi.org/10.1145/3068287

1. INTRODUCTION

An efficient traffic network can provide a wide range of advantages, including smoother traffic flow and a greener environment—factors that can in turn improve economic competitiveness. As traffic demand increases, it is vital to better manage intersections

This work was supported by the Malaysian Ministry of Education under Fundamental Research Grant Scheme FRGS/1/2014/ICT03/SYUC/02/2 and Sunway-Lancaster Grant Scheme SGSSL-FST-CSNS-0114-05 and PVM1024.
Authors' addresses: K.-L. A. Yau (Corresponding Author) and M. H. Ling, Department of Computing and Information Systems, Faculty of Science and Technology, Sunway University, No. 5 Jalan Universiti, Bandar Sunway, 47500 Selangor, Malaysia; emails: koklimy@sunway.edu.my, mhling@sunway.edu.my; J. Qadir, Electrical Engineering Department, Information Technology University, Arfa Software Technology Park, Ferozepur Road, Lahore 54000, Pakistan; email: junaid.qadir@itu.edu.pk; H. L. Khoo, Department of Civil Engineering, Universiti Tunku Abdul Rahman, Jalan Sungai Long, Bandar Sungai Long, 43000 Selangor, Malaysia; email: khoohl@utar.edu.my; P. Komisarczuk, Information Security Group, Royal Holloway University of London, Engham Hill, Engham TW20 0EX, United Kingdom; email: peter.komisarczuk@rhul.ac.uk.

DOI: http://dx.doi.org/10.1145/3068287

where traffic bottlenecks can cause congestion despite being monitored using a traffic signal controller that has three traditional signal colors: red indicating the "stop" sign, amber indicating the "slow down" sign, and green indicating the "go" sign. A complete *cycle* consists of a predetermined sequence of traffic phases, and the *cycle length* is the time interval of a complete cycle. Each *traffic phase* consists of a combination of green signals allocated to a set of lanes simultaneously for nonconflicting movements at an intersection. A short moment of all red signals is included in between the traffic phases to provide safe transition, causing some time loss. A *traffic phase split* is the time interval, which is part of a cycle length, allocated for a traffic phase.

Traditionally, the traffic signal controller is deterministic in nature, and there are two main types, namely pretimed and actuated control systems. A pretimed control system presets the timing of the green signals (or the green time) using the Webster formula based on historical traffic data collected from the traffic network at different times [Yin et al. 2016]; however, this offline approach is not responsive to the dynamicity of the traffic conditions. An actuated control system is responsive to the dynamicity of the traffic being served; however, it takes account of the current (or instantaneous) traffic conditions only, rather than longer-term traffic conditions. As an example, it activates green signals at lanes with vehicles. Hence, the pretimed and actuated control systems do not take into account both current and longer-term traffic conditions, such as queue size, at the same time. Note that the queue size can be measured [El-Tantawy et al. 2013; Li et al. 2009; Medina and Benekohal 2012; Prabuchandran et al. 2015; Prashanth and Bhatnagar 2012] using two inductive loop detectors (see Section 2.2.1), although only a single detector is normally installed at a lane to detect the presence of vehicles in practice. During congestion, traffic flowing into an intersection is higher than the traffic flowing out of it, so the queue size of the vehicles may not be reduced despite a green signal being activated, causing vehicles to move slowly or stop. A more substantial consequence is *cross-blocking,* when no vehicle can cross an intersection despite a green signal being activated as the respective lane of the downstream intersection has become fully occupied. At the other extreme is *green idling,* when no vehicle is at an intersection when a green signal is activated. Congestion that occurs in a single lane has a domino and single-point-of-failure effect as it can affect the traffic conditions of the other lanes at the same and neighboring intersections.

Research has been undertaken to investigate traffic signal control that can optimize traffic signal scheduling and timing, such as extending the green time at the right time, in order to ameliorate traffic bottlenecks at a single or multiple intersections with medium- or heavy-load traffics. Typically, in a traditional pretimed control system, each of the available traffic phases must appear in a cycle for a certain traffic phase split. However, using reinforcement learning (RL), which is an unsupervised artificial intelligence technique, the traffic phase split can be shortened or lengthened, and some of the traffic phases can even be skipped according to the unpredictable and time-varying dynamic traffic conditions. The RL approach is an online adaptive approach responsive to the dynamicity of both current and longer-term traffics.

While general reviews of designing the traffic signal control systems using multiagent systems [Bazzan 2009] and various artificial intelligence approaches [Zhao et al. 2012] have been presented, this article complements their works by focusing on the RL approach, particularly on how RL models and algorithms can be applied to formulate the traffic signal control problem, and how the strengths of various RL approaches can provide added advantages in addressing the challenges brought about by traffic signal control. This article is timely due to the need to apply, as well as to further enhance, this approach to ease congestion at intersections as traffic volume continues to increase despite congestion being an ongoing serious issue in most urban areas. To the best of our knowledge, this is the first comprehensive article on RL applied to traffic signal control.

Table I. Generic Notations Used in This Article

Category	Notation	Description
General	T	A single episode. Each episode consists of a sequence of time instants $t_T = 1, 2, \ldots, \lvert T \rvert$
	t	A single time instant $t = 1, 2, \ldots$
	t_w	A time window
	t_{slot}	A time slot
	n_{slot}	Number of time slots
	η_i	Weight factor, where $\sum_i \eta_i = 1$
	$N_t(\bullet)$	Number of times component(s) $\bullet$ has/have been visited up to time t since the beginning $0, 1, 2, \ldots$
Traffic	M	Set of traffic signal controllers $M = [1, 2, \ldots, \lvert M-1 \rvert, \lvert M \rvert]$
	$d^i \in D^i$	Leg, or lane, of intersection i with $D^i = [1, 2, \ldots, d^i, \ldots, \lvert D^i \rvert]$
	H^i	Traffic phase of traffic signal controller i with $H^i = [1, 2, \ldots, h^i, \ldots, \lvert H^i \rvert]$
	$n_{q,t}^{i,d^i}$	Queue size of a lane d^i at intersection i at time t
	$n_{c,t}^{i}$	Number of vehicles crossing an intersection i at time t
	t_p^i	Current traffic phase split of intersection i, where the minimum traffic phase split is $t_{p,\min}^i$
	$t_{r,t}^{i,d^i}$	Red time of lane d^i at intersection i at time t
	$t_{v,t}^{i}$	Vehicular delay of vehicles at intersection i at time t
	$t_{w,t}^{i}$	Waiting time of vehicles at intersection i at time t
	β^i	Traffic arrival rate at intersection i
	$P_{t_w}^{i,n}$	Probability of n vehicles arriving at an intersection i within a time window t_w
	Ψ_q^i	Threshold i for the queue size of a lane with $\Psi_q^1 < \Psi_q^2$
	Ψ_r^i	Threshold i for the red timing of a lane with $\Psi_r^1 < \Psi_r^2$
RL	I	Set of agents $I = [1, 2, \ldots, \lvert I-1 \rvert, \lvert I \rvert]$
	$\mathcal{J}^i$	Set of agent i's neighboring agents $\mathcal{J}^i = [1, 2, \ldots, \lvert \mathcal{J}^i - 1 \rvert, \lvert \mathcal{J}^i \rvert]$
	$s_t^i \in S^i$	State of agent i at time t. $S^i = \{s_1^i, s_2^i, \ldots, s_{n_s}^i, \ldots, s_{\lvert S^i \rvert}^i\}$ is a set of candidate states at agent i
	$\boldsymbol{s_t^i} \in S^i$	State of agent i consists of a set of substates $\boldsymbol{s_t^i} = [s_t^{i,1}, s_t^{i,2}, \ldots, s_t^{i,n_{\boldsymbol{s}}}, \ldots, s_t^{i,N_{\boldsymbol{s}}}]$ at time t
	$a_t^i \in A^i$	Action of agent i at time t. $A^i = \{a_1^i, a_2^i, \ldots, a_{n_a}^i, \ldots, a_{\lvert A^i \rvert}^i\}$ is a set of candidate actions at agent i
	$a_t^{i,*}$	Optimal action of agent i at time t
	$r_{t+1}^i(s_{t+1}^i) \in \Re^i$	Delayed reward of agent i received under state s_{t+1}^i at time $t+1$. $\Re^i$ is a set of potential rewards at agent i
	$\delta_t^i(s_t^i, a_t^i)$	Temporal difference of a state-action pair (s_t^i, a_t^i) of agent i at time t
	$V_t^i(s_t^i)$	Value function of agent i under state s_t^i at time t
	$Q_t^i(s_t^i, a_t^i)$	Q-value of a state-action pair (s_t^i, a_t^i) of agent i at time t
	$0 < \alpha < 1$	Learning rate
	$0 < \gamma < 1$	Discount factor

The rest of this article is organized as follows. Sections 1.1 and 1.2 present an overview of RL and motivation, respectively. Section 2 presents the attributes of the traffic signal control problem. Section 3 presents the attributes of RL for traffic signal control. Section 4 presents the RL models and algorithms for traffic signal control, while Section 5 provides further enhancement of RL. Section 6 presents performance measures, simulation platforms, and complexity analysis. Section 7 presents open issues. Finally, Section 8 presents conclusions. In addition, Tables I and II present the generic notations used in this article and the specific notations used in the RL models and algorithms, respectively.

Table II. Specific Notations Used in RL Models and Algorithms

Category	Section	Notation	Description
Max-plus RL	4.2	$u_t^{ij}(a_t^j)$	Payoff message sent by agent i to a neighboring agent j, who is taking action a_t^j, at time t
		$f_t^i(a_t^i)$	Local payoff value of an action a_t^i of agent i at time t
		$f_t^{ij}(a_t^i, a_t^j)$	Payoff value of agent i's neighboring agent $j \in J^i$ when agent i takes action a_t^i and agent j takes action a_t^j at time t
		$g_t^i(a_t^i)$	Global payoff value of agent i when it takes action a_t^i at time t
Model-based RL	4.3	$f \in F$	Set of factors, such as waiting time, affecting the reward values
Multistep Backups RL	4.5	$0 \leq \varphi \leq 1$	Trace decay of an eligibility trace
RL with Function Approximation	4.6	θ_t^i	A d-dimensional tunable weight vector of agent i at time t with $\theta_t^i = (\theta_t^{i,1}, \ldots, \theta_t^{i,d})$
		$\sigma_t^i(s_t^i, a_t^i)$	A d-dimensional feature vector for a state-action pair (s_t^i, a_t^i) of agent i at time t with $\sigma_t^i(s_t^i, a_t^i) = (\sigma_t^{i,1}(s_t^i, a_t^i), \ldots, \sigma_t^{i,d}(s_t^i, a_t^i))$
		$n_{f,s_t^i}^i$	Number of features relevant to a particular state s_t^i
		$n_{f,a_t^i}^i$	Number of features relevant to a particular action a_t^i

1.1. Reinforcement Learning: An Overview

Reinforcement learning enables an agent to learn on the fly using temporal learning in a trial-and-error manner as time goes by $t = 1, 2, \ldots$ without the need for external supervision [Sutton and Barto 1998]. There are two main approaches of temporal difference learning, namely, off-policy and on-policy, that differ among themselves in the way in which the knowledge (or the *Q-value*) is updated. A popular off-policy approach is Q-learning [Zhao et al. 2012], and a popular on-policy approach is SARSA [Jin and Ma 2015a]. At time t, an agent i observes its Markovian (or memoryless) decision-making factors (or *state* $s_t^i \in S^i$) in the dynamic and stochastic operating environment, takes an *action* $a_t^i \in A^i$, and subsequently receives positive or negative consequences at the next time instant $t + 1$. There are two types of consequences, namely, the *delayed reward* $r_{t+1}^i(s_{t+1}^i) \in \Re^i$ received at time $t + 1$, and the *discounted reward* (see Section 1.1.1). As time progresses, the agent explores all state-action pairs (s_t^i, a_t^i) and ranks them according to the Q-value. The Q-value of a state-action pair represents the appropriateness of taking action a_t^i in state s_t^i in the long term, and it is maintained and updated in a Q-table with $|S^i| \times |A^i|$ entries using the Q-function as follows:

$$Q_{t+1}^i(s_t^i, a_t^i) \leftarrow Q_t^i(s_t^i, a_t^i) + \alpha \delta_t^i(s_t^i, a_t^i), \tag{1}$$

where $0 < \alpha < 1$ represents the learning rate and $\delta_t^i(s_t^i, a_t^i)$ represents the temporal difference, which is the difference between two successive estimations in terms of delayed and discounted rewards (see Section 1.1.1). Higher learning rate α increases responsiveness due to the preference to the temporal difference compared to the current Q-value $Q_t^i(s_t^i, a_t^i)$.

During action selection, either an *exploitation* or *exploration* action can be selected. Exploitation selects the best-known (or greedy) action that maximizes the *value function* as follows:

$$V_t^i(s_t^i) = \max_{a \in A} Q_t^i(s_t^i, a). \tag{2}$$

ALGORITHM 1: Traditional RL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
1: **procedure**			
2: Observe current state s_t^i			
3: Select action $a_t^{i,*}$ using Equation (3)			
4: Receive delayed reward $r_{t+1}^i(s_{t+1}^i)$			
5: Update Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ using Equation (1)	$\mathcal{O}(\|A\|)$		1
6: **end procedure**			

Hence, an agent selects an action with the maximum Q-value as follows:

$$a_t^{i,*} = \underset{a \in A}{\operatorname{argmax}} Q_t^i(s_t^i, a). \tag{3}$$

Exploration selects a random action so that its Q-value can be updated in order to discover better actions in a dynamic and stochastic operating environment as time progresses. Algorithm 1 shows the traditional RL algorithm embedded in each agent i, including its step-wise complexity (see Section 6.3). For simplicity, only exploitation action selection is shown in the algorithms presented in this article.

As an example, at time instant t, an agent i (e.g., a traffic signal controller at an intersection) observes its state s_t^i (e.g., the queue size of its lanes) and selects an action a_t^i (e.g., the traffic phase split of the current traffic phase). At the next time instant $t+1$, the agent receives delayed reward $r_{t+1}^i(s_{t+1}^i)$ (e.g., the waiting time of the vehicles), which is used to calculate temporal difference $\delta_t^i(s_t^i, a_t^i)$, as well as the next state s_{t+1}^i, from the operating environment. The agent selects the optimal action $a_t^{i,*}$ for the respective state $s_t^i \leftarrow s_{t+1}^i$ in order to maximize the value function $V_t^i(s_t^i)$ as time goes by $t = 1, 2, \ldots$. Due to the dynamicity of the state, learning is essential as the action that maximizes the rewards for a particular state varies with time.

1.1.1. Q-Learning and SARSA. There are two main approaches of temporal difference learning, namely, off-policy (i.e., Q-learning) and on-policy (i.e., SARSA) approaches, as follows (see Figure 4 for a taxonomy of the RL attributes for traffic signal control):

H.1 *Off-policy/Q-learning*. In the off-policy approach, particularly Q-learning, an agent i updates its Q-value $Q_t^i(s_t^i, a_t^i)$ based on the maximum Q-value of the next state s_{t+1}^i, which is based on the assumption that the agent chooses the optimal action while in any state. In Q-learning, the temporal difference $\delta_t^i(s_t^i, a_t^i)$ is calculated as follows:

$$\delta_t^i(s_t^i, a_t^i) = r_{t+1}^i(s_{t+1}^i) + \gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a) - Q_t^i(s_t^i, a_t^i), \tag{4}$$

where $0 < \gamma < 1$ represents the discount factor, and the discounted reward $\gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a)$ represents the highest possible accumulated reward expected to be received at time $t+1$ and so on. Higher discount factor γ increases the preference to the discounted reward. As an example, with $\gamma = 0.8$, a discounted reward is considered up to 30 iterations, after which its effect becomes negligible as $\gamma^{30} < 0.001$. Based on Equations (1) and (4), the Q-function is commonly written as follows:

$$Q_{t+1}^i(s_t^i, a_t^i) \leftarrow (1-\alpha) Q_t^i(s_t^i, a_t^i) + \alpha \left[r_{t+1}^i(s_{t+1}^i) + \gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a) \right]. \tag{5}$$

H.2 *On-policy/SARSA*. In the on-policy approach, particularly SARSA, an agent i updates its Q-value $Q_t^i(s_t^i, a_t^i)$ based on the Q-value $Q_t^i(s_{t+1}^i, a_{t+1}^i)$ of the next state s_{t+1}^i

after taking both current action a_t^i and next action a_{t+1}^i. In SARSA, the temporal difference $\delta_t^i(s_t^i, a_t^i)$ is calculated as follows:

$$\delta_t^i(s_t^i, a_t^i) = r_{t+1}^i(s_{t+1}^i) + \gamma Q_t^i(s_{t+1}^i, a_{t+1}^i) - Q_t^i(s_t^i, a_t^i), \tag{6}$$

where the discounted reward $\gamma Q_t^i(s_{t+1}^i, a_{t+1}^i)$ represents the accumulated reward of the next state-action pair (s_{t+1}^i, a_{t+1}^i) at time $t+1$. Based on Equations (1) and (6), the Q-function is commonly written as follows:

$$Q_{t+1}^i(s_t^i, a_t^i) \leftarrow (1-\alpha) Q_t^i(s_t^i, a_t^i) + \alpha \left[r_{t+1}^i(s_{t+1}^i) + \gamma Q_t^i(s_{t+1}^i, a_{t+1}^i) \right]. \tag{7}$$

Most of the traffic signal controllers proposed in the literature adopt Q-learning. There is only a perfunctory effort to investigate the use of SARSA to improve traffic signal control [Jin and Ma 2015b].

1.2. Motivation of Using Reinforcement Learning for Traffic Signal Control

The use of RL for traffic signal control is motivated by four main advantages. First, RL adapts to real-time traffic, which evolves in a complex and unpredictable manner due to unexpected circumstances such as road accidents and bad weather conditions. Second, RL is a model-free approach that enables an agent to perform self-learning on the fly without the need of external supervision and prior knowledge (e.g., transition probability) of the operating environment. Third, an explicit predefined model covering every single factor affecting the system performance from the operating environment is not required because the delayed reward uses system performance metric(s), which takes account of various factors, to represent the main goals. Fourth, RL addresses the *curse of dimensionality* in which the number of state-action pairs becomes large in number (for discrete space) or unlimited (for continuous space) as it increases exponentially with the number of agents, leading to two main shortcomings that reduce scalability: (1) larger storage capacity is required to store the Q-values, and (2) higher computational cost and longer learning time are required to explore the large number of state-action pairs impeding timely selection of optimal actions. RL possesses added advantages compared to other artificial intelligence approaches. Fuzzy logic defines membership functions that map inputs to outputs, and so it does not learn about the outputs for inputs that lie outside its range [Azimirad et al. 2010; Jain and Sethi 2012]. Genetic algorithm requires complicated representations and a larger number of chromosomes as the problem expands [Teo et al. 2010]. Dynamic programming requires mechanisms to address computational intractability, as well as to compute the transition probability of the operating environment [El-Tantawy and Abdulhai 2012].

2. ATTRIBUTES OF TRAFFIC SIGNAL CONTROL PROBLEM

The attributes of the intersections, traffic, and traffic signal controllers have posed significant challenges to traffic management. This section presents such attributes in order to better understand the traffic signal control problem to be solved by the RL models and algorithms in the subsequent sections. Figure 1 presents various aspects of the traffic signal control problem.

2.1. Challenges

RL models and algorithms address the following two main challenges of traffic signal control:

C.1 *Inappropriate traffic phase sequence*. A traffic phase consists of a combination of green signals allocated to a set of lanes simultaneously for nonconflicting movements at an intersection. Different kinds of traffic phases, either with (T.2.1)

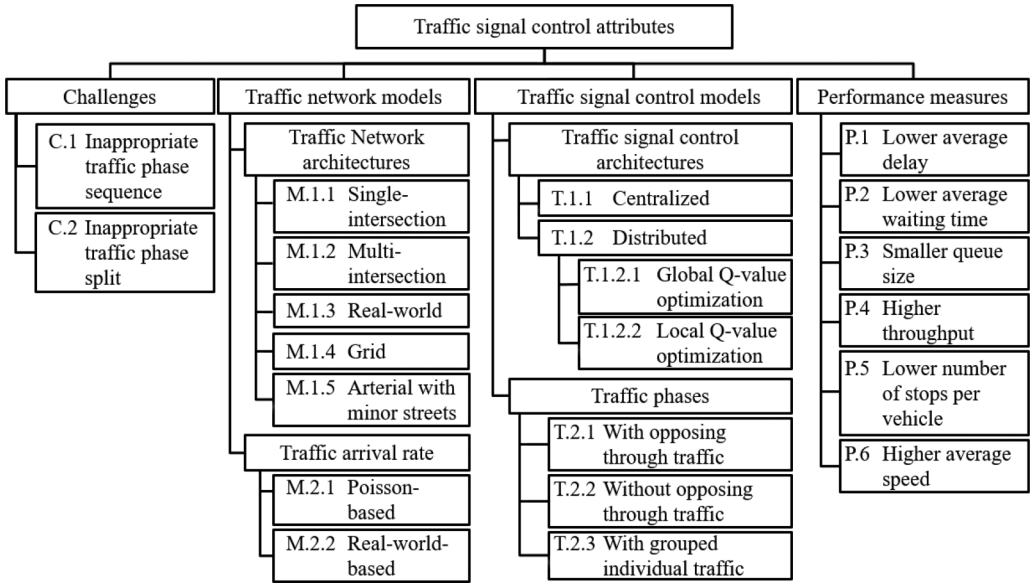


Fig. 1. Traffic signal control attributes.

or without (T.2.2) opposing through traffic, or with group-based individual traffic (T.2.3), can be activated in an in-order (i.e., round-robin) or out-of-order manner. Activating an inappropriate traffic phase can cause congestion, cross-blocking, or green idling.

C.2 *Inappropriate traffic phase split*. A traffic phase split is the time interval allocated for a traffic phase. For simplicity, we can focus on the traffic phase with green signals, while the rest of the traffic phases are receiving red signals. Extended green time can cause cross-blocking when the traffic volume is high and green idling when the traffic volume is low. Too short of a green time can increase the queue size of a lane, resulting in congestion. Maximum and minimum durations for green time can be imposed at lanes. A maximum duration prevents long waiting time for vehicles at other lanes, while a minimum duration ensures that at least a single waiting vehicle can cross an intersection.

2.2. Traffic Network Models

The traffic networks can be characterized by their architectures and traffic arrival rate, which characterizes the traffic conditions.

2.2.1. Traffic Network Architectures. A traffic network consists of intersections and edge nodes. Each intersection has multiple legs in different directions, and each leg has a single or multiple lanes so that a vehicle can either turn left, go straight, or turn right. In a closed traffic environment, a vehicle enters the traffic network through an edge node, traverses from one intersection to another, and leaves through another edge node. In this article, a left-hand traffic network is considered, although a similar description applies to a right-hand traffic network.

M.1.1 *Single-intersection traffic network* consists of a single intersection in the traffic network.

M.1.2 *Multi-intersection traffic network* consists of multiple intersections, such as two intersections with a single closed link in between the intersections (see

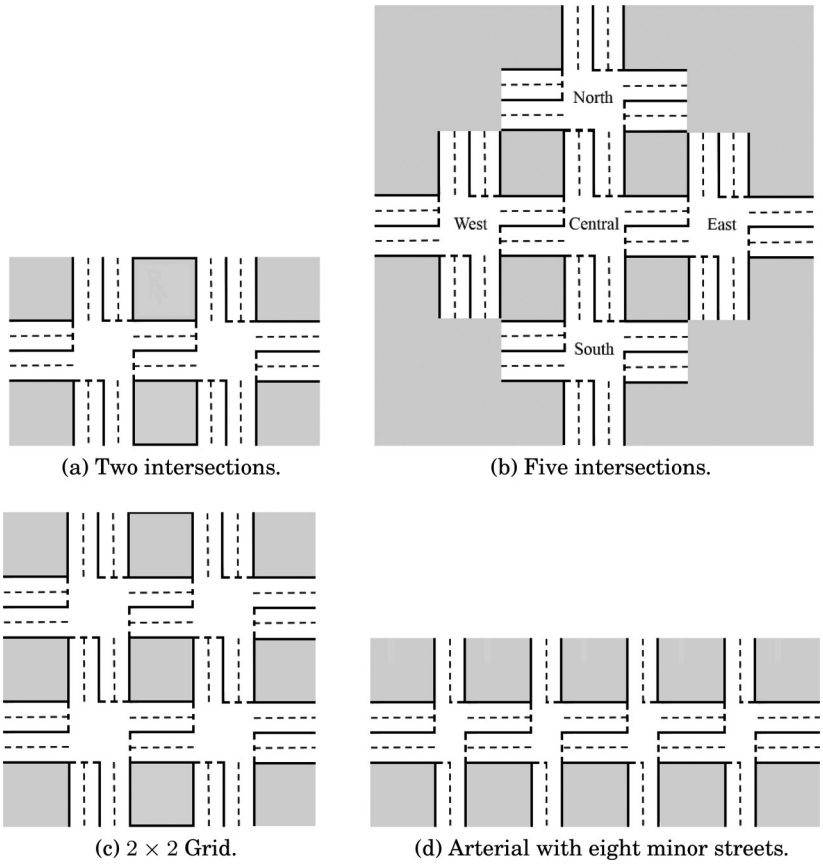


Fig. 2. Traffic network architectures.

Figure 2(a)) and five intersections with a central intersection and four outbound intersections (see Figure 2(b)) [Yin et al. 2016]. Each vehicle crosses either one intersection (e.g., north) or three intersections (e.g., north-central-south).

M.1.3 *Real-world traffic network*, which is based on the layout of a city, considers a larger number of intersections, such as 20 intersections in Bangalore [Prabuchandran et al. 2015], 59 in Toronto [El-Tantawy et al. 2013], and 64 in Dublin [Salkam et al. 2008].

M.1.4 *Grid traffic network* consists of intersections connected in a grid topology such as 2×2 (see Figure 2(c)) and 3×3 traffic networks.

M.1.5 *Arterial with minor streets traffic network* consists of an arterial and minor streets, such as five minor streets [Medina et al. 2010] and eight minor streets (see Figure 2(d)) [Prashanth and Bhatnagar 2011]. The arterial has more vehicles, such as 95% in the arterial and 5% in the minor streets [Prashanth and Bhatnagar 2012].

A traffic network also consists of hardware devices installed at an intersection (e.g., inductive loop detectors and video-based traffic detectors) or a vehicle (e.g., vehicular sensors) to gather local statistics (e.g., the traffic arrival and departure rates, as well as the waiting time, of the vehicles) at the intersection over time. Subsequently,

the devices process (e.g., aggregate) and send the statistics to the agents (e.g., the traffic signal controllers) in order to estimate useful longer-term information (e.g., the average queue size of the lanes). Alternatively, the hardware devices (e.g., camera sensors) can gather shorter-term information (e.g., the instantaneous queue size) at the intersection at any time instant. The traffic signal controllers can communicate using wired connections, while the vehicles can communicate using wireless communication. At an intersection i, the inductive loop detectors can be placed: (1) near the stop line to detect small queue size Ψ_q^1 and measure the waiting time of a vehicle, and (2) further away from the stop line to detect large queue size Ψ_q^2 and measure the time gap between two vehicles such that a reduced gap indicates increased traffic volume [Jin and Ma 2015b]. The two detectors segregate the queue size $n_{q,t}^i$ into three levels to represent different congestion levels: low $n_{q,t}^i < \Psi_q^1$, medium $\Psi_q^1 \leq n_{q,t}^i < \Psi_q^2$, and high $n_{q,t}^i \geq \Psi_q^2$ [Prabuchandran et al. 2014]. Traffic statistics (e.g., the traffic departure rate) gathered by the outgoing leg of an intersection can be sent to neighboring intersections to predict their traffic arrival rates.

2.2.2. Traffic Arrival Rate. Traffic arrival rate characterizes the number of vehicles entering a traffic network through edge nodes or arriving at an intersection within a time unit (e.g., 10,000 vehicles per hour) [Prabuchandran et al. 2015]. The traffic arrival rate affects the traffic volume to provide a sparse or crowded traffic network. A vehicle arriving at one of the legs of an intersection is placed at the end of a queue in one of the lanes. Each lane can accommodate a maximum number of vehicles. Either distributions or a traffic model can be used to characterize the steady-state dynamics of the traffic arrival rate as follows:

M.2.1 *Poisson-based traffic arrival process* calculates the probability of n vehicles arriving at an intersection i within a time window t_w, specifically $P_{t_w}^{i,n} = [(\beta^i t_w)^n / n!] e^{-\beta^i t_w}$. The interarrival time of the arriving vehicles is exponentially distributed with a mean traffic arrival rate β^i. Queuing models based on the Poisson process, such as M/M/1, can be adopted to model memoryless and deterministic queueing scenarios at an intersection [Chanloha et al. 2012].

M.2.2 *Real-world-based traffic arrival process* provides the vehicles with traversing properties (e.g., regular and maximum driving speed, driving direction, physical position of the destination, vehicle overtaking, and lane switching) following some real-world traffic characteristics, such as the Nagel-Schereckenberg model [Nagel and Schreckenberg 1992] and the vehicle-following model [Yin et al. 2016].

2.3. Traffic Signal Control Models

The traffic signal controllers can be characterized by their architectures and traffic phases. In general, the architectures represent the traffic signal controllers and their relationships at the network (or global) level, while the traffic phases represent a combination of green signals activated by a traffic signal controller at the individual (or local) level.

2.3.1. Traffic Signal Control Architectures. There are two main types of traffic signal control architectures as follows:

T.1.1 *Centralized model* consists of a central agent that gathers local statistics (e.g., the queue size of the lanes) from distributed agents (e.g., neighboring intersections), selects an action (e.g., the traffic phase split of the current traffic phase) that

optimizes the system-wide performance, and sends the decisions made back to the distributed agents who follow the decision made. There are three main open issues with regard to robustness, scalability, and efficiency. First, the centralized model is prone to a single point of failure as a malfunctioning central agent can affect the traffic conditions of the entire traffic network. Second, the curse of dimensionality becomes critical at the central agent. Third, communication overhead increases at the central agent. The centralized model has been widely adopted by traditional traffic signal control, including SCAT [Sims and Dobinson 1980], SCOOT [Robertson and Bretherton 1991], and GLIDE [Keong 1993].

T.1.2 *Distributed model* consists of a number of distributed agents that gather local statistics and select their respective actions. Hence, a complex problem can be segregated into subproblems solved by the distributed agents. In general, this model improves robustness, scalability, and efficiency. In the context of RL, there are two types of distributed models as follows:

T.1.2.1 *Global Q-value optimization* optimizes a global Q-value computed at the system-wide level in order to achieve the global objective of the agents in a traffic network. To provide a global view of the operating environment, the distributed agents observe their respective local operating environment and learn about others' information (e.g., rewards and Q-values). Subsequently, they select their respective actions while ensuring that the global Q-value converges to an optimal equilibrium in order to achieve an optimal joint action as time goes by.

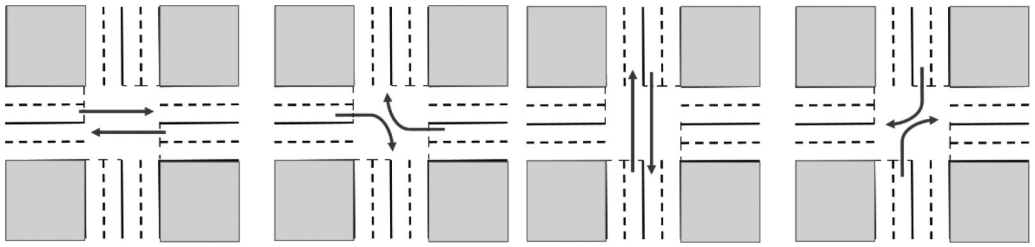
T.1.2.2 *Local Q-value optimization* optimizes the local Q-value computed at the individual level in order to achieve the objective of the individual agents in a traffic network. An agent selects its actions without considering its neighboring agents' actions. Without information exchange, the communication overhead reduces; however, the global Q-value may not converge to an optimal equilibrium.

2.3.2. Traffic Phases. A traffic phase consists of a combination of green signals assigned to a set of lanes simultaneously for nonconflicting movements at an intersection. Since a left-hand traffic network is considered, the left turn is not considered for simplicity because it is either a protected traffic movement or nonconflicting traffic movement. There are three main types of traffic phases as follows:

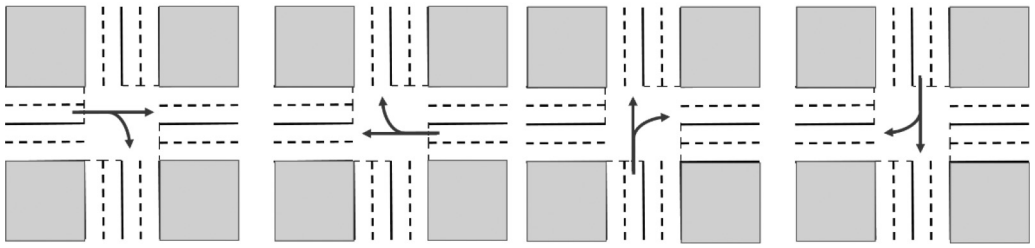
T.2.1 *Traffic phases with opposing through traffic* consist of a four-phase traffic sequence of opposing through traffic (see Figure 3(a)). It is normally adopted at intersections where (1) either turning or through traffic volume is significantly higher, and (2) the turning and through traffic do not share a single lane [Yin et al. 2016].

T.2.2 *Traffic phases without opposing through traffic* consist of a four-phase traffic sequence of traffic from a particular leg (see Figure 3(b)). It is normally adopted at intersections where (1) the traffic volumes of turning and through traffic are comparable, and (2) the turning and through traffic share a single lane [Teo et al. 2014].

T.2.3 *Traffic phases with grouped individual traffics* consist of a traffic sequence in which each traffic movement receives a green signal individually for a particular time period as long as the selected combination of traffic movements are nonconflicting at an intersection [Jin and Ma 2015b]. In other words, a traffic phase is a random (rather than fixed) combination of nonconflicting traffic movements. So, a traffic movement to be activated with green signal must not conflict with the movements in the current traffic phase.



(a) Traffic phases with opposing through traffic.



(b) Traffic phases without opposing through traffic.

Fig. 3. Traffic phases.

Further details about the traffic signal control attributes in Figure 1, particularly the performance measures achieved by the RL models and algorithms, are shown in Section 6.1.

3. APPLICATION OF REINFORCEMENT LEARNING TO TRAFFIC SIGNAL CONTROL

The traditional Q-learning approach has been widely adopted in the literature [Abdoos et al. 2011; Araghi et al. 2013; Arel et al. 2010; Chanloha et al. 2012; Chin et al. 2012; Dai et al. 2010; Teo et al. 2014], and it has been extended with enhanced features as presented in Sections 4 and 5. In traffic signal control, RL can be embedded in the following autonomous agents:

G.1 *Traffic signal controller* to coordinate vehicles. Each traffic signal controller [Prabuchandran et al. 2015] can be embedded with a single agent. Alternatively, each lane of an intersection can be embedded with a single agent to provide greater flexibility [Su and Tham 2007], which is suitable for traffic phases with grouped individual traffic (T.2.3).

G.2 *Vehicle* to estimate its waiting time [Khamis and Gomaa 2012, 2014]. So, a traffic signal controller gathers and processes the waiting time of all vehicles at an intersection.

G.3 *Traffic movement* to assess its appropriateness to be activated with a green signal in order to constitute one of the nonconflicting movements in a traffic phase.

The rest of this section presents the representations and a taxonomy (see Figure 4) of RL for solving the traffic signal control problem.

3.1. State Representation

The state $s_t^i \in S^i = \{s_1^i, s_2^i, \ldots, s_{n_s}^i, \ldots, s_{|S^i|}^i\}$ represents an agent's decision-making factors. Each state can consist of multiple substates (e.g., $\boldsymbol{s_t^i} = [s_t^{i,1}, s_t^{i,2}]$). The substates $s_t^{i,1}$ and $s_t^{i,2}$ may represent two distinguishing factors. As an example, at an intersection i, (1) substate $s_t^{i,1}$ represents the queue size at intersection i, which is the immediate

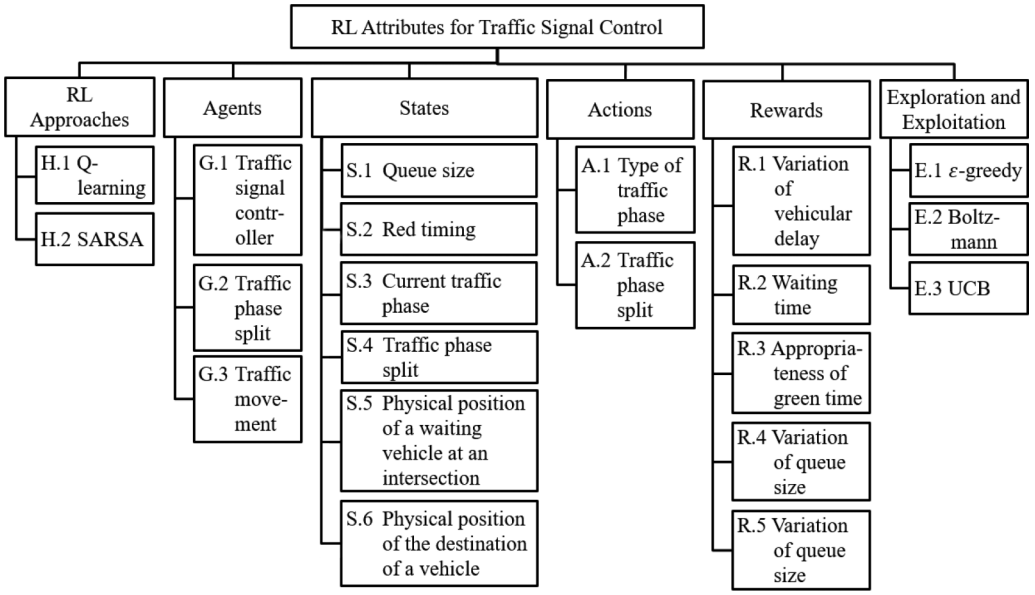


Fig. 4. Taxonomy of RL attributes for traffic signal control.

consequence of action a_t^i, and (2) substate $s_t^{i,2}$ represents the number of arriving vehicles from a neighboring intersection j, which is the future consequence of its action a_t^j that cannot appear at intersection i immediately [Yin et al. 2016]. Hence, an agent's state can incorporate its neighboring agents' states to improve traffic conditions among neighboring intersections [Medina et al. 2010]. For simplicity and consistency with the traditional RL approach, the state $\boldsymbol{s_t^i}$ is written as s_t^i henceforth, unless otherwise mentioned. There are six main representations for a state s_t^i:

S.1 *Queue size* represents the number of vehicles waiting in a lane or a leg (e.g., speed is less than 5kph), and it changes with the traffic departure and arrival rates. There are three main approaches to represent a queue size at an intersection i with four legs $D^i = [1, 2, 3, 4]$; each $d^i \in D^i$ has a single lane. First, the state $\boldsymbol{s_t^{i,d^i}} = [s_t^{i,1}, s_t^{i,2}, s_t^{i,3}, s_t^{i,4}]$ can represent the queue size $n_{q,t}^{i,d^i}$ of a lane, which can be characterized by different levels: low ($s_t^{i,d^i} : n_{q,t}^{i,d^i} < \Psi_q^1$), medium ($s_t^{i,d^i} : \Psi_q^1 < n_{q,t}^{i,d^i} < \Psi_q^2$), and high ($s_t^{i,d^i} : n_{q,t}^{i,d^i} > \Psi_q^2$), where $\Psi_q^1 < \Psi_q^2$ are the thresholds [Teo et al. 2014], and so the number of states is $4^4 = 256$. Second, the state s_t^i can represent the relative queue size among the four lanes. An example is ($n_{q,t}^{i,2} > n_{q,t}^{i,4} > n_{q,t}^{i,1} > n_{q,t}^{i,3}$), which indicates that the largest queue size is lane $d^i = 2$, followed by $d^i = 4$, $d^i = 1$ and $d^i = 3$ [Abdoos et al. 2011], and so the number of states is $4! = 24$, which is lower than that in the first approach. However, the relative queue size does not indicate the congestion level; for instance, $n_{q,t}^{i,1} < n_{q,t}^{i,2}$ does not differentiate both ($n_{q,t}^{i,1} = 1, n_{q,t}^{i,2} = 5$) and ($n_{q,t}^{i,1} = 10, n_{q,t}^{i,2} = 50$) cases. Third, the state s_t^i can represent the maximum queue size among the lanes, and so the number of states is the maximum queue size $n_{q,t}^{i,d^i}$ [El-Tantawy and Abdulhai 2010].

S.2 *Red timing* represents the time elapsed since the signal of a lane turned into red. It is used to prevent red signals from being activated for too long to improve

fairness among the lanes. Consider an intersection i with four legs $D^i = [1, 2, 3, 4]$; each $d^i \in D^i$ has a single lane. The state $\boldsymbol{s_t^{i,d}} = [s_t^{i,1}, s_t^{i,2}, s_t^{i,3}, s_t^{i,4}]$ can represent the red timing $t_{r,t}^{i,d^i}$ of the lanes, which can be characterized by different levels: low ($s_t^{i,d^i} : t_{r,t}^{i,d^i} < \Psi_r^1$) and high ($s_t^{i,d^i} : t_{r,t}^{i,d^i} > \Psi_r^1$), and so the number of states is $4^2 = 16$ [Prabuchandran et al. 2015].

S.3 *Current traffic phase* represents a combination of green signals activated simultaneously for nonconflicting movements at an intersection. The state $s_t^i \in S^i = \{s_1^i, s_2^i, \ldots, s_{n_s}^i, \ldots, s_{|S^i|}^i\}$ can represent the traffic phase at intersection i at time t, and so the number of states is by the number of candidate traffic phases $|S^i|$ [Arel et al. 2010].

S.4 *Traffic phase split* represents the time interval allocated for a traffic phase. The state $s_t^i \in S^i = \{s_1^i, s_2^i, \ldots, s_{n_s}^i, \ldots, s_{|S^i|}^i\}$ can represent one of the elapsed times for the current traffic phase at intersection i at time t, and so the number of states is the maximum level of a traffic phase split $|S^i|$.

S.5 *Physical position of a waiting vehicle at an intersection*. Consider a lane segmented into small cells; each can accommodate a vehicle. The state $s_t^i \in S^i = \{s_1^i, s_2^i, \ldots, s_{n_s}^i, \ldots, s_{|S^i|}^i\}$ represents a cell position from the intersection at time t, with the cell $s_1^i = 1$ being nearest to the intersection i, and so the number of states is the maximum queue size $s_{|S^i|}^i$.

S.6 *Physical position of the destination of a vehicle*. The state $s_t^i \in S^i = \{s_1^i, s_2^i, \ldots, s_{n_s}^i, \ldots, s_{|S^i|}^i\}$ represents one of the edge nodes, and so the number of states is the number of edge nodes in a traffic network $s_{|S^i|}^i$.

3.2. Action Representation

The action $a_t^i \in A^i = \{a_1^i, a_2^i, \ldots, a_{n_a}^i, \ldots, a_{|A^i|}^i\}$ represents an agent's selected action. There are two main representations for an action a_t^i:

A.1 *Type of traffic phase* represents the selection of a combination of green signals to be activated simultaneously for nonconflicting movements at an intersection. The activation of the traffic phases can be (1) in-order, in which the traffic phases are activated in a round-robin manner $a_t^i = [a_n^i | n = (n_a + 1) \text{mod} |A^i|]$, and (2) out-of-order, in which it is similar to the equivalent state representation (S.3), and action $a_t^i \in A^i = \{a_1^i, a_2^i, \ldots, a_{n_a}^i, \ldots, a_{|A^i|}^i\}$ can be applied to represent one of the traffic phases to be activated at intersection i at time t. The number of actions is given by the number of traffic phases $|A^i|$ in both approaches.

A.2 *Traffic phase split* represents the selection of a time interval to be allocated for a traffic phase at an intersection. There are two main approaches to represent a traffic phase split at an intersection i. First, the action $a_t^i \in A^i = \{a_1^i, a_2^i, \ldots, a_{n_a}^i, \ldots, a_{|A^i|}^i\}$ represents the traffic phase split for one of the lanes of intersection i with $a_1^i < a_2^i < \cdots < a_{|A^i|}^i$ [Araghi et al. 2013]. The number of actions is given by the number of candidate traffic phases $|A^i|$. Second, $a_t^i \in A^i = \{a_1^i, a_2^i\}$ represents the choice of either keeping the current traffic phase (a_1^i) or switching to another traffic phase (a_2^i). An agent takes action a_1^i until it does not yield the best reward, and so it switches to action a_2^i [Chanloha et al. 2012]. The traffic phase split t_p^i can increase in multiples of a time slot, specifically $t_p^i = t_{p,\min}^i + (n_{slot} \times t_{slot})$, where $t_{p_{\min}}^i$ represents the minimum traffic phase split, t_{slot} represents a time slot, and n_{slot} represents the number of time slots [Abdoos et al. 2011].

3.3. Reward Representation

The delayed reward $r^i_{t+1}(s^i_{t+1}) \in \Re^i$ represents the goal, where $\Re^i$ is a set of potential rewards at agent i. For simplicity, the delayed reward $r^i_{t+1}(s^i_{t+1})$ is written as r^i_{t+1} in this subsection. The reward value can be a fixed value (e.g., $r^i_{t+1} = 1$ represents a reward and $r^i_{t+1} = 0$ represents a punishment [Teo et al. 2014]), or a variable. A variable reward can consist of multiple elements. With multiple elements, there are three main approaches to calculate the variable reward value. First, the reward $r^i_{t+1} = r^{i,1}_{t+1} + r^{i,2}_{t+1}$ has two elements $r^{i,1}_{t+1}$ and $r^{i,2}_{t+1}$ with equal priority [Prashanth and Bhatnagar 2012]. Second, a weighted reward $r^i_{t+1} = [\eta_1 \times (r^{i,j,1}_{t+1} + r^{i,j,2}_{t+1})] + [(1-\eta_1) \times (r^{i,k,1}_{t+1} + r^{i,k,2}_{t+1})]$ with $0 \leq \eta_1 \leq 1$ provides two priority levels to two different elements. In this case, lane j (e.g., main street) has higher priority than lane k (e.g., minor street) at intersection i [Liu et al. 2014]. Third, a weighted reward with multiple weights, specifically $r^i_{t+1} = [\eta_1 \times [(\eta_2 \times r^{i,j}_{t+1,1}) + ((1-\eta_2) \times r^{i,k}_{t+1,1})]] + [(1-\eta_1) \times [(\eta_2 \times r^{i,j}_{t+1,2}) + ((1-\eta_2) \times r^{i,k}_{t+1,2})]]$, where $0 \leq \eta_1, \eta_2 \leq 1$ provides more than two priority levels to two different elements [Prashanth and Bhatnagar 2012]. For instance, higher η_1 reduces queue size, while lower η_1 reduces red timing; and higher η_2 ensures lane j (e.g., main street) has higher priority than lane k (e.g., minor street) at intersection i. There are five main representations for a reward r^i_{t+1}:

R.1 *Variation of vehicular delay.* An agent is rewarded when the vehicular delay of a lane reduces. There are two main approaches to represent the variation of vehicular delay at an intersection i with four legs $D^i = [1, 2, 3, 4]$; each $d^i \in D^i$ has a single lane. First, the reward $r^{i,j}_{t+1}$ is an absolute value of the total delay of vehicles in lane j of intersection i [El-Tantawy and Abdulhai 2010]. Second, the reward $r^{i,j}_{t+1}$ is a relative value, which can be a difference or a ratio. The reward $r^i_{t+1} = t^i_{v,t} - t^i_{v,t+1}$ represents the difference of the total delay of all vehicles at intersection i at time $t+1$ and time t (or between traffic phases), or a reference value if $t^i_{v,t} = t^i_{v,ref}$ [Jin and Ma 2015b]. In addition, the reward $r^{i,d^i}_t = t^{i,d^i}_{v,t} / \sum_{c \in D^i} [t^{i,c}_{v,t} / |D^i|]$ represents a ratio of the total delay of the vehicles in a lane d^i to the average delay of the vehicles in all lanes of intersection i [Arel et al. 2010].

R.2 *Waiting time*. An agent is penalized when the average waiting time of the vehicles at an intersection increases due to red signal, congestion, or cross-blocking. As an example, in Li et al. [2009], the reward $r^i_{t+1} = 1/t^i_{w,t+1}$ is inversely proportional to the average waiting time of the vehicles at intersection i at time $t+1$.

R.3 *Appropriateness of green time*. An agent is penalized with $r^i_{t+1} = -1$ when the green timing is inappropriate, resulting in either cross-blocking or green idling; otherwise, it is awarded with $r^i_{t+1} = 1$ [Teo et al. 2014]. In general, the loss of green time in a traffic phase is a common loss shared by other traffic phases at an intersection.

R.4 *Variation of queue size*. An agent is rewarded when the queue size reduces with an increasing number of vehicles crossing an intersection within a time interval (e.g., a single cycle) [Teo et al. 2014]. There are two main approaches to represent the variation of queue size in a lane. First, similar to the reward representation (R.1), it can be based on a relative value $n^i_{c,t} - n^i_{c,t+1}$, which represents the difference of the number of vehicles crossing an intersection i at time t and time $t+1$ (or in between cycles) [Mikami and Kakazu 1994]. Second, the reward $r^i_{t+1} = n^i_{c,t+1} - n^i_{q,t+1}$ represents the difference between the number of vehicles crossing an intersection $n^i_{c,t+1}$ and the queue size $n^i_{q,t+1}$, which indicates whether the green

time is sufficient, at intersection i. Specifically, a positive reward indicates that the number of vehicles crossing an intersection is higher than the queue size of the respective lane at the end of a cycle, while a negative reward indicates otherwise [Araghi et al. 2013].

R.5 *Cost of phase transition* represents the cost, which is the time delay incurred during a traffic phase transition (e.g., green to red signal transition) [Medina and Benekohal 2012].

3.4. Exploration and Exploitation

Three main approaches to select either an exploration or an exploitation action are as follows:

E.1 The *ε-greedy* approach selects the greedy action with exploitation probability $1-\varepsilon$ and a random action with a small exploration probability ε.

E.2 The *Boltzmann (or softmax)* approach calculates the probability of an action $a \in A^i$ being selected by agent i under state s_t^i at time t:

$$P_t^i(s_t^i, a) = \frac{e^{Q_t^i(s_t^i,a)/\tau}}{\sum_{b \in A^i} e^{Q_t^i(s_t^i,b)/\tau}}, \tag{8}$$

where higher temperature τ increases exploration. The greedy action $a_t^{i,*}$, which has the highest Q-value under state s_t^i at time t, has the highest probability $P_t^i(s_t^i, a)$, while the rest of the actions $a_t^i \in A \backslash a_t^{i,*}$ have lower probability $P_t^i(s_t^i, a)$.

E.3 The *UCB* approach selects an action based on the upper confidence bound index:

$$a_t^{i,*} = \underset{a \in A}{\operatorname{argmax}} \left\{ -Q_t^i(s_t^i, a) + \sqrt{\frac{\ln N_t(s^i)}{N_t(s^i, a)}} \right\}, \tag{9}$$

where $N_t(s^i)$ and $N_t(s^i, a)$ represent the number of times a state s^i and a state-action pair (s^i, a) have been visited up to time t, respectively. The second term is inversely proportional to $N_t(s^i, a)$; so, lower $N_t(s^i, a)$ increases exploration due to the domination of the second term over the first term [Prabuchandran et al. 2015].

The selection of exploration actions can affect the performance. As an example, in a cliff-walking task [Sutton and Barto 1998], the selection of the exploration probability ε in the ε-greedy approach affects the performance achieved by Q-learning (H.1). Specifically, SARSA (H.2) has been shown to outperform Q-learning; however, when the exploration probability becomes a zero value $\varepsilon \to 0$, both Q-learning and SARSA can achieve optimal performance. This is because the optimal and worst actions are close to each other in a cliff-walking task, and exploration causes Q-learning to select the worst action; however, when the exploration probability is reduced as time goes by, the performance achieved by Q-learning improves.

4. REINFORCEMENT LEARNING MODELS AND ALGORITHMS FOR TRAFFIC SIGNAL CONTROL

This section presents the RL models and algorithms applied to traffic signal control. Table III presents a summary of RL models and algorithms, as well as their strengths to address the challenges of traffic signal control. Tables IV, V, and VI present a summary of the traffic signal control attributes applied in RL-based traffic signal control systems and RL attributes for traffic signal control, as well as simulation platforms and performance measures achieved by RL, respectively.

Table III. Summary of RL Models and Their Strengths to Address the Challenges of Traffic Signal Control

RL Model	Section	Description	Strength
Multiagent reinforcement learning (MARL)	4.1	Agents learn about each other's information (e.g., delayed reward and Q-value) and use it to select their respective actions as part of the optimal joint action.	Decomposes a complex problem into smaller problems and addresses the curse of dimensionality.
Max-plus RL	4.2	Agents calculate and exchange payoff values and use them to select their respective actions as part of the optimal joint action.	Decomposes a complex problem into smaller problems and addresses the curse of dimensionality.
Model-based approach	4.3	Agents create models of the operating environment or the action selection by neighboring agents.	Improves learning speed (or the convergence rate).
Actor-critic approach	4.4	Agents have their respective actor and critic. The critic corrects the delayed rewards using temporal difference. Subsequently, the actor updates the Q-values using the temporal difference.	Improves learning speed (or the convergence rate).
Multistep backup reinforcement learning	4.5	Agents update the Q-values based on the average effects of the temporal differences gathered in an episode (or a sequence of time instants).	Takes account of longer-term effects of an action under a state.
RL with function approximation	4.6	Agents represent Q-values using tunable weight vectors and feature vectors.	Addresses the curse of dimensionality.

4.1. Multiagent Reinforcement Learning

Consider a set of agents $I = [1, 2, \ldots, i, \ldots, |I - 1|, |I|]$ in the operating environment. With the increasing number of agents, the number of state-action pairs increases exponentially $|S^1| \times \cdots \times |S^i| \times \cdots \times |S^{|I|}| \times |A^1| \times \cdots \times |A^i| \times \cdots \times |A^{|I|}|$. Multiagent RL (MARL) enables the agents to exchange information (e.g., the delayed reward and Q-value of their state-action pair) and coordinate their respective actions in order to achieve global Q-value optimization (T.1.2.1). Therefore, the traditional RL approach maximizes the local Q-value, while the MARL approach decomposes a complex problem into smaller problems solved by the agents via collaboration and parallelism. Hence, MARL is more scalable (i.e., with distributed and decentralized processing) and robust (i.e., without single point of failure).

In MARL, the main challenge of the agents is to learn and select their respective actions in a moving target and shared environment. Due to the moving target, the agents learn and select their respective actions as part of the joint action simultaneously. Due to the shared environment, the actions of an agent at an intersection can affect, and vary with, the actions selected by the agents at neighboring intersections, which affect the agent's own performance in return. The main reason is that, as vehicles traverse from one intersection (or agent) to another, the state (or traffic conditions and congestion level) at a downstream intersection reflects the consequence of the action selected by an upstream intersection. An agent's action at an intersection can cause congestion in neighboring intersections; thus, each agent must consider the optimal actions taken by its neighboring agents and coordinate with each other. As an example, the agents (i.e., traffic signal controllers) exchange information (i.e., queue size) and coordinate actions (i.e., the type of traffic phase to be activated in the next time instant (A.1))

Table IV. Summary of Traffic Signal Control Attributes Applied in RL-Based Traffic Signal Control Systems

RL Models	Challenges			Traffic Network Models										Traffic Signal Control Models							
				Network Architectures							Traffic Characteristics			Traffic Signal Control Architectures					Traffic Phases		
	(C.1) Inappropriate traffic phase sequence	(C.2) Inappropriate traffic phase split	(M.1.1) Single intersection	(M.1.2) Multi-intersection	(M.1.3) Real world	(M.1.4) Grid	(M.1.5) Arterial with minor streets	(M.2.1) Poisson based	(M.2.2) Real-world based	(T.1.1) Centralized model	(T.1.2.1) Global Q-value optimization	(T.1.2.2) Local Q-value optimization	(T.2.1) With opposing through traffic	(T.2.2) Without opposing through traffic	(T.2.3) With grouped individual traffic
MARL [Prabuchandran et al. 2015]	*				*				*		*				
Max-plus RL [Medina and Benekohal 2012]	*				*				*		*				
Model-based RL [Khamis et al. 2012a]	*	*				*			*	*					
Model-based RL [El-Tantawy et al. 2013]	*	*			*				*		*				
Actor-critic RL [Li et al. 2009]		*	*					*		*					
Multistep backups RL [Jin and Ma 2015b]	*	*	*						*			*			*
RL with function approximation [Prashanth and Bhatnagar 2012]	*			*	*	*	*	*		*					

Table V. Summary of RL Attributes for Traffic Signal Control Systems

RL Models	RL-Approaches		Representations																		
			Agent			State						Action		Reward					Exploration and Exploitation		
	(H.1) Q-learning	(H.2) SARSA	(G.1) Traffic signal controllers	(G.2) Vehicles	(G.3) Traffic movement	(S.1) Queue size	(S.2) Red timing	(S.3) Current traffic phase	(S.4) Traffic phase split	(S.5) Position of a waiting vehicle	(S.6) Position of the destination of a vehicle	(A.1) Type of traffic phase	(A.2) Traffic phase split	(R.1) Variation of vehicular delay	(R.2) Waiting time	(R.3) Appropriateness of green time	(R.4) Variation of queue size	(R.5) Cost of phase transition	(E.1) ε-greedy	(E.2) Boltzmann	(E.3) UCB
MARL [Prabuchandran et al. 2015]	*		*			*	*					*			*		*		*		*
Max-plus RL [Medina and Benekohal 2012]	*		*			*		*	*			*				*	*	*			
Model-based RL [Khamis et al. 2012a]	*			*				*		*	*	*			*						
Model-based RL [El-Tantawy et al. 2013]	*		*			*		*	*			*		*					*		
Actor-critic RL [Li et al. 2009]	*		*			*		*					*		*					*	
Multistep backups RL [Jin and Ma 2015b]		*			*			*	*	*		*	*	*						*	
RL with function approximation [Prashanthand Bhatnagar 2012]	*		*			*	*					*			*		*			*	

among themselves in order to reduce red time (R.2) and the queue size of the lanes (R.4) [Prabuchandran et al. 2015].

There are three main steps for the distributed agents. First, since each agent has a local view of the operating environment only, it must learn about its neighboring agents' information, such as state, action, reward [Prabuchandran et al. 2015], and Q-value [Su and Tham 2007], via direct communication or without communication. Without communication, the agents create models of the operating environment to infer the information that helps to reduce communication overhead. Second, the agents update their respective Q-values using the neighboring agents' information. An agent i may use delayed rewards from itself and its neighboring agents to calculate a weighted sum of the delayed rewards [Arel et al. 2010] or may use value functions (see Equation (2)) received from neighboring agents to update the Q-value as follows [Liu et al. 2014]:

$$Q^i_{t+1}(s^i_t, a^i_t) \leftarrow (1-\alpha)Q^i_t(s^i_t, a^i_t) + \alpha \left[r^i_{t+1}(s^i_{t+1}) + \gamma \sum_{j \in J^i} \eta^{i,j} \max_{a^j \in A} Q^j_t(s^j_t, a^j) \right], \tag{10}$$

where J^i is a set of agent i's neighboring nodes, and $\eta^{i,j}$ is the weight between agents i and j. For instance, with $\eta^{i,j} = 1/|J^i|$ and $\sum_{j \in J^i} \eta^{i,j} = 1$, the Q-value of each neighboring agent $j \in J^i$ has an equal effect on an agent i's Q-value $Q^i_{t+1}(s^i_t, a^i_t)$. Third, the agents select their respective actions while ensuring that the global Q-value converges to a unique and optimal equilibrium, such as the Nash equilibrium [El-Tantawy et al. 2013], as time goes by. The global Q-value, which represents the global objective function, sums up the local Q-value of each agent [Mikami and Kakazu 1994]. At equilibrium, each agent's action is the best response to the others, and so an optimal joint action based on the global reward is achieved, rather than a locally optimal action based on the local reward in the traditional RL approach.

Prabuchandran's Multiagent Reinforcement Learning Approach. In Prabuchandran et al. [2015], the MARL approach optimizes the global Q-value (T.1.2.1) to address the challenge of an inappropriate traffic phase sequence (C.1) using a distributed model (T.1.2). An urban traffic network based on Bangalore (M.1.3) and traffic characterized by a real-world traffic model (M.2.2) are considered. The MARL model is embedded in the traffic signal controller of each intersection (G.1). Algorithm 2, which is based on Q-learning (H.1), shows the self-explanatory MARL algorithm embedded in each intersection i, which is based on the traditional RL algorithm (see Algorithm 1). The state s^i_t represents a two-tuple, namely, the queue size (S.1) and the red time (S.2) of the lanes. The action a^i_t represents the type of traffic phase to be activated in the next time instant (A.1). The delayed cost (or negative reward) $r^i_{t+1}(s^i_{t+1})$ represents the red time (R.2) and the queue size of the lanes (R.4), and it is calculated using Equation (11) as follows:

$$r^i_{t+1}(s^i_{t+1}) = \eta_1 \left(\frac{1}{|H^i|} \sum_{d^i \in D^i} t^{i,d^i}_{r,t+1} \right) + (1-\eta_1) \left(\frac{1}{|J^i|} \sum_{j \in J^i} \sum_{d^j \in D^j} \frac{n^{j,d^j}_{q,t+1}}{|D^j|} \right), \tag{11}$$

where $|H^i|$ represents the number of traffic phases at agent (or intersection) i, D^i represents a set of lanes at agent i, J^i represents a set of agent i's neighboring agents, and $0 \leq \eta_1 \leq 1$ represents a weight factor. Using Equation (11), the total red time $t^{i,d^i}_{r,t+1}$ of lane d^i at agent i at time $t+1$ and the queue size $n^{j,d^j}_{q,t+1}$ of lane d^j at neighboring agent j at time $t+1$ are taken into account. The Q-value $Q^i_t(s^i_t, a^i_t)$ is updated using the Q-function (see Equation (5)). The greedy action is the action with the maximum

ALGORITHM 2: Prabuchandran's MARL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
1: **procedure**			
2: Observe current state s_t^i			
3: Select action $a_t^{i,*}$ using Equation (3)			
4: Receive queue size $n_{q,t+1}^{j,d^j}$ from neighboring agent $j \in J^i$		$\leq \lvert J \rvert$	
5: Receive total red time $t_{r,t+1}^{i,d^i}$		1	
Calculate delayed reward $r_{t+1}^i(s_{t+1}^i)$ using Equation (11)	$\mathcal{O}(1)$		
6: Update Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ using Equation (5)	$\mathcal{O}\lvert A \rvert$		1
7: **end procedure**			

Q-value selected using Equation (3). Two exploration approaches are adopted, namely, the ε-greedy approach (E.1) and the UCB approach (E.3). The MARL approach has been shown to reduce average delay (P.1) and average waiting time (P.2). Compared to the ε-greedy approach, the UCB approach achieves better system performance due to faster exploration.

4.2. Max-Plus Reinforcement Learning

The max-plus RL approach, which is based on the MARL approach, enables an agent to learn about its neighboring agents' information, particularly locally optimized payoffs, rather than readily available information in RL, such as reward [Prabuchandran et al. 2015] and Q-value [Su and Tham 2007]. Subsequently, the agent calculates and maximizes the sum of the locally optimized payoffs from itself and neighboring agents in order to maximize the global payoff.

Medina and Benekohal's Max-Plus Reinforcement Learning Approach. In Medina and Benekohal [2012], the max-plus RL (or payoff propagation) approach optimizes the global payoff value (T.1.2.1) to address the challenge of an inappropriate traffic phase sequence (C.1) using a distributed model (T.1.2). An urban traffic network based on Springfield, IL (M.1.3), and traffic characterized by a real-world traffic model (M.2.2) are considered. The max-plus RL model is embedded in each intersection (G.1). Algorithm 3, which is based on Q-learning (H.1), shows that the max-plus RL algorithm is embedded in each intersection i. The state s_t^i represents three types of information: (1) the queue size (S.1) of the lanes at an agent and its neighboring agents, (2) the current traffic phase (S.3), and (3) the current traffic phase split (S.4). The action a_t^i represents the type of traffic phase to be activated in the next time instant (A.1). The delayed cost (or negative reward) $r_{t+1}^i(a_t^i)$ represents a reward ratio calculated based on three types of information: (1) the green signals activated when cross-blocking occurs a nd during normal operation, respectively (R.3); (2) the queue size of a lane created when red and green signals are activated, respectively (R.4); and (3) the delay (or time lost) caused by green-to-red signal transition (R.5). The reward ratio is a weighted reward value (or performance) achieved by an action to a weighted reward value achieved by other actions. Higher reward ratio for an action indicates better performance achieved by the action compared to other actions. Each agent is embedded with the traditional Q-learning approach (see Section 1.1) to learn about the local payoff value, which can be the Q-value $f_t^i(a_t^i) = Q_t^i(a_t^i)$. An agent i sends a payoff message $u_t^{ij}(a_t^j)$ to its neighboring agent $j \in J^i$, who is taking action a_t^j, at time t. The payoff message $u_t^{ij}(a_t^j)$ represents the maximum payoff received by agent i for taking its best possible action, which may not be the locally optimal action, while neighboring agent j is assumed to be taking

ALGORITHM 3: Medina and Benekohal's Max-Plus RL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
procedure			
Observe current state s_t^i			
Calculate global payoff value $g_t^i(a_t^i)$ using Equation (13)	$\mathcal{O}(1)$		
Select optimal action $a_t^{i,*}$ using Equation (14)			
Calculate delayed reward $r_{t+1}^i(s_{t+1}^i)$	$\mathcal{O}(1)$		
Update Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ using Equation (5)	$\mathcal{O}(\lvert A\rvert)$		1
Calculate payoff message $u_t^{ij}(a_t^j)$	$\mathcal{O}(1)$		
Send payoff message $u_t^{ij}(a_t^j)$ to each neighboring agent $j \in J^i$		$\leq \lvert J\rvert$	
end procedure			

an optimal action $a_t^{j,*}$ as part of the optimal joint action. An agent i exchanges payoff messages with its neighboring agents J^i until the global payoff converges to the optimal global payoff. Suppose agents i and j have a similar set of actions $A = A^i = A^j$. The payoff message $u_t^{ij}(a_t^j)$ sent by agent i to neighboring agent $j \in J^i$ is as follows:

$$u_t^{ij}(a_t^j) = \max_{a_j \in A} \left[f_t^i(a_t^i) + f_t^{ij}(a_t^i, a_t^j) + \sum_{k \in J^i \setminus j} u_t^{ki}(a_t^i) \right] + c_{ij} \qquad j \forall N_n^i, \tag{12}$$

where $J^i \setminus j$ represents a set of neighboring agents of agent i except neighboring agent j, $f_t^i(a_t^i)$ represents the local payoff value of agent i when agent i is taking action $a_t^i \in A^i$, $f_t^{ij}(a_t^i, a_t^j)$ represents the payoff value of neighboring agent j while agent i takes action a_t^i and agent j takes action a_t^j, $u_t^{ki}(a_t^i)$ represents payoff messages received by agent i from neighboring agents J^i except agent j, and c_{ij} is a normalization value that prevents the payoff message $u_t^{ij}(a_t^j)$ from becoming extremely large. The global payoff $g_t^i(a_t^i)$ of an agent i is as follows:

$$g_t^i(a_t^i) = f_t^i(a_t^i) + \sum_{j \in J^i} u_t^{ji}(a_t^i). \tag{13}$$

The greedy action has the maximum global payoff value as follows:

$$a_t^{i,*} = \text{argmax}_{a \in A} g_t^i(a). \tag{14}$$

The Medina and Benekohal's max-plus RL approach has been shown to increase throughput (P.4) and reduce the number of stops per vehicle (P.5).

4.3. Model-Based Reinforcement Learning

Traditionally, RL is a model-free approach that does not compute a transition probability, which characterizes the state transition of the operating environment. Nevertheless, models can be created to increase the learning speed (or the convergence rate). A common approach to compute a transition probability is using counters. As an example, the transition probability $P(s_t^i, a_t^i, s_{t+1}^i) = N_t(s_t^i, a_t^i, s_{t+1}^i)/N_t(s_t^i, a_t^i)$ represents the probability of the next state being s_{t+1}^i given the current state s_t^i and action a_t^i. The counters $N_t(s_t^i, a_t^i, s_{t+1}^i)$ and $N_t(s_t^i, a_t^i)$ count the number of occurrences of $(s_t^i, a_t^i, s_{t+1}^i)$ and (s_t^i, a_t^i) up to time t since the beginning 0,1,2, ... The rest of this section presents two types of model-based RL, which creates the model of the operating environment [Khamis and Gomaa 2012, 2014] and action selection [El-Tantawy et al. 2013].

Khamis's Model-Based Reinforcement Learning Approach. In Khamis et al. [2012a], the model-based RL approach optimizes the global system performance to address the challenges of inappropriate traffic phase sequence (C.1) and inappropriate traffic phase split (C.2) in a grid traffic network (M.1.4) using a centralized model (T.1.1). The traffic is characterized by a real-world traffic model (M.2.2). The model-based RL approach uses a Bayesian approach to estimate the model of the operating environment (or transition probability). The model-based RL model is embedded in each vehicle (G.2) to keep track of the total estimated waiting time of a vehicle. Subsequently, the traffic light controller at the intersection gathers information from the vehicles to select the traffic phase to be activated. Hence, while the RL model is embedded in each vehicle, the action of the RL model is given by the traffic light controller. By embedding the RL model in each vehicle, the effect of the curse of dimensionality is minimized as the traffic signal controller does not store and keep track of a large number of state-action pairs. Algorithm 4, which is based on Q-learning (H.1), shows the model-based RL algorithm embedded in the vehicular agent i with action selection made by the traffic signal controller m. The state s_t^i represents the current traffic phase (S.3) and two types of information about a vehicle: (1) the physical position of the vehicle while waiting at an intersection (S.5) and (2) the physical position of the destination of the vehicle (S.6). The action a_t^m represents the current type of traffic phase (A.1), whether red or green light, activated by a traffic signal controller $m \in M$. Various factors affect the reward value $r_{t+1}^i(s_{t+1}^i)$, and so a weight factor is used to represent their impact to the Q-values: flow rate, the safety factor, the average time of the trips, and the average waiting time of the trips and at the intersections (R.2). As an example of the average waiting time of a trip, a reward value of 0 is awarded to a vehicle (or an agent) when it moves, and a cost value of 1 is penalized against the vehicle when it stops and waits. The Q-value $Q_t^i(s_t^i, a_t^m)$ keeps track of the total estimated waiting time of a vehicle traversing from a source until it reaches its destination. The waiting time is estimated under the scenario of action a_t^m being taken by the traffic signal controller m at its intersection under state s_t^i in a traffic network. The Q-value $Q_t^i(s_t^i, a_t^m)$ is updated using the Q-function as follows:

$$Q_{t+1}^i(s_t^i, a_t^m) \leftarrow \sum_{s_{t+1}^i \in S} \left[P(s_t^i, a_t^m, s_{t+1}^i) \times \sum_{f \in F} \left(\eta_f^i \times r_{t+1}^{i,f}(s_{t+1}^i) \right) + \gamma V_t^i(s_{t+1}^i) \right], \tag{15}$$

where $f \in F$ represents a factor affecting a reward value, and η_f^i represents the weight factor of one of the factors $f \in F$ affecting the reward value $r_{t+1}^{i,f}(s_{t+1}^i)$. Higher weight factor η_f^i represents greater impact of the factor f to the Q-value $Q_{t+1}^i(s_t^i, a_t^m)$, and value function $V_t^i(s_{t+1}^i) = \sum_{a \in A^m}[P(a|s_t^i) \times Q_t^i(s_t^i, a)]$ represents the average waiting time of a vehicle under state s_t^i regardless of the action a_t^m taken by the traffic signal controller m. In this model-based approach, the posterior probability $P(s_t^i, a_t^m, s_{t+1}^i) = \{2/[N_t(s_t^i) \times (N_t(s_t^i)+1)]\} \times N_t(s_t^i, a_t^m, s_{t+1}^i)$, which is based on the Bayes rule, represents the model of the operating environment (or transition probability), where $N_t(\bullet)$ represents the number of times the component $\bullet$ has been visited up to time t since the beginning 0,1,2, ... The traffic light controller m at the intersection sums up the individual gains of each vehicle $i \in I$ at the intersection $\sum_i [Q_t^i(s_t^i, a_t^m = red) - Q_t^i(s_t^i, a_t^m = green)]$, which is the difference between the total estimated waiting time of the vehicles during red lights and during green lights, and selects the type of traffic phase that can minimize the waiting time of the vehicles as follows:

ALGORITHM 4: Khamis's Model-Based RL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
procedure			
/* At each vehicular agent i to keep track of waiting time */			
Observe current state s_t^i			
Observe current action a_t^m			
Update models $P(s_t^i, a_t^m, s_{t+1}^i)$ and $P(a\|s_t^i)$	$\mathcal{O}(\|S\|\|A\|)$		2
Receive delayed reward $r_{t+1}^i(s_{t+1}^i)$			
Update Q-value $Q_{t+1}^i(s_t^i, a_t^m)$ using Equation (15)	$\mathcal{O}(\|A\|)$		1
Send Q-value $Q_{t+1}^i(s_t^i, a_t^m)$ to signal controller m		1	
end procedure			
procedure			
/* At each traffic signal controller m to select a traffic phase */			
Receive Q-value $Q_{t+1}^i(s_t^i, a_t^m)$ from vehicular agent i at its intersection			
Select action $a_t^{m,*}$ using Equation (16)			
end procedure			

$$a_t^{m,*} = \text{argmax}_a \sum_i \left[Q_t^i\left(s_t^i, a = red\right) - Q_t^i\left(s_t^i, a = green\right) \right]. \tag{16}$$

The Khamis model-based RL approach [Khamis and Gomaa 2012, 2014] has been shown to increase throughput (P.4) and average speed (P.6), as well as to reduce average waiting time (P.2). A similar model-based RL approach has also been applied in Wiering et al. [2004].

El-Tantawy's Model-Based Multiagent Reinforcement Learning Approach. In El-Tantawy et al. [2013], the model-based MARL approach optimizes the global Q-value (T.1.2.1) to address the challenges of inappropriate traffic phase sequence (C.1) and inappropriate traffic phase split (C.2) using a distributed model (T.1.2). By creating a model of the policy (or a series of action selections) of neighboring agents, an agent considers the optimal actions taken by its neighboring agents, and so it is based on the MARL approach. An urban traffic network based on Toronto (M.1.3) and traffic characterized by a real-world traffic model (M.2.2) are considered. The model-based RL model is embedded in each intersection (G.1). Algorithm 5, which is based on Q-learning (H.1), shows the model-based algorithm embedded in each intersection i. The state represents a three-tuple information, namely, the maximum queue size among the lanes of each phase (S.1), the time elapsed for the current phase (S.4), and the index of the current traffic phase with green signals activated (S.3). The action represents the type of traffic phase to be activated in the next time instant (A.1). The reward represents the reduction of vehicular delay at an intersection (R.1). For each neighboring agent $j \in J^i$, an agent i uses counters to calculate the probability of a state-action pair $P(s_t^i, s_t^j, a_t^j) = N_t(s_t^i, s_t^j, a_t^j)/N_t(s_t^i, s_t^j)$, where higher probability $P(s_t^i, s_t^j, a_t^j)$ indicates the higher possibility of a neighboring agent j taking action a_t^j under a joint state $[s_t^i, s_t^j]$. The Q-value $Q_{t+1}^i(s_t^i, s_t^j, a_t^i, a_t^j)$ is updated using Q-function as follows:

$$\begin{aligned} Q_{t+1}^i\left(s_t^i, s_t^j, a_t^i, a_t^j\right) \leftarrow (1-\alpha) Q_t^i\left(s_t^i, s_t^j, a_t^i, a_t^j\right) + \alpha \Bigg[& r_{t+1}^i\left(s_{t+1}^i, s_{t+1}^j\right) \\ & + \max_{a^i \in A^i} \sum_{a^j \in A^j} P\left(s_t^i, s_t^j, a^j\right) \times Q_t^i\left(s_t^i, s_t^j, a^i, a^j\right) \Bigg]. \end{aligned} \tag{17}$$

ALGORITHM 5: El-Tantawy's Model-Based RL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
1: **procedure**			
2: Observe current state s_t^i			
3: Observe current state s_t^j and action a_t^j from neighboring agents $j \in J^i$			
4: Update model $P(s_t^i, s_t^j, a_t^j)$	$\mathcal{O}(\|S\|\|A\|)$		1
5: Select action $a_t^{i,*}$ using Equation (18)			
6: Receive delayed reward $r_{t+1}^i(s_{t+1}^i, s_{t+1}^j)$			
7: Update Q-value $Q_{t+1}^i(s_t^i, s_t^j, a_t^i, a_t^j)$ using Equation (17)	$\mathcal{O}(\|A\|)$		1
8: **end procedure**			

An agent i selects an action with the maximum Q-value as follows:

$$a_t^{i,*} = \underset{a^i \in A^i}{\operatorname{argmax}} \sum_{j \in J^i} \sum_{a^j \in A^j} Q_{t+1}^i\left(s_t^i, s_t^j, a^i, a^j\right) \times P\left(s_t^i, s_t^j, a^j\right). \tag{18}$$

The El-Tantawy's model-based MARL approach [El-Tantawy et al. 2013] has been shown to increase throughput (P.4), as well as to reduce average delay (P.1), average waiting time (P.2), and queue size (P.3).

4.4. Actor-Critic Reinforcement Learning

Traditionally, RL updates a Q-value using a delayed reward. The actor-critic RL approach corrects the delayed reward using temporal difference, and it is subsequently used to update the Q-value in order to expedite the learning process. To achieve this, in the actor-critic RL approach, the update of the Q-value and the calculation of temporal difference procedures are separated, and they are delegated to two different entities called actor and critic. The critic calculates the temporal difference of a state-action pair to correct the delayed reward of a state-action pair, while the actor uses the temporal difference, instead of the delayed reward, to update the Q-value of the state-action pair.

Li's Actor-Critic Reinforcement Learning Approach. In Li et al. [2009], the actor-critic RL approach optimizes the local Q-value (T.1.2.2) to address the challenge of inappropriate traffic phase split (C.2) in a single-intersection traffic network (M.1.1) using a centralized model (T.1.1). The traffic is characterized by Poisson process (M.2.1). The actor-critic RL model is embedded in each intersection (G.1). Algorithm 6, which is based on Q-learning (H.1), shows the actor-critic RL algorithm embedded in each intersection i. The state s_t^i represents the queue size (S.1) and the current traffic phase (S.3). The action a_t^i represents the choice to either keep the current traffic phase or switch to the next traffic phase in a predetermined sequence of traffic phases at the next time instant (A.2). The reward $r_{t+1}^i(s_{t+1}^i)$ represents the average waiting time (R.2). There are two main steps. First, the critic calculates a temporal difference, which is based on the value function (see Equation (2)), used to criticize the actions made by the actor. The temporal difference $\delta_t^i(s_t^i, a_t^i) = r_{t+1}^i(s_{t+1}^i) + V_t^i(s_{t+1}^i) - V_t^i(s_t^i)$ represents the difference between two successive estimations of the Q-value for a state-action pair. The temporal difference becomes $\delta_t^i(s_t^i, a_t^i) \rightarrow 0$, and so $Q_{t+1}^i(s_t^i, a_t^i) \approx Q_t^i$ (see Equation (1)) upon convergence. It is used to evaluate the improvement of the Q-value toward convergence by strengthening or weakening the possibility of selecting an action under a state. Second, the actor uses the temporal difference $\delta_t^i(s_t^i, a_t^i)$ to update the Q-value for a state-action pair using Equation (1). The optimal action $a_t^{i,*}$ is selected

ALGORITHM 6: Li's Actor-Critic RL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
procedure			
/* Critic */			
Observe current state s_t^i			
Select action $a_t^{i,*}$ using Equation (3)			
Receive delayed reward $r_{t+1}^i(s_{t+1}^i)$			
Calculate temporal difference $\delta_t^i(s_t^i, a_t^i)$	$\mathcal{O}(1)$		
/* Actor */			
Update Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ using Equation (1)	$\mathcal{O}(\|A\|)$		1
end procedure			

using the Boltzmann approach (E.2). A self-programmed simulator is developed using MATLAB to investigate Li's actor-critic RL approach [Li et al. 2009], and it has been shown to reduce the average vehicular delay (R.1).

4.5. Multistep Backup Reinforcement Learning

Traditionally, RL selects an optimal action $a_t^{i,*}$ based on the current state s_t^i; however, an action a_t^i may affect a consecutive series of states $s_t^i, s_{t+1}^i, s_{t+2}^i, \ldots$. Multistep backup RL allows an agent to look backward so that a series of states in an episode, which is a sequence of time instants, is taken into account, and uses eligibility traces to achieve the average effects of the temporal difference (see Equation (6)). The eligibility trace initiates a short-term memory (or trace) of a state each time it is visited; otherwise, it decays gradually over time.

Jin and Ma's Multistep Backup Reinforcement Learning Approach. In Jin and Ma [2015b], the multistep backup RL approach optimizes the local Q-value (T.1.2.2) to address the challenges of inappropriate traffic phase sequence (C.1) and inappropriate traffic phase split (C.2) in a single-intersection traffic network (M.1.1) using traffic phases with grouped individual traffic (T.2.3). The traffic is characterized by a real-world traffic model (M.2.2). The multistep backup RL model is embedded in each or a combination of nonconflicting traffic movements (G.3). Algorithm 7, which is based on SARSA (H.2), shows the multistep backup RL algorithm embedded in each or a combination of nonconflicting traffic movements i. An episode starts with the activation of green signals for a combination of traffic movements, and it ends with the termination of the green signals. The state s_t^i represents several bits of information, including the index of the current traffic phase with green signals activated (S.3), the time elapsed and the maximum time of the current traffic phase (S.4), the occupancy (or whether at least a single vehicle is waiting at an intersection) of intersection i and neighboring intersections J (S.5), and the time gap between two vehicles at intersection i and neighboring intersections J. The action a_t^i represents the choice to either keep the current traffic phase (A.2) or to switch to another traffic phase (A.1) at the next time instant. Note that, unlike most schemes in the literature, the Jin and Ma multistep backup RL approach takes both traffic phase and traffic phase split into account (see Table V). The traffic phase split increases in multiples of the minimum duration, specifically $a_t^i = \{0, 1, 2, 3, 4\}$ seconds. The reward represents the increment and reduction of vehicular delay at an intersection (R.1). The Q-value $Q_T^i(s_t^i, a_t^i)$ of an episode $T = 1, 2, \ldots, t_T, \ldots, |T|$ is updated using the Q-function as follows, with the temporal difference $\delta_t^i(s_t^i, a_t^i)$, $\delta_{t+1}^i(s_{t+1}^i, a_{t+1}^i), \ldots, \delta_{t+|T|}^i(s_{t+|T|}^i, a_{t+|T|}^i)$ (see Equation (6))

ALGORITHM 7: Jin and Ma Multistep Backup RL Algorithm	Computational Complexity	Message Complexity	Storage Complexity
procedure			
Observe current state s_t^i			
Select action $a_t^{i,*}$ using Equation (3)			
for $(t_T = 1; t_T \leq \lvert T \rvert; t_T{+}{+})$			
Receive delayed reward $r_{t+t_T}^i(s_{t+t_T}^i)$			
Calculate temporal difference $\delta_{t+t_T}^i(s_{t+t_T}^i, a_{t+t_T}^i)$ using Equation (6)	$\mathcal{O}(\lvert T \rvert)$		
Calculate eligibility traces $e_{t+t_T}^i(s_{t+t_T}^i, a_{t+t_T}^i)$ using Equation (20)	$\mathcal{O}(\lvert T \rvert)$		
end for			
Update Q-value $Q_T^i(s_t^i, a_t^i)$ using Equation (19)	$\mathcal{O}(\lvert A \rvert)$		1
end procedure			

being weighted by the eligibility traces $e_t^i(s_t^i, a_t^i), e_{t+1}^i(s_{t+1}^i, a_{t+1}^i), \ldots, e_{t+|T|}^i(s_{t+|T|}^i, a_{t+|T|}^i)$:

$$\begin{aligned} Q_T^i(s_t^i, a_t^i) \leftarrow Q_t^i(s_t^i, a_t^i) &+ \alpha\delta_t^i(s_t^i, a_t^i)e_t^i(s_t^i, a_t^i) + \alpha\delta_{t+1}^i(s_{t+1}^i, a_{t+1}^i)e_{t+1}^i(s_{t+1}^i, a_{t+1}^i) \\ &+ \cdots + \alpha\delta_{t+|T|}^i(s_{t+|T|}^i, a_{t+|T|}^i)e_{t+|T|}^i(s_{t+|T|}^i, a_{t+|T|}^i). \end{aligned} \tag{19}$$

The eligibility trace $e_t^i(s^i, a^i)$ adds more credits to recent state-action pairs (s_t^i, a_t^i) by setting its value to one each time a state-action pair is visited, and it is updated at time t as follows:

$$e_t^i(s^i, a^i) = \begin{cases} 1, & \text{if } s^i = s_t^i \text{ and } a^i = a_t^i. \\ 0, & \text{if } s^i = s_t^i \text{ and } a^i \neq a_t^i. \\ \gamma\varphi e_{t-t_T}^i(s^i, a^i), & \text{if } s^i \neq s_t^i, \end{cases} \tag{20}$$

where $0 \leq \varphi \leq 1$ represents the trace decay that exponentially decays the eligibility trace of a state-action pair that is not visited. The Jin and Ma multistep backup RL approach [Jin and Ma 2015b] has been shown to reduce average delay (P.1).

4.6. Reinforcement Learning with Function Approximation

Traditionally, RL is concerned with the issue of the curse of dimensionality (see Section 1.1) as the number of state-action pairs can be highly dimensional and large in number. Function approximation stores and keeps track of a significantly smaller number of features, instead of a large number of state-action pairs. As an example, in Prashanth and Bhatnagar [2011, 2012], a function approximation approach keeps track of 200 features, instead of $|S| \times |A| = 10^{101}$ state-action pairs. This helps to improve scalability with reduced requirements on memory or storage capacity, as well as reduced learning time.

Prashanth and Bhatnagar's Reinforcement Learning Approach with Function Approximation. In Prashanth and Bhatnagar [2011, 2012], the RL with function approximation approach optimizes the global system performance to address the challenge of inappropriate traffic phase sequence (C.1) in a two-intersection (M.1.2) real-world network based on Bangalore (M.1.3), 2×2 and 3×3 grids (M.1.4), and an arterial with 16 minor streets traffic network (M.1.5) using a centralized model (T.1.1). The traffic is characterized by Poisson process (M.2.1). The RL model with function approximation is embedded in a central controller, which can be one of the traffic signal controllers (G.1). Algorithm 8, which is based on Q-learning (H.1), shows the RL with function approximation algorithm embedded in a central controller i. The state $\boldsymbol{s_t^i} = [s_t^{i,1}, s_t^{i,2}] \in S^i$ represents a two-tuple information, namely, the queue size $s_t^{i,1}$ (S.1) and the red time $s_t^{i,2}$ (S.2) of the lanes. The action a_t^i

ALGORITHM 8: Prashanth and Bhatnagar's RL with Function Approximation	Computational Complexity	Message Complexity	Storage Complexity
1: **procedure**			
2: Observe current state s_t^i			
3: Select action $a_t^{i,*}$ using Equation (23)			
4: Receive delayed reward $r_{t+1}^i(s_{t+1}^i)$			
5: Calculate temporal difference $\delta_t^i(s_t^i, a_t^i)$ using Equation (21)	$\mathcal{O}(1)$		
Update θ_{t+1}^i using Equation (22)	$\mathcal{O}(1)$		
Update Q-value $Q_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i) \approx \theta_t^{i^T} \times \sigma_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i))$	$\mathcal{O}(n_{f,a_t^i}^i)$		1
6: **end procedure**			

represents the type of traffic phase to be activated in the next time instant (A.1). The delayed cost (or negative reward) $r_{t+1}^i(\boldsymbol{s_{t+1}^i})$ represents the red time (R.2) and the queue size of the lanes (R.4). In this RL model, each feature is a value that represents a state-action pair. As an example, a feature value of 0.5 represents states $\Psi_q^1 < s_t^{i,1} < \Psi_q^2$ and $s_t^{i,2} > \Psi_r^1$ as well as an action $a_t^i = green$. The Q-value $Q_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i) \approx \theta_t^{i^T} \times \sigma_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i)$ is approximated by updating a d-dimensional tunable weight vector $\theta_t^{i^T} = (\theta_t^{i,1}, \ldots, \theta_t^{i,d})$, which is a transpose of the matrix θ_t^i, with respect to a d-dimensional feature vector $\sigma_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i) = (\sigma_t^{i,1}(s_t^{i,1}, s_t^{i,2}, a_t^i), \ldots, \sigma_t^{i,d}(s_t^{i,1}, s_t^{i,2}, a_t^i))$ that corresponds to a state-action pair. Hence, the Q-value $\theta_t^{i^T} \times \sigma_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i)$ has a d dimension, with the number of features relevant to a particular state $\boldsymbol{s_t^i}$ and action a_t^i being $n_{f,\boldsymbol{s_t^i}}^i$ and $n_{f,a_t^i}^i$, respectively. The temporal difference is the difference between two successive estimations as follows:

$$\delta_{t+1}^i(s_t^i, a_t^i) = r_{t+1}^i(s_{t+1}^i) + \gamma \max_{a \in A^i} \theta_t^{i^T} \sigma_t^i(s_{t+1}^i, a) - \theta_t^{i^T} \sigma_t^i(s_t^i, a_t^i). \tag{21}$$

An agent i updates the tunable weight vector θ_t^i that represents the impact of each feature as follows:

$$\theta_{t+1}^i = \theta_t^i + \left[\alpha \times \sigma_t^i(s_t^{i,1}, s_t^{i,2}, a_t^i) \times \delta_{t+1}^i(s_t^i, a_t^i)\right]. \tag{22}$$

An agent i selects an action with the maximum approximated Q-value as follows:

$$a_t^{i,*} = \text{argmax}_{a \in A}\left[\theta_t^{i^T} \times \sigma_t^i(s_t^i, a)\right]. \tag{23}$$

Prashanth and Bhatnagar's RL with function approximation approach [Prashanth and Bhatnagar 2011, 2012] has been shown to increase throughput (P.4) and to reduce waiting time (P.2). A similar function approximation approach has also been applied in Chu and Wang [2015] and Yin et al. [2016].

5. FURTHER ENHANCEMENT OF REINFORCEMENT LEARNING FOR TRAFFIC SIGNAL CONTROL

This section presents two further enhancements to RL applied to traffic signal control.

5.1. Quantization of State and Action Spaces

While additional infrastructure, such as grid computing, can be used to share computing resources to solve complex problems, as well as storing resources to store Q-values [Su and Tham 2007], quantization can be applied to reduce the cardinality of the state-action pairs.

The state space can be segregated to form clusters of states. As an example, the queue size of each lane can be segregated into *low*, *medium*, and *high* levels with the maximum congestion level being 60 vehicles [Chin et al. 2012]. The thresholds for the different levels can be determined using a fuzzy function [Wen et al. 2010]. In addition, a maximum value can be used. As an example, the maximum vehicle occupancy time of the lanes represents the vehicle occupancy time of a leg [Balaji et al. 2010].

A similar approach has also been applied to represent the action space. As an example, the traffic phase split of a lane for each traffic phase can be segregated into different levels ($low = 10s$, $medium = 20s$, and $high = 30s$) [Araghi et al. 2013; Prabuchandran et al. 2014]. The thresholds for the different levels can be dynamically adjusted using an online threshold-tuning algorithm [Prashanth and Bhatnagar 2012]. As another example, the traffic phase split can be extended through integer times of a time slot [Chin et al. 2012].

5.2. Integration with Other Artificial Intelligence Approaches

Artificial intelligence approaches have been applied to store Q-values in an efficient manner in order to address the curse of dimensionality, as well as to achieve optimal or near-optimal actions in order to maximize the global reward. There are three main considerations in the choice of an artificial intelligence approach. First, it provides incremental updates and makes decisions when necessary without the need to wait for a large batch of data points to be processed. Second, it does not forget about its past knowledge. Third, it provides a confidence estimate on the predictions made.

Feed-forward neural network has been applied to store Q-values [Arel et al. 2010; Choy et al. 2003; Dai et al. 2010]. There are three main layers, namely, input, hidden, and output layers. The *input* layer consists of a set of input nodes to represent a state vector with multiple substates (S.1) to (S.6). As an example, a state vector $\boldsymbol{s_t^i} = [s_t^{i,1}, s_t^{i,2}]$ is represented by two input nodes $s_t^{i,1}$ and $s_t^{i,2}$. Each input node is connected to a number of subnodes to provide a combination of states. As an another example, in Arel et al. [2010], each input node is connected to eight subnodes to effectively provide 40 input nodes. The *hidden* layer consists of neurons. Each neuron is connected to nodes in the output layer, and each connection has a weight value to represent its strength. The neuron uses a log-sigmoid transfer function to update the weight value, which serves as the Q-value, using the gradient descent approach. The weight value can be increased (reduced) by a positive (negative) reward, and so the action with higher weight value is selected with greater possibility. The *output* layer consists of output nodes to represent the selected actions. As the final example, in Arel et al. [2010] and Dai et al. [2010], the output layer has a single output node that represents the next traffic phase to be activated (A.1).

Genetic algorithm has been applied to find optimal or near-optimal actions for RL [Mikami and Kakazu 1994]. There are two main representations: (1) a chromosome represents a state-action pair, and (2) the fitness value of a chromosome represents the optimality of the state-action pair. As an example, a chromosome represents a traffic phase and its traffic phase split, and the fitness value represents the throughput achieved by the chromosome. The genetic algorithm performs two main tasks on a pair of selected chromosomes. First, crossover swaps the components of a chromosome if it fulfills a crossover ratio to generate a new chromosome. Second, mutation alters the components that have undergone crossover with random values if they fulfill the crossover ratio. The chromosome with a higher fitness value is chosen with higher possibility. Due to the large search space, searching the optimal or near-optimal solutions requires 1 week to 1 month time of field testing [Mikami and Kakazu 1994].

6. PERFORMANCE MEASURES AND COMPLEXITY ANALYSIS OF RL APPROACHES APPLIED TO TRAFFIC SIGNAL CONTROL SYSTEMS

This section presents performance measures and the simulation platforms used in the investigations and conducts complexity analysis for various RL models and algorithms applied to traffic signal control.

6.1. Performance Measures

Table VI summarizes the performance measures achieved by various extended RL approaches compared with the traditional RL approach, as well as the pretimed and actuated control systems (see Section 1). Most investigations have been performed with respect to *traffic arrival rate* (see Section 2.2.2), which can be increased in simulation to portray future scenarios. The performance measures achieved by the extended RL models and algorithms are as follows:

P.1 *Lower average delay* reduces the average time incurred by vehicles to cross an intersection or to traverse from a source to a destination, including the average waiting and traveling times during congestion and cross-blocking. The average delay can be an absolute value (i.e., the ratio of the delay of all vehicles to the number of vehicles [Yin et al. 2016]) or a relative value (i.e., the ratio of the time incurred by vehicles in a scenario with traffic to the equivalent scenario without traffic [Prabuchandran et al. 2015]).
P.2 *Lower average waiting time* reduces the waiting time of vehicles (see (R.2) in Section 3.3).
P.3 *Smaller queue size* reduces queue size (see (S.1) in Section 3.1).
P.4 *Higher throughput* increases the number of vehicles crossing an intersection, or reaching their destinations, within a time interval (e.g., a single cycle).
P.5 *Lower number of stops per vehicle* reduces the number of stops of each vehicle while crossing an intersection or traversing from a source to a destination [Heinen et al. 2011].
P.6 *Higher average speed* increases the speed of vehicles (i.e., the ratio of the total distance traversed by all vehicles to their time spent in a traffic network).

6.2. Simulation Platforms

Table VI also summarizes the simulation platforms applied to investigate the RL-based traffic signal control systems. There are two main types of simulation platforms applied to investigate RL-based traffic signal control. First, *self-programmed discrete event simulators* are developed by the authors of the literature themselves using programming languages such as C/C++ and tools such as MATLAB [Li et al. 2009; Su and Tham 2007]. A time step is equivalent to a single time unit (e.g., 2 seconds [Medina and Benekohal 2012]) needed to process a single vehicle (e.g., crossing an intersection), and a traffic signal controller selects an action every 20 time steps [Arel et al. 2010]. Second, *traffic simulators* provide a graphical user interface and essential features for roads and vehicles, as well as gather local and global statistics (e.g., the waiting time of the vehicles at an intersection and in a traffic network), respectively. Two approaches are (1) the *macroscopic* approach, which considers vehicles in a collective and homogeneous manner in the form of traffic flows observed over a longer period of time (e.g., hourly and daily), such as the speed limit of a lane, and (2) the *microscopic* approach, which considers the individual attributes of the vehicles observed at real time, such as the driving speed and direction. Most traffic simulators adopt the microscopic approach, including green light district (L.1), Paramics (L.2), VISSIM (L.3), SUMO (L.4), TSIS, and ITSUMO [Bazzan et al. 2010].

Table VI. Summary of Traffic Simulators and Performance Measures

RL Models	Traffic Simulators			Performance Measures						Complexities		
	(L.1) Green light district	(L.2) Paramics	(L.3) VISSIM	(L.4) SUMO	(P.1) Lower average delay	(P.2) Lower average waiting time	(P.3) Smaller queue size	(P.4) Higher throughput	(P.5) Lower number of stops per vehicle	(P.6) Higher average speed	Computational	Message	Storage
MARL [Prabuchandran et al. 2015]		*	*		*	*	*				$\mathcal{O}(\lvert I\rvert\lvert S\rvert\lvert A\rvert)$	$\leq \lvert I\rvert\lvert J\rvert$	$\leq \lvert I\rvert\lvert S\rvert\lvert A\rvert$
Max-plus RL [Medina and Benekohal 2012]			*					*	*		$\mathcal{O}(\lvert I\rvert\lvert S\rvert\lvert A\rvert)$	$\leq \lvert I\rvert\lvert J\rvert$	$\leq \lvert I\rvert\lvert S\rvert\lvert A\rvert$
Model-based RL [Khamis et al. 2012a]	*					*		*		*	$\mathcal{O}(\lvert I\rvert\lvert S\rvert^2\lvert A\rvert)$	$\leq \lvert I\rvert$	$\leq 2\lvert I\rvert\lvert S\rvert\lvert A\rvert + \lvert I\rvert\lvert S\rvert^2\lvert A\rvert$
Model-based RL [El-Tantawy et al. 2013]		*			*	*	*	*			$\mathcal{O}(\lvert I\rvert\lvert S\rvert^2\lvert A\rvert)$		$\leq \lvert I\rvert\lvert S\rvert^2\lvert A\rvert + \lvert I\rvert\lvert S\rvert^2\lvert A\rvert^2$
Actor-critic RL [Li et al. 2009]				*							$\mathcal{O}(\lvert I\rvert\lvert S\rvert\lvert A\rvert)$		$\leq \lvert I\rvert\lvert S\rvert\lvert A\rvert$
Multistep backup RL [Jin and Ma 2015b]				*	*						$\mathcal{O}(\lvert I\rvert\lvert S\rvert\lvert A\rvert)$		$\leq \lvert I\rvert\lvert S\rvert\lvert A\rvert$
RL with function approximation [Prashanth and Bhatnagar 2012]	*					*		*			$\mathcal{O}(\lvert I\rvert\lvert S\rvert\lvert A\rvert)$		$\leq \lvert I\rvert\lvert S\rvert^2\lvert A\rvert$

6.3. Complexity Analysis

This section investigates the computational, message, and storage complexities of RL algorithms. The complexity analysis, which is inspired by similar investigations in Bettstetter and Konig [2002] and Heinen et al. [2011], has three levels: *step-wise* considers a single iteration or execution of the RL algorithm, *agent-wise* considers all the state-action pairs of an agent, and *network-wide* considers all agents in the network. The step-wise complexities of the RL algorithms are shown in Algorithms 1 to 8, while the network-wide complexities are shown in Table VI. Note that, in the analysis, we mainly focus on the exploitation of actions in the RL algorithms. For brevity, since the agent-wise complexities have been intuitively used to derive the network-wide complexities, they are not shown in Table VI.

Computational complexity is the maximum number of times an RL algorithm is being executed in order to update the Q-values for all actions of the agents. For the step-wise computational complexity, this is considered under a particular state. As an example, in the traditional RL algorithm (see Algorithm 1), an agent i updates its Q-value upon receiving a delayed reward, so the step-wise computational complexity is given by $\mathcal{O}(|A|)$ (Step 5) since there are $|A|$ actions for each state. The agent-wise complexity is calculated as $\mathcal{O}(|S||A|)$ as there are $|S|$ states. Hence, the network-wide complexity is $\mathcal{O}(|I||S||A|)$ as there are $|I|$ agents in the network. As another example, in Prabuchandran's MARL algorithm (see Algorithm 2), an agent i calculates the delayed reward (Step 5) and updates its Q-value (Step 6), so the step-wise computational complexity is given by $\mathcal{O}(1) + \mathcal{O}(|A|)$, which can be simplified as $\mathcal{O}(|A|)$. The agent-wise complexity is calculated as $(\mathcal{O}(|S|) + \mathcal{O}(|S||A|))$, which can be simplified as $\mathcal{O}(|S||A|)$. Hence, the network-wide complexity is $(\mathcal{O}(|I|(|S|)) + \mathcal{O}(|I|(|S||A|)))$, which can be simplified as $\mathcal{O}(|I||S||A|)$.

Message complexity is the number of messages exchanged among the agents in order to update a Q-value. As an example, in Prabuchandran's MARL algorithm (see Algorithm 2), each agent i exchanges its traffic network condition (i.e., queue size) with its neighboring agents J (Step 4), so the step-wise message complexity is given by $\leq |J|$ since there are $|J|$ neighboring agents. The agent-wise complexity is calculated as $\leq |J|$. Hence, the network-wide complexity is calculated as $\leq |I||J|$. As another example, in Khamis's model-based RL algorithm (see Algorithm 4), each agent i sends a message consisting of its Q-value to the traffic signal controller m (Step 8), so the step-wise message complexity is given by 1. The agent-wise complexity is calculated as 1. Hence, the network-wide complexity is calculated as $\leq |I|$.

Storage complexity is the amount of memory required to store local statistics and Q-values. As an example, in the traditional RL algorithm (see Algorithm 1), an agent i stores the Q-value of a state-action pair (Step 5), so the step-wise storage complexity has a value of 1. The agent-wise complexity is calculated as $\leq |S||A|$ since an agent maintains $\leq |S||A|$ state-action pairs in a Q-table. Hence, the network-wide complexity is calculated as $\leq |I||S||A|$ since there are $|I|$ agents in the network. As another example, in Khamis's model-based RL algorithm (see Algorithm 4), an agent i stores the probabilities of two models (Step 5) and the Q-value of a state-action pair (Step 7), so the step-wise storage complexity has a value of 3. The agent-wise complexity is calculated as $\leq 2|S||A| + |S|^2|A|$ since an agent maintains $\leq |S|^2|A|$ storage for the $P(s_t^i, a_t^m, s_{t+1}^i)$ model, $\leq |S||A|$ storage for the $P(a|s_t^i)$ model, and $\leq |S||A|$ state-action pairs in a Q-table. Hence, the network-wide complexity is calculated as $\leq 2|I||S||A| + |I||S|^2|A|$.

7. OPEN ISSUES

RL is a promising technique to enhance system performance in traffic signal control systems, though not without shortcomings. There are three aspects of shortcomings.

First, the shortcomings of RL include poorer system performance caused by inappropriate parameters and during the initial learning phase, as well as the high amount of payoff message overhead (see Section 7.1). Second, the full potential and limitations of RL cannot be realized in the context of traffic signal control systems without incorporating recent advances in intelligent transportation systems (see Section 7.2) and a wider range of dynamic factors (i.e., environment, traffic, road users, and vehicles) (see Section 7.3) into the system. Third, existing traffic simulators applied to investigate RL are not suitable for investigation involving a combination of both macroscopic and microscopic models, as well as time-continuous models (see Section 7.4). This section discusses the open issues that can be pursued to address these shortcomings.

7.1. Addressing the Shortcomings of RL Models and Algorithms

The RL models and algorithms can be further enhanced in the following aspects:

First, various investigations into the application of RL to traffic signal control have shown poorer system performance achieved during the initial learning phase [Prabuchandran et al. 2015; Su and Tham 2007]. The use of prior knowledge (e.g., learned knowledge and historical traffic data), estimations, and traditional pretimed and actuated control systems for initial learning can be investigated. As an example, in Dusparic et al. [2016], traffic observers are proposed to count the number of vehicles in order to determine the congestion level, specifically the number of vehicles with respect to road capacity, which is based on the road layout and the number of lanes. The traffic signal controllers gather information from the traffic observers and adjust the traffic phase split accordingly. Future investigations could be pursued such that the agents use the local information provided by traffic observers to increase learning rate and initialize the knowledge (i.e., the Q-values in the Q-tables) during the initial learning or relearning phase, and subsequently achieve system-wide performance enhancement.

Second, fixed and nonadaptive values have been widely used for learning rate, discount factor, and exploration probability, such as $\varepsilon = 0.02$ and $\gamma = 0.95$ in Arel et al. [2010]. The nonadaptive approach sets these parameters with higher values at the beginning, and gradually reduces them by a decay factor as time goes by. As an example, the learning rate is $\alpha_t = 0.1$ when $t < 10^5$ seconds, and it reduces to $\alpha_t = 10^4/t$ when $t > 10^5$ seconds [El-Tantawy and Abdulhai 2012]. As another example, the learning rate is reduced with the number of visits to a particular state-action pair as time goes by; specifically, $\alpha_t = 1/(1 + N_t(s_t^i, a_t^i))$, where $N_t(s_t^i, a_t^i)$ represents the number of times a state-action pair (s_t^i, a_t^i) has been visited up to time t since the beginning [El-Tantawy and Abdulhai 2012]. Future investigations could be pursued to perform self-configuration of these parameters so that they can adapt to the dynamicity of the traffic conditions, such as the use of an adaptive decay factor. In addition, a minimum value, particularly the learning rate and exploration probability, can be imposed to ensure that learning and exploration are performed as time goes by due to the dynamicity of the traffic networks.

Third, the exchange of information for learning purposes, such as the payoff messages in the max-plus RL approach (see Section 4.2), can increase control overhead and computational cost among neighboring agents significantly. Future investigations could be pursued to identify unnecessary control message exchanges in order to enhance interoperability among the agents. The exchanged control messages should help to improve system performance of RL such as higher learning speed; otherwise, they should not be exchanged.

Fourth, various artificial intelligent approaches have been applied to enhance RL models and algorithms in order to provide nonlinear function approximation (see Section 5.2), and their parameters are generally configured with fixed values. While the

feed-forward neural network has been shown to improve system performance [Arel et al. 2010; Dai et al. 2010], incorrect configuration, such as the number of input nodes, neurons, and output nodes, as well as the number of hidden layers and the initial weights, can cause divergence from the optimal action [Baird 1995]. Future investigations could be pursued to determine the right choice of configuration that adapts to the dynamicity of the traffic conditions through self-configuration.

In addition, future investigations could be pursued to design hybrid methods of different RL models and algorithms, and address the open issues raised therein. As an example, a joint MARL and multistep RL approach can tap the strengths of both approaches in order to address the curse of dimensionality and take into account longer-term effects of an action under a state. Other artificial intelligence approaches [Zhao et al. 2012], such as fuzzy system, swarm intelligence, and game theory, can also be applied along with RL. As examples, the fuzzy system can be incorporated to include a priori knowledge into RL, and game theory can be incorporated to find the Nash equilibrium in order to achieve optimal actions.

7.2. Extension of RL Models and Algorithms to Incorporate Recent Advancements

The experimental traffic models (M.2.2) can be extended with recent advancements, such as intelligent transportation systems, to provide more comprehensive reflection of future scenarios.

The traffic signal controller or agent can interact with intelligent components and devices (e.g., wireless systems and sensors installed in infrastructure along roadsides, as well as onboard digital maps, GPS, and alert systems installed in smart vehicles) and existing infrastructure (e.g., ramp signal controllers that monitor traffic entering a freeway [Fares and Gomaa 2014], buildings and car parks located near an intersection). The agent can gather sensing outcomes (e.g., physical position and driving speed of a vehicle), remove noise, and process and exchange sensing outcomes with neighboring agents. Other important functions can be incorporated into traffic signal controllers. As an example, green-light wave activates traffic phases of traffic signal controllers according to expected arrival of vehicles [Cools et al. 2013; Gershenson and Rosenbluet 2009]. The main challenge is that it can only be activated on a single traffic phase (or two directions), resulting in poor performance in the rest of the traffic phases. Preliminary study has shown that green-light wave reduces average delay (P.1) and the number of stops per vehicle (P.5) [Khamis et al. 2012a, 2012b]. The agents can collect sensing outcomes and perform green-light wave based on the sensing outcomes in order to open and close lanes prior to the arrival of emergency and high-priority vehicles, contributing to a reduced number of stops of such vehicles (P.5). As another example, in extreme cases in which traffic congestion may lead to a deadlock due to heavy traffic arrivals, a traffic signal controller must prevent and break a deadlock.

7.3. Extension of RL Models and Algorithms to Address Unaccustomed Traffic

A wide range of dynamic factors (e.g., environment, traffic, road users, and vehicles) affecting the traffic conditions are yet to be investigated in the context of RL. The experimental models can be extended to accommodate these factors in order to provide more accurate and comprehensive reflection of the real-world scenario, including models tailor-made for local traffic conditions.

The *environment* can be affected by weather conditions (e.g., rainfall, snowfall, flood, and fog), ongoing construction, and residential and school areas. The *traffic* can be affected by different arrival rates of traffic flowing into an intersection from different incoming legs and traffic flow disturbances (e.g., unexpected on-street parking, accidents, special events, emergency or high-priority vehicles such as police cars and ambulance). The *road users* and *vehicles* can be affected by the types of vehicles (e.g.,

bikes and motorbikes); driving behaviors, which can be characterized by polite, selfish, and aggressive (e.g., driving speed, lane switching, time lost caused by driving behaviors, acceleration and deceleration), and underlying traffic networks (e.g., U-turning). Since the sequence of the traffic phases can be out of order, and a traffic phase can even be skipped due to the unpredictable and time-varying traffic conditions [El-Tantawy and Abdulhai 2010; El-Tantawy et al. 2013; Salkam et al. 2008; Yin et al. 2016], it is necessary for the road users to minimize confusion and to adapt to the out-of-order traffic phases. In addition to the widely used Poisson process (M.2.1), other processes, such as Erlang, Gaussian, and Weibull, can be adopted; most importantly, novel traffic models (M.2.2) that take into account the subtle and abrupt changes of the environment, traffic, road users, and vehicles can be designed and adopted.

Future investigations could be pursued to address two main challenges brought about by these dynamic factors and novel traffic models to RL. First, inaccurate or highly dynamic states, as well as delayed and discounted rewards, are expected to reduce the learning speed (or the convergence rate). Second, an agent must learn new and unexpected states and actions as time goes by as new circumstances of the operating environment may be encountered. The newly discovered states and actions are expected to affect the convergence to the optimal action, as well as the convergence rate.

7.4. Enhancement of Traffic Simulators for RL Investigation

In general, traffic simulators possess two main characteristics. First, most simulators adopt the microscopic approach, although macroscopic attributes can be applied to enhance learning. For instance, in Chu and Wang [2015], the macroscopic attributes are used to partition a traffic network into homogeneous subnetworks heuristically in order to reduce the number of state-action pairs (or feature vectors). Second, the discrete and time-step-based simulators have two major effects: (1) the road is segregated into cells, each of which can be occupied by a vehicle, and the length of the road is given by the number of cells on the road, and (2) the agent updates Q-values at the beginning or at the end of every time step only.

Future investigations could be pursued to explore the use of macroscopic attributes in order to improve the performance achieved by the microscopic-only approach and design time-continuous models to provide more accurate results.

7.5. Other Open Issues

Future investigations could be pursued to understand and enhance various aspects of RL models and algorithms applied to traffic signal control by exploring open issues using Tables IV, V, and VI. While the tables present a comprehensive overview of the research carried out on this topic, they show a wide range of potential enhancements and open issues that can be further investigated. As an example, there has been lack of research interest to maximize the global Q-value (T.1.2.1) of traffic signal controllers in arterial traffic networks with minor streets (M.1.5). As another example, there has been lack of research interest to maximize the global Q-value (T.1.2.1) of traffic signal controllers in the presence of traffic phases with grouped individual traffic (T.2.3).

Q-learning (H.1), which is an off-policy approach, has been adopted by most traffic signal controllers proposed in the literature. There is only a perfunctory effort to investigate the use of the on-policy approach, particularly SARSA (H.2), to improve traffic signal control. In Jin and Ma [2015b], SARSA is applied in multistep backup RL (see Section 4.5) in order to update the Q-values based on the average effects of the temporal differences gathered in an episode (or a sequence of time instants). In Jin and Ma [2015a], SARSA has shown to outperform Q-learning due to the instability of Q-learning as a result of considering the maximum possible Q-value, which is

dynamic in nature, of the next state. Further investigations could be pursued to adopt the on-policy approach (i.e., SARSA) to improve traffic conditions in traffic signal control systems.

In addition, future investigations could be pursued to compare the RL approaches, particularly those with similar strengths, such as MARL and max-plus RL, as well as the model-based and actor-critic RL approaches (see Table III).

8. CONCLUSIONS

This article presents a comprehensive review on the application of reinforcement learning (RL) models and algorithms to traffic signal control. The underlying intersection with different architectures (e.g., multi-intersection and arterial with minor streets) and dynamic traffic arrival rates (e.g., Poisson-based and real-world-based traffic models) have posed significant challenges to traffic signal controllers to select the right choice of traffic phase, as well as its duration, for smoother traffic flow. This article discusses how traffic signal control and the underlying intersection can be formulated as an RL problem using appropriate representations (i.e., state, action, and reward). Subsequently, this article presents various kinds of extended RL models and algorithms, and highlights their strengths in addressing the challenges brought about by the medium- and heavy-loaded traffic at intersections. The performance measures and complexities of the RL approaches are also investigated. Certainly, there remains a large amount of future work to address the open issues associated with RL models and algorithms for traffic signal control, and this article has laid a solid foundation and opened up new research interests in this area.

REFERENCES

M. Abdoos, N. Mozayani, and A. L. C. Bazzan. 2011. Traffic light control in non-stationary environments based on multi agent q-learning. In *Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems (ITSC'11)*. IEEE, 1580–1585. DOI: http://dx.doi.org/10.1109/ITSC.2011.6083114

S. Araghi, A. Khosravi, M. Johnstone, and D. Creighton. 2013. Q-learning method for controlling traffic signal phase time in a single intersection. In *Proceedings of the 16th International Conference on Intelligent Transportation Systems (ITSC'13)*. IEEE, 1261–1265. DOI: http://dx.doi.org/10.1109/ITSC.2013.6728404

I. Arel, C. Liu, T. Urbanik, and A. G. Kohls. 2010. Reinforcement learning-based multi-agent system for network traffic signal control. *IET Intelligent Transportation Systems* 4, 2 (2010), 128–135. DOI: http://dx.doi.org/10.1049/iet-its.2009.0070

E. Azimirad, N. Pariz, and M. B. N. Sistani. 2010. A novel fuzzy model and control of single intersection at urban traffic network. *IEEE Systems* 4, 1 (March 2010), 107–111. DOI: http://dx.doi.org/10.1109/JSYST.2010.2043159

L. Baird. 1995. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Machine Learning (ICML'95)*. IEEE, 30–37.

P. G. Balaji, X. German, and D. Srinivasan. 2010. Urban traffic signal control using reinforcement learning agents. *IET Intelligent Transportation Systems* 4, 3 (Sept. 2010), 177–188. DOI: http://dx.doi.org/10.1049/iet-its.2009.0096

A. L. C. Bazzan. 2009. Opportunities for multiagent systems and multiagent reinforcement learning in traffic control. *Autonomous Agents and Multi-Agent Systems* 18, 3 (June 2009), 342–375. DOI: http://dx.doi.org/10.1007/s10458-008-9062-9

A. L. C. Bazzan, M. de B. do Amarante, T. Sommer, and A. J. Benavides. 2010. ITSUMO: An agent-based simulator for ITS applications. In *Proceedings of the 4th Workshop on Artificial Transportation Systems and Simulation (ATSS'10)*. IEEE, 1–7.

C. Bettstetter and S. Konig. 2002. On the message and time complexity of a distributed mobility-adaptive clustering algorithm in wireless ad hoc networks. In *Proceedings of the 4th European Wireless Conference (EW'02)*. 128–134.

P. Chanloha, W. Usaha, J. Chinrungrueng, and C. Aswakul. 2012. Performance comparison between queueing theoretical optimality and q-learning approach for intersection traffic signal control. In *Proceedings of the 4th International Conference on Computational Intelligence, Modelling and Simulation (CIMSim'12)*. IEEE, 172–177. DOI: http://dx.doi.org/10.1109/CIMSim.2012.12

Y. K. Chin, W. Y. Kow, W. L. Khong, M. K. Tan, and K. T. K. Teo. 2012. Q-learning traffic signal optimization within multiple intersections traffic network. In *Proceedings of the 6th European Symposium on Computer Modeling and Simulation (EMS'12)*. IEEE, 343–348. DOI:http://dx.doi.org/10.1109/EMS.2012.75

M. C. Choy, D. Srinivasan, and R. L. Cheu. 2003. Cooperative, hybrid agent architecture for real-time traffic signal control. *IEEE Transactions on Systems, MAN, and Cybernetics – Part A: Systems and Humans* 33, 5 (Sept. 2003), 597–607. DOI:http://dx.doi.org/10.1109/TSMCA.2003.817394

T. Chu and J. Wang. 2015. Traffic signal control with macroscopic fundamental diagrams. In *Proceedings of the American Control Conference (AACC'15)*. IEEE, 4380–4385. DOI:http://dx.doi.org/10.1109/ACC.2015.7172018

S.-B. Cools, C. Gershenson, and B. D'Hooghe. 2013. Self-Organizing Traffic Lights: A Realistic Simulation. In *Advances in Applied Self-Organizing Systems*. Springer, London. 45–55. DOI:http://dx.doi.org/10.1007/978-1-4471-5113-5_3

Y. Dai, D. Zhao, and J. Yi. 2010. A comparative study of urban traffic signal control with reinforcement learning and adaptive dynamic programming. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'10)*. IEEE, 1–7. DOI:http://dx.doi.org/10.1109/IJCNN.2010.5596480

I. Dusparic, J. Monteil, and V. Cahill. 2016. Towards automatic urban traffic control with collaborative multi-policy reinforcement learning. In *Proceedings of the IEEE 19th International Conference o Intelligent Transportation Systems (ITSC'16)*. IEEE, 2065–2070.

S. El-Tantawy and B. Abdulhai. 2010. An agent-based learning towards decentralized and coordinated traffic signal control. In *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems (ITSC'10)*. IEEE, 665–670. DOI:http://dx.doi.org/10.1109/ITSC.2010.5625066

S. El-Tantawy and B. Abdulhai. 2012. Multi-agent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC). In *Proceedings of the 15th International Conference on Intelligent Transportation Systems (ITSC'12)*. IEEE, 319–326. DOI:http://dx.doi.org/10.1109/ITSC.2012.6338707

S. El-Tantawy, B. Abdulhai, and H. Abdelgawad. 2013. Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto. *IEEE Transactions on Intelligent Transportation Systems* 14, 3 (Sept. 2013), 1140–1150. DOI:http://dx.doi.org/10.1109/TITS.2013.2255286

A. Fares and W. Gomaa. 2014. Freeway ramp-metering control based on reinforcement learning. In *Proceedings of the 11th International Conference on Control and Automation (ICCA'14)*. IEEE, 1226–1231. DOI:http://dx.doi.org/10.1109/ICCA.2014.6871097

C. Gershenson and D. Rosenbluet. 2009. *Modeling Self-organizing Traffic Lights with Elementary Cellular Automata*. Technical Report 0907.1925. Universidad Nacional Autonoma de Mexico Ciuda.

M. R. Heinen, A. L. C. Bazzan, and P. M. Engel. 2011. Dealing with continuous-state reinforcement learning for intelligent control of traffic signals. In *Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems (ITSC'11)*. IEEE, 890–895. DOI:http://dx.doi.org/10.1109/ITSC.2011.6083107

P. Jain and M. Sethi. 2012. Fuzzy based real time traffic signal controller to optimize congestion delays. In *Proceedings of the International Conference of Advanced Computing and Communication Technology (ACCT'12)*. IEEE, 204–207. DOI:http://dx.doi.org/10.1109/ACCT.2012.55

J. Jin and X. Ma. 2015a. Adaptive group-based signal control by reinforcement learning. *Transportation Research Procedia* 10 (July 2015), 207–216. DOI:http://dx.doi.org/10.1016/j.trpro.2015.09.070

J. Jin and X. Ma. 2015b. Adaptive group-based signal control using reinforcement learning with eligibility traces. In *Proceedings of the IEEE 18th International Conference on Intelligent Transportation Systems (ITSC'15)*. IEEE, 2412–2417. DOI:http://dx.doi.org/10.1109/ITSC.2015.389

C. K. Keong. 1993. The GLIDE system-Singapore's urban traffic control system. *Transport Reviews* 13, 4 (March 1993), 295–305. DOI:http://dx.doi.org/10.1080/01441649308716854

M. A. Khamis and W. Gomaa. 2012. Enhanced multiagent multi-objective reinforcement learning for urban traffic light control. In *Proceedings of the 11th International Conference on Machine Learning and Applications (ICMLA'12)*. IEEE, 586–591. DOI:http://dx.doi.org/10.1109/ICMLA.2012.108

M. A. Khamis and W. Gomaa. 2014. Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. *Engineering Applications of Artificial Intelligence* 29 (March 2014), 134–151. DOI:http://dx.doi.org/10.1016/j.engappai.2014.01.007

M. A. Khamis, W. Gomaa, A. El-Mahdy, and A. Shoukry. 2012b. Adaptive traffic control system based on Bayesian probability interpretation. In *Proceedings of the Japan-Egypt Conference on Electronics, Communications and Computers (JEC-ECC'12)*. IEEE, 151–156. DOI:http://dx.doi.org/10.1109/JEC-ECC.2012.6186974

M. A. Khamis, W. Gomaa, and H. El-Shishiny. 2012a. Multi-objective traffic light control system based on Bayesian probability interpretation. In *Proceedings of the 15th International Conference on Intelligent Transportation Systems (ITSC'12)*. IEEE, 995–1000. DOI:http://dx.doi.org/10.1109/ITSC.2012.6338853

C.-G. Li, M. Wang, Z.-G. Sun, F.-Y. Lin, and Z.-F. Zhang. 2009. Urban traffic signal learning control using fuzzy actor-critic methods. In *Proceedings of the 5th International Conference of Natural Computation (ICNC'09)*. IEEE, 368–372. DOI:http://dx.doi.org/10.1109/ICNC.2009.374

W. Liu, J. Liu, J. Peng, and Z. Zhu. 2014. Cooperative multi-agent traffic signal control system using fast gradient-descent function approximation for V2I networks. In *Proceedings of the IEEE International Conference on Communications (ICC'14)*. IEEE, 2562–2567. DOI:http://dx.doi.org/10.1109/ICC.2014.6883709

J. C. Medina and R. F. Benekohal. 2012. Traffic signal control using reinforcement learning and the max-plus algorithm as a coordinating strategy. In *Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems (ITSC'12)*. IEEE, 596–601. DOI:http://dx.doi.org/10.1109/ITSC.2012.6338911

J. C. Medina, A. Hajbabaie, and R. F. Benekohal. 2010. Arterial traffic control using reinforcement learning agents and information from adjacent intersections in the state and reward structure. In *Proceedings of the 13th Annual Conference on Intelligent Transportation Systems (ITSC'10)*. IEEE, 525–530. DOI:http://dx.doi.org/10.1109/ITSC.2010.5624977

S. Mikami and Y. Kakazu. 1994. Genetic reinforcement learning for cooperative traffic signal control. In *Proceedings of the 1st IEEE Conference on Evolutionary Computation (ICEC'94)*. IEEE, 223–228. DOI:http://dx.doi.org/10.1109/ICEC.1994.350012

K. Nagel and M. Schreckenberg. 1992. A cellular automation model for freeway traffic. *Journal de Physique* 2, 12 (Sept. 1992), 2221–2229. DOI:http://dx.doi.org/10.1051/jp1:1992277

K. J. Prabuchandran, A. N. H. Kumar, and S. Bhatnagar. 2014. Multi-agent reinforcement learning for traffic signal control. In *Proceedings of the 17th International Conference on Intelligent Transportation Systems (ITSC'14)*. IEEE, 2529–2534. DOI:http://dx.doi.org/10.1109/ITSC.2014.6958095

K. J. Prabuchandran, A. N. H. Kumar, and S. Bhatnagar. 2015. Decentralized learning for traffic signal control. In *Proceedings of the 7th International Conference on Communication Systems and Networks (COMSNETS'15)*. IEEE, 2529–2534. DOI:http://dx.doi.org/10.1109/COMSNETS.2015.7098712

L. A. Prashanth and S. Bhatnagar. 2011. Reinforcement learning with function approximation for traffic signal control. *IEEE Transactions on Intelligent Transportation Systems* 12, 2 (June 2011), 412–421. DOI:http://dx.doi.org/10.1109/TITS.2010.2091408

L. A. Prashanth and S. Bhatnagar. 2012. Threshold tuning using stochastic optimization for graded signal control. *IEEE Transactions on Vehicular Technology* 61, 9 (Nov. 2012), 3865–3880. DOI:http://dx.doi.org/10.1109/TVT.2012.2209904

D. I. Robertson and R. D. Bretherton. 1991. Optimizing networks of traffic signals in real time-the SCOOT method. *IEEE Transactions on Vehicular Technology* 40, 1 (Feb. 1991), 11–15. DOI:http://dx.doi.org/10.1109/25.69966

A. Salkam, R. Cunningham, A. Garg, and V. Cahill. 2008. A collaborative reinforcement learning approach to urban traffic control optimization. In *Proceedings of the International Conference on Web Intelligence & Intelligent Agent Technology (WIIAT'08)*. IEEE, 560–566. DOI:http://dx.doi.org/10.1109/WIIAT.2008.88

A. G. Sims and K. W. Dobinson. 1980. The Sydney coordinated adaptive traffic (SCAT) system philosophy and benefits. *IEEE Transactions on Vehicular Technology* 29, 2 (May 1980), 130–137. DOI:http://dx.doi.org/10.1109/T-VT.1980.23833

S. Su and C.-K. Tham. 2007. SensorGrid for real-time traffic management. In *Proceedings of the 3rd International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP'07)*. IEEE, 443–448. DOI:http://dx.doi.org/10.1109/ISSNIP.2007.4496884

R. S. Sutton and A. G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

K. T. K. Teo, W. Y. Kow, and Y. K. Chin. 2010. Optimization of traffic flow within an urban traffic light intersection with genetic algorithm. In *Proceedings of the 2nd International Conference on Computational Intelligence, Modelling and Simulation (CIMSiM'10)*. IEEE, 172–177. DOI:http://dx.doi.org/10.1109/CIMSiM.2010.95

K. T. K. Teo, K. B. Yeo, Y. K. Chin, H. S. E. Chuo, and M. K. Tan. 2014. Agent-based traffic flow optimization at multiple signalized intersections. In *Proceedings of the 8th Asia Modelling Symposium (AMS'14)*. IEEE, 21–26. DOI:http://dx.doi.org/10.1109/AMS.2014.16

K. Wen, W. Yang, and S. Qu. 2010. A stochastic adaptive traffic signal control model based on fuzzy reinforcement learning. In *Proceedings of the Computer and Automation Engineering (ICCAE'10)*. IEEE, 467–471. DOI:http://dx.doi.org/10.1109/ICCAE.2010.5451248

M. Wiering, J. Vreeken, J. van Veenen, and A. Koopman. 2004. Simulation and optimization of traffic in a city. In *Proceedings of the IEEE Intelligent Vehicles Symposium*. IEEE, 453–458. DOI: http://dx.doi.org/10.1109/IVS.2004.1336426

B. Yin, M. Dridi, and A. El Moudni. 2016. Traffic network micro-simulation model and control algorithm based on approximate dynamic programming. *IET Intelligent Transport Systems* 10, 3 (March 2016), 186–196. DOI: http://dx.doi.org/10.1049/IET-ITS.2015.0108

D. Zhao, Y. Dai, and Z. Zhang. 2012. Computational intelligence in urban traffic signal control: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 42, 4 (July 2012), 485–494. DOI: http://dx.doi.org/10.1109/TSMCC.2011.2161577

Received July 2016; revised March 2017; accepted March 2017