Hierarchical multi-agent control of traffic lights based on collective learning

Junchen Jin[a,*], Xiaoliang Ma[a,b]

[a] *System Simulation & Control (S2CLab), Department of Transport Science, KTH Royal Institute of Technology, Teknikringen 10, Stockholm 10044, Sweden.*
[b] *iTekn Solutions, Stockholm Sweden.*

Abstract

Increasing traffic congestion poses significant challenges for urban planning and management in metropolitan areas around the world. One way to tackle the problem is to resort to the emerging technologies in artificial intelligence. Traffic light control is one of the most traditional and important instruments for urban traffic management. The present study proposes a traffic light control system enabled by a hierarchical multi-agent modeling framework in a decentralized manner. In the framework, a traffic network is decomposed into regions represented by region agents. Each region consists of intersections, modeled by intersection agents who coordinate with neighboring intersection agents through communication. For each intersection, a collection of turning movement agents operate individually and implement optimal actions according to local control policies. By employing a reinforcement learning algorithm for each turning movement agent, the intersection controllers are enabled with the capability to make their timing decisions in a complex and dynamic environment. In addition, the traffic light control operates with an advanced phase composition process dynamically combining compatible turning movements. Moreover, the collective operations performed by the agents in a road network are further coordinated by varying the priority settings for relevant turning movements. A case study was carried out by simulations to evaluate the performance of the proposed control system while comparing it with an optimized vehicle-actuated control system. The results show that the proposed traffic light system, after a collective machine learning process, not only improves the local signal operations at individual intersections but also enhances the traffic performance at the regional level through coordination of specific turning movements.

Key words: Hierarchical model of traffic system, multi-agent traffic light control, decentralized system, learning-based control, collective machine learning.

1. Introduction

Most people living in a populated area suffer from traffic congestion problems as traffic consumes time, energy, and patience. Traffic light control (TLC) is a crucial traffic management instrument in urban areas. For several decades, researchers have been using mathematical and computational methods to facilitate efficient traffic signal operations. A conventional approach in TLC planning applies an off-line optimization model to find the most appropriate signal parameters according to historical traffic observations (e.g., Ma et al., 2014). Although it is still a common practice in traffic engineering, the off-line approaches are limited because overall traffic patterns are stochastic and time-varying and historical data cannot adequately capture real-time traffic situations.

Along with the evolution of new concepts and technologies, some adaptive TLC systems have been proposed to address the issues in conventional TLC systems. Adaptive TLC systems normally adjust control parameters in accordance with real-time traffic patterns. For instance, SCOOT (Hunt et al., 1982), SCATS (Sims and Dobinson, 1980), RHODES (Mirchandani and Head, 2001) and TUC (Boillot et al., 2006) are examples of adaptive traffic signal

*Corresponding author. Tel: +46 87908499; Email: junchen@kth.se.

Postprint of Paper V: **Jin, J.** and Ma, X. (2018). Hierarchical multi-agent control of traffic lights based on collective learning. *Engineering Applications of Artificial Intelligence*, 68: 236-248.

systems that have been implemented in real applications. An adaptive TLC system can be further classified as a centralized system or a decentralized system.

In a centralized system, traffic light controllers are managed through a traffic control center that monitors traffic networks and performs optimization techniques to utilize the existing infrastructure better. Despite the fact that the ultimate goal of the centralized approach is to optimize the system-wide performance measures, its efficiency is questionable. Three issues of robustness, scalability, and efficiency were raised in previous studies for large-scale systems or networks with complex structures (e.g., El-Tantawy and Abdulhai, 2013). Conversely, each intersection in a decentralized system operates individually and autonomously. Due to the relative ease of implementation and other advantages mentioned above, there have been increasing efforts in developing decentralized systems (e.g., Cools et al., 2013). Simultaneously, emerging technologies for inexpensive computing and communication devices provide accessible opportunities to introduce decentralized schemes into the TLC system in more markets (Jin et al., 2017a).

In recent studies, a multi-agent framework is widely used in modeling TLC operations. Meanwhile, emerging methods in machine learning enable agents in such framework to build their knowledge and operating guidelines based on feedback information concerning mobility performance and other measures like energy efficiency and environmental impacts. Approximate dynamic programming (ADP) or reinforcement learning (RL) (Barto, 1998), rooted in the perspectives of mimicking human-level intelligence, provides an insightful approach on how intelligent agents optimize their control within an application context, such as in games with high-complexity (Mnih et al., 2015).

This paper extends the RL-based intersection adaptive control approach proposed in Jin and Ma (2017) for operating a network of signalized intersections. The rest of this paper is organized as follows. Section 2 reviews several state-of-the-art TLC systems based on multi-agent modeling framework and ADP- or RL-based approaches. In section 3, a hierarchical modeling framework is introduced to represent a decentralized TLC system. This is followed by a detailed presentation of the intersection control (in section 4) and a description of the proposed collective learning process (in section 5). A case study is finally carried out with experimental setup, agent design, traffic simulations, result analysis, and discussions elaborated in section 6. The last section concludes this paper by summarizing the main findings together with an outlook on future research.

2. Relevant studies

The application of RL to TLC systems was first introduced in 1996 when Thorpe and Anderson (1996) proposed an adaptive intersection control scheme capable of modifying signal timing according to traffic conditions. The authors claimed that the proposed controller outperformed a fixed-time controller by reducing the average waiting time of vehicles at an intersection. Since then, research in this area has moved towards integrating agent-based modeling technology with advanced RL or ADP methods. Table 1 summarizes the recent developments in RL-based TLC systems, and their system designs, concerning state, action, reward function and learning algorithm, are compared in details.

For intersection control, the major difference between the proposed systems lies in their agent design approach. They can be categorized by taking different entities as agents, including vehicle, intersection, and component of a signal controller. According to the design level of detail, the component of a signal controller could refer to:

- a turning movement;
- a signal group representing a group of turning movements;
- a signal phase composed of a collection of signal groups.

Among the agent designs above, several recent approaches model traffic light controller by intersection agents that determine signal plans (e.g., Bazzan et al., 2010), or green duration of each signal phase (e.g., Balaji et al., 2010; Abdoos et al., 2014), or a selection of phase with a defined time interval (e.g., Arel et al., 2010; El-Tantawy et al., 2013). The phase sequence of each intersection agent is predetermined in these studies. Group-based phasing approaches have proved their utilities by dynamically generating phase structures and sequences with respect to traffic detected at intersection (e.g., Wong and Wong, 2003; Jin et al., 2017b). A previous study by the same authors proposed an RL-based adaptive TLC system suitable for group-based phasing strategies (Jin and Ma, 2017). The system showed the benefits in improving traffic mobility when compared to a conventional logic-based timing approach.

Table 1
A summary of typical studies that apply RL-based multi-agent modeling frameworks to TLC systems

Literature	Agent	RL model	State	Action	Reward
Bazzan et al. (2010)	Intersection	Q-learning	Three-value state according to the vehicle loading	One of three pre-defined signal plans	Average queue length in all links
Arel et al. (2010)	Intersection	Q-Learning with neural network for function approximation	The total delay of vehicles in a lane divided by the average delay at all lanes	One of the pre-defined available phases	Variation in travel delay
Balaji et al. (2010)	Intersection	Q-learning	Occupancy ratio, local traffic variations, and neighboring states	Green time for each phase	Variation in queue length
El-Tantawy et al. (2013)	Intersection	Model-based Q-learning (MARLIN)	Index of current phase, elapsed time, queue length associated with each lane	One of the pre-defined available phases	Variation in travel delay
Abdoos et al. (2014)	Bottom level: intersection; Top level: region	Bottom level: Q-learning; Top level: Q-learning with tile coding for function approximation	Bottom level: ranks determined by average queue lengths; Top level: average queue length of all links inside the region	Bottom level: the portion of green time for the phases Top level: one of three restrictions for agents at the bottom level	Average queue length in all links
Khamis and Gomaa (2014)	Vehicle	Model-based Q-learning	The status of the traffic light of the lane in which the vehicle is moving or waiting, vehicle position, and vehicle traveling destination	Green or red indication for the traffic lights that the vehicle agent is associated with	Average trip waiting time and average travel time
Jin and Ma (2017)	Signal group	SARSA with multiple-step backups	Vehicle arrival gap, occupancy ratio, elapsed green time, phase status, and neighboring states	Green extension between 0 and 4 seconds	Variation in travel delay

Unlike intersection and signal component agents, the state of vehicle agent includes vehicle information (e.g., vehicle position and destination) at an intersection (Khamis and Gomaa, 2014). However, the implementation of such system is not cost-effective since it requires replacement of the deployed infrastructure to support a broad coverage of connected vehicles such that vehicle information can be obtained through vehicle-to-infrastructure (V2I) communication. However, when considering traffic engineering practice, it is still preferred to developing an advanced TLC

system based on information from existing infrastructure deployed.

In terms of the specific learning algorithm applied, both model-free (e.g., Bazzan et al., 2010; Arel et al., 2010; Balaji et al., 2010; Abdoos et al., 2013, 2014; Jin and Ma, 2017) and model-based (e.g., El-Tantawy et al., 2013; Khamis and Gomaa, 2014) approaches have been incorporated in TLC systems. For model-based RL, an agent is required to adequately model the external environment so that it can be used to find an optimal action sequence. While model-based RL algorithms may show better learning efficiency compared to model-free approaches, it is rather difficult to model traffic system given its stochastic properties and uncertainties involved in driver behaviors.

Two techniques of model-free RL, including off-policy (i.e., Q-learning) and on-policy knowledge update methods (i.e., SARSA), have been tested in the literature. Most TLC systems adopted Q-learning, whereas little effort was put on applications of SARSA (i.e., Thorpe and Anderson, 1996; Jin and Ma, 2017). Moreover, some studies applied function approximation (FA) to address the traditional issue of "the curse of dimensionality" in RL problems (e.g., Arel et al., 2010; Prashanth and Bhatnagar, 2011; Abdoos et al., 2014). The studies in Cai et al. (2009) and Jin and Ma (2017) enhanced the model-free RL/ADP by incorporating multiple-step backups (i.e., the eligibility trace strategy) that take into account longer-term effects of an action.

Recent research direction in the control of several signalized intersections in a network is to adopt a hierarchical framework to operate traffic lights in a decentralized multi-agent environment, where a network is decomposed into multiple sub-networks and each sub-network is considered as a region agent (e.g., Bazzan et al., 2010; Abdoos et al., 2013, 2014). These studies explored the benefits of reducing problem complexity and improving system performance and learning efficiency when cooperation between agents is enabled. However, few accounted for signal coordination within a region to generate "green wave" scenario (i.e., a series of traffic lights coordinate to allow continuous traffic flow over several intersections in one direction), which is important for network traffic control in engineering practice.

This study employs a hierarchical framework for the network control, and the focus here is on dealing with the "green wave" scenario via communication between agents at different intersections within a region. For each intersection, a general turning movement-based phasing control is utilized to generate phase sequences in real time. FA is also applied to enhance learning efficiency and accuracy.

3. Decentralized signal control system

It is common practice to divide a network of signalized intersections into regions with signal controllers in each region carrying out the same control strategy. The partition on regions mainly follows a high-level operational objective. A signal controller usually operates traffic lights that are associated with an intersection. The basic component of intersection control is a turning movement. Consequently, network-wide TLC can be represented as a hierarchical multi-agent framework. Three agents, from top to bottom levels of the hierarchy, include the region agent (RA), intersection agent (IA) and turning movement agent (TA).

Fig. 1 illustrates the concept and essential elements for a decentralized TLC framework in a network along with an RA consisting of three closely-spaced intersections. Agents share a communication language that enables their social capability when they are at the same level. Whereas, the hierarchical framework is designed in a decentralized manner such that an agent is limited to only communicating with its neighboring agents. Specifically, communication between RAs works with regional control plans (e.g., peak hour plan and off-peak hour plan), while communication among IAs exchanges the information about the directions generating the "green wave" scenarios.

An agent at lower level acts to follow its individual goal while cooperating and working towards a common goal as instructed by the agent at the next higher level. It is required that no conflicts exist between the goals of agents at different levels of the hierarchy. An RA instructs its subordinating IAs for signal operations and coordination. Based on the command from the RA and communicating information from the neighboring IAs, an IA sends customized restrictions to each TA performing at the intersection.

In Fig. 1, twelve, seven, and seven TAs are formulated in the three intersections, respectively. At each intersection, TAs share a common external environment, i.e., intersection traffic system. By relying on the deployed traffic detection system, TAs are able to perceive states from the environment during their operation. Since they can approach the full detection information at an intersection, there is no need to impose communication among the agents within an intersection.

While the agents in a system are often endowed with behaviors designed in advance, applications also require them to enhance their behaviors on-line so that the performance of each agent, or the whole multi-agent system, may

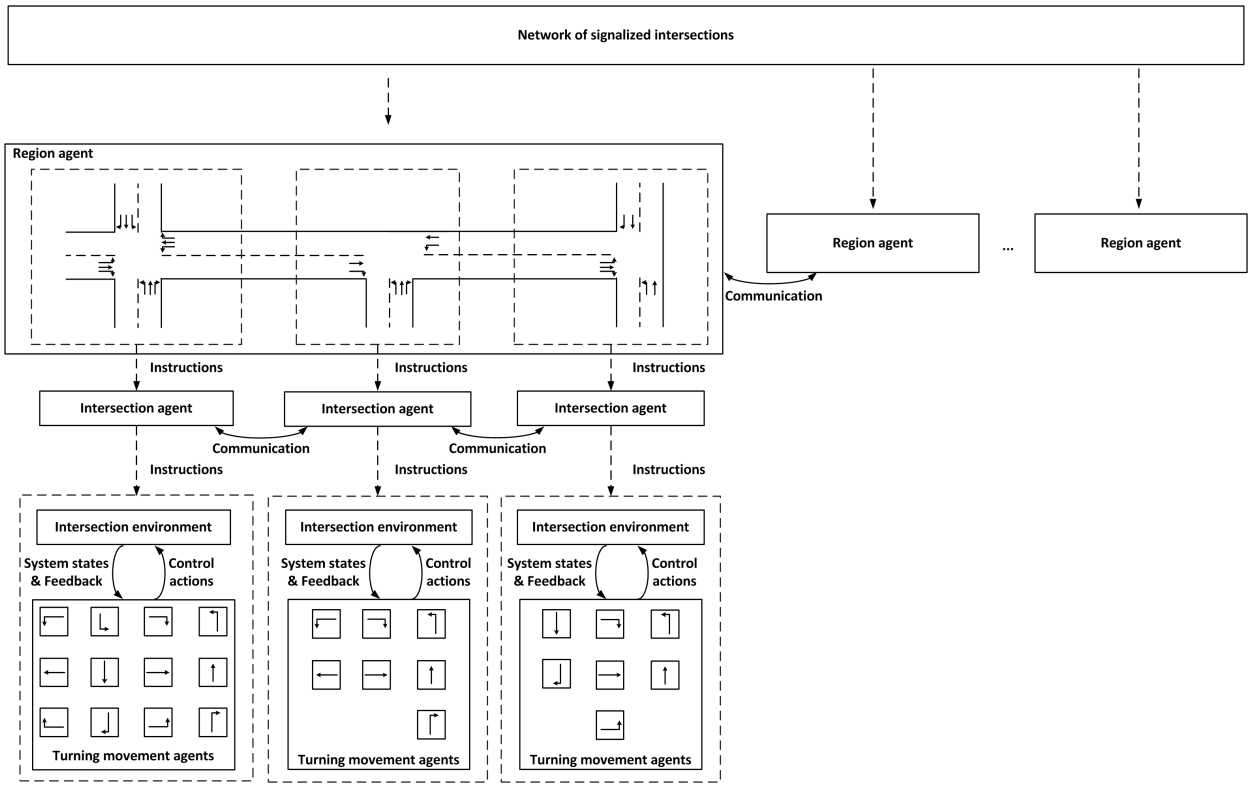

Fig. 1. A multi-agent representation of a decentralized TLC system for a network of signalized intersections

Table 2
The essential information for the decision process of a TA

System element	Symbol	Explanation
State	$\mathbf{x}$	An element that describes the detected traffic information
Action	$\mathbf{u}$	Signal operation of the controlled turning movement
Reward	r	The received benefit for an agent carrying out an action in a certain state
Control policy	π	A mapping from state to action
Instructions	$\boldsymbol{\phi}$	The instructions for the associated IA

improve gradually. Such a requirement can be fulfilled by using the feedback from the environment after a TA acts. Specifically, the agent builds up its knowledge base through a learning algorithm using input from the aforementioned information (i.e., state, action, and feedback). Then, a control action is dynamically identified according to the stored knowledge base. With such a learning system, each TA considers two interrelated skills:

- how to perform a subtask optimally;
- how to define the system state cooperatively.

Consequently, the collective behaviors performed by TAs provide the signal operations at an intersection, and a region of intersections is operated in a decentralized scheme incorporating these collective behaviors from all the comprising intersections. The essential information required to enable the decision process of a TA is summarized in Table 2 along with the symbols used throughout this paper.

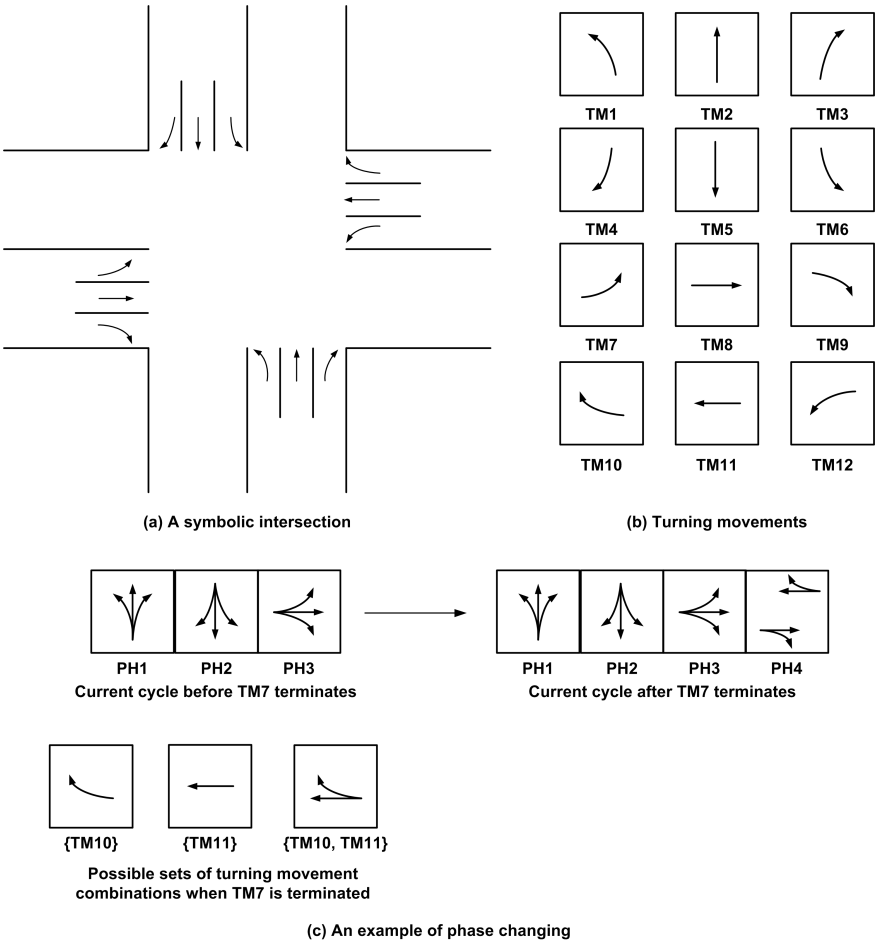

Fig. 2. A symbolic intersection, its traffic movements and a phasing change operation

4. Intersection control approach

In an advanced TLC system, signal control strategies at an isolated intersection are defined from two perspectives: signal phasing and signal timing. In real operation, it is possible for compatible turning movements to show the same traffic light at all time. Phasing control determines how compatible turning movements are formed in operation. Traffic lights alternate the right-of-way by displaying green, yellow, and red indications in a sequence while timing control refers to allocating durations of traffic light indications. This section first explains a phase control approach, called turning movement-based phasing, which is extended from the group-based phasing in Jin and Ma (2017). In this approach, lane markings are pre-determined and dynamically changing lane markings is not considered.

4.1. Turning movement-based phasing

A phase is usually comprised of more than one turning movements that are not mutually in conflict. In Fig. 2, a four-armed symbolic intersection and its corresponding twelve turning movements are presented. A conflict matrix is

used to represent the conflict situations among the turning movements, and below shows one example:

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 2 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 0 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 2 & 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 2 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 2 \\ 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 0 & 2 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{1}$$

In the matrix, a value more than zero refers to the all-red duration between the pair of conflict movements due to safety concerns, whereas a value of zero represents that the corresponding two turning movements may be activated at the same time.

The termination time of each turning movement in a phase is not identical due to different timing assigned. As long as there are some turning movements with an order to terminate, the current phase is expected to reach a termination. When a phase terminates, a new phase is then created by combining the remaining turning movements and the newly activated turning movements. Turning movements satisfying the following three conditions are considered as a candidate set:

- they are not in conflict with each other;
- each turning movement has no conflict with any of the remaining turning movements in the current phase;
- they have not been activated in the current cycle.

The set, with the largest cardinality out of all possible combinations, is chosen to be newly activated turning movements since the phasing control promotes as many turning movements as possible to be activated at once. Fig. 2c is an example showing how a phase is evolved. Assume that three phases, PH1, PH2, and PH3, are already activated in the current cycle. TM7, TM8, and TM9 form the current phase, PH4. If TM7 is ordered to terminate, then three possible sets of turning movements are available, including {TM10}, {TM11}, and {TM10, TM11} (see the left-bottom corner of Fig. 2c). Among these, the set of {TM10, TM11} satisfies the three conditions above and owns the largest cardinality, so the turning movements in this set are selected as the newly inserted turning movements. Accordingly, the current cycle after the phase changing is presented on the right-hand side of Fig. 2c.

4.2. Learning-based timing control

When the control feedback is involved, the signal timing scheme of an agent can be modeled as a generalized Markov decision process and is subsequently solved by the RL approaches presented in Jin and Ma (2017). Several RL methods, including Q-learning, SARSA, and SARSA with multiple-step backups, were implemented and tested for group-based TLC systems (Jin and Ma, 2015a, 2017). In an earlier study, two eligibility tracing strategies were investigated for the group-based system (Jin and Ma, 2015b) applying SARSA with multiple-step backups. As a result, SARSA outperforms Q-learning with respect to learning efficiency, especially when traffic demand fluctuates. Also, the replacing trace strategy is superior to the accumulating trace strategy in terms of improving mobility efficiency. The following subsection, thus, presents a general formulation of the intelligent timing scheme, and the corresponding solution approach using SARSA with multiple-step backups.

4.2.1. Basic principles

As introduced in Jin and Ma (2017), the knowledge base of an intelligent agent is represented by the expected value of the summation of immediate rewards in TLC system. Let $Q_i^{\pi_i}(\mathbf{x}_{i,0}, \mathbf{u}_{i,0})$ denote the knowledge base of TA i

under a sequence action $\pi_i = \{\mathbf{u}_{i,1}, \mathbf{u}_{i,2}, \dots\}$ given the initial state-action pair $(\mathbf{x}_{i,0}, \mathbf{u}_{i,0})$, where $\mathbf{x}_{i,t}$ and $\mathbf{u}_{i,t}$, respectively, denote the state and action of TA i at time t. TA's knowledge base is expressed by

$$\begin{aligned} Q_i^{\pi_i}(\mathbf{x}_{i,0}, \mathbf{u}_{i,0}) &= \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R_{i,t} \sigma_{i,t} | \mathbf{x}_{i,0}, \boldsymbol{u}_{i,0}] \\ &= \sum_{t=0}^{\infty} \sum_{\mathbf{x}_{i,t+1} \in \mathcal{X}_i} \gamma^t p_i(\mathbf{x}_{i,t+1} | \mathbf{x}_{i,0:t}, \mathbf{u}_{i,0:t}) r_i(\mathbf{x}_{i,0:t+1}, \mathbf{u}_{i,0:t}) \sigma_{i,t}, \end{aligned} \tag{2}$$

where $R_{i,t}$ denotes the reward variable and $\mathcal{X}_i$ is the state set of the agent. $p_i(\mathbf{x}_{i,t+1} | \mathbf{x}_{i,0:t}, \mathbf{u}_{i,0:t})$ represents the probability of the state transiting to $\mathbf{x}_{i,t+1}$ given the previous state-action pairs $(\mathbf{x}_{i,0:t}, \mathbf{u}_{i,0:t})$, where $\mathbf{x}_{i,0:t} = \{\mathbf{x}_{i,0}, \mathbf{x}_{i,1}, \mathbf{x}_{i,2}, ..., \mathbf{x}_{i,t}\}$ and $\mathbf{u}_{i,0:t} = \{\mathbf{u}_{i,0}, \mathbf{u}_{i,1}, \mathbf{u}_{i,2}, ..., \mathbf{u}_{i,t}\}$ representing a sequence of states and actions over time, respectively. $\gamma \in [0, 1]$ denotes the discount rate, which accounts for the level of importance of the future rewards. $\sigma_{i,t}$ refers to a binary variable indicating whether the TA is active at the time, i.e.,

$$\sigma_{i,t} = \begin{cases} 1, & \text{if TA } i \text{ is active at } t; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

To apply an appropriate timing action for an active TA, the optimal control can be obtained by solving

$$\mathbf{u}_{i,t}^* = \arg\max_{\mathbf{u}_{i,t}} \sum_{i}^{n} Q_i^{\pi_i}(\mathbf{x}_{i,t}, \mathbf{u}_{i,t}), \quad \mathbf{u}_{i,t} \in \mathcal{U}_i, \tag{4}$$

where n denotes the number of TAs at an intersection, and $\mathcal{U}_i$ denotes the action set of TA i. A more detailed derivation is given in Jin and Ma (2017).

SARSA with multiple-step backups is employed to find the optimal solution without knowing any information about the system dynamics, including the transition probability and immediate reward functions. The algorithm enables a TA to look backward to the beginning of the defined learning horizon. The following example briefly illustrates the update procedure of SARSA with multiple-step backups. Assume the state of TA i is $\mathbf{x}_{i,t}$ and the agent takes action $\mathbf{u}_{i,t}$ at t. Then the agent receives a reward value of $r_{i,t+1}$ and its state vector becomes $\mathbf{x}_{i,t+1}$. The estimated optimal cumulative reward, $Q_{i,t}(\mathbf{x}_{i,t}, \mathbf{u}_{i,t})$, corresponding to the state-action pair, $(\mathbf{x}_{i,t}, \mathbf{u}_{i,t})$, is updated through

$$Q_{i,t+1}(\mathbf{x}_i, \mathbf{u}_i) = Q_{i,t}(\mathbf{x}_i, \mathbf{u}_i) + \alpha \delta_{i,t}(\mathbf{x}_{i,t}, \mathbf{u}_{i,t}) e_{i,t}(\mathbf{x}_i, \mathbf{u}_i), \quad \forall \mathbf{x}_i \in \mathcal{X}_i, \forall \mathbf{u}_i \in \mathcal{U}_i, \tag{5}$$

where $\alpha \in [0, 1]$ refers to the learning rate. $\delta_{i,t}(\mathbf{x}_{i,t}, \mathbf{u}_t)$ and $e_{i,t}(\mathbf{x}_t, \mathbf{u}_t)$, respectively, represent the temporal difference and eligibility trace at time t. The temporal difference is computed by

$$\delta_{i,t}(\mathbf{x}_{i,t}, \mathbf{u}_{i,t}) = r_{i,t+1} + \gamma Q_{i,t}(\mathbf{x}_{i,t+1}, \mathbf{u}_{i,t+1}) - Q_{i,t}(\mathbf{x}_{i,t}, \mathbf{u}_{i,t}), \tag{6}$$

and the eligibility trace is updated using

$$e_{i,t}(\mathbf{x}_i, \mathbf{u}_i) = \begin{cases} 1, & \text{if } \mathbf{x}_i = \mathbf{x}_{i,t} \text{ and } \mathbf{u}_i = \mathbf{u}_{i,t}; \\ 0, & \text{if } \mathbf{x}_i = \mathbf{x}_{i,t} \text{ and } \mathbf{u}_i \neq \mathbf{u}_{i,t}; \\ \gamma \lambda e_{i,t-1}(\mathbf{x}_i, \mathbf{u}_i), & \text{if } \mathbf{x}_i \neq \mathbf{x}_{i,t}, \end{cases} \tag{7}$$

where $\lambda \in [0, 1]$ denotes the trace decay rate.

The next control $\mathbf{u}_{i,t+1}$ in the temporal difference is determined by the epsilon-greedy policy in which a random action is selected with $\epsilon_{i,t}$ probability while a greedy action is selected with a probability of $1 - \epsilon_{i,t}$.

$$\mathbf{u}_{i,t+1} = \begin{cases} \text{uniform}(\mathbf{u}_i), & \text{if } \xi < \epsilon_{i,t}; \\ \arg\max_{\mathbf{u}_i} Q_{i,t}(\mathbf{x}_{i,t+1}, \mathbf{u}_i), & \text{otherwise,} \end{cases} \quad \mathbf{u}_i \in \mathcal{U}_i, \tag{8}$$

where $0 \leq \xi \leq 1$ is a random number drawn from a uniform distribution. ϵ is decayed every episode by a predefined decay parameter τ from a defined initial value $\epsilon_{i,0}$ if the TA is active, i.e.,

$$\epsilon_{i,t} = \begin{cases} \epsilon_{i,0} & \text{if } t = 0; \\ \tau^{\sigma_{i,t}} \epsilon_{i,t-1} & \text{otherwise.} \end{cases} \tag{9}$$

In this way, the agent tends to explore the action space in the beginning, and then exploit the area beyond the ones previously explored during the knowledge building process.

4.2.2. Function approximation

This study extends the previously applied RL approaches by utilizing a function approximator to increase learning efficiency and accuracy. FA is a supervised learning approach that facilitates the representation and memory of agent knowledge (i.e., the expected value of the cumulative reward) using the incrementally acquired state-action data and feedback information. Linear function approximators are the most common FA approach to incorporate with RL algorithms, in which the cumulative reward is represented by a weighted linear summation of a set of features (e.g., Prashanth and Bhatnagar, 2011; Jin and Ma, 2016). However, they have a limitation of defining features for continuous variables in a complex system. This study applies the k-nearest neighbor (KNN) approach for FA (McCallum et al., 1995) to address such an issue.

KNN approximates the Q values with respect to unvisited states using those Q values that are generated by the previously visited states. At an update step t, KNN approximates a Q value corresponding to a given state-action pair, $(\mathbf{x}_{i,t}, \mathbf{u}_{i,t})$, for agent i by means of three components:

- $\mathcal{X}_{i,t}^{visit}$: a set storing the previously-visited states;
- $\mathcal{X}_{i,t}^{knn}$: a set of states which collects the k nearest neighbors with respect to the state $\mathbf{x}_{i,t}$ from $\mathcal{X}_{i,t}^{visit}$;
- $\psi(\cdot, \cdot)$: a weight function defining a particular proportional activation of each of the visited states.

For each updated Q value, a nearer neighbor has a higher contribution than those at a relative distance. Analytically, the estimation of the Q value for each unvisited state is computed by

$$\hat{Q}_{i,t}(\mathbf{x}_i, \mathbf{u}_i) = \sum_{\mathbf{x}_i^{knn} \in \mathcal{X}_{i,t}^{knn}} \frac{Q_{i,t}(\mathbf{x}_i^{knn}, \mathbf{u}_t)\psi(\mathbf{x}_i, \mathbf{x}_i^{knn})}{\psi(\mathbf{x}_i, \mathbf{x}_i^{knn})}, \quad \mathcal{X}_{i,t}^{knn} \subseteq \mathcal{X}_{i,t}^{visit}, \forall \mathbf{x}_i \in \mathcal{X}_i \backslash \mathcal{X}_{i,t}^{visit}, \forall \mathbf{u}_i \in \mathcal{U}_i. \tag{10}$$

where $\hat{Q}_{i,t}(\mathbf{x}_i, \mathbf{u}_i)$ denotes an estimation of $Q_{i,t}(\mathbf{x}_i, \mathbf{u}_i)$. The weight is represented by the Euclidean distance in a Gaussian kernel, i.e.,

$$\psi(\mathbf{x}_i, \mathbf{x}_i^{knn}) = e^{-\frac{|\mathbf{x}_i - \mathbf{x}_i^{knn}|^2}{b}}, \tag{11}$$

where b is the parameter that controls the breadth of generalization over the space of neighboring states.

5. Collective learning process

For a TLC system containing several intersections, the operation is a result of collective behaviors of the involved TAs at each intersection. A collective operation process is implemented for the proposed system, as summarized in Fig. 3. Two operational modes are used to separate the operation processes before and after the system is deployed in real-world situations: off-line training and on-line operation. The off-line training mode aims at building the knowledge base of TAs using simulation data or empirical data in advance, whereas the system can simultaneously be enhanced when it is employed in the on-line operation mode.

In the off-line training process, some learning iterations, N, are performed for the sake of exploring the space of the knowledge base. In each trial, the number of learning steps, T, is given. For each learning step, states and rewards are obtained for each agent in a signal controller to update the knowledge base. Following the updated knowledge bases, the signal controller carries out actions for the subsequent one-step operation. After the agents build their

```
   Input: S^sig: a set of signal controllers;
          M: operational mode;
          N: the number of learning iterations;
          T: the step size in each learning iteration;
          α, γ, ϵ_0, τ, λ: learning parameters;
          g^up, g^low, y, C: signal control parameters.
   // The operational mode is off-line training
 1 if M == "off-line training" then
 2     for all j ∈ {1, 2, ..., N} do
 3         for all t ∈ {1, 2, ..., T} do
 4             for all sig ∈ S^sig do
 5                 x, r = get_states_and_rewards(sig);
 6                 sig.update_knowledge(x, r, α, γ, ϵ_0, τ, λ, g^up, g^low, y, C);
 7                 u = sig.get_actions();
 8                 carry_out_actions(sig, u);
 9             end
10         end
11     end
   // The operational mode is on-line learning
12 else
13     while M = "on-line operation" do
14         for all sig ∈ S^sig do
               // Check if there are new states and rewards being received
15             E = sig.states_rewards_are_received();
16             while E do
17                 x, r = receive_states_and_rewards();
18                 sig.update_knowledge(x, r, α, γ, ϵ_0, τ, λ, g^up, g^low, y, C );
19                 u = sig.get_actions();
20                 carry_out_actions(sig, u);
21             end
22         end
23     end
24 end
```

Fig. 3. The operation processes for the decentralized system in the "off-line training" and "on-line operation" modes

knowledge bases through repeatedly running the defined learning iterations, the system is ready to be deployed for controlling traffic lights at intersections.

When the system is in the on-line operation mode, its functionality of updating knowledge base becomes event-based, and the system keeps listening to the deployed traffic surveillance system. Every time a signal controller receives new states and rewards, it calls for the function to update the knowledge base of the involved TAs. The following steps are same as those in the off-line training mode, where the agents carry out actions according to their developed knowledge bases.

The function describing the updating process of the knowledge base for a TA is presented by the pseudo-code in Fig. 4. Each TA is first checked if it is active or not, and then only the active one is requested to conduct the following process (see line 3). Assume that one TA tm is active. The specific current state and reward, $\mathbf{x}_{tm}$ and r_{tm}, as well as the previous state and action, $\tilde{\mathbf{x}}_{tm}$ and $\tilde{\mathbf{u}}_{tm}$, are extracted with respect to the active TA (see lines 4 – 5). The associated set, $\mathcal{E}_{tm}$, storing eligibility traces is also obtained.

Lines 7 – 23 are the core part of the knowledge updating process. A previous cumulative reward value Q is

Input : $\alpha, \gamma, \epsilon_0, \tau, \lambda$: learning parameters;
g^{up}, g^{low}, y, C: signal control parameters;
$\mathbf{x}, \mathbf{r}$: current states and rewards for the TAs associated with a signal controller.
Attribute: $\mathcal{S}^{tm}$: the set of turning movements of a signal controller;
$\mathbf{u}^{terminate}$: the defined termination action;
$e^{threshold}$: the defined threshold of decay trace rate.

for ***all*** $tm \in \mathcal{S}^{tm}$ **do**
  $\mathcal{A}$ = tm.is_active();
  if $\mathcal{A}$ **then**
    $\mathbf{x}_{tm}, r_{tm}$ = extract_state_control_reward($tm, \mathbf{x}, \mathbf{r}$);
    $\tilde{\mathbf{x}}_{tm}, \tilde{\mathbf{u}}_{tm}$ = tm. get_previous_state_and_control() ;
    $\mathcal{E}_{tm}$ = tm.extract_eligibility_trace_set();
    `// Get the` Q `value for the previous state-action pair according to the FA in Equation 10`
    $\tilde{Q}$ = function_approximator($\tilde{\mathbf{x}}_{tm}, \tilde{\mathbf{u}}_{tm}$) ;
    `// Get the` Q `value for the current state-action pair`
    if $\tilde{\mathbf{u}}_{tm} == \mathbf{u}^{terminate}$ **then**
      $\mathbf{u}_{tm}$ = $Null$;
      Q = 0;
    else
      $\mathbf{u}_{tm}$ = action_policy($\mathbf{x}_{tm}, \epsilon_0, \tau$);
      Q = function_approximator($\mathbf{x}_{tm}, \mathbf{u}_{tm}$) ;
    end
    `// Compute the temporal difference value according to Equation 6`
    δ = compute_temporal_difference($r_{tm}, Q, \tilde{Q}, \gamma$) ;
    `// Update the` Q `Values according to Equation 5`
    update_Q_values($\mathcal{E}_{tm}, \delta, e, \alpha$);
    for ***all*** $\{\mathbf{x}_{\mathcal{E}}, \mathbf{u}_{\mathcal{E}}\} \in \mathcal{E}_{tm}$ **do**
      `// Update the eligibility traces according to Equation 7`
      $e(\mathbf{x}_{\mathcal{E}}, \mathbf{u}_{\mathcal{E}})$ = update_eligibility_traces($\mathbf{x}_{\mathcal{E}}, \mathbf{u}_{\mathcal{E}}, \gamma, \lambda$) ;
      if $e(\mathbf{x}_{\mathcal{E}}, \mathbf{u}_{\mathcal{E}}) < e^{threshold}$ **then**
        $\mathcal{E}_{tm}$.remove($\{\mathbf{x}_{\mathcal{E}}, \mathbf{u}_{\mathcal{E}}\}$);
      end
    end
    tm.update_eligibility_trace_set($\mathcal{E}_{tm}$);
    if $\tilde{\mathbf{u}}_{tm} == \mathbf{u}^{terminate}$ **then**
      $\mathbf{x}_{tm}$ = $Null$; $\mathbf{u}_{tm}$ = $Null$; $\tilde{\mathbf{x}}_{tm}$ = $Null$; $\tilde{\mathbf{u}}_{tm}$ = $Null$; r = 0;
      $\mathcal{E}_{tm}$.clear();
      tm.deactivate() ;
    else
      $\mathcal{E}_{tm}$.add($\mathbf{x}_{tm}, \mathbf{u}_{tm}$);
      tm. set_previous_state_and_control($\mathbf{x}_{tm}, \mathbf{u}_{tm}$)
    end
  end
end

Fig. 4. One-step update procedure for a signal controller enabled by the proposed RL algorithm

obtained by a function approximator given the previous state-action pair, $(\tilde{\mathbf{x}}_{tm}, \tilde{\mathbf{u}}_{tm})$, for agent tm. Next, the previous action is checked. If this is a terminating action, then the current Q value is estimated to be zero together with a "Null"

value for the current action. Otherwise, the current action, $\mathbf{u}_{tm}$, is chosen according to the applied control policy, and the Q value is the result of the function approximator with respect to the current state-action pair $(\mathbf{x}_{tm}, \mathbf{u}_{tm})$(see lines 8 – 14). Using the two cumulative reward values, Q and $\tilde{Q}$, and the received reward, the temporal difference (δ) is calculated according to Equation 6.

The agent's knowledge base is updated by using the temporal difference and eligibility trace values (see line 16). The update procedure of eligibility traces is as follows. For each state-action pair in the eligibility set, $\mathcal{E}_{tm}$, the value of the corresponding eligibility trace is updated by Equation 7. However, if the decayed trace rate is less than a pre-defined threshold, the state-action pair is considered ineligible and is removed from the training set (see lines 19 – 21).

At the end of the learning process, the previous action is rechecked. If it is a termination action, the current episode terminates, and the traffic light indication associated with TA turns to yellow. The current state and action and the previous state and action are set to null. The training set is cleared, and the agent becomes inactive (see lines 25 – 27). Conversely, if the previous action is not a termination, the current state-action pair is stored in the set of eligibility traces. Then, the previous state-action pair is updated by changing to the current state-action pair (see lines 29 – 30).

The required amount of time taken by most functions in Fig. 4 remains the same regardless of the input data size except for two functions, *function_approximator* and *update_Q_values*. Since KNN is the function approximator and the training set for KNN is the set of eligibility traces, the time required is directly proportional to $K|\mathcal{E}_i||\mathcal{U}_i|$ for turning movement i, where K denotes the pre-determined size of the set of the nearest neighbors. To update the Q values, the time required is proportional to $|\mathcal{E}_i|$. It is common that most TAs in a signal controller are inactive at any time of a cycle since turning movements associated with the north-south direction are normally in conflict with those in the west-east direction. Thus, the running time of the knowledge update algorithm for a signal controller is proportional to $|\mathcal{S}^{tm}|/2$ by assuming that a half of all turning movements are active. Thus, the asymptotic time complexity of one-step update, T_C, can be estimated as follows:

$$T_C \approx O(K \log(|\mathcal{S}^{tm}|)|\mathcal{E}_i||\mathcal{U}_i|). \tag{12}$$

The system is decentralized meaning that each signal controller operates individually and its operation processes can be performed in a parallel computing environment. By taking into account the number of learning iterations and the step size in each iteration, the time complexity of a signal controller in the "off-line training" mode is

$$T_C^{off} \approx O(NTK \log(|\mathcal{S}^{tm}|)|\mathcal{E}_i||\mathcal{U}_i|), \tag{13}$$

and the counterpart of the "on-line operation" mode remains as T_C. The values of both the number of turning movements in a signal controller, $|\mathcal{S}^{tm}|$, and the size of the set of eligibility traces, $|\mathcal{E}_i|$, are on the order of magnitude of 10 in this application. K and $|\mathcal{U}_i|$ are user-defined and usually are below 10. Consequently, the "on-line operation" mode can be achieved in real time.

6. Case study

6.1. Experiment setup

The proposed decentralized TLC system was tested in a region of three connected intersections at the 226 road (Huddingevägen) in Stockholm. Fig. 5 presents the intersection layouts and the corresponding turning movements. For convenience, in the following descriptions, Huddingevägen-Lännavägen, Huddingevägen-Björkängsvägen and Huddingevägen-Ågestavägen are marked as I1, I2, and I3, respectively. The distances between two adjacent intersections, I1 and I2, and I2 and I3, are 360 and 750 meters, respectively. For the turning movements that share the lane, the actions of the agents are restricted to be identical. In addition, bicycle and pedestrian signals are not taken into account in this case study.

There are two detectors, a long detector and a short detector, being placed in a lane. The detector configuration is derived from the deployed Swedish detection system. The existing detection system in Sweden is placed to facilitate a vehicle actuated (VA) control system generating signal timing according to the detection of vehicle presence. VA makes timing actions following a logic-based algorithm without on-line adaption, namely a gap-seeking algorithm, by taking into account the current traffic condition associated with the signal component (e.g., a signal group or a phase).

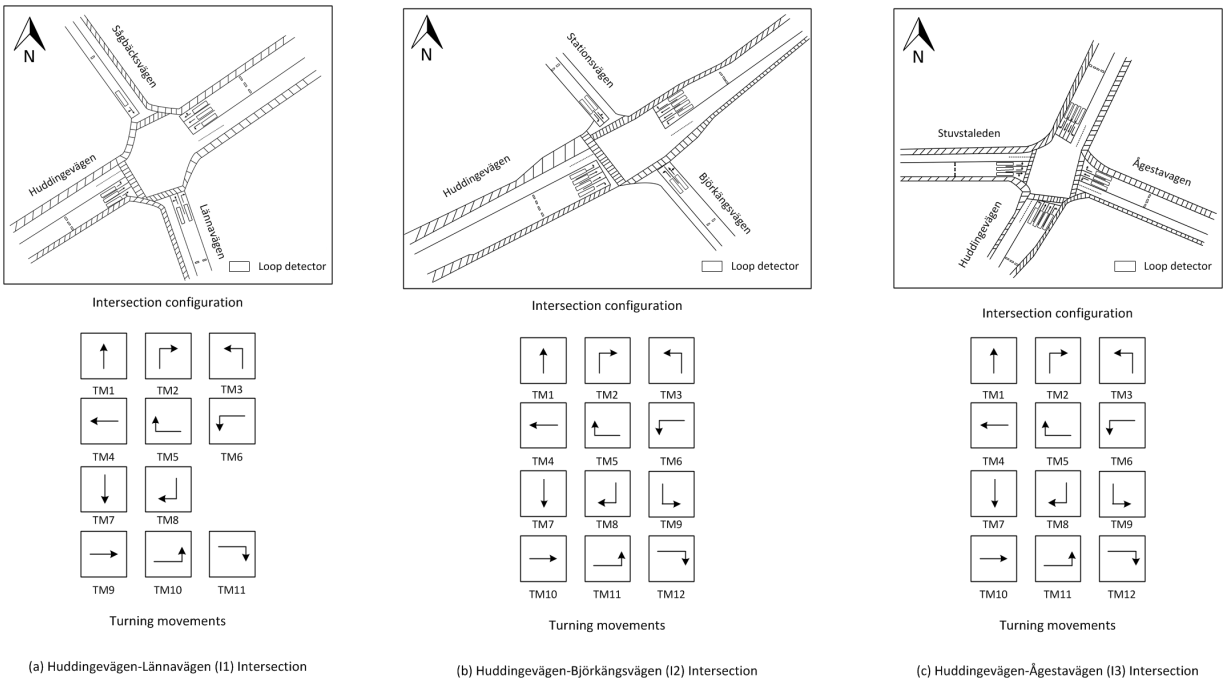

Fig. 5. Intersection layouts of the case study and the corresponding turning movements

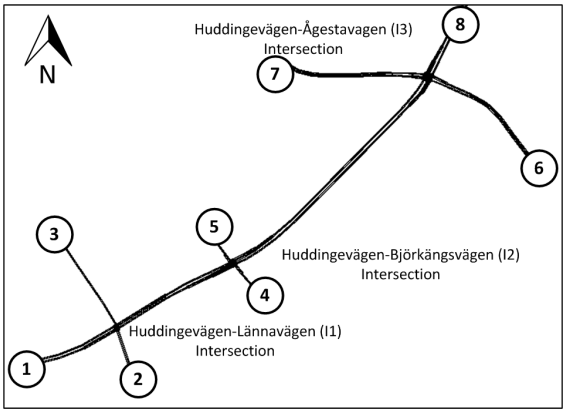

Fig. 6. SUMO simulation model for the applied three intersections and node identifications.

A simulation model was built in SUMO (Simulation of Urban Mobility) version 0.19.0 Krajzewicz et al. (2012) for the applied network, and its GUI (graphical user interface) is shown in Fig. 6. SUMO is an open-source software and implements discrete-time microscopic traffic simulation models, including models of traffic networks, models of road infrastructures (such as deployed sensors and traffic signal control), and driving behavior models (i.e., a collision-free car-following model Krauss et al. (1997) and a four-layered lane-changing model Erdmann (2015)). Compared to other simulation tools, SUMO provides the advantage of applying a portable API interface (TraCI) through which external programs are easily executed during a simulation while interacting with the SUMO models.

A self-developed software program was implemented to connect with the SUMO simulator so that the proposed TLC system could govern the traffic lights in the simulation. Moreover, SUMO simultaneously sends the detection information to the signal program. For computing the immediate rewards, the instantaneous vehicle information is accessible from the SUMO simulator via the TraCI interface.

Table 3
The learning and control parameters for TAs used in the experiments

Control parameter	Description (Unit)	Value
α	Learning rate (–)	0.85
γ	Discount rate (–)	0.9
ϵ	Initial exploration rate (–)	0.9
τ	Decay of exploration rate (–)	0.8
λ	Trace decay rate (–)	0.8
g^{up}	The upper bound of green time (seconds)	50
g^{low}	The lower bound of green time (seconds)	5
y	Yellow time(seconds)	3

A base traffic demand matrix infers the hourly average traffic flows among all OD pairs as follow:

$$D = \begin{bmatrix} 0 & 100 & 100 & 100 & 100 & 100 & 100 & 400 \\ 100 & 0 & 200 & 0 & 0 & 0 & 0 & 100 \\ 100 & 200 & 0 & 0 & 0 & 0 & 0 & 0 \\ 100 & 0 & 0 & 0 & 200 & 0 & 0 & 100 \\ 100 & 0 & 0 & 200 & 0 & 0 & 0 & 100 \\ 100 & 0 & 0 & 0 & 0 & 0 & 200 & 100 \\ 100 & 0 & 0 & 0 & 0 & 200 & 0 & 100 \\ 400 & 100 & 100 & 100 & 100 & 100 & 100 & 0 \end{bmatrix}. \tag{14}$$

The index of the matrix corresponds to the node number shown in Fig. 6. The demand matrix is used for the off-line training process to establish the decentralized traffic light control system by running 100 simulation iterations, where the length of one simulation run is 30 minutes. According to the flow matrix, vehicles are generated randomly at each node. The number of departures in a given time interval follows the Poisson distribution. So the time between two successive vehicles driving from node A to note B follows the negative exponential distribution with a mean value of $1/D_{A,B}$.

In the following experiments, thirty simulation runs were performed to make the final analysis statistically significant. Each simulation runs for one hour and fifteen minutes, including a warm-up time of fifteen minutes in the beginning to avoid initial loading effects. We evaluate both local and regional effects when the proposed decentralized TLC system is employed in the network.

6.2. Agent design

In the experiments, TAs are homogeneous, meaning that the learning parameters along with state, action, and reward definitions are identical. The learning parameters were tested with their sensitivity and the selected ones generating appropriate learning efficiency are summarized in Table 3. The conflict matrices for the three intersections are given below.

$$C_1 = \begin{bmatrix} 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 0 & 2 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 0 & 0 & 0 \end{bmatrix}, C_2 = C_3 = \begin{bmatrix} 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 2 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 \\ 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 \\ 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 \\ 2 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 \\ 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 \\ 2 & 2 & 2 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 0 \end{bmatrix}, \tag{15}$$

where C_1, C_2, and C_3 refer to the conflict matrices for intersections I1, I2, and I3, respectively. In Table 3, the upper and lower bounds of green signal time of the a TA correspond to the instructions from the direct IA to restrict its actions. Specifically, a minimum green signal time is required before the agent is ordered to terminate, and a maximum green time is defined to limit the authorization of green signal extension. These two parameters can also be adaptively changed in real-time by an optimization approach, but they are determined in advance in this study according to the engineering practice.

The definitions of state, action, and reward are similar to those in Jin and Ma (2017) which are based on the Swedish loop detector system. In addition to considering the detection information reported by the neighboring agents (Jin and Ma, 2017), the state definition of an agent is based on of the detection information at the whole intersection.

In the detection system, a digital pulse signal is sent to the controller by the loop detector if and only if the loop detector is occupied. Based on this principle, four elements are defined in the state vector of a TA. The signal controller reports another state elements, the elapsed green signal time. The state vector of TA i at time t is represented by

$$\mathbf{x}_{i,t} = [g_{i,t}^{ela}, det_{i,t}^{short}, det_{i,t}^{long}, o_{i,t}, v_{i,t}]^T, \tag{16}$$

where $g_{i,t}^{ela}$ refers to the elapsed green time, $det_{i,t}^{short}$ and $det_{i,t}^{long}$ represent the time duration since the last detection sent by the short detector and by the long detector, respectively. $o_{i,t}$ denotes the relative time occupancy and $v_{i,t}$ denotes the relative traffic flow. $o_{i,t}$ is computed by

$$o_{i,t} = \frac{o_{i,t}^{own}}{\bar{o}_{i,t}^{others}}, \tag{17}$$

where $o_{i,t}^{own}$ denotes the time occupancy associated with the short detector in the lane that is controlled by the TA i and $\bar{o}_{i,t}^{others}$ denotes the average time occupancy associated with other short detectors at the intersection. Similarly, the computation of $v_{i,t}$ is given by

$$v_{i,t} = \frac{f_{i,t}^{own}}{\bar{f}_{i,t}^{others}} \tag{18}$$

where $f_{i,t}^{own}$ and $\bar{f}_{i,t}^{others}$ denotes the detected traffic flow for the TA and for the others, respectively. $g_{i,t}^{ela}$, $det_{i,t}^{short}$ and $det_{i,t}^{long}$ are shorted-sighted states indicating the instant traffic condition. Whereas, $o_{i,t}$ and $v_{i,t}$ represent long-term system states, which suggest changes in traffic patterns.

At each action point, a TA decides either to extend for additional green signal time or to terminate. For the extension action, the agent chooses a valid integer value. As mentioned, the action is also restricted by bounds, supplied by the IA. Thus, the action can be analytically represented by assigning a green signal extension, $g_{i,t}^{ela}$, to an integer value between 0 and g^{max}, where $g_{i,t}^{ext} = 0$ means that the agent i is ordered to terminate, i.e.,

$$\mathbf{u}_{i,t} = [g_{i,t}^{ext}], \quad g_{i,t}^{ext} \in \{g | 0 \le g \le g^{max} \text{ and } g^{low} \le g_{i,t}^{ela} + g \le g^{up}; g \in \mathcal{Z}\}, \tag{19}$$

where $\mathcal{Z}$ represents the set of integers, and, in this paper, $g^{max} = 5$.

Since the proposed traffic light controller's goal is to improve traffic mobility efficiency, the reward function is related to average travel delay of vehicles. All the TAs of an intersection have a common reward value. The reward function is, therefore, represented by

$$r_{1,t} = r_{2,t} = \cdots = r_{n,t} = -\bar{d}_t, \tag{20}$$

where $\bar{d}_t$ denotes the weighted average of travel delays for all vehicles at the intersection.

To compute $\bar{d}_t$, the travel delay is weighted taking into account turning movement priorities. The priority weight of a turning movement is a part of the instructions given by the direct IA resulting from the communication information among the neighboring IAs. A TA with a relatively high value of the weight is promoted for improving traffic mobility for this turning movement. In the paper, all TAs share the same reward function. Let vehicles, associated with the turning movement i, be indexed with $1, 2, \ldots, n_{i,t}^v$. The weighted average travel delay is formulated as

$$\bar{d}_t = \sum_{i=1}^{n} \phi_i \sum_{\beta=1}^{n_{i,t}^v} \frac{d_{\beta,t}}{n_{i,t}^v}, \tag{21}$$

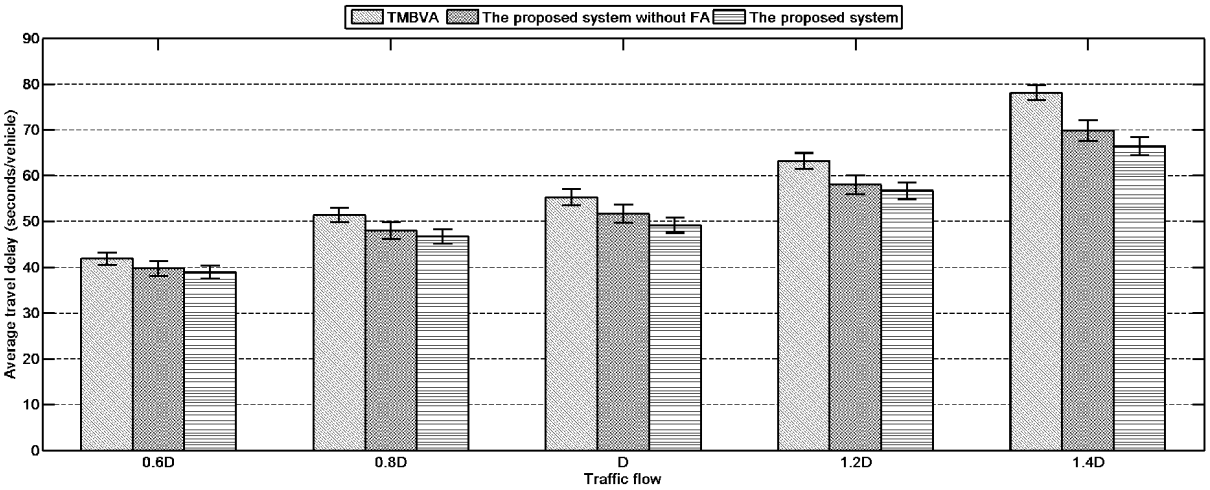

Fig. 7. The comparison results reported by TMBVA, the proposed decentralized system without function approximation and the proposed decentralized system in terms of average travel delay [seconds/vehicle] per intersection

where ϕ_i denotes the priority weight of TA i, and $d_{\beta,t}$ denotes the travel delay for vehicle β between $t-1$ and t.

Travel delay is defined by the difference between the actual travel time and ideal travel time to complete a journey. The ideal travel time is the time that a vehicle spends on completing the journey at the desired speed. The travel delay for vehicle β is calculated by

$$d_{\beta,t} = \Delta t_{\beta,t} - \frac{l_{\beta,t}}{v_{\beta,t}^{des}}, \tag{22}$$

where $l_{\beta,t}$ denotes the traveled distance within the time interval $\Delta t_{\beta,t}$, and $v_{\beta,t}^{des}$ refers to the desired speed of vehicle β.

6.3. Local effect analysis

In the following local effect analysis, a turning movement based vehicle actuated control (TMBVA) system is used as a benchmark signal control system for comparison. The controller applies the same phasing logic as the proposed control system with the primary difference being only in the timing scheme for intersection control. The maximum green times of the TMBVA system are the tuning parameters which are optimized when travel delay is set as the objective function. The optimal signal control parameters are generated using a genetic algorithm based optimization framework. The descriptions of VA timing and the optimization framework can be found in Jin et al. (2017b). In addition to comparing with the optimized TMBVA system, the analysis examines the performance of the proposed system without using FA.

Fig. 7 summarizes the average travel delay results for the three intersections with deployments of the TMBVA system, the proposed system without FA, and the proposed system. Five different traffic flow scenarios, varying from $0.6D$ to $1.4D$, were applied to each traffic light controller in the experiments. The intelligent timing scheme has potential to improve traffic mobility by comparing the travel delay results with the TMBVA system. In particular, the trend appears more obvious when traffic flow becomes high. For example, an increased reduction of 4% (from 11% to 15%) in travel delay was achieved by replacing the TMBVA system with the proposed system when the applied traffic demand matrix changed from D to $1.4D$.

In addition, the results show that the proposed system was slightly enhanced when incorporating FA in terms of reducing the average travel delay of vehicles. This finding is in line with the benefits brought by KNN when SARSA learning with multiple-step backups was applied. The agent used FA to estimate the knowledge for the state-action pairs that were not exhausted by exploring the ever acquired knowledge from the previous learning samples. However,

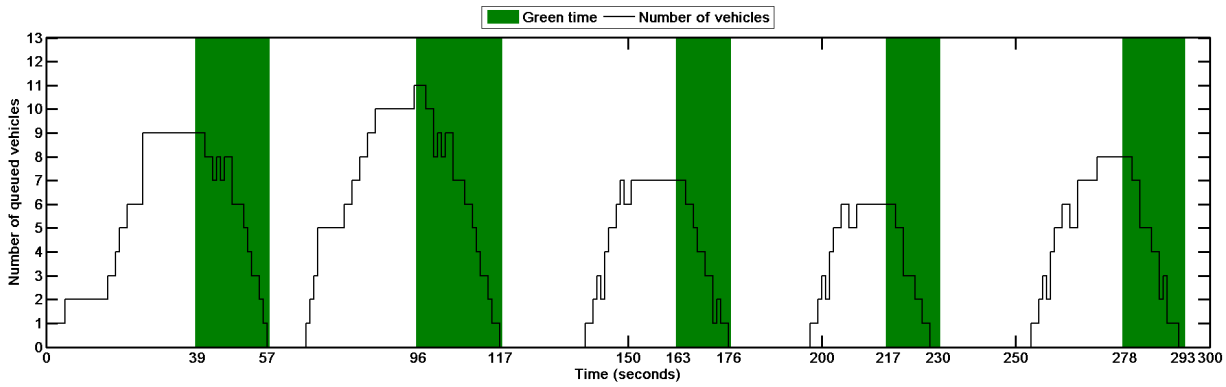

Fig. 8. The change of number of vehicles together with the change of green time for turning movement TM9 at intersection I1

Table 4
Performance measures per intersection with different priority weights for the coordinated turning movements

Experiment description	Average travel delay per intersection (seconds/vehicle)	Average number of stops (stops/100 vehicles)
$\phi_1^{I1} = \phi_2^{I1} = \cdots = \phi_{11}^{I1} = 1, \phi_1^{I2} = \phi_2^{I2} = \cdots = \phi_{12}^{I2} = 1,$ $\phi_1^{I3} = \phi_2^{I3} = \cdots = \phi_{12}^{I3} = 1$	49.20 ± 1.65	178.96 ± 8.34
$\phi_4^{I1} = \phi_9^{I1} = 2, \phi_4^{I2} = \phi_{10}^{I2} = 2, \phi_1^{I3} = \phi_7^{I3} = 2$	45.39 ± 1.47	164.22 ± 7.47
$\phi_4^{I1} = \phi_9^{I1} = 5, \phi_4^{I2} = \phi_{10}^{I2} = 5, \phi_1^{I3} = \phi_7^{I3} = 5$	43.54 ± 1.33	155.88 ± 6.69
$\phi_4^{I1} = \phi_9^{I1} = 10, \phi_4^{I2} = \phi_{10}^{I2} = 10, \phi_1^{I3} = \phi_7^{I3} = 10$	44.97 ± 1.41	157.03 ± 7.18

Note: ϕ_i^q represents the weight of turning movement i at intersection q; In rows $2-4$, $\phi_i^q = 1$ for the turning movements that are not mentioned.

the agent without FA enabled was likely to perform a random action based on the implemented action policy, which is risky as it may lead to a deterioration in system performance.

One important goal of the TLC system is to dissolve queues at the controlled intersection efficiently. To provide some insight into the operation of the proposed system, Fig. 8 depicts the changes in green allocations along with the number of vehicles with zero speed in the lanes. These results were obtained within a randomly selected 5 minutes simulation period. In the figure, a solid black line represents the change of the average number of queued vehicles over time in the lanes controlled by TM9 at intersection I1. The green area represents the green signal period for the agent.

The proposed system is capable of allocating longer green time when more vehicles enter the controlled lane. Specifically, the green length of the agent was 18 seconds in the first cycle of the selected period when the queue is accumulated to nine vehicles. So the TA agent allocated three more seconds to allow the dissipation of a queue of 11 vehicles in the next cycle. In summary, from the results of the selected five cycles, approximately a 2-second green time on average is assigned to each vehicle in the queue. So, the system is queue responsive after the machine learning process, even though the states are quantified using measurements of loop detectors.

6.4. Regional effect analysis

As previously discussed, this study emphasizes the promotion of the "green wave" scenario in a region with coordinated intersections. The set of priority weights of turning movements is part of the instructing action of an IA that advises the TAs in the lower level of the system's hierarchy. Additional experiments were conducted for testing the effects of priority weights, in which the Huddingevägen street is the arterial road, and relative high priority weights were assigned to the associated turning movements.

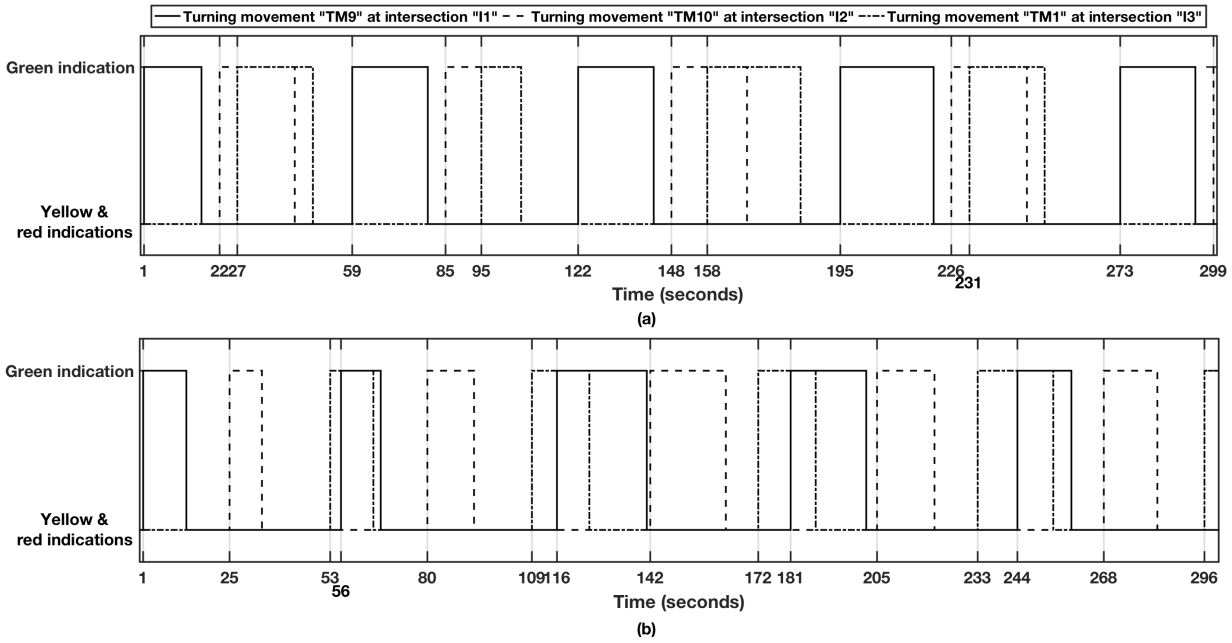

Fig. 9. The switching time of traffic light indications for TM9 at intersection I1, TM10 at intersection I2, and TM1 at intersection I3

Table 4 presents a sensitivity analysis with four sets of priority weights for the turning movements, and the performance measures are the average travel delay per intersection and the average number of stops for vehicles driving through the three intersections. The system is improved by reducing average delay and the number of stops when the coordinated turning movements are given a high priority. An example is that the priority weights of the turning movements are set to five whereas the others are one. Vehicles on average can reduce nearly 13% number of stops when driving through the three intersections in comparison to the case that the priority weights are identically set to one. However, a trade-off appears between the benefits of coordinating signals on the direction of the arterial road and the performance of the entire region. If the priority weights are set as high as 10 for the coordinated turning movements, both average delay and number of stops start to increase. It is, therefore, important to determine a proper set of the priority weights for the turning movements in regional operation.

In traffic engineering practice, an offset value is usually set to facilitate the "green wave" scenario. Here, the offset refers to the beginning time of the coordinated turning movements in relation to a pre-defined reference intersection. If the offset is properly set, a platoon of waiting vehicles can smoothly pass through several coordinated intersections with their desired speeds after the traffic light indications turn to green at the reference intersection. Let us assume intersection I1 is the reference intersection, and the desired speed of vehicles is 50 km/h in the study. Then according to the distance between two adjacent intersections, the appropriate offset values for intersection I2 and I3 should be approximately 25 and 53 seconds, respectively.

Fig. 9 demonstrates the switch times of green indications for the coordinated turning movements at the three intersections. In Fig. 9a, the priority weights are set to one for all turning movements whereas the weights are equal to 5 for the coordinated turning movements in Fig. 9b. The beginnings of green indications for the coordinated turning movements are marked in the plots. There is no clear coordination seen for the corresponding turning movements when all the priority weights are identically one. In the second case, the offset values, though with small fluctuation, approximately match the theoretical values in Fig. 9b. This indicates that, by learning, TAs succeed in reinforcing their signal operations to achieve the goal of generating the "green wave" scenario.

7. Conclusions

This paper proposes an intelligent traffic light control system to tackle the challenges caused by the rising traffic congestion. The system is operated using a hierarchical multi-agent modeling framework where three levels of the hierarchy, including RA, IA, and TA, are introduced. An agent is normally instructed by the agent at the next higher level according to applied control strategies while communication is enabled between agents at the same level.

This study extends the previous work on learning-based adaptive intersection control using a group-based phasing strategy. The local control is enhanced by the phasing scheme based on turning movement. The phasing control is logic-based and combines turning movements into phases dynamically. In accordance with the previous paper, signal timing is formulated as an optimal control problem and solved by an RL algorithm. FA based on a KNN approach is introduced to enhance the computational efficiency of the learning algorithm.

Another major contribution is the extension of the local intelligent control to network operation. Generating "green wave" scenario is an important traffic engineering practice for coordination between several intersections of an arterial road. In the development, such a scenario is achieved by assigning different priority weights to the turning movements within a region and hence implement coordination operation associated with certain directions in a region.

A collective learning process with two operational modes, "off-line training" and "on-line operation," is involved in the learning of the signal controller. The TAs individually learn the knowledge and jointly generate collective behaviors for intersection control. Signal controllers in one or several regions may operate and learn to respond to dynamic traffic patterns simultaneously.

A case study on a Swedish road network, consisting of three intersections, was simulated using an open-source microscopic traffic simulator, SUMO. Both local and regional effects are analyzed by traffic simulations. Based on the computational experiments, three major findings are concluded:

- The proposed decentralized signal control system has the potential of mitigating traffic congestion by reducing travel delays compared to the TMBVA system;
- The decentralized control system is queue responsive and adapts the green time allocation to the change of traffic patterns;
- Coordination, especially the "green wave" scenario, is generated by changing the operating priorities of associated turning movements, even though the system design is purely decentralized.

Although signal coordination can be implemented by carefully adjusting the operation priorities of turning movements in our case study, it is still an open question on how to optimally set the priority weights, especially when numerous intersections are involved in the studied network. The complexity of such problem grows exponentially since the dimension of the system state grows proportional to the increased number of intersections. To make the TLC system scalable, an efficient learning scheme, such as deep RL approaches (Li, 2017), is required for handling the high-dimensionality issue. This will be treated in our future research.

Acknowledgments

This work is partially supported by J. Gust. Richert Foundation (2015-00205) and Chinese Scholarship Council (#201407930010). The financial support is greatly acknowledged.

References

Abdoos, M., Mozayani, N., Bazzan, A. L., 2013. Holonic multi-agent system for traffic signals control. Engineering Applications of Artificial Intelligence 26 (5), 1575–1587.

Abdoos, M., Mozayani, N., Bazzan, A. L., 2014. Hierarchical control of traffic signals using q-learning with tile coding. Applied intelligence 40 (2), 201–213.

Arel, I., Liu, C., Urbanik, T., Kohls, A., 2010. Reinforcement learning-based multi-agent system for network traffic signal control. IET Intelligent Transport Systems 4 (2), 128–135.

Balaji, P., German, X., Srinivasan, D., 2010. Urban traffic signal control using reinforcement learning agents. IET Intelligent Transport Systems 4 (3), 177–188.

Barto, A. G., 1998. Reinforcement Learning: An Introduction. MIT press.

Bazzan, A. L., de Oliveira, D., da Silva, B. C., 2010. Learning in groups of traffic signals. Engineering Applications of Artificial Intelligence 23 (4), 560–568.

Boillot, F., Midenet, S., Pierrelée, J.-C., 2006. The real-time urban traffic control system cronos: Algorithm and experiments. Transportation Research Part C: Emerging Technologies 14 (1), 18–38.

Cai, C., Wong, C. K., Heydecker, B. G., 2009. Adaptive traffic signal control using approximate dynamic programming. Transportation Research Part C: Emerging Technologies 17 (5), 456–474.

Cools, S.-B., Gershenson, C., D'Hooghe, B., 2013. Self-organizing traffic lights: A realistic simulation. In: Advances in Applied Self-Organizing Systems. Springer, pp. 45–55.

El-Tantawy, S., Abdulhai, B., 2013. Towards multi-agent reinforcement learning for integrated network of optimal traffic controllers (marlin-otc). Transportation Letters.

El-Tantawy, S., Abdulhai, B., Abdelgawad, H., 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), 1140–1150.

Erdmann, J., 2015. Sumo's lane-changing model. In: Modeling Mobility with Open Data. Springer, pp. 105–123.

Hunt, P., Robertson, D., Bretherton, R., Royle, M. C., 1982. The SCOOT on-line traffic signal optimisation technique. Traffic Engineering & Control 23 (4).

Jin, J., Ma, X., 2015a. Adaptive group-based signal control by reinforcement learning. Transportaion Research Procedia 10, 207–216.

Jin, J., Ma, X., 2015b. Adaptive group-based signal control using reinforcement learning with eligibility traces. Proceedings of the 18th International IEEE Conference on Intelligent Transportation Systems (ITSC).

Jin, J., Ma, X., 2016. A learning-based adaptive group-based signal control system under oversaturated conditions. IFAC-PapersOnLine 49 (5), 291–296.

Jin, J., Ma, X., 2017. A group-based traffic signal control with adaptive learning ability. Engineering Applications of Artificial Intelligence 65, 282–293.

Jin, J., Ma, X., Kosonen, I., 2017a. An intelligent control system for traffic lights with simulation-based evaluation. Control Engineering Practice 58, 24–33.

Jin, J., Ma, X., Kosonen, I., 2017b. A stochastic optimization framework for road traffic controls based on evolutionary algorithms and traffic simulation. Advances in Engineering Software, in press.

Khamis, M. A., Gomaa, W., 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, 134–151.

Krajzewicz, D., Erdmann, J., Behrisch, M., Bieker, L., 2012. Recent development and applications of SUMO–simulation of urban mobility. International Journal On Advances in Systems and Measurements 5 (3 and 4), 128–138.

Krauss, S., Wagner, P., Gawron, C., 1997. Metastable states in a microscopic model of traffic flow. Physical Review E 55 (5), 5597.

Li, Y., 2017. Deep reinforcement learning: An overview. arXiv preprint arXiv:1701.07274.

Ma, X., Jin, J., Lei, W., 2014. Multi-criteria analysis of optimal signal plans using microscopic traffic models. Transportation Research Part D: Transport and Environment 32, 1–14.

McCallum, R. A., Tesauro, G., Touretzky, D., Leen, T., 1995. Instance-based state identification for reinforcement learning. Advances in Neural Information Processing Systems, 377–384.

Mirchandani, P., Head, L., 2001. A real-time traffic signal control system: architecture, algorithms, and analysis. Transportation Research Part C: Emerging Technologies 9 (6), 415–432.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.

Prashanth, L., Bhatnagar, S., 2011. Reinforcement learning with function approximation for traffic signal control. IEEE Transactions on Intelligent Transportation Systems 12 (2), 412–421.

Sims, A. G., Dobinson, K. W., 1980. The Sydney coordinated adaptive traffic (SCAT) system philosophy and benefits. IEEE Transactions on Vehicular Technology 29 (2), 130–137.

Thorpe, T. L., Anderson, C. W., 1996. Traffic light control using SARSA with three state representations. Tech. rep., Citeseer.

Wong, C., Wong, S., 2003. Lane-based optimization of signal timings for isolated junctions. Transportation Research Part B: Methodological 37 (1), 63–84.