Future Generation Computer Systems 109 (2020) 431–445

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Deep reinforcement learning for traffic signal control under disturbances: A case study on Sunway city, Malaysia

Faizan Rasheed, Kok-Lim Alvin Yau *, Yeh-Ching Low

Department of Computing and Information Systems, Sunway University, Subang Jaya 47500, Malaysia

ARTICLE INFO

Article history:
Received 25 October 2019
Received in revised form 7 February 2020
Accepted 31 March 2020
Available online 9 April 2020

Keywords:
Reinforcement learning
Deep reinforcement learning
Multi-agent reinforcement learning
Deep Q-network
Multi-agent deep Q-network
Traffic signal control

ABSTRACT

In most urban areas, traffic congestion is a vexing, complex and growing issue day by day. Reinforcement learning (RL) enables a single decision maker (or an agent) to learn and make optimal actions in an independent manner, while multi-agent reinforcement learning (MARL) enables multiple agents to exchange knowledge, learn, and make optimal joint actions in a collaborative manner. The integration of the newly emerging deep learning and the traditional RL approach has created an advanced technique called deep Q-network (DQN) that has shown promising results in solving high-dimensional and complex problems, including traffic congestion. In this paper, DQN is embedded in traffic signal control to solve traffic congestion issue, which has been plagued with the curse of dimensionality whereby the representation of the operating environment can be highly dimensional and complex when the traditional RL approach is used. Most importantly, this paper proposes multi-agent DQN (MADQN) and investigates its use to further address the curse of dimensionality under traffic network scenarios with high traffic volume and disturbances. To investigate the effectiveness of our proposed scheme, a case study based on an urban area, namely Sunway city in Malaysia, is conducted. We evaluate our scheme via simulation using a traffic network simulator called simulation of urban mobility (SUMO) and a simulation tool called MATLAB. Simulation results show that our proposed scheme reduces the total travel time of the vehicles.

1. Introduction

Traffic congestion is a serious and growing problem in most urban areas. Traffic signal controllers are installed at intersections to monitor traffic flows and alleviate traffic congestion. Generally speaking, a traffic signal controller can be characterized by: (a) *signal color* (i.e., red indicates "stop", yellow indicates "slow down", and green indicates "go"), (b) *traffic phase*, which represents a combination of green signals allocated to all lanes of an intersection simultaneously for safe and non-conflicting traffic flows (see Fig. 1), and (c) *traffic phase split*, which represents the time interval of a traffic phase. Unexpected traffic disturbances, such as rainfall or bad weather conditions, can increase traffic congestion. Meanwhile, Poisson process has been widely used in the literature to model the arrival of vehicles in which the vehicles' inter-arrival times are assumed to follow the exponential distribution [1], but the Poisson process does not incorporate traffic disturbances. This paper adopts the Burr distribution that generalizes the Poisson process using non-exponential distribution to model the inter-arrival time of vehicles under scenarios with high traffic volume and disturbances.

Traditional traffic signal controllers select traffic phases and traffic phase splits using three main approaches. *Firstly*, both traffic phases and traffic phase splits are deterministic in nature [2]. Specifically, a series of traffic phases are executed in a round-robin fashion with certain periods of traffic phase splits. *Secondly*, traffic phases are deterministic, however the traffic phase splits are dynamically adjusted based on short-term information, particularly the presence or absence of vehicle(s) at a lane [3]. *Thirdly*, both traffic phases and traffic phase splits are dynamic in nature. Similar to the second approach, the difference is that the traffic phase splits are dynamically adjusted based on long-term information, such as the waiting time and the queue length of vehicles at a lane. The third approach has commonly been accomplished using reinforcement learning (RL), which is an artificial intelligence approach [4]. RL possesses the capability to learn the relationships between actions and their effects on the operating environment (or states), and so it can adapt to the real-time changes of traffic flows. There are two main approaches in RL, namely the traditional single-agent approach (called RL for simplicity) and the multi-agent approach (called multi-agent reinforcement learning, or MARL). RL enables a single decision maker (or agent) to learn and make optimal action in an independent manner, while MARL enables multiple agents to exchange knowledge, learn, and make optimal joint action in a collaborative

* Corresponding author.
E-mail address: koklimy@sunway.edu.my (K.-L.A. Yau).

https://doi.org/10.1016/j.future.2020.03.065
0167-739X/

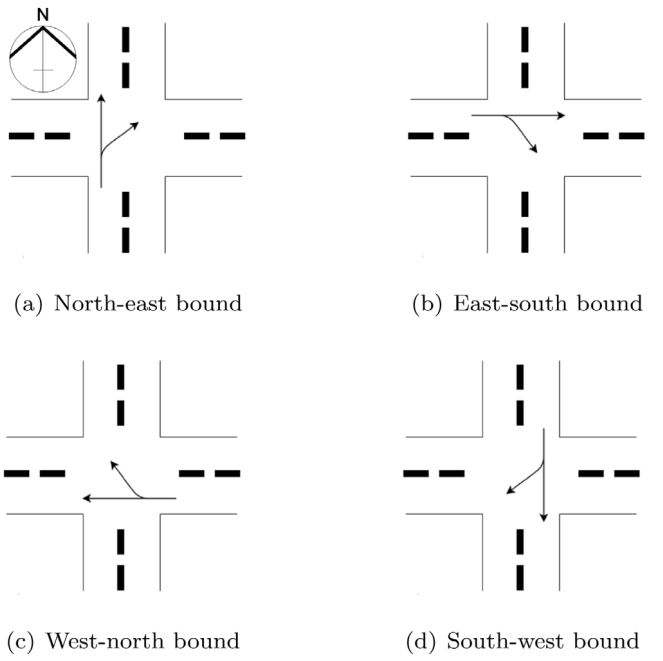


Fig. 1. A series of traffic phases of a traffic signal controller at an intersection.

manner [5]. Nevertheless, RL and MARL are marred by the curse of dimensionality, an issue whereby the number of states (or the state space) becomes too large, leading to two main shortcomings [6]. *Firstly*, higher computational cost and longer learning time are required to explore all state–action pairs in order to identify the optimal actions. *Secondly*, larger storage capacity is required to store knowledge (or Q-values). Traffic disturbances can worsen the curse of dimensionality because more factors, such as rainfall and flash flood, must be incorporated into the state representation.

This paper adopts deep Q-network (DQN) [7], which is based on RL and deep learning [8], to address the curse of dimensionality. In DQN, the use of artificial neural network (ANN) [9] in deep learning provides: (a) a continuous representation of state space, and so there are unlimited number of state–action pairs, and (b) efficient storage as several layers of neurons are used to provide abstract representations of high-dimensional and complex input, which is the state representation, and so the curse of dimensionality is addressed.

1.1. Our contributions

There are two main contributions in this paper as follows:

- Traffic signal controllers based on multi-agent DQN (MADQN), which is a novel approach that extends the traditional single-agent DQN. While the traditional DQN addresses the curse of dimensionality under scenarios with high traffic volume and disturbances, MADQN enables the traffic signal controllers to exchange knowledge, learn, and make optimal joint actions in a collaborative manner. A case study based on an urban area, namely Sunway city in Malaysia, is conducted to investigate the effectiveness of the proposed scheme.
- Traffic signal controllers that take account of traffic disturbances (i.e., irregular inter-arrival time of vehicles at a lane due to heavy rainfall) using a traffic model based on the Burr type XII distribution, which has not been investigated in the literature.

1.2. Organization of the paper

The rest of this paper is organized as follows. Section 2 presents the background of RL, MARL, and DQN. Section 3 presents related work. Section 4 presents our proposed model for traffic signal controller, and the MADQN algorithm for the proposed model. Section 5 presents a case study based on Sunway city. Section 6 presents simulation results and discussion. Finally, Section 7 presents conclusions and future work.

2. Background

This section presents an overview of RL, MARL, and DQN.

2.1. Reinforcement learning

Traditional RL model enables a decision maker (or an agent) to explore and exploit different state–action pairs so that it receives the lowest possible cost (or the highest possible reward) for system performance enhancement as time goes by $t = 1, 2, 3, \ldots$ [10]. Fig. 2 presents an RL agent, and Algorithm 1 presents the RL algorithm. There are three main representations: (a) *state* set S represents the decision making factors in the operating environment, (b) *action* set A, and (c) *delayed reward* (or *delayed cost*) $r_{t+1}(s_{t+1})$ represents the appropriateness of a state–action pair (s_t, a_t) that leads to next state s_{t+1}. At time instant t, an agent observes the current state $s_t \in S$, and selects an *action* $a_t \in A$ (see Fig. 2(a)). Subsequently, at time instant $t+1$, the agent receives a delayed reward $r_{t+1}(s_{t+1})$ for the state–action pair (s_t, a_t) under the next state $s_{t+1} \in S$ (see Fig. 2(b)), and updates Q-value $Q_t(s_t, a_t)$ for the state–action pair, which represents knowledge. The Q-value $Q_t(s_t, a_t)$ represents the appropriateness of taking action a_t under state s_t, and it is updated using Q-function as follows [11]:

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

where $0 \leq \alpha \leq 1$ is the learning rate, and $\delta_t(s_t, a_t)$ is the temporal difference, which is based on the Bellman equation, that represents the difference of rewards, in terms of delayed and discounted rewards, between two successive estimations as follows [12]:

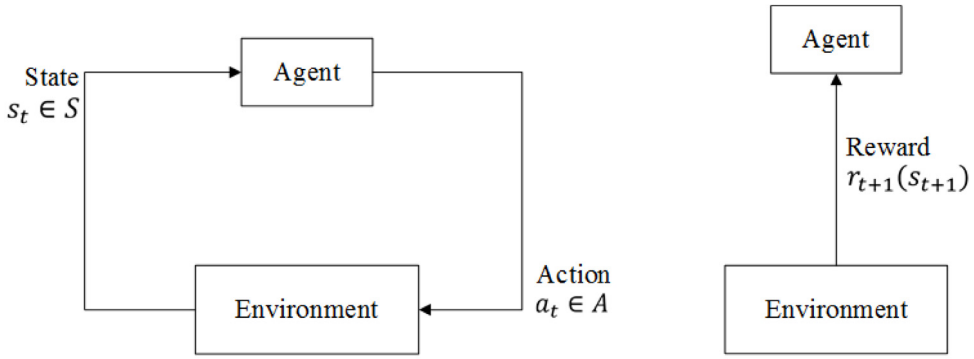
$$\delta_t(s_t, a_t) = r_{t+1}(s_{t+1}) + \gamma \max_{a \in A} Q_t(s_{t+1}, a) - Q_t(s_t, a_t) \tag{2}$$

where $0 \leq \gamma \leq 1$ represents a discount factor that shows the preference for the discounted reward, and $\gamma \max_{a \in A} Q_t(s_{t+1}, a)$ represents the discounted reward, which shows the expected maximum Q-value at time $t+1$ and so on. In other words, the delayed reward $r_{t+1}(s_{t+1})$ represents a short-term reward, while the discounted reward $\gamma \max_{a \in A} Q_t(s_{t+1}, a)$ represents a long-term reward. As time goes by $t = 1, 2, 3, \ldots$, the agent explores all the state–action pairs (s_t, a_t), as well as updates and stores their respective Q-values $Q_t(s_{t+1}, a)$ in a two-dimensional Q-table.

Algorithm 1 Traditional RL algorithm embedded at an agent

1: **Procedure**
2: observe current state $s_t \in S$
3: select action $a_t \in A$ using Equation (3)
4: receive delayed reward $r_{t+1}(s_{t+1})$
5: update Q-value $Q_{t+1}(s_t, a_t)$ using Equation (1)
6: **End Procedure**

Using ε-greedy, the agent performs: (a) *exploration* whereby, with a small probability ε, a random action is selected to update



(a) RL agent at time t (b) RL agent at time $t+1$

Fig. 2. Traditional RL agent at time t and $t+1$.

the Q-values of candidate actions $a_t \in A$ so that the best-known action may be identified, and (b) *exploitation* whereby, with probability $1-\varepsilon$, the best-known action a_t^*, which has the maximum Q-value, is selected as follows:

$$a_t^* = \arg\max_{a \in A} Q_t(s_t, a) \tag{3}$$

For simplicity, only exploitation is shown in Algorithm 1.

2.2. Multi-agent reinforcement learning

MARL, which is an extension to RL, enables agents to exchange information (i.e., delayed rewards and Q-values) with each other in order to coordinate their actions [13]. The purpose is to optimize a network-wide objective function or the global Q-value, which is the summation of the local Q-values of all agents in a network, as time goes by $t = 1, 2, 3, \ldots$. Algorithm 2 presents the MARL algorithm. At time instant t, an agent i observes the current state $s_t^i \in S$, sends its own Q-value $Q_t^i(s_t^i, a_t^i)$ to neighboring agents J^i, receives the optimal Q-value $\max_{a^j \in A} Q_t^j(s_t^j, a^j)$ from each neighboring agent $j \in J^i$, and selects an *action* $a_t^i \in A$. Subsequently, at time instant $t+1$, the agent i receives a delayed reward $r_{t+1}^i(s_{t+1}^i)$ for the state–action pair (s_t^i, a_t^i) under the next state $s_{t+1}^i \in S$, and updates Q-value $Q_t^i(s_t^i, a_t^i)$ for the state–action pair. Based on Eq. (1), the Q-value $Q_t^i(s_t^i, a_t^i)$ is updated using Q-function as follows [14]:

$$Q_{t+1}^i(s_t^i, a_t^i) \leftarrow Q_t^i(s_t^i, a_t^i) + \alpha \delta_t^i(s_t^i, a_t^i) \tag{4}$$

where the temporal difference $\delta_t^i(s_t^i, a_t^i)$ of agent i is as follows [15]:

$$\delta_t^i(s_t^i, a_t^i) = r_{t+1}^i(s_{t+1}^i) + \gamma \sum_{j \in J^i} n^{i,j} \max_{a^j \in A} Q_t^j(s_t^j, a^j) \tag{5}$$

where $n^{i,j}$ represents the weight (or importance) of neighboring agent j at agent i, and $\sum_{j \in J^i} n^{i,j} = 1$.

Algorithm 2 MARL algorithm embedded at agent i

1: **Procedure**
2: observe current state $s_t^i \in S$
3: send Q-value $Q_t^i(s_t^i, a_t^i)$ to neighboring agents J^i
4: receive $\max_{a^j \in A} Q_t^j(s_t^j, a^j)$ from agent $j \in J^i$
5: select action $a_t^i \in A$ using Equation (3)
6: receive delayed reward $r_{t+1}^i(s_{t+1}^i)$
7: update Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ using Equation (4)
8: **End Procedure**

2.3. Deep Q-network

In DQN, the ANN is comprised of three convolutional layers whereby data is flowed from the *input layer* to the *hidden layer*, and finally the *output layer* during training as shown in Fig. 3. The input layer represents the state, and each neuron is fully-connected with those in the hidden layer. The hidden layer represents the patterns of the high-dimensional and complex states generated using nonlinear functions, and each neuron is fully-connected with those in the input and output layers. The output layer represents the Q-value $Q_t(s_t, a_t)$ of possible actions a_t, and each neuron is fully-connected with those in the hidden layer. Each connecting link is associated with a *weight*. The output of a neuron k is as follows [16]:

$$y_k = \varphi\left(\sum_{j=0}^{m} w_{kj}.x_j\right) \tag{6}$$

where: (a) w_{kj} represents the weight, which is assigned on the basis of the relative importance of input x_j compared to other inputs, at neuron k, and (b) $\varphi(.)$ represents a sigmoid activation function used to exhibit a balanced behavior between linear and non-linear functions at neuron k.

Compared to the traditional RL approach, DQN has two main features, namely *experience replay* and *target network* [17]. Using experience replay, an agent stores an experience $e_t = (s_t, a_t, r_{t+1}, s_{t+1})$ in a replay memory $D_t = (e_1, e_2, \ldots, e_t)$, and subsequently trains itself using experiences randomly selected from the replay memory. Using target network, weight θ_k is used to approximate the Q-values $Q(s, a; \theta_k)$ at iteration k.

There are two main activities in DQN. Firstly, during action selection, the data is flowed from the input layer to the output layer, whereby the output layer generates the Q-value for each possible action. Secondly, during training, an agent stores an experience in a replay memory, and subsequently trains itself using experiences randomly selected from the replay memory. The agent also utilizes a duplicate of the main network to generate target Q-values, which approximate the weights of the main network. By backpropagation, the target Q-values are used to compute the loss of a selected action in order to stabilize training. The weight of the duplicate network is updated with the weight of the main network every certain number of iterations.

3. Related work

This section presents related works on traffic flow models, and traffic signal control techniques.

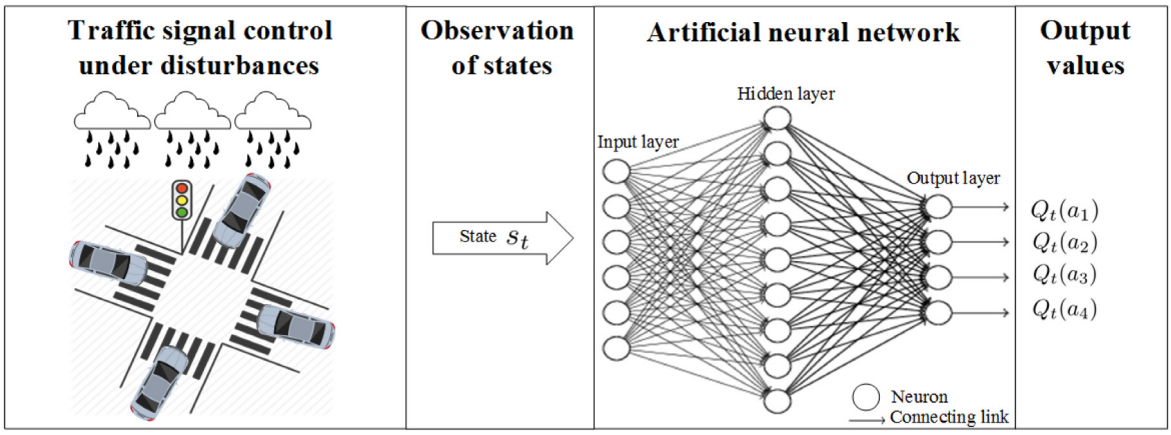


Fig. 3. DQN for estimating the Q-values of actions of a traffic signal controller under disturbances at an intersection.

3.1. Traffic flow models

The presence of disturbances, particularly rainfall, causes unpredictable vehicle arrivals and increased queue length of vehicles, so a non-exponential distribution has been proposed to model vehicle time headway (or headway in short) that represents the inter-arrival time X between two consecutive vehicles. In an empirical study [18], the use of the Burr type XII distribution is compared with several statistical probability distributions in modeling headway under various rainfall intensities. The Burr type XII distribution has been shown to give the best fit to data collected from a main route called J5 in Johor Bahru, Malaysia. The Burr type XII distribution has a probability density function given as:

$$f(x) = \frac{ck(\frac{x}{\beta})^{c-1}}{\beta\left(1+(\frac{x}{\beta})^{c}\right)^{k+1}} \tag{7}$$

where β is a scale parameter (i.e., a higher and lower β value stretches and shrinks the distribution, respectively), as well as c and k are shape parameters which are reciprocals of the scale parameter.

Table 1 presents a summary of various statistical probability distribution models for different rainfall conditions [18]. In traffic signal control investigations, while Poisson process has been widely applied to model headway using exponential distribution [1], Table 2 presents a summary of various statistical probability distribution models for different traffic types.

In this study, vehicle time headway at an intersection is modeled using the Burr type XII distribution, which results in a modified Poisson process, to account for traffic disturbances, particularly rainfall.

3.2. Traffic signal controllers

There are three main types of traffic signal controllers. Firstly, pretimed traffic signal controllers use the Webster formula to determine the traffic phases and traffic phase splits based on the historical traffic data collected at different times of day [2]. Due to the deterministic nature of the traffic phases and traffic phase splits, the traffic signal controllers are not responsive to the dynamicity of the traffic conditions. Secondly, the actuated traffic signal controllers that determine traffic phases and traffic phase splits based on instantaneous traffic conditions. For instance, green signals are activated at lanes with vehicles [3], and traffic phase splits are increased with the queue length of vehicles at lanes [25]. Thirdly, RL- and MARL-based traffic signal controllers that determine traffic phases and traffic phase splits based on longer-term traffic conditions. RL and MARL enables traffic signal controllers to learn about the relationships between actions and rewards (i.e., delayed and discounted rewards) in order to adapt to longer-term traffic conditions that may change in real time [26], such as the waiting time and the queue length of vehicles at a lane. Nevertheless, RL and MARL are marred by the curse of dimensionality.

DQN approaches have been embedded in traffic signal controllers. The traditional DQN approach is adopted in [27–30]. The stacked auto encoder (SAE) neural network, which is adopted in [31], enables agents to compress and store inputs in an efficient manner, and generates outputs that resemble the inputs as much as possible. The dueling DQN, which is adopted in [32], enables two separate estimators to predict the states and the priority of each action, respectively. The value-based DQN, which is adopted in [33], updates more than a single Q-value at any time.

There are three main representations in the DQN models. The state represents the decision making factors, such as the presence and speed of a vehicle at a lane [27–29,33], the queue length of vehicles at a lane [30,31], and the position of a vehicle at a lane [32]. Examples of actions are traffic phase splits [27,28,32,33] and traffic phases [29–31]. The reward represents the appropriateness of a state–action pair, such as: (a) the changes of the travel delay (i.e., the additional time incurred compared to travel time without traffic congestion) of vehicles at an intersection between two successive actions [27–30]; and (b) the travel time (or travel delay) of vehicles [31–33]. The state is fed via the input layer, and the output layer provides the Q-values of all the possible traffic phase splits [27,28,32,33] and traffic phases [29–31]. The proposed scheme has shown to increase throughput (or the number of vehicles crossing an intersection) [30,33], and reduce the average travel time (or travel delay) of vehicles [27,28,31,32].

In this research, the traditional DQN approach is applied to traffic signal controllers to monitor traffic flows and alleviate traffic congestion at intersections in the presence of disturbances (i.e., rainfall). In addition, the multi-agent DQN is first investigated and applied to traffic signal controllers so that they can exchange knowledge, learn, and make optimal joint actions in a collaborative manner.

4. Proposed model for RL-based and DQN-based traffic signal control

Consider a set of intersections I in a traffic network. Each intersection $i \in I$ has: (a) a set of incoming lanes K^i, and (b) a set of neighboring intersections J^i. The intersection activates traffic phases in a round-robin fashion, and adjusts the traffic phase

Table 1
Probability distributions for modeling vehicle headways under different rainfall intensities.

Probability distribution	Description	Suitable for
Burr type XII	Uses flexible parameters, particularly positive random variables, to achieve a wide range of distribution shapes in continuous probability function.	Different kinds of weather conditions (i.e., no, light, medium, and heavy rainfall).
Frechet (or inverse Weibull or generalized extreme Type II)	Uses continuous probability distribution to represent headway data in order to model extreme events.	Different kinds of weather conditions (i.e., no, light, and medium rainfall).
Pearson type VI	Uses flexible parameters in continuous probability distribution to represent a large data set.	Different kinds of weather conditions (i.e., no, light, and medium rainfall).
Generalized extreme	Uses continuous probability distribution to model the smallest or the largest values of a large data set comprised of independent distributed random variables representing headway.	Heavy rainfall.
Generalized Pareto	Uses continuous probability distribution to model the tails of a distribution.	Heavy rainfall.
Lognormal (or Galton)	Uses continuous probability distribution, in which its logarithm has a normal distribution, to represent headway data.	Heavy rainfall.

Table 2
Probability distributions for modeling vehicle headways in traffic signal control investigations.

Probability distribution	Differences compared to the Poisson process	Suitable for
Queuing model [19]	This model uses random and non-random arrival patterns to model independent and dependent vehicle arrivals, respectively.	Traffic with slow moving vehicles.
Shifted exponential model [20]	This model uses exponential and semi-Poisson distributions to model flow rate and the distance (or gap) between vehicles, respectively.	Traffic with short and moderate headway.
Semi-Poisson model [21]	This model modifies the traditional Poisson process to model a large headway data set.	Traffic with long headway.
Modified Poisson model with gamma (or Erlang) distribution [22][23]	This model uses a negative binomial distribution to provide a better fit as compared to the traditional Poisson process.	Traffic with short headway.
Burr type XII [24]	This model uses non-random arrival pattern for dependent vehicle arrivals as compared to the traditional Poisson process.	High traffic volume with disturbances (i.e., rainfall).

splits for the traffic phases using DQN. The rest of this section presents the representations of our proposed MADQN model, including the state space, action space, and delayed reward, applied to traffic signal controllers. The DQN and MADQN algorithms are also presented.

Algorithm 4 MADQN algorithm embedded at agent i

Procedure
  for episode = 1 : M **do**
    observe current state s_t^i
    send Q-value $Q_t^i(s_t^i, a_t^i)$ to neighboring agents J^i
    receive $\max_{a^j \in A} Q_t^j(s_t^j, a^j)$ from agent $j \in J^i$
    for $t = 1 : T$ **do**
      perform steps 5 to 13 of Algorithm 3
    end for
    update Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ using Equation (4)
  end for
End Procedure

4.1. State space

At an intersection i, each state $s_t^i = (s_{1,t}^i, s_{2,k,t}^i, s_{3,k,t}^i, s_{4,t}^i, s_{5,t}^i)$ represents a five-tuple information, namely:

- $s_{1,t}^i = \{0, 1, 2, 3, \ldots, s_{1,max}^i\}$ represents the current traffic phase at intersection i. For instance, in Fig. 1, a 0 value represents the north-east bound traffic phase, 1 represents east-south, 2 represents west-north, and 3 represents south-west.
- $s_{2,k,t}^i = \{0, 1, 2, 3, \ldots\}, \forall k \in K^i$ represents the queue lengths of all the incoming lanes K^i of intersection i.
- $s_{3,k,t}^i = \{0, 1, 2, 3, \ldots\}, \forall k \in K^i$ represents the queue lengths of all the incoming lanes $k \in K^j$ at a neighboring intersection j.
- $s_{4,t}^i = t_{red,t}^{i,k}$ represents red timing, which is the time elapsed since the signal of a lane $k \in K^i$ turned into red, at intersection i.
- $s_{5,t}^i = \{0, 1, 2, 3, \ldots, s_{5,}^i\}$ represents the rainfall intensity with a zero value being no rain and the maximum value $s_{5,\max}^i$ being the heaviest rain. In other words, substate $s_{5,t}^i$ represents the intensity of disturbance, which can be measured using sensors.

4.2. Action space

Consider a series of traffic phases executed in a deterministic round-robin fashion, such as the traffic phases shown in Fig. 1. The action space represents a traffic phase split $a_t^i = \{0, 1, 2, \ldots, a_{\max}\}$ that can be selected by a traffic signal controller i, where $a_t^i = 0$ skips a traffic phase in a fixed predetermined sequence of traffic phases (e.g. lack of waiting vehicles at a lane).

4.3. Delayed reward

The delayed reward $r_t^i(s_t^i)$ represents the difference in the total waiting time of all vehicles at an intersection i at time t and time $t + 1$. Hence, the delayed reward captures the increment

Algorithm 3 The traditional single-agent DQN algorithm

Procedure
  for episode = 1 : M **do**
    observe current state s_t^i
    for $t = 1 : T$ **do**
      select action a_t^i using Equation (3)
      receive delayed reward $r_{t+1}^i(s_{t+1}^i)$ and next state s_{t+1}^i
      store experience $(s_t^i, a_t^i, r_{t+1}^i(s_{t+1}^i), s_{t+1}^i)$ in replay memory D_t^i
      sample a random minibatch of experiences $(s_t^i, a_t^i, r_{t+1}^i(s_{t+1}^i), s_{t+1}^i)$ from replay memory D_t^i
      for $j = 1 : N$ **do**
      set target y_j^i

$$= \begin{cases} r_{j+1}^i(s_{j+1}^i), & \text{if episode terminates} \\ r_{j+1}^i(s_{j+1}^i) + \gamma \max_a Q(s_{j+1}^i, a; \theta_j^i), & \text{otherwise} \end{cases}$$

      perform a gradient descent optimization on $(y_j^i - Q(s_j^i, a_j^i; \theta_j^i))^2$ with respect to θ_j^i
      end for
    end for
  end for
End Procedure

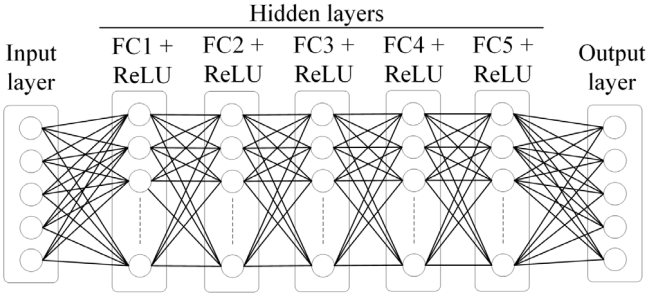


Fig. 4. DQN architecture.

and decrement of the total waiting time of all vehicles at the intersection i between before taking an action and after taking the action. The delayed reward is as follows:

$$r_t^i(s_t^i) = W_t^i - W_{t+1}^i \tag{8}$$

The agent receives a positive delayed reward when $W_t^i > W_{t+1}^i$, a negative delayed reward when $W_t^i < W_{t+1}^i$, and a zero delayed reward when $W_t^i = W_{t+1}^i$.

4.4. DQN and MADQN

This section presents the architecture and algorithm of the traditional DQN approach, as well as the multi-agent DQN, applied in this paper.

4.4.1. DQN architecture

Fig. 4 shows the architecture of DQN used in this paper. DQN is embedded in traffic signal control at intersection i. Three main types of layers are: (a) the *input layer* has 5 neurons, each representing a state; (b) 5 fully connected (FC) *hidden layers* with 400 neurons each; and (c) the *output layer* has 5 neurons, each representing a possible action. Each link is associated with a weight. Each node has a rectified linear activation function (ReLU) that performs gradient descent. During training, the 5 substates of state $s_t^i = (s_{1,t}^i, s_{2,k,t}^i, s_{3,k,t}^i, s_{4,t}^i, s_{5,t}^i)$ are fed into the neurons of the input layer. Subsequently, information flows forward to the hidden layers, and finally to the output layer that provides the Q-values $Q_t^i(s_t^i, a_t^i)$ of its possible actions $a^i = (0, 1, 2, 3, 4)$, where $a_{\max} = 4$ at intersection i.

4.4.2. DQN algorithm

Algorithm 3 shows the algorithm for DQN; for simplicity, only exploitation is shown in the algorithm. At episode $m \in M$, an agent i observes the current state $s_m^i \in S$ as part of initialization. At time instant $t \in T$, agent i selects an action $a_t^i \in A$ using Eq. (3), and stores its experience $e_t^i = (s_t^i, a_t^i, r_t^i, s_{t+1}^i, a_{t+1}^i)$ in a replay memory $D_t^i = (e_1^i, e_2^i, \ldots, e_t^i)$. Subsequently, the agent samples a minibatch of experiences from the replay memory D_t^i in a random manner. At iteration $j \in J$, agent i learns the weight θ_j^i and $Q_j^i(s_j^i, a_j^{i*}, \theta_j^i) \approx Q^*(s_j^i, a_j^i)$. In order to train the DQN, the loss function at iteration j is minimized as follows:

$$L_j^i(\theta_j^i) = \mathbb{E}_{s_j^i, a_j^i \sim p(.)}\left[\left(y_j^i - Q_j^i(s_j^i, a_j^i; \theta_j^i)\right)^2\right] \tag{9}$$

where $p(s, a)$ represents the probability distribution of a state–action pair (s, a), and y_j^i represents the target given by θ_{j-1}^i in the previous iteration $j-1$. The gradient of the loss function $\nabla_{\theta_i} L_j^i(\theta_j^i)$ is given as follows:

$$\begin{aligned} \nabla_{\theta_i} L_j^i(\theta_j^i) = & \mathbb{E}_{s_j^i, a_j^i \sim p(.); s_{j+1}^i \sim \mathcal{E}}\Big[\left(y_j^i - Q_j^i(s_j^i, a_j^i; \theta_j^i)\right) \\ & \nabla_{\theta_j^i} Q_j^i(s_j^i, a_j^i; \theta_j^i)\Big] \end{aligned} \tag{10}$$

4.4.3. MADQN

In this work, the traditional single-agent DQN approach is extended to MADQN, which has not been investigated in the literature. MADQN is evaluated under different traffic scenarios with disturbances.

Similar to MARL, MADQN enables multiple DQN agents to exchange knowledge, learn, and make optimal joint action in a collaborative manner. The main challenge of MADQN is to achieve stability, or to converge to an optimal joint action, in a moving target and shared environment. Under the moving target environment, the MADQN agents perform their respective actions simultaneously, and so an agent's action affects the operating environment of neighboring agents. For instance, the action of the traffic signal controller at an upstream intersection can affect the operating environment (e.g., the congestion level) at neighboring and downstream intersections since vehicles traverse from one intersection to another. Hence, the actions of an agent at an intersection can affect the actions selected by the agents at neighboring intersections. Consequently, the dynamicity of the operating environment increases and affects stability. In order to address the moving target issue, the agents consider the actions

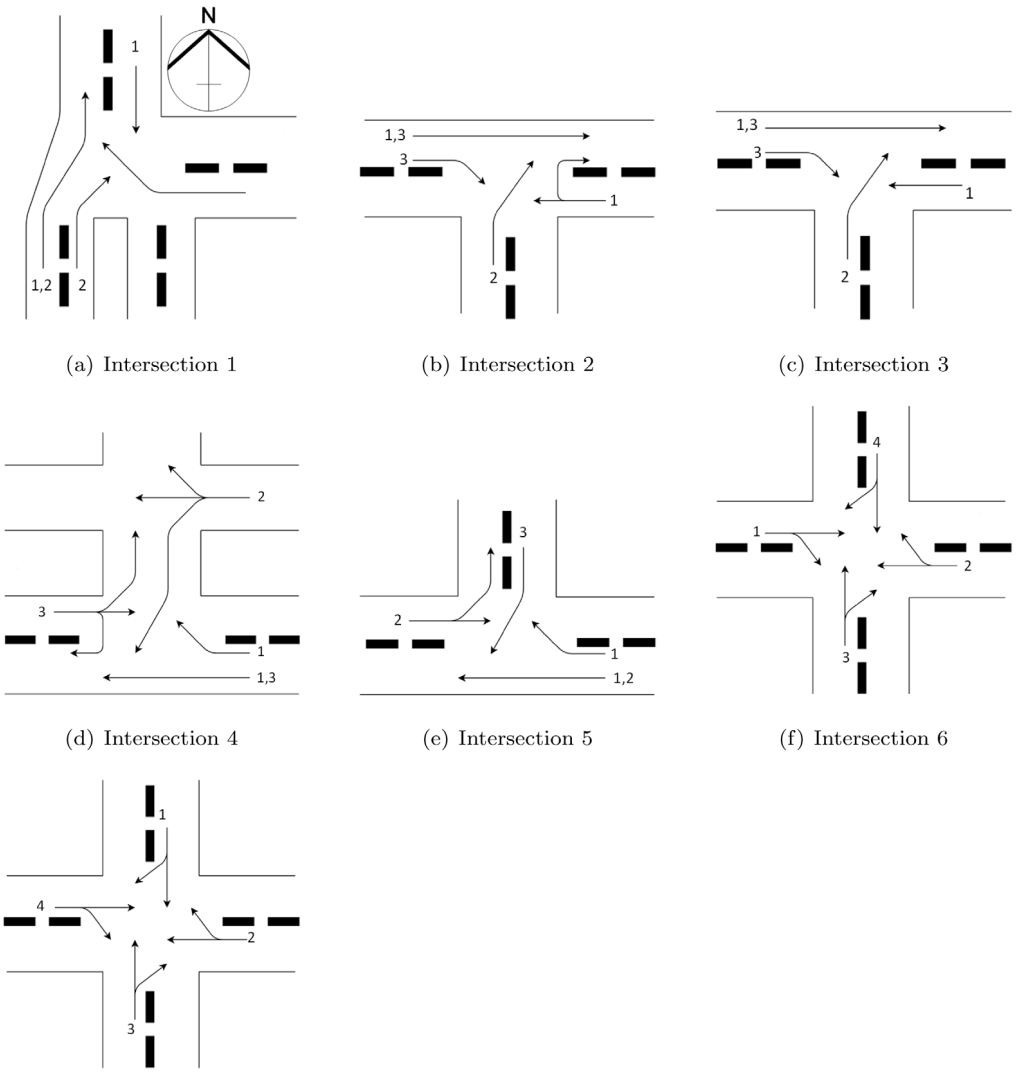


Fig. 5. Seven intersections in the Sunway city shown in Fig. 6.

taken by their respective neighboring agents, and coordinate with each other in a collaborative manner. Nevertheless, the convergence of a multi-agent system has been shown in [13], whereby multiple agents exchange knowledge, learn, select their respective actions, and subsequently converge to an optimal joint action as time goes by. Specifically, the global Q-value, which sums up the local Q-value of each agent and represents the global objective function, converges to an optimal equilibrium. The convergence is attributed to: (a) the availability of the local view of neighboring agents to an agent; (b) the update of the Q-values of an agent using neighboring agents' information (e.g., Q-values); and (c) the action of an agent being the best response to the neighboring agents. Figs. 8 and 9 show the convergence of the delayed reward as the episode increases, and Figs. 10 to 15 show that MADQN achieves higher stability compared to MARL.

Algorithm 4 presents the MADQN algorithm. At time instant t, a DQN agent i observes the state $s_t^i = (s_{1,t}^i, s_{2,k,t}^i, s_{3,k,t}^i, s_{4,t}^i, s_{5,t}^i)$, sends its own Q-value $Q_t^i(s_t^i, a_t^i)$ to neighboring agents J^i, receives the optimal Q-value $\max_{a^j \in A} Q_t^j(s_t^j, a^j)$ from each neighboring agent $j \in J^i$, and selects its *action* using Eq. (3). Subsequently, at time instant $t+1$, the agent i receives a delayed reward $r_{t+1}^i(s_{t+1}^i)$ for the state–action pair (s_t^i, a_t^i) under the next state s_{t+1}^i, and updates Q-value $Q_t^i(s_t^i, a_t^i)$ for the state–action pair.

There are three main advantages of MADQN over MARL.

- MADQN uses ANN that provides a continuous representation of the state space, and so an unlimited number of state–action pairs can be represented.
- MADQN provides an efficient storage for complex input in order to address the curse of dimensionality.
- MADQN uses experience replay and target network that allows more stable training as compared to MARL.

5. Case study

This paper conducts a case study on a real traffic network in an urban area called Sunway city in Malaysia. In addition, a grid traffic network is also investigated.

5.1. Sunway city

Sunway city is a busy residential and commercial area surrounded by higher educational institutions (i.e., Sunway University and Monash University Malaysia campus), high density residential areas (i.e., Sunway Monash Residence and LaCosta), theme park (i.e., Sunway Lagoon), hospital (i.e., Sunway Medical

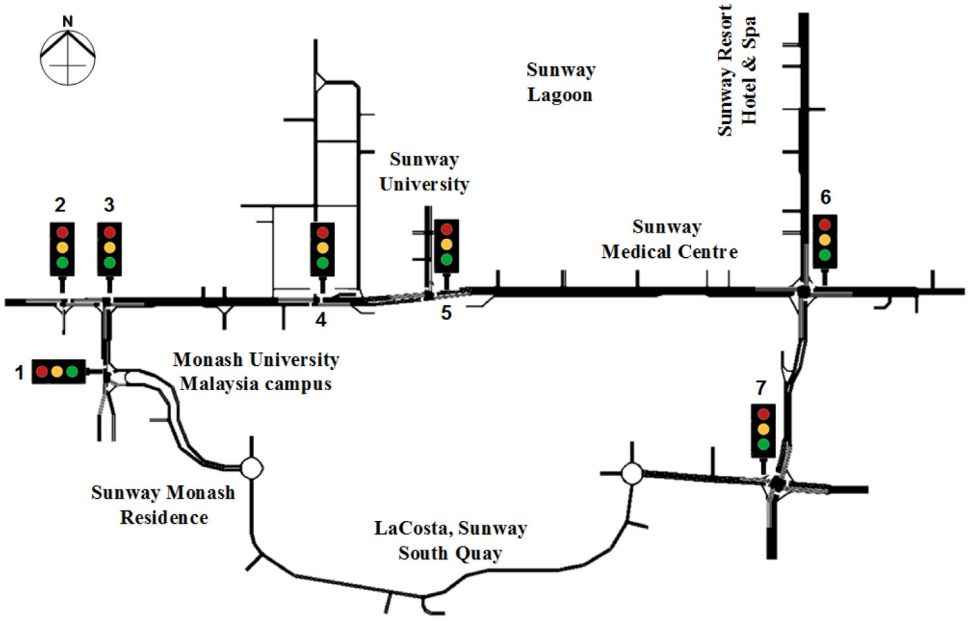


Fig. 6. Traffic network and the locations of traffic signal controllers in the Sunway city.

Table 3
Traffic phase splits for deterministic traffic signal controllers in Sunway city.

Intersection	Traffic phase split			
	1	2	3	4
1	15	10	40	–
2	25	25	15	–
3	30	10	30	–
4	20	20	20	–
5	30	30	30	–
6	40	25	40	25
7	20	60	20	10

Centre), and so on, as shown in Fig. 6. In Fig. 6, there are seven intersections, and each intersection has a traffic signal controller. At each intersection, inductive loop detectors are installed to gather short-term information (i.e., the presence or absence of vehicle(s) at a lane), and this enables the traffic signal controllers to skip traffic phases without waiting vehicles. Malaysia ranks fifth and third worldwide in rainfall (i.e., approximately 1000 mm per year [34]) and lightening strikes (i.e., approximately 240 thunderstorm days per year [35]), respectively. Hence, traffic congestion is a serious problem, particularly during the peak hours, and the Malaysian weather (i.e., rainfall as the disturbance) compounds the problem.

5.2. Existing traffic signal controllers

At present, the traffic signal controllers select traffic phases in a deterministic manner, in which traffic phases are executed in a deterministic round-robin fashion with certain periods of traffic phase splits. The traffic phase splits can be dynamically adjusted based on short-term information, particularly the presence or absence of vehicle(s) at a lane as detected by inductive loop detectors. Fig. 5 shows the traffic phases, and Table 3 shows the traffic phase splits at all intersections in Sunway city (see Fig. 6).

5.3. Grid traffic network

In order to show the effectiveness of MADQN, we have extended our simulation to a grid traffic network, which has been

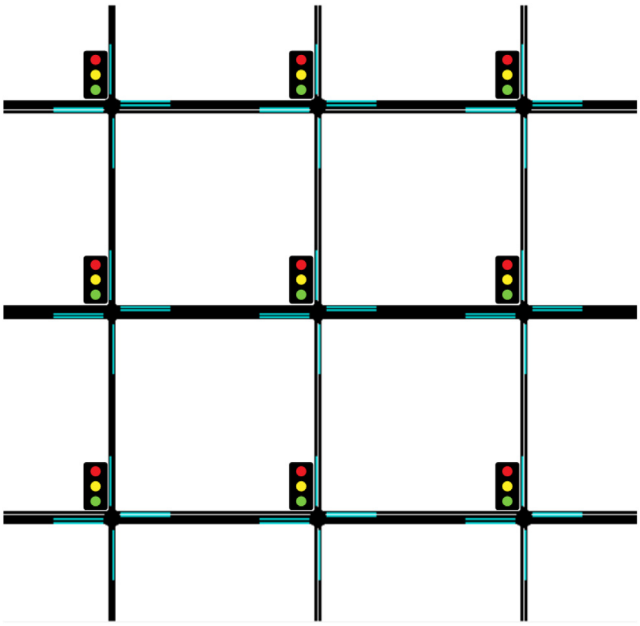


Fig. 7. 3 × 3 grid traffic network with 9 intersections.

widely used in the literature [28,36–38]. As shown in Fig. 7, the size of the grid traffic network is 3 × 3, and so there are 9 intersections. Each intersection has 4 legs, and all intersections are installed with inductive loop detectors as seen in the traffic network of the Sunway city. These inductive loop detectors gather short-term information (i.e., the presence or absence of vehicle(s) at a lane), and this enables the traffic signal controllers to skip traffic phases without waiting vehicles.

6. Simulation and results discussion

There are four simulated approaches, namely deterministic, RL, MARL, and MADQN. The deterministic traffic signal controllers are currently in use at all intersections in the Sunway city. The RL-based and MARL-based traffic signal controllers have been widely

Table 4
Performance measures under increased traffic volume.

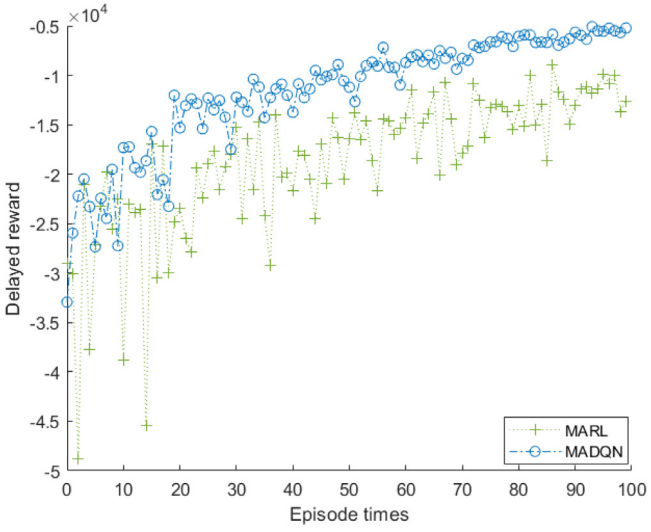
Performance measure	Metric	Unit	Recurring traffic congestion				Non-recurring traffic congestion			
			Deterministic	RL	MARL	MADQN	Deterministic	RL	MARL	MADQN
Queue length	Max	No. of vehicles	12	12	9	9	19	18	16	10
	Mean		5.8400	5.9100	4.5300	4.0319	9.7300	9.4900	7.5600	4.0537
	Median		6	5	4	3.7203	10	9	7	3.6796
	Min		2	2	1	2	2	4	3	2
Waiting time	Max	Seconds	22	35	30	8.6250	30	27	25	9
	Mean		9.0900	11.2800	7.9900	2.8284	5.6400	11.5400	7.9200	2.8969
	Median		8	9	7	2.1875	4	11	8	2.3348
	Min		0	0	0	0	0	0	0	0
Throughput	Max	No: of vehicles	19	38	49	97	22	42	50	96
	Mean		6.7800	18.9500	22.2700	78.1900	9.3300	22.0200	25.9600	77.2700
	Median		6	18	21	78.5000	9	21	26	79
	Min		1	5	10	59	3	4	7	54

investigated in the literature, and so they are chosen as baselines in this study. The single-agent DQN approach has been widely investigated in the literature, and has shown to outperform the deterministic and RL approaches. This work is mainly focused on the multi-agent system, and so MADQN is proposed. RL, MARL, and MADQN are unable to be deployed in real-world as this affects the traffic in a real traffic network. Hence, we have chosen to investigate these approaches using simulation platform SUMO, which has been the preferred tool for similar investigation in the literature [27–30,32,38–40]. The vehicle arrivals are dependent on the Burr type XII distribution, which is a modification of the Poisson process. Table 2 presents various probability distribution models used in the literature, and it shows that the Burr type XII distribution is suitable to model a traffic network with high traffic volume with disturbances, which reflect the real traffic network. The Burr type XII distribution has been shown to reflect a traffic network with disturbance in Johor Bahru, Malaysia [18]. Table 3 presents the traffic phase split of existing traffic signal controllers in the Sunway city, which was observed during the evening peak hours (i.e., 5–7 pm) of a working day. The traffic phase splits were measured using a stopwatch, and they may differ for different traffic phases.

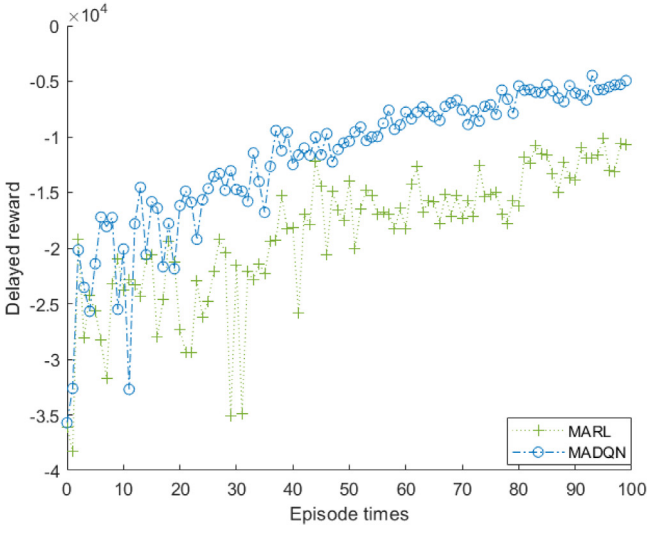
This section presents the simulation environment and experimental results discussion. Simulation results are presented in two separate sections for recurring and non-recurring traffic congestion, respectively. The performance measures for the four kinds of traffic signal controllers (i.e., deterministic, RL, MARL, and MADQN) under recurring traffic congestion (i.e., caused by increased traffic volume) and non-recurring traffic congestion (i.e., caused by disturbances) are summarized in Table 4.

6.1. Simulation setup

Simulation is conducted using traffic simulator SUMO (i.e., version 1.1.0) and MATLAB (i.e., version 9.5) that are interconnected with each other. SUMO, which is an open source traffic simulator, provides real-time microscopic traffic simulation [41]. MATLAB, which is a dynamic programming software used to develop algorithms, computes large arrays and matrices, as well as accumulates and records statistics. In order to interconnect SUMO and MATLAB, the SUMO TraCI (Traffic Control Interface) protocol, namely TraCI4Matlab, is used so that the traffic simulation in SUMO can interact with MATLAB scripts using TCP/IP in a client–server manner, whereby SUMO acts as the server, and MATLAB as the client. The XML resource files, which provide details on the traffic arrival rate and speed limits of vehicles, defines two main types of traffic congestions, namely recurring and non-recurring traffic congestions. The recurring traffic congestion represents the congestions caused by an increased traffic volume (i.e., particularly during peak hours), while the non-recurring traffic congestion represents the congestions caused by disturbance (i.e., particularly rainfall).



(a) Recurring traffic congestion



(b) Non-recurring traffic congestion

Fig. 8. Cumulative delayed reward under recurring and non-recurring traffic congestion in the Sunway city traffic network increases with episode. MADQN achieves a higher value compared to MARL. Higher value improves the performance of traffic network.

Table 5
Simulation parameters for the Burr type XII distribution model.

Parameters	Disturbance condition			
	NR	LR	MR	HR
Shape parameter c	4.74	4.75	4.88	5.00
Shape parameter k	0.18	0.21	0.22	0.27
Scale parameter β	0.94	1.03	1.07	1.33

Table 6
Simulation parameters for the DQN agent.

Parameters	Values
Replay memory size	50 000
Minibatch size	100
Learning rate α	0.00025
Discount factor γ	0.75
Experience sampling	0.5

6.2. Simulation parameters and performance measures

For the Burr type XII distribution model, simulation parameters are presented in Table 5. NR, LR, MR, and HR represents different rainfall intensities, namely *no rain*, *light rain*, *moderate rain*, and *heavy rain* scenarios, respectively, and these scenarios can be detected by weather sensors. The scale parameter β, as well as the shape parameters c and k, increase with the rainfall intensity [18]. For the DQN agent, the simulation parameters that provide the best possible results are presented in Table 6.

There are three performance measures.

- The *queue length* represents the average number of waiting vehicles (i.e., with a speed of 0 km/h) at an intersection i at end of a red timing.
- The *waiting time* represents the average waiting time of all the vehicles at an intersection i at the end of a red timing.
- The *throughput* represents the number of vehicles crossing an intersection i within a single traffic phase during a green timing, which is the time elapsed since the signal of a lane $k \in K^i$ turned into green, at intersection i.

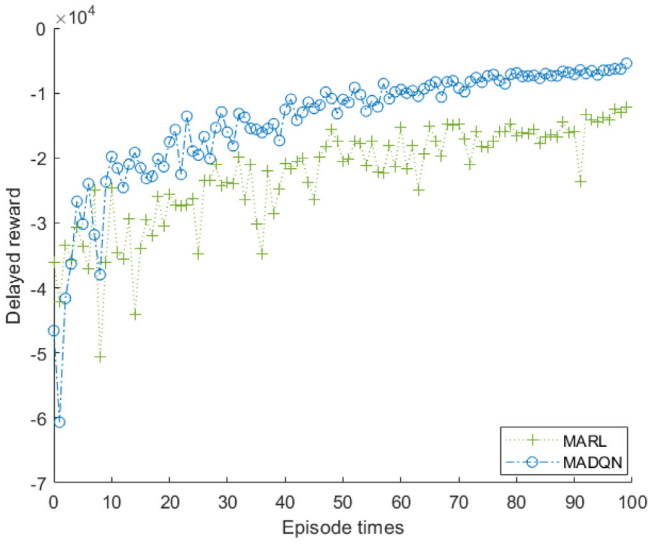
Our proposed schemes aims to: (a) reduce the queue length and waiting time of all the vehicles at intersections; and (b) maximize throughput and delayed reward, which helps to reduce the queue length of all lanes.

6.3. Simulation results

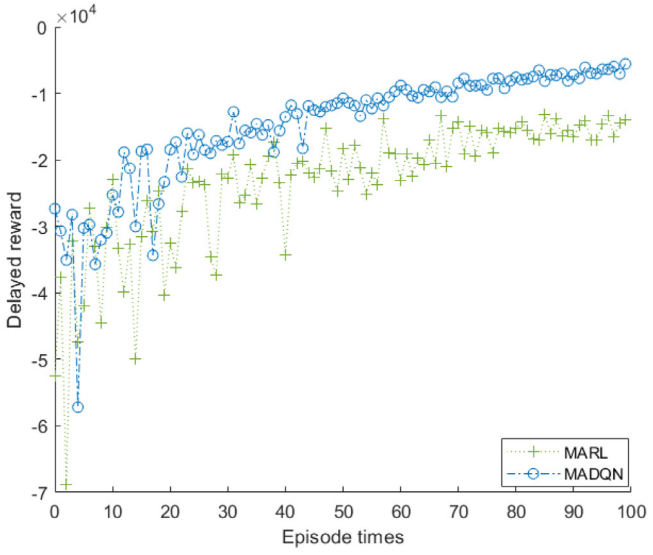
This section shows the evaluation results of our proposed MADQN approach by comparing its performance with those obtained from the deterministic, RL, and MARL approaches under recurring traffic congestion (i.e., caused by increased traffic volume) and non-recurring traffic congestion (i.e., caused by disturbances).

6.3.1. Cumulative delayed reward

The accumulated delayed reward for MADQN and MARL under recurring and non-recurring traffic congestions as the episode increases is shown in Figs. 8(a), 9(a), and 8(b), 9(b), respectively. The convergence of the accumulated delayed reward for MADQN and MARL for the Sunway city traffic network and the grid traffic network is shown in Figs. 8 and 9, respectively. MADQN achieves a higher accumulated delayed reward as compared to MARL in both types of traffic congestions and traffic networks. At the initial episodes (i.e., less than 20 episodes), the accumulated delayed reward for both MADQN and MARL is unstable; however, MADQN is more stable attributed to its main features, namely experience replay and target network, which have shown to improve stability [17].



(a) Recurring traffic congestion

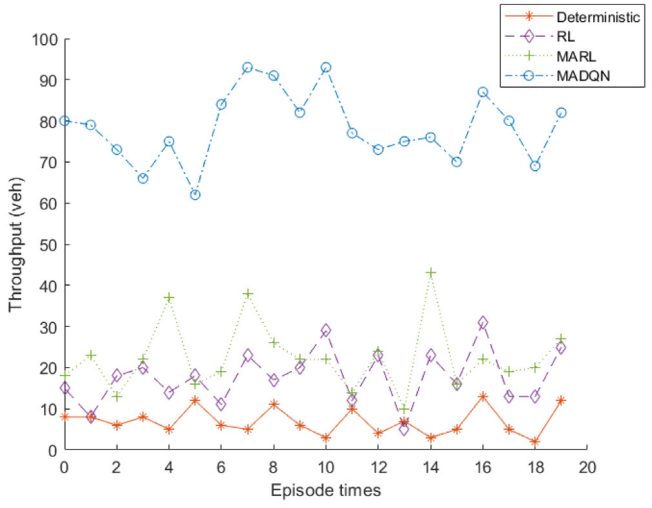


(b) Non-recurring traffic congestion

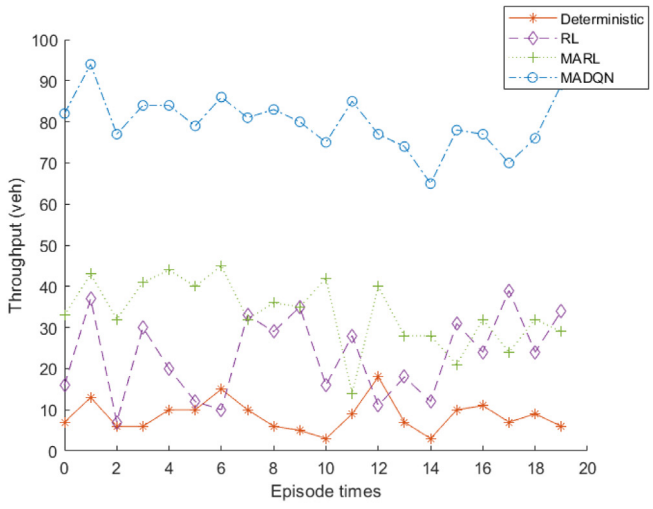
Fig. 9. Cumulative delayed reward under recurring and non-recurring traffic congestion in the grid traffic network increases with episode. MADQN achieves a higher value compared to MARL. Higher value improves the performance of traffic network.

6.3.2. Throughput

The throughput of the four kinds of traffic signal controllers in the Sunway city traffic network under recurring and non-recurring traffic congestions as the episode increases is shown in Figs. 10(a) and 10(b), respectively. MADQN outperforms the other approaches with its throughput more than 90 vehicles. For recurring traffic congestion, the throughput varies up to 20 vehicles for deterministic, up to 30 vehicles for RL, and up to 40 vehicles for MARL. For non-recurring traffic congestion, the throughput of MADQN is more than 90 vehicles, and the throughput varies up to 20 vehicles for deterministic, up to 40 vehicles for RL, and up to 50 vehicles for MARL. Similar trend is observed in the grid traffic network as shown in Fig. 11. For recurring traffic congestion, the throughput of MADQN is more than 90 vehicles, and the throughput varies up to 30 vehicles for deterministic, and

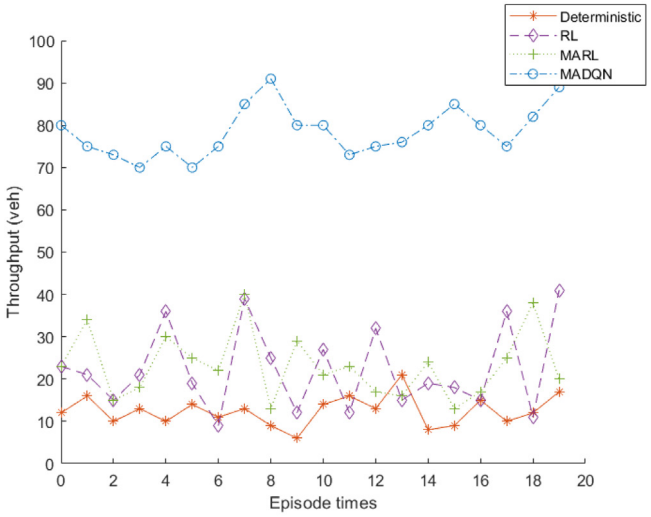


(a) Recurring traffic congestion

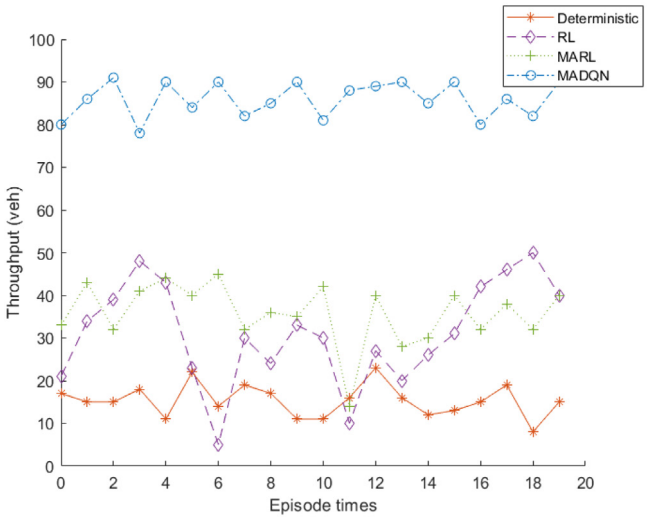


(b) Non-recurring traffic congestion

Fig. 10. Average throughput under recurring and non-recurring traffic congestion in the Sunway city traffic network increases with episode. MADQN achieves the highest value, followed by MARL, RL, and Deterministic. Higher average throughput improves the performance of traffic network.



(a) Recurring traffic congestion



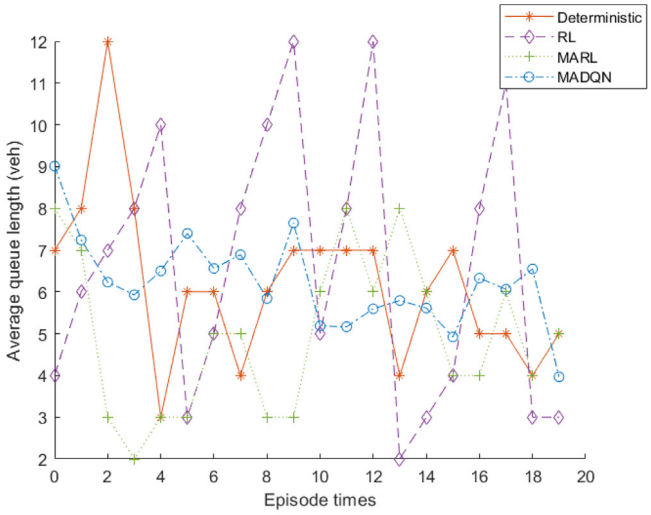
(b) Non-recurring traffic congestion

Fig. 11. Average throughput under recurring and non-recurring traffic congestion in the grid traffic network increases with episode. MADQN achieves the highest value, followed by MARL, RL, and Deterministic. Higher average throughput improves the performance of traffic network.

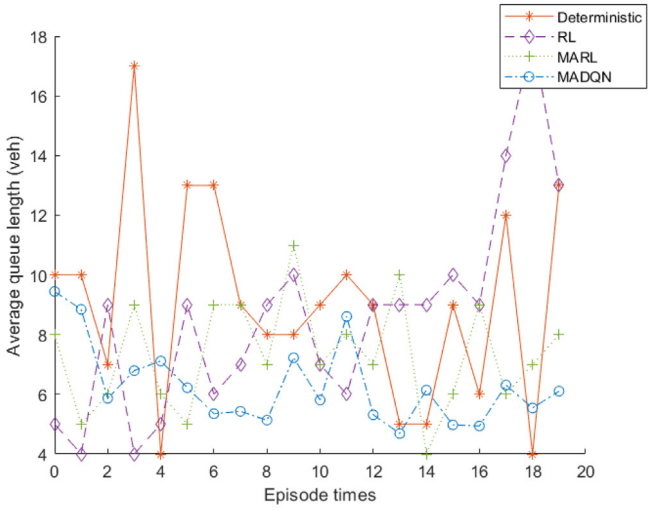
up to 40 vehicles for both RL and MARL. For non-recurring traffic congestion, the throughput of MADQN is more than 90 vehicles, and the throughput varies up to 30 vehicles for deterministic, and up to 50 vehicles for both RL and MARL. Hence, MADQN increases the throughput by up to 70%, and so it can increase throughput of vehicles at intersections for both recurring and non-recurring traffic congestion scenarios.

6.3.3. Queue length

The queue length of vehicles of the four kinds of traffic signal controllers in the Sunway city traffic network under recurring and non-recurring traffic congestions as the episode increases is shown in Figs. 12(a) and 12(b), respectively. For recurring traffic congestion, the queue length varies up to 12 vehicles for both deterministic and RL approaches, and less than 10 vehicles for MARL. MADQN has its queue length less than 8 vehicles and reduces with episode. For non-recurring traffic congestion, the queue length varies up to 20 vehicles for deterministic, up to 18 vehicles for RL, up to 12 vehicles for MARL. MADQN has its queue length less than 10 vehicles and reduces with episode. It reduces to less than 3 vehicles after 80 episodes for both types of traffic congestions. Similar trend is observed in the grid traffic network as shown in Fig. 13. For recurring traffic congestion, the queue length varies up to 16 vehicles for deterministic, MARL and MADQN approaches, and up to 18 vehicles for RL. The deterministic and RL approaches reduces their queue length to 8 vehicles, the MARL and MADQN approaches reduces their queue length to 6 vehicles with episodes. For non-recurring traffic congestion, the queue length varies up to 13 vehicles for deterministic, up to 11 vehicles for both RL and MARL. MADQN has its queue length less than 10 vehicles and reduces with episode. It reduces to less than 3 vehicles after 80 episodes for both types of traffic congestions. It is also more stable attributed to experience replay and target network [17]. Hence, MADQN reduces queue length by up to 75%, and so it can reduce queue length of vehicles at intersections.

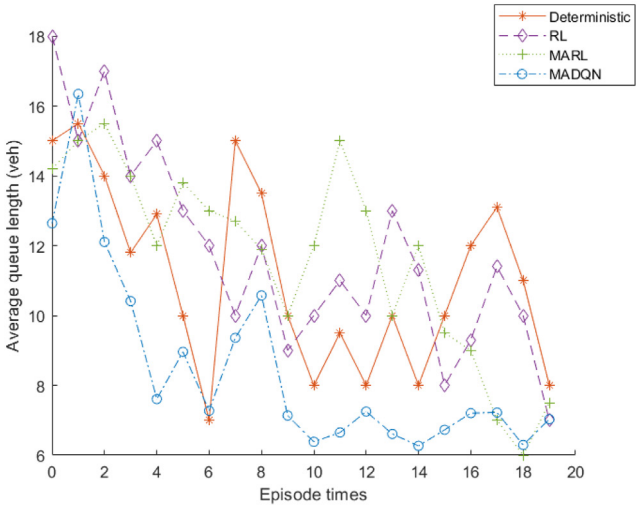


(a) Recurring traffic congestion

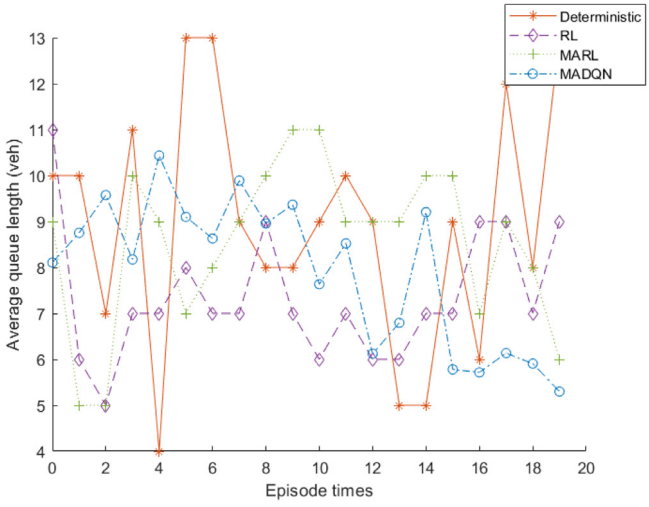


(b) Non-recurring traffic congestion

Fig. 12. Average queue length under recurring and non-recurring traffic congestion in the Sunway city traffic network increases with episode. MADQN achieves the lowest value, followed by MARL, RL, and Deterministic. Lower average queue length improves the performance of traffic network.



(a) Recurring traffic congestion



(b) Non-recurring traffic congestion

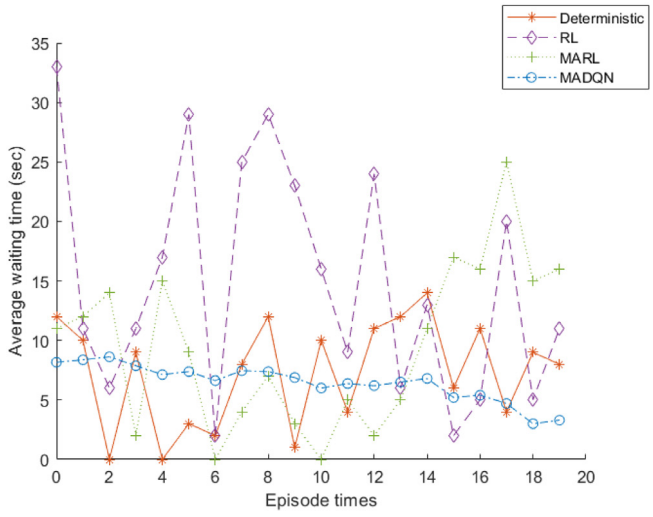
Fig. 13. Average queue length under recurring and non-recurring traffic congestion in the grid traffic network increases with episode. MADQN achieves the lowest value, followed by MARL, RL, and Deterministic. Lower average queue length improves the performance of traffic network.

6.3.4. *Waiting time*

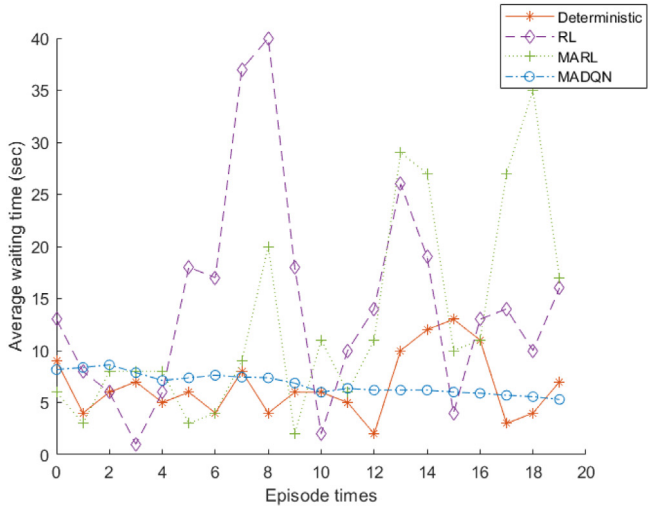
The waiting time of vehicles of the four kinds of traffic signal controllers in the Sunway city traffic network under recurring and non-recurring traffic congestions as the episode increases is shown in Figs. 14(a) and 14(b), respectively. For recurring traffic congestion, the waiting time varies up to 35 s for RL, up to 30 s for MARL, and up to 15 s for deterministic. MADQN has its waiting time less than 10 s and reduces with episode. For non-recurring traffic congestion, the waiting time varies up to 15 s for deterministic, up to 40 s for RL, up to 35 s for MARL. MADQN has its waiting time varies up to 9 s and reduces with episode. It reduces to less than 3 s after 50 episodes for both types of traffic congestions. Similar trend is observed in the grid traffic network as shown in Fig. 15. For recurring traffic congestion, the waiting time varies up to 30 s for deterministic, and up to 26 s for both RL and MARL. MADQN has its waiting time varies up to 10 s and reduces with episode. For non-recurring traffic congestion, the waiting time varies up to 22 s for deterministic, RL, and MARL. MADQN has its waiting time varies up to 10 s and reduces with episode. It reduces to less than 3 s after 50 episodes for both types of traffic congestions. It is also more stable attributed to experience replay and target network [17]. Hence, MADQN reduces waiting time by up to 70%, and so it can reduce waiting time of vehicles at intersections.

7. Conclusion and future work

In this paper, the recurring traffic congestion (i.e. caused by high traffic volume) and non-recurring traffic congestion (i.e. caused by disturbances) are addressed using an artificial intelligence approach called deep reinforcement learning, specifically deep Q-network (DQN), which is a single-agent approach, to address the curse of dimensionality. This paper extends DQN to provide multi-agent DQN (MADQN) in order to solve multi-agent problem by exchanging information (i.e., Q-values) among DQN agents in order to coordinate their actions. MADQN uses artificial neural network to represent and store continuous and complex

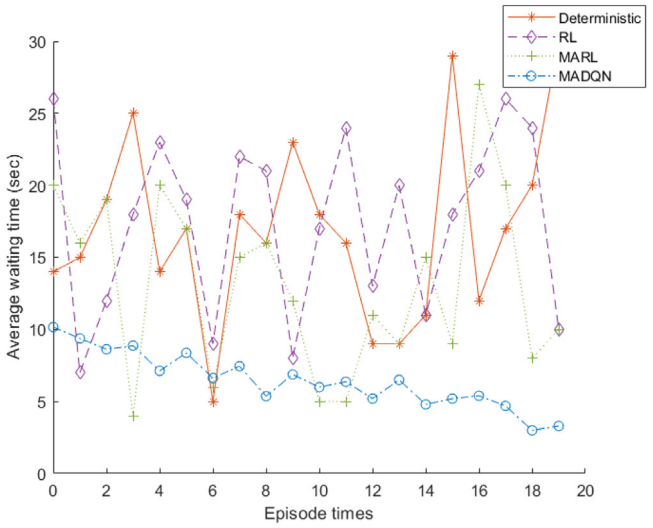


(a) Recurring traffic congestion

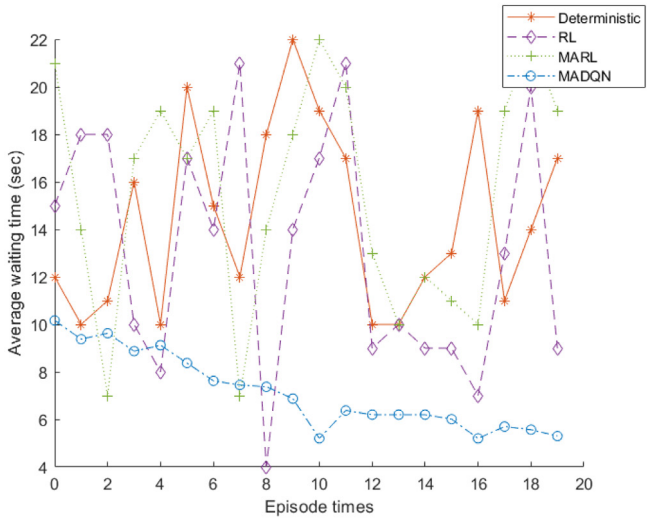


(b) Non-recurring traffic congestion

Fig. 14. Average waiting time under recurring and non-recurring traffic congestion in the Sunway city traffic network increases with episode. MADQN achieves the lowest value, followed by MARL, RL, and Deterministic. Lower average waiting time improves the performance of traffic network.



(a) Recurring traffic congestion



(b) Non-recurring traffic congestion

Fig. 15. Average waiting time under recurring and non-recurring traffic congestion in the grid traffic network increases with episode. MADQN achieves the lowest value, followed by MARL, RL, and Deterministic. Lower average waiting time improves the performance of traffic network.

states, as well as uses experience replay and target network to provide stable training, in the presence of multiple traffic signal controllers. A case study based on Sunway city and an investigation of the traditional grid traffic network show the effectiveness of our proposed scheme. Simulation of the Sunway city and the grid traffic network using SUMO and MATLAB demonstrates that MADQN outperforms other state-of-the-art approaches, including single agent reinforcement learning (RL), multi-agent reinforcement learning (MARL), and the existing deterministic traffic signal controllers. Specifically, in the simulation, MADQN outperforms other state-of-the-art approaches by increasing throughput by up to 70%, as well as reducing the queue length by up to 75% and the waiting time by up to 70%.

Future research could be pursued to prioritize the experiences during experience replay for faster learning, and take account of other kinds of traffic disturbances, such as traffic collisions, that can increase the queue length of vehicles significantly.

Declaration of competing interest

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

Acknowledgments

This work was part of the project entitled “A Novel Clustering Algorithm based on Reinforcement Learning for the Optimization of Global and Local Network Performances in Mobile Networks” funded by the Malaysian Ministry of Education under Fundamental Research Grant Scheme FRGS/1/2019/ICT03/SYUC/01/1.

References

[1] W.F. Adams, Road traffic considered as a random series, Institution Civil Engineers J/UK/.

[2] B. Yin, M. Dridi, A. El Moudni, Traffic network micro-simulation model and control algorithm based on approximate dynamic programming, IET Intell. Transp. Syst. 10 (3) (2016) 186–196.

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

[4] Z. Li, P. Liu, C. Xu, H. Duan, W. Wang, Reinforcement learning-based variable speed limit control strategy to reduce traffic congestion at freeway recurrent bottlenecks, IEEE Trans. Intell. Transp. Syst. 18 (11) (2017) 3204–3217.

[5] S. Kumar, P. Shah, D. Hakkani-Tur, L. Heck, Federated control with hierarchical multi-agent deep reinforcement learning, arXiv preprint arXiv:1712.08266.

[6] K.-L.A. Yau, J. Qadir, H.L. Khoo, M.H. Ling, P. Komisarczuk, A survey on reinforcement learning models and algorithms for traffic signal control, ACM Comput. Surv. 50 (3) (2017) 34.

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

[8] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep Learning, Vol. 1, 2016.

[9] Z. Zhang, Zhang, Khelifi, Multivariate Time Series Analysis in Climate and Environmental Research, Springer, 2018.

[10] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT press, 2018.

[11] R.S. Sutton, A.G. Barto, Introduction to Reinforcement Learning, Vol. 135, MIT press, Cambridge, 1998.

[12] S. Araghi, A. Khosravi, M. Johnstone, D. Creighton, Q-learning method for controlling traffic signal phase time in a single intersection, in: Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on, IEEE, 2013, pp. 1261–1265.

[13] Y. Lasheng, A. Marin, H. Fei, L. Jian, Studies on hierarchical reinforcement learning in multi-agent environment, in: 2008 IEEE International Conference on Networking, Sensing and Control, IEEE, 2008, pp. 1714–1720.

[14] M. Tan, Multi-agent reinforcement learning: Independent vs. cooperative agents, in: Proceedings of the tenth International Conference on Machine Learning, 1993, pp. 330–337.

[15] K. Prabuchandran, H.K. An, S. Bhatnagar, Multi-agent reinforcement learning for traffic signal control, in: Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on, IEEE, 2014, pp. 2529–2534.

[16] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall PTR, 1994.

[17] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, arXiv preprint arXiv:1312.5602.

[18] H.M. Alhassan, J. Ben-Edigbe, Effect of rain on probability distributions fitted to vehicle time headways, Int. J. Adv. Sci. Eng. Inf. Technol. 2 (2) (2012) 144–150.

[19] A.J. Miller, A queueing model for road traffic flow, J. R. Stat. Soc. Ser. B Stat. Methodol. (1961) 64–90.

[20] W.D. Ashton, Distributions for gaps in road traffic, IMA J. Appl. Math. 7 (1) (1971) 37–46.

[21] A. Schuhl, The probability theory applied to distribution of vehicles on two-lane highways.

[22] A.D. May, Tra c Flow Fundamentals, University of California, Berkeley, Prentice-Hall, Inc., New Jersey.

[23] F.A. Haight, B.F. Whisler, W.W. Mosher, New statistical method for describing highway distribution of cars, in: Highway Research Board Proceedings, Vol. 40, 1961.

[24] A. Maurya, S. Dey, S. Das, Speed and time headway distribution under mixed traffic condition, J. East. Asia Soc. Transp. Stud. 11 (2015) 1774–1792.

[25] K. Prabuchandran, H.K. An, S. Bhatnagar, Decentralized learning for traffic signal control, in: 2015 7th International Conference on Communication Systems and Networks (COMSNETS), IEEE, 2015, pp. 1–6.

[26] P. Mannion, J. Duggan, E. Howley, An experimental review of reinforcement learning algorithms for adaptive traffic signal control, in: Autonomic Road Transport Support Systems, Springer, 2016, pp. 47–66.

[27] W. Genders, S. Razavi, Using a deep reinforcement learning agent for traffic signal control, arXiv preprint arXiv:1611.01142.

[28] E. van der Pol, Deep reinforcement learning for coordination in traffic light control, Master's thesis, University of Amsterdam.

[29] A. Vidali, L. Crociani, G. Vizzari, S. Bandini, A deep reinforcement learning approach to adaptive traffic lights management.

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

[31] L. Li, Y. Lv, F.-Y. Wang, Traffic signal timing via deep reinforcement learning, IEEE/CAA J. Autom. Sin. 3 (3) (2016) 247–254.

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

[33] C.-H. Wan, M.-C. Hwang, Value-based deep reinforcement learning for adaptive isolated intersection signal control, IET Intell. Transp. Syst. 12 (9) (2018) 1005–1010.

[34] N. Krishnan, M. Prasanna, H. Vijith, Fluctuations in monthly and annual rainfall trend in the limbang river basin, malaysia: A statistical assessment to detect the influence of climate change, J. Clim. Change 4 (2) (2018) 15–29.

[35] A. Syakura, M. Ab Kadir, C. Gomes, A. Elistina, M. Cooper, Comparative study on lightning fatality rate in malaysia between 2008 and 2017, in: 2018 34th International Conference on Lightning Protection (ICLP), IEEE, 2018, pp. 1–6.

[36] L. Prashanth, S. Bhatnagar, Threshold tuning using stochastic optimization for graded signal control, IEEE Trans. Veh. Technol. 61 (9) (2012) 3865–3880.

[37] M.A. Khamis, W. Gomaa, H. El-Shishiny, Multi-objective traffic light control system based on bayesian probability interpretation, in: 2012 15th International IEEE Conference on Intelligent Transportation Systems, IEEE, 2012, pp. 995–1000.

[38] T. Chu, J. Wang, L. Codecà, Z. Li, Multi-agent deep reinforcement learning for large-scale traffic signal control, IEEE Trans. Intell. Transp. Syst..

[39] J. Gao, Y. Shen, J. Liu, M. Ito, N. Shiratori, Adaptive traffic signal control: Deep reinforcement learning algorithm with experience replay and target network, arXiv preprint arXiv:1705.02755.

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

[41] M. Behrisch, L. Bieker, J. Erdmann, D. Krajzewicz, Sumo–simulation of urban mobility: an overview, in: Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation, ThinkMind, 2011.

Faizan Rasheed received the Bachelor's degree in electronics from Isra University, Pakistan, in 2018. He is currently pursuing the Master's degree in computer science at Sunway University, Malaysia, under the joint program of Sunway University, Malaysia and Lancaster University, UK. His research interests lie in the domain of intelligent transportation system, machine learning, artificial intelligence, and robotics.

Kok-Lim Alvin Yau received the B.Eng. degree (Hons.) in electrical and electronics engineering from Universiti Teknologi Petronas, Malaysia, in 2005, the M.Sc. degree in electrical engineering from the National University of Singapore in 2007, and the Ph.D. degree in network engineering from the Victoria University of Wellington, New Zealand, in 2010.

He is currently a Professor with the Department of Computing and Information Systems, Sunway University. He is also a Researcher, a Lecturer, and a Consultant in 5G, cognitive radio, wireless networks,

applied artificial intelligence, and reinforcement learning. He serves as a TPC member and a reviewer for major international conferences, including ICC, VTC, LCN, GLOBECOM, and AINA. He was a recipient of the 2007 Professional Engineer Board of Singapore Gold Medal for being the best graduate of the M.Sc. degree in 2006/2007.

Dr. Yau serves as an Associate Editor for the IEEE ACCESS, an Editor of the KSII Transactions on Internet and Information Systems, a Guest Editor of the Special Issues of IEEE ACCESS, IET Networks, IEEE Computational Intelligence Magazine, and the Springer Journal of Ambient Intelligence and Humanized Computing, and a regular reviewer for over 0 journals, including the IEEE journals and magazines, the Ad Hoc Networks, the IET Communications, and others. He also served as the General Co-Chair of the IET ICFCNA'14 and the Co-Chair of the Organizing Committee of the IET ICWCA'12.

Yeh-Ching Low received the B.Sc. degree (Hons.) in statistics from University of Malaya, in 2004, the M.Sc. degree in statistics in 2007 from University of Malaya, and the Ph.D. degree in statistics from the University of Malaya, in 2016.

Dr. Low is currently a lecturer at Sunway University, Selangor, Malaysia. Her research interest is in the area of count data analysis, Monte Carlo methods, applications of statistical inference and probabilistic models, Bayesian inference and statistics education. She also serves as a reviewer for international conferences such as the ISI World Statistics Congress 019 as well as for journals such as Computers and Industrial Engineering. Dr. Low has eight years' experience of teaching mathematics and statistics to undergraduate computing students. Currently she is a member of IEEE, Association for Computing Machinery and International Statistical Institute.