IET Intelligent Transport Systems

Research Article

IET Journals
The Institution of Engineering and Technology

Adaptive traffic signal control system using composite reward architecture based deep reinforcement learning

ISSN 1751-956X
Received on 17th July 2020
Revised 11th November 2020
Accepted on 4th January 2021
doi: 10.1049/iet-its.2020.0443
www.ietdl.org

Abu Rafe Md Jamil[1], Kishan Kumar Ganguly[1], Naushin Nower[1] ✉

[1]*Institute of Information Technology, University of Dhaka, Dhaka 1000, Bangladesh*

✉ *E-mail: naushin@iit.du.ac.bd*

Abstract: The increasing traffic congestion problem can be solved by an adaptive traffic signal control (ATSC) system as it utilises real-time traffic information to control traffic signals. Recently, deep reinforcement learning (DRL) has shown its potential in solving the traffic signal timing. However, one of the main challenges of DRL is to design a proper reward function and special attention needs for a multi-objective reward design. Since the feedback to the agent depends on the reward function, a proper design of reward function is needed for fast and stable learning. In this study, the authors proposed a new reward architecture called composite reward architecture (CRA) for multi-objective ATSC to optimise multiple objectives. It calculates multiple rewards in parallel for each action and applies the majority voting method to choose the desired action. Since the traffic signal of one intersection affects the adjacent intersections, a new coordination approach is proposed to get the overall smooth traffic flow. The proposed reward architecture CRA is compared with several existing reward functions used in the literature for different traffic scenarios. The new coordinated approach is compared with the non-coordinated approach. The authors demonstrated that the proposed approaches outperform the others concerning waiting time, halting the number of vehicles, and so on.

1 Introduction

Traffic jam has become a serious issue in many cities in the world as it affects people's daily life, fuel consumption, time, and air pollution. Travellers get stuck in their vehicles for almost seven billion extra hours (42 h per traveller) in the USA each year [1]. Further, it costs billions in Europe [2]. Inadequate traffic infrastructure, traffic signal mismanagement, increasing population, and so on are some of the main reasons for this problem. The increasing number of vehicles on the roads makes this problem more severe. Improving traffic infrastructure could be a solution to the problem. However, it is expensive, and without proper traffic signal management, it cannot be a standalone solution to the problem. An Intelligent Transportation System (ITS) [3] can be used for proper traffic management by using real-time traffic information where infrastructure can be connected and communicated with each other. An ITS provides information, communication, and sensor technologies to vehicles and transportation infrastructure in order to provide real-time traffic information for road users and transportation system operators to make better decisions. An ITS mainly focuses on five application areas, Advanced Traveller Information Systems, Advanced Transportation Management System (ATMS), ITS-Enabled Transportation Pricing Systems, Advanced Public Transportation Systems, and Vehicle-to-Infrastructure Integration and Vehicle-to-Vehicle Integration [4]. The ATMS improves the flow of vehicles on the road by considering adaptive traffic signal and phase timing rather than the fixed signal and phase. It uses real-time traffic data from road intersections and makes dynamic traffic signalling based on the incoming data [5].

Traditionally, traffic signals are controlled by predefined fixed time plan [6–8]. The fixed time is determined by historical traffic demand for all green traffic signal phases without considering real-time traffic fluctuation. There is another type of traffic signal control system named vehicle-actuated method [9, 10]. In this approach, the traffic signal timing is determined by only the current traffic scenario. The main difficulty with various actuated control systems is that they adjust the signal phase lengths in response to traffic flow without any optimisation [11]. On the other hand, adaptive traffic signal control (ATSC) adapts to real-time traffic patterns to optimise the traffic flow by dynamically changing the traffic phases. It is currently the most demanding traffic control system. In this system, the signal timing plan utilises the current traffic conditions measured by a detection system.

ATSC has recently gained the attention of the researcher because it can regulate the road traffic signal more intelligently by using real-time traffic information (queue length of lanes, waiting time of the vehicles etc.). Most of the recent studies used deep reinforcement learning (DRL) to train the traffic controller. The traffic signal control problem is such a problem where the proper decision is unknown to everyone. It is only possible to know the effect of the action after taking action in the environment. If the action improves the environment then the action can be labelled as good action in that particular condition and labelled as bad otherwise. This particular characteristic of the traffic signal control problem fits very well in the deep Q-learning network (DQN) which is a form of DRL. Therefore, recent studies have mostly used DQN to implement their systems [12–15]. In these studies, current road traffic conditions are considered as a state, and traffic signal modification is considered as an action. After taking an action, the traffic scenario changes and a scalar reward value is gained from the environment as shown in Fig. 1. This scalar reward plays an important role in the action taken as it can be thought of as an evaluation of the action. Thus, the goal of an reinforcement learning (RL) agent is to maximise the reward collected from the environment over the time. More specifically, reward function defines the problem the agent is trying to solve, thus it implicitly describes the optimal behaviour also. As a result, the proper choice of reward function can have a strong effect on how long the agent learns the optimal behaviour. Therefore, the success of the algorithms depends on designing a proper reward function [16].

Reward function can be divided into two categories – single objective and multi-objective. In a single objective reward function, one parameter is used for optimisation. However, many decision making and adaptive control problems in the real world have multi-objective characteristics. As a result, the multi-objective reward function is more preferable than a single objective reward function since it optimises more than one objective at a time where each objective has its reward. Multi-objective reward function is more suitable for traffic signal control [17] as it optimises multiple parameters at a time and makes the learning faster. It works well

© The Institution of Engineering and Technology 2021

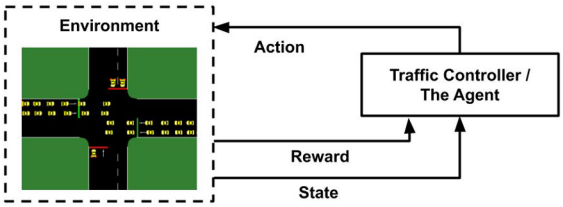

Fig. 1 *Basic concept of ATSC with DRL*

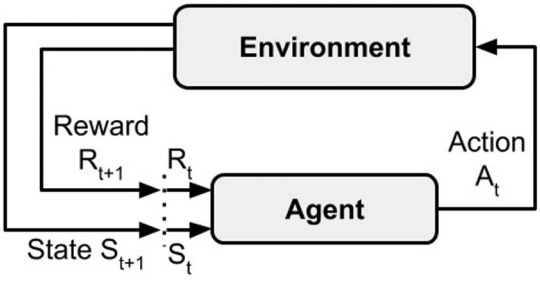

Fig. 2 *Agent–environment interaction in a MDP [31]*

when all the parameters are completely distinct then they can be optimised and combined together to get the final reward. However, the studies in [16, 18] experimentally showed that due to the correlation among the parameters in multi-objective reward functions it cannot optimise the parameters properly. Sometimes, a multi-objective reward function misleads the algorithms by giving inaccurate reward value because of combining conflicting objectives. If there are conflicting objectives, any policy can only maximise one of the objectives, or realise a trade-off among the conflicting objectives [11]. Besides these two basic reward functions, a derived reward function called hybrid reward architecture (HRA) [19] decomposes the reward function of the environment into n different reward functions. Each of them is assigned to a separate agent and each agent gives an action value. The final action is selected based on aggregated values from all the agents. This technique works well for some games (e.g. ATARI game, Pac-Man game etc.) where there are many possible actions [20]. Moreover, aggregated value may also affect the reward if parameters are correlated.

To solve this problem, composite reward architecture (CRA) is proposed for ATCS. In this approach, multiple objectives are optimised and reward value is calculated for each objective separately. Each reward function optimises one traffic feature (e.g. halting number of vehicles, waiting time, delay etc.) and provides a Q-value for a given state according to their Q-value functions. The action of the learning agent is determined by the Q-value. Since there are several Q-values for the different reward functions, there can be several decisions for taking action. The agent's final action is selected according to the majority voting method considering all the reward function's decisions. In this approach, each reward function optimises one traffic feature and since the reward functions are calculated independently, this approach does not suffer from the correlation of the parameters problem. Furthermore, it optimises multiple traffic features at a time by considering majority's decision.

Moreover, in real-world traffic, one intersection's traffic signal influences the nearby intersections, particularly when the intersections are geographically close and the traffic flow is high. Therefore, each agent faces a moving-target learning problem because the optimal policy changes as the others change their policies. This reinforces the need for coordination among agents [21]. According to the reviewed literature, many studies considered independent learning agent [13, 22, 23]. They have proposed solutions only for an isolated road intersection, where an isolated intersection in real life is impractical. There exist some studies [24, 25] that used coordination based multi-agent reinforcement learning [26]. However, they suffer from the curse of dimensionality problem as the state space grows exponentially with the number of agents. To improve the overall performance of the control system, traffic information needs to be coordinated only from the neighbouring intersections [27]. There are some approaches [21, 23, 28, 29] which have considered the neighbour intersections' traffic condition for predicting the present traffic phase. They proposed their solutions using a multi-agent distributed system. In these approaches, each agent controls the traffic signal of one road intersection and one agent communicates with its neighbour agents to collect their traffic status. The agent chooses its traffic phase considering the neighbour's traffic condition. Thus, one agent depends on others to take action, which creates dependency and communication overhead. If one agent can consider other information to decide without any communication, it will solve the dependency and communication overhead problem. To the best of the authors' knowledge, no work has ever done in this concept of considering neighbours' traffic conditions without any communication in a single agent.

To achieve the second goal, a new technique to coordinate the adjacent road intersections named coordinated adjacent intersection approach (CAIA) is designed. In this technique, one single agent can consider the traffic conditions of nearby intersections without any communication with others. It works with remote partial views. In this approach, one agent knows the traffic condition of both the incoming and outgoing lanes. The agent knows the neighbouring intersection's traffic condition by the scenario of the outgoing lanes. Since the agent gets a partial knowledge about the traffic condition of one road intersection by looking at the outgoing lane, it is called a partial view. The partial view of the nearby intersections is considered to choose the action for the current traffic phase. Further, as this method considers only the incoming and outgoing lanes traffic condition, it does not suffer from the curse of dimensionality problem.

Therefore, our approach extends the research in the ATSC by making two important contributions: (i) a new reward function architecture named CRA and (ii) an intersection coordinated CAIA approach using a single agent without communication. The proposed CRA is compared with six other reward functions in different traffic configurations. Among the six reward functions, three of them are single objective reward functions, one is multi-objective reward function, one is conventional method, and one is hybrid reward function HRA. The result shows that CRA outperforms the others in all the configurations in terms of waiting time and halting number. The proposed intersection coordinated approach CAIA is compared with the intersection non-coordinated approach. The intersection coordinated approach covers the whole metropolitan area and represents the realistic road scenario. The proposed CRA shows its effectiveness in the coordinated approach as well.

2 Background study

Due to the growing number of vehicles on roads, the ATSC has become a requirement of time [30]. Recent studies have preferred to use the RL approach to implement agents to control the traffic phase by changing the lights dynamically. A deep neural network is combined with RL to design the technique named as DRL. The deep neural network is used to maintain the state-action combination that increases exponentially in traffic signal control problem. This section describes the basic concept of RL and the combination of deep neural networks with it.

2.1 Reinforcement learning

RL is the learning process of how to map scenarios to actions to maximise a numeric reward value. The proper actions are unknown to the learner. The learner needs to find the actions which provide high reward values by trying them. The proper actions not only affect the immediate reward but also maximise the next subsequent rewards. The trial and error search and future rewards are the two most important features of RL [31].

The basic idea of RL is described in Fig. 2. An RL agent is deployed in an environment without any prior knowledge of how to behave. The agent observes the environment and takes action. The agent receives a scalar reward based on the outcome of the previously taken action. The reward value is the feedback of the

© The Institution of Engineering and Technology 2021

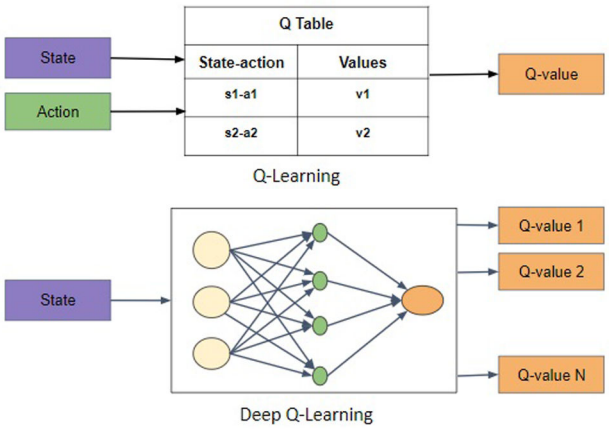

Fig. 3 *Q-learning and deep Q-learning*

environment for the agent's taken actions. The reward is a scalar real number value. It can be either negative or positive. According to the feedback, the agent can iteratively learn how to behave with the environment. Doing this, the agent gains some experience and knows the good actions. Q-values represent the expected reward for each state-action pair. With the q-values, the agent can decide which action is most suitable in a certain situation. The Q-values are stored in a tabular method using a matrix, which is the knowledge learned by the RL agent.

An RL problem is generally modelled as a Markov decision process (MDP), which formalises the problem of learning sequential decision making [32]. An MDP can be represented by the set of states S, set of actions A, a transition function T, and a reward function R create the tuple $< S, A, T, R >$. In any given state $s \in \mathrm{S}$, selecting an action $a \in A$ will change the environment to a new state $s' \in S$ with the probability $T(s, a, s') \in (0, 1)$ and the environment gives a reward $R(s, a)$. In an iterative process of getting rewards, the agent generates a policy that determines the mapping from a certain state to action ($\pi: S \times A \rightarrow [0, 1]$). The goal of an MDP is to find the best policy for an agent. The policy π defines the probability of selecting action a_t in state s_t. The policy is generated in such a way that, the expected cumulative discounted reward is maximised over time. The discounted future reward, R_t at time t is defined as follows:

$$R_t = E\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k}\right], \tag{1}$$

here, γ is the discount factor which indicates impact of future rewards.

RL can be divided into two categories: model-based and model-free. Q-learning is a model-free approach of RL which does not need the transition function and it is commonly used in ATSC. In this learning, the Q-values are updated according to the following equation:

$$\begin{aligned} 0.45!Q_{t+1}(s_t, a_t) &= Q_t(s_t, a_t) \\ &+\alpha(r_t + \gamma Q_{\max_a}(s_{t+1}, a) - Q_t(s_t, a_t)) \end{aligned} \tag{2}$$

where $r_t + \gamma Q_{\max_a}(s_{t+1}, a)$ is the target value, and $Q_t(s_t, a_t)$ is the predicted value. When training the neural network, we get the current Q-values ($Q_t(s_t, a_t)$) from the neural network and the reward value (r_t) from the environment. The maximum expected future reward value ($Q_{\max_a}(s_{t+1}, a)$) is calculated using the target network. The learning rate $\alpha \in [0, 1]$ is an input parameter for RL which determines by how much Q-values are updated at each time step t. Usually, the learning rate is initialised by a low value and may be decreased over time or may be constant [33]. A learning rate of $\alpha = 0$ would stop the learning process, which indicates the satisfactory result is reached. It is very difficult to determine the appropriate the value of alpha, because a very high value may result in instability of the solution and very low value may be very time consuming to reach an optimal goal. Instead of calculating the optimum learning rate analytically, a good enough learning rate must be discovered through trial and error. The discount factor $\gamma \in [0, 1]$ determines how the agent will take the future rewards in consideration. Generally, it is initialised with a high value. $\gamma = 0$ indicates that the agent is concerned only about immediate rewards, on the other hand very high value of γ indicates that the agent is more concerned with the future reward value [16]. All those values are used to generate real Q-values according to (2). In the back-propagation, the error is calculated using the real Q-values and network generated Q-values. Each layer of the neural network uses the error to update the weights and back-propagate the error to the previous layer.

One of the challenges that arise in RL is the trade-off between exploration and exploitation. After getting a state as input, an agent can have two different approaches to decide the action. The agent can take action from its previous experience (exploitation) or it can take a random action to gain new experience (exploration). The agent has to exploit what it has already experienced to obtain the higher reward, again it has to explore to make better action selections that produce high rewards in the future. The agent needs to balance the exploitation of experience and exploration of random actions to find some new experiences. This implies, the agent will not always choose an action from learned experiences, sometimes it takes random actions to gather new experiences. In RL, balancing the trade-off between exploration and exploitation is a crucial aspect. Exploration is necessary for a model-free learning approach. The exploitation of known good actions helps the agent to accumulate the maximum possible reward. Exploring random actions will allow the agent to find more possible actions that could perform better than the known actions. Excessive exploration limits the agent's capacity to accumulate rewards over its lifetime and may reduce performance. Softmax and ϵ-greedy algorithms are two commonly used approaches to balance the exploration and exploitation trade-off [34].

2.2 Deep learning and deep reinforcement learning

In RL, the reward value of each state-action pair is stored in a table. However, the state-action pair in the ATSC problem becomes so huge that managing them in a tabular method is infeasible. To solve this problem, many studies have used deep neural networks for function approximator. Neural networks are function approximators, which are particularly useful in RL when the state space or action space is too large to be completely known. In the deep neural network [35], there are many layers of neurons with weights. It can easily maintain the exponentially increasing state-action pair. The environment is fitted to the neural network with some numeric values as a state. In each layer l, the neural network multiplies the input value x with its weights W and adds some bias b. Then, the output x' is generated by applying a non-linear activation function σ (3) of the layer [36]

$$x' = \sigma_l(W_l x + b_l) \tag{3}$$

The output of one layer is sent as the input of the next layer. The output of the last layer is considered to be the expected Q-values (described in Fig. 3). The Q-values are compared with the real reward values and the mean squared error loss is calculated using (4). The loss is backpropagated from the output layer to the input layer. Each layer adjusts its weights to minimise the loss. According to the Q-values, an action is selected for the given state in the neural network

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Q_i - \hat{Q}_i\right)^2 \tag{4}$$

The basic concept of DRL for ATSC is shown in Fig. 1. The traffic scenario represents the environment. The state is represented by some numeric value of traffic features (queue lengths of all lanes, waiting time of vehicles, number of vehicles in all the lanes etc.). This network takes these values as input and generates some Q-

© The Institution of Engineering and Technology 2021

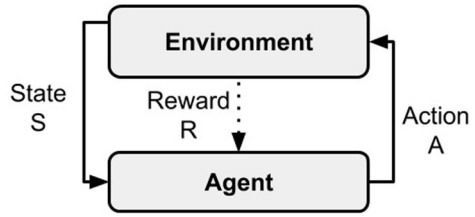

Fig. 4 *Single objective RL*

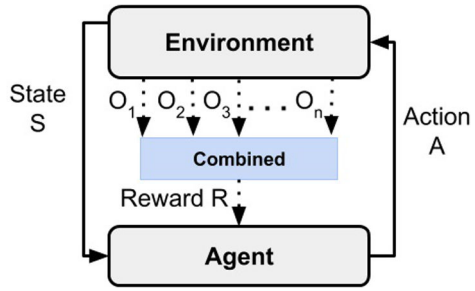

Fig. 5 *Multi-objective RL*

values. According to the Q-values, the agent takes an action to the environment. The environment changes after taking the action and the agent gets the real reward value. According to the real reward value, the weights of the network are adjusted following the aforementioned process. There is a memory to preserve the experience alongside the network. At each time step, the network saves its experience in the memory as (s_t, a_t, r_t, s_{t+1}) tuple. At the time of exploiting, it takes some random tuples from the memory and tries to maximise the accumulated reward. This approach is called experience replay [37].

One of the main challenges of DRL is the target continuously changes in each iteration. In deep learning, the target variable, i.e. is the desired output, does not change and hence the training is stable, which is not true for RL. In RL, a better solution is searched by exploring more. The input space is continually modified and the behaviour that is looked for. Therefore, the target value is updated at each step. Since the input and output are under constant modifications, the RL agents always chase a non-stationary target [12]. The problem occurred because of one single network generating the target and prediction (the second part of (2) multiplied by α) value. To solve this problem, a separate network can be used to estimate the target. This target network has the same architecture as the function approximator. After a fixed iteration (a hyperparameter), the parameters from the prediction network are copied to the target network. This leads to more stable training because it keeps the target function fixed for a while. This approach is called the target network.

3 Literature review

Recent research on RL has suggested many potential solutions to solve the problem of traffic signal control, where it reveals that various algorithms and design of neural networks can be used. In RL, the reward reflects the environmental feedback after an action has been taken by the agent. The agent uses the reward value to consider the result of the action taken and to update the model for future choice of action. Therefore, the reward is one of the most vital parts of the learning process. Furthermore, the traffic light phase of one road intersection affects the nearby intersections. For this reason, there is a need to coordinate the intersections to gain a smoother traffic flow on the roads. Therefore, the coordination of the intersections is another concern in ATSC. Based on the above discussion this section is divided into two sub-sections to discuss the existing literature with two aspects, reward function, and coordination of intersections.

3.1 Reward function

Although there are many techniques proposed for ATSC using RL in the literature, very few of them concern about designing a proper reward function. There are multiple ways to design a reward function. In the literature, the reward function is designed in three ways which are mentioned below:

(i) Single objective
(ii) Multi-objective
(iii) Hybrid reward

3.1.1 Single objective: A reward function is called a single objective reward function when it is designed with a single parameter as shown in Fig. 4. In this approach, the agent determines the action based on the single objective and gets a single reward value from the environment after taking the action. The single objective RL is shown in Fig. 4. The learning agent optimises the parameter that is used to design the reward function. In traffic signal control, the parameter can be the traffic feature of road intersection like queue length [13], waiting time [12, 38], delay [15, 39], and so on. Here delay is defined with the following equation:

$$D = 1 - \frac{\text{average speed of vehicles in lanes}}{\text{maximum allowed lane speed}} \tag{5}$$

This delay definition indicates the lack of speed that vehicles are failed to utilise compared to the maximum vehicles allowed speed on the lane. The generic form of the reward functions calculation is mentioned in the following equation:

$$R_{\text{TF},t} = \text{TF}_{t-1} - \text{TF}_t \tag{6}$$

where $R_{\text{TF},t}$ indicates the reward for traffic feature, TF_t is the traffic feature at time t, and TF_{t-1} is the traffic feature at time $t-1$.

All of the single objective approaches optimise the parameter that is used in the reward function. However, most of the real-life problems are multi-objective. Therefore, multiple traffic features are needed to optimise for smooth traffic flow in a real-life scenario.

3.1.2 Multi-objective: Most of the real-world problems are multi-objective that means it needs to optimise more than one parameter at a time. Here, each parameter defines an objective to optimise for the agent. Thus, a multi-objective reward function optimises more than one parameter at a time. In this approach, one single reward value is calculated by combining multiple objectives as shown in Fig. 5, where there are n objectives. The learning agent optimises the set of multiple objectives simultaneously. In traffic signal control, queue length, delay, waiting time, travel time, fuel consumption, and so on parameters could be combined for multi-objective optimisation. Different multi-objective reward functions are designed in [14, 16, 40]. For example, the reward function is designed in [14] by considering the sum of queue length L, the sum of delay D, the sum of updated waiting time W, an indicator of light switches C, the total number of vehicles passed N, and total travel time T of the passing vehicles

$$r_t = -0.25\sum_{i\in I} L_i - 0.25\sum_{i\in I} D_i - 0.25\sum_{i\in I} W_i - 5C + N + T \tag{7}$$

The generic form of the multi-objective reward function (8) is the weighted sum of the traffic features

$$\text{Reward} = \sum_{i=1}^{n} W_i \times \text{trafficFeature}_i \tag{8}$$

where W_i is the weight of ith traffic feature.

A different multi-objective reward function for traffic light optimisation is proposed in [41]. Their reward function takes into

© The Institution of Engineering and Technology 2021

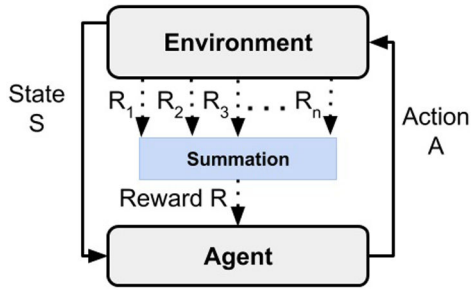

Fig. 6 *Hybrid reward architecture*

consideration both the traffic flow and traffic delay at the same time. The reward function is

$$R_t = D_{t-1} - D_t + \frac{\text{Occupancy}}{\text{Number of Halting Vehicles} + c} \tag{9}$$

Here, $c \in (0, 1)$ is used to prevent zero division. The above-mentioned literature did not provide any formal justification for the reward function. Further, the robustness of this reward function for different traffic condition is not mentioned. Thus, existing multi-objective reward functions calculate the reward by applying the weighted sum of different objectives. However, if the objectives are conflicting or correlated with each other, then the weighted sum approach does not work well [11].

3.1.3 Hybrid reward: HRA breaks down a task into smaller pieces in such a way that sub-tasks are directly relevant to the original task [19]. Although DRL works well in many domains, it is not easy to reduce the optimal value function to a low-dimensional representation, and learning can be very slow and unstable. To solve this problem, HRA breaks down a reward function into several components (Fig. 6) and learns a different value function for each component.

The reward function of HRA is defined as follows:

$$R^{\text{HRA}}(s, a, s') = \sum_{i=1}^{n} R_i(s, a, s') \tag{10}$$

for all state s, action a. Each agent has its own Q-value function, because of the individual reward function. Each agent can be viewed alternatively as a single DQN and agents can share multiple lower layers of a deep Q-network. The Q-value function of HRA is defined as the sum of all agents as follows:

$$Q^{\text{HRA}}(s, a; \theta) = \sum_{i=1}^{n} Q_i(s, a; \theta), \tag{11}$$

where θ are training parameters. For a specific state, the neural network predicts multiple Q-values for each action according to multiple Q-value functions. This Q-values are summed to get one value for each action and one action is selected based on the Q-values. This approach is well suited in some games [20] where there are many actions and one objective. However, this approach is not used in traffic signal control problem. In addition, the weighted sum of Q-values in HRA may combine the correlated traffic features that could hamper the best performance.

3.2 Coordination of intersections

Cooperation is crucial in the control of traffic signals because one signal's actions may influence the other, particularly when two intersections are geographically close, and the traffic flow is high. Strong coordination between the traffic signal agents helps the vehicles to move more smoothly across intersections [42]. The most straightforward way to achieve cooperation in RL is to notify the target agent through contact with local observations of other agents. When formulating multi-agent problems, there is a variation from centralised decision making (one agent collects all the information, makes a decision, and distributes that decision) to decentralised decision making (all agents have their local state and do not coordinate with others) [43].

In the centralised decision making, all other agents' data extends the central agent's knowledge for detailed information about the environment. The more detailed the observation is, the more parameters the central agent needs to estimate, it becomes computationally very complex, and the learning time gets longer. In the decentralised decision-making approach, each agent takes its own decision for its road intersection. The agents do not share any information with others. However, in traffic signal control, the effect of any agent's action also depends on the actions of other agents [21]. To get the overall smooth traffic, the agents need to coordinate with each other [44]. However, when the agents coordinated with each other, each agent needs to keep a set of tables whose size is exponential in the number of agents: $|S_1| \times \cdots \times |S_N| \times |A_1| \times \cdots \times |A_N|$, where S_i and A_i represent the state and action spaces for agent i, respectively. The state space grows exponentially with the number of agents. This problem is named as the curse of dimensionality [21].

Therefore, it is preferable to pick only the agents that are important for incorporation in the observation. It is the trade-off of the two extremes (centralised and decentralised) formally called as DCEE framework [45]. The adjacent agents information are coordinated for better traffic flows in [21, 23, 27–29]. Only the downstream information are added in [21, 28, 29] to create the state for action prediction. All neighbouring agents' information are added in [22, 23] and neighbours hidden states are added in [46].

However, in all of the above-mentioned approaches, the target agent depends on the other nearby agents to get their traffic information. It creates a dependency and communication overhead. If one agent can consider other information to decide without any communication, it will solve the dependency and communication overhead problem. From the best of the authors' knowledge, no work has done in this concept of considering neighbour's traffic condition in a single agent.

4 Proposed methodology

In this section, we propose a new reward architecture for DRL agent to solve traffic signal control problem and a new coordinated approach for the adjacent intersection without any communication. We first introduce the new reward architecture named CRA in Section 4.1. Then, the new coordinated approach CAIA is described in Section 4.2.

4.1 Composite reward architecture

In this section, the proposed CRA for ATCS is discussed as a multi-objective reward function design. The reward function is a critical parameter for the ATCS since it indicates whether the traffic signal improves the traffic condition or not. It is widely recognised that the system performance can be improved by designing proper reward function. However, multi-objective reward function design is even challenging because of the correlation among objectives. Existing multi-objective reward functions suffer from this problem because of combining different objectives. To solve this problem, rather than combining objectives, CRA emphasises their action decisions. In CRA, multiple objectives are optimised by multiple reward functions separately. Thus, several decisions are generated from multiple reward functions. For better decision making, the ensemble technique is known as a powerful approach as it solicits advice from many parties [47]. Thus, CRA combines the decisions from multiple reward functions by majority voting strategy.

The overview of the ATCS using the proposed reward architecture CRA is depicted in Fig. 7. The present traffic condition of the road intersection is sent to the agent as a state. The n traffic features are extracted from the environment to construct the state (details in Section 5.1). According to the given state, the agent chooses a decision of which traffic phase to apply to the intersection. After getting the state, the DRL agent generates Q-

© The Institution of Engineering and Technology 2021

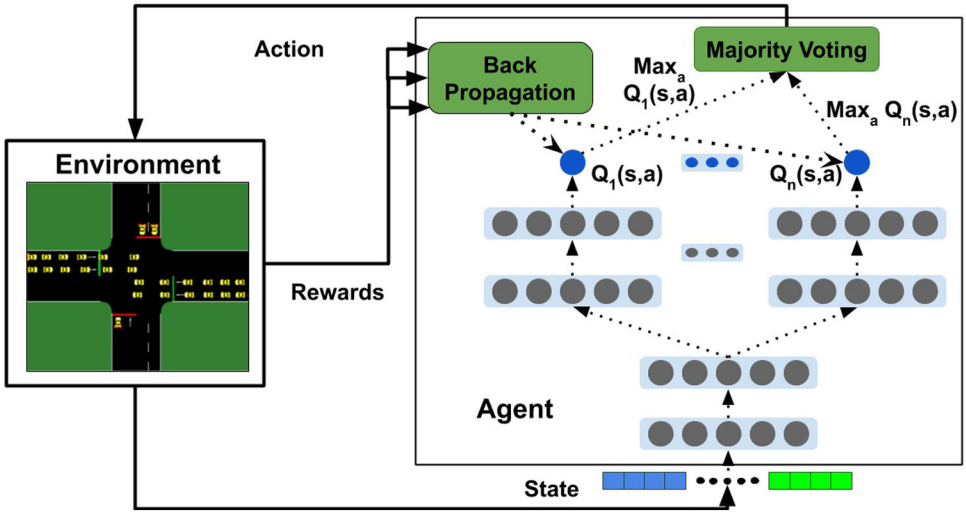

Fig. 7 *Overview of the ATCS with proposed CRA*

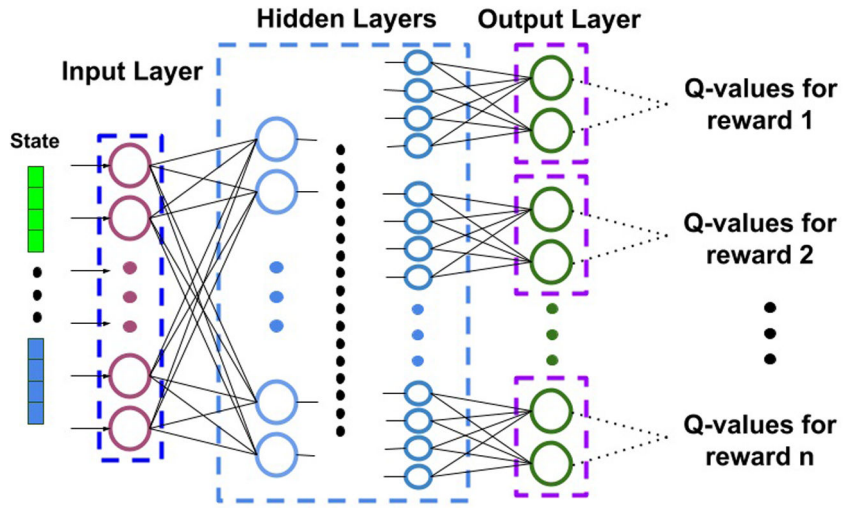

Fig. 8 *General neural network structure for CRA*

values for each available action according to the Q-value functions. The action with the highest Q-value is selected to apply in the environment. In this design, there are n reward functions for n objectives where each reward is a single objective function. Thus reward function in CRA is

$$R_{\text{CRA}} = \{R_1(s, a, s'), R_2(s, a, s'), \ldots, R_n(s, a, s')\} \tag{12}$$

for all s, a, s'. All the reward functions are designed with one traffic feature since they are single objective. There is a Q-value function for each reward function to predict the Q-values of different actions. The Q-values set is named as Q_{CRA} consisting of all the Q-values produced by the Q-value functions

$$Q_{\text{CRA}}(s, a; \theta) = \{Q_1(s, a; \theta), Q_2(s, a; \theta), \ldots, Q_n(s, a; \theta)\} \tag{13}$$

for all s, a and θ is the weight of the agent.

A deep neural network is used as a function approximator for all the Q-value functions as shown in Fig. 8. The neural network is designed based on the proposed method to fit any number of states and actions. The state is defined with n traffic features for eight lanes of the road intersection. Then the neural network has a total of $n \times 8$ nodes in the input layers. The number of nodes in the output layer will depend on the number of objectives and the number of actions. If we use x objectives to decide y action, then in the output layer there will be a total of $x \times y$ nodes. According to the number of nodes in the input and output layer, the number of hidden layers and the number of nodes in the hidden layers will need to be adjusted. As a result, the designed neural network is applicable in any traffic scenarios by adjusting the number of nodes accordingly. The network is defined to represent all the Q-values as CRA. At a time step t, the traffic condition from the environment is gained as state s_t. The state is sent to the network's input layer. The network produces n Q-value pairs for two actions (details in Section 5.2). Each Q-value pair gives a decision to take a specific action $a_t \in \{0, 1\}$ based on the highest Q-value. Therefore, there can be different opinions about which action to take. Finally, CRA performs majority voting approach by considering all the decisions made by the reward functions.

The majority voting approach for the final decision selection is described in the following equation:

$$\text{action} = \begin{cases} 0, & \left(\sum_{i=1}^{n} \text{argmax}(Q \text{ values}_i)\right) <= \dfrac{n}{2} \\ 1, & \text{otherwise} \end{cases} \tag{14}$$

In this research, we have two actions (0 or 1) and it considers decisions of different objectives which are either 0 or 1. Here, 0 means not to change the current traffic phase, and 1 means change the current traffic phase to the next phase. When the majority of the objectives (more than half) decide to change the present traffic condition, the final decision will be 1 otherwise the current traffic phase will continue. To do this, if we calculate the sum of the decisions we will get the number of decisions which are 1. This is done in the first case of (14) $\left(\left(\sum_{i=1}^{n} \text{argmax}(Q \text{ values}_i)\right) <= \dfrac{n}{2}\right)$. The 'argmax' operator gives the index number which has the highest Q-value for the decision of different objectives. The '$\sum_{i=1}^{n}$'

© The Institution of Engineering and Technology 2021

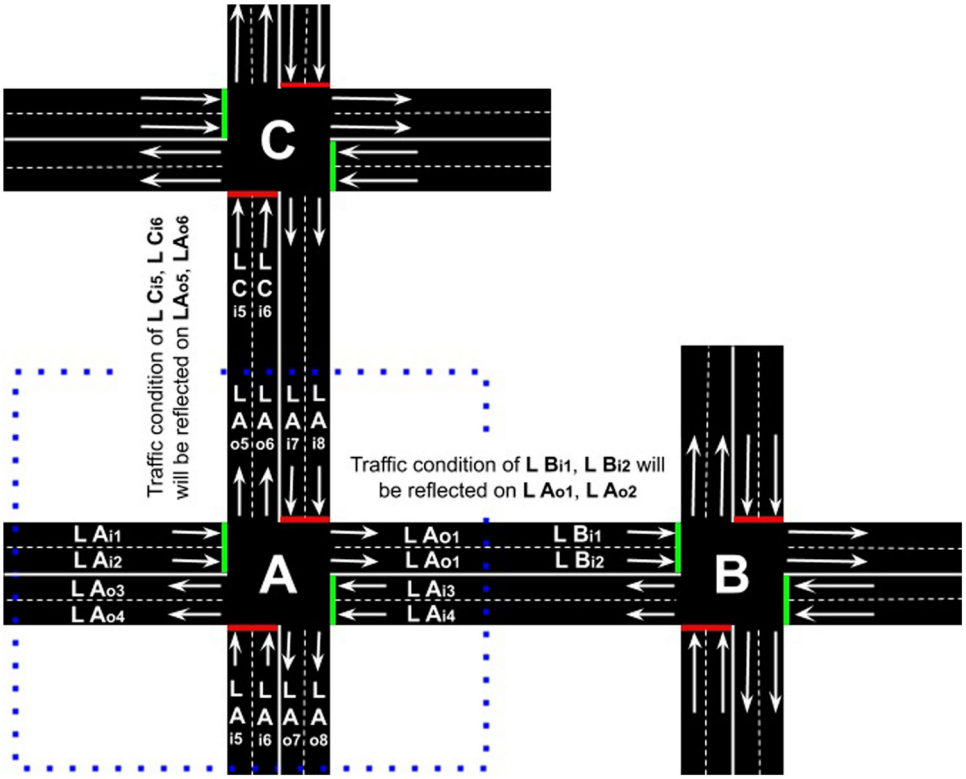

Fig. 9 *Road intersection for coordination*

operator adds the decisions of the objectives to get the number of decisions which are 1. The summation is checked whether it is less than or equal to half of n (n is the number of objectives) or not. If the condition is true, it selects the action as 0, and action 1 is decided if the condition is false. Additionally, n can be either even or odd. When we have a tie (i.e. the decisions for 0 and 1 are equal), it will not change the current traffic phase, rather, it will continue the current phase. The current traffic phase will be changed only if more than half of the objectives decide to change it. Moreover, using these two actions any number of traffic phases can be controlled. For example, for eight traffic phases, 0 means stay on the current phase and 1 indicates change to the next phase. In this manner, we can make an sequential traffic phasing by dynamically changing the duration of the traffic phases.

Each Q-value function minimises the loss function according to the following equation:

$$\text{Loss}_{n,i} = E\left[(y_{n,i} - Q_n(s,a;\theta_i))^2\right] \tag{15}$$

where

$$y_{n,i} = R_n(s,a,s') + \gamma \max_{a'} Q_n(s',a';\boldsymbol{\theta}_{i-1}) \tag{16}$$

where i is the iteration number and $\boldsymbol{\theta}_{i-1}$ is the weight vector from previous iteration. By minimising these loss functions, the CRA provides the optimal action-value functions under the different reward functions. Each reward function responds independently for the final action selection. Therefore, the CRA handles multiple parameters at a time without interaction among the parameters.

4.2 Coordinated adjacent intersection approach without communication

In traffic signal control, one intersection's traffic condition affects the adjacent intersections. One agent's optimal traffic signal can be changed suddenly by the change of neighbouring intersection's traffic phase. One of the major problems for the solutions that do not consider other road intersections traffic condition is cross-blocking [23]. It happens when the neighbouring intersection lanes are already filled with vehicles, and the traffic controller allows the vehicles to pass that intersection. In this scenario, every vehicle gets stuck, and no vehicles can pass the intersection. It hampers the overall performance of the traffic control system. Thus, for better traffic flow management, one agent needs to consider adjacent traffic condition. This is done by communicating with others in the existing literature. Our proposed CAIA without communication considers the adjacent traffic condition to choose the traffic phase without any communication. It takes the incoming and outgoing lanes information for global optimisation by providing intersection synchronisation and traffic flow management.

Consider Fig. 9, there are three intersections A, B, and C. Each intersection has eight incoming lanes and eight outgoing lanes. To control the traffic signal of intersection A in Fig. 9, the traffic information of the two adjacent intersections B and C is needed. This information is needed for intersection synchronisation and smooth traffic flow management, because the heavy traffic flow in one intersection affects its adjacent intersections also. Existing solutions exchange traffic information with the adjacent intersections by communicating with each other. Whenever the road intersections A and B are geographically close, then the traffic information of the outgoing lane of A can be used to predict the condition of the intersection B. The proposed approach considers these phenomena to get the condition of adjacent intersection B without any communication network. For example, in Fig. 9, the incoming lanes of A are denoted as $LA_{i1}, LA_{i2}, \ldots LA_{i8}$ and outgoing lanes are $LA_{o1}, LA_{o2}, \ldots LA_{o8}$. From the traffic information of LA_{o1} and LA_{o2}, the traffic condition of the incoming lane LB_{i1} and lane LB_{i2} can be predicted. If the outgoing lanes (LB_{i1} and LB_{i2}) are empty or have very few vehicles, it is assumed that the vehicles are passing intersection B smoothly. On the other hand, the outgoing lanes with long queues of vehicles imply that intersection B has huge pressure on vehicles, and the vehicles cannot pass smoothly. The same is applicable for the road intersection C. Therefore, the traffic condition of outgoing lanes gives the agent a partial view of the other intersections. This partial view is used in the proposed approach to decide the preferred traffic phase. The existing approaches [21, 23, 27–29] communicate with the neighbour intersections to get their traffic condition and consider the others traffic condition when choosing the preferred action. To avoid the communication we have used the partial views of others traffic condition. If the intersections are geographically close and the traffic pressure is high, the partial view can forecast the neighbour

© The Institution of Engineering and Technology 2021

Initialize weights, θ with random values
Allocate experience replay memory, M
for each epoch **do**
  initialize s with the current traffic condition
  repeat each step in the simulation
    choose action a, according to Equation 14 based on the Q-values provided by n Q-value functions
    take action a, observe reward r and next state s'
    store the tuple (s, a, r, s') in M
    $s \leftarrow s'$
    $b \leftarrow$ random samples from M
    for each sample(s_j, a_j, r_j, s'_j) in b **do**
      if s'_j is terminal **then**
        $y_i \leftarrow r_j$
      else
        $y_j = r_j + \gamma \max_{a'} Q(s'_j, a'; \theta_{i-1})$
      end if
      update weights θ
    end for
  until s is terminal
end for

Fig. 10 *Algorithm 1: DRL algorithm with CRA*

intersections traffic condition. With the hypotheses, in this approach, we need to know about the traffic condition of both the incoming lanes and outgoing lanes of any road intersection. The agent will consider the outgoing lanes traffic condition as the neighbours traffic condition and will choose action for the overall smooth traffic flow. Therefore, we have taken both the incoming lanes and outgoing lanes traffic condition to construct the combined state

$$\text{Combined State}, S_{\text{combined}} = \begin{bmatrix} S_{\text{IL}} & S_{\text{OL}} \end{bmatrix} \tag{17}$$

where S_{IL} is the state for the incoming lanes and S_{OL} is the state for the outgoing lanes. To get both states the agent needs to have access to all the lanes traffic data. Therefore, the traffic condition of all the lanes under the area in the blue box in Fig. 9 are known to the agent that controls the intersection A. The traffic condition of all the lanes in a road intersection is sent to the agent as a state to choose action (total 16 lanes as in Fig. 9). Algorithm 1 (Fig. 10) presents the pseudo-code for the learning of the agent.

5 System description

The ATSC problem is formulated as a DRL problem. In DRL, the agent interacts with the environment (road traffic scenario in this case). At each time step, the agent observes the environment, which is labelled as the state. According to the policy of the agent, it changes the traffic signal phase which is the action of the agent to the environment. Based on the action, the agent gets a scalar reward value from the environment as feedback. The reward value determines the effectiveness of the taken action. The agent saves the experience in a memory called experience replay memory [12]. The agent does the same process iteratively and gains experiences through a trial-error fashion. The state, action, design of reward functions, and neural network structure are described in this section.

5.1 State

The state represents the environment to the agent. Therefore, the state should contain as much information as possible about the environment. The state is designed in many ways in the literature. Some approaches take single traffic feature information for state [22, 48] and some consider multiple features [15, 38, 39, 49]. To give the agent more information about the environment, in CRA the state is designed with three traffic features information. The traffic features are waiting time of vehicles, the number of halting vehicles and delay (5) of the lanes. These three information for eight incoming lanes create the state with a size of 24 values. Therefore, the state of incoming lanes S_{IL}

$$S_{\text{IL}} = \begin{bmatrix} W_{\text{IL}} & Hn_{\text{IL}} & D_{\text{IL}} \end{bmatrix} \tag{18}$$

where W_{IL} is the waiting time, Hn_{IL} is the halting number of vehicles, and D_{IL} is the delay of incoming lanes.

For adjacent intersection coordination, three traffic features of the outgoing lanes alongside the incoming lanes are needed. There are a total of 16 lanes (in total) in the road intersection and three traffic features of all the lanes create the combined state with a size of 48 values. The state of the outgoing lanes is as follows:

$$S_{\text{OL}} = \begin{bmatrix} W_{\text{OL}} & Hn_{\text{OL}} & D_{\text{OL}} \end{bmatrix} \tag{19}$$

where W_{OL} is the waiting time, Hn_{OL} is the halting number of vehicles, and D_{OL} is the delay of the outgoing lanes.

5.2 Action

The traffic signal has two phases where in one phase the vehicles heading east and west across the intersection and the vehicles heading north and south across the intersection on the other phase. Thus, the agent has two different actions for the environment. In one action, the traffic phase will be green for the vehicles heading to east and west. In another action, the traffic light will be green for vehicles heading north and south. At time step t, the agent can take action $a_t \in A$, where $A = \{0, 1\}$. Here 0 means to stay on the current phase, and 1 indicates to change the traffic phase to the next phase. For simplicity, two-phase traffic scenario is used in this research; however, it is easy to extend the number of traffic phases such as four or six phases traffic with the two available actions. In that case, one action can be to stay in the current phase, and the other will be to get to the next phase. With this fashion, it is easy to create cyclic traffic phases with four, six, or more number of traffic phases.

5.3 Reward

The proposed CRA as a reward function with n parameters is used in this research, where there are n different reward functions for n objectives. In this paper, $n = 3$ is used. Three different reward functions are designed using three traffic features. Three traffic features are waiting time of vehicles (W), the number of halting vehicles (Hn), and delay (D) on lanes. For a smooth and comfortable travel experience, people want to wait for the minimum time in the road and want to travel as fast as possible, i.e. utilize the maximum capability of the road. Further, it is desirable to meet the minimum number of vehicles on the road to get the best travel experience. For this reason, the optimization of these three traffic features will allow travellers for a comfortable travel experience. Thus, these three traffic features are used for calculating reward function. The generic form of the reward functions calculation is mentioned in (6). The reward functions are calculated using waiting time, queue length, and halting number as the traffic feature in (6).

For all the traffic features used in this paper, a lower value is better for smooth traffic flow. The present traffic feature value is subtracted from the previous value in the reward functions. Therefore, if the traffic feature value decreases after taken action, the reward will be positive and it will be negative otherwise. In this process, the agent learns to take action that minimizes the traffic feature value.

5.4 Neural network structure

In this research, for CRA the state is defined with three traffic features for eight lanes of the road intersection. Therefore, the state is a vector of 24 values and these 24 values of the state are sent to the neural network as input. To take the input, the neural network has 24 nodes in the input layer. The designed neural network has a total of five layers (including input and output layer). Among the five layers, the first three layers are shared and the last two layers are specific for different objectives, i.e. they are not connected to each other. The second and third layers have 12 and 8 nodes, respectively. In the fourth layer, there are four nodes for each

© The Institution of Engineering and Technology 2021

objective (12 nodes for 3 objectives) and in the output layer, there are two nodes for each objective (six nodes for three objectives) to produce Q-values. The Sigmoid activation function is used in the hidden layers. The weights of the network are updated using the RMSProp optimizer [50].

6 Simulation and result analysis

The simulation setup and the results are discussed in this section.

6.1 Simulation setup

To validate the effectiveness of the proposed CRA and CAIA, we carry out several simulation tests. All experiments were conducted using the traffic micro-simulator Simulation of Urban MObility (SUMO) (Version 1.3.1) [51] and its Python API [52]. It allows the user to build a custom route topology, and also it helps to import different lane formats in many cities worldwide. The simulation has a four ways road intersection, and each road has two incoming and two outgoing lanes. The length of each road is 300 m. The maximum speed of the incoming lane can be 70 km/h, and the outgoing lane's maximum speed can be 40 km/h. In real-life road traffic scenarios, we have different speed limits of vehicles in different lanes. Travelling all the lanes at the same speed is not a practical scenario. When we are on a highway, generally a very high speed is allowed for the vehicles. However, in the residential area, the speed limit is obviously less than that of the highway. To take this phenomenon into consideration, we have designed the lanes with various speed limits. There are four different routes for the vehicles on the intersection. Vehicles can go (i) from West to East (W–E); (ii) from East to West (E–W); (iii) from North to South (N–S); and (iv) from South to North (S–N). For the CAIA, we have designed the simulation environment with five road intersections (for simplicity only three are shown in Fig. 9). We control the traffic signal of the middle intersection A, and the others are directly connected with the middle intersection. The vehicles are inserted into the environment from the other four intersections. The inserted vehicles have above-mentioned four routes to travel in each intersection. The vehicles can come to the central intersection after passing the other four intersections. The other attributes of the environment are the same as the single intersection.

The hyper-parameters used in the learning are mentioned in Table 1. The proposed method has experimented several times with different combination of learning rate (α), discount factor (γ), and ϵ. Among the combinations, the values in Table 1 produced the best result. Among the different combinations, a gradually decreasing learning rate is used in one combination. However, the learning rate value (0.005) outperformed the other values and the gradually decreasing learning rate. Therefore, 0.005 is used as the learning rate in the proposed method. Further, to solve the explore-exploit dilemma, the ϵ-greedy approach is used.

Table 1 Agent parameters

Agent parameters	Values
minimum time of action (s)	5
update period of target network (s)	5
learning rate	0.005
memory size	1000
sample size	300
training epochs	500
ϵ for exploration	0.05
discount factor γ	0.9

Table 2 Traffic configurations

Conf.	Routes	Arrival rate, Veh/s	Start time, s	End time, s
1	W–E	0.2	0	7200
	N–S	0.2	0	7200
2	W–E	0.2	0	7200
	N–S	0.4	0	7200
3	W–E	0.4	0	7200
	N–S	0.4	0	7200
4		Conf. 1	0	7200
		Conf. 2	7201	14,400
		Conf. 3	14,401	21,600
5	W–E	0.42	0	7200
	N–S	0.42	0	7200
6	W–E	0.42	0	7200
	N–S	0.55	0	7200
7	W–E	0.55	0	7200
	N–S	0.55	0	7200
8		Conf. 5	0	7200
		Conf. 6	7201	14,400
		Conf. 7	14,401	21,600

6.2 Dataset

To generate different traffic scenarios in the simulation, synthetic data is used. Eight traffic scenarios with different traffic flow rates are considered for simulation which are described in Table 2. The first four configurations are used to test the effectiveness of the proposed CRA. The first configuration has a steady, low traffic flow rate in all directions of the road. For all road directions, the vehicle arrival rate is 0.2 veh/s. The second configuration has an imbalanced vehicle's arrival rate at different road directions. The vehicle arrival rate in the East–West direction is two times less than the North–South direction. The third configuration is like the first configuration with steady traffic flow. However, the vehicle arrival rate is two times more than the first configuration in all road directions. The fourth configuration is the combination of the first three configurations. This configuration is closer to real-life traffic flow than the other configurations. In real-life traffic flow, the traffic pressure or vehicle arrival rate changes from time to time. Forth configuration represents that scenario.

The last four are used to test the CAIA using partial views. Since the coordinated approach is designed with the hypothesis of high vehicle pressure, the four configurations are designed with high arrival rates. These four configurations have higher vehicle arrival rates than the first four configurations but the traffic pattern remains the same. The fifth configuration has a steady traffic flow with an arrival rate of 0.42 veh/s. The sixth has imbalance arrival rates in the East–West and North–South direction. The seventh has also steady flow in all the directions but for a higher arrival rate than the fifth. The last configuration is the combination of fifth, sixth, and seventh configuration, where traffic flow changes after some time.

6.3 Performance metrics

To evaluate the performance and effectiveness of the proposed approach, three performance metrics are used. For smooth traffic flow, it is preferable to have a minimum waiting time and the minimum number of vehicles on the road. Considering these facts, the following three traffic features are selected as our performance metrics.

(i) Number of halting vehicles
(ii) Waiting time (minute)
(iii) Waiting time per vehicle (minute)

6.4 Compared methods

The proposed CRA is compared with six other methods to evaluate the improvement. The effectiveness of the CAIA using partial views is compared with the non-coordinated approach and Synchro [53].

(i) *Halting numbers as a single objective reward function*: In this method, the state and action are the same as our proposed method. Again, the parameters for the algorithm are also the same. The single objective reward function is designed with a different

© The Institution of Engineering and Technology 2021

number of halting vehicles in this approach. To the best of the author's knowledge, this reward function has not been used in any literature. However, the difference of queue lengths is used as reward function in [13]. The queue length can be calculated by adding the length of all vehicles that are halted in the lane. Therefore, these two features can be considered as an alternative for each other.

(ii) *Waiting time as a single objective reward function:* Waiting time is an important traffic feature that travellers want to minimize. Thus, waiting time as a reward function is used in many research [12, 38].

(iii) *Delay as a single objective reward function*: Delay as a reward function is designed using (6). This reward function is used in multiple studies [15, 39] since it is also a crucial parameter for minimize.

(iv) *HRA:* This approach is proposed in [19]. The reward function is decomposed into different reward functions in this approach. The reward functions have their own Q-value function and they produce different Q-values. To select the final action, the weighted sum of the Q-values are calculated.

(v) *IntelliLight:* This multi-objective reward function with six traffic features is proposed in [14].

(vi) *Synchro:* It is a macroscopic analysis and optimization software application [53].

Table 3 Result for configuration 1

Method	Hal. num	W-time	W-time/veh
halting number	4.61	0.63	0.107
waiting time [12, 38]	4.77	0.821	0.109
delay [15, 39]	14.9	89	0.39
HRA [19]	4.59	0.61	0.107
IntelliLight [14]	11.84	14.33	0.124
Synchro [53]	6.52	10.8	0.226
CRA (proposed)	**4.57**	**0.606**	**0.106**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 4 Result for configuration 2

Method	Hal. num	W-time	W-time/veh
halting number	203.26	1306.39	5.13
waiting time [12, 38]	199.89	1232.41	4.89
delay [15, 39]	197.46	1211.1	4.83
HRA [19]	191.18	1341.01	5.16
IntelliLight [14]	200.05	163.4	0.709
Synchro [53]	104.3	124.0	0.45
CRA (proposed)	**37.47**	**18.06**	**0.206**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 5 Result for configuration 3

Method	Hal. num	W-time	W-time/veh
halting number	250.18	1676.26	6.11
waiting time [12, 38]	242.44	1738.89	6.34
delay [15, 39]	242.18	1551.14	5.69
HRA [19]	236.35	1512.59	5.56
IntelliLight [14]	205.88	215.47	0.88
Synchro [53]	231.8	240.0	0.96
CRA (proposed)	**190.07**	**29.17**	**0.152**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 6 Result for configuration 4

Method	Hal. num	W-time	W-time/veh
halting number	179.8	3440.61	12.19
waiting time [12, 38]	174.89	2642.63	9.45
delay [15, 39]	225.1	3044.27	11.08
HRA [19]	116.41	103.55	0.544
IntelliLight [14]	**77.07**	86.18	0.41
CRA (proposed)	106	**47**	**0.309**

Bold values indicate the best value for the performance metric in the traffic configuration.

6.5 Result

The proposed CRA is compared with six other approaches which have been mentioned in Section 6.4. Tables 3–6 describe the results for three performance metrics. Three important traffic features (halting number, waiting time, and delay) are chosen as a single objective reward function for comparative analysis. These three parameters are used in CRA. Besides these, HRA, Synchro, and another multi-objective reward function are compared with ours for evaluation. The lower the better for every performance metrics values in the following tables.

Table 3 describes the result for traffic configuration 1. The configuration is created with an equal and low vehicle arrival rate in all the road directions. From Table 3, it is observed that all the methods perform competitively besides the reward designed with delay. The configuration has a low vehicle's rate, therefore all the performance metrics values are small which indicates smooth traffic flow. CRA considers multiple traffic features to choose the desired action, thus it outperforms the others in terms of all the performance metrics. Although IntelliLight [14] optimizes multiple traffic features, it cannot perform well in this configuration because of combining objectives without considering the relation between them.

The result for traffic configuration 2 is described in Table 4. The configuration is created with an unequal and high vehicle arrival rate for the routes. Therefore, in one road direction, there will be always high traffic pressure. If we choose the action according to the vehicle's pressure only, one road direction will always be open to cross the intersection and the other will be neglected. Analysing the result, we can say that the proposed method outperforms the other six methods as it considers multiple traffic features as a reward and considers their decisions by majority voting to choose action. CRA beats the others in terms of all the three performance metrics. The halting number of the other five methods (except Synchro [53]) is almost five to six times higher than CRA. The Synchro [53] performed the second best in this configuration. Furthermore, the lowest values for waiting time for other compared methods are almost seven times higher than CRA. Besides, the lowest value of waiting time/veh is more than double compared to CRA.

Configuration 3 is generated by an equal and high vehicle rate for all the routes. In this configuration, the traffic pressure is two times higher than configuration 1. Therefore, the performance metrics values are expected to be higher than the values for configuration 1. The result for this configuration is described in Table 5. The proposed method outperforms all the methods for all the performance metrics by providing significantly lower values. The other five methods cannot perform competitively against CRA. The first three methods consider one parameter at a time, thus it cannot beat CRA. HRA performs well compared to the single objective reward functions but it cannot outperform the multi-objective reward functions. Although IntelliLight [14] combines multiple features, it fails to provide the best result because of a simple combination. CRA shows the effectiveness over others by considering multiple features in choosing the desired action. Although the Synchro [53] produced better result than other four methods it could not beat IntelliLight [14] and CRA.

Configuration 4 is a combination of the first three configurations. In this configuration, the traffic flow rate changes after a fixed time. The configuration is designed for three times longer time steps to the other three configurations. Therefore, all the performance metrics values in the result are expected to be higher than the other three results. At a glance, it seems that all the methods performed closely in this configuration. IntelliLight [14] beats the others in terms of halting number. The proposed method

© The Institution of Engineering and Technology 2021

Table 7 Result for configuration 5

Method	Hal. num	W-time	W-time/veh
Synchro [53]	298.06	3299.4	9.07
CRA without coordination	**273.93**	2186.37	6.2
CRA with coordination	282.82	**1467.56**	**4.27**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 8 Result for configuration 6

Method	Hal. num	W-time	W-time/veh
Synchro [53]	296.41	3225.8	8.65
CRA without coordination	286.67	2077.33	5.91
CRA with coordination	**278.43**	**1508.07**	**4.26**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 9 Result for configuration 7

Method	Hal. num	W-time	W-time/veh
Synchro [53]	288.22	2431.32	8.85
CRA without coordination	**281.34**	1856.36	5.34
CRA with coordination	286.28	**1631.83**	**4.64**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 10 Result for configuration 8

Method	Hal. num	W-time	W-time/veh
CRA without coordination	318.75	4458.35	12
CRA with coordination	**216.28**	**29.11**	**0.14**

Bold values indicate the best value for the performance metric in the traffic configuration.

Table 11 Result for computation time

Method	Computation time, ms
halting number	0.4325
waiting time [12, 38]	0.5353
delay [15, 39]	0.4818
HRA [19]	0.5976
IntelliLight [14]	2.9244
CRA (proposed)	0.5899
CRA with CAIA (proposed)	0.9443

(CRA) beats the others in terms of waiting time and waiting time/veh. For a better travel experience, everybody wants to wait a minimum of time on the road. Further, it is preferable to have a minimum number of vehicles waiting in the queue to pass road intersection. However, sometimes there can be a large number of vehicles waiting for high vehicle pressure on the road. In this case, ensuring the minimum waiting time for vehicles becomes more preferable than the long queue of waiting vehicles. When the vehicles wait for a long time, it hampers the smooth traffic flow. Therefore, we can pay more attention to waiting time rather than the length of the vehicle's queue when there is high vehicle pressure on the road. Keeping this fact in mind, we can conclude that the proposed method performs better than the others in this configuration. The proposed method manifests two times lower waiting time than IntelliLight [14]. It was not possible to implement this configuration in Synchro [53]. Therefore, five methods are compared rather than six in Table 6.

Our proposed coordinated approach CAIA is compared with non-coordination approach and Synchro [53]. The performances are described in Tables 7–10.

All the methods (the coordinated, non-coordinated, and Synchro [53]) are simulated on a five intersections traffic environment. Combined state (17) is used to choose action in the coordinated approach.

The proposed coordinated approach CAIA ensures lower waiting time than the non-coordinated approach and Synchro [53] in all the traffic configurations. In configurations 5 and 7, the non-coordinated approach produces better value for halting number of vehicles. However, the waiting time values are much higher than the coordinated approach. If the controller allows the vehicles to pass from only one direction (East–West or North–South), the other road intersection will be filled with vehicles. In this case, the halting number of vehicles reaches a maximum number (since lane size is fixed). Since the halting number reaches a fixed maximum, the average value gets to a fixed number for eight lanes. Although the halting number does not increase after the lanes are filled, the waiting time values keep on increasing. Whenever the road intersections are geographically close, this exact scenario happens for traffic configurations 5 and 7. Although a lower halting number of vehicles is an indicator for smooth traffic flow, a lower waiting time becomes more preferable than a lower halting number. In this scenario, the vehicles get stuck in one direction for a long time. This hampers the smooth travel experience of the vehicles that are travelling through the road direction where the vehicles are stuck. Therefore, the coordinated approach CAIA produces a more preferable traffic scenario than the non-coordination approach in configurations 5 and 7.

The coordinated approach CAIA outperforms the non-coordinated approach in terms of all performance metrics in configurations 6 and 8. Since the real-life traffic flow changes dynamically, all the configurations are constructed to design different traffic flows in the road. Among all the configurations, configuration 8 is closer to real-life traffic conditions because of the traffic flow changes after sometimes. However, like configuration 4, configuration 8 was not possible to implement in Synchro [53]. Therefore, Table 10 compares only the non-coordinated and coordinated approach. The other configurations are created to test a particular traffic scenario. In this case, waiting time values are much lower for the coordinated approach. This implies that the proposed method can maintain the traffic signal in such a way that it will ensure the overall smooth traffic flow. Further, this shows the effectiveness of the partial views to avoid communication for coordination in adjacent intersections.

Furthermore, the proposed method is compared with others in terms of computation time in Table 11. From the result, it is observed that the single objective approaches (halting number, delay [15, 39], waiting time [12, 38]) are faster than the multi-objective approaches (CRA, HRA [19], Intellilight [14]). This is obvious since the multi-objective approaches need more computation than a single objective to make a decision. Among the multi-objective approaches, our method (CRA) is faster than the other methods. Moreover, the coordination approach (CAIA) is faster than Intellilight [14]. It proves that the proposed method can produce better results faster than the other multi-objective methods.

7 Conclusion

In this paper, we have designed and evaluated a new reward architecture CRA for an ATSC system that uses DRL. The proposed architecture is introduced to solve the correlation among parameters problem for multi-objective reward function. Furthermore, we have introduced a new technique CAIA for coordination among adjacent intersections. Our technique does not require communication with adjacent intersection's agent and it is designed to avoid dependency while choosing an action. We have shown through simulation that the proposed methods performed better than the others by providing smooth traffic flow.

We would like to acknowledge our current approach's limitations and point out some important future directions. Firstly, our approach is tested on a simulation. Therefore, the received results are simulated. A real-world experiment can help to gain real-life feedback and validate the applicability of the proposed reward function and coordinated method. Secondly, our simulated road intersection has only two traffic phases. This can be extended to four or more traffic phases for more realistic road intersection representation. Lastly, we have proposed and evaluated a new reward function CRA for ATSC. However, the proposed CRA can

© The Institution of Engineering and Technology 2021

be applied to the other applications of multi-objective RL. Further analysis is needed in this aspect.

8 References

[1] Scorecard, U.M.: 'The texas a&m transportation institute and inrix', Inc, USA, 2015, **9**, (2015), p. 10

[2] Economist, T.: 'The cost of traffic jams', 2014, https://www.economist.com/the-economist-explains/2014/11/03/the-cost-of-traffic-jams, (Accessed on 17 02 2020)

[3] Alam, M., Ferreira, J., Fonseca, J.: 'Introduction to intelligent transportation systems', In: Alam, M., Ferreira, J., Fonseca, J. (Eds.): '*Intelligent transportation systems*' (Springer, Switzerland, 2016), pp. 1–17

[4] Sussman, J.S.: '*Perspectives on intelligent transportation systems (ITS)*' (Springer Science & Business Media, USA, 2008)

[5] Ezell, S.: 'Explaining international it application leadership: intelligent transportation systems', 2010

[6] Dion, F., Rakha, H., Kang, Y.S.: 'Comparison of delay estimates at under-saturated and over-saturated pre-timed signalized intersections', *Transp. Res. B, Methodol.*, 2004, **38**, (2), pp. 99–122

[7] Miller, A.J.: 'Settings for fixed-cycle traffic signals', *J. Oper. Res. Soc.*, 1963, **14**, (4), pp. 373–386

[8] Webster, F.V.: 'Traffic signal settings'. Road Research, London, 1958

[9] Cools, S.B., Gershenson, C., D'Hooghe, B.: 'Self-organizing traffic lights: A realistic simulation', In Prokopenko, M. (Eds.): '*Advances in applied self-organizing systems*', (Springer, UK, 2013), pp. 45–55

[10] Porche, I., Lafortune, S.: 'Adaptive look-ahead optimization of traffic signals', *J. Intell. Transp. Syst.*, 1999, **4**, (3–4), pp. 209–254

[11] Liu, C., Xu, X., Hu, D.: 'Multiobjective reinforcement learning: a comprehensive overview', *IEEE Trans. Syst., Man, Cybern.: Syst.*, 2014, **45**, (3), pp. 385–398

[12] Gao, J., Shen, Y., Liu, J., *et al.*: 'Adaptive traffic signal control: deep reinforcement learning algorithm with experience replay and target network', arXiv preprint arXiv:170502755, 2017

[13] Li, L., Lv, Y., Wang, F.Y.: 'Traffic signal timing via deep reinforcement learning', *IEEE/CAA J. Autom. Sinica*, 2016, **3**, (3), pp. 247–254

[14] Wei, H., Zheng, G., Yao, H., *et al.*: 'Intellilight: A reinforcement learning approach for intelligent traffic light control'. Proc. of the 24th ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining, 2018, pp. 2496–2505

[15] Genders, W., Razavi, S.: 'Using a deep reinforcement learning agent for traffic signal control', arXiv preprint arXiv:161101142, 2016

[16] Mannion, P., Duggan, J., Howley, E.: 'An experimental review of reinforcement learning algorithms for adaptive traffic signal control', In Leo McCluskey, T., Kotsialos, A., Müller, J.P., *et al.* (Eds.): '*Autonomic road transport support systems*' (Springer, Switzerland, 2016), pp. 47–66

[17] Brys, T., Harutyunyan, A., Vrancx, P., *et al.*: 'Multi-objectivization of reinforcement learning problems by reward shaping'. 2014 Int. Joint Conf. on Neural Networks (IJCNN), Beijing, China, 2014, pp. 2315–2322

[18] Jamil, A.R.M., Ganguly, K.K., Nower, N.: 'An experimental analysis of reward functions for adaptive traffic signal control system'. 2020 Int. Conf. on Distributed Sensing and Intelligent System (ICDSIS), University of North Texas, USA, 2020

[19] Van.Seijen, H., Fatemi, M., Romoff, J., *et al.*: 'Hybrid reward architecture for reinforcement learning'. Advances in Neural Information Processing Systems, Long beach, CA, United States, 2017, pp. 5392–5402

[20] Takano, Y., Ouyang, W., Ito, S., *et al.*: 'Applying hybrid reward architecture to a fighting game ai'. 2018 IEEE Conf. on Computational Intelligence and Games (CIG), Masstricht, The Netherlands, 2018, pp. 1–4

[21] El Tantawy, S., Abdulhai, B., Abdelgawad, H.: 'Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): methodology and large-scale application on downtown toronto', *IEEE Trans. Intell. Transp. Syst.*, 2013, **14**, (3), pp. 1140–1150

[22] Wiering, M.: 'Multi-agent reinforcement learning for traffic light control'. Machine Learning: Proc. of the Seventeenth Int. Conf. (ICML'2000), 2000, pp. 1151–1158

[23] Arel, I., Liu, C., Urbanik, T., *et al.*: 'Reinforcement learning-based multiagent system for network traffic signal control', *IET Intell. Transp. Syst.*, 2010, **4**, (2), pp. 128–135

[24] Claus, C., Boutilier, C.: 'The dynamics of reinforcement learning in cooperative multiagent systems', *AAAI/IAAI*, 1998, **1998**, (746–752), p. 2

[25] Weinberg, M., Rosenschein, J.S.: 'Best-response multiagent learning in nonstationary environments'. Proc. of the Third Int. Joint Conf. on Autonomous Agents and Multiagent Systems-Volume 2. (IEEE Computer Society, 2004, pp. 506–513

[26] Tan, M.: 'Multi-agent reinforcement learning: independent vs. Cooperative agents'. Proc. of the Tenth Int. Conf. on Machine Learning, San Francisco, CA, United States, 1993, pp. 330–337

[27] Liu, Y., Liu, L., Chen, W.P.: 'Intelligent traffic light control using distributed multiagent q learning'. 2017 IEEE 20th Int. Conf. on Intelligent Transportation Systems (ITSC), Yokohama, JAPAN, 2017, pp. 1–8

[28] Dresner, K., Stone, P.: 'Multiagent traffic management: opportunities for multiagent learning'. In: Int. Workshop on Learning and Adaption in Multi-Agent Systems, Utecht, The Netherlands, 2005, pp. 129–138

[29] Da Silva, B.C., Basso, E.W., Perotto, F.S.C., *et al.*: 'Improving reinforcement learning with context detection'. Proc. of the Fifth Int. Joint Conf. on Autonomous Agents and Multiagent Systems, Hakodate, Japan, 2006, pp. 810–812

[30] Hossan, S., Nower, N.: 'Fog-based dynamic traffic light control system for improving public transport', *Public Transp.*, 2020, **12**, pp. 1–24

[31] Sutton, R.S., Barto, A.G.: '*Introduction to reinforcement learning*' (MIT Press Cambridge, USA, 1998), vol. **135**

[32] Van Otterlo, M., Wiering, M.: 'Reinforcement learning and markov decision processes'. In Wiering, M., van Otterlo, M. (Eds.): '*Reinforcement learning*' (Springer, Germany, 2012), pp. 3–42

[33] Zeiler, M.D.: 'Adadelta: an adaptive learning rate method', arXiv preprint arXiv:12125701, 2012

[34] Tokic, M., Palm, G.: 'Value-difference based exploration: adaptive control between epsilon-greedy and softmax'. Annual Conf. on Artificial Intelligence, Berlin, Germany, 2011, pp. 335–346

[35] Goodfellow, I., Bengio, Y., Courville, A.: '*Deep learning*' (MIT Press, USA, 2016)

[36] Gurney, K.: '*An introduction to neural networks*' (CRC Press, USA, 1997)

[37] Lin, L.J.: 'Reinforcement learning for robots using neural networks', Carnegie-Mellon Univ, Pittsburgh, PA, School of Computer Science, 1993

[38] Liang, X., Du, X., Wang, G., *et al.*: 'Deep reinforcement learning for traffic light control in vehicular networks', arXiv preprint arXiv:180311115, 2018

[39] Mousavi, S.S., Schukat, M., Howley, E.: 'Traffic light control using deep policygradient and value-function-based reinforcement learning', *IET Intell. Transp. Syst.*, 2017, **11**, (7), pp. 417–423

[40] Van der Pol, E., Oliehoek, F.A.: 'Coordinated deep reinforcement learners for traffic light control'. Proc. of Learning, Inference and Control of Multi-Agent Systems (at NIPS 2016), Barcelona, Spain, 2016

[41] Coşkun, M., Baggag, A., Chawla, S.: 'Deep reinforcement learning for traffic light optimization'. 2018 IEEE Int. Conf. on Data Mining Workshops (ICDMW), Singapore, Singapore, 2018, pp. 564–571

[42] Wei, H., Xu, N., Zhang, H., *et al.*: 'Colight: learning network-level cooperation for traffic signal control'. Proc. of the 28th ACM Int. Conf. on Information and Knowledge Management, Beijing, China, 2019, pp. 1913–1922

[43] Brys, T., Pham, T.T., Taylor, M.E.: 'Distributed learning and multi-objectivity in traffic light control', *Connect. Sci.*, 2014, **26**, (1), pp. 65–83

[44] Bu, L., Babu, R., De.Schutter, B., *et al.*: 'A comprehensive survey of multiagent reinforcement learning', *IEEE Trans. Syst. Man Cybern., C, Appl. Rev.*, 2008, **38**, (2), pp. 156–172

[45] Taylor, M.E., Jain, M., Tandon, P., *et al.*: 'Distributed on-line multi-agent optimization under uncertainty: balancing exploration and exploitation', *Adv. Complex Syst.*, 2011, **14**, (3), pp. 471–528

[46] Nishi, T., Otaki, K., Hayakawa, K., *et al.*: 'Traffic signal control based on reinforcement learning with graph convolutional neural nets'. 2018 21st Int. Conf. on Intelligent Transportation Systems (ITSC), Maui, Hawaii, China, 2018, pp. 877–883

[47] Brys, T., Harutyunyan, A., Vrancx, P., *et al.*: 'Multiobjectivization and ensembles of shapings in reinforcement learning', *Neurocomputing*, 2017, **263**, pp. 48–59

[48] Abdulhai, B., Pringle, R., Karakoulas, G.J.: 'Reinforcement learning for true adaptive traffic signal control', *J. Transp. Eng.*, 2003, **129**, (3), pp. 278–285

[49] Balaji, P., German, X., Srinivasan, D.: 'Urban traffic signal control using reinforcement learning agents', *IET Intell. Transp. Syst.*, 2010, **4**, (3), pp. 177–188

[50] Kingma, D.P., Ba, J.: 'Adam: a method for stochastic optimization', arXiv preprint arXiv:14126980, 2014

[51] Krajzewicz, D., Erdmann, J., Behrisch, M., *et al.*: 'Recent development and applications of sumo-simulation of urban mobility', *Int. J. Adv. Syst. Meas.*, 2012, **5**, (3&4), pp. 128–138

[52] Lin, Y., Dai, X., Li, L., *et al.*: 'An efficient deep reinforcement learning model for urban traffic control', arXiv preprint arXiv:180801876, 2018

[53] Trafficware, L.: '*Synchro studio 10*' (Trafficware LLC, Sugar Land, 2017)

© The Institution of Engineering and Technology 2021