Transportation Research Part C 103 (2019) 246–260

Contents lists available at ScienceDirect

Transportation Research Part C

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

DCL-AIM: Decentralized coordination learning of autonomous intersection management for connected and automated vehicles

Yuanyuan Wu[a], Haipeng Chen[b], Feng Zhu[a,*]

[a] School of Civil and Environmental Engineering, Nanyang Technological University, Singapore
[b] Department of Computer Science, Dartmouth College, NH, United States

ARTICLE INFO

Keywords:
Multi-agent coordination
Reinforcement learning
Intersection management
Connected and automated vehicles

ABSTRACT

Conventional intersection managements, such as signalized intersections, may not necessarily be the optimal strategies when it comes to connected and automated vehicles (CAVs) environment. Autonomous intersection management (AIM) is tailored for CAVs aiming at replacing the conventional traffic control strategies. In this work, using the communication and computation technologies of CAVs, the sequential movements of vehicles through intersections are modelled as multi-agent Markov decision processes (MAMDPs) in which vehicle agents cooperate to minimize intersection delay with collision-free constraints. To handle the huge dimension scale incurred by the nature of multi-agent decision making problems, the state space of CAVs are decomposed into independent part and coordinated part by exploiting the structural properties of the AIM problem, and a decentralized coordination multi-agent learning approach (DCL-AIM) is proposed to solve the problem efficiently by exploiting both global and localized agent coordination needs in AIM. The main feature of the proposed approach is to explicitly identify and dynamically adapt agent coordination needs during the learning process so that the curse of dimensionality and environment nonstationarity problems in multi-agent learning can be alleviated.

The effectiveness of the proposed method is demonstrated under a variety of traffic conditions. The comparison analysis is performed between DCL-AIM and the First-Come-First-Serve based AIM (FCFS-AIM), with Longest-Queue-First (LQF-AIM) policy and the signal control based on the Webster's method (Signal) as benchmarks. Experimental results show that the sequential decisions from DCL-AIM outperform the other control policies.

1. Introduction

Connected and automated vehicles (CAVs) are seen as the future of transportation (Fagnant and Kockelman, 2015). CAVs have the ability to communicate with each other (Vehicle-to-Vehicle, V2V), road infrastructures (Vehicle-to-Infrastructure, V2I), and even non-motorized road users including pedestrians (Vehicle-to-X, V2X). After reaching a certain level, CAVs can drive themselves without the input of human beings, relying only on the computer control systems they are equipped with. Currently, these communication technologies and control systems are still in the research and testing phase. However, the promising benefits of CAVs are foreseeable. On one aspect, CAVs can make the road traffic safer because they can coordinate their movements through connectivity. Moreover, as the movements are coordinated, unnecessary stop-and-go behaviors will be reduced, thereby reducing emission.

* Corresponding author at: 50 Nanyang Avenue, N1-01b-45, Singapore 639798, Singapore.
E-mail addresses: wuyuanyuan@ntu.edu.sg (Y. Wu), haipeng.chen@dartmouth.edu (H. Chen), zhufeng@ntu.edu.sg (F. Zhu).

https://doi.org/10.1016/j.trc.2019.04.012
Received 10 November 2018; Received in revised form 7 April 2019; Accepted 13 April 2019
Available online 22 April 2019

Further, CAVs can also provide intelligent transport services, such as the shared autonomous vehicles (Lokhandwala and Cai, 2018).

Once CAVs become a reality, further traffic flow improvements can be expected. CAVs will change the way people travel and reshape the existing transportation system. The consequent updates of traffic control and management are necessary. As the crux of traffic network, updated intersection management for CAVs has been researched (Chen and Englund, 2016). Autonomous intersection management (AIM) is one of the updated intersection control strategies tailored for CAVs. AIM, proposed by Dresner and Stone (2004), divides intersections into a grid of space-time cells to manage the coordinated movement of CAVs. CAVs wishing to cross the intersection make requests to reserve cells from the Intersection Manager (IM). The IM accepts or rejects reservations in accordance with its control policy. The most commonly used policy is first-come-first-serve (FCFS), in which reservations of CAVs are prioritized by their arriving time and reservation time. Based on FCFS policy, further researches assigned higher priority for emergency vehicles (Dresner and Stone, 2006) and determined the priority pattern by using auctions (Schepperle and and Böhm, 2007). It has been demonstrated that FCFS based AIM and its variants can reduce delay and emission compared to conventional signal control under certain traffic conditions (Fajardo et al., 2011; Li et al., 2013).

However, for a dynamic system like AIM where the traffic environment changes over time, a predefined policy may become suboptimal or even inappropriate. Levin et al. (2016) demonstrated that FCFS based AIM may not outperform traffic signals on some realistic networks. This indicates that the optimal policy based on environmental dynamics deserves more in-depth research. In this paper, we aim to propose an optimal AIM framework which fully incorporates the real-time traffic dynamics for CAVs.

In our research framework, the processes of how CAVs cross the intersection are modelled as multi-agent Markov decision processes (MAMDPs) which are extended from single-agent MDPs. The curse of dimensionality incurred by the large number of agents is the main problem in solving MAMDPs. Reinforcement learning (RL), especially multi-agent reinforcement learning (MARL), has been proposed to solve MAMDPs. Generally, in RL, agents learn the optimal policy by trial-and-error interaction with the dynamic environment which can be formally described by Markov decision processes (MDPs) (Sutton and Barto, 1998). In AIM, CAVs are the autonomous agents who perceive and interact with the real time traffic situations (environment) using their advanced communication technologies and sensors. More specifically, we customize an MARL algorithm, called DCL-AIM algorithm, to solve the decision-making problems of CAVs.

The contributions of this paper are as follows. The process of passing CAVs through an intersection is modelled as MAMDP. Decentralized learning is applied to reduce the impact of dimensionality in solving the model, while coordination learning is adopted to mitigate the effects of environment nonstationarity problem in decentralized multi-agent learning. Taking advantage of the sparse interacting characteristic between CAVs, DCL-AIM algorithm decomposes the multi-agent decision making problem in AIM into subproblems by explicitly identifying and dynamically adapting agent coordination needs during the learning process. The *coordination needs* are limited to specific parts of the state space where there are potential conflicts. *Coordinated joint-actions* ensure that the policies learned are collision-free, and reward function is designed to enable the learned policies to be equal to the intersection delay minimization.

The remainder of this paper is organized as follows. Section 2 briefly reviews previous work on AIM. Section 3 is the problem statement, detailing the multi-agent formulation of AIM problem. Section 4 introduces the main methods we used in the research, including how we model the optimization problem of AIM into decentralized coordination learning problem and the algorithms proposed to solve the problem. Section 5 presents the details of simulation implementations, the performance of the solution methods and the analysis results of the effectiveness comparison. Section 6 concludes the paper and identifies the direction of future work.

2. Related work

Conventional intersection managements, such as signalized intersections, are not necessarily the optimal strategies when it comes to CAVs environment. There are new alternative intersection managements customized for CAVs.

One such new intersection management that has attracted a lot of attention is the reservation based autonomous intersection management (AIM) proposed by Dresner and Stone (2004). AIM divides intersections into a grid of space-time cells to manage the coordinated movement of CAVs. The vehicle agents attempt to reserve cells in the intersection, then the *intersection manager* decides whether to grant or reject requested reservations according to an *intersection control policy*, named FCFS, because of the *First-Come-First-Serve* nature of the reservations. Based on FCFS, emergency vehicles are given higher priorities in Dresner and Stone (2006), and auctions are used to determine the priority pattern in Carlino et al. (2013). The seminal work of Stone and his team (Dresner and Stone, 2008; Hausknecht et al., 2011) have demonstrated that the FCFS protocol outperforms conventional intersection control protocols, such as signal control and all-stop control, under certain traffic conditions. The similar efficacy of auction-based control protocol also has been validated (Schepperle and and Böhm, 2007; Schepperle and Böhm, 2008; Vasirani and Ossowski, 2012; Carlino et al., 2013).

Although it has been demonstrated in the literature that AIM and its variants have the potential to mitigate traffic congestion in urban areas, their pre-determined control policies may result in the priority assignments being only feasible but not optimal.

Some further studies focused on the optimization of AIM. Lee et al. (2013) proposed a cooperative vehicle intersection control (CVIC) algorithm to minimize total length of overlapped trajectories within the intersection zone. Their objective function is equal to delay or travel time minimization, however, the constraints are nonlinear, which may result in the final solution set containing overlapped trajectories (collisions). In order to overcome the shortcomings of the nonlinear optimizing algorithms, Zhu and Ukkusuri (2015) proposed a linear programming formulation of AIM, which minimizes the total travel time by modelling the conflict points. Levin et al. (2017) aggregated the conflict points model into conflict regions and derived a corresponding integer programming to further improve computational efficiency. Sun et al. (2017) proposed a multi-objective mixed-integer non-linear programming

designed to maximize intersection capacity. Mirheli et al. (2018) proposed a signal-head-free intersection control logic (SICL) for AIM to maximize the intersection throughput under collision-free constraints. Further, taking SICL as benchmark, a distributed SICL (DC-SICL) is proposed (Mirheli et al., 2019) where the vehicle-level mixed-integer non-linear programming is formulated with the objective of minimizing travel time and speed variations of each CAV. There are also works on fuel consumption minimization (Zhang et al., 2016), energy-optimal control (Malikopoulos et al., 2018), and multiple performance measures (delay, fuel consumption, and safety) (Guo et al., 2019), etc.

The optimization approaches for AIM mentioned above are more or less faced with the computational intractability problem caused by traffic dynamics. More simple and effective optimization control is worth studying.

The ability to learn and find out the optimal decisions through dynamic interaction with the environment enables RL to be competent for the adaptive control tasks of many intelligent transportation systems (Bazzan and Klügl, 2014), and RL is applied at most to the adaptive traffic signal control system. One of the first trials was conducted by Abdulhai et al. (2003). They demonstrated that the RL-based adaptive traffic signal has better performance than conventional traffic signal control through a case study of an isolated intersection. This isolated intersection case then was extended and applied to a signal network situation using multi-agent RL approaches (Arel et al., 2010; Jin et al., 2012). There is a detailed review of RL-based adaptive traffic signal control given by Mannion et al. (2016). In addition, RL is also applied to the dynamic speed limit control of connected vehicles (Zhu and Ukkusuri, 2014), human-like autonomous car-following behavior learning (Zhu et al., 2018), dynamic tolling approach for traffic congestion alleviation (Chen et al., 2018), optimal fuel consumption control (Qi et al., 2019), etc.

In the literature, there are relatively few studies using RL to solve AIM problems. Isele et al. (2017) proposed a single-agent RL approach to navigate one autonomous vehicle through the intersection. They only focus on one agent, but the traffic system is essentially multi-agent. Chen et al. (2018) proposed an RL-based dynamic tolling approach for traffic congestion alleviation. However, their framework is targeted at macro-scopic problems over an entire traffic network, and thus cannot be applied to our problem. In our paper, multi-agent reinforcement learning (MARL) is used to find the optimal sequential actions for each CAV (agent). To the best of our knowledge, this paper is one of the first trials to use MARL to solve AIM problems.

Most MARL algorithms are derived from a model-free single-agent RL algorithm called *Q-learning* (Watkins and Dayan, 1992; Peng and Williams, 1994; Littman, 2001). MARL methods can be mainly divided into two categories: centralized learning (CL) and decentralized learning (DL) (Busoniu et al., 2008). In CL, there is a central agent with powerful computing ability that learns the value and/or policy of the multi-agent joint-action in the system. The main challenge of CL is the scalability issue caused by the exponential growth of state space. In order to avoid the curse of dimensionality, in DL, each agent independently learns its own value of actions and/or its own policy, treating other agents as part of the environment (Tan, 1993). However, DL introduces a new challenge for MARL that the environment becomes non-stationary from the point of view of each agent, as it contains other agents who are themselves learning, rolling out any convergence guarantees. Fortunately, substantial empirical evidence has shown that DL approach are often effective in practice (Matignon et al., 2012).

Decentralized coordination learning (DCL) has the potential to solve the environment nonstationarity problem in decentralized MARL. The main idea of DCL is to capture local dependencies or coordination needs between the different agent in MAMDPs, and allow the *overall Q-function* to be decomposed into *local Q-functions* that can be more easily optimized (Guestrin et al., 2002; Kok and Vlassis, 2004). DCL is more effective for systems where the interaction between agents is sparse or loosely coupled. Melo and Veloso (2011) modeled these systems as sparse-interaction multi-agent systems, Dec-SIMDPs. In addition to the theoretical contributions, they also proposed a solution method that utilizes the particular structure of Dec-SIMDPs. Further on, instead of specifying the coordination needs, Yu et al. (2015) proposed an algorithm to enable agents to learn efficient coordinated behaviors by exploiting agent independence in loosely coupled multi-agent systems.

In AIM, the interaction between CAVs is also sparse and not coupled at every decision step but only in relatively infrequent situations — CAVs only need to coordinate with the vehicles on their conflicting lanes when their moving intentions have potential conflicts. Due to this property, DCL is applicable for the decision-making problem in AIM. However, a key difficulty of the AIM problem is that the number of agents keeps changing over time, which makes it infeasible to directly apply DCL approaches to our problem. Based on the characteristics of traffic flow, we propose several novel adaptations and customize a decentralized coordination learning algorithm for AIM, called DCL-AIM algorithm. Details are shown in the following sections.

3. Problem statement

3.1. Problem description

Introduced by Dresner and Stone (2004), AIM is a multi-agent system, which applies a reservation-based protocol. In the protocol, vehicle agents need to send a request to the intersection manager (IM) agent reserving the space-time occupancy of the intersection. The space of the intersection is divided into cells, and the reservation is about the desired occupancy of cells along its trajectory for a certain duration. There may be conflicts in all reservations, i.e. the same cells are requested at the same time. Then the IM agent needs to decide whether to grant or reject the requested reservations. Different intersection control policies are employed. At each simulated step, the employed policy determines cells the vehicle will occupy using the state of vehicle agents including the time and velocity of arrival, vehicle size, etc. Our goal is to find an optimal control policy that minimizes intersection delay.

For the definition of delay of an intersection, we follow the same definition from the seminal work of Dresner and Stone (2004). The number of all vehicles passing through an intersection in a period of time (e.g. one minute) is denoted as C. Assume that there are no other vehicles on the road, vehicle v_i would pass the intersection with the maximally allowable speed in time OT_i. However, due to

the presence of other vehicles and the need of avoiding potential conflicts, the actual travel time of v_i is AT_i. Hence, the intersection delay of an intersection (*DoI*) is:

$$DoI = \sum_{v_i \in C} (AT_i - OT_i) \tag{1}$$

However, the centralized optimization of reservation-based AIM is ultimately a dynamic traffic assignment problem and thus is extremely challenging in the sense of computational tractability (Levin et al., 2017). In order to avoid the curse of dimensionality in the centralized optimization, we propose a multi-agent formulation of the AIM problem, where instead of sending reservations to the IM agent and relying on the optimal assignments of the IM agent, vehicle agents cooperate and learn a coordinated cell space-time occupancy policy to minimize the defined delay.

3.2. A multi-agent formulation of AIM problem

Once entering the intersection area, without the control of IM agent, vehicle agents need to coordinate and negotiate with each other about the space-time occupancy of the cells, so as to avoid collisions as the primary condition and minimize the delay. It is a multi-agent sequential decision making problem, which can be modeled as multi-agent Markov decision process (MAMDP): $\langle n, \mathscr{S}, A_1, A_2, ...,A_n, \delta, r_1, r_2, ...,r_n, \gamma\rangle$, where n is the number of agents, $\mathscr{S}$ is the finite set of environment states describing the possible joint-states of all agents, A_i is the finite set of actions available to agents yielding the joint action set $\mathscr{A} = A_1 \times ...\times A_n$, δ: $\mathscr{S} \times \mathscr{A} \rightarrow \mathscr{S}$ is the state transition function, and r_i: $\mathscr{S} \times \mathscr{A} \times \mathscr{S} \rightarrow \mathbb{R}$, $i = 1, ...,n$ is the reward function.

In the following, the elements of MAMDP are introduced in the context of AIM.

• *States*: states are the information used to determine what happens next. Vehicle agents need to make a decision based on the joint-states of each other. For each agent, the individual state is the combination of its *current position*, *speed*, *moving intention* and the *queue length* of its current lane.

The intersection is divided into cells, where each cell has its own fixed coordinates. The position of a vehicle can be represented by the cells it is occupying. A cell can only be occupied by one vehicle at the same time, and if any part of the cell is occupied, the state of the cell will be regarded as occupied (see Fig. 1).

The speed of a vehicle is discretized into the number of cells per time step. The arc length of the turning movements is converted to the length of the cell. This discretization is commonly used in the simulation of traffic flow using Cellular Automata model and has been demonstrated to capture the characteristics of traffic flow well (Nagel and and Schreckenberg, 1992). The speed is related to the length of cell and the step size of the simulation.

Example 1. A speed of 20 m/s is equal to 2 cell/step when the length of a cell is 5 m and the step size is 0.5 s.

The concept of *desired cells* is proposed to describe the moving intention of a vehicle at each time step which is related to its destination, trajectory and current speed *V*. Vehicles desire to cross the intersection in the shortest amount of time, i.e. accelerate to or maintain the speed limit and move towards destinations (this behavior is referred to as *'keep going'* in this paper). However, for safety reasons, it has to maintain a safe stopping distance, i.e. safety buffer. Therefore, the number of desired cells (*DCs*) ahead of moving direction can be calculated as:

$$DCs = \begin{cases} max(\lceil V^2/2a \rceil, \lceil V\Delta t + \frac{1}{2}a\Delta t^2 \rceil), & V < V_m \\ max(\lceil V^2/2a \rceil, \lceil V\Delta t \rceil), & V = V_m \end{cases} \tag{2}$$

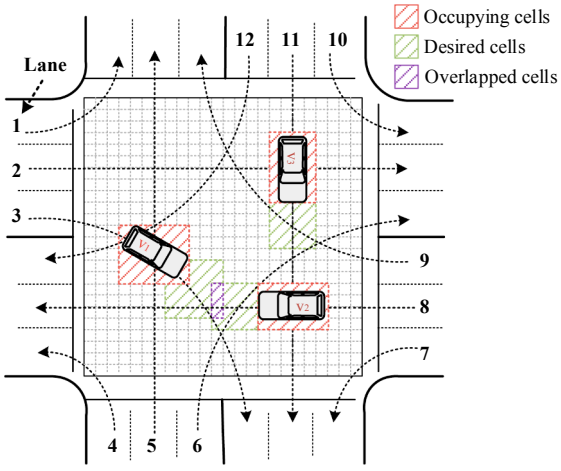

Fig. 1. An illustration of cell occupancies.

Table 1
Action choices of vehicles.

Current speed	V_0	$\{V_1, \cdots, V_{m-1}\}$	V_m
Action choices	$\{+a, 0\}$	$\{\pm a, 0\}$	$\{-a, 0\}$

where a is the acceleration/deceleration value of the vehicle in unit of cell/step^2, V_m is the speed limit, and $V^2/2a$ is the safe stopping distance needed to avoid potential collisions. Step size Δt is set close to 0, and it takes at least one step for a vehicle to accelerate to speed limit. If a cell is occupied and desired by the same vehicle, the cell will be regarded as the desired cell of this vehicle.

Example 2. For a vehicle with a current speed $V_m = 4$ cell/step, and acceleration $a = 1$ cell/step^2, its $DCs = 8$ cells. For a vehicle with a speed of 3 cell/step, its $DCs = \lceil 4.5 \text{ cells} \rceil = 5$ cells.

Example 3. Let the intersection depicted in Fig. 1 be represented as a 24×24 two-dimensional array S_e. Assume that the current speed of vehicle v_3 is $V_m = 4$ cell/step and acceleration $a = 2$ cell/step^2, and there are 10 vehicles queueing on lane 11. The individual state (i.e. the position, speed, moving intention and queue length) of v_3 can be represented as:

$\{S_e[3{:}\ 8][17{:}\ 20],\ 4 \text{ cell/step},\ S_e[9{:}\ 12][17{:}\ 20],\ 10 \text{ veh}\}$

• *Actions*: vehicle agents execute actions based on the states they observed. The actions of vehicles are their state of motion. Generally, the state of motion of a vehicle can be expressed in term of its speed $(V_0, \cdots, V_m)$ and acceleration/deceleration $(\pm a, 0)$. The choice of an action is constrained by the relationship between speed and acceleration/deceleration and the range of speed: $[V_0, V_m]$ (see Table 1).

The global state and action (S, A) is an aggregation of the individual states and actions (s_i, a_i).

• *Rewards*: scalar rewards are used to indicate how well agents are doing at time step t in achieving the goal. In order to discourage congestion, our objective is to minimize the intersection delay, so the negative of delay (Eq. 1) is set as a reward. Assuming that the distance traveled by vehicle v_i at time step t is L_i, then the optimal time required to pass the distance is L_i/V_m, so the reward at time step t can be calculated as:

$$r(S, A) = -\sum_{v_i \in C} (\Delta t - L_i/V_m) \tag{3}$$

• *State transition*: the states of vehicles are transited according to the law of motion.

For the cooperative MAMDP of AIM, our objective is to find a joint-policy Π^*: $\mathscr{S} \rightarrow \mathscr{A}$ which maximizes the total rewards of all agents. Π^* can maximize the long-term discounted reward for all agents, and it can be split into n component policies for each agent π_i: $\mathscr{S} \rightarrow A_i$ with $\Pi^* = (\pi_1, ..., \pi_n)$. One way to find the optimal policy Π^* is by computing an optimal joint-action value function:

$$Q^*(S, A) = max_{\Pi^*} Q(S, A) \tag{4}$$

Theoretically, Eq. (4) can be solved through the use of dynamic programming techniques. By iterating, evaluating and calculating value functions, it is possible to find the optimal policies. However, it is almost impossible to compute the joint-action value function for a system like AIM within acceptable execution time because the (S, A) pairs in AIM system are enormous. In multi-agent decision making problem, each agent adds its own variables to the joint state-action space which leads to the complexity of solving MAMDPs being exponential in the number of agents. However, since the interaction between CAVs in AIM is sparse and not coupled at every decision step, but only in relatively infrequent situations, with the new MAMDP formulation, we are able to decompose the original centralized optimization into a distributed one. This leaves us the possibility of using a decentralized learning framework towards solving the AIM problem. In the following, we adapt the decentralized coordination learning (DCL) framework into solving our formulated multi-agent AIM problem.

4. DCL-AIM: decentralized coordination learning of AIM

Instead of learning an optimal joint policy, agents in DCL learn and update the coordinated policy. DCL is driven by the fact that many multi-agent systems can exhibit a large amount of context-specific independence so that a general MAMDP problem can be decomposed into sub-problems which are easier to solve (Roth et al., 2007). One such context-specific independent state of a vehicle

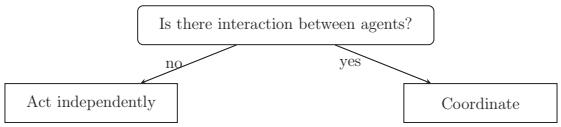

Fig. 2. Procedure of coordination learning.

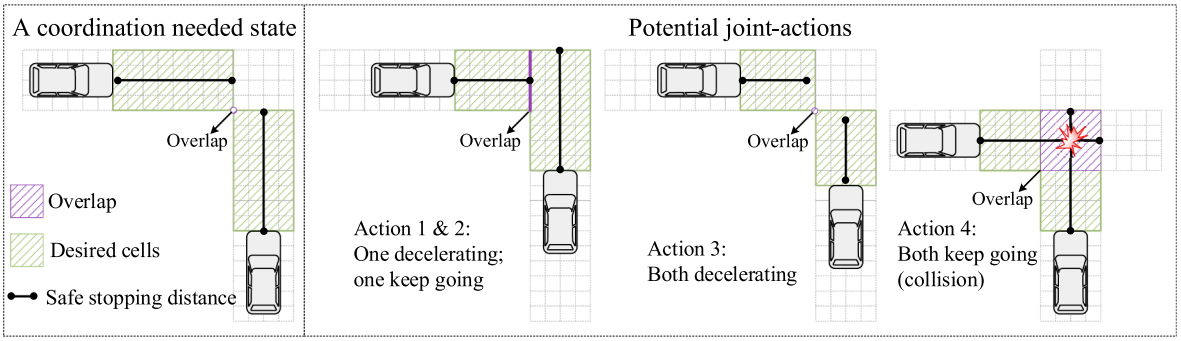

Fig. 3. A coordination needed state and potential joint-actions involving two CAVs.

agent in AIM is that there are no vehicles on the conflicting lanes. The optimal action in this state is easier to determine, that is, the vehicle acts independently, accelerates to or maintains the maximum speed.

The primary task of coordination learning algorithms is to train the agents to learn where and how to coordinate (Yu et al., 2015). At each learning step, the main procedure of coordination learning is shown in Fig. 2.

In DCL, the large-scale environment state space is divided into two parts: *coordinated states*, in which agents need to coordinate their actions so that the decision-making process can be improved; *independent states*, in which agents can act independently and coordination with other agents does not improve the decision-making process.

In AIM, for an agent i, its independent states are the states that its desired cells are not occupied and there is no overlap (even a point) with the occupying or desired cells of other vehicles. The rest are the states that need to be coordinated.

Example 4. Consider the state depicted in Fig. 1, in the current state, vehicle v_3 is independent of other vehicles, thus no coordination is needed; vehicle v_1 and vehicle v_2 are in a coordinated state. The overlap of desired cells indicates that collisions may occur if vehicles v_1 and v_2 both take independent actions. Coordination is needed between vehicles v_1 and v_2.

We define *coordinated joint-actions* as the available actions that can be taken in coordinated states, excluding the joint-actions that would cause any conflicts.

Example 5. For a coordination-needed state shown in Fig. 3, the coordinated joint-actions are Actions 1, 2 and 3, while Action 4 is eliminated. If these two vehicles both choose 'keep going' action, a collision cannot be avoided at the next state, because the safe stopping distance cannot be guaranteed.

Based on this idea, an algorithm DCL-AIM (Algorithms 1 and 2) is proposed to solve the decision-making problem of AIM by explicitly identifying and dynamically adapting agent coordination needs during the learning process.

Algorithm 1. DCL-AIM

```
Establish Intersection Environment:
    reset and update states
    emit rewards and observations
Generate Vehicles: in-flow x for each lane l
Initialize Q − table: Q_i and Q^I
for episode ← 1 to MaxEpisodes do
    for t ← 0 to T do
        update the number of agents
        update the states
        for each agent i do
            Algorithm 2
return Q − table for each agent i
```

Algorithm 2. LEARNING ALGORITHM FOR AGENT i

```
Inherit Q − table: Q_i and Q^I
while agent i is within the intersection do
    get observations
    if s_i, s_K not in Q − table.index then
        append the new state to Q − table
    # choose action (ε−greedy)
    if agent i is involved in coordination then
        choose a coordinated action
    else
        choose an independent action
    obtain rewards
    update Q − table
```

In DCL-AIM algorithm, vehicles in the same direction from the same lane all inherit and contribute to learning and updating the same *Q-function*. Corresponding to the decomposition of state, *Q-function* of agent i also has two parts: a single-action value table $Q_i(s_i, a_i)$ and a joint-action value table $Q^I(s_K, a_K)$ for states which require coordination. At independent states, agents will act independently and choose actions based on the optimal Q-function Q_i^*. At coordinated states, CAVs will choose actions from joint optimal Q-function $Q^{I^*}(s_K, a_K)$.

The decomposition relies on the assumption that in any given state, the agents are either independent or can share state information, as such agents can use state information from the other agents in the coordinated states to choose its actions. With the advanced sensors and the V2V communication technologies, each CAV is able to perceive unambiguously its own position in the environment. Also, when agents are simultaneously present in the coordination needed area, namely coordinated states, they are able to sense each other and thus can perceive their joint state clearly.

At time step t, the global reward corresponding to Eq. (3) is calculated in Eq. (5).

$$r(S, A) = -\sum_{i=1}^{N^t} wt_i \tag{5}$$

where N^t is number of CAVs present at time t. wt_i is the delay of agent i at step t. Since the vehicles are always moving in and out, N^t changes over time.

Correspondingly, when an agent i is in an independent state at time t, its *individual reward* is:

$$r(s_i, a_i) = -wt_i \tag{6}$$

Otherwise, when agent i is in a coordinated state and is in subset $K = \{i_1, ..., i_h\}$, its *joint-reward* is:

$$r(s_K, a_K) = -\sum_{i=1}^{h} wt_i \tag{7}$$

The learning progress begins with an initialization of the action value functions and followed by the optimal policy learning process. In order to guarantee that the sequence Q_t converges to Q^*, the agents need to keep trying all actions in all states with nonzero probability. The trade-off between exploitation and exploration guarantees the convergence. $\varepsilon - greedy$ exploration is a widely used approach. At each step, the agent can choose a random action with probability $\varepsilon \in (0, 1)$, and the greedy action with probability $(1 - \varepsilon)$.

Each episode in the algorithm simulates a possible process of the vehicles crossing the intersection, and ends when all CAVs exit the intersection, followed by a reset to the initial state. The DCL-AIM algorithm estimates all the possibilities of how CAVs cross the intersection, and then outputs the best decisions for the actual operation of CAVs.

The main tasks of DCL-AIM algorithm at every time step t are as follows:

- Update the environment.
  - The environment will be updated based on the actions of the current agents and the new agents just arrived. Function *env. update*() returns the new occupancy state of the intersection and *env. agents*() gives out the presence of agents including local states. In addition, the environment will be reset to the same initial state using the function *env. reset*() at the end of each episode.
  - The environment class also includes the functions of rewards: *env. rewards*() and observations: *env. observations*(), which are in

charge of emitting rewards and observations to agents based on the environment states resulted from their actions.

- Update the Q-values.

At time step t, if agent i is not involved in any interaction, and moves from an independent state s_i to another independent state s'_i, the independent Q-values can be estimated by the standard Q-learning operation (Watkins and Dayan, 1992), shown in Eq. (8).

$$Q_{e+1}(s, a) = Q_e(s, a) + \alpha[r(s, a) + \gamma \max_{a' \in A} Q_e(s', a') - Q_e(s, a)] \tag{8}$$

where α is the learning rate and specifies how far the current estimate $Q_e(s, a)$ at the eth update is adjusted toward the update target $\gamma \sum_{s' \in S} f(s, a, s') \max_{a' \in A} Q^*(s', a')$.

Otherwise, let K denote the subset of all h agents interacting with agent i at time step t.

◦ Case 1, agent i moves from a coordinated state s_K to an independent state s_i. The joint Q-values are updated by:

$$Q^I(s_K, a_K) = (1 - \alpha) Q^I(s_K, a_K) + \alpha \sum_{i=1}^{h} [r_i(s_K, a_K) + \gamma \max_{a'_i} Q_i(s_i, a'_i)] \tag{9}$$

◦ Case 2, conversely, agent i moves from an independent state s_i to a coordinated state s_K. The independent Q-values are updated by:

$$Q_i(s_i, a_i) = (1 - \alpha) Q_i(s_i, a_i) + \alpha[r_i(s_i, a_i) + \gamma \frac{1}{h} \max_{a'_K} Q^I(s_K, a'_K)] \tag{10}$$

That is, each agent is rewarded with the same fraction of the expected future discounted reward from the resulting coordinated state. This essentially implies that each agent contributes equally to the coordination.

◦ Case 3, agent i moves from a coordinated state s_K to another coordinated state s'_K. The update of Q-values of $Q^I(s_K, a_K)$ also follows the operation shown in Eq. (8) according to the joint-rewards.

QUpdate() is used to compactly denote the Q-values update operation described above.

5. Simulations and results

DCL-AIM algorithm has three main components: intersection environment, vehicle generation and *Q-table* learning. The specific details of these three parts in the simulation implementation are described in the following subsections. The results of the simulations are then analyzed and presented.

5.1. A 4-way-6-lane intersection

All simulations are conducted using Python 3.6, and run on a computer with a i7-7700 CPU @3.60 GHz, 8.00 GB RAM and 64-bit Operating Windows System. A typical 4-way-6-lane intersection configuration is simulated. The movements of CAVs are set to strictly follow the lane instruction, i.e. the trajectories of vehicles from different lanes are settled.

The control range of the intersection is set as 210 m within the effective communication range of dedicated short range

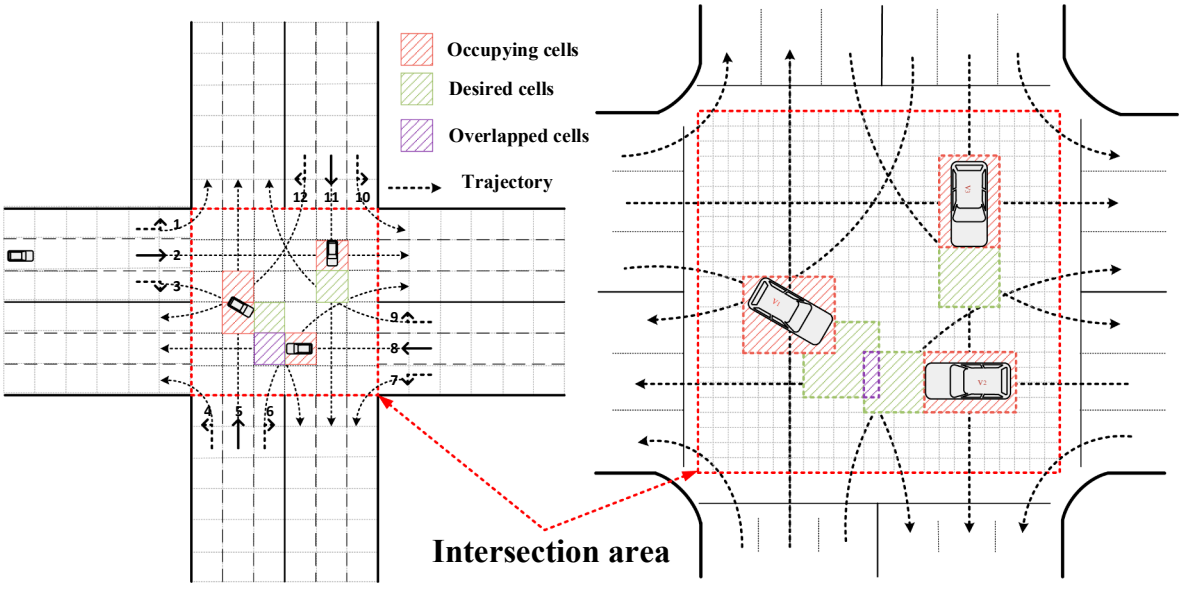

Fig. 4. Different discretization levels of intersection area.

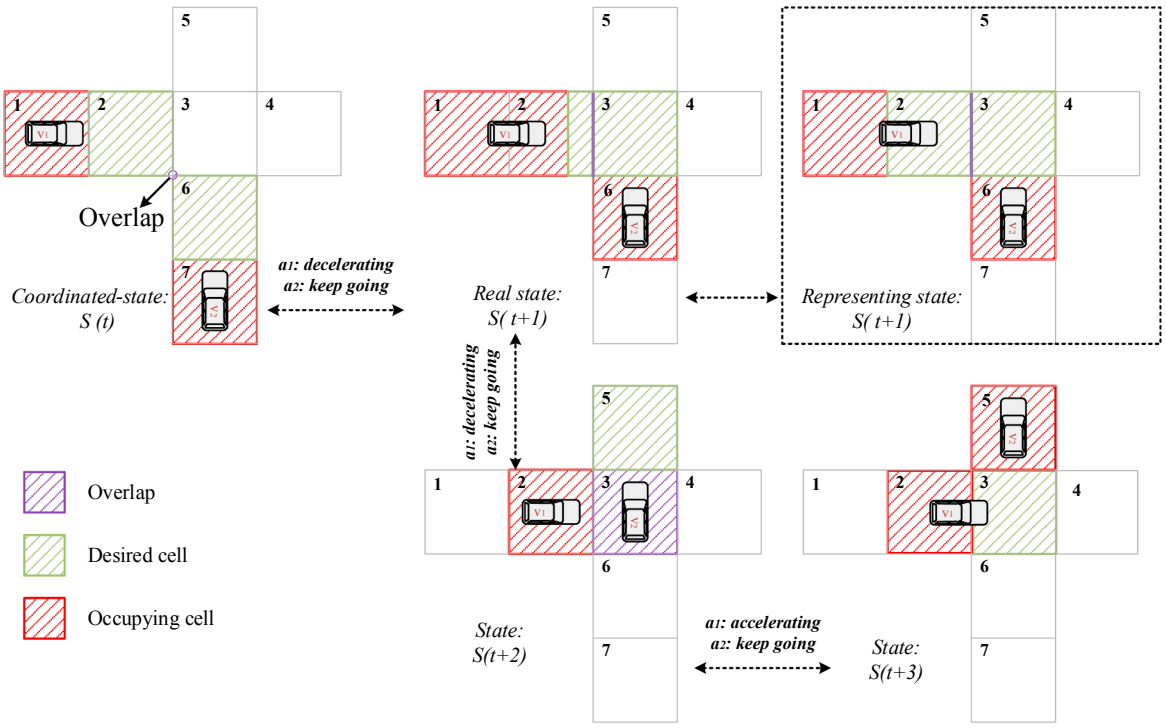

Fig. 5. Illustration of a sequential decision process and the cell occupancies.

communication (DSRC) technology (Kenney, 2011). The width of a lane is 3.5 m, the speed limit is set as 7 m/s, the simulation step size is 0.5 s.

In AIM, the intersection area is discretized into cells, the length of the cell represents the level of discretization, and two different levels of discretization are simulated, as shown in Fig. 4.

Discretization level 1. The intersection area is divided into 6×6 cells, each cell has the length/width of $3.5m$. The length of an CAV is set less than 3.5 m, so the occupancy of a vehicle is at least 1 cell, the specific occupancy is related to its trajectory, such as vehicle v_1 shown in Fig. 4 (left) is occupying more than one cells. The maximum moving distance of each CAV in each decision step is 7 m/s $\times$ 0.5 s= 3.5 m (the length of one cell), the set of actions of each agent is represented as $A = \{0, 1\}$ accordingly. '0' represents the actions of decelerating or waiting, '1' represents the action of accelerating or keeping going. The arc length of the turning movement is converted to the length of the cell. The allowable acceleration/deceleration is set as $\{\pm 0.5, 0\}$(cell/step^2).

Example 6. For the two vehicles in a coordinated state depicted in Fig. 5, in which the cells are numbered to refer to cell coordinates. At the coordination-needed state $S(t)$, they choose the coordinated actions of $\{a_1$: 0 $-$ *decelerating*, a_2: 1 $-$ *keep going*$\}$, and move to the next state $S(t+1)$. At $t+1$, parts of cell 2 is occupied by vehicle v_1 ($V_{v_1}\Delta t - 0.5a\Delta t^2 = 0.75$ *cell*), while the rest of cell 2 is desired by vehicle v_1 ($0.5a\Delta t^2 = 0.25$ *cell*). In this case, we still use cell 1 to represent the current position of vehicle v_1, and cell 2 is used to represent its moving intention. The state of vehicle v_1 at $t+1$ can be represented as $\{$*cell* 1, 0.5 *cell*/*step*, *cell* 2, $q_1\}$. Similarly, the state of vehicle v_2: $\{$*cell* 6, 1 *cell*/*step*, *cell* 3, $q2\}$.

Discretization level 2. The intersection area is divided into 24×24 cells, each cell having the length/width of 0.875 m. The length of an CAV is set as 4.5 m and width is 1.8 m, so the occupancy of a vehicle is at least 6×3 cells, and the specific occupancy is related to its trajectory, for example, a turning vehicle v_1 in Fig. 4 (right) is occupying 6×5 cells. The speed limit is equal to 4 *cell*/*step*, then the speed of a vehicle $V \in \{0, 1, 2, 3, 4\}$(cell/step). In order to provide better passenger experience, the allowable acceleration/deceleration is set as $\{\pm 1, 0\}$(cell/step^2).

The objective of the decision model of CAVs is to maximize the long-term rewards. During the learning process, the coordination needs are explicitly identified and dynamically adapted, collision-free is guaranteed by the coordinated joint-actions, and the reward function minimizes delay. As a result, the optimal policy given by the model equals the minimization of the intersection delay and is collision-free.

5.2. Vehicle generation

In order to simulate the traffic flow more realistically, vehicles are generated randomly using Poisson distribution. Each lane has different vehicle arriving rate λ, and intersection traffic generating rate is the sum of each lane. A Poisson Process is a kind of Markov Chain process, and has unique characteristics:

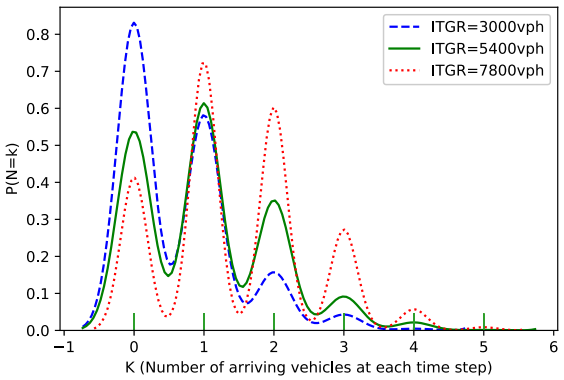

Fig. 6. Traffic in-flow distribution.

- the number of arrivals in non-overlapping intervals are independent, i.e., for every integer $m = 2, 3, \ldots$ and time points $t_0 = 0 < t_1 < t_2 < \ldots < t_m$, the number of vehicle arrivals of the intervals $N((t_0, t_1])$, $N((t_1, t_2])$, $\cdots$,$N((t_{m-1}, t_m])$ are independent.
- for any time t and positive number Δt, the probability distribution of $N((t, t + \Delta t])$ (the number of vehicle arrivals occurring between t and $t + \Delta t$) depends only on the interval length Δt , and not on the time t.
- as $\Delta t \to 0^+$, the probability of at least one vehicle arrives, is proportional to the interval length Δt:

$$P\{N((t, t + \Delta t]) \geqslant 1\} = \lambda \Delta t + o(\Delta t), \Delta t \to 0^+ \tag{11}$$

- the probability of two or more vehicles arrive in an interval of length $\Delta t \to 0^+$ is negligible:

$$P\{N((t, t + \Delta t]) \geqslant 2\} = o(\Delta t), \Delta t \to 0^+ \tag{12}$$

With the characteristics described above, we use the probability of at least one vehicle arrives to generate vehicles at each simulation step ($\Delta t = 0.5s$). The generating rule is as following:

$$if \quad P\{N((t, t + \Delta t]) \geqslant 1\} > random(0, 1), \quad then \quad v_i \quad generated \tag{13}$$

where, $random(0, 1)$ is a pseudo-random number in $(0, 1)$, v_i is the i_{th} generated vehicle at time t.

Three examples of traffic in-flow distribution are shown in Fig. 6. *ITGR* represents intersection traffic generating rate. *vph* represents the number of arriving vehicles per hour. In this way, the uniform distribution of traffic flow is avoided and more realistic traffic flows are generated.

5.3. Learning performance

The setting of parameters affects the performance of the *Q-values* learning performance. The parameter setting in the experiment is shown in Table 2.

The learning rate α affects how quickly the model can converge. However, higher learning rate means higher loss in the learning process, while lower learning rate means the longer convergence runtime. After comparing the performance of $\alpha = 0.01, 0.05, 0.1$, we set $\alpha = 0.05$. An example of the convergence of DCL-AIM algorithm in two discretization level experiments is shown in Fig. 7.

As can be seen from Fig. 7, the episodic reward is well converged after approximately 5000 episodes for each discretization level experiment, and it also converges well in the case of heavy traffic. Note that the fluctuations in reward afterwards are caused by the $\varepsilon - greedy$ policy.

The construction of reward function also has a significant influence on the performance and convergence of DCL-AIM algorithm. Fig. 8 (left) shows a convergence example (discretization level 1 experiment) in which the agents update their *Q-values* only depending on individual rewards in all states, while Fig. 8 (right) shows an example in which *Q-values* are updated according to *QUpdate ()*. When updating and learning with individual rewards in all states, the algorithm still converges, but it will converge to a poorer policy (less episodic rewards, which means higher delay) compared to the policy converged using *QUpdate()*. The advantage of DCL-

Table 2
Parameter settings for DCL-AIM algorithm.

Greedy policy: epsilon ε	0.9
Learning rate: alpha α	0.05
Discount factor: gamma γ	0.9
Step size: Δt	0.5

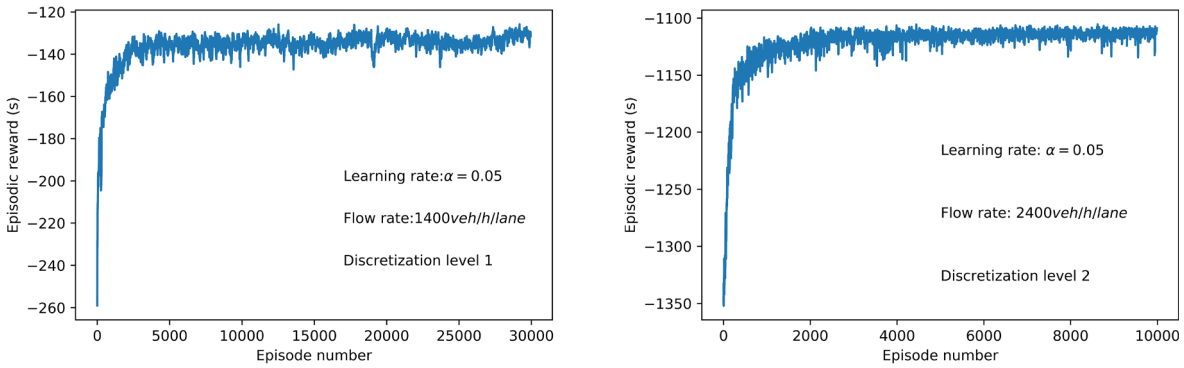

Fig. 7. An example of convergence of DCL-AIM algorithm in two discretization level experiments.

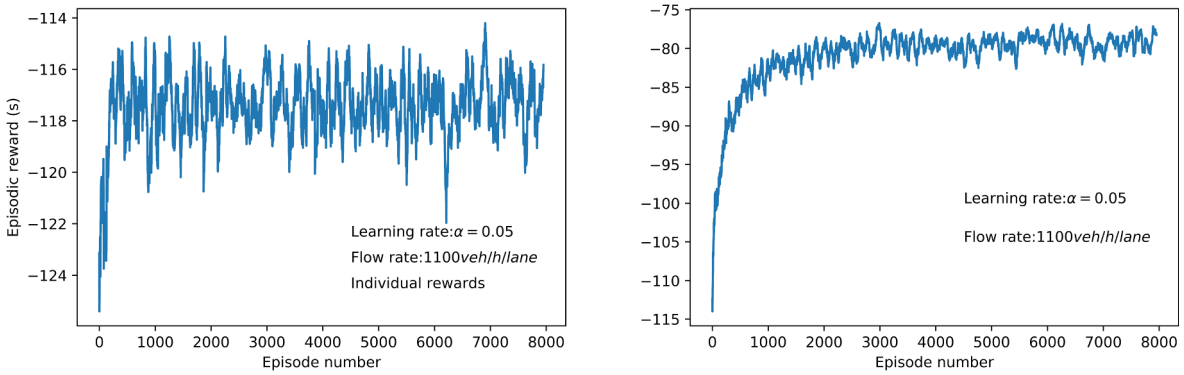

Fig. 8. Convergence of individual rewards (left), coordination learning (right).

AIM algorithm is that it can effectively decompose the difficult decision-making problem into sub-problems without losing too much global performance.

5.4. Effectiveness comparison

This section focuses on comparing the performance of FCFS-AIM and the proposed multi-agent coordination learning based AIM (DCL-AIM) after the *Q-values* are trained and learned, using the data of discretization level 1 experiment. In addition, Longest-Queue-First policy (LQF-AIM) and the traditional signal control (labeled Signal in Fig. 9) based on the Webster's method (Webster, 1958) are set as benchmarks. It is worth pointing out that after the policy is trained and learnt, the policy obtained by DCL-AIM algorithm can be read and executed in less than 10 ms at each decision step (much less than the step size) for real-world scale problems, just as other rule-based policies (FCFS, LQF and Signal).

For all control policies, for the sake of fairness, the traffic in-flow and time horizon are the same. The major road - minor road intersecting situations are simulated. The traffic flow on the major road is twice that of the minor road. And, the through-flow is twice the turning-flow on each approach. Different traffic scenarios with different traffic generating rate are simulated. The comparison of average delay per vehicle is shown in Fig. 9, where *traffic generating rate* is the through-flow rate of the major road. Because the traffic flow in this paper is randomly generated, in order to eliminate the impact of the randomness on the results when comparing the effectiveness of the policies, for each traffic generating rate scenario, the simulations were conducted 100 times. The average delay (s/veh) with 95% confidence interval of the 100 simulation runs are also shown in Fig. 9.

To compare the stability of different methods, we show the results across different simulation runs, as shown in Fig. 10. Under the same traffic generation rate scenarios, the intersection delay of each simulation is different. Traffic dynamics have the greatest impact on FCFS policy. It is because that the priority of FCFS is assigned based on the arrival time of vehicles. The amplitude of fluctuation shown in Fig. 10 reflects that FCFS-AIM is not as good as DCL-AIM in dealing with uncertainty. DCL-AIM addresses the uncertainty problem through the trade-off between exploration and exploitation.

The experimental results show that DCL-AIM significantly outperforms FCFS-AIM under different random traffic conditions. *Safety buffer* is an important concept in FCFS-AIM (Dresner, 2009), which is used to enhance the safety of the control. The less buffer the less delay, so we set the extra time safety buffer as 0s in the comparison. In our approach, the *front and rear collisions* are eliminated by the concept of occupying cells and the desired cells, DCL-AIM system does not need additional safety enhancements, the coordinated actions in DCL-AIM algorithm eliminate the conflicting situations.

From the perspective of system optimization, at a certain time point t, it seems better to give higher priority for the longer queue,

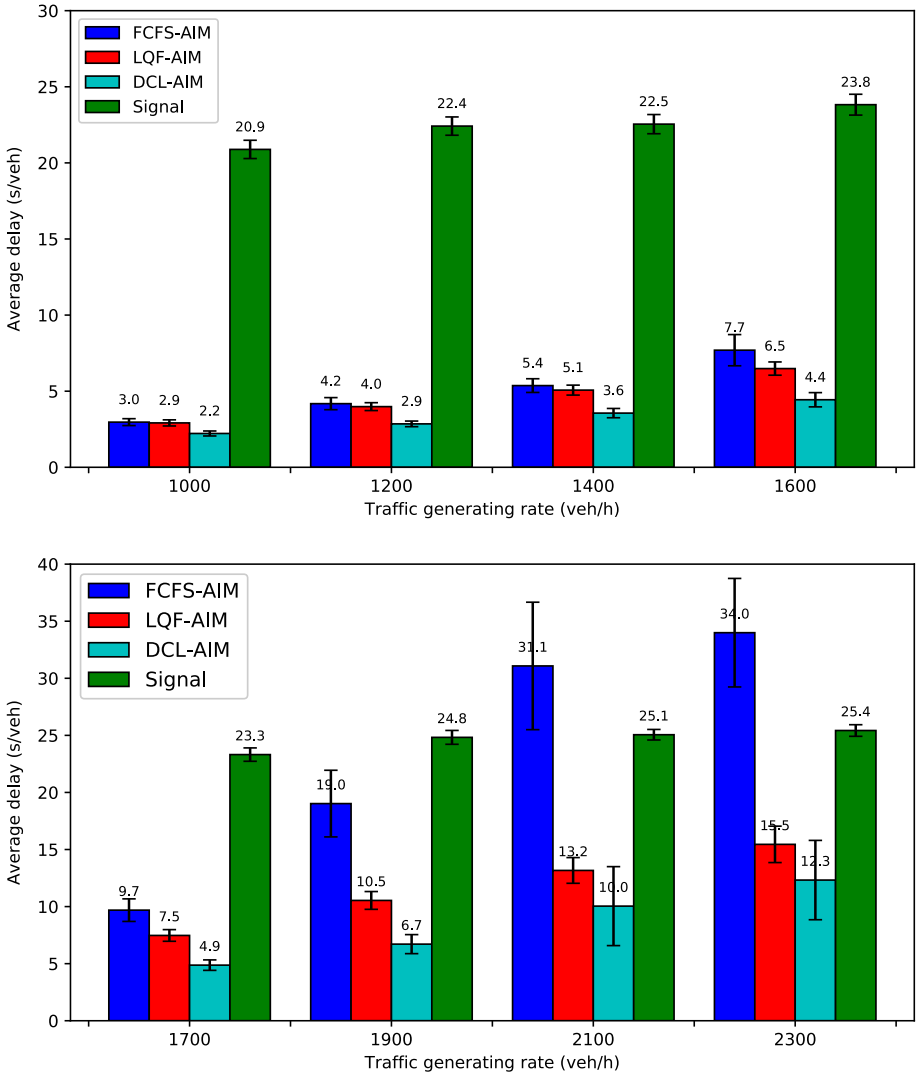

Fig. 9. Average delay comparison.

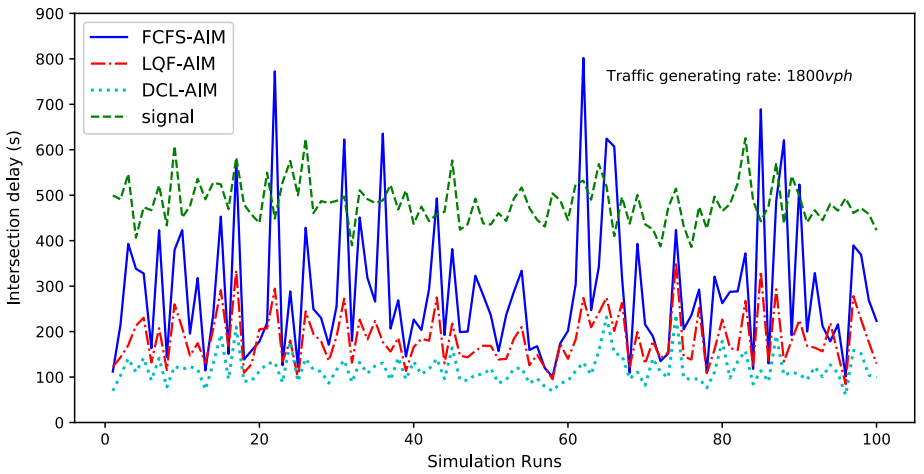

Fig. 10. Fluctuation of intersection delay of 100 simulation runs.

which is why the longer queue first (LQF) policy is often applied to intersection management, especially in the adaptive signal control. However, in the long run, it may not be the best choice.

From Fig. 9 and Fig. 10, we can see that the Webster's signal control policy is not dominant in the management of CAVs. The major reason for the higher delay of signal control (even when the traffic flow is relatively light) is that it cannot make good use of the space-time resources of the intersection. In the signal control, only up to two streams can be discharged simultaneously, while the spatiotemporal resources in other directions are "wasted". Another characteristic of signal control can also be seen from the figures, that is, its standard deviation is relatively small, indicating that the performance of signal control is little affected by the randomness of traffic flow. This is why signal control may be superior to FCFS-AIM in some cases, which can also be seen in Fig. 9. It is worth mentioning that the signal control compared in this paper is not adaptive. The adaptive and optimal signal control for CAVs is another research topic that has been studied extensively in the literature but out of the scope of this paper.

As the traffic flow approaches the intersection capacity, the intersection delay increases dramatically, and the effectiveness of all control policies begins to decline, especially FCFS-AIM policy, while DCL-AIM consistently performs the best under all the traffic generating rates. The main reason for the decline in effectiveness is the limited availability of the space-time cells of the intersection. In order to show how the spatiotemporal resources affect the performance of the control policies, we have designed some scenarios specifically.

Scenario 1. Lane 2 is congested, with the traffic in-flow rate 3000 veh/h, while the arriving vehicles on lane 5, lane 8 and lane 11 are randomly generated as in-flow rate 1500 veh/h. No vehicles are generated on other lanes.

Scenario 2. Lane 2 and lane 8 are congested, with the traffic in-flow rate 3000 veh/h, while the arriving vehicles on other lanes are randomly generated as 1000 veh/h.

Scenario 3. All of Lane 2, lane 5, Lane 8 and Lane 11 are congested, with the traffic in-flow rate 3000 veh/h. No vehicles are generated on other lanes.

For each scenario, we run the simulation 20 times, the averaged intersection delay of the policies is shown in Table 3.

For an asymmetrical traffic flow designed in Scenario 1, there are still spatiotemporal resources available in the other three directions, so all the three AIM control policies can manage the intersection well. While, for Scenario 2 and 3 where congestion is developing in all directions, there are relatively less spatiotemporal resources can be used, it is difficult for all three strategies to handle. However, it is promising to see that DCL-AIM algorithm still can produce better control policy.

Fig. 11 shows the step delay comparison of the three control policies in a 2-min simulation of Scenario 3. As can be seen, DCL-AIM maintains a better performance at every time step and needs less time to evacuate the congestion.

5.5. Results of discretization level 2

The proposed solution technique is readily applicable to different levels of discretization. The intersection delay comparison results of discretization level 2 are shown in Fig. 12. From the comparison results and trends shown in Fig. 12, we can draw the same conclusions as the experiment of discretization level 1, that the proposed DCL-AIM algorithm can learn a better policy and the FCFS-AIM policy is most affected by the randomness of traffic flow.

6. Conclusions

AIM is an alternative intersection control tailored for CAVs. With the wireless communication technologies and advanced computer control systems, CAVs have the ability to benefit the traffic system substantially. Although CAVs in AIM system can be programmed with behaviors designed in advance, for example FCFS, it is often necessary that they learn new behaviors online, such that the performance of the agent and the whole multi-agent traffic system gradually improves. In this paper, a multi-agent reinforcement learning approach is proposed to learn the optimal policy from the real-time environment states.

We model the crossing processes of CAVs as multi-agent MDPs, in which vehicle agents cooperate to maximize rewards with collision-free constraints. The optimal sequential actions of vehicles can be given by the proposed DCL-AIM algorithm. The main feature of the proposed approach is to explicitly identify and dynamically adapt agent coordination needs during the learning process so that the curse of dimensionality and environment nonstationarity problems in multi-agent learning can be alleviated.

A variety of traffic conditions are simulated within a typical 4-way-6-lane intersection, demonstrating the effectiveness of the proposed model. In order to more realistically simulate the traffic flow, vehicles are randomly generated rather than uniformly distributed. The comparative analysis is conducted between the DCL-AIM and FCFS-AIM, with LQF policy and the Webster's signal control as benchmarks. Results shows that the optimal sequential decisions from DCL-AIM outperform the other control policies.

Table 3
Comparison of intersection delays.

Scenario	Intersection delay (s)		
	FCFS	LQF	DCL
1	333.5	297.5	111.5
2	3589.6	2752.4	2582.2
3	3585.2	3036.4	2542.1

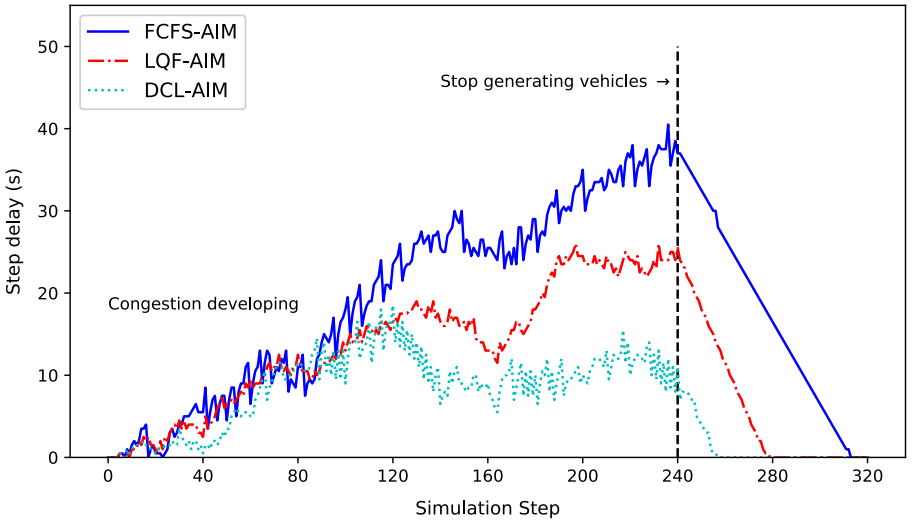

Fig. 11. Step delay comparison.

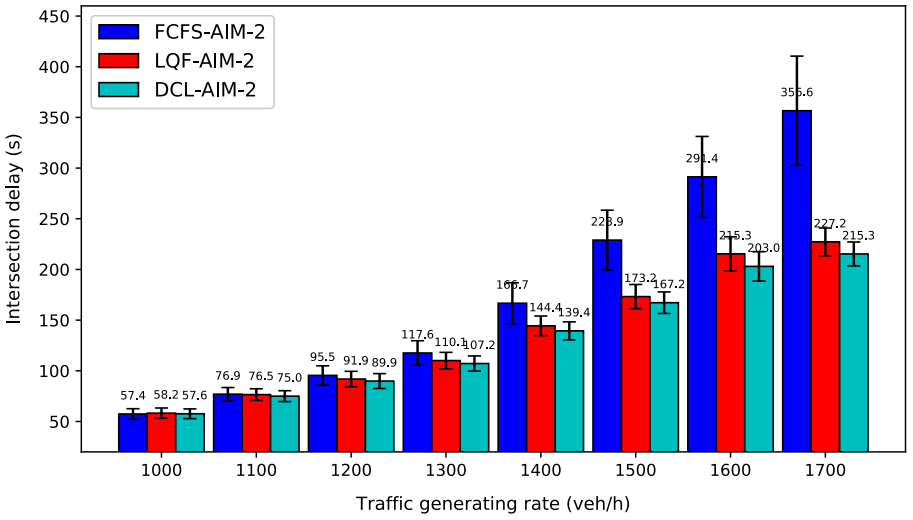

Fig. 12. Intersection delay comparison of discretization level 2.

In this paper, the solution algorithms are based on the assumptions that the environment is fully observable in the case of an isolated intersection. In the future, we will try to extend current work to a partially observable traffic environment and solve a network-wide intersection management problem of CAVs.

Acknowledgments

This study is supported by Singapore Ministry of Education Academic Research Fund Tier 2 MOE2017-T2-1-029.

References

Abdulhai, B., Pringle, R., Karakoulas, G.J., 2003. Reinforcement learning for true adaptive traffic signal control. J. Transp. Eng. 129, 278–285.
Arel, I., Liu, C., Urbanik, T., Kohls, A., 2010. Reinforcement learning-based multi-agent system for network traffic signal control. IET Intel. Transport Syst. 4, 128–135.
Bazzan, A.L., Klügl, F., 2014. A review on agent-based technology for traffic and transportation. Knowl. Eng. Rev. 29, 375–403.
Busoniu, L., Babuska, R., De Schutter, B., 2008. A comprehensive survey of multiagent reinforcement learning. IEEE Trans. Syst. Man Cybernet. Part C 38, 156–172.
Carlino, D., Boyles, S.D., Stone, P., 2013. Auction-based autonomous intersection management. In: 2013 16th International IEEE Conference on Intelligent Transportation Systems-(ITSC). IEEE, pp. 529–534.
Chen, H., An, B., Sharon, G., Hanna, J.P., Stone, P., Miao, C., Soh, Y.C., 2018. Dyetc: Dynamic electronic toll collection for traffic congestion alleviation. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp. (AAAI-18)..
Chen, L., Englund, C., 2016. Cooperative intersection management: a survey. IEEE Trans. Intell. Transp. Syst. 17, 570–586.
Dresner, K., Stone, P., 2004. Multiagent traffic management: a reservation-based intersection control mechanism. In: Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2. IEEE Computer Society, pp. 530–537.

Dresner, K., Stone, P., 2006. Human-usable and emergency vehicle-aware control policies for autonomous intersection management. In: Fourth International Workshop on Agents in Traffic and Transportation (ATT), Hakodate, Japan.
Dresner, K., Stone, P., 2008. A multiagent approach to autonomous intersection management. J. Artificial Intell. Res. 31, 591–656.
Dresner, K.M., 2009. Autonomous intersection management. Technical Report. University of Texas at Austin Austin United States.
Fagnant, D.J., Kockelman, K., 2015. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. Transp. Res. Part A: Policy Pract. 77, 167–181.
Fajardo, D., Au, T.C., Waller, S., Stone, P., Yang, D., 2011. Automated intersection control: performance of future innovation versus current traffic signal control. Transp. Res. Rec.: J. Transp. Res. Board, 223–232.
Guestrin, C., Venkataraman, S., Koller, D., 2002. Context-specific multiagent coordination and planning with factored mdps. AAAI/IAAI 253–259.
Guo, Y., Ma, J., Xiong, C., Li, X., Zhou, F., Hao, W., 2019. Joint optimization of vehicle trajectories and intersection controllers with connected automated vehicles: Combined dynamic programming and shooting heuristic approach. Transp. Res. Part C: Emerg. Technol. 98, 54–72.
Hausknecht, M., Au, T.C., Stone, P., 2011. Autonomous intersection management: ,ulti-intersection optimization. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 4581–4586.
Isele, D., Cosgun, A., Subramanian, K., Fujimura, K., 2017. Navigating intersections with autonomous vehicles using deep reinforcement learning. arXiv preprint arXiv:<1705.01196>.
Jin, Q., Wu, G., Boriboonsomsin, K., Barth, M., 2012. Multi-agent intersection management for connected vehicles using an optimal scheduling approach. In: 2012 International Conference on Connected Vehicles and Expo (ICCVE), IEEE, pp. 185–190.
Kenney, J.B., 2011. Dedicated short-range communications (dsrc) standards in the united states. Proc. IEEE 99, 1162–1182.
Kok, J.R., Vlassis, N., 2004. Sparse cooperative q-learning. In: Proceedings of the Twenty-first International Conference on Machine Learning. ACM, pp. 61.
Lee, J., Park, B.B., Malakorn, K., So, J.J., 2013. Sustainability assessments of cooperative vehicle intersection control at an urban corridor. Transp. Res. Part C: Emerg. Technol. 32, 193–206.
Levin, M.W., Boyles, S.D., Patel, R., 2016. Paradoxes of reservation-based intersection controls in traffic networks. Transp. Res. Part A: Policy Pract. 90, 14–25.
Levin, M.W., Fritz, H., Boyles, S.D., 2017. On optimizing reservation-based intersection controls. IEEE Trans. Intell. Transp. Syst. 18, 505–515.
Li, Z., Chitturi, M., Zheng, D., Bill, A., Noyce, D., 2013. Modeling reservation-based autonomous intersection control in vissim. Transp. Res. Rec.: J. Transp. Res. Board 81–90.
Littman, M.L., 2001. Value-function reinforcement learning in markov games. Cognit. Syst. Res. 2, 55–66.
Lokhandwala, M., Cai, H., 2018. Dynamic ride sharing using traditional taxis and shared autonomous taxis: a case study of nyc. Transp. Res. Part C: Emerg. Technol. 97, 45–60.
Malikopoulos, A.A., Cassandras, C.G., Zhang, Y.J., 2018. A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections. Automatica 93, 244–256.
Mannion, P., Duggan, J., Howley, E., 2016. An experimental review of reinforcement learning algorithms for adaptive traffic signal control. In: Autonomic Road Transport Support Systems. Springer, pp. 47–66.
Matignon, L., Laurent, G.J., Le Fort-Piat, N., 2012. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. Knowl. Eng. Rev. 27, 1–31.
Melo, F.S., Veloso, M., 2011. Decentralized mdps with sparse interactions. Artif. Intell. 175, 1757–1789.
Mirheli, A., Hajibabai, L., Hajbabaie, A., 2018. Development of a signal-head-free intersection control logic in a fully connected and autonomous vehicle environment. Transp. Res. Part C: Emerg. Technol. 92, 412–425.
Mirheli, A., Tajalli, M., Hajibabai, L., Hajbabaie, A., 2019. A consensus-based distributed trajectory control in a signal-free intersection. Transp. Res. Part C: Emerg. Technol. 100, 161–176.
Nagel, K., Schreckenberg, M., 1992. A cellular automaton model for freeway traffic. Journal de physique I 2, 2221–2229.
Peng, J., Williams, R.J., 1994. Incremental multi-step q-learning. In: Machine Learning Proceedings 1994. Elsevier, pp. 226–232.
Qi, X., Luo, Y., Wu, G., Boriboonsomsin, K., Barth, M., 2019. Deep reinforcement learning enabled self-learning control for energy efficient driving. Transp. Res. Part C: Emerg. Technol. 99, 67–81.
Roth, M., Simmons, R., Veloso, M., 2007. Exploiting factored representations for decentralized execution in multiagent teams. In: Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multiagent Systems. ACM, pp. 72.
Schepperle, H., Böhm, K., 2007. Agent-based traffic control using auctions. In: International Workshop on Cooperative Information Agents. Springer, pp. 119–133.
Schepperle, H., Böhm, K., 2008. Auction-based traffic management: towards effective concurrent utilization of road intersections. In: 2008 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing. IEEE, pp. 105–112.
Sun, W., Zheng, J., Liu, H.X., 2017. A capacity maximization scheme for intersection management with automated vehicles. Transp. Res. Procedia 23, 121–136.
Sutton, R.S., Barto, A.G., 1998. Reinforcement learning: An introduction, vol. 1 MIT Press Cambridge.
Tan, M., 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In: Proceedings of the Tenth International Conference on Machine Learning, pp. 330–337.
Vasirani, M., Ossowski, S., 2012. A market-inspired approach for intersection management in urban road traffic networks. J. Artificial Intell. Res. 43, 621–659.
Watkins, C.J., Dayan, P., 1992. Q-learning. Mac. Learn. 8, 279–292.
Webster, F.V., 1958. Traffic signal settings. Technical Report.
Yu, C., Zhang, M., Ren, F., Tan, G., 2015. Multiagent learning of coordination in loosely coupled multiagent systems. IEEE Trans. Cybernet. 45, 2853–2867.
Zhang, Y.J., Malikopoulos, A.A., Cassandras, C.G., 2016. Optimal control and coordination of connected and automated vehicles at urban traffic intersections. In: American Control Conference (ACC), 2016, IEEE. pp. 6227–6232.
Zhu, F., Ukkusuri, S.V., 2014. Accounting for dynamic speed limit control in a stochastic traffic environment: A reinforcement learning approach. Transp. Res. Part C: Emerg. Technol. 41, 30–47.
Zhu, F., Ukkusuri, S.V., 2015. A linear programming formulation for autonomous intersection control within a dynamic traffic assignment and connected vehicle environment. Transp. Res. Part C: Emerg. Technol. 55, 363–378.
Zhu, M., Wang, X., Wang, Y., 2018. Human-like autonomous car-following model with deep reinforcement learning. Transp. Res. Part C: Emerg. Technol. 97, 348–368.