Reinforcement Learning for Cooperative Overtaking

Chao Yu
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
cy496@dlut.edu.cn

Xin Wang
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
1109525927@qq.com

Jianye Hao
School of Software, Tianjin University
Tianjin, China
jianye.hao@tju.edu.cn

Zhanbo Feng
School of Computer Science & Technology, Dalian University of Technology, Dalian, China
571102482@qq.com

ABSTRACT

This paper solves the cooperative overtaking problem in autonomous driving using reinforcement learning techniques. Learning in such a situation is challenging due to vehicular mobility, which renders a continuously changing environment for each learning vehicle. Without no explicit coordination mechanisms, inefficient behaviors among vehicles might cause fatal uncoordinated outcomes. To solve this issue, we propose two basic coordination models to enable distributed learning of cooperative overtaking maneuvers in a group of vehicles. Extension mechanisms are then presented to make these models workable in more complex and realistic settings with any number of vehicles. Experiments verify that, by capturing the underlying consistency of identities or positions during vehicles' movement, efficient coordinated behaviors can be achieved simply through vehicles' local learning interactions.

KEYWORDS

Reinforcement Learning; Autonomous Driving; Coordination Graph; Cooperative Overtaking; Multiagent Learning

ACM Reference Format:
Chao Yu, Xin Wang, Jianye Hao, and Zhanbo Feng. 2019. Reinforcement Learning for Cooperative Overtaking. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

Autonomous driving is one of the most important AI applications and has attracted extensive interest in recent years from both technology companies and research institutes, due to its promise in improving safety, efficiency, energy consumption, comfort and mobility [5]. In order to enable fully autonomous driving functionalities, a vehicle should form safe, controllable, and robust driving policies of following, overtaking, changing lanes or taking turns, etc. However, since there are many possible scenarios (*e.g.*, innumerable traffic patterns and varying driving styles), manually tackling all possible cases will likely yield too simplistic and suboptimal policies. Even with significant expert domain knowledge, hand crafting a controller that operates effectively in all cases may not be always feasible.

These challenges naturally suggest using machine learning approaches, particularly *Reinforcement Learning* (RL) [27], to learn optimal driving strategies that are able to adapt to changing environments and unseen scenarios. A large number of studies have applied RL in various settings of autonomous driving, using tabular forms of *Q-learning* [15, 18], RL with *function approximations* [23] or *policy gradient* approaches [4]. The recent integration of *deep learning* has greatly promoted the successful applications of RL in solving real-world complex control problems in autonomous driving [2, 9, 24, 30].

All these studies, however, only focus on learning a driving policy for a single vehicle. This is contradictory to the fact that autonomous driving is a typical *Multi-Agent System* (MAS), due to the coexistence of multiple vehicles and their concurrent decision making and interaction processes [25]. In fact, the development of wireless communication and vehicle automation technologies has rapidly lead to the advent of *Connected Autonomous Vehicles* (CAVs) [16], that are capable of not only collecting real-time traffic data such as individual vehicle's state information, traveling maneuver and trajectories but also sharing these data with other surrounding vehicles. In CAVs, a major technical issue is how to design high-level strategic control policies to coordinate multiple vehicles for avoiding potential conflicts. Along this paradigm, several studies investigated RL in multi-vehicle applications, such as *cooperative adaptive cruise control* (CACC) [4], cooperative lane changing [17], and intersection navigation [10]. All these studies, however, directly applied simple distributed RL in a multi-vehicle context. Since no explicit coordination mechanisms have been applied, this simplification may result in inefficient driving policies and sometimes fatal uncoordinated outcomes [10].

This paper aims to solve the coordination problems in autonomous driving using *multiagent RL* (MARL) techniques [3]. Particularly, we focus on high-level strategic decision making of *following* or *overtaking* in a group of vehicles on highways. This problem is considered because determining if and when to make lane changing and overtaking maneuvers are the two main strategies for autonomous driving [19]. We cast such coordination problems as *factored MDP* problems [6], drawing on *Coordination Graph* (CG) to explicitly

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

model dependency among vehicles and decrease computation complexity of the overall decision making problem. However, directly applying CG in autonomous driving is a tricky problem, due to the continuously changing topology of the moving vehicles. In such situations, it is impossible to build a constant CG for the vehicles. This dynamics can significantly violate the basic assumption of static topology in traditional coordinated learning approaches based on CG.

To solve this problem, two basic coordination models are proposed in this paper to enable effective distributed learning of cooperative maneuvers in a group of vehicles. The first one is the *identity-based* approach that distinguishes each vehicle's identification and builds a new CG once the topology has changed. The other one is the *position-based* approach that builds a constant CG based on the relative positions of the vehicles. Several mechanisms are then proposed to extend these two basic models in order to make them workable in more complex and realistic settings with any number of vehicles. The primary difficulty for this extension is how to properly tackle the conflicts and coordination among different subgroups of vehicles, each of which is governed by a basic coordination model. Experiments verified the benefits of the proposed learning approach, compared to other approaches that learn without coordination or rely on some mobility models and expert driving rules.

2 COORDINATED MARL AND CG

The major difficulty in MARL is that the computation complexity grows exponentially with the number of agents. One way to alleviate this problem is to exploit certain level of independence among agents. *Coordination Graph* (CG) [7] is one of such effective approaches, which decomposes global payoff function $Q(\mathbf{s}, \mathbf{a})$ into a linear combination of local payoff functions. This decomposition can be depicted using an undirected graph $G = (V, E)$, in which each node $i \in V$ represents an agent and an edge $(i, j) \in E$ indicates that the corresponding agents have to coordinate their actions. Allowing payoff functions defined over at most two agents, the global payoff function $Q(\mathbf{s}, \mathbf{a})$ can be given by:

$$Q(\mathbf{s}, \mathbf{a}) = \sum_{(i,j)\in E} Q_{ij}(s_{ij}^{-}, a_i, a_j) \tag{1}$$

where $\mathbf{s} = \langle s_1, ..., s_n \rangle \in S$ and $\mathbf{a} = \langle a_1, ..., a_n \rangle \in A$ are the joint state and joint action of all agents, respectively, and s_{ij}^{-} represents the relevant state variables for agent i and j.

The main goal of CG is to find a coordination strategy of actions for the agents to maximize $Q(\mathbf{s}, \mathbf{a})$ at state $\mathbf{s}$. The *Variable Elimination* (VE) algorithm [7] can be applied for this purpose. In VE, an agent first collects all payoff functions related to its edges before it is eliminated. It then computes a conditional payoff function which returns the maximal value it is able to contribute to the system for every action combination of its neighbors, and a best-response function (or conditional strategy) which returns the action corresponding to the maximizing value. The conditional payoff function is then communicated to its neighbors and the agent is eliminated from the graph.

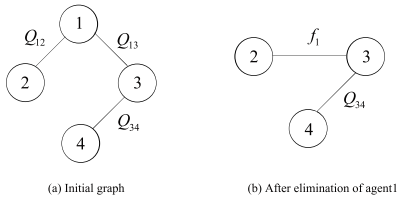

Figure 1: CG for a 4-agent coordination problem.

Figure 1 shows the coordination graph for an example of four agents and its elimination process of agent 1. The overall value function can be written as follows:

$$Q(\mathbf{a}) = Q_{12}(a_1, a_2) + Q_{13}(a_1, a_3) + Q_{34}(a_3, a_4), \tag{2}$$

We first eliminate agent 1. This agent does not depend on the local payoff function Q_{34}. Therefore, the maximization of $Q(\mathbf{a})$ can be written as:

$$\max_{a_1,a_2,a_3,a_4} Q(\mathbf{a}) = \max_{a_2,a_3,a_4} \{Q_{34}(a_3, a_4) + f_1(a_2, a_3)\}, \tag{3}$$

where $f_1(a_2, a_3) = \max_{a_1}[Q_{12}(a_1, a_2) + Q_{13}(a_1, a_3)]$ is a conditional payoff function indicating the maximal value that agent 1 can achieve given the actions of agent 2 and 3. When there is the last remaining agent, the whole process is reversed in which each agent computes its optimal (unconditional) action from its best-response function and fixed actions from its neighbors. The VE algorithm has been proved to produce the optimal joint action and the coordination result does not depend on the elimination order.

Once the optimal joint action for a given CG structure can be computed, it is then possible to employ coordinated RL approaches [7] for sequential decision making problems. The basic idea is to maintain a local value function for each agent, which can be represented, e.g., by a neural network, and update it along the gradient of the global square TD error, resulting the following update rule $\Delta w_i = \alpha[r + \gamma \max_{\mathbf{a}} Q(\mathbf{s}', \mathbf{a}, w) - Q(\mathbf{s}, \mathbf{a}, w)]\nabla_{w_i} Q_i(s_i, a_i, w_i)$, where the optimal joint action $\mathbf{a}$ in the new state $\mathbf{s}'$ can be computed using VE on the CG. Due to page limit, readers are referred to [6, 7] for more details of related background information.

3 RL FOR AUTONOMOUS DRIVING

3.1 MDP Formalization

The decision making process of each vehicle is modeled as a *Markov Decision Process* (MDP) by a tuple of $\langle S, A, T, R \rangle$, in which S is a finite set of states, A a finite set of actions, T a transition function defined as $T : S \times A \times S \to [0, 1]$ and R a reward function defined as $R : S \times A \times S \to R$. Each vehicle's current state is composed of all the factors that can impact its decision making: the *lane* l where the vehicle is currently on ($l = 1$ for the driving lane and $l = 2$ the overtaking lane), the *longitudinal speed* v_0 of the vehicle, and the *distance* d_i and *speed* v_i of its four neighboring vehicles ($i = 1, = 2$ for the lead, lag neighboring vehicles on the

driving lane, respectively, and $i = 3, = 4$ for the lead, lag neighboring vehicles on the overtaking lane, respectively). This consists of a ten tuple of state $\langle l, v_0, d_i, v_i \rangle (i = 1, 2, 3, 4)$ for each vehicle. However, since the distance and speed of neighboring vehicles can be synthesized into the *remnant reaction time* (RRT), the state dimension can be reduced to five as $\langle l, t_i \rangle (i = 1, 2, 3, 4)$, where $t_1 = (d_1 - d_{s_1})/v_0$ and $t_3 = (d_3 - d_{s_3})/v_0$ are the focal vehicle's RRT on the driving and overtaking lane[1], respectively, while $t_2 = (d_2 - d_{s_2})/v_2$ and $t_4 = (d_4 - d_{s_4})/v_4$ are the lag vehicle's RRT on the driving and overtaking lane, respectively.

To model the local observability of vehicles, two valuables f_v and b_v are used to represent the forward and backward *view range* of a vehicle, which are set to $160m$ in this study. This means that a vehicle can only receive the speed and position information from neighboring vehicles within $160m$ for the state representation[2]. As we aim at designing high-level strategic controllers, two actions are considered here: *driving on the driving lane* a_d, which means following in a limited speed of $v_t = 30m/s$ on the driving lane, and *driving on the overtaking lane* a_o, which means following in a limited speed of $v_o = 40m/s$ on the overtaking lane[3]. Finally, the reward function can be defined as follows:

$$r_s = \begin{cases} \min(t_1, t_2) & \text{if } l = 1 \text{ and } d_1 > 3 \text{ and } d_2 > 3 \\ \min(t_3, t_4) & \text{if } l = 2 \text{ and } d_3 > 3 \text{ and } d_4 > 3 \\ -5 & \text{else} \end{cases} \quad (4)$$

Eq. (4) means that when the vehicle is on the driving lane ($l = 1$) and the distance between the vehicle and the lead/lag vehicle is not too close ($d_1 > 3$ and $d_2 > 3$), the reward is the smaller remnant reaction time on the driving lane. This means that larger t_1 and t_2 indicate more remnant reaction time during emergency, and thus higher safety. The same explanation applies for the overtaking lane. In all other situations when any d_i is less than 3 meters, the vehicle is heavily penalized by -5 due to being too close to the lead/lag vehicle. Note that we also implemented other versions of reward function, e.g., by introducing a gradient penalty according to relative speed and distance between two vehicles, but found similar result patterns. As evaluation of exact value of the penalty is not the main focus of the paper, this result is omitted here due to page limits.

The reward function is to evaluate the strategic decision making of *following* or *overtaking*, not explicit driving actions such as *acceleration* or *deceleration*. So, if a vehicle is moving at a low speed and far away from the former vehicle, in which case the highest forward remnant reaction time might occur, the vehicle cannot stay in this safe state as it may keep on choosing *following* to increase its speed according to the car-following mobility model, and thus the reward will be decreased accordingly. Moreover, the reward function is defined over the minimum of the forward and backward remnant reaction time. Thus, the overall reward can be quite low if a vehicle is moving slowly at a far distance from the former vehicle, since this may bring about little backward remnant reaction time. Therefore, a vehicle should learn to make the best trade-off between the forward and backward safety situation, which nicely mimics real-life driving behaviors. In fact, taking into account not only the car ahead, but the car behind allows information to flow in both directions, thus reducing the instability problem in traffic flow [8]. Also, many other criterion are available to evaluate the decision making of an autonomous vehicle, e.g., the *smoothness* to evaluate how fast the speed has been changed to avoid uncomfortable drastic acceleration or deceleration, and the *number of lane change* to evaluate the effectiveness of decision making. The RRT is considered here as safety is the ultimate goal of autonomous driving. However, other criteria can also be intergraded into the reward function using different weight parameters, which is left for future investigation.

To validate the above MDP formulation, we apply basic tubular form of Q learning to solve it, with learning rate of 0.1, discount factor of 0.95, and initial exploration rate ε of 0.1 decaying to 0.9ε every 10 episodes. The experiment involves one decision making (learning) vehicle and its four environmental vehicles randomly generated at a distance in-between [100m,160m] far away. The initial speed of all the vehicles is randomly set in-between [18m/s,27m/s] to model regular movement on highways. However, the learning vehicle can adapt its speed based on the chosen overtaking or following decisions. The RL approach is compared to the famous mobility model MOBIL [11] and an expert approach [14] to demonstrate the effectiveness of learning. An episode ends if two vehicles collide or 400 decision steps have passed. Each run consists of 1000 episodes and the final results are averaged over 30 independent runs.

Figure 2 (a) plots the dynamics of average reward using Q-learning approach with the original 10-dimension state and the reduced 5-dimension state. It is clear that the RL approaches can converge to higher rewards than the MOBIL model and the expert rule-based approach. The vehicle makes decisions randomly at first, causing frequent lane change. As learning proceeds, the vehicle can learn to take overtaking only when necessary, significantly reducing the number of lane change. On the contrary, the vehicle using MOBIL and the expert approach is rather conservative, which means that the vehicle prefers following the front vehicles most of the time and only overtakes occasionally. That is why the minimum forward distance using MOBIL and the expert approach is smaller than that using the RL approaches. As for the average speed, the RL approaches can bring about more stable speed than the other two approaches due to the learned overtaking behaviors. Although MOBIL can achieve a bit higher speed than RL approaches due to vehicles following in a limited

[1] d_{s_i} is the *shortest safety distance* when the lead vehicle decelerates with $-6m/s^2$ and the lag vehicle decelerates with $-4m/s^2$.

[2] When a neighboring vehicle is outside of view, to still validate the state formalization, we imagine a virtual vehicle moving at the same speed with the focal vehicle at the location of $160m$. In this way, the virtual vehicle would have little impact on the decision making of the focal vehicle, but still be valid in the state definition.

[3] Following in a limited speed means that the vehicle takes the smaller value of planned speed v_p and v_t (or v_o) as traveling speed. The speed v_p can be computed using various car-following models.

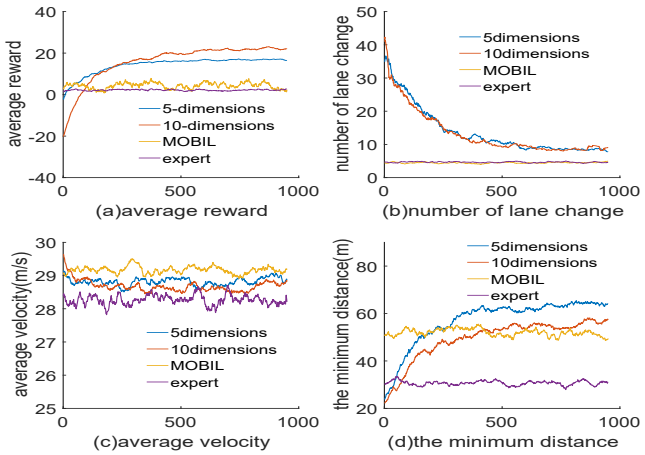

Figure 2: Performance comparison of different approaches with regard to the learning episode.

speed of $v = 40m/s$ on both lanes, it is not as safe as RL methods as given by Figure 2 (a).

3.2 Two Basic Coordination Models

Autonomous driving is a typical cooperative multiagent domain, in which all the vehicles aim at achieving the same goal of moving fast as a group and at same time keeping safe. In reality, however, some vehicles might be selfish and only care about their own short term interests. Extra trust or reputation mechanisms can be employed to penalize these free riders. Nevertheless, in this paper, we simply focus on cooperative settings by assuming the homogeneity of all the vehicles and their willingness for cooperation. The goal is to compute an overall policy π to maximize the value function $Q^{\pi}(js, ja) = E_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(js_t, ja_t) | js_0 = js\right]$, in which js and ja are respectively the joint state and action of all vehicles. To alleviate computation complexity, the overall $Q^{\pi}(js, ja)$ can be decomposed as $Q^{\pi}(js, ja) = \sum_{(i,j)\in E} Q_{ij}(s_{ij}, a_i, a_j)$, in which E is the set of edges on the CG to indicate dependency between two vehicles. The definitions of individual actions, states and rewards are the same as the single vehicle case as introduced ahead.

We are now interested in how multiple vehicles can achieve coordinated overtaking policies through their distributed and concurrent learning behaviors. Although several coordinated RL approaches on CG have been proposed previously [7], directly applying these approaches to the multi-vehicle coordination problem is infeasible due to the continuously changing dependencies of moving vehicles. This mobility can greatly change the topology of the CG at each time step, making the problem into a *Dynamic CG* (DCG) problem. On a DCG, a link may disappear from the graph due to the break of link between two agents or it may connect two distinct agents at different time steps. Learning for coordinated behaviors in such a dynamic context is thus a challenging task since agents not only need to reason about the behaviors of other agents, but also need to reason about how to adapt to these behaviors under a continuously changing environment.

Algorithm 1: *Basic Coordinated Learning Model*

1 *Initialize Q values and learning parameters*;
2 **repeat**
3 *Initialize speed and position of all vehicles V*;
4 **while** *collision or a time threshold is not met* **do**
5 $G \leftarrow$ **BuildIdCG(V)** *or* **BuildPsCG(V)**;
6 **foreach** $v_i \in V$ **do**
7 $v_i.cur\_state \leftarrow getState(v_i)$;
8 **foreach** $(v_i, v_j) \in E$ **do**
9 $s_{ij} \leftarrow getJointState(v_i, v_j)$;
10 *Compute* $\arg\max_{ja} Q(js, ja, w)$ *using* VE;
11 *Excute ja using an exploration strategy*;
12 $G' \leftarrow$ **BuildIdCG(V)** *or* **BuildPsCG(V)**
13 **foreach** $v_i \in V$ **do**
14 $v_i.next\_state \leftarrow getState(v_i)$;
15 $v_i.reward \leftarrow getReward(v_i)$;
16 **foreach** $(v_i, v_j) \in E$ **do**
17 $s'_{ij} \leftarrow getJointState(v_i, v_j)$;
18 *Compute global reward r by suming each vehicle's reward*;
19 *Compute* $\max_{ja} Q'(js', ja, w)$ *using* VE;
20 *Update Q value on each edge using Eq.*(5);
21 **until** *convergence*;

However, we can still discover some aspects of latent *consistency* on a DCG. The agents can keep their *identities* or relative spatial *positions* unchanged on a DCG no matter how they have moved on the graph. By capturing this *consistency*, a CG can then be built at each time step to model dynamic interactions among agents. We now propose two *basic coordinated learning models* to realize distributed learning among multiple vehicles. We call them basic models because only a focal vehicle and its four neighboring vehicles are involved, and these five vehicles form the basic coordination unit in autonomous driving. **Algorithm 1** gives the sketch of the basic coordinated learning model, in which $Q(js, ja, w)$ is the global Q value for all the vehicles and $Q_{ij}(s_{ij}, a_i, a_j, w_{ij})$ is the local Q value for each edge to indicate coordination between vehicle i and j. Each local Q_{ij} value is approximated by a neural network with weights w_{ij}.

3.2.1 Identity-based coordinated learning model. Since each edge in the CG indicates the influence and potential conflicts between two vehicles, there is a link between any two neighboring vehicles on the same lane, and between the closest vehicles on different lanes[4]. At each time step, each vehicle chooses its optimal action (i.e., *following* or *overtaking*) using the VE algorithm based on the CG (Line 10) and execute

[4]It should be noted that a link on a CG might link two vehicles that are out sight of each other (i.e., beyond $160m$). This is because the sight range of $160m$ is used as the state definition due to local observability, while the communication range (usually $400m - 500m$ using the vehicular dedicated short-range communications (DSRC) techniques) is for coordination between vehicles.

Algorithm 2: *BuildIdCG(V)*

Input: The set of vehicles V, $G \leftarrow \emptyset$;
Output: The identity-based coordination graph G;
for *all* $v_i \in V$ **do**
  $N \leftarrow$ *find all four nearest vehicles of* v_i;
  for *all* $v_j \in N$ **do**
    if $(v_i, v_j) \notin G$ **then**
      $G \leftarrow G + (v_i, v_j)$
return G;

the action using a predefined exploration strategy (Line 11). After the vehicles moved to new positions, a new CG can be built (Line 12). A critical problem is then how to update the Q values on previous CG using the new constructed CG at this step. In the *identity-based learning model*, it is assumed that coordination dependencies between vehicles are kept unchanged during the learning process, as long as they still satisfy the coordination constraints on the new CG (**Algorithm 2**). This means that link Q_{ij} on the original CG will still represent the coordination dependency between vehicle i and j, if there is still a link between them on the new CG. Once a CG has been built, each local Q value on the previous CG can be updated by:

$$\Delta w_{ij} = \alpha[r + \gamma \max_{ja} Q'(js', ja, w) - Q(js, ja, w)]\nabla_{w_{ij}} Q_{ij}(s_{ij}, a_i, a_j, w_{ij}) \quad (5)$$

where r is the reward for the group (Line 18), $Q(js, ja, w)$ is the global Q value which can be computed by fixing the actions actually executed by the vehicles, and $\max_{ja} Q'(js', ja, w)$ indicates the maximum Q values in a new joint state js' (Line 19), under the new constructed CG.

Algorithm 3: *BuildPsCG(V)*

Input: The set of vehicles V;
Output: The position-based coordination graph G;
$G \leftarrow \emptyset$, $DirFlag \leftarrow True$, $v_i \leftarrow TheFirstVehicle(V)$;
while *a closed loop is not formed* **do**
  if $DirFlag$ **then**
    $v_j \leftarrow$ *nearest lead vehicle on the same lane*;
    $v_k \leftarrow$ *farthest vehicle on the other lane*;
  else
    $v_j \leftarrow$ *nearest lag vehicle on the same lane*;
    $v_k \leftarrow$ *farthest vehicle on the other lane*;
  if v_j *does not exit* **then**
    $DirFlag \leftarrow\sim DirFlag$;
    $v_j \leftarrow v_k$;
  $G \leftarrow G + \{(v_i, v_j)\}$;
  $v_i \leftarrow v_j$;
return G;

3.2.2 Position-based coordinated learning model. Unlike the identity-based model, which recognizes each vehicle's identification during the process of topology change, the *position-based learning model* makes simplifications by ignoring this consistency. A CG with five links is built at each time step to link the vehicles in a sequence. The positions of the corresponding Q values on these links are kept unchanged during the learning process, no matter how the physical positions of vehicles have changed. **Algorithm 3** shows the basic procedure of building a position-based CG, where $DirFlag$ is a direction indicator. For example, we first set $DirFlag$ as clockwise and choose the leftmost vehicle v_i on one lane. The algorithm then searches all the vehicles ahead of v_i and builds the links sequentially until the rightmost vehicle. Then, the algorithm searches all the vehicles on the other lane from right to left until a closed loop is formed. The principle of only capturing the relative positions among vehicles and neglecting their identities has also been advocated by previous studies in mixed-autonomy traffic settings [28]. Eq. (5) is then applied to update the CG at each step.

Table 1: Parameters for multiple learning vehicles.

Environment	Distance. Init	Speed. Init	Distance. Visible
	$100m - 160m$	$18m/s - 27m/s$	$160m$
RL	Learning Rate	Discount Factor	Exploration Rate. Init
	0.1	0.95	0.1
BP Network	No. Input	No. Hidden	No. Output
	10	12	4

3.2.3 Experimental evaluation. To validate the proposed two basic models, we compared them to the *individual RL* approach, the MOBIL model and the *expert rule-based* approach. In the individual RL approach, each vehicle is modeled as an independent MDP and learns its strategy without coordination. Comparison to this approach thus demonstrates the effectiveness of coordination in the proposed approaches since individual learning for an optimal behavior may not guarantee an optimal group performance. As the topology of CG is constantly changing, making it impossible to apply any other existing coordinated learning methods for comparison. The benchmarks of individual learning, existing famous mobility model MOBIL and an expert rule-based strategy can fully demonstrate the benefits of our proposed methods from different perspectives. In the proposed approaches, each local Q value is defined over the joint states and actions over two linked vehicles. Thus, we adopted a neural network with 10 input, 12 hidden and 4 output neurons to represent the Q values. Each output indicates the Q value for each action combination of the two connected vehicles. Learning rate α is 0.1, and the discount factor γ is 0.95. Exploration rate ε is 0.1 and decays to 0.9ε every 10 episodes. We choose a 16km long highway to simulate the learning process. Four environmental vehicles are randomly generated at a distance in-between [100m,160m] away from the focal vehicle, and the initial speed of all vehicles is randomly set in-between [18m/s,27m/s]. A learning episode ends either when a collision occurs or 400 time steps have passed. One run has 5000 episodes and the final results are averaged over 10 runs. Table 1 summarizes the main parameter settings in the simulation.

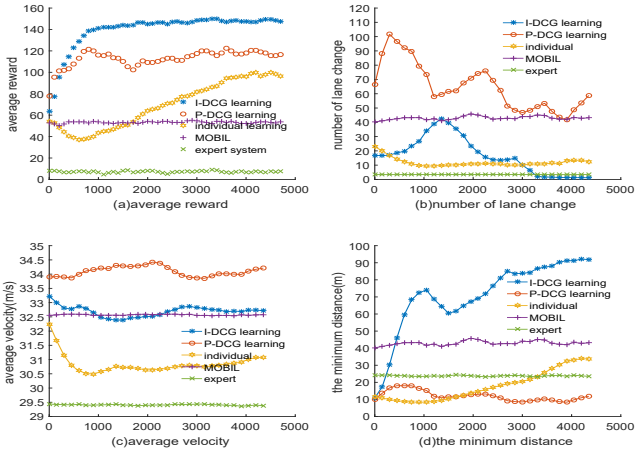

Figure 3: Average reward and microscopic mobility.

Figure 3 (a) shows the learning dynamics with regard to learning episodes over the five vehicles. *I-DCG learning* and *P-DCG learning* denote the *identity-based* and *position-based* coordinated learning approaches, respectively. It is clear that the two proposed coordinated learning approaches enable much higher average rewards than the other two approaches, which demonstrates the benefits of coordination among vehicles. Upon a deeper investigation on the motion dynamics in the vehicle group, we found that vehicles using the position-based approach were clustered together more closely and took overtaking actions more often than vehicles using other approaches. This frequent overtaking enables vehicles to move at a faster speed. The identity-based approach, however, mainly learnt a policy that all vehicles were moving at a safe distance and made overtaking actions occasionally. That is why the identity-based approach can achieve a safer policy than the position-based approach. All these phenomena can be observed in Figure 3 (b)-(d), which respectively plot the dynamics of lane change, average speed, and minimum forward distance using different approaches.

4 THREE EXTENSION MECHANISMS

When extending the basic models to any number of vehicles, several issues need to be carefully resolved. The first is how to divide the vehicle group into several sub-groups, each of which can be expressed by a basic coordination model. The second is how to deal with the conflict between vehicles belonging to different sub-groups. The last is how to realize coordinated learning in each sub-group, given the changing topology of vehicles and influence from other sub-groups.

4.1 The Extension Mechanisms

The sequential coordination mechanism. In order to divide the vehicle group into sub-groups (**Algorithm 5**), the lead vehicle on the driving lane is labeled as the focal vehicle and then its four nearest vehicles on both driving and overtaking lanes can be determined (line 3-5). This makes

Algorithm 4: *The Sequential Extension Mechanism*

```
1  Initialize Q values and learning parameters;
2  repeat
3      Initialize all vehicles V;
4      while collision or predefined time threshold is not met
       do
5          G_set, V_f ← BuildCurCG(V); % Algorithm 5
6          for all G_i ∈ G_set do
7              Conflict vehicles fix its action as in G_{i-1};
8              Compute arg max_ja Q_i(js, ja, w) by VE;
9              Excute ja using an exploration strategy;
10         G'_set ← BuildNextCG(V, V_f); % Algorithm 6
11         Each vehicle acquires its reward r_i;
12         for all G_i ∈ G'_set do
13             R_i ← sum each vehicle's reward in G_i;
14             Conflict vehicles fix its action as in G'_{i-1};
15             Compute max_ja Q'_i(js', ja, w) using VE;
16         for all G_i ∈ G_set do
17             Update Q values on G_i by Eq.(5) using R_i;
18 until convergence;
```

the first basic coordination group (line 6). Then, the next neighboring vehicle on the driving lane that has not been included in the previous sub-group is labeled to be the second focal vehicle, and its four neighboring vehicles can be determined. This process continues until all the vehicles on the driving lane have been covered (line 8). For those vehicles that have not been included in any group, they will learn individually. **Algorithm 4** gives the sequential coordination mechanism, in which G_i denotes sub-group i, G_{set} and G'_{set} represent the set of sub-groups before and after the change of CG topology, V_f is the set of all the focal vehicles, and R_i is reward for sub-group G_i. After the vehicles have been divided into several sub-groups (line 5), the vehicles in each sub-group apply VE algorithm to choose the optimal actions (Line 6-9). The proposed sequential coordination mechanism determines the optimal actions in the ordered sub-groups one by one (thus called *sequential*). More specifically, it starts from the first sub-group and determines the optimal actions for the vehicles in this group. Then, it moves to determine the optimal actions for the next sub-group. However, some vehicles might belong to two neighboring sub-groups at the same time, and the optimal actions of these vehicles have already been determined in the previous sub-group (Line 7). In this case, the remaining vehicles in the next sub-group can determine their optimal actions by using VE that fixes the actions of those vehicles co-existing in both sub-groups (line 8-9). After the vehicles moved to new positions, change of topology makes it difficult to recognize the sub-group at previous time step. To solve this problem, the focal vehicles at previous time step are still labeled as the focal vehicles at current time step (line 10). The new CG is then used to update the Q values in previous CG using Eq. (5) (Line 17).

Figure 4 gives an illustration of the sequential coordination mechanism using the identity-based learning model when

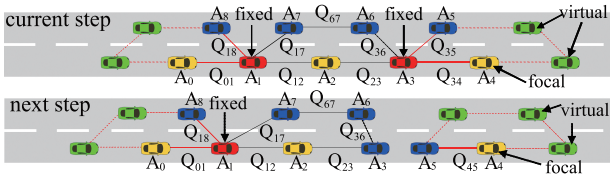

Figure 4: Illustration of the sequential mechanism.

Algorithm 5: BuildCurCG(V)

Input: The set of vehicles V;
Output: Set of subgroups G_{set}, set of focal vehicles V_f;

$V_f \leftarrow \emptyset$, $G_{set} \leftarrow \emptyset$;
repeat
  $v_{focal} \leftarrow$ *the first vehicle on the driving lane and not in any basic model*;
  $V_f \leftarrow V_f + \{v_{focal}\}$;
  $V \leftarrow v_{focal}$ *and its four neighbors*;
  $G \leftarrow$ **BuildIdCG(V)** *or* **BuildPsCG(V)**;
  $G_{set} \leftarrow G_{set} + G$;
until *each vehicle on driving lane is in a basic model*;
return G_{set}, V_f;

Algorithm 6: BuildNextCG(V,V_f)

Input: The set of vehicles V and the set of focal vehicles V_f;
Output: The updated G_{set};

$G_{set} \leftarrow \emptyset$;
for *all* $v_i \in V_f$ **do**
  $V \leftarrow v_i$ *and its four neighbors*;
  $G \leftarrow$ **BuildIdCG(V)** *or* **BuildPsCG(V)**;
  $G_{set} \leftarrow G_{set} + G$;
return G_{set};

vehicles are moving from the left to the right. At current step, A_4, A_2 and A_0 are determined as the focal vehicle in each sub-group. The action of A_3 has been determined in the group of A_4. When the vehicles in group of A_2 are to determine their joint optimal actions, the action of A_3 is fixed as the action determined in previous sub-group, and the remaining vehicles apply VE to find their optimal actions.

The concurrent coordination mechanism. In this mechanism, a vehicle belonging to two neighboring sub-groups first determines its optimal action in each sub-group, and then computes the loss of global Q value for each sub-group as if it has chosen the opposite action. The vehicle will choose the action with a lower loss of global Q value to indicate a smaller influence on the sub-group if the vehicle has changed its original optimal action. To be more clear, consider the situation in Figure 4, in which A_4, A_2 and A_0 are the focal vehicles, and the corresponding sub-groups are G_1, G_2 and G_3 respectively. For sub-groups G_1 and G_2, assume that optimal actions of conflicting vehicle A_3 in G_1 and G_2 be a_1 and a_2, respectively, and the maximum Q value in G_1 and G_2 be Q_1 and Q_2, respectively. The vehicle then computes the maximum Q value in G_2, denoted as Q_2', by fixing its action as a_1. The loss of Q value is then $\triangle Q_2 = Q_2 - Q_2'$. Similarly, $\triangle Q_1$ can be computed. If $\triangle Q_1 < \triangle Q_2$, A_3 will choose action a_2 as it will cause a lower loss of value in G_1. Otherwise, A_3 will choose action a_1. Similarly, if there is a conflict of A_1 in action selection between G_2 and G_3, the process of minimizing the loss of Q value is conducted in the same way by fixing the action of A_3.

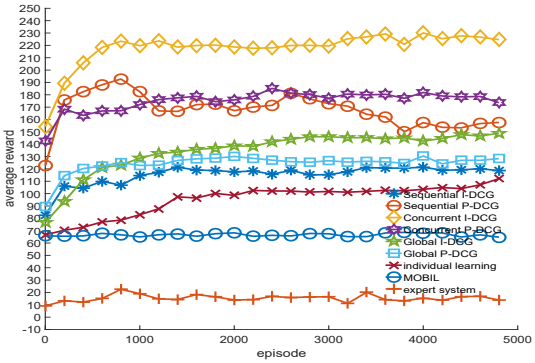

Figure 5: Performance of different approaches.

The global coordination mechanism. This mechanism considers all the vehicles as a single group by connecting them into a CG in the form of a close loop. The basic coordination models can then be easily extended to this larger CG. The only difference is that each edge (Q value) in the CG is now updated based on the global CG, not on the small unit of CG as did in the basic models. The global coordination mechanism is simple and easy to implement. However, since all the vehicles are considered as a single group and each two of them should build a link, the computational complexity becomes exponential in the size of vehicle group. The high computational complexity will definitely cause delay in message passing and thus reduce the efficiency accordingly.

4.2 Experimental Evaluation

We apply the above mechanisms in coordinating different numbers of vehicles to model realistic traffic situations. Parameter settings are the same as in Sec. 3.2.3, except that now more than five vehicles are randomly generated at the beginning. Due to local interactions and decentralized learning among vehicles, our coordinated learning approach can have good scalability, generating quite similar result patterns with regard to the number of vehicles. The result of 10 vehicles in Figure 5 shows that the proposed mechanisms can extend the basic models nicely to more complex situations, resulting in much higher average rewards (i.e., safer policies) than the other approaches. It is clear that the concurrent coordination mechanism is the most effective approach, due to properly resolving the conflicts between different subgroups, while the global coordination mechanism is less efficient due to lack of conflict detection and elimination in the sub-groups. Microscopic evaluation after convergence (Table 2) indicates that the coordinated learning approaches can generally learn

Table 2: Results of microscopic evaluation

Mechanisms	Approaches	Ave. Speed	No. Lane Change	Min. Distance
N/A	*Individual*	31.41	1.85	8.75
	Expert	29.24	1.01	111.66
	MOBIL	28.74	31.89	21.55
Sequential	*I-DCG*	34.25	22.41	7.35
	P-DCG	32.86	56.21	16.54
Concurrent	*I-DCG*	33.62	1.31	11.82
	P-DCG	35.39	1.08	15.54
Global	*I-DCG*	33.05	10.28	6.35
	P-DCG	33.17	35.47	7.73

faster driving policies by properly coordinating their overtaking behaviors. Particularly, the concurrent mechanism enables vehicles to move faster as a whole group by overtaking occasionally, while the other two mechanisms require more overtaking to maintain high speed and thus safer policies as shown by the finial rewards. The diverse microscopic results demonstrate rich patterns of coordinated behaviors learned using the proposed approaches.

5 RELATED WORK

RL has been widely applied in autonomous driving domains. Pyeatt and Howe [23] applied neural network version of Q-learning to learn racing behaviors. Loiacono et al. [15] used tabular Q-learning to learn overtaking strategies. Later, a multiple-goal RL method was proposed to consider a multitude of abilities and criteria for overtaking [18]. More recently, a number of researchers have resorted to *deep RL* for a vehicle's optimal control. Xia et al. [29] proposed deep Q-learning with filtered experiences for autonomous vehicles. Sharifzadeh et al. [26] applied an *inverse RL* approach to extract the rewards in autonomous driving. Several studies used deep RL to realize motion planning at interactions [9, 21]. Other works employed deep RL to realize end-to-end learning of driving policies in partially observable scenarios [24] or from raw visual inputs [2]. All these studies only focus on single vehicle's strategic decision making and do not consider interactions among multiple vehicles and their coordination.

Coordination among vehicles under CAV environments has received an increasing attention for its potential benefits [16]. Moriarty and Langley [17] used RL to solve the intelligent lane selection problem for a group of vehicles. Pendrith [22] presented a distributed *Q-learning* to lane change advisory system. A function approximation RL technique was used for the secure longitudinal following of a front vehicle [4]. Kalantari et al. [10] introduced a novel approach to distributed autonomous navigation through traffic intersections based on a distributed RL framework. Shalev-Shwartz et al. [25] applied deep RL approach to the problem of forming long term driving strategies in multiple vehicles. All these studies, however, did not model interaction dependencies among vehicles. Since no explicit coordination mechanisms have been applied, inefficient and sometimes fatal uncoordinated outcomes may occur [10]. Although some rule-based or utility-based approaches have also been proposed for strategic driving decision making [1, 19, 20], they have limits in terms of time-consuming parameter tuning and tractability difficulties. Moreover, none of these studies considered interactions and concurrent decision making among a group of vehicles.

6 CONCLUSIONS

Applying RL in coordinating multiple autonomous vehicles is a tough problem because of the rule-based nature of autonomous driving as well as the everlastingly changing topology of the moving vehicles. This paper makes an initial contribution in successfully employing CG-based MARL approaches in coordinating overtaking maneuvers for a group of vehicles. Simulations verified that the proposed coordination approaches can guarantee higher levels of safety by properly coordinating vehicles' overtaking behaviors.

Our work contributes to the literature from different technical perspectives. We proposed 1) a dynamic CG formulation of a distributed coordination problem with continuously changing dependencies, 2) a distributed MARL approach in dynamic and open environments, and 3) a strategic learning solution for coordinating multiple vehicles. Each of these advances is nontrivial considering the inherent complexity in the original problem. For example, most research in MARL still focuses on static and close environments [12, 13, 31, 32], in which agent's roles, relationships and tasks are fixed beforehand and kept unchanged during learning. Although learning in such situations has already posed significant challenges due to the concurrent learning and co-adaptation of multiple agents, open and dynamic environments can cause extra burdens on the agents. Our work has demonstrated that, by carefully manipulating the dynamics using the *consistency* principle, effective coordination can be achieved simply via local and distributed learning among agents in a dynamic environment, as in the vehicular environment in this paper.

In this paper, we assume the homogeneity of all the vehicles and their willingness to cooperate with each other. This assumption of full cooperativeness of vehicles might be a bit strong at current stage (considering the long transition period of mixed driving to fully connected autonomous driving), but can be foreseen in the near future traffic scenarios, given the rapid rise of vehicle-to-vehicle communication mechanisms like DSRC. Actually, this assumption has been widely adopted in current research of cooperative driving in platoons or other related studies. Nevertheless, it is still an interesting topic to investigate cooperative driving in mixed situations that involve human drivers or heterogamous vehicles. A potential solution can be including the non-learners or vehicles with fixed behaviors in the built of CG but fixing their behaviors during the VE process in determining the joint optimal actions for the whole group. We leave this issue to our future work.

7 ACKNOWLEDGMENTS

This work was supported by the Joint Key Program of National Natural Science Foundation of China and Liaoning Province under Grant U1808206.

REFERENCES

[1] Noor Cholis Basjaruddin, Kuspriyanto Kuspriyanto, Didin Saefudin, Edi Rakhman, and Adin Mochammad Ramadlan. 2014. Overtaking Assistant System Based on Fuzzy Logic. *Telkomnika (Telecommunication Computing Electronics and Control)* 13, 1 (2014), 76–84.

[2] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[3] Lucian Bu, Robert Babu, Bart De Schutter, et al. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172.

[4] Charles Desjardins and Brahim Chaib-draa. 2011. Cooperative adaptive cruise control: A reinforcement learning approach. *IEEE Transactions on Intelligent Transportation Systems* 12, 4 (2011), 1248–1260.

[5] Daniel J Fagnant and Kara Kockelman. 2015. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice* 77 (2015), 167–181.

[6] Carlos Guestrin, Daphne Koller, and Ronald Parr. 2002. Multiagent planning with factored MDPs. In *NIPS'02*. MIT Press, Cambridge, USA, 1523–1530.

[7] Carlos Guestrin, Michail Lagoudakis, and Ronald Parr. 2002. Coordinated reinforcement learning. In *ICML'02*, Vol. 2. ACM, New York, NY, USA, 227–234.

[8] Berthold KP Horn. 2013. Suppressing traffic flow instabilities. In *IEEE ITS'16*. IEEE, Los Alamitos, CA, 13–20.

[9] David Isele, Akansel Cosgun, Kaushik Subramanian, and Kikuo Fujimura. 2017. Navigating Intersections with Autonomous Vehicles using Deep Reinforcement Learning. *arXiv preprint arXiv:1705.01196* (2017).

[10] Rahi Kalantari, Michael Motro, Joydeep Ghosh, and Chandra Bhat. 2016. A distributed, collective intelligence framework for collision-free navigation through busy intersections. In *IEEE ITS'16*. IEEE, Los Alamitos, CA, 1378–1383.

[11] Arne Kesting, Martin Treiber, and Dirk Helbing. 2007. General lane-changing model MOBIL for car-following models. *Transportation Research Record* 1999, 1 (2007), 86–94.

[12] Jelle R Kok and Nikos Vlassis. 2006. Collaborative multiagent reinforcement learning by payoff propagation. *Journal of Machine Learning Research* 7, Sep (2006), 1789–1828.

[13] Lior Kuyer, Shimon Whiteson, Bram Bakker, and Nikos Vlassis. 2008. Multiagent reinforcement learning for urban traffic control using coordination graphs. *Machine Learning and Knowledge Discovery in Databases* (2008), 656–671.

[14] Xin Li, Xin Xu, and Lei Zuo. 2015. Reinforcement learning based overtaking decision-making for highway autonomous driving. In *ICIP'15*. IEEE, Los Alamitos, CA, 336–342.

[15] D Loiacono, A Prete, P. L Lanzi, and L Cardamone. 2010. Learning to overtake in TORCS using simple reinforcement learning. In *EC'10*. IEEE, Los Alamitos, CA, 1–8.

[16] Ning Lu, Nan Cheng, Ning Zhang, Xuemin Shen, and Jon W Mark. 2014. Connected vehicles: Solutions and challenges. *IEEE Internet of Things Journal* 1, 4 (2014), 289–299.

[17] David E Moriarty and Pat Langley. 1998. Learning cooperative lane selection strategies for highways. In *AAAI/IAAI'98*, Vol. 1998. MIT Press, Cambridge, USA, 684–691.

[18] Daniel Chi Kit Ngai and Nelson Hon Ching Yung. 2011. A multiple-goal reinforcement learning method for complex vehicle overtaking maneuvers. *IEEE Transactions on Intelligent Transportation Systems* 12, 2 (2011), 509–522.

[19] Julia Nilsson, Jonatan Silvlin, Mattias Brannstrom, Erik Coelingh, and Jonas Fredriksson. 2016. If, when, and how to perform lane change maneuvers on highways. *IEEE Intelligent Transportation Systems Magazine* 8, 4 (2016), 68–78.

[20] Julia Nilsson and Jonas Sjöberg. 2013. Strategic decision making for automated driving on two-lane, one way roads using model predictive control. In *IVS'13*. IEEE, Los Alamitos, CA, 1253–1258.

[21] Chris Paxton, Vasumathi Raman, Gregory D Hager, and Marin Kobilarov. 2017. Combining Neural Networks and Tree Search for Task and Motion Planning in Challenging Environments. *arXiv preprint arXiv:1703.07887* (2017).

[22] Mark D Pendrith. 2000. Distributed reinforcement learning for a traffic engineering application. In *ICA'00*. ACM, New York, NY, 404–411.

[23] Larry D. Pyeatt and Adele E. Howe. 1998. Learning to Race: Experiments with a Simulated Race Car. In *AIRSC'98*. IEEE, Los Alamitos, CA, 357–361.

[24] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. 2017. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging* 2017, 19 (2017), 70–76.

[25] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. 2016. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295* (2016).

[26] Sahand Sharifzadeh, Ioannis Chiotellis, Rudolph Triebel, and Daniel Cremers. 2016. Learning to Drive using Inverse Reinforcement Learning and Deep Q-Networks. *arXiv preprint arXiv:1612.03653* (2016).

[27] Marco Wiering and Martijn Van Otterlo. 2012. *Reinforcement learning*. Springer, Berlin, Heidelberg.

[28] Cathy Wu, Aboudy Kreidieh, Eugene Vinitsky, and Alexandre M Bayen. 2017. Emergent Behaviors in Mixed-Autonomy Traffic. In *CoRL'17*. MIT Press, Cambridge, USA, 398–407.

[29] Wei Xia, Huiyun Li, and Baopu Li. 2016. A Control Strategy of Autonomous Vehicles Based on Deep Reinforcement Learning. In *ISCID'16*, Vol. 2. IEEE, Los Alamitos, CA, 198–201.

[30] Yurong You, Xinlei Pan, Ziyan Wang, and Cewu Lu. 2017. Virtual to Real Reinforcement Learning for Autonomous Driving. *arXiv preprint arXiv:1704.03952* (2017).

[31] Chao Yu, Minjie Zhang, Fenghui Ren, and Guozhen Tan. 2015. Emotional multiagent reinforcement learning in spatial social dilemmas. *IEEE Transactions on Neural Networks and Learning Systems* 26, 12 (2015), 3083–3096.

[32] Chao Yu, Minjie Zhang, Fenghui Ren, and Guozhen Tan. 2015. Multiagent learning of coordination in loosely coupled multiagent systems. *IEEE Transactions on Cybernetics* 45, 12 (2015), 2853–2867.