Explicit Dynamic Coordination Reinforcement Learning Based on Utility

Huaiwei Si[1], Guozhen Tan[1*], Yifu Yuan[1], Yanfei peng[1], Jianping Li[1]
[1] School of Computer Science and Technology, Dalian University of Technology, Dalian, LiaoNing,116024,People's Republic of China
[e-mail: sihuaiwei@mail.dlut.edu.cn]
[*]Corresponding author: Guozhen Tan

Received July 30, 2021; revised October 1, 2021; revised February 5, 2022; accepted February 16, 2022; published March 31, 2022

Abstract

Multi-agent systems often need to achieve the goal of learning more effectively for a task through coordination. Although the introduction of deep learning has addressed the state space problems, multi-agent learning remains infeasible because of the joint action spaces. Large-scale joint action spaces can be sparse according to implicit or explicit coordination structure, which can ensure reasonable coordination action through the coordination structure. In general, the multi-agent system is dynamic, which makes the relations among agents and the coordination structure are dynamic. Therefore, the explicit coordination structure can better represent the coordinative relationship among agents and achieve better coordination between agents. Inspired by the maximization of social group utility, we dynamically construct a factor graph as an explicit coordination structure to express the coordinative relationship according to the utility among agents and estimate the joint action values based on the local utility transfer among factor graphs. We present the application of such techniques in the scenario of multiple intelligent vehicle systems, where state space and action space are a problem and have too many interactions among agents. The results on the multiple intelligent vehicle systems demonstrate the efficiency and effectiveness of our proposed methods.

Keywords: Reinforcement Learning, Multi-agent System, Explicit Coordination Learning, Utility Dependence, Intelligent Vehicle

http://doi.org/10.3837/tiis.2022.03.003 **ISSN : 1976-7277**

1. Introduction

Many real-world problems, such as multi-robot and urban traffic control systems, are naturally modeled as multi-agent reinforcement learning problems [1]. A multi-agent system consists of a group of interacting intelligent agents and has been widely used in a variety of domains from industries to militaries. Network packet delivery [2][3], urban traffic control [4], resource management [5], and collaborative decision support system [6] should naturally be modeled as coordination multi-agent systems [7].

Although deep reinforcement learning (DRL) has been applied successfully to single-agent learning [8], which solves the problem of high dimensional state space in reinforcement learning (RL) [9][10]. Unfortunately, in coordination multi-agent systems, tackling such problems with traditional RL or DRL is not straightforward because the coordination multi-agent systems not only need to learn which actions to execute in an unknown environment but also to coordinate their actions in the working environment [11].

Independent Q-learning (IQL) [12] is a popular method in which each agent independently learns its policy, treating other agents as part of the observations or environments. IQL avoids the scalability problems of centralized learning but it introduces a problem: ignoring the interactions among agents, and each agent's learning is confounded by the learning and exploration of others.

At the other extreme, some studies used a fully centralized training method to train coordination multi-agent systems [13]. In fully centralized training methods, a group of agents observes the global states and then models a coordination multi-agent system as a single meta-agent. However, in meta-agent settings, the learning is intractable because of the exponential growth of the problem size with the increasing number of agents. Recently, many multi-agent reinforcement learning methods that use centralized training and distributed execution methods have been introduced, such as counterfactual multi-agent (COMA) [14], multi-agent actor-critic for mixed cooperative-competitive (MADDPG) [15], etc. These methods use a centralized value function network to represent the joint value function of all agents and use it to guide the policy network of each agent (as a baseline). However, if too many agents are in the multi-agent system, training a centralized network for all agents is difficult and may not converge.

Between these two extremes, a value decomposition network (VND) [16] method is applied to learn a factored joint value function. The global value function is used to guide agents to learn decentralized local policy by decomposing the global joint value function into a linear combination of local value functions. The QMIX method improves the linear combination method of VDN and uses a hybrid network to approximate the nonlinear decomposition of each agent [17]. Because the QMIX does not know the relationship between agents, the global value function needs to be approximated by the nonlinear parameters, which leads to an implicit coordination approach. VDN and QMIX restrict the relation representation between agents while the local Q-values (utility of agent action) are estimated only from local observations. Those implicit coordination methods cannot clearly express the coordination relationship between agents (the learning process needs to consider agents that do not have a cooperative relationship), which affects the coordination efficiency, and may not be able to obtain a coordination policy. Moreover, to obtain the optimal policy, agents not only need to express the coordination relationship clearly but also need to obtain the coordination utility.

In this paper, we combine factor graphs with DRL to propose an explicit reinforcement learning method for the coordination learning of a multi-agent system. The factor graph not only expresses the coordination relationship but also represents the joint utility between agents.

In our method, the coordination relationship is expressed explicitly by the factor graph. We use the local value function to obtain the global value function. The local value function is observed by each agent, and the global value function integration is carried out according to the message passing algorithm. For the construction of the factor graph [18][19], we do not want to use rules and pre-defined methods, we want to find a way to adjust our coordination structure dynamically according to the state transition. Inspired by the group social utility theory [20], a coordination multi-agent system can be decomposed and combined according to the relationship between agents dynamically, whose relationship is a type of interactive utility (the payoff of coordination). Our coordination structure differs from previous methods, which is a dynamic construction method based on utility.

The second section introduces the related work of multi-agent reinforcement learning. The third section describes the multi-agent MDP model, factor graph model, and Max-sum algorithm. The fourth section describes the method in detail. In the fifth section, we verify our method in the multi intelligent vehicle environment and compare it with other methods. The sixth section summarizes this paper and the possible future research directions.

2. Related Work

Multi-agent RL has a rich and long history [21] and one of the most commonly used methods is independent Q-learning (IQL) [12][22][23]. The IQL method takes each agent in a multi-agent system as an independent entity to learn completely independent action-value functions. This method can be combined with deep learning and applied to other high-dimensional agent systems. However, this kind of independent learning is difficult to converge and its stability cannot be guaranteed because it does not consider the coordination relationship. Although the studies [24][25]solved part of the stability problem by extending the state space, the convergence is still not guaranteed.

Other studies regard multi-agent as a single meta-agent [26], where the action is a joint action for all agents and the reward is the total reward for all agents. The immediate difficulty with this approach is that the action space is quite large: if there are n agents, each of them can take m actions, then the action space is m^n. On this basis, some studies have adopted centralized training distributed execution mechanisms, but most of these methods use actor-critic architecture [27]. Gupta et al., “proposed to train agents with one actor-critic and only use an actor to make decisions when executing [28]. MADDPG [15] also estimates other agents policies in actor-critic architecture for training. COMA [14] method uses centralized critic to train decentralized actors and uses the counterfactual ideal to solve the credit allocation problem. Although that method reduces exponential complexity, it is still difficult to train and converge for the multi-agent centralized critic.

Recently, many studies have decomposed the centralized global value function into the combination of local functions, such as the value decomposition network (VDN) proposed by Sunehang et al., “[16]. VDN method adds all value functions of agents linearly to guide the training of agents and does not require additional information from other agents. Rashid et al., use a QMIX network to decompose and sum up all agent value functions based on VDN [17]. They further employed a nonlinear combination of each agent value function. However, this improved method requires a monotonic increasing constraint, which is guaranteed by the maximum operation of each agent’s value function. Moreover, it is an implicit expression of the coordinated relationship that limits the relationship representation between agents and affects the accuracy of the agent value function. It does not have the representational capacity to distinguish the values of coordinated and uncoordinated actions, an optimal policy cannot

be learned.

Thus, it is very important to explicitly represent the coordinated relationship and value function. Guestrin et al., "[29] and Kok et al., "[30] used a coordination graph to represent the dependence among agents. According to this dependence, the global value function is decomposed into the sum of local value functions. This method uses the coordination graph in large savings in the state-action representation and combines it with RL to obtain the coordination policy. However, the coordination graph method based on the local relationship decomposition needs to define the dependency relationship among agents in advance, which is inflexible. Although the paper [31] proposed a method to decompose the value function of a dynamic coordination graph being studied, the dynamic decomposition is based only on the relative position relationship or in a specific order to construct the dynamic coordination graph, while the internal utility and graph modeling method based on predefined rules are not considered. The real coordination is determined by the utilities of agents and automatically adjusts the coordination structure according to status.

3. Preliminaries

3.1 Markov Decision Progress and Deep Reinforcement Learning

Markov decision processes (MDP) [33] is an important stochastic decision model and are frequently used in sequential decision-making. The MDP is defined by four-tuples $\{S, A, r, P\}$, where S is a discrete or continuous state space, A is a discrete or continuous action space, and $r: S \times A \to R$ is a reward function. P is a transition probability function.

$$R_T = E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right] \tag{1}$$

R_T is the total expected reward, where E is a mathematic expectation, $\gamma \in [0,1)$ is the discount factor, and r_t is an immediate reward for performing an action in a state at time t produced by the reward function r. Reinforcement learning optimizes the object of the value function or policy to realize the control optimization. Assume $s \in S$ and $a \in A$ at time t. π is a policy at time t, which is a mapping $\pi: S \to A$. The decision objective can be maximized with any initial state. The state-action value (Q-value) function is defined as follows.

$$Q^{\pi}(s,a) = E^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a\right] \tag{2}$$

For any policy π, according to transition probability $p(s,a,s')$, the state action value function (3) satisfy the Bellman equation, and the optimal policy π^* is defined (4):

$$Q^{\pi^*}(s,a) = \sum_{s' \in S} p(s,a,s')\left[r(s,a,s') + \gamma \max_{a' \in A} Q^{\pi^*}(s',a')\right] \tag{3}$$

$$\pi^*(s) = \arg\max_{a \in A} Q^{\pi^*}(s,a) \tag{4}$$

When the state transition probability $p(s,a,s')$ is unknown, a common solution is Q-learning, which is a popular temporal difference learning method. Q-learning estimates each state-action $Q(s,a)$ value function (Q-value) for assignment. It is based on the state and action for learning. Q-learning determines an expected discount on future rewards. The state s takes action a at time t and r is the reward value of the state s at time t. They are all observed, and s' is the next state, a' is the next action. The synchronization of the Q-values is updated

as shown in (5):

$$Q(s,a) := Q(s,a) + \alpha\left[r(s,a) + \gamma \max_{a'} Q\left(s',a'\right) - Q(s,a)\right] \quad (5)$$

where $\alpha \in (0,1)$ is a learning rate. Q-learning converges to an optimal $Q^*(s,a)$ value if all state action pairs have been detected with a reasonable exploration strategy.

Q-learning does not apply to high-dimensional state environments. In Deep Q-learning Network (DQN) [34], the Q-value function can be approximated by a neural network. The Q-value function is represented by $Q(s,a;\theta)$ and parameterized by θ. The DQN has two mechanisms for improving learning performance: experience replay and the target network [35]. The state correlations are weakened by uniformly sampling from the replay memory. The temporal difference loss function can be written as $(y - Q(s,a:\theta))^2$, where $y = Q\left(s',a';\theta'\right) + r$ and $a' = \arg\max_{a \in A} Q\left(s',a|\ \theta'\right)$. θ represents the current Q function parameter, and θ' represents the Q function parameter before the n times, which is updated to θ with a fixed frequency. DQN can use the high-dimensional information as the Q-learning inputs; for example, using pictures as the inputs.

3.2 Multi-agent MDP

In multi-agent systems, the single agent MDP should be extended to the multi-agent system. A tuples $< n, \mathbf{S}, \left\{A^i\right\}_{i=1}^n, T, R, \gamma >$, n is the number of agents, $\mathbf{S}$ is the global states, and A^i is a set of actions which the agent in multi-agent system. $T : \mathbf{S} \times A \times \mathbf{S} \to [0,1]$ is the transition function that describes the multi-agent system from $\mathbf{S}$ to $\mathbf{S}'$. The reward function is R. Each agent i has a policy π^i conditioned on its observations $s \in \mathbf{S}'$, and obtains the reward r_t at step t. A joint value function $Q^{\pi}\left(s_t, \mathbf{a}_t\right) = E[G_t \mid s_t, \mathbf{a}_t]$, where $G_t = \sum_{t=0}^{\infty} \gamma^t r_t$ and the $\mathbf{a}_t$ is the joint action and r_t is reward at time step t.

3.3 Factor Graph and Max-Sum Algorithm

The factor graph [36] can represent the relationship between agents in the form of local joint utility, and an explicit representation of agent relationship. The factor graph is usually divided into function and variable nodes, as shown in **Fig. 1(a)**. The definition of specific factor graph is as follows: a global valued function F, which is dependent on a set of variable-nodes $x_1, x_2, \ldots, x_n$ and a set of function-nodes $f_{i,j}, f_{i,j,k}, \ldots, f_{i,j,k,l}, (i,j,k,l) \in \{1,2,\ldots,n\}$. Each function-nodes is a factor, which is a function of a subset x_m of the variables. The factor graph follows the representation of the bipartite graph. Variable nodes are only connected with function nodes, and function nodes are only connected with variable nodes. A factor graph can be constructed in two ways: one is based on interaction and the other is based on utility value. We construct a factor graph according to the utility construction rule. Each agent i $(A(i))$ maintains the joint utility function and its action variables. When an agent interacts with another agent, the current agent takes its action as the variable of the utility function, as shown in **Fig. 1 (a)**.

With the representation of the agent relationship in the factor graph, we use the Max-Sum algorithm based on social utility to solve the global maximum utility value. The Max-Sum

algorithm [37] is a set of message propagation algorithms used to solve the social benefit problem in a group of interactive individuals. The Max-Sum algorithm iteratively calculates the value of the message passing between the variable and function nodes until the value reaches stability or the iterative calculation reaches a certain round. To facilitate the description of this message propagation, we use a simplified factor graph (**Fig. 1(b)**) to illustrate the message passing process, the x_1, x_2, x_3 represents the variable nodes, and f_1, f_2 represents the function nodes. The information calculation rules are as follows:

(a) Information transfer rules from variable node i to function node j

$$q_{i\to j}(x_i) = \sum_{k\in M_i\setminus j} r_{k\to i}(x_i) \tag{6}$$

where $M_i\setminus j$ is the index of all function nodes connected to the current variable, except for itself.

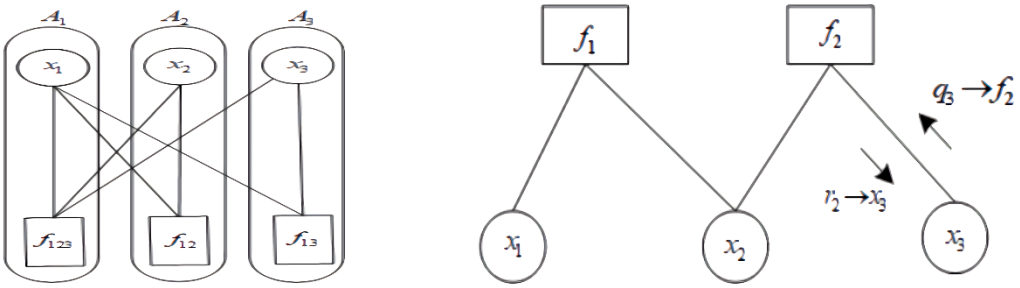
(b) The information transfer rules from function node j to variable node i

$$r_{j\to i}(x_i) = \max_{\mathbf{x}_j\setminus i}\left[f_j(x_j) + \sum_{k\in N_j\setminus i} q_{k\to j}(x_k)\right] \tag{7}$$

where $N_j\setminus i$ represents the index of all variable nodes connected to function node j, except for itself. The message sent and received by the variable nodes in the factor graph is the maximum utility of the population when these variables are assigned different values. At any time during the propagation of these messages, the i th agent can determine which value it should adopt to maximize the sum of all agents utility. Each variable node calculates the local utility function value $z_i(x_i)$. Through $\arg\max_{x_i} z_i(x_i)$ operation to find the value of the variable-node with the maximum global utility. Each variable node calculates the local utility function value $z_i(x_i)$;

$$z_i(x_i) = \sum_{j\in M_i} r_{j\to i}(x_i) \tag{8}$$

Through $\arg\max_{x_i} z_i(x_i)$ operation to find the value of the variable-node with the maximum global utility.



(a) Factor graph (b) Max-Sum algorithm

Fig. 1. Factor graph and Max-sum algorithm

3.4 Intelligent Vehicle RL and MDP Model

A behavior planning decision, such as a car-following or overtaking decision, of an intelligent vehicle can be modelled as MDP $< S, A, R >$. Here, vehicle 0 is used to explain the state of each intelligent vehicle, as illustrated in **Fig. 2**. The state includes all of the factors affecting the decision making of the intelligent vehicles. The ten dimensional states entailed in

(l, v_0, v_i, d_i) $(i = 1, 2, 3, 4)$ are shown in **Fig. 2**.

Here, l is the lane where the vehicle is located. Subsequently, $l = 1$ means that the vehicle is driving on the driving lane, and $l = 2$ means that the vehicle is driving on the overtaking lane.

(1) v_0 is the speed of the vehicle.

(2) v_1 is the speed of the leading vehicle that is closest to the vehicle on the driving lane.

(3) v_2 is the speed of the lagging vehicle that is closest to the vehicle on the driving lane.

(4) v_3 is the speed of the leading vehicle that is closest to the vehicle on the overtaking lane.

(5) v_4 is the speed of the lagging vehicle that is closest to the vehicle on the overtaking lane.

(6) d_1 is the distance between the leading vehicle and the vehicle on the driving lane that is closest to the vehicle.

(7) d_2 is the distance between the lagging vehicle and the vehicle on the driving lane that is closest to the vehicle.

(8) d_3 is the distance between the leading vehicle and the vehicle on the overtaking lane that is closest to the vehicle.

(9) d_4 is the distance between the lagging vehicle and the vehicle on the overtaking lane that is closest to the vehicle.

The abovementioned state definition regards the position and speed information of neighboring vehicles as the two dimensions of the state. In fact, the position and speed of the neighboring vehicle can be combined with the remaining reaction time (RRT) to reduce the redundancy of information, thereby further reducing the dimension of the state. As such, the dimension of the autonomous vehicle's state is reduced from ten to five dimensions. After the dimensionality reduction process, the state of the vehicle can be expressed as (l, t_i) $(i = 1, 2, 3, 4)$.

(1) $t_1 : t_1 = (d_1 - d_{s_1}) / v_0$ indicates the remaining reaction time between the vehicle and the leading vehicle on the driving lane.

(2) $t_2 : t_2 = (d_2 - d_{s_2}) / v_2$ indicates the remaining reaction time between the vehicle and the lagging vehicle on the driving lane.

(3) $t_3 : t_3 = (d_3 - d_{s_3}) / v_0$ indicates the remaining reaction time between the vehicle and the leading vehicle on the overtaking lane.

(4) $t_4 : t_4 = (d_4 - d_{s_4}) / v_4$ indicates the remaining reaction time between the vehicle and the lagging vehicle on the overtaking lane.

In the formulas, d_{s_i} means the shortest safe distance between two vehicles when the current vehicle is driving at an acceleration of $-6m/s^2$, and the following vehicle is driving at an acceleration of $-4m/s^2$. For example, the minimum safe distance between a vehicle and the leading vehicle on the driving lane is $d_{s2} = v_0^2 / (2 * a_0) - v_2^2 / (2 * a_2) + 10$, where $a_0 = -6m/s^2$, $a_1 = -4m/s^2$, and 10 is the predetermined distance that considers the reaction time of the driver of the leading vehicle and the length of the vehicle. As this study focuses on high-level driving decisions during vehicle driving, the action set mainly includes the following two actions:

(1) Driving on the driving lane: The vehicle follows the leading vehicle. For the planned speed v_p on the driving lane, the speed limit on this driving lane is $v_d = 35m/s$.

(2) Driving on the overtaking lane: The vehicle follows the leading vehicle. For the planned speed v_p on the overtaking lane, the speed limit on this overtaking lane is $v_o = 40m/s$.

The driving planned speed v_p means that the vehicle follows the leading vehicle at the planned speed v_p. When the planned speed v_p is greater than the speed limit (v_d or v_0) of the lane, the vehicle follows the leading vehicle at a certain speed (v_d or v_0) where it is driving on. The planned speed of a vehicle can be calculated using various car-following models [37] [38] based on the relative position and relative speed of the vehicle in front. We use the heuristic method to implement the car-following model and calculate the planned speed v_p as follows: $v_p : ((d_1 - d_f > 10m) \& (v_1 - v_0) > 3.6km/h) \Rightarrow v_p = (v_0 + 0.25(d_1 - d_f) + 1.5(v_1 - v_0))km/h$

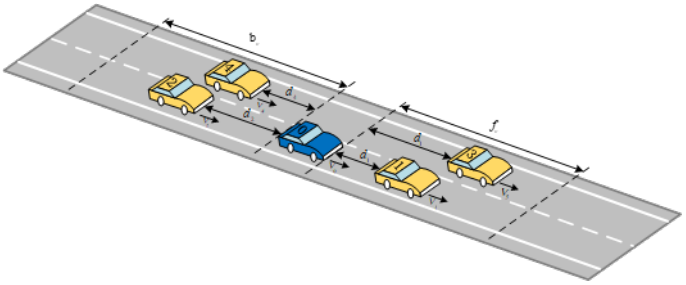


Fig. 2. The state of vehicle

where $d_f = (v_0^2/(-2a_1) + 5)m$ is the vehicle following distance, a=-6m/s^2 is the preset braking acceleration of vehicle deceleration, and 5m is the length of the vehicle. In this heuristic method, if the distance between vehicles is large $((d_1 - d_f) > 10m)$, and the relative speed between vehicles is relatively large $((v_1 - v_0) > 3.6km/h)$ at the same time, then the lagging vehicle will accelerate to reduce the distance from the leading vehicle.

The last important element of MDP is the reward function, which is an index used to evaluate the learning performance of an agent. In autonomous driving, safety is the ultimate goal and the most important criterion for evaluating the decision making of an agent. By considering the safety of driving, the definition of the reward function is given by

$$r_{\text{safe}} = \begin{cases} -40 & if\ \ collision \\ \min(t_1, t_2) & if\ \ l = 1\ and\ d_1 > 3\ and\ d_2 > 3 \\ \min(t_3, t_4) & if\ \ l = 2\ \ and\ d_3 > 3\ and\ d_4 > 3 \\ -5 & else \end{cases} \tag{9}$$

$$r_{\text{speed}} = \begin{cases} v_p - v_t & if\ \ v_p > v_t \\ 0 & else \end{cases} \tag{10}$$

where v_t is the task speed of the last decision. When the vehicle runs on the driving lane, and the distance between the vehicles (leading vehicle and lagging vehicle) is not too close $(d_1, d_2 > 3\text{m})$, the reward value is the minimum of the remaining reaction time on the driving lane $(\min(t_1, t_2))$. Thus, the greater are the values of t_1 and t_2, the greater is the safety reward of the vehicle on the driving lane. When the vehicle runs on the overtaking lane, and the distance between the vehicles (leading vehicle and lagging vehicle) is not too close $(d_3, d_4 > 3\text{m})$, the reward value is the minimum of the remaining reaction time on the overtaking lane. Given $(\min(t_3, t_4))$, the larger are the values of t_3 and t_4, the greater is the reward to be obtained by

the vehicle on the overtaking lane. If the distance between the vehicles and any vehicle in the same lane is less than 3m, then the vehicle will be apprehended; that is, the reward value is −5.

In this paper, the action we designed in the autonomous driving simulation is a discrete driving policy (behavior planning), not continuous action, for example, the direction and speed. But these continuous actions belong to the control level, and our actions only involve the planning level. These driving policies, such as overtaking and lane changing, are designed as discrete actions. According to [41] and [42], the tasks of an intelligent vehicle can be divided into three categories, namely, perception, planning, and control. Furthermore, according to [43] and [44], the planning can be divided into three main levels of route planning, behavior planning, and motion planning. Behavior planning describes the tactical behavior of the vehicle at maneuver layer. The tactical decisions in such scenarios include lane keeping or lane changing to the left or to the right. Many other tactical behavior, such as lane merging policy on intersections or ramps. This paper mainly investigates the lane keeping and lane changing on highway.

4. Method

The coordination structure has been applied to multi-agent systems to coordinate actions among agents. According to the coordination structure, the global value function can decompose into a combination of local value functions. We use an explicit coordination structure to decompose the value function of each agent and the local joint utility value function. In this paper, the explicit coordination structure is the factor graph, which is a representative structure of the joint utility. When an agent is added to the factor graph, each agent has its utility, and at the same time, it will produce local joint utility with the coordinative agent. Specifically, if two agents have interactive utility, then the action sets of the two agents are variable nodes; these variable pairs are regarded as the set E of an edge in the factor graph and the local joint utility of the two agents in the function node. The global value function is (11), which maximizes the global value function composed of each value function and joint utility value function to achieve the purpose of coordination learning.

$$Q(s,a) := \sum_{i=1}^{n} Q_i\left(s_i, a_i\right) + \sum_{(i,j)\in E} Q_{ij}\left(s_{ij}, a_{ij}\right) \tag{11}$$

In formula (11), Q_i represents the value function of each agent in the factor graph and Q_{ij} represents the local joint utility function (payoff function) of two agents with common factors. In previous works, the coordination structure is pre-defined or determined by a simple location relationship, which are then used to message the propagation algorithm to compute global utilities. In this paper, the coordination structure is represented by a factor graph and need not pre-defined or simply construct according to the relative position. We use the utility between agents to construct a factor graph to represent the relationship between agents dynamically. When the joint utility value of agent A_1 and A_2 differs from the sum of independent decisions, it is necessary to establish a coordination relationship in utility theory. A common factor node is established, and the value function of each agent is a function node.

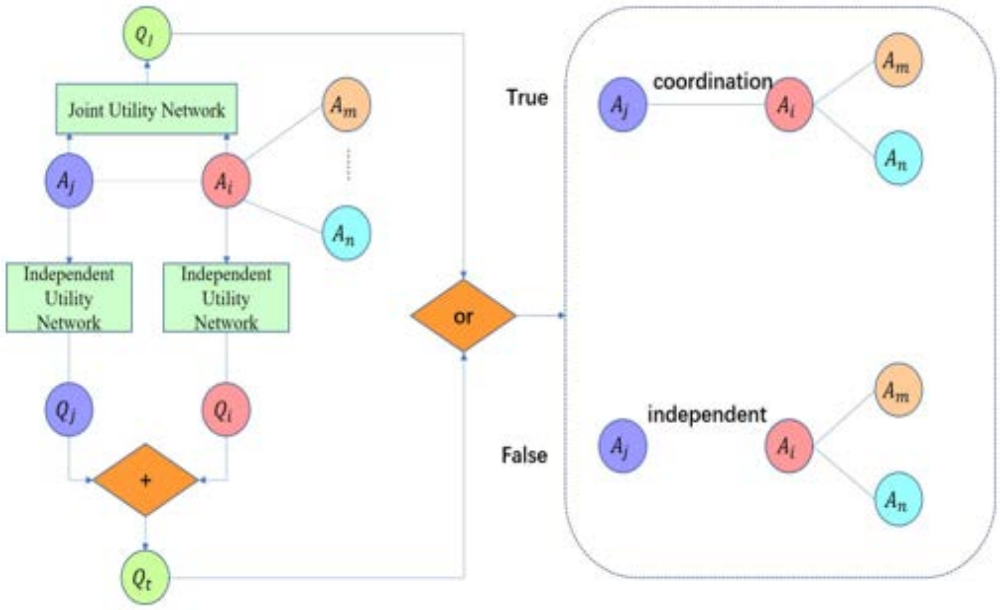


Fig. 3. Construction of dynamic coordination structure based on utility

The basic principle of the dynamic construction factor graph is as follows: According to the difference utility between two agents in a coordination and non-coordination state, we judge whether the two agents construct in the factor graph. We use the significance test method to test the difference between the utility of the agent in the coordinative and the non-coordinative state. A significant difference is observed, the joint action function of the two agents is taken as the common factor of the factor graph. To judge whether an independent agent needs to coordinate with a set of agents which are already in the factor graph, two kinds of utility need to be compared: one is the utility generated by the agent independently from the agents which are in the factor graph and the other is the utility generated by the independent agent added to the factor graph. To simply the calculation further, we can use a utility measure to check the joint utility of two agents, one is independent and the other is already in the factor graph (coordination group). We check the joint utility between each independent agent and each agent that is the coordination group. If the coordination utility is significant, the coordination structure should include the independent agent. This method only checks each couple (local) agent of the coordination utility with statistical significance instead of computing every agent coordination utility using the coordination structure.

The local utility measure can use any multi-agent reinforcement learning methods. The only requirement is that the joint utility of two agents can be calculated. The method we use is a type of joint training method that takes multi-agent as a single meta-agent. Although the single meta-agent method increases exponentially with the number of agents, the amount of computations is acceptable when the number of agents is small. We use a joint utility network to measure the two agents joint utility, which is similar rules. The joint utility network that can pre-train by deep reinforcement learning is used to decide on the agent and whether to add to the factor graph and coordination learning.

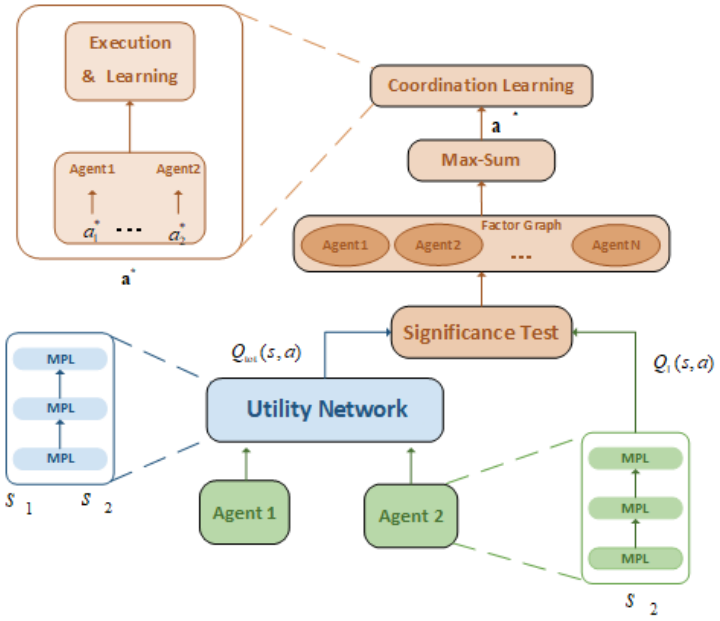


Fig. 4. Utility dependence coordination learning

If an agent is independent of other agents, the agent uses the independent network to learn its strategy and select action. However, if the agent has an effect on global utility, then the agent should enter the coordination structure for coordination learning. Sometimes, the effect should be ignored when the agent only has a greater effect on global utility accidentally, which is also called non-significant in statistics. We used the statistics of multi-agent training samples. These samples are the agent independent learning samples Q_t, $Q_t = Q_i + Q_j$, where Q_j is the utility of independent agent A_j, we check whether to add the coordination group, A_i is the agent already in the coordination group, and utility Q_i is predicted by the independent value network. The joint actions of learning samples Q_J is produced by the utility network (single meta agent learning method), which is the agent A_i and the agent A_j states as the utility network inputs. In these samples, we used uniform sampling in memories to sampling N samples. **Fig. 3** shows that for each couple A_i and A_j, we independently compute the total utility of Q_t on A_i and A_j compute the mean $\bar{Q}_t$. These statistics are only computed by the independent networks, and the agents only used the memories to estimate the action value instead of executing the actions. We also need to compute the variances $\hat{Q}_t$. Meanwhile, the utility network computes the mean utility $\bar{Q}_J$ and variances $\hat{Q}_J$ of the two agents joint action for N samples in the same period. In our method, we use the t−test to detect significant differences between the two utility values. The t-test definition is:

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{\sum x_1^2 + \sum x_2^2}{n_2 + n_2 - 2} \times \frac{n_1 + n_2}{n_1 \times n_2}}} \tag{12}$$

In our method, we use the Q_t and Q_J to compute the t :

$$t = \frac{\bar{Q}_t - \bar{Q}_J}{\sqrt{(2/N)(N-1)\hat{Q}_t + (N-1)\hat{Q}_J / (2N-2)}} \tag{13}$$

In the t-test, the $(2N-2)$ refers to the freedom. We assume that the significance probability $p = 0.05$, the p to prove when the agent A_i joins the group the utility value changing is significant. As with other statistical methods, we use an additional statistical effect size measurement d to determine whether the signature is statistically. The d definition is as follows:

$$d = \frac{\bar{Q}_t - \bar{Q}_J}{r_{\max} - r_{\min}} \tag{14}$$

The d use the max reward ($r_{\max}$) and ($r_{\min}$), max joint utility value, and the mean utility value to prove the statistical data are not only significant but also sufficiently enough. If the p is significantly and d is sufficiently large ($p < P_{t-testtable(N)}, d > D_{t-testtable(N)}$, $P_{t-testtable(N)}$ and $D_{t-testtable(N)}$ are all derived from the t-bound table and are hypothesis test constant), which proves that when the agent A_i joint the coordination group is beneficial. If the $p > P_{t-\text{testtable(N)}}$ proves the A_i in coordination group does not have more utility, and if $d < D_{t-\text{testtable(N)}}$ proves that it is occasional, it can obtain more utility. If the agent joins the coordination group, it should coordinate learning with other agents in the group. In this paper, we use the factor graph to represent the coordination group. Pseudocode is shown in algorithm 1. When we obtain the factor graph, we use coordination learning to training the agents.

Because our computation is based on the samples in memory, an agent that is independent of the coordination group can compute in a completely and synchronization manner with the coordination group to determine whether it needs to join the coordination group. When an agent joins the coordination group, a factor node joins in the graph and an edge also joins in the graph. In the factor graph, we integrate all local utility to obtain the global utility using a message-passing algorithm combined with reinforcement learning to run coordination learning. We construct a factor graph based on utility and decompose the global value function according to formula (11). The Max-Sum algorithm is used to solve formula (15), which obtains maximum joint action a^* and the maximum global utility(as a baseline) to guide each agent leaning. **Fig. 4** describes the whole process of the algorithm, and we called it the utility dependence coordination learning (UDCL)

$$a^* = \arg\left\{\sum_{i=1}^{n} Q_i(s_i, a_i) + \sum_{(i,j)\in E} Q_{ij}(s_{ij}, a_{ij})\right\} \tag{15}$$

Then use actions a^* to guide each agent learning in formula (16), as follows and pseudocode is shown in algorithm 2.

$$Q_i(s,a) := Q_i(s,a) + \alpha\left[R_i(s,a) + \gamma Q\left(s', a^*\right) - Q_i(s,a)\right] \tag{16}$$

Algorithm 1 Utility dependence algorithm

Initialise Coordination Group (CG) CG = $\varnothing$, Independent Group (IG) IG = {Numbs agents}
for t=1 to T_0 **do**
  for each training episode e **do**
  Random select two agents A_1 and A_2

Utility network(A_1 , A_2)
end for
end for
save the utility network //check utility network
select one agent A
for t=1 to T1 **do**
for each training episode e **do**
single − meta(A)
end for
end for
save the single-meta network
while $s_t \neq$ terminal and $t < T$ **do**
t=t+1
for t=1 to T_{total} **do**
Random select two agents from group IG ,A_I,A_C
$A_C \to CG$
$Q_I = Q_t^I + Q_t^C$ //use the independent network
$Q_J =$ Utility network$\left(A_t^I, A_t^C\right)$
end for
if significance test(Q_I,Q_J)==true
$A_I \to CG$
if significance test $\left(Q_I, Q_J\right)$!=true
The A not into CG
end while

Algorithm 2 Utility Dependence Coordination Learning

Initialise network Q_i for each agent i
for each training episode e **do**
for every agent **do**
build factor graph // Algorithm 1
Decompose the global value function according to formula 9
while if $U_{ij}(a_i)$ small change or 15 loops do
for the agent i of all neighbors $j \in \Gamma(i)$ **do**
send j message $U_{ij}(a_i) = \mathrm{Max}\left\{Q_i(a_i) + Q_{ij}(a_i, a_j) + \sum_{k \in \Gamma_{(i)/j}} U_{ki}(a_i)\right\}$
$g(a_i) = \max\left\{Q_i(a_i) + \sum_{j \in \Gamma} U_{ji}(a_i)\right\}$ and $a^* = \arg\left(g(a_i)\right)$
end for
end while
update the according to formula 14
end for
end for

5. Experimental Evaluation

With the development of artificial intelligence (AI) and more advanced hardware technology, traffic systems have been transforming from human to automobile driving. Intelligent vehicles as wheeled robots are applied in intelligent transportation systems (ITS) [39]. The intelligent vehicles will perhaps become the first widely used agents in our lifetimes. In recent years, intelligent vehicles have been launched in many research institutions, and the number of intelligent vehicles on the road will increase in the future. Therefore, more advanced coordinated decision-making is required for driving policies [40].

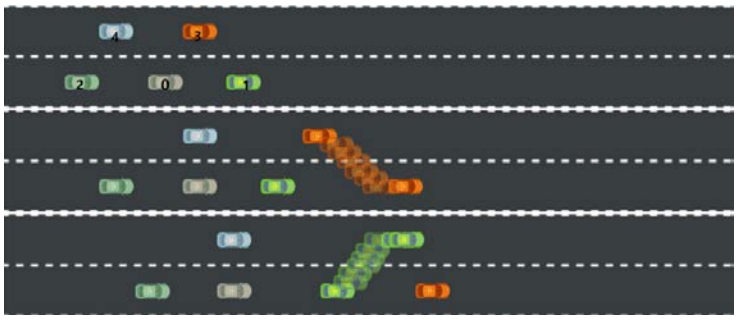


Fig. 5. Traffic flow figure of freeways

We present an MDP model of multi-intelligent vehicle autonomous driving decisions. The MDP model is solved according to the coordination reinforcement learning method through a simulation experiment that evaluates the intelligent vehicle coordination. In this paper, we simulate highway scenarios in **Fig. 5**. Environmental information perceived by each intelligent vehicle is shown in **Fig. 2**. The Section 3 introduces the MDP model of intelligent vehicles. The highway task speed is randomly generated in the [25,40]m/s and the max velocity is 40m/s. At the same time, we simulate the high-level decision-making actions of intelligent vehicles like humans, which are vehicle following and lane changing. When the vehicle decides to overtake, it will change lanes smoothly in advanced design. When it decides to follow, it needs to compute a planning velocity v_{plan}, which is a vehicle following speed and the calculation method as shown in paper [37]. The specific experimental parameters are defined in **Table 1**.

Table 1. Training time and movement parameters of 5-11 vehicles

vehicles	method	Lane Change	Speed	Min. Distance	Times(hour)
5 vehicles	UDCL	12.3	35.1	97.8	4
	Independent	51.8	32.5	30.2	2
	QMIX	17.6	33.5	88.9	8
	COMA	32.5	29.7	105.6	13
	IDCG	33.2	34.1	80.78	23
8 vehicles	UDCL	17.4	33.8	100.2	6
	Independent	60.7	30.3	34.7	3
	QMIX	20.2	34.4	90.2	11
	COMA	22.8	22.8	110.1	22
	IDCG	38.6	38.6	79.8	43
11 vehicles	UDCL	21.3	21.3	103.2	9
	Independent	66.5	28.5	41.2	6
	QMIX	25.9	25.9	90.3	18
	COMA	20.1	20.1	78.9	29
	IDCG	17.9	17.9	87.8	50

According to the method of the fourth part, we first train a joint utility network to judge the coordination relationship between vehicles, and then build the factor graph according to the relationship. Finally, the coordination learning method (UDCL) is used to train multi-intelligent vehicles system. We verify that dynamic local coordination utility is effective compared to the fixed coordination structure and independent learning methods. We choose two vehicles to learn jointly and independently in a multi-vehicle environment. **Fig. 6** shows the learning results of three cases, in which the horizontal axis represents each episode and the vertical axis represents the average cumulative reward value of each episode. Dynamic joint training represents two vehicles of cumulative reward in our dynamic utility method. Fixed joint training represents two vehicles of cumulative reward in a predefined method and independent training represents the reward convergence of independent learning of two vehicles. A comparison of the three reward curves shows that the dynamic utility method of joint learning is more effective.

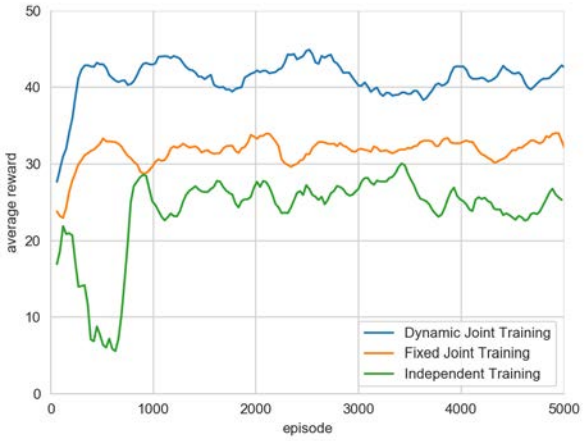
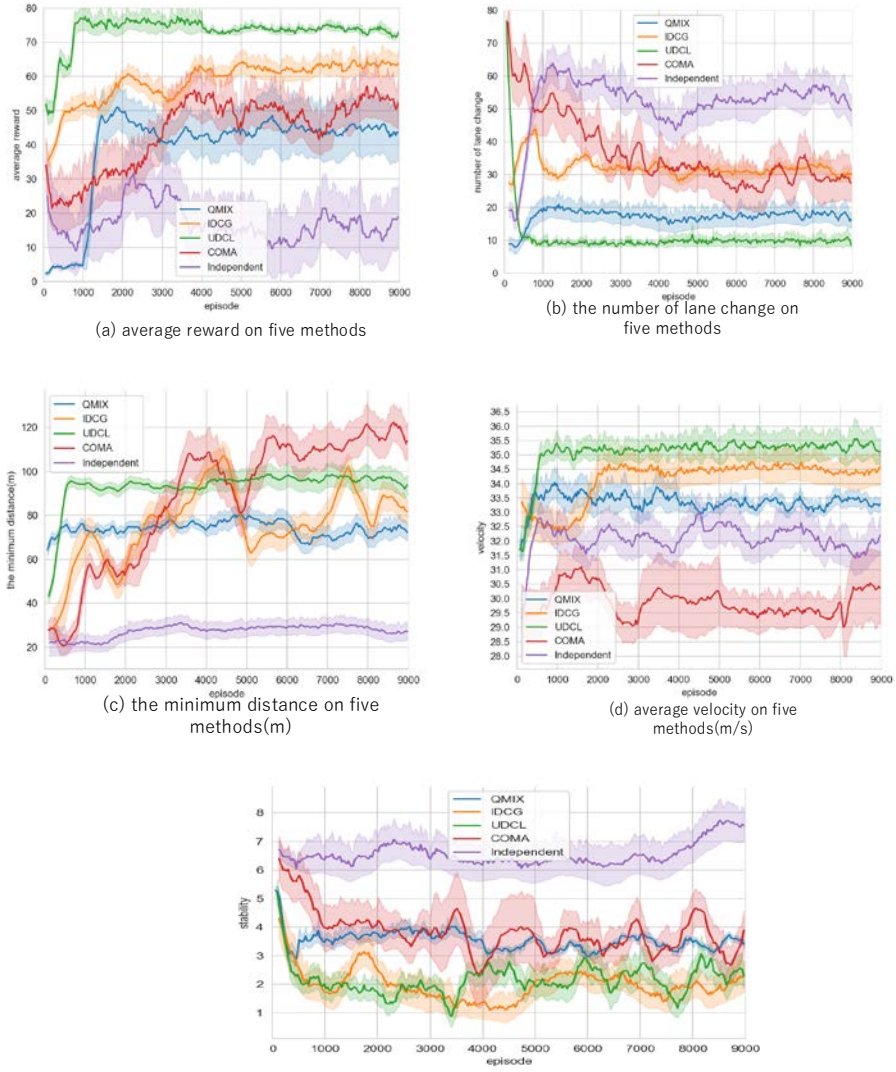


Fig. 6. Three local coordinative construction methods

In **Fig. 7**, we analyze the situation of five intelligent vehicles in the simulated environment. The five vehicles on the highway use our method (UDCL), centralized training coordination method (COMA), implicit coordination learning (QMIX), artificial fixed coordination structure learning method (I-DCG), and independent learning method. **Fig. 7(a)** shows the average cumulative reward curves in the training and five vehicles were used in the simulation environment. In the training, every episode has 400 decision-making simulations. We compare the average cumulative reward for different methods. In our UDCL method, the significance parameter is $P = 0.05$ and the effect size is $D = 0.01$. **Fig. 7(a)** shows that the coordination method (UDCL, I-DCG, QMIX, and COMA) converges faster than the independent method and more stable. We find that an explicit coordinated method (UDCL, I-DCG) obtains more cumulative reward than an implicit collaborative method (QMIX). At the same time, our UDCL method is significantly better than other coordination learning methods (I-DCG) in learning convergence speed and overall reward. Although the I-DCG method also considers the explicit coordinative structure because the method cooperates with all the surrounding intelligent vehicles according to the identification, this method will coordinate with the vehicle which unnecessarily coordinated, such that reduces the final global coordination.



(a) average reward on five methods

(b) the number of lane change on five methods

(c) the minimum distance on five methods(m)

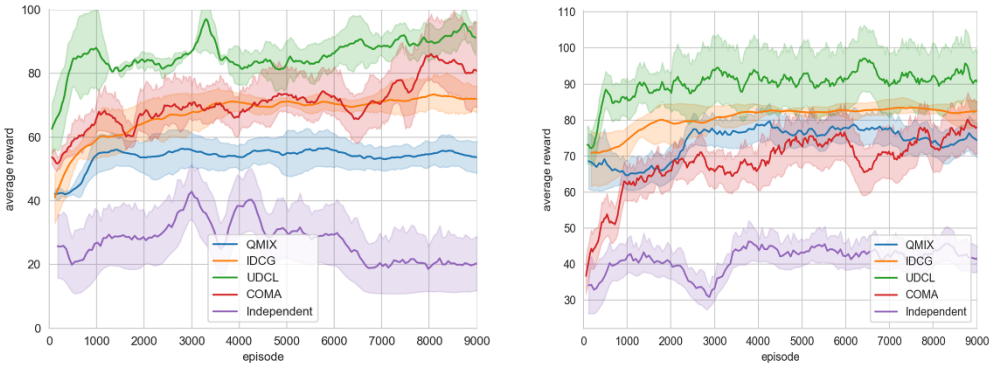
(d) average velocity on five methods(m/s)

(e) stability on five methods

Fig. 7. The average reward, number of lane change, minimum distance, average velocity and stability of intelligent vehicles on the five learning methods

When we compare the micro-movement indicators, we find that the policies learned by coordinative learning are safer and more conservative than the independent method. The independent method will lead to the learned policies being very radical, resulting in frequent lane changing (**Fig. 7(b)**), and keeping a relatively dangerous distance from other vehicles (**Fig. 7(c)**). A large driving velocity can be obtained when considering another vehicle in training (**Fig. 7(d)**). The coordination method can not only obtain higher driving velocity, the velocity stability is also better because of learning a better driving policy (**Fig. 7(e)**). Compared with other coordination methods, our UDCL coordination method maintains a relatively stable minimum distance between vehicles and fewer lane changing times, while the average velocity is guaranteed and stable in a small interval.

We increase the number of experimental vehicles from five to eight and eleven (in **Fig. 8**) and find that the implicit coordination learning method converges more slowly in the learning process. The fully centralized COMA method is also unstable when it trains in the more vehicles environment. At the same time, the implicit coordination learning method converges much more slowly in the learning process and the fully centralized COMA method is unstable in an environment with more vehicles. However, explicit collaboration is always better than QMIX and COMA, which highlights the necessity of explicit coordination. Although I-DCG also maintains a high and stable reward than QMIX and COMA, I-DCG uses the relative position to learn driving policy, the actual utility between vehicles is not considered, which creates more unnecessary coordination. As a result, the convergence of the cumulative reward becomes slower and better coordination of the policy is not obtained.



(a) comparison of the rewards of five learning methods in the case of eight vehicles

(b) comparison of the rewards of five learning methods in the case of eleven vehicles

Fig. 8. Comparison of the rewards of different learning methods in the case of eight vehicles and eleven vehicles

Table 1 shows the training times and movement parameters of 5-11 vehicles for the five algorithms. In the training, we increase the number of vehicles from five to eleven. The training time of each method and the changes in the movement parameters are shown in **Table 1**. The training time for our UDCL method is less than the other implicit coordination learning methods (QMIX), with greater rewards. Because our coordination learning requires vehicles to test the utility among vehicles, more time is required than independent learning. The training time is shorter than all other methods because of the independent learning method does not need to calculate the global utility. With the increase in the number of vehicles, our method still maintains better stability in training time, cumulative reward, and various moving, because we use the residual utility between vehicles as the basis for coordination learning.

Compared with implicit coordination learning, our method clearly defines the coordinative relationships of each vehicle, which can reason the global joint utility. We use the joint utility as a baseline to guide coordination learning. At the same time, compared with the explicit coordination method that builds the coordination relation among vehicles in human-defined rules or order, too much unnecessary coordination will occur, which affects efficiency and effectiveness.

6. Conclusions

In this work, we developed a set of techniques to build the factor graph in multi-agent systems. The factor graph explicitly represents the dependence between agents combined with deep reinforcement learning to train multi-agent systems. However, most efforts focus on the combination of fixed structures, implicit coordination structure, full centralized training, or independent training. The construction of coordination structure affects the efficiency of coordinative computing in the process of coordination. When an agent cooperates with other agents, it is measured by their joint utility, which proves that the agent should cooperate with other agents in the coordination structure. In this paper, inspired by the social utility mechanism, we use the factor graph as the representation structure of the explicit cooperative relationship between agents. We used a joint utility measurement method to measure the joint utility of independent agents within the coordination group and to determine whether independent agents should join the coordination group in combination with the significance test. Our approach focuses on using the dependency among agents to build an explicit coordination structure.

We believe that we have achieved promising results with a lower computation cost. As shown in Table 1, when the training time is less, we consider the computational cost is less. Although our approach requires training the utility network to check two agents utilities, we managed to use DRL in a complex domain using the factor graph that mitigates the problem of large action-space. This approach can be adapted for other cooperative multi-agent systems, where the factor graph is predefined. Our approach can be used to build the factor graph dynamically.

In this paper, simulation experiments are carried out on cooperative learning of multi-agent vehicles. The action of our intelligent vehicle is discrete. In future work, we intend to improve our method in two ways. First, we will study how we can carry out cooperative reinforcement learning of multi-agents in continuous action space. We also plan to explore further the relationship between agents by not only building the coordinative relationship between agents dynamically but also considering this relationship when making decisions. Second, we aim to eliminate the need to predicting the future state-space for each agent in the coordination learning progress. Furthermore, instead of using the free-model RL as the agent learning method, we aim to consider the model-based RL method to agent learning progress that accelerates learning.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant No. U1808206

References

[1] W. Du, S. Ding, “A survey on multi-agent deep reinforcement learning: from the perspective of challenges and applications,” *Artificial Intelligence Review*, 54, 3215-3238, 2021. Article (CrossRef Link)

[2] D. Ye, M. Zhang, Y. Yang, “A multi-agent framework for packet routing in wireless sensor networks,” *sensors*, 15(5), 10026-10047, 2015. Article (CrossRef Link)

[3] N. A. Khalid, Q. Bai, A. Al-Anbuky, “An adaptive agent-based partner selection for routing packet in distributed wireless sensor network,” in *Proc. of IEEE International Conference on Agents*, 2016. Article (CrossRef Link)

[4] Kuyer, L, "Multiagent Reinforcement Learning for Urban Traffic Control Using Coordination Graphs," in *Proc. of European Conference on Machine Learning & Knowledge Discovery in Databases*, Springer, Berlin, Heidelberg, 2008. Article (CrossRef Link)
[5] Stone P, Veloso M, "Multiagent Systems: A Survey from a Machine Learning Perspective," *Autonomous Robots*, vol. 8, pp. 345–383, 2000. Article (CrossRef Link)
[6] Parunak, H. Van Dyke, "Industrial and Practical Applications of DAI," in *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, MIT Press, 2000, pp. 377-421.
[7] P. Kravets, V. Lytvyn, V. Vysotska, Y. Ryshkovets, S. Vyshemyrska, S. Smailova, "Dynamic coordination of strategies for multi-agent systems," in *Proc. of International Scientific Conference Intellectual Systems of Decision Making and Problem of Computational Intelligence*, Springer, pp.653-670, 2020. Article (CrossRef Link)
[8] H. Liu, C. Yu, H. Wu, Z. Duan, G. Yan, "A new hybrid ensemble deep reinforcement learning model for wind speed short term forecasting," *Energy*, 202, 117794, 2020. Article (CrossRef Link)
[9] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. "Playing atari with deep reinforcement learning," *arXiv Preprint arXiv:1312.5602*, 2013. Article (CrossRef Link).
[10] Volodymyr, M., Koray, K., David, S., Rusu, A. A., Joel, V., & Bellemare, M. G., et al., "Human-level control through deep reinforcement learning," *Nature*, 518(7540), 529-533, 2015. Article (CrossRef Link)
[11] W. B¨ohmer, V. Kurin, S. Whiteson, "Deep coordination graphs," in *Proc. of International Conference on Machine Learning*, PMLR, pp. 980-991, 2020. Article (CrossRef Link)
[12] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," *Machine Learning Proceedings*, 330-337, 1993. Article (CrossRef Link)
[13] Sharma P K, Zaroukian E G, Fernandez R, et al., "Survey of recent multi-agent reinforcement learning algorithms utilizing centralized training," *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III*, 2021. Article (CrossRef Link)
[14] Foerster J, Farquhar G, Afouras T, et al., "Counterfactual Multi-Agent Policy Gradients," 2017.
[15] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, I. Mordatch, "Multiagent actor-critic for mixed cooperative-competitive environments," in *Proc. of Advances in neural information processing systems*, pp. 6382-6393, 2017. Article (CrossRef Link)
[16] Sunehag P, Lever G, Gruslys A, et al., "Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward," in *Proc. of AAMAS*, pp. 2085-2087, 2018. Article (CrossRef Link)
[17] Rashid, T., et al., "QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning," in *Proc. of the 35th International Conference on Machine Learning*, pp. 4295-4304, 2018. Article (CrossRef Link)
[18] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, J. Wang, "Mean field multiagent reinforcement learning," in *Proc. of International Conference on Machine Learning*, PMLR, pp. 5571-5580, 2018. Article (CrossRef Link)
[19] Jiang, Jiechuan, and Zongqing Lu, "Learning attentional communication for multi-agent cooperation," in *Proc. of the 32nd International Conference on Neural Information Processing Systems*, pp. 7265-7275, 2018. Article (CrossRef Link)
[20] R. Dechter, "Bucket elimination: A unifying framework for reasoning," *Artificial Intelligence*, 113(1-2), 41-85, 1999. Article (CrossRef Link)
[21] Littman, M. L, "Markov games as a framework for multi-agent reinforcement learning," in *Machine Learning Proceedings 1994*, Morgan Kauffman Publishers, pp. 157-163, 1994. Article (CrossRef Link)
[22] K. Shah, M. Kumar, "Distributed independent reinforcement learning (dirl) approach to resource management in wireless sensor networks," in *Proc. of Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, 2007. Article (CrossRef Link)
[23] Zawadzki, E., A. Lipson, and K. Leyton-Brown, "Empirically evaluating multiagent learning algorithms," *Computer Science*, 2014. Article (CrossRef Link)

[24] Zhang Y, Sun P, Yin Y, et al., "Human-like Autonomous Vehicle Speed Control by Deep Reinforcement Learning with Double Q-Learning," in *Proc. of 2018 IEEE Intelligent Vehicles Symposium (IV)*, 1251-1256, 2018. Article (CrossRef Link)
[25] Foerster J, Nardelli N, Farquhar G, et al., "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proc. of International conference on machine learning*, 1146-1155, 2017. Article (CrossRef Link)
[26] Z. Zhang, "Integrating independent and centralized multi-agent reinforcement learning for traffic signal network optimization," in *Proc. of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2083-2085, 2020. Ph.D. thesis, Georgia Institute of Technology, 2019. Article (CrossRef Link)
[27] Q. Wei, L. Wang, Y. Liu, M. M. Polycarpou, "Optimal elevator group control via deep asynchronous actor-critic learning," *IEEE transactions on neural networks and learning systems*, 31(12), 5245-5256, 2020. Article (CrossRef Link)
[28] Gupta, J. K., M. Egorov, and M. Kochenderfer, "Cooperative Multi-agent Control Using Deep Reinforcement Learning," in *Proc. of International Conference on Autonomous Agents and Multiagent Systems*, Springer, Cham, 2017. Article (CrossRef Link)
[29] C. Guestrin, M. Lagoudakis, R. Parr, "Coordinated reinforcement learning," in *Proc. of ICML*, Vol. 2, Citeseer, pp. 227-234, 2002. Article (CrossRef Link)
[30] Kok J R, Vlassis N., "Collaborative multiagent reinforcement learning by payoff propagation," *Journal of Machine Learning Research*, 7, 1789-1828, 2006. Article (CrossRef Link)
[31] C. Yu, X. Wang, X. Xu, M. Zhang, H. Ge, J. Ren, L. Sun, B. Chen, G. Tan, "Distributed multiagent coordinated learning for autonomous driving in highways based on dynamic coordination graphs," *IEEE Transactions on Intelligent Transportation Systems*, 21(2), 735-748, 2020. Article (CrossRef Link)
[32] D. Koller, N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009. Article (CrossRef Link)
[33] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2014. Article (CrossRef Link)
[34] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, et al., "Deep q-learning from demonstrations," in *Proc. of the AAAI Conference on Artificial Intelligence*, Vol. 32, 2018. Article (CrossRef Link)
[35] Lin M, Zhao B, D Liu, "Policy Gradient Adaptive Critic Designs for Model-Free Optimal Tracking Control With Experience Replay," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 1-12, 2021. Article (CrossRef Link)
[36] D. Huang, J. Lai, C.-D. Wang, "Ensemble clustering using factor graph," *Pattern Recognition*, 50, 131-142, 2016. Article (CrossRef Link)
[37] Farinelli A, Rogers A, Petcu A, et al., "Decentralised coordination of low-power embedded devices using the max-sum algorithm," in *Proc. of International Joint Conference on Autonomous Agents & Multiagent Systems, International Foundation for Autonomous Agents and Multiagent Systems*, 2008. Article (CrossRef Link)
[38] X. Li, X. Xu, L. Zuo, "Reinforcement learning based overtaking decision making for highway autonomous driving," in *Proc. of 2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP)*, IEEE, pp. 336-342, 2015. Article (CrossRef Link)
[39] Zhu, Z., and H. Zhao, "A Survey of Deep RL and IL for Autonomous Driving Policy Learning," 2021. Article (CrossRef Link)
[40] Grigorescu S, Trasnea B, Cocias T, et al., "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, 37(3), 362-386, 2020. Article (CrossRef Link)
[41] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius Brito Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago Meireles Paixão, Filipe Mutz, et al., "Self-driving cars: A survey," *Expert Systems with Applications*, vol. 165, pp. 113816, 2021. Article (CrossRef Link)

[42] Smirnov N, Liu Y, Validi A, et al., "A game theory-based approach for modeling autonomous vehicle behavior in congested, urban lane-changing scenarios," *Sensors*, 21(4),1523, 2021. Article (CrossRef Link)
[43] Meiyu Liu and Jing Shi, "A cellular automata traffic flow model combined with a bp neural network based microscopic lane changing decision model," *Journal of Intelligent Transportation Systems*, 23(4), 309–318, 2019. Article (CrossRef Link)
[44] Zeyu Zhu and Huijing Zhao, "A Survey of Deep RL and IL for Autonomous Driving Policy Learning," *arXiv preprint arXiv:2101.01993*, 2021. Article (CrossRef Link).

Si Huaiwei is a doctoral candidate in School of Computer Science and Technology from Dalian University of Technology, P.R. China. His research interests include group intelligent decision making and intelligent connected autonomous driving.

Guozhen Tan received the Ph.D. degree from the Dalian University of Technology, Dalian, China. He is currently a Professor with the School of Computer Science and Technology, Dalian University of Technology. Since 2012, he has been director of Engineering and Technology Research Center for the Internet of things and Collaborative Sensing, Liaoning province, P. R. China. He is the author of two books, more than 200 articles, and more than 30 inventions. His research interests include group intelligent decision making and intelligent connected autonomous driving. Dr..Tan was a recipient of Second prize winner of National Science and Technology Progress Award.

Yifu Yuan received the master degree in computer science from the Dalian University of Technology, Dalian, China, in 2020. His research interests include group intelligent decision making and intelligent connected autonomous driving.

Yanfei Peng is a doctoral candidate in School of Computer Science and Technology from Dalian University of Technology, P.R. China. His research interests include group intelligent decision making and intelligent connected autonomous driving.

Jianping Li is a doctoral candidate in School of Computer Science and Technology from Dalian University of Technology, P.R. China. His research interests include group intelligent decision making and intelligent connected autonomous driving.