Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2016, Article ID 3207460, 17 pages
http://dx.doi.org/10.1155/2016/3207460

Research Article

Coordinated Learning by Model Difference Identification in Multiagent Systems with Sparse Interactions

Qi Zhang, Peng Jiao, Quanjun Yin, and Lin Sun

College of Information Systems and Management, National University of Defense Technology, Changsha, Hunan, China

Correspondence should be addressed to Quanjun Yin; yin_quanjun@163.com

Received 20 March 2016; Revised 2 September 2016; Accepted 19 September 2016

Academic Editor: Paolo Renna

Multiagent Reinforcement Learning (MARL) is a promising technique for agents learning effective coordinated policy in Multiagent Systems (MASs). In many MASs, interactions between agents are usually sparse, and then a lot of MARL methods were devised for them. These methods divide learning process into independent learning and joint learning in coordinated states to improve traditional joint state-action space learning. However, most of those methods identify coordinated states based on assumptions about domain structure (e.g., dependencies) or agent (e.g., prior individual optimal policy and agent homogeneity). Moreover, situations that current methods cannot deal with still exist. In this paper, a modified approach is proposed to learn where and how to coordinate agents' behaviors in more general MASs with sparse interactions. Our approach introduces sample grouping and a more accurate metric of model difference degree to identify which states of other agents should be considered in coordinated states, without strong additional assumptions. Experimental results show that the proposed approach outperforms its competitors by improving the average agent reward per step and works well in some broader scenarios.

1. Introduction

Multiagent Reinforcement Learning (MARL) provides a promising technique for autonomous agents to solve sequential decision problems in Multiagent Systems (MASs) [1], which has been applied to a variety of problem domains, such as multirobot teams [2], distributed control [3], resource management [4], and computer games [5, 6]. In such fields, traditional Reinforcement Learning (RL) for single agent is usually inapplicable because of the concurrency and dynamics in MASs [7]. To solve the above problems, various mathematical models have been introduced in MARL, such as Markov Game (MG) [8], multiagent MDPs (MMDPs) [9], and decentralized partially observable Markov Decision Process (Dec-POMDP) [10]. However, most of such MARL models require sufficient information of other agents, including states information and selected actions, which leads to the joint state-action space increasing exponentially with the number of agents. Actually it is difficult to get sufficient information due to the limitations of communication and privacy.

In fact, the interactions between agents are usually sparse in many real-world problems [11, 12]. In such problems, agents only need to consider coordinating their behaviors in sparse states influenced by others. Take the multirobot path finding as an example; coordination only happens when agents are close to each other. Lots of works have been done to exploit the interactions sparseness explicitly to reduce state-action space and thus improve the performance of MARL approaches [13–15]. Traditionally, researchers mainly exploit the hierarchical structure [16] or interdependencies of agents in specific domain problems to reduce the size of joint action space, such as coordination graphs (CGs) [13, 14]. However those dependencies or coordination situations must be predefined for different domain problems.

Recently, much more attention has been attracted to learning in which states agent needs to coordinate with others [11]. The state-of-the-art approaches include CQ-learning [15], independent degree learning (IDL) [17], and MTGA [18], which usually identify coordinated states from statistics to decompose the learning process and receive favorable performance in specific conditions. However, several limitations still exist as follows. Firstly, current approaches usually make strong assumptions about agent or domain structure, which

would confine their practical applications. For instance, CQ-learning and MTGA assume that agent has prior individual optimal policy [15, 18], while IDL and MTGA require agent homogeneity [17, 18]. Secondly, approaches like MTGA construct joint coordination for all influenced agents that are identified simply through monitoring state and reward changes. However, a situation exists when agents influencing others having their own state and reward remain unchanged (e.g., unintentional signal interference) and thus will not be included in the joint coordination. This would lead to potential miscoordination. Lastly, in approaches like CQ-learning, coordinated states are identified only through changes of immediate rewards, which cannot reflect all information about how the environmental dynamics change [18]. However, it may fail in real applications when agents have subtle or even no reward feedback while state transition actually changes. Thus state transition changes should also be considered for all unanticipated situations without valid reward feedback.

To overcome the aforementioned limitations, a modified approach is proposed for effective MARL through exploiting sample grouping and the concept of model difference degree, without additional assumptions about agents or domain structure. First and foremost, a modified Monte Carlo sampling is performed in the Markov Game. Agents record not only their own reward and state transition information, but also the state information of others. This information could be used for further grouping collected samples before identifying which states of other agents bring changes to the agent's current state. After that, a modified concept of model difference degree is introduced to detect changes and measure the difference between the agent performing the task collectively and that performing the task separately. This degree integrates changes of both reward and state transition to evaluate full environmental dynamics. Based on that, the agent's learning process in MASs can then be divided into two different branches. When the degree exceeds certain threshold, those identified states of other agents would be included and coordinated learning is performed; otherwise independent Q-learning is conducted. Experimental results show that the modified approach, with no additional assumptions but better generalization, has its advantage in adapting to broader scenarios. Moreover, in terms of average agent reward per step and convergence, it can learn agents' policy better than existing approaches like CQ-learning and MTGA.

The remainder of this paper is organized as follows. Section 2 introduces necessary background and related work around learning in MASs with sparse interactions. Section 3 describes the proposed coordinated learning approaches. Section 4 tests the proposed approaches in various grid-world environments. Finally Section 5 draws some conclusions and suggests directions for future research.

2. Background and Related Work

In this section, we review some key concepts of Multiagent Reinforcement Learning and related works of learning in MASs with sparse interactions.

2.1. MDP and Markov Game. A Markov Decision Process (MDP) describes a single-agent sequential decision-making problem, in which an agent must choose an action at every time step to maximize its accumulated reward [19–21]. It is the fundamental model of Reinforcement Learning (RL) to learn an agent's individual optimal policy.

Definition 1. A Markov Decision Process is a tuple $\langle S, A, R, T\rangle$, in which S is a finite set of state space, A is a set of actions available to the agent, $R : S \times A \to \mathfrak{R}$ is the reward function that returns the immediate reward $R(s, a)$ to the agent after taking action a in state s, and $T : S \times A \times S \to [0, 1]$ is the transition function representing the transition probability from one state to another when action a is taken. The transition function T and reward function R together define the complete model of the MDP.

The objective of an agent in an MDP is to learn an optimal policy π which maximizes the expected discounted sum of rewards for each state s at each time step t:

$$V^{\pi}(s) = E_{\pi}\left\{\sum_{t=0}^{\infty}\gamma^{t}R\left(s_{t}, \pi\left(s_{t}\right)\right) \mid s_{0} = s\right\}, \tag{1}$$

where $\pi : S \times A \to [0, 1]$ denotes the policy of an agent, E_{π} stands for expectation under policy π, $\gamma \in [0, 1)$ is a discount factor, and s_t denotes the state at time t. This goal can be formulated equivalently by explicitly storing the expected discounted reward for every state-action pair's Q-values:

$$Q^{*}(s, a) = R(s, a) + \gamma\sum_{s'}T\left(s, a, s'\right)\max_{a'}Q\left(s', a'\right). \tag{2}$$

An optimal policy Q^* can be found by computing the optimal state-action value function. One of the most classic and widely used RL algorithms is Q-learning, which is an off-policy model free temporal difference approach to iteratively approximating Q^* by the following update rule:

$$\begin{aligned} Q_{t+1}(s, a) \longleftarrow{}& \left(1 - \alpha_{t}\right)Q_{t}(s, a) \\ &+ \alpha_{t}\left[R(s, a) + \gamma\max_{a'}Q_{t}\left(s', a'\right)\right], \end{aligned} \tag{3}$$

where $\alpha_t \in [0, 1]$ is the learning rate at time step t.

An extension of the single-agent MDP to the multiagent case can be defined by Markov Game, which generalizes the MDP and is proposed as the standard framework for MARL recently [8, 11]. In a Markov Game, joint actions are the result of multiple agents choosing an action independently.

Definition 2. An n-agent ($n \geq 2$) Markov Game is a tuple $M = \langle N, S, \{A_i\}_{i=1}^{n}, \{R_i\}_{i=1}^{n}, T\rangle$, where N is the set of agents in the system, S is a finite set of state space, and A_i is the set of actions available to agent i ($i = 1, 2 \ldots, n$). Let $A = A_1 \times \cdots \times A_n$ be the joint action space. $R_i : S \times A \to \mathfrak{R}$ is the reward function of agent i and $T : S \times A \times S \to [0, 1]$ is the transition function.

In an MG, transition probabilities $T(s_t, a^t, s_{t+1})$ now depend on current state s_t, next state s_{t+1}, and a joint action from state s_t; that is, $a^t = (a_1^t, \ldots, a_n^t)$ with $a_i^t \in A_i$. The reward

function $R_i(s_t, a^t)$ is now individual to each agent i, meaning that different agents can receive different rewards for the same state transition. Furthermore, the reward function R implies the collaboration relation between agents in MASs. In a fully cooperative MG, which is also called Team Markov Game, all agents share the same reward function [9]. In this case, the team joint optimal policy consists with all agents' individual optimal policy. In a noncooperative MG, individual reward functions are not the same or even opposite. Then agents try to learn an equilibrium between agent policies instead of the joint optimal policy [22]. However, it is expensive to calculate the equilibrium policy. Moreover, it is difficult for agents to acquire or estimate the complete state-action information and Q-values of all the other agents in the game. Thus a more general approach is proposed to calculate the expected individual optimal policy based on agents' joint state information and individual selected action.

2.2. Learning in MASs with Sparse Interactions. A wide range of researches have been performed to exploit sparse interactions so as to learn coordinated policy in MASs [11–17]. In earlier researches, much attention is attracted to improving the learning performance by exploiting the hierarchical structure or interdependencies of agents in specific domain problems [13, 16]. For instance, in the hierarchical MARL [16], an overall task is subdivided into a hierarchy of subtasks manually according to prior domain knowledge, so coordinated learning is just considered in the joint of different hierarchies. In sparse cooperative Q-learning (SCQ) [13], Kok and Vlassis adopt coordination graph (CG) to decompose the overall Q-function into local Q-functions which can be optimized individually in the joint state space. However, the approach is limited to fully cooperative tasks and the state space is still exponential with the number of agents. Besides, all the above approaches assume that the dependencies or coordinated situations of agents are constructed beforehand explicitly through network or problem structure.

Some approaches have been developed in recent years with the aim of learning when coordination is beneficial in MASs with sparse interactions [12, 14, 15, 17, 18], which is significant in reducing learning cost and relaxing learning conditions for real-world MASs. For example, in [12], Melo and Veloso propose a two-layer extension of Q-learning algorithm called "learning to coordination" to enable agent to learn its coordinated states. Nevertheless, the learning performance of this algorithm is strongly affected by the penalty value. Kok et al. [14] introduce utile coordination to learn CGs automatically from statistical immediate rewards of other agents. Their approach is still only suited for collaborative multiagent MDP.

Recently, De Hauwere et al. [15] propose an algorithm called CQ-learning to learn in which states an agent coordinating with others is beneficial. The algorithm identifies augmented states by detecting significant difference between the observed reward statistics in MASs and the expected reward in individual MDP. However, the algorithm depends on the assumption that each agent has a prior individual optimal policy. What is more, it only updates Q-values of the coordinated states while it ignores their effect on the former uncoordinated states; therefore the retained optimal individual policy is not guaranteed any more. In FCQ-learning [23], they extend the former with an enhanced detecting mechanism to solve the delayed coordination problems. Yu et al. [17] propose a method named independent degree learning (IDL) to learn coordinated behaviors in loosely coupled MASs. The independent degree for signifying coordination probability is calculated according to individual local observation and can be adjusted dynamically. However, this method is limited to the navigation scenario with two homogeneous robots and needs to be demonstrated in MASs with more agents. From a different view of knowledge transfer and game abstraction, Hu et al. [18] investigate a mechanism called Model Transfer-Based Game Abstraction (MTGA) for efficient sparse interactions MARL. They abstract the action selection in MASs as a one-shot game in each state and reduce the game by removing agents whose similarities of both reward and state transition are not changed significantly. The mechanism achieves better asymptotic performance and higher total rewards than former approaches. However, it still needs individual optimal policy as prior knowledge and computes inaccurate similarity for the state transition changes. What is more, a common situation exists when agents influencing others having their own state and reward remain unchanged and thus will not be included in the reduced game. This would lead to potential miscoordination.

Our work differs from the earlier approaches to learning coordinated states automatically instead of specifying them in advance. Comparing with recent approaches like CQ-learning, IDL, and MTGA, our method requires no strong assumptions about agent or domain structure. We adopt sample grouping technique to identify the specific influence of each pair of agents so as to avoid miscoordination in MTGA. Each agent selects its own action based on augmented coordinated states as CQ-learning, which is different from those using joint action or joint state-action information like game abstraction in MTGA. Besides, a more accurate model difference degree than MTGA is defined for each agent's state to signify the coordination necessity, which evaluates the changes of both reward and state transition and achieves better performance in broader scenarios.

3. Coordinated Learning Approach

This section introduces our modified coordinated learning approach to learn effective coordinated policy in MASs with sparse interactions. We first give a basic approach based on the assumption that agents have already learned the optimal individual policy by completing the single-agent task. Note that the assumption here could be satisfied in certain situations using existing approaches. The main concepts of the approach are as follows:

(1) Identifying coordinated states automatically by grouping collected samples and calculating model difference degree.

(2) Learning agents' coordinated policy according to divided learning processes based on the identified coordinated states.

Based on the basic approach, an extended one is proposed to cover more flexible situations without prior individual knowledge.

3.1. Identify Coordinated States Automatically. For a given MAS, we can build a full MG $M = \langle N, S, \{A_i\}_{i=1}^n, \{R_i\}_{i=1}^n, T\rangle$. Suppose, for each agent i, we have an individual original MDP model $\widehat{M}_i = \langle S_i, A_i, \widehat{R}_i, \widehat{T}_i\rangle$ to finish its single-agent task. This is natural for agent i to apply individual policy to finish its task in the MG when the interactions between agents are sparse. Under this situation, the reward feedback and state transition may be different from those in $\widehat{M}_i$ with other agents' influence. We can construct the empirical local environment model $M_i' = \langle S_i, A_i, R_i', T_i'\rangle$ by conducting Monte Carlo sampling with a random policy in the MG, where R_i' and T_i' are the statistic changed reward function and state transition function. Then the model difference between $\widehat{M}_i$ and M_i' can be evaluated in each state s_i to detect whether agent i should consider coordinating with others in state s_i. Moreover, to specify which state of another agent should be augmented to deal with the coordination, we can group the samples to get a more particular empirical local MDP $M_i'(s_j)$ when another agent j is in state s_j. Then model difference between $\widehat{M}_i$ and $M_i'(s_j)$ can be evaluated to detect whether agent i should consider s_j of agent j to deal with the coordination. Next we elaborate the definition of the model difference degree and the complete coordinated states identification method.

3.1.1. Evaluate Difference of Environmental Dynamics. In an MDP, the environmental dynamics are reflected in reward function and state transition function. To evaluate the coordination necessity, it is vital to evaluate accurately the synthesized difference of the reward and state transition of the individual original MDP and empirical local MDP in the MG.

Inspired by the concept of state distance in an MDP proposed by Ferns et al. [24], we define a concept of model difference degree to evaluate differences in the same state between the individual original MDP and empirical local MDP in an MG, which is a more accurate metric compared with the similarity proposed by Hu et al. [18]. The definition of the model difference degree is given as follows.

Definition 3. Given a Markov Game $M = \langle N, S, \{A_i\}_{i=1}^n, \{R_i\}_{i=1}^n, T\rangle$, for each agent i, let $\widehat{M}_i = \langle S_i, A_i, \widehat{R}_i, \widehat{T}_i\rangle$ be the individual original MDP when agent i acts alone in the Single-Agent System and let $M_i' = \langle S_i, A_i, R_i', T_i'\rangle$ be the empirical local MDP when agent i acts together with other agents in M. For any state $s_i \in S_i$, the model difference degree between $\widehat{M}_i$ and M_i' in s_i is defined as

$$D_{M_i,M_i'}(s_i) = \sum_{a_i \in A_i} \left\{ \left| \widehat{R}_i(s_i, a_i) - R_i'(s_i, a_i) \right| + \gamma \Gamma_{d_{M_i}}^k \left(\widehat{T}_i(s_i, a_i), T_i'(s_i, a_i) \right) \right\}, \tag{4}$$

where γ is the discount factor and $\Gamma_{d_{M_i}}^k(\widehat{T}_i(s_i, a_i), T_i(s_i', a_i))$ is the Kantorovich distance between the probabilistic distributions $\widehat{T}_i(s_i, a_i)$ and $T_i'(s_i, a_i)$ that measures the supremum of all the probability difference for each potential transferred state.

From (4), it can be found that difference between the two MDPs in the same state s_i is equal to the weighted sum of the reward difference and the state transition probabilities difference for each available action, summarizing all information about the changes of the environmental dynamics in the same state s_i. If other agents have no impact on agent i in s_i, the reward function and the state transition function will remain unchanged; thus the model difference degree in s_i between the two MDPs is equal to zero approximately. If both reward and state transition of agent i are changed after interacting with other agents, the contribution of state transition dynamics will reinforce the identification of the reward change. Even in some situations when the intermediate reward feedback is not provided beforehand and state transition is changed because of others' influence, we can still evaluate the difference according to state transition change.

In comparison, the model similarity proposed by Hu et al. in MTGA [18] computes the similarities of two MDPs by the summation of relative state distance between current state s_i and all the other states in state space. It can reflect reward difference effectively but cannot accurately evaluate the state transition change according to the concept of Kantorovich distance [24]. This is because the supremum of state transition distance between s_i and other states always keeps 1 and no state transition difference can be detected. The computational complexity of the proposed model difference degree is $O(1)$ and is much smaller than $O(m)$ of MTGA, where m denotes the size of the state space. Compared with the KS-test [15] and Friedmann-test in FCQ-learning [23], it is more flexible because of the consideration of the state transition dynamics.

3.1.2. Model-Based Method for Identification. In this section, our complete coordinated states identification method is elaborated based on the model difference degree defined in Section 3.1.1.

Figure 1 is a detailed graphical representation of our method to identify coordinated states in MASs with sparse interactions. As the top of Figure 1 shows, for each agent i in the MG, Monte Carlo sampling is first conducted to collect data at each step, including current state, executed action, received reward, transferred next state, and the states information of other agents. The action selection is performed as ε-greedy random policy, where ε is set to a small value like 0.1. After sampling, we group the rewards and transferred states recorded based on the local state of other agents. For instance, in Figure 1, the collected data of agent i in s_i^3 are grouped according to agent j's local states s_j^1 and s_j^2. Thus we can calculate new reward function $R_i'(s_i^3, a_i^1, s_j^2)$ and state transition function $T_i'(s_i^3, a_i^1, s_j^2)$ statistically for the situation that agent i takes action a_i^1 in state s_i^3 at the time agent j is in state s_j^2. The reward function is the average of the collected rewards and the state transition function is the probability distribution of the recorded transferred states. The set of the calculated reward functions and state transition functions

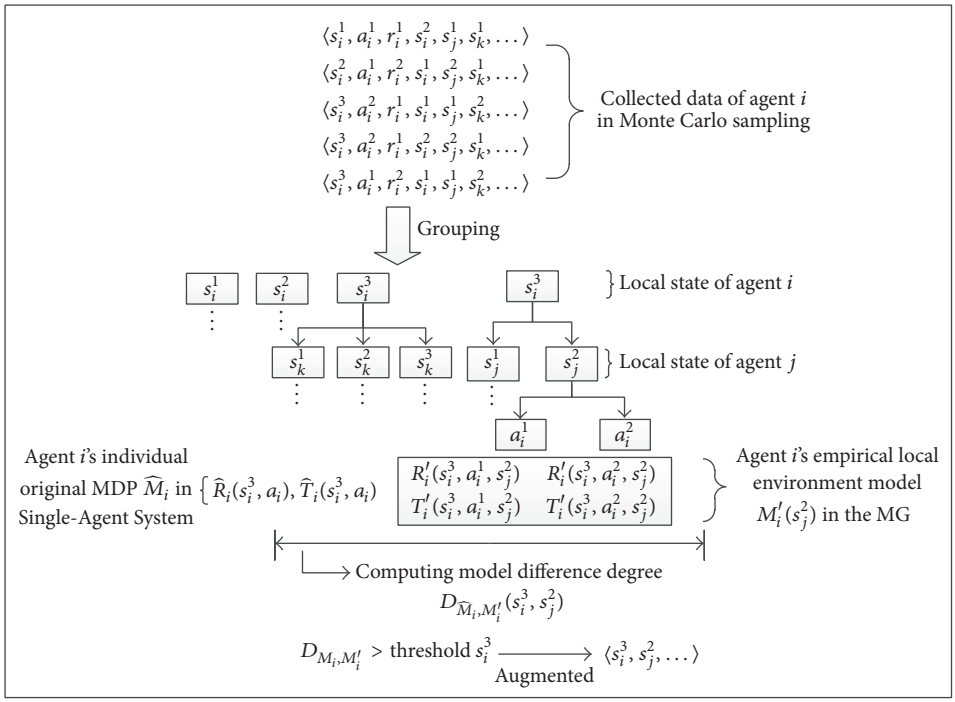


Figure 1: Identify coordinated states with the proposed approach.

for all the available actions depicts the particular empirical local environment model $M_i'(s_j^2)$. Thirdly, we compute the particular model difference degree $D_{\widehat{M}_i,M_i'}(s_i^3, s_j^2)$ for agent i in state s_i^3 to evaluate the changes of environmental dynamics caused by agent j in state s_j^2. This difference is calculated based on agent i's individual original MDP $\widehat{M}_i$ and $M_i'(s_j^2)$ according to (4) in Section 3.1.1. If the degree exceeds a significant difference threshold, agent i should augment the coordinated state in state s_i^3 to include state s_j^2 of agent j to deal with the coordination; otherwise it performs action as finishing a single-agent task.

The complete pseudocode of our coordinated states identification method is given in Algorithm 1, through which the agent can determine whether or not the coordination is needed in each state and which agents should be considered in its coordination. It should be noted that although all states' information of other agents is recorded for grouping in Algorithm 1, we can restrict the number of agents recorded according to specific influence distance (e.g., sensor radius and field of fire) in real-world MASs. The distance can be a variable to describe the interacting bound depending on the agent type and current state. It can release the computational cost of the approach further.

3.2. Learning Coordinated Policy. After determining the coordinated states of each agent, we can divide the multiagent learning process in an MG into independent learning in uncoordinated states and joint learning in coordinated states. In this section, a Q-learning based learning rule is proposed to guide agents to learn optimal coordinated policy. It performs action selection and Q-values update according to the two subprocesses, respectively.

The action selection works as follows: when the learning process starts, an agent checks whether its current local state is included in its coordinated states. If so, it will further check if the observed global state contains its augmented coordinated state in current local state. If this is the case, it selects its action according to the augmented coordinated state; otherwise it selects action independently only using its own local state information. If the local state of agent is not included in its coordinated states or not augmented in the current joint state, it can also select action independently.

According to the relation of current local state and transferred state, there are following scenarios to update the Q-values in the learning process.

(1) An agent is in a coordinated state and it needs to select an action using global state information in the coordination union.

In this situation, we will check if the next transferred state is also in a coordinated state needing to use global state information to select an action. If this is the case, the following update rule is used:

$$Q_i^j\left(js_i, a_i\right) \longleftarrow (1-\alpha)\, Q_i^j\left(js_i, a_i\right) + \alpha\left[R\left(s_i, a_i\right) + \gamma \max_{a_i'} Q_i^j\left(js_i', a_i'\right)\right]. \tag{5}$$

Input: Individual original MDP $\widehat{M}_i$, Individual optimal Q-values Q_i of agent i, threshold value proportion τ, integer L for Monte Carlo sampling times, exploration factor ε
Output: The set of coordinated states for agent i
// performing Monte Carlo sampling to get empirical local MDPs.
(1) **for** $t = 1, 2, \ldots, L$ **do**
(2) % decreases ε down to a small value $m\varepsilon$ multiplying with factor $d\varepsilon$;
(3) selects $a_i(t)$ according to Q_i using local state $s_i(t)$ with random policy ε;
(4) observes local r_i, s_i' according to global environmental state and action;
(5) records r_i, s_i' and state information of other agents;
(6) % updates individual Q-values according to received experience $\langle s_i, a_i, r_i, s_i' \rangle$;
(7) **end for**
(8) **for** $\forall$ agent $j \in N$ and $j \neq i$ **do**
(9) **for** $\forall$ state $s_j \in S_j$ **do**
(10) gets empirical local MDP $M_i'(s_j)$ of agent i when agent j is in state s_j;
(11) **end for**
(12) **end for**
// determining the coordinated states of agent i according to the two MDPs.
(13) **for** each state $s_i \in S_i$, $\forall$ agent $j \in N$ and $j \neq i$, state $s_j \in S_j$ **do**
(14) computing all the model difference degree $D_{\widehat{M}_i, M_i'}(s_i, s_j)$ according to (4);
(15) **if** $D_{\widehat{M}_i, M_i'}(s_i, s_j)$ is bigger than τ **then**
(16) augments the coordinated state of agent i in state s_i to include s_j of agent j;
(17) **end if**
(18) **end for**

Algorithm 1: A model-based method for identifying coordinated states of agent i.

Otherwise, the individual maximal Q-values of next transferred state are used to update current coordinated Q-values:

$$Q_i^j(js_i, a_i) \longleftarrow (1-\alpha) Q_i^j(js_i, a_i) + \alpha \left[R(s_i, a_i) + \gamma \max_{a_i'} Q_i(s_i', a_i') \right]. \quad (6)$$

(2) An agent is in a state where it selects an action using only its own state information.

In this case, if the next transferred state is in a coordinated state needing to use global state information to select an action, the following update rule is used:

$$Q_i(s_i, a_i) \longleftarrow (1-\alpha) Q_i(s_i, a_i) + \alpha \left[R(s_i, a_i) + \gamma \max_{a_i'} Q_i^j(js_i', a_i') \right]. \quad (7)$$

Otherwise, the standard single-agent Q-learning rule is used with only local state information.

$$Q_i(s_i, a_i) \longleftarrow (1-\alpha) Q_i(s_i, a_i) + \alpha \left[R(s_i, a_i) + \gamma \max_{a_i'} Q_i(s_i', a_i') \right]. \quad (8)$$

In the above equations (5) to (8), Q_i stands for the Q-values of individual local state s_i and action a_i pair of agent i, which can be initialized by prior individual optimal Q-table if single-agent knowledge has been learned. Q_i^j stands for the Q-values of the augmented state js_i and action a_i pair of agent i. Note that js_i contains s_i and the augmented Q-table is initially empty. Let s_i' be the transferred state after performing action a_i. If s_i' is in the coordinated state and the observed transferred global state information contains augmented state of agent i, the augmented state is denoted as js_i'. a_i' is the next selected action of agent i.

Equations (5) to (8) show us the Q-values update of the possible state transition relation in the learning process, which bootstraps Q-values of the augmented state and local state, respectively, to optimal convergence. Equation (5) and (6) represent the accumulation of learning experience step by step to solve the coordination problem in sparse coordinated states. It learns effective coordination by taking other agents' influencing states into consideration to select individual action in the sufficient trials. Equation (7) and (8) represent the local adaptation of the prior individual Q-table to get an optimal independent learning policy, which can avoid suboptimal policy because of the change of hinder Q-values. In comparison, CQ-learning only updates its joint Q-table and ignores coordinated states' effects on the Q-values of former uncoordinated state; thus the individual policy that remained beforehand will not be guaranteed as an optimal policy.

The pseudocode for our whole model-based difference degree learning approach is given in Algorithm 2. We can see that the approach is executed in two phases. In coordinated states identification process (line (2)), the model difference degree between $\widehat{M}_i$ and $M_i'(s_j)$ for each of the agents i and j costs major computational time of the identification approach. Thus the computational complexity of Algorithm 1 for all the agents is $O(m^2 n^2)$, where m denotes the size of state space and n denotes the number of agents in the

Input: Learning rate α, discount factor γ, learning exploration factor ε, Individual original MDP M_i, Individual optimal Q-values Q_i of agent i, threshold value proportion τ, integer L for Monte Carlo sampling, time limit for per episode learning $Maxstep$

(1) Initialize $Q_i(s_i, a_i)$ with individual optimal policy of agent i, initialize Q_i^j to {};
(2) Identify coordinated states for agent i calling Algorithm 1;
(3) Initialize local state s_i for agent i, check whether initial states is in coordination;
(4) **for** $t = 1, 2, \ldots, Maxstep$ **do**
(5) observe current local state s_i for agent i;
(6) $s_g \leftarrow (s_1, s_2, \ldots, s_N)$;
(7) **if** $\forall$ agent $i \in N$, agent i is in coordination at time t **then**
(8) select $a_i(t)$ according to Q_i^j using $js_i(t)$;
(9) **else**
(10) select $a_i(t)$ according to Q_i using $s_i(t)$;
(11) **end if**
(12) receive reward r_i and transition state s_i' for each agent i;
(13) **if** $\forall$ agent $i \in N$, s_i' is part of an augmented coordinated state js_i' and js_i' is included in the new global state s_g' **then**
(14) **if** js_i' is not in state space Q_i^j **then**
(15) extend Q_i^j to include joint state and all the available actions pair (js_i', a_i);
(16) $Q_i^j(js_i', a_i) \leftarrow 0$;
(17) **end if**
(18) mark agent i is in coordination at time $t + 1$ and coordinated states for agent i is js_i';
(19) **end if**
(20) **if** $\forall$ agent $i \in N$, agent i is in coordination at time t **then**
(21) **if** agent i is in coordination at time $t + 1$ **then**
(22) Update $Q_i^j(js_i, a_i)$ according to (5);
(23) **else**
(24) Update $Q_i^j(js_i, a_i)$ according to (6);
(25) **end if**
(26) **else**
(27) **if** agent i is in coordination at time $t + 1$ **then**
(28) Update $Q_i(s_i, a_i)$ according to (7);
(29) **else**
(30) Update $Q_i(s_i, a_i)$ according to (8);
(31) **end if**
(32) **end if**
(33) $js_i \leftarrow js_i'$, $s_i \leftarrow s_i'$, $s_g \leftarrow s_g'$;
(34) **if** s_g is a terminal state **then** return;
(35) **end for**

Algorithm 2: Model-based difference degree learning approach for agent i.

system. In comparison, the complexity in MTGA is $O(m^2 n)$, because it only computes the similarities to identify whether agent i should coordinate with others to avoid confliction. Compared with the above identification process running only once, the coordinated learning process (lines (3) to (35)) runs a great number of steps until the terminal states are reached. The computational complexity of the proposed approach at each time step is $O(n)$, which is the same as CQ-learning. For MTGA, the equilibrium should be computed based on the joint action space for the identified sparse coordinated states, so the computational complexity is higher than that of our approach.

3.3. Learning without Prior Individual Q-Values. In the above approach, we assume that individual optimal policy in completing a single-agent task has been trained beforehand as prior knowledge, which is sometimes not practical in real world. In this section, we introduce a variation of our approach to learn coordinated policy without prior individual optimal policy. The main difference lies in that we perform not only sampling but also individual Q-values update like independent learning in the phase of coordinated states identification. Through that, we can get not only empirical local MDP model to compute model difference degree of each agent, but also individual empirical knowledge to initialize individual Q-table in coordinated learning.

Specifically, we make the following changes in the pseudocode of the proposed approach. In Algorithm 1, we extend lines (2) and (6) to implement individual knowledge learning. Line (2) shows us the change of action selection policy in sampling, and line (6) is a simple process of individual Q-value update. The action selection policy is changed as

follows. At the beginning of Monte Carlo sampling, the action selection policy ε is set to a big value approaching 1 like 0.9999, which simulates completely random exploration without prior knowledge. ε is then decreased by multiplying with factor $d\varepsilon$ in each episode and down to a minimal value $m\varepsilon$ like 0.1, leading the sampling process from complete exploration to full exploitation. Through above changes in coordinated states identification, we can perform completely random exploration at the beginning to access the real model in the state space and then accumulate experience and make use of the learned knowledge to detect environment delicately around the suboptimal path later. In the learning phase, the accumulated local empirical Q-values in sampling can be used to replace individual optimal policy to initialize individual Q-table in line (1) of Algorithm 2, which will speed up the learning convergence.

4. Experiments

In this section, a series of experiments are carried out on grid-world games to test the effectiveness of our approaches in different scales of MASs with sparse interactions, especially in situations where the existing approaches cannot solve the coordination correctly. The experiments are run single-threaded on an Intel Core i5, 3.20 GHz CPU with 3.45 GB using the Windows XP 32-bit operating system and Matlab 2014a. The basic learning approach is denoted simply as difference degree learning (DDL), and the extended approach without initialized individual optimal Q-values is denoted as DDL-NI.

4.1. Experimental Setting. The benchmark environment is a set of multiagent grid-world games presented in Figure 2. Games (a) to (f) with 2 agents are the same as those used by De Hauwere et al. [15], where agents can collide with other agents in any state. Games (g) and (h) with more than 2 agents are the same as those used by Hu et al. [18], where agents collide with others only in the shaded grids areas. In all these games, agents are required to reach their goals within certain steps and avoid collisions. The goal of each agent in 2-agent games is the starting position of the other one, while goals in games (g) and (h) are denoted by G_i. At each step, agent can choose to move in four directions, that is, *Up*, *Down*, *Left*, and *Right*, and transfers to another state with certain probabilities. In Sections 4.2.2 and 4.2.3, if agents collide with each other, both will break down and are transferred back to their original states.

Our approaches, DDL and DDL-NI, are compared with two state-of-the-art approaches, CQ-learning and MTGA. The approaches CQ-learning, MTGA, and DDL all need prior individual optimal policy to initialize individual policy or lead sampling. The equilibrium calculation in MTGA differs from our prior conditions for it requires states information and corresponding Q-values from all the other agents in the game. To compare the effectiveness of coordinated states identification, we implement MTGA with the same action selection mechanism to select the optimal action based on joint coordinated states information.

The basic parameter settings are presented in the following list, which is similar to De Hauwere et al.'s [15] except the nondeterministic state transition. All experiments based on the four approaches are run with a learning rate of 0.05 and a discount factor of 0.9. The exploration is regulated using a fixed ε-greedy policy with $\varepsilon = 0.1$. In DDL-NI, the sampling exploration is set to 0.9999 at first, the minimal exploration value $m\varepsilon$ to 0.1, and the decreasing factor $d\varepsilon$ to 0.99. In DDL, DDL-NI, and MTGA, the number of sampling episodes is 500, with the threshold value τ set to 0.2 times the maximal value of the computed MDP differences as that used by Hu et al. [18]. Rewards are given as follows: the expected reward for an agent to reach its goal is set to +20 except CMU and the expected penalty for colliding with a wall is set to −1. Because the size of CMU is larger than other games, the expected reward for an agent to reach goal in CMU is set to +200 to lead learning. The expected penalty for colliding with another agent is different in various environment settings, denoted as σ. State transitions are made stochastic by assigning a success probability ρ to agents' actions in different experiment settings. Agents were trained for 10,000 episodes with corresponding configuration and the resulting policy is then played greedily for 1000 episodes. Each episode has a time limit of 50,000 steps. All results are then averaged over 10 runs.

Parameter Settings in Our Experiments. (Parameters: Meanings, Values)

α: the learning rate, 0.05

γ: the discount factor, 0.9

ε: the exploration in ε-greedy policy, 0.1

$m\varepsilon$: the minimal exploration value in DDL-NI, 0.1

$d\varepsilon$: the decreasing factor of exploration policy in DDL-NI, 0.99

τ: the threshold value to identify coordination in MTGA, DDL, and DDL-NI, 0.2

R: the expected reward for reaching goal except CMU (+200 in CMU), +20

R: the expected reward of colliding with a wall, −1

Maxstep: the time limit for per-episode learning, 50000

L: the number of sampling episodes in MTGA, DDL, and DDL-NI, 500

N_l: the number of learning episodes, 10000

N_{avg}: the number of final policy played times in each run, 1000

Run: the number of all the learning process performed times, 10.

4.2. Results and Analysis. Experiments under different environment conditions are carried out to demonstrate our approaches' advantages, which are performed as three parts. Firstly, experiments are conducted under more general

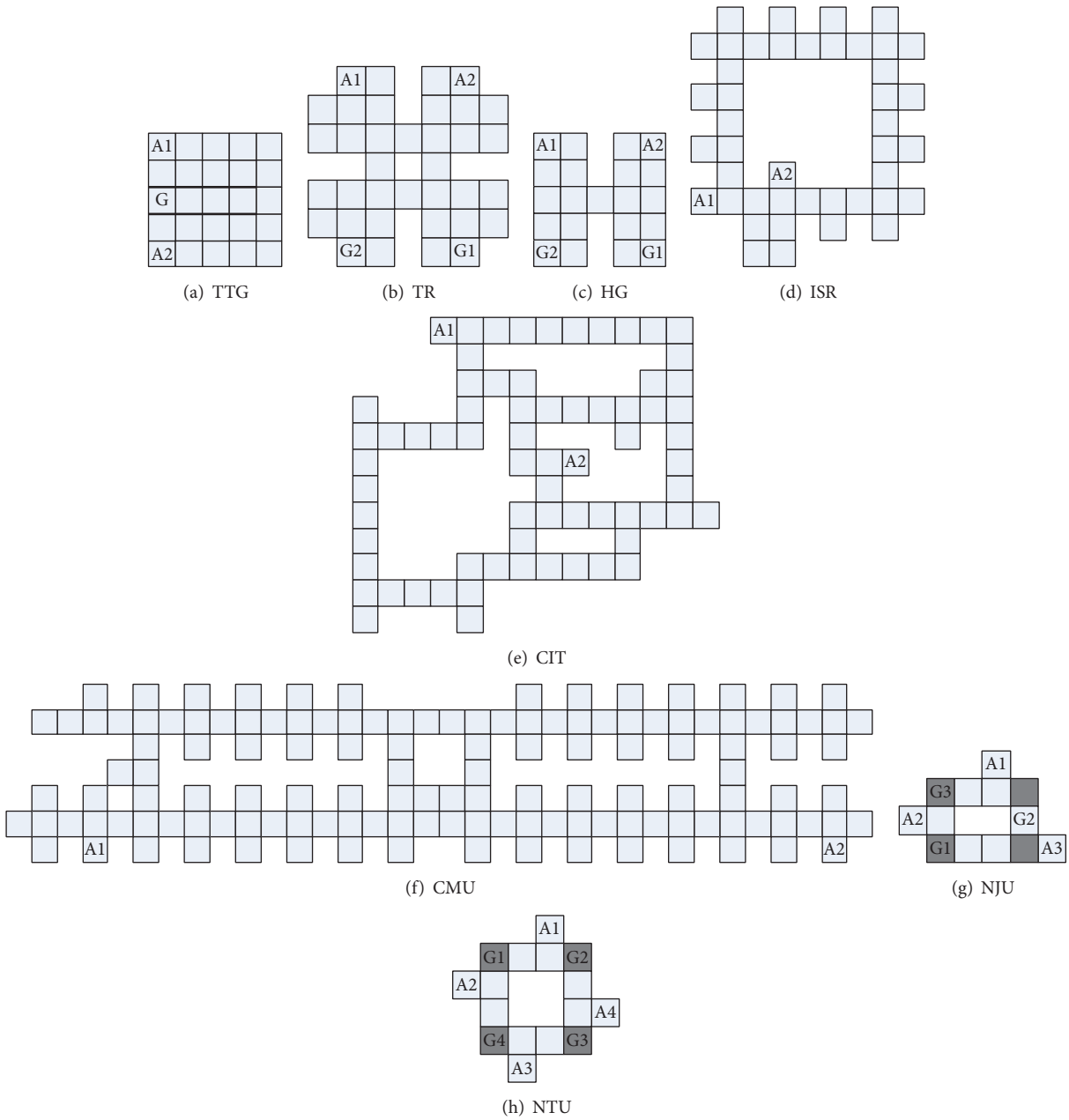


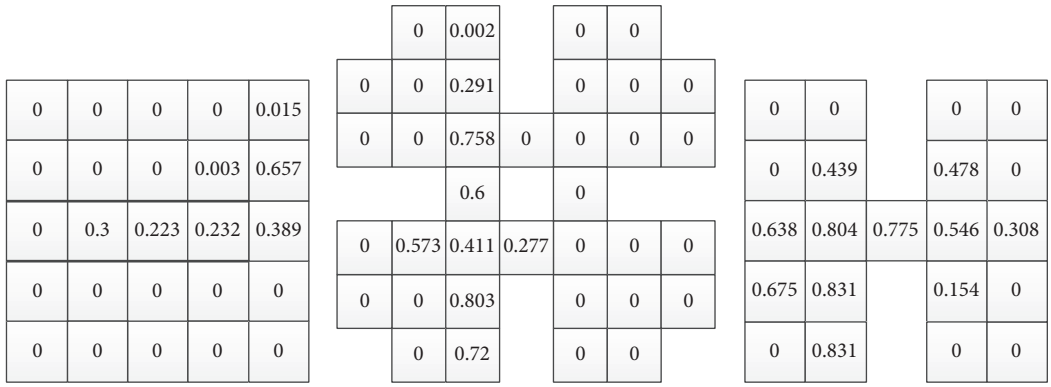
Figure 2: Different multiagent grid-world games used throughout this experiment.

conditions. Secondly, the results of basic learning performance are compared with the tested approaches in 2-agent nondeterministic environment. At last, we demonstrate our approaches in MASs with more than two agents.

During the process, we record the number of steps reaching the goal, reward, collision times, and the average reward per step. The average number of steps before reaching the goal depicts the time cost of finishing the task. The average reward specifies the summation of reward to goal and penalty for colliding with other agents or walls. The average reward per step (ARPS), which is the ratio of average reward and average number of steps to goal, is a synthesized criterion to measure each agent's learning performance. Therefore, the ARPS curves with episodes can reflect the learning dynamics and convergence performance with their final values portraying the performance of final learned policy.

4.2.1. Learning Performance Comparison in Some Broader Scenarios. In this section, we examine the tested approaches to compare their effectiveness and robustness in two broader types of 2-agent scenarios.

In the first scenario, agents are presented in deterministic environment; that is to say, $\rho = 1.0$. The particular condition is that agents have no penalty feedback $\sigma = 0$ when one agent collides with the other. It represents the situation when intermediate state rewards can not be listed beforehand for



(a) TTG (b) TR (c) HG

Figure 3: The computed model difference degrees in small scale games TTG, TR, and HG.

all unanticipated circumstances; thus agents should consider the change of state transition to identify coordinated states.

Figure 3 shows the model difference degree in three small domains from the perspective of agent 1, which are computed with (4). These values describe the difference degree between the individual original MDP when agent 1 acts alone and the empirical local MDP when agent 1 acts together with agent 2. A higher value means that agent 1 is more potentially influenced by agent 2, and thus coordinating with agent 2 during decision-making should be considered. As expected, the values in the conflicting states, that is, the areas near the entrance or doorway, are much higher than that of those "safe" states. For example, in TTG, the state with the biggest value 0.657 is exactly the position where agent 1 collides with agent 2 frequently when both agents perform actions under individual optimal policy. The change of difference degree reflects the level of necessity for agent 1 coordinating with agent 2. We can see that difference degrees of states surrounded by walls are bigger than those of states on the left of the most "dangerous" one. This is because the former states are laid on the only way to goal while the latter one is far from the optimal route of agent 2 to collide. (b) and (c) depict the coordination necessity when two agents perform tasks as near-optimal policy in games TR and HG.

Note that the difference degree is computed only based on the collected data of state transition, which is inaccessible to those methods dependent on reward only, like CQ-learning. Therefore, those methods cannot deal with collisions to learn a coordinated policy. For MTGA, the similarities computed are all 0 in the conditions of agents having no reward feedback. Because the supremum of state transition distance keeps 1 all the same and no state transition difference will be detected. it cannot identify accurate coordinated states in the scenario.

Figure 4 shows ARPS learning curves of the tested approaches in different games in the first scenario. To be specific, without penalty feedback for the collision, agents using CQ-learning cannot identify coordinated states to avoid collision; thus they cannot learn coordinated policy to finish tasks. For MTGA, agents also cannot detect the difference of the two MDPs owing to the inaccurate evaluation of state transition changes; consequently they cannot reach goals. For DDL and DDL-NI, agents can learn effective coordinated policy as well as in the conditions without reward feedback.

In terms of the observed data, the changes of states transition are more primary than the reward feedback in reflecting environmental dynamics' changes caused by other agents. Thus, it is more important to consider the changes of states transition to exploit the relations between agents, which can learn approximate optimal coordinated policy based on the least observed information more flexibly.

In the second scenario, agents are also presented in deterministic environment. There is a special condition when agent 1 receives expected penalty −10 and returns to its original state after colliding with agent 2, while the state transition and reward of agent 2 are not influenced by agent 1. This is a quite common situation when influences of an interaction are unilateral in a heterogeneous MAS.

Figure 5 shows the learning curves of agent 1 by the tested approaches in the second scenario. We can see that the CQ-learning, DDL, and DDL-NI all learn favorable final policy in different games. For example, in game HG, the final ARPS values achieved by CQ-learning, DDL, and DDL-NI all reach around 1.8. In game CMU, this value reaches about 5.6 for CQ-learning and DDL-NI, indicating that CQ-learning can learn equivalent policy to our approaches in deterministic environment. However, the final ARPS value achieved by MTGA is obviously lower than that of all the other approaches except in game TR, which is only 1.0 in game HG. In game CMU, MTGA cannot learn a convergent policy within the allowed time limit. Because game TR provides multiple routes for agents to reach goals, agents can learn an optimal policy without collisions in MTGA.

Note that influences of the interactions between two agents are not as mutual as collision of robots. For CQ-learning, DDL, and DDL-NI, two agents deal with the changes caused by the other agent's interaction in each state, respectively. Specifically, after identifying the state where agent 2 influences agent 1 obviously, agent 1 will construct a joint state for itself and thus to avoid miscoordination. However, without changes being detected, agent 2 still selects its action on its own. For MTGA, it only constructs an overall

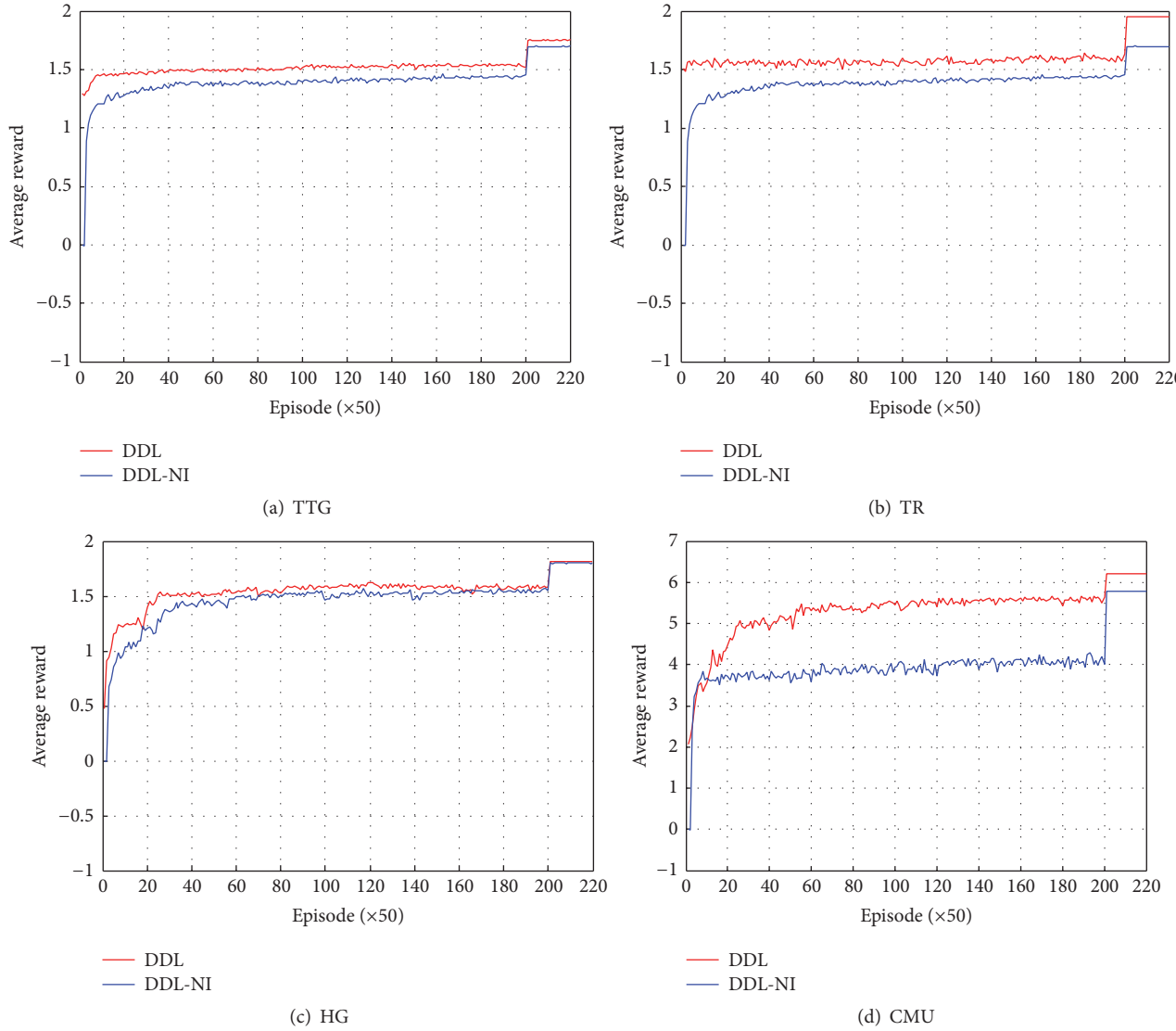


Figure 4: The learning curves of the tested approaches in games TTG, TR, HG, and CMU when $\rho = 1.0$ and $\sigma = 0$.

joint coordination including influenced agent 1. Thus MTGA performs like an independent learner in these conditions, which treats the remaining agents simply as part of the environment and ignores the coordination requirements. So the learning curves of MTGA show big fluctuating errors and lead to a nonconvergent policy in complex MASs like CMU.

4.2.2. Learning Performance Comparison in 2-Agent Nondeterministic Environment. In this section, agents are assigned a stochastic transition probability of 0.8 to reach its expected state and 0.2 of failure in original state. The expected penalty for colliding with another agent is −10. Due to space limitation, we only show in Figure 6 the ARPS learning curves of four tested approaches in partially typical games, including three small ones (TTG, TR, and HG) and the most complicated 2-agent game CMU. The complete results of final policy for four approaches are showed in Table 1.

In Figure 6, we examine the average results of the two agents according to the learning process and final policy received. It should be noted that, at episode 10000, there is abrupt climbing of ARPS because the final learned policy is performed without random action selection any more.

For the learning process, we can see that, at the beginning, DDL, DDL-NI, and MTGA perform rapid climbing and receive favorable ARPS values compared to CQ-learning. Because the former approaches identify coordinated states beforehand in the sampling of early episodes, it can make use of prior knowledge transfer. CQ-learning augments coordinated states continually as reward collecting. As a result, it fluctuates obviously with the exploration of coordination and takes more time to reach convergence.

In terms of final ARPS values achieved, our approaches can learn a better policy compared with CQ-learning and MTGA. For example, the values achieved by DDL and DDL-NI finally reach almost 2.0 in game HG, while MTGA reaches

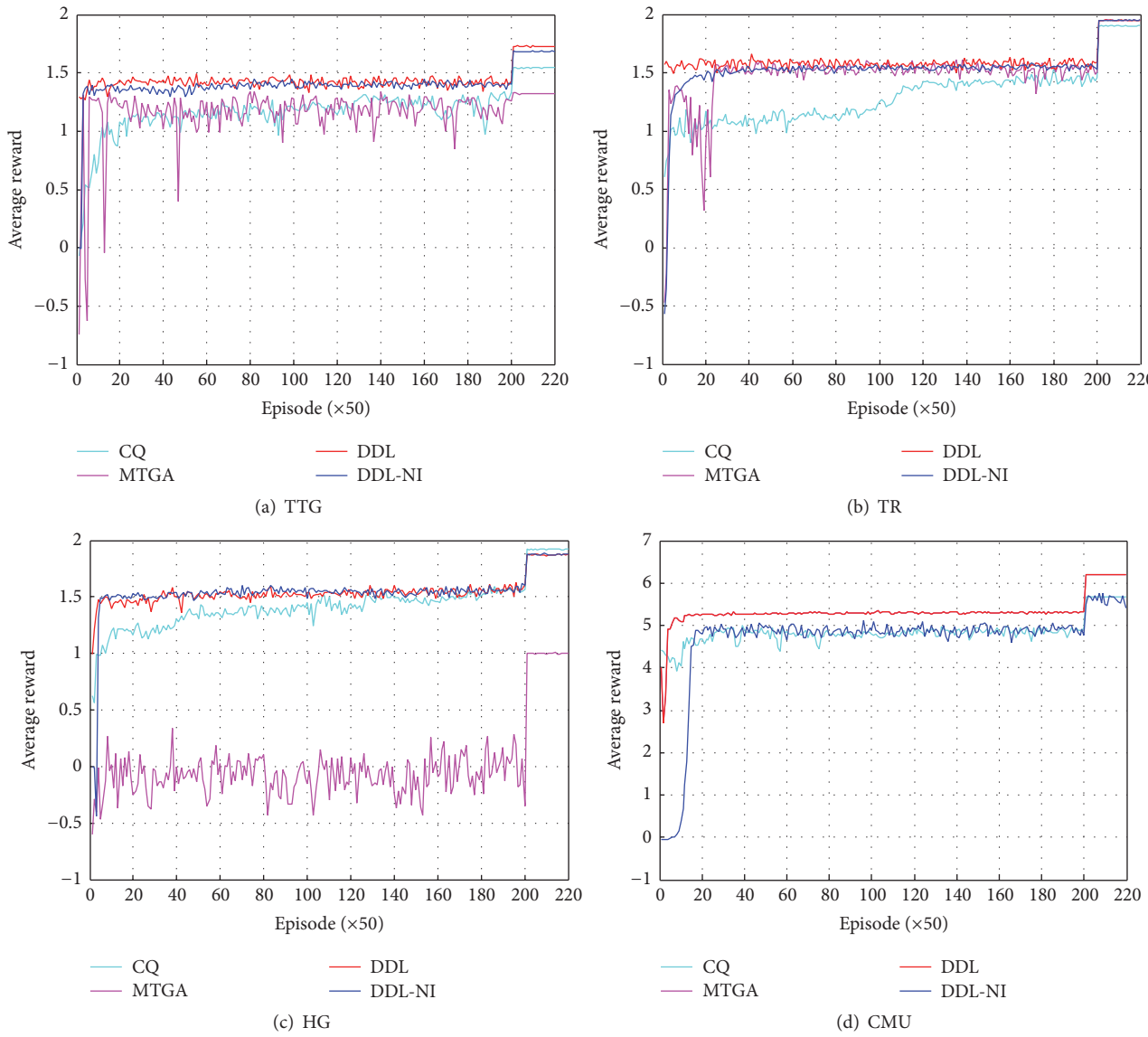


Figure 5: The learning curves of the tested approaches in games TTG, TR, HG, and CMU when interactions between agents are unilateral.

only 1.6. In game CMU, the difference is more significant. DDL finally reaches a high value of 5.9, compared with DDL-NI reaching a value of about 5.7, MTGA reaching a value around 4.0, and CQ-learning having no convergent policies due to exceeding the maximal time limit. In MTGA, as the accuracy of model similarity would decrease because of its rough evaluation of state transition, redundant coordinated states would be expanded to avoid collision and assure convergence. Clearly, for CQ-learning, nondeterministic state transition would give rise to the number of coordinated states which should be taken into consideration, making immediate reward difference get blurred to identify. What is more, it only updates coordinated states' Q-values, which restrict the exploration of possible coordinated states. Thus necessary coordinated states may not be expanded in CQ-learning when the size of environment state is large like CMU. In comparison, the model difference degree in DDL and DDL-NI would keep pace with the changes of environmental dynamics to highlight the coordination necessity accurately. Due to the absence of individual optimal policy, DDL-NI cannot get final ARPS values as well as DDL. But it also receives approximate optimal policy better than MTGA and CQ-learning in most situations, which encourages our approach to be applied in more flexible MASs.

The overall results of average final learned policy for the 2-agent games are given in Table 1, including number of augmented states, number of collisions, number of steps, and received reward value in distinct environments by certain algorithms. In agreement with the results in Figure 6, Table 1 shows that our approaches take the least number of moving steps and receive the biggest reward in most games, whereas CQ-learning cannot assure learning convergent policy in large scale nondeterministic games. For example, in TTG, CQ-learning takes average 13.503 steps and receives average

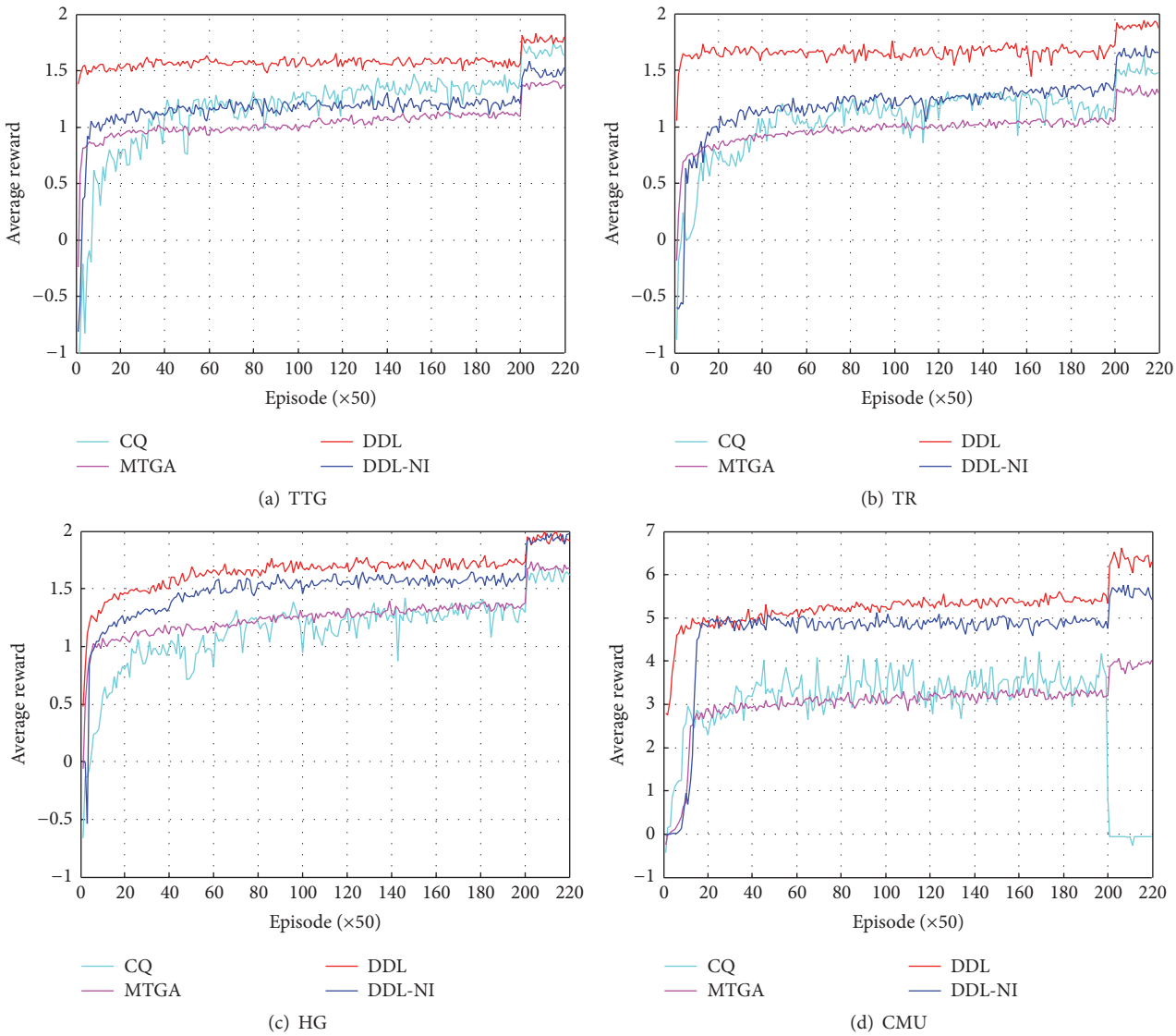


Figure 6: The learning curves of the tested approaches in games TTG, TR, HG, and CMU when $\rho = 0.8$ and $\sigma = -10$.

reward value of about 22.594, which is better than the values of 15.767 and 21.719 in MTGA, respectively. In CMU, CQ-learning gets a large average number of moving steps of 39685 and average reward of −645.73. The number of collisions also shows that agents can learn effective policy without collision into other agents in most games except CMU by CQ-learning. The number of total augmented states reflects the main reason of performance difference. It can be found that MTGA expands most coordinated states, in TTG it augments about 76.4 states, and in CMU it augments more than 16000. In comparison, DDL and DDL-NI expand the least necessary coordinated states to avoid collision.

As a whole, the proposed approaches perform favorable convergence and learn better coordinated policy in 2-agent nondeterministic environment, which outperforms CQ-learning and MTGA.

4.2.3. Learning Performance Comparison with More Than 2 Agents in Nondeterministic Environment. In this section, we show our experimental results in MASs with more than 2 agents. Games NJU and NTU have 3 and 4 agents, respectively. The transition probability is the same as those in 2-agent games. Agents receive an expected penalty of −10 when colliding with others in the shaded areas. For exhibiting the results better, each subfigure in Figures 7 and 8 only plots the learning curves of one agent using different approaches.

Figure 7 depicts the learning curves of the tested approaches in game NJU. We can see that CQ-learning cannot learn a convergent policy in the end because of large size of state space and nondeterministic environment. In comparison, the other three approaches get favorable convergence for the final learned policy. The ARPS values of agents 1, 2, and 3 for MTGA finally converge to 2.1, 1.8, and

Table 1: Results of final learned policy by the tested approaches in the different 2-agent games.

Env	Alg	# state	# actions	# coll	# step	# reward
TTG	CQ	76.4 ± 10.807	4	0 ± 0	13.503 ± 0.169	22.594 ± 0.771
	MTGA	218.9 ± 25.101	4	0 ± 0	15.767 ± 0.115	21.719 ± 0.461
	DDL	27.5 ± 0	4	0 ± 0	13.117 ± 0.089	23.27 ± 0.518
	DDL-NI	46.45 ± 3.316	4	0 ± 0	12.866 ± 0.086	19.135 ± 0.654
TR	CQ	137.7 ± 17.598	4	0 ± 0	15.528 ± 0.246	23.188 ± 0.611
	MTGA	1143.2 ± 37.91	4	0 ± 0	17.080 ± 0.112	22.365 ± 0.566
	DDL	49.4 ± 1.538	4	0 ± 0	12.700 ± 0.062	24.061 ± 0.585
	DDL-NI	109.4 ± 9.85	4	0 ± 0	13.393 ± 0.104	22.053 ± 0.571
HG	CQ	108.8 ± 5.862	4	0 ± 0	14.194 ± 0.259	22.937 ± 0.709
	MTGA	414.6 ± 13.802	4	0 ± 0	13.668 ± 0.126	22.963 ± 0.521
	DDL	35.3 ± 0.823	4	0 ± 0	12.051 ± 0.104	23.356 ± 0.549
	DDL-NI	77.3 ± 2.975	4	0 ± 0	11.981 ± 0.124	23.186 ± 0.475
ISR	CQ	99 ± 3.606	4	0 ± 0	7.683 ± 0.349	23.081 ± 2.306
	MTGA	796.6 ± 134.58	4	0 ± 0	7.389 ± 0.101	22.658 ± 0.691
	DDL	52 ± 0	4	0 ± 0	6.518 ± 0.113	23.212 ± 0.867
	DDL-NI	62.7 ± 0.498	4	0 ± 0	6.462 ± 0.096	23.445 ± 0.555
CIT	CQ	126.8 ± 9.102	4	0 ± 0	19.997 ± 0.653	21.405 ± 1.744
	MTGA	4203 ± 140.254	4	0 ± 0	19.249 ± 0.132	19.277 ± 0.589
	DDL	93 ± 0	4	0 ± 0	16.083 ± 0.222	22.063 ± 0.575
	DDL-NI	90 ± 0.642	4	0 ± 0	19.513 ± 0.147	18.211 ± 0.841
CMU	CQ	340.4 ± 32.362	4	9.995 ± 32.4	39685 ± 573.7	−645.73 ± 468.1
	MTGA	16803 ± 84.819	4	0 ± 0	60.351 ± 0.225	236.656 ± 5.995
	DDL	181.35 ± 0.880	4	0 ± 0	39.197 ± 0.153	248.615 ± 6.71
	DDL-NI	296 ± 15.973	4	0 ± 0	43.732 ± 0.209	245.101 ± 7.568

2.4. For DDL and DDL-NI, the values are 3.0, 3.0, and 2.6 and 3.6, 3.0, and 3.1, respectively. It indicates that our two approaches learn better policy than CQ-learning and MTGA in game NJU.

In Figure 8, similar results can be observed in NTU. Due to the layout difference, agents in NJU are more likely to collide with each other than in NTU though it possesses fewer agents. For instance, in NTU, agent 1 only needs to coordinate with agent 2 and ignores agent 3 and agent 4. Thus an interesting point that should be noted is that the performance of MTGA in NTU drops dramatically compared to that in NJU, while DDL and DDL-NI in NTU perform better than those in NJU. The final ARPS values achieved by MTGA and DDL in NTU for the four agents are 0.3, 0.5, 0.3, and 0.5 and 7.0, 3.8, 5.0, and 3.7, respectively. The dispersion of final ARPS for MTGA and DDL in NTU is obviously much bigger than that in NJU. The main reason is that MTGA augments more redundant states than DDL and DDL-NI. MTGA constructs an overall joint coordination for all those influenced agents in each state but ignores which states of other agents for a specific agent should be augmented. Thus it augments almost all the joint states in the nondeterministic environment, which costs great computation and leads to more moving steps in each episode. Through our sample grouping mechanism, agent 1 only considers coordination with agent 2 and ignores influences of agent 3 and agent 4 to a certain extent, which reduce the learning space to improve final APRS effectively.

Table 2: The average runtime of the tested approaches in games TTG, ISR, CMU, NJU, and NTU.

Algorithm	TTG	ISR	CMU	NJU	NTU
CQ-learning	291.82 s	195.90 s	21795.21 s	24316.7 s	32520.60 s
MTGA	58.70 s	40.14 s	513.75 s	322.91 s	338.28 s
DDL	45.26 s	58.13 s	235.79 s	84.01 s	101.16 s
DDL-NI	68.71 s	54.93 s	2206.20 s	94.42 s	200.27 s

Table 2 shows the average runtime results of the tested algorithm during a run. Due to the space limitations, we only show the average runtimes of CQ-learning, MTGA, DDL, and DDL-NI in games TTG, ISR, CMU, NJU, and NTU. We can see that, in general, DDL and DDL-NI are faster than CQ-learning and MTGA, especially in games with large size of state space. In small games, MTGA is sometimes faster than DDL and DDL-NI because the process of computing model difference degree needs more time than similarities calculation in MTGA. For instance, in game ISR, runtimes of MTGA, DDL, and DDL-NI are 40.14 s, 58.13 s, and 54.93 s,

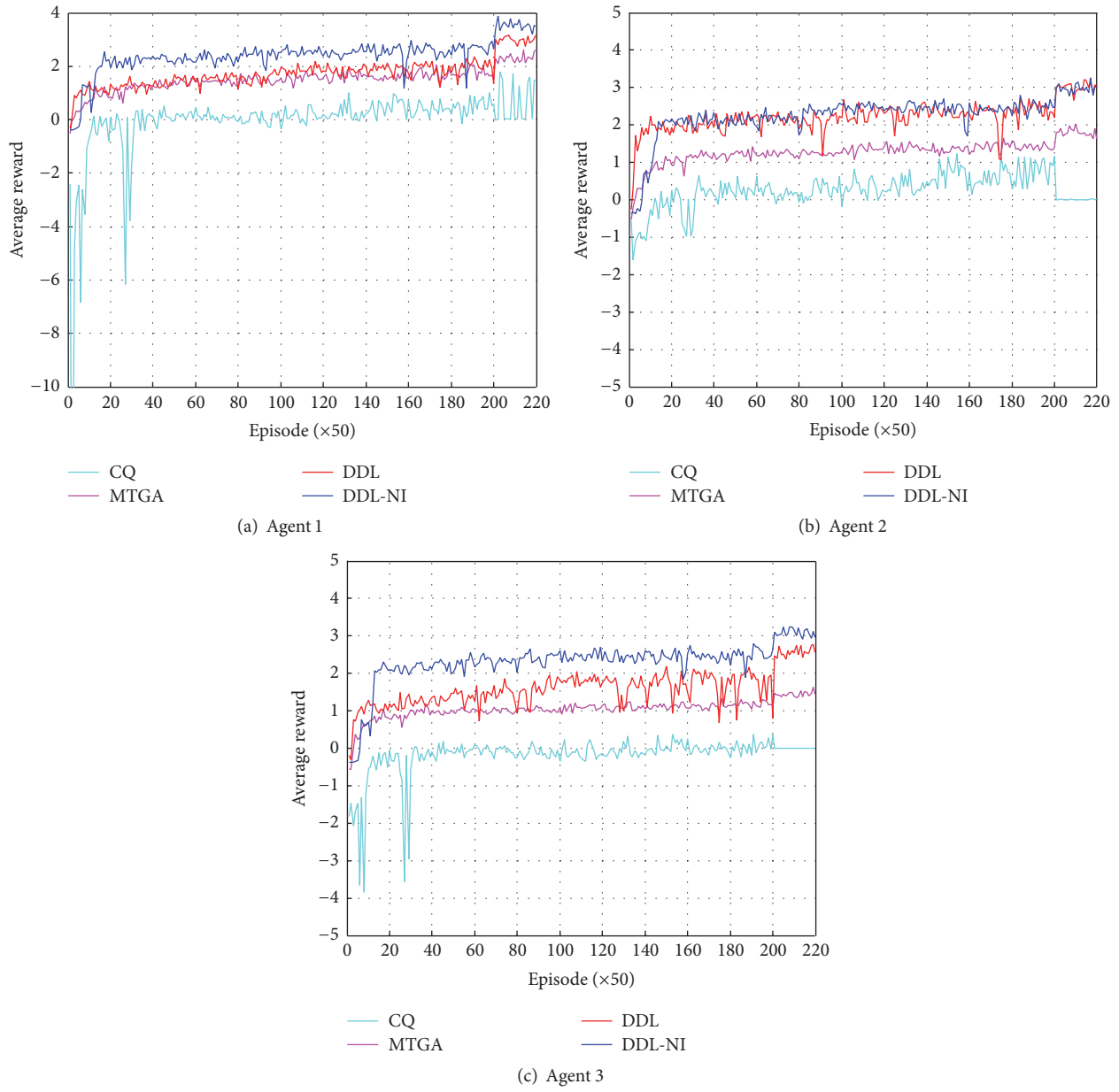


Figure 7: The learning curves of the tested approaches in game NJU.

respectively. With more redundant coordinated states being expanded in larger games like CMU, NJU, and NTU, runtimes taken in MTGA for each episode are obviously longer than those of our approaches, which are 513.75 s, 322.91 s, and 338.28 s, respectively. For CQ-learning, although extra time is not needed in coordinated states identification beforehand, it augments coordinated states continually along with rewards being collected during learning, which leads to slow convergence. Thus, CQ-learning is the slowest. Moreover, in large nondeterministic games like CMU and NTU, CQ-learning cannot learn convergent policies within the allowed time limit. In comparison, DDL is the fastest because of the accurate coordinated states identification and knowledge transfer. For DDL-NI with less accurate individual knowledge collected through limited sample times, it takes more time in games CMU and NTU.

5. Conclusion

This paper proposes a modified coordinated learning approach for MASs with sparse interactions. The approach enables agents to learn effective coordinated policy through sample grouping and evaluating the model difference degree of environmental dynamics. The grouped samples help an agent to identify not only whether it should coordinate with others to avoid confliction in current local state, but also which states of other agents should be considered to avoid miscoordination. The modified model difference degree

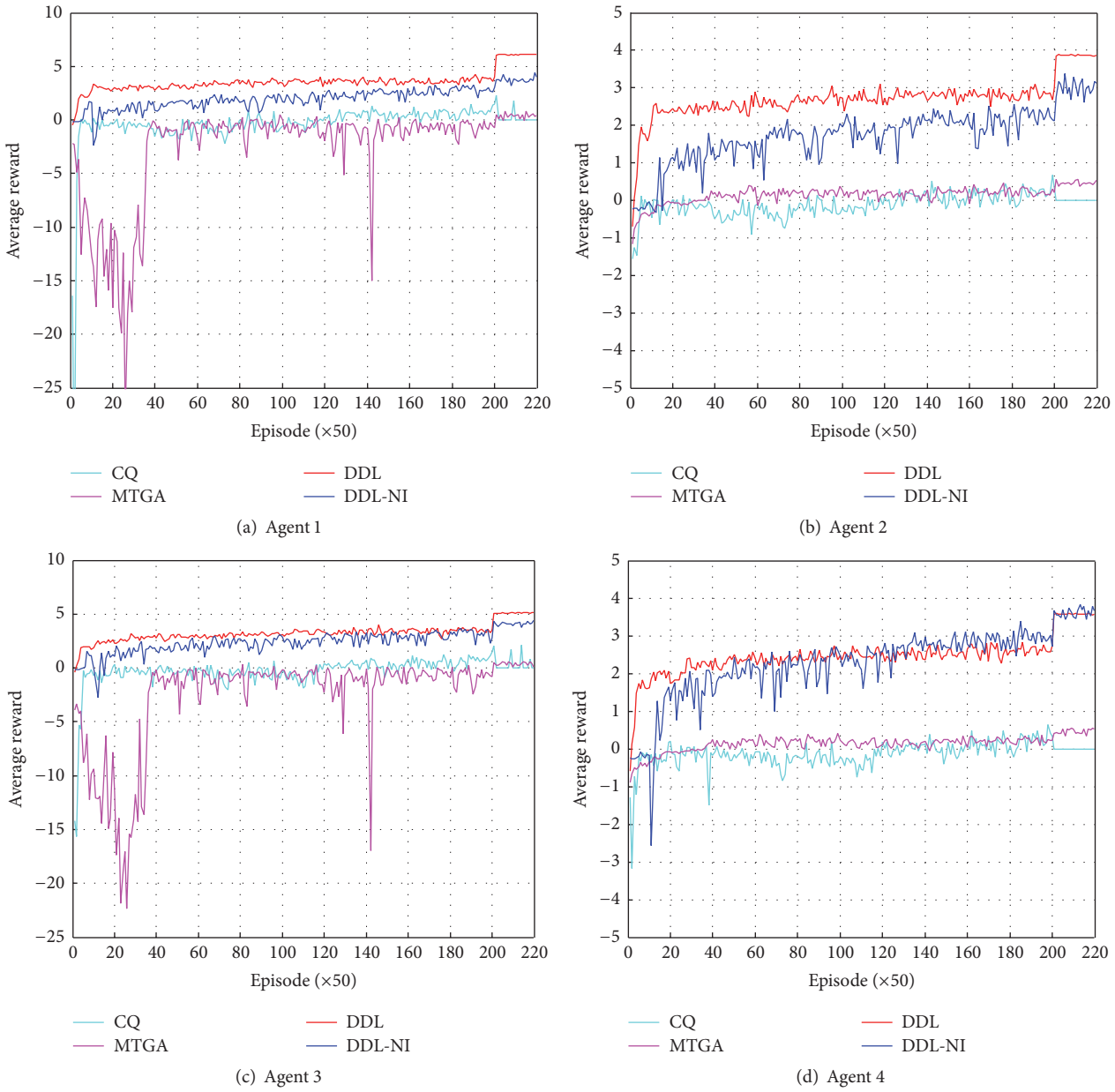


Figure 8: The learning curves of the tested approaches in game NTU.

makes full use of changes of reward and state transition to improve the learned policy. Moreover, our approaches require neither prior knowledge about domain structures (e.g., dependencies or coordinated states predefined) nor assumptions about agents (e.g., prior individual optimal policy). These features make our research apart from most existing approaches and render it in a broader way as a technique to solve practical applications. Experimental results show that the proposed approach improves the learned policy in various nondeterministic environment compared to existing algorithms, like CQ-learning and MTGA. Furthermore, it adapts to some broader scenarios which existing methods cannot deal with.

In our future work, we will investigate the problems with incomplete or inaccurate observation information about multiagent environment in real world, which may be studied based on the POMDP model [10]. Another interesting direction is the reward shaping for specified application of our approach in the wider context of computer games [25]. We will aim at improving the adaptation and efficiency of coordinated learning in complex multiplayer games by introducing model abstraction and reward shaping.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this manuscript.

Acknowledgments

The authors would like to acknowledge the support for this work from the National Science Foundation of China (Grants 61473300, 61573369, and 61403402) and the helpful discussions and suggestions with Xiaocheng Liu, Wei Duan, and Yong Peng.

References

[1] L. Buşoniu, R. Babuška, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 38, no. 2, pp. 156–172, 2008.

[2] E. Yang and D. Gu, "Multiagent reinforcement learning for multi-robot systems: a survey," Tech. Rep., 2004.

[3] C. V. Goldman and S. Zilberstein, "Decentralized control of cooperative systems: categorization and complexity analysis," *Journal of Artificial Intelligence Research*, vol. 22, pp. 143–174, 2004.

[4] A. Galstyan, K. Czajkowski, and K. Lerman, "Resource allocation in the grid with learning agents," *Journal of Grid Computing*, vol. 3, no. 1-2, pp. 91–100, 2005.

[5] I. Szita, "Reinforcement learning in games," in *Reinforcement Learning*, pp. 539–577, Springer, Berlin, Germany, 2012.

[6] J. Duan, N. E. Gough, and Q. H. Mehdi, "Multi-agent reinforcement learning for computer game agents," in *Proceedings of the 3rd International Conference on Intelligent Games and Simulation (GAME-ON '02)*, pp. 104–109, The University of Wolverhampton, London, UK, November 2002.

[7] L. Panait and S. Luke, "Cooperative multi-agent learning: the state of the art," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 3, pp. 387–434, 2005.

[8] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Proceedings of the 11th International Conference on Machine Learning*, vol. 157, pp. 157–163, 1994.

[9] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge (TARK '96)*, pp. 195–210, Morgan Kaufmann, San Francisco, Calif, USA, 1996.

[10] B. Banerjee, J. Lyle, L. Kraemer, and R. Yellamraju, "Sample bounded distributed reinforcement learning for decentralized POMDPs," in *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, pp. 1256–1262, July 2012.

[11] A. Nowé, P. Vrancx, and Y. M. De Hauwere, "Game theory and multi-agent reinforcement learning," in *Reinforcement Learning*, pp. 441–470, Springer, Berlin, Germany, 2012.

[12] F. S. Melo and M. Veloso, "Learning of coordination: exploiting sparse interactions in multiagent systems," in *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '09)*, pp. 773–780, International Foundation for Autonomous Agents and Multiagent Systems, Budapest, Hungary, 2009.

[13] J. R. Kok and N. Vlassis, "Sparse cooperative Q-learning," in *Proceedings of the ACM 21st International Conference on Machine Learning (ICML '04)*, p. 61, 2004.

[14] J. R. Kok, E. J. Hoen, B. Bakker, and N. Vlassis, "Utile coordination: learning interdependencies among cooperative agents," in *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, pp. 29–36, Colchester, UK, 2005.

[15] Y. M. De Hauwere, P. Vrancx, and A. Nowé, "Learning multi-agent state space representations," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, vol. 1, pp. 715–722, International Foundation for Autonomous Agents and Multiagent Systems, Toronto, Canada, 2010.

[16] M. Ghavamzadeh, S. Mahadevan, and R. Makar, "Hierarchical multi-agent reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 2, pp. 197–229, 2006.

[17] C. Yu, M. Zhang, F. Ren, and G. Tan, "Multiagent learning of coordination in loosely coupled multiagent systems," *IEEE Transactions on Cybernetics*, vol. 45, no. 12, pp. 2853–2867, 2015.

[18] Y. Hu, Y. Gao, and B. An, "Learning in multi-agent systems with sparse interactions by knowledge transfer and game abstraction," in *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*, pp. 753–761, International Foundation for Autonomous Agents and Multiagent Systems, Istanbul, Turkey, May 2015.

[19] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 2011.

[20] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, New York, NY, USA, 2014.

[21] Y. Gao, S. F. Chen, and X. Lu, "Research on reinforcement learning technology: a review," *Acta Automatica Sinica*, vol. 30, no. 1, pp. 86–100, 2004.

[22] J. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games," *The Journal of Machine Learning Research*, vol. 4, no. 6, pp. 1039–1069, 2004.

[23] Y. M. De Hauwere, P. Vrancx, and A. Nowé, "Detecting and solving future multi-agent interactions," in *Proceedings of the AAMAS Workshop on Adaptive and Learning Agents*, pp. 45–52, Taipei, Taiwan, 2011.

[24] N. Ferns, P. Panangaden, and D. Precup, "Metrics for finite Markov decision processes," in *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 162–169, AUAI Press, Banff, Canada, July 2004.

[25] S. Devlin and D. Kudenko, "Plan-based reward shaping for multi-agent reinforcement learning," in *Proceedings of the AAMAS Workshop on Adaptive and Learning Agents (ALA '12)*, 2012.