Faster Convergence to Cooperative Policy by Autonomous Detection of Interference States in Multiagent Reinforcement Learning

Sachiyo Arai(✉) and Haichi Xu

Faculty of Engineering, Chiba University,
P1-33 Yayoi-cho, Inage-ku, Chiba 263-8522, Japan
arai@tu.chiba-u.ac.jp

Abstract. In this paper, we propose a method for ameliorating the state-space explosion that can occur in the context of multiagent reinforcement learning. In our method, an agent considers other agents' states only when they interfere with each other in attaining their goals. Our idea is that the initial state-space of each agent does not include information about other spaces. Agents then automatically expand their state-space if they detect interference states. We adopt the information theory measure of entropy to detect the interference states for which agents should consider the state information of other agents. We demonstrate the advantage of our method with respect to the efficiency of global convergence.

Keywords: Multiagent system · Reinforcement learning · Conflict resolution

1 Introduction

In general, multiagent systems are applied to large and complex problems. It is often difficult to design the behavior rules of each agent beforehand, so they are expected to learn adaptively and autonomously. Multiagent Reinforcement Learning (MARL) is an effective approach for this design problem, and has attracted the attention of many researchers. In many existing MARL studies, it is assumed that the state of the environment is observable, including the selected actions and states of all other agents [3–7]. This method does not scale well, because the state-space that agents must learn is usually exponential with respect to the number of agents. In addition, the solution will often be sub-optimal or instable because of the agents' simultaneous and independent learning [16].

To solve these problems, recent research has proposed the observation of only a limited part of the other agents' state-spaces [9,11]. However, the detection of which part of the state-space should be considered relies on a heuristic approach, which requires different knowledge for different problems. In this paper, we propose a novel approach to detect the states in which extra information from

R. Booth and M.-L. Zhang (Eds.): PRICAI 2016, LNAI 9810, pp. 16–29, 2016.
DOI: 10.1007/978-3-319-42911-3_2

other agents is necessary. Our method adopts the information theory measure of entropy to reduce the state-space properly, thus achieving good performance in both learning speed and solution quality.

The remainder of this paper is organized as follows. In Sect. 2, we give the basic background information needed to understand our approach. In Sect. 3, we explain the class of problems, introduce some related work in this domain, and explain the main differences with our approach. Section 4 presents the proposed method. Experimental results are presented in Sect. 5, before our conclusions and ideas for future work are given in Sect. 6.

2 Glossary

In this section, we review the Markov Decision Process framework, Q-learning, Markov Game framework, and multiagent Q-learning, which are the basic concepts behind our proposed method.

2.1 Markov Decision Processes and Q-learning

Markov Decision Processes (MDPs) provide a theoretical framework for single-agent decision making, and are the basis on which reinforcement learning is built. An MDP can be described as a tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} >$, where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a set of actions available to the agent, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the transition function that describes the probability $P(s'|s, a)$ of ending up in state s', and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a reward function that returns the reward $R(s, a)$ for taking action a in state s.

An agent's policy is defined as a mapping $\pi : \mathcal{S} \rightarrow \mathcal{A}$. The objective is to find the optimal policy π^* that maximizes the expected discounted future reward $U^*(s) = \max_\pi E[\sum_{t=0}^{\infty} \gamma^t R(s_t)|\pi, s_0 = s]$ for each state s. The expectation operator $E[\cdot]$ is averaged over the reward and stochastic transitions, and $\gamma \in [0, 1)$ is the discount factor. This objective can also be expressed using Q-values, which store the expected discounted future reward for each state s and action a:

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q^*(s', a') \tag{1}$$

The optimal policy for a state s is the argument of $\max_a Q^{*(s',a')}$ that maximizes the expected future discounted reward. Watkins [2] described a Q-learning algorithm to iteratively approximate Q^*. Q-learning starts from some initial estimate $Q(s, a)$ for each state-action pair. When an exploration action a is taken in state s, the reward $R(s, a)$ is received and the next state s' is observed. The Q-values are updated according to the following update rule:

$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha(R(s, a) + \gamma V(s')) \tag{2}$$

$$V(s') \leftarrow \max_{a'} Q(s', a') \tag{3}$$

where $\alpha \in (0, 1)$ is an appropriate learning rate. Under certain conditions, Q-learning is known to converge to the optimal $Q^*(s, a)$ [2].

2.2 Markov Games and Multiagent Q-learning

Markov Games (MGs) [3] multiagent decision making. An MG can be described by a tuple $< \mathcal{N}, \mathcal{S}, \mathcal{A}_1, \ldots, \mathcal{A}_n, \mathcal{P}, \mathcal{R}_1, \ldots, \mathcal{R}_n >$, where $\mathcal{N}$ is the set of $(n = |\mathcal{N}|)$ agents, $\mathcal{S}$ is the finite set of states, and $\mathcal{A}_i (i \in 1, \ldots, n)$ is the set of actions available to agent i. The transition function $\mathcal{P} : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \times \mathcal{S} \to [0, 1]$ represents the probability $P(s'|s, a)$ that the state will transit from state s to s' after performing the joint action $a \in \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$, and $\mathcal{R}_i : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \to \mathbb{R}$ is the reward function that returns the reward $R_i(s, a)$ for agent i after joint action a is taken in state s. Note that, when $n = |\mathcal{N}| = 1$, the MG is equivalent to an MDP.

Table 1 classifies MGs by the relationship between $\mathcal{R}_i$, and summarizes their existing multiagent Q-learning algorithms. In these algorithms, it is necessary for an agent to observe the state-space consisting of its own state and those of other agents at all times.

Table 1. Markov Games and existing algorithms

Reward	Classification	Algorithms
$R_1 + R_2 = 0$	Zero-sum MGs	Minimax Q-learning [3]
$\Sigma_{k=1}^{n} R_k = Const.$	General-sum MGs	Nash Q-learning [6], Correlated-Q [7]
$R_1 = \cdots = R_n = R$	Team MGs	Team-Q [4], OAL [5]

3 Problem Domain

In the existing MARL research discussed in Sect. 2.2, the states of other agents must be observed at all times. However, as mentioned in Sect. 1, such approaches do not scale well, as the state-space often becomes exponential in the number of agents. In addition, the experimental results of Busoniu et al. [1] showed that agents tend to learn slowly, and the solution will often be sub-optimal due to this explosion in the state-space.

In reality, it is difficult to observe the states of other agents at all times [8]. Further, in general, as the multiagent system relies upon the sparse interaction between agents, there should be no need to observe the complete state of the other agents. If complete observation is necessary, we can achieve a control system via the central management of a super-agent. Therefore, in a multiagent system with sparse interaction between agents, recognizing which state-spaces should be considered becomes an important issue.

3.1 Definition of Terminology

In this paper, we divide the state-space of the agent into **collision states**, **interference states**, and **non-interference states**. In collision states and

interference states, agents interact with each other. There is no interaction among agents in the non-interference state. The set of **collision states** is defined as the set in which agents may be present in the same state transition simultaneously. The set of **interference states** is defined as the set of states prior to the transition to collision states. In addition, we call the actions such that agents transit from interference states to collision states. A conceptual diagram of the collision states and interference states is shown in Fig. 1.

Consider the state transition shown in Fig. 1. Here, the observation of agent$i$ is s^i, and that of other agents'is $\boldsymbol{s}^{-i}$. Because the destination state of the transition from interference states is dependent on the action of other agents, different rewards will be given depending on the transition to a collision state or non-interference state. These are marked by r_{ic}, r_{in}, respectively, in Fig. 1. An agent recognizes a transition to the same state, and the action value in the interference state will be updated by the different rewards. Therefore, incorrect Q-values will propagate through the interference states and their previous states, and lead to uncertain policies in these states. Thus, we should observe the state of other agents in interference states, but not necessarily in the other states. Recently, research has shown that it is possible to detect these interference states. In this paper, we propose a novel method for detecting the interference states, and develop a learning algorithm for different types of states.

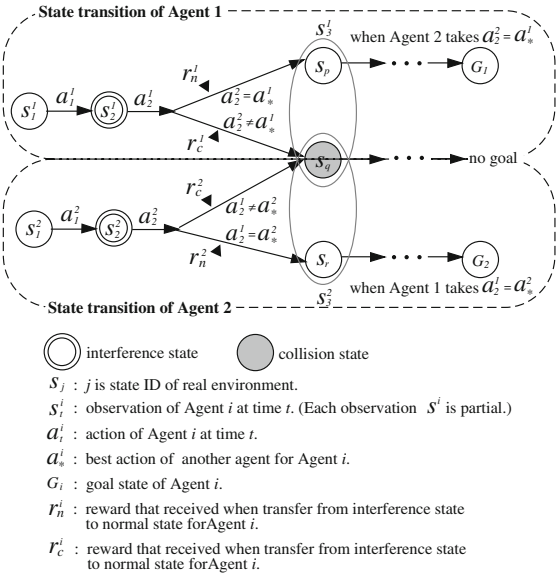


Fig. 1. Image of interference states and collision states

3.2 Related Work

In this section, we present a brief overview of the most relevant existing work in this problem domain, and describe the differences, as well as some similarities, between our proposed method and these approaches.

Kok et al. [9] described an approach in which agents know which states are interference states beforehand, allowing the successful reduction of state-spaces. In [10], Kok et al. use Coordination Graphs (CGs) to represent the relationship between agents in interference states and interference actions. They propose an approach for learning the CGs during the learning process, which enables the detection of interference states and reduction of state-spaces. However, this approach is limited to Team MGs.

Melo et al. introduced interaction-driven MGs, which contain a set of states in which interaction occurs between agents. The list of interference states was then applied to the agents beforehand [11]. In their later work, an algorithm was proposed in which agents learn to detect the interference states autonomously, rather than being given them [12]. In other words, agents detect the states in which interaction is necessary via a learning process. To achieve this, the action space of each agent is augmented with a pseudo-coordination action that performs an active perception step. In this perception step, agents observe the states of other agents, and determine whether the other agents' states should be considered. Because the penalty for collision is bigger than the cost of this active perception, the agents learn to take this action in interference states. However, determining the cost of using active perception is a challenge, because it is necessary to consider the penalty of collision comprehensively.

De Hauwere et al. succeeded in detecting the interference states using a Generalized Learning Automaton (GLA) [13]. The GLA receives the Manhattan distance between two agents as input, and the agents can then learn which are the interference states. In addition, De Hauwere et al. introduced an effective statistical method that focuses on the reward sequence given in each state for detecting the interference states [14,15]. Although similar to the approach of [10], this method further reduces the state-space. However, although this approach is effective when considering immediate rewards, it is not applicable under a delayed-reward environment.

Busoniu et al. [8] proposed a method for detecting interference states by focusing on the differences in Q-value convergence between the interference states and non-interference states. Their approach does not consider rewards, unlike [14,15]. However, to analyze the convergence of the Q-values, many parameters must be set appropriately, which is a challenging task.

In this paper, we propose an approach in which agents learn to detect interference states autonomously during the learning process, but have no knowledge of the interference states beforehand. Our approach focuses on the fluctuation of the Q-values, which is similar to the approach of [8] and different from those of [10,14,15]. Our approach formulates the variation of the Q-values as the entropy of information theory, and detects the interference states by observing fluctuations in the entropy.

4 Proposed Method

4.1 Entropy Based Approach

In information theory, entropy is a measure of the uncertainty in a random variable. The information entropy is defined as $H(S) = -\sum_{i=1}^{m} p_i \log p_i$, where $\{p_1, p_2, \ldots, p_m\}$ is the probability distribution of random variable S on a discrete set with m elements. We developed a measure based on information theory [16] for evaluating the degree of interaction in a multiagent system during the learning process. In this paper, we detect the interference states by adopting the information theory measure of entropy, in accordance with this quantification. The quantification of the uncertainty of a specific policy is summarized as follows.

Consider the state set $\mathcal{S} = \{s_1, \ldots, s_h, \ldots, s_m\}$, let the action set that can be selected by an agent be $\mathcal{A} = \{a_1, \ldots, a_i, \ldots, a_n\}$, and assume that the policy is $\pi(\mathcal{S}, \mathcal{A})$. In this context, the policy in state s_h is then $\sum_{i=1}^{n} \pi(s_h, a_i) = 1$. According to the definition of information entropy, the entropy of policy π in state s_h can be calculated by Eq. (4). In the following, $H(\pi(s_h, \mathcal{A}))$ is abbreviated to $H(s_h)$ for simplification.

$$H(\pi(s_h, \mathcal{A})) = -\sum_{i=1}^{n} \pi(s_h, a_i) \log \pi(s_h, a_i) \tag{4}$$

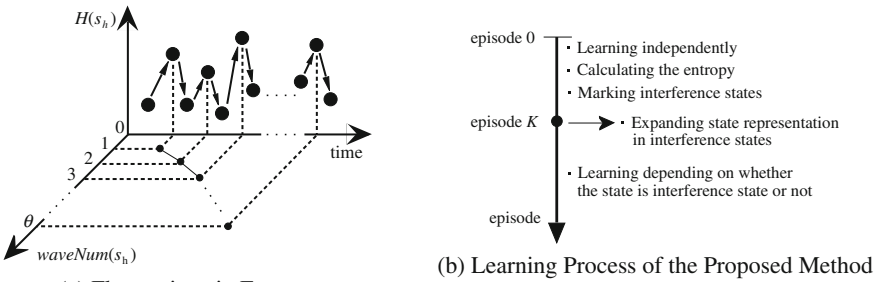
4.2 Detection of Interference States by Entropy

In terms of the probability distribution of policy $\pi(s_h, \mathcal{A})$, the entropy in state s_h has the following properties.

- if $\pi(s_h, a_1) = \cdots = \pi(s_h, a_i) = \cdots = \pi(s_h, a_n)$, $H(s_h)$ becomes the maximum value
- if $\exists a_i \in \mathcal{A}$, $\pi(s_h, a_i) = 1$, $H(s_h)$=0.

Therefore, during the learning process, if the policy convergences to a "deterministic policy," then $H(s_h) = 0$. However, $H(s_h)$ does not decrease monotonically. In the early learning stages, to avoid convergence to a local solution, action a in which $Q(s, a)$ is maximized may not be selected because of the exploration process. In addition, in the multiagent environment, the influence of simultaneous learning between agents means that the magnitude relation between the value of $Q(s, a)$ varies frequently, and the optimal action also changes.

Further, as mentioned above, with the progress of reinforcement learning under single agent MDPs, the policy converges to a "deterministic policy," and $H(s_h)$ should become 0. In MGs, however, more than one action may be effective, and the Q-values of these actions will be similar. Hence, the policy might convergence to a "deterministic policy" even when the learning is at an advanced stage. This is because the agent cannot observe the internal states of other agents in the multiagent environment if they learn independently. In this case, although the entropy is not 0, the fluctuation of entropy is eliminated. On the other hand, the entropy of a policy in the interference states does not tend to 0, and the



(a) Fluctuations in Entropy

(b) Learning Process of the Proposed Method

Fig. 2. Basic idea to find interferences

value fluctuates frequently. In the non-interference states, the entropy of a policy may fluctuate in the early learning process due to the randomness of the action selection, but it should become 0, or at least remain stable.

Based on the description above, the entropy of a policy in the interference states fluctuates frequently for long periods of time during the learning process, but this phenomenon does not occur in other states. As shown in Fig. 2(a), the transition to the collision state from the interference state varies because of the influence of other agents' actions. In particular, the higher the state-value in the posterior state of the collision state, the more the entropy of the policy in the interference state prior to the collision state fluctuates. This is because of the increase during the transition to the collision state. Therefore, to detect interference states, we examine the frequency of the entropy fluctuation. That is, as shown in Fig. 2(a), we aggregate the increase or decrease in entropy over a certain learning time. If this exceeds a predetermined threshold, the state is determined to be an interference state. Here, we denote the increase or decrease in the entropy of a policy in state $s_h \in \mathcal{S}$ as $waveNum(s_h)$, and the predetermined threshold value as θ. Equation (4) calculates the entropy of agent$i$'s in state $s_h \in \mathcal{S}$ using Q values. Here, τ is a parameter for the exploration.

$$\pi(s_h, a_i) = \frac{e^{Q(s_h,a_i)/\tau}}{\sum_{b\in\mathcal{A}} e^{Q(s_h,b)/\tau}} \tag{5}$$

4.3 Algorithm

An outline of the proposed method is shown in Fig. 2(b). The proposed learning algorithm is shown in Figs. 3 and 5. Note that the superscripts i and $-i$ in Figs. 3 and 5 refer to agent i and the other agents, respectively.

First, the set of interference states is initialized to the empty set $\mathcal{S}_o^i = \emptyset$. Before the K-th episode, agents learn independently by single-agent Q-learning without observing the other agents' states, and we calculate $H(s^i)$, which is the entropy of a policy in each state s^i. When the entropy $H(s^i)$ increases or decreases, $waveNum(s^i)$ is updated as $waveNum(s^i) \leftarrow waveNum(s^i) + 1$.

Initialize Q^i and Q^i_o
Set $episode = 0$, $\mathcal{S}^i_o = \emptyset$, and θ
Set waveNum(s^i)=0 for $\forall s^i \in \mathcal{S}^i$
while $2 \leq episode < K$ **do**
  Repeat for steps:
    Observe s^i
    Choose a^i from Q^i
    Observe $r^i, s^{i\prime}$
    Update $Q^i(s^i, a^i) \leftarrow (1-\alpha)Q^i(s^i, a^i) +$
      $\alpha(r^i + \gamma \max_{a^i \in \mathcal{A}^i} Q(s^{i\prime}, a^i))$
  Calculate $H(s^i)_{episode}$
  if $H(s^i)_{episode-1} > H(s^i)_{episode-2}$ and
    $H(s^i)_{episode-1} < H(s^i)_{episode}$ **then**
      $waveNum(s^i) \leftarrow waveNum(s^i) + 1$
  end if
  if $waveNum(s^i) \geq \theta$
      Mark s^i as an interference state and
      add s^i to $\mathcal{S}^i_o$
  end if
  $episode \leftarrow episode + 1$
end while
if $episode = K$ **then**
  Expand Q-table in interference state
    $Q^i_o(s^i, \boldsymbol{s}^{-i}, a^i) \leftarrow Q^i(s^i, a^i)$
    for $\forall s^i \in \mathcal{S}^i_o, \boldsymbol{s}^{-i} \in \mathcal{S}^{-i}, a^i \in \mathcal{A}$
end if

Fig. 3. Agent Algorithm before K-th Episode in the Detection Phase of Interference States

When $waveNum(s_h)$ exceeds the threshold value θ, s_h is marked as an interference state and added to the set of interference states $\mathcal{S}^i_o$.

At the end of episode K, to take advantage of the learning results up to that point, we use Eq. (6) to initialize the Q-value in the interference states.

$$\begin{aligned} Q^i_o(s^i, s^{-i}, a^i) &\leftarrow Q^i(s^i, a^i) \\ \forall s^i \in \mathcal{S}^i_o, s^{-i} &\in \mathcal{S}^{-i}, a^i \in \mathcal{A}^i \end{aligned} \tag{6}$$

where $Q^i(s^i, a^i)$ denotes the Q-table learned by episode K, and $Q^i_o(s^i, s^{-i}, a^i)$ represents the combination of interference state s^i and the state of other agents s^{-i}. In Fig. 4, we show an example of this extension of the Q-table. The states of agent 2 are added to the state representation of agent 1 in states 4 and 6, because states 4 and 6 are detected as interference states. By this extension of the Q-table, we are able to reduce the state-space.

After episode K, the process proceeds as follows. When an agent selects an action and updates the Q-values, it checks whether its current state is an interference state. If not, it will use $Q^i(s^i, a^i)$ to select an action, and the rule

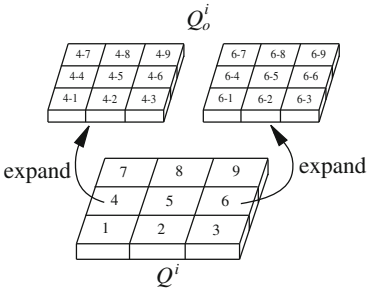


Fig. 4. Expanding the Q-table in the Interference States

Repeat for episodes:
  Observe s^i
  if $s^i \notin \mathcal{S}_o^i$ **then** //s^i is not an interference state
    Choose a^i from Q^i
    Observe $r^i, s^{i\prime}$
    Update $Q^i(s^i, a^i) \leftarrow (1-\alpha)Q^i(s^i, a^i)+$
      $\alpha(r^i + \gamma \max_{a^i \in \mathcal{A}^i} Q(s^{i\prime}, a^i))$
  else //s^i is an interference state
    Observe s^{-i}
    Choose a^i from Q_o^i
    Observe $r^i, s^{i\prime}$
    Update $Q_o^i(s^i, \boldsymbol{s}^{-i}, a^i) \leftarrow (1-\alpha)Q_o^i(s^i, \boldsymbol{s}^{-i}, a^i)+$
      $\alpha(r^i + \gamma \max_{a^i \in \mathcal{A}^i} Q(s^{i\prime}, a^i))$
  end if

Fig. 5. Agent Algorithm after $(K+1)$-th Episode for Updating Q-values

represented as Eq. (7) will be used to update the Q-values.

$$Q^i(s^i, a^i) = (1-\alpha)Q^i(s^i, a^i) + \alpha(r^i + \gamma \max_{a^i \in \mathcal{A}^i} Q(s^{i\prime}, a^i)) \tag{7}$$

On the other hand, if the agent is currently in an interference state, it will select an action by $Q_o^i(s^i, s^{-i}, a^i)$, and the rule represented as Eq. (8) will be used to update the Q-values.

$$Q_o^i(s^i, s^{-i}, a^i) = (1-\alpha)Q_o^i(s^i, s^{-i}, a^i) + \alpha(r^i + \gamma \max_{a^i \in \mathcal{A}^i} Q(s^{i\prime}, a^i)) \tag{8}$$

Also note that when the value of $Q(s^{i\prime}, a^i)$ in Eqs. (7) and (8), is calculated, $s^{i\prime}$ is considered as a non-interference states. This is because each agent iterates a cycle of observation, action, then transit a next state, the agent cannot identify

whether new (transited) state is a non-interference state or not, when agent updates $Q^i(s^i, a^i)$. In other words, the Q value of the transited state, is a previous value of expansion.

5 Experiments

5.1 Experimental Environment and Settings

Our experimental environment is shown in the left of Fig. 6. It is a maze environment containing two agents, and their start states and goal states are denoted by S_1, S_2, and G_1, G_2, respectively. The black blocks represent walls that agents cannot pass through. The agents' task is to find the shortest path to their goals. However, the task is not completed when only one agent reaches its goal.

The action set available to the agents is $<$ UP, DOWN, LEFT, RIGHT, STOP $>$. These cause the agent to move simultaneously one cell up, down, left, or right, or to stop. The transitions are deterministic. Initially, each agent is placed in their start state. An episode is defined as the period from the start time of the initial state to the time at which both agents have reached their goal states. Once an agent reaches its goal state, it remains there. When the length of an episode exceeds some upper limit, the episode is terminated.

Each agent receives a reward of $R_i = 500$ when they complete the task, and a reward of -50 when they collide with each other. When an agent collides with a wall, it receives a reward of -20. In all other cases, the reward is 0. Note that, when an agent collides with another agent or a wall, it returns to the state in the previous time step. This settings of reward correspond to the Markov Game situation as mentioned in Sect. 2.2.

Agents learn for 400 episodes, and the upper limit of time steps in an episode is set to 500. We use an ϵ-greedy action selection strategy, where ϵ is set to 0.2 in the first 300 episodes and $\epsilon = 0$ from then on. The learning rate is set to $\alpha = 0.3$, the discount factor is set to $\gamma = 0.9$, and the initial Q-values are set to 0.1. We set the parameters related to the detection of interference states as $K = 50$ and the threshold $\theta = 2$.

5.2 Experimental Results

Acquired Behavior: First, in the right of Fig. 6, we show which states were detected as the interference states by the agents. We can see that agents have correctly learned to detect the states in which collisions are frequent or most likely. Two typical examples of acquired behaviors are shown in Fig. 7(a), and Fig. 8(a) where the red and blue colored lines indicate the behaviors of agent1's and agent2's, respectively. It is found that the both behaviors are globally optimal.

Fluctuation of Entropy in Interference States. Figure 7(a) shows that agent2 wait around $(4, 4)$ to avoid collision while Fig. 8(a) shows that agent1 wait around $(4, 2)$ to avoid collision. The fluctuation of entropy of both situations are

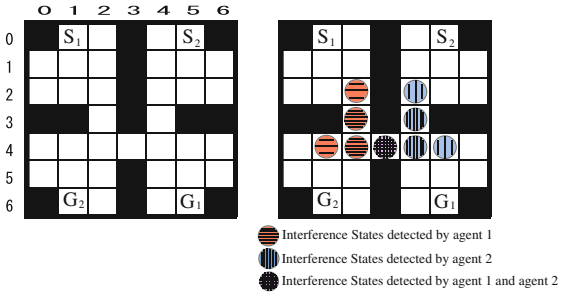
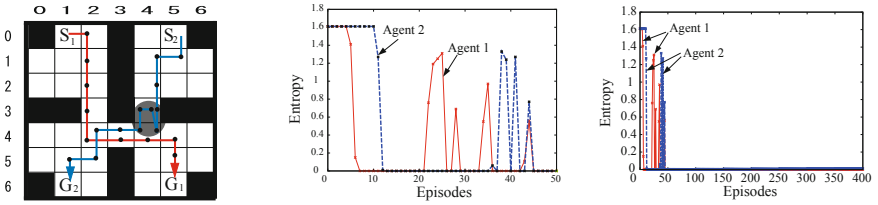
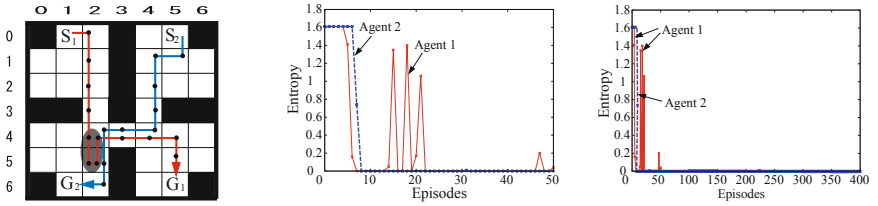


Fig. 6. Detected Interference States. Sparse lines with circles indicate the interference states with a low frequency of detection. (Color figure online)



(a) Pattern 1 (b) Before 50 Episodes (c) All 400 Episodes

Fig. 7. Pattern1: Finally Obtained Path. (The red and blue lines indicate the path of agent 1 and agent 2 respectively.), Fluctuations of Entropy in State (4, 2) of Agent 1 (Color figure online)



(a) Pattern 2 (b) Before 50 Episodes (c) All 400 Episodes

Fig. 8. Pattern2: Finally Obtained Path. (The red and blue lines indicate the path of agent 1 and agent 2 respectively.), Fluctuations of Entropy in State (4, 4) of Agent 2 (Color figure online)

shown in Figs. 7(b) and 8(b). The vertical axis shows the value of entropy of each state, and the horizontal axis records the number of episodes. Figures 7(c) and 8 show the convergence of learning after 400 episodes.

5.3 Evaluation of Learning Efficiency

Next, to evaluate the performance of the proposed method, we compared it with two other multiagent Q-learning methods. One is **Independent learners**, which learn without any information about the state of other agents, and the second is **Joint-state learners**, which receive the joint location of the agents as state information, but choose their actions independently.

Figure 9 shows the number of steps taken to complete each episode, averaged over 10 runs. The vertical axis shows the number of time steps needed to complete one episode, and the horizontal axis records the number of episodes. We also show the size of the state-space, the converged steps, and the required steps to acquire the optimal/best solution in each method in Table 2.

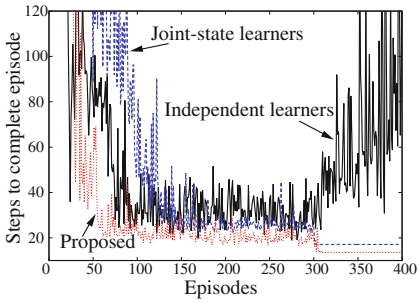


Fig. 9. Experiment on the Efficiency of the Proposed Method, Compared with the Independent Learners and Joint-state Learners, Averaged over 10 Runs.

Table 2. Comparison of learning performance. Average and s.d 10 series of experiments

Algorithm	State space ± s.d	Steps ± s.d	Required Episodes to 17 steps ± s.d
Independent learners	35	not converge	not converge
Joint-state learners	1225	17.1±1.22	164.90±45.66
Proposed method	100.8±20.4	13.5±0.92	79.4±33.26

Both the Independent learners and our proposed method learn quickly in early episodes (<= 50) compared with the Joint-state learners, because learning is based on a smaller state-space. However, the Independent learners do not converge to a stable policy, but instead oscillate because the policies in the interference states are uncertain. Our proposed method converges to a stable policy, because the agents observe the states of the other agents after the 50th episode. The Joint-state learners were consistently able to converge to a stable policy, but learned slowly due to the large size of the state-space. Our proposed

method learns quickly and converges to a better stable policy than the Joint-state learners, because of the reduction in the state-space brought about by detecting the interference states.

6 Conclusions

In this paper, we presented an approach for reducing the state-space in the MARL domain. An overly large state-space usually causes agents to learn slowly and reach sub-optimal solutions. We defined the states in which agents should consider the influence of other agents' actions as interference states. Our proposed method is made up of two phases: one for detecting the interference states, and another for learning different types of states.

We noted that if the states of other agents are ignored when they should be considered, agents' policies became uncertain. The interference states were detected by calculating the degree of this uncertainty via the entropy of information theory. We reduced the state-space by limiting the observations of other agents' states, thus improving the learning speed. In addition, it was possible to avoid the incomplete perception caused by the actions of agents, which led to an improvement in the optimality of the solution.

However, as shown in Eq. (6) and Fig. 4 in Sect. 4.3, by expanding the Q-table in the interference states, all possible states of other agents are considered comprehensively. In fact, we can speculate that the influence of other agents' states on a given agent's own actions only arise in a part therein. In future work, we will focus on finding a method to achieve a more compact state-space representation, whereby an agent expands its own state information using that of other agents. Furthermore, we need to investigate methods to set a suitable threshold value for the number of entropy fluctuations. This was set by preliminary experiments in our current implementation.

References

1. Busoniu, L., De Schutter, B., Babuška, R.: A comprehensive survey of multiagent reinforcement learning. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **38**(2), 156–172 (2008)
2. Watkins, C., Dayan, P.: Q-learning. Mach. Learn. **8**(3), 279–292 (1992)
3. Littman, M.: Markov games as a framework for multi-agent reinforcement learning. In: Proceedings of the Eleventh International Conference on Machine Learning, pp. 242–250 (1994)
4. Littman, M.: Value-function reinforcement learning in Markov games. Cogn. Syst. Res. **2**(1), 55–66 (2001)
5. Wang, X., Sandholm, T.: Reinforcement learning to play an optimal Nash equilibrium in team Markov games. In: Advances in Neural Information Processing Systems vol. 15, pp. 1571–1578 (2002)
6. Hu, J., Wellman, M.: Nash Q-learning for general-sum stochastic games. J. Mach. Learn. Res. **4**, 1039–1069 (2003)

7. Greenwald, A., Zinkevich, M., Kaelbling, P.: Correlated Q-learning. In: Proceedings of the Twentieth International Conference on Machine Learning, pp. 242–249 (2003)
8. Busoniu, L., De Schutter, B., Babuška, R.: Multiagent reinforcement learning with adaptive state focus. In: Proceedings of the Seventeenth Belgian-Dutch Conference on Artificial Intelligence, pp. 35–42 (2005)
9. Kok, J.R., Vlassis, N.: Sparse cooperative Q-learning. In: Proceedings of the Twenty-First International Conference on Machine Learning, pp. 61–68 (2004)
10. Kok, J.R., Hoen, P., Bakker, B., Vlassis, N.: Utile coordination: learning interdependencies among cooperative agents. In: Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG), pp. 29–36 (2005)
11. Spaan, M.T.J., Melo, F.S.: Interaction-driven Markov games for decentralized multiagent planning under uncertainty. In: Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems, pp. 525–532 (2008)
12. Melo, F.S., Veloso, M.: Learning of coordination: exploiting sparse interactions in multiagent systems. In: Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems, pp. 773–780 (2009)
13. De Hauwere, Y., Vrancx, P., Nowé, A.: Learning what to observe in multi-agent systems. In: Proceedings of the Twentieth Belgian-Dutch Conference on Artificial Intelligence, pp. 83–90 (2009)
14. De Hauwere, Y., Vrancx, P., Nowé, A.: Learning multi-agent state space representations. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems, pp. 715–722 (2010)
15. De Hauwere, Y., Vrancx, P., Nowé, A.: Adaptive state representations for multiagent reinforcement learning. In: Proceedings of the 3rd International Conference on Agents and Artificial Intelligence, pp. 181–189 (2011)
16. Arai, S., Ishigaki, Y.: Information theoretic approach for measuring interaction in multiagent domain. J. Adv. Comput. Intell. Intell. Inform. **13**(6), 649–657 (2009)

http://www.springer.com/978-3-319-42910-6

PRICAI 2016: Trends in Artificial Intelligence
14th Pacific Rim International Conference on Artificial Intelligence, Phuket, Thailand, August 22-26, 2016, Proceedings
Booth, R.; Zhang, M.-L. (Eds.)
2016, XXII, 821 p. 252 illus., Softcover
ISBN: 978-3-319-42910-6