Reward Value-Based Goal Selection for Agents' Cooperative Route Learning Without Communication in Reward and Goal Dynamism

Fumito Uwano¹ · Keiki Takadama²

Received: 28 February 2020 / Accepted: 8 May 2020 / Published online: 23 May 2020
© The Author(s) 2020

Abstract

This paper proposes a goal selection method to operate agents get maximum reward values per time by noncommunicative learning. In particular, that method aims to enable agents to cooperate along to dynamism of reward values and goal locations. Adaptation against to these dynamisms can enable agents to learn cooperative actions along to changing transportation tasks and changing incomes/rewards because of transporting tasks for heavy/valuable and light/valueless items in a storehouse. Concretely, this paper extends the previous noncommunicative cooperative action learning method (Profit minimizing reinforcement learning with oblivion of memory: PMRL-OM) and sets the two unified conditions combined of the number of time steps and the rewards. One of the unified conditions is calculated the approximated number of time steps if the expected reward values are the same each other for all purposes, and the other is the minimum number of time steps divided by the reward value. The proposed method makes all agents learn to achieve the purposes in the order in which they have the minimum number of the condition values. After that, each agent learns cooperative policy by PMRL-OM as the previous method. This paper analyzes the unified conditions and derives that the condition calculating the approximated time steps can be combined both evaluations with almost same weight unlike the value the other condition, that is, the condition can help the agents to select the appropriate purposes among them with the small difference in terms of the two evaluations. This paper tests empirically the performances of PMRL-OM with the two conditions by comparing with the PMRL-OM in three cases of grid world problems whose goal locations and reward values are changed dynamically. The results of this derive that the unified conditions perform better than PMRL-OM without some conditions in grid world problems. In particular, it is clear that the condition calculating the approximated time step can direct the appropriate goals for the agents.

Keywords Multi-agent system · Reinforcement learning · No communication · Reward design · Intrinsic motivation

Introduction

Multi-agent system (MAS) simulates real-world problems by several number of agents which have the own roles in the simulated environment, e.g., robots in a storehouse, traffic lights in an intersection and more [2, 4]. To control the agents cooperatively, each agent learns the own action selection policy by reinforcement learning (RL). Generally,

for cooperation, the most important factor is communication among agents. As for the related works, modeling other agent based on the communicated information becomes important [1]. These techniques make an agent model other agent, predict its action and purpose, and act by the cooperative policy based on the action and the purpose. For example, in storehouse robot problem, the simulated robots (agents) communicate to receive the information of the others' operation and plan the own routes to carry luggage to some points without collisions each other. Although these techniques achieve the agents' complex cooperation, much information is required. This suggests that the cooperation cannot be achieved in some situations, that is, non- or less-communication like field robotics or many agents. In addition, noncommunicative cooperation by reinforcement learning is challenging and important topic. For example, Raileanu et al. proposed cooperative action learning method

✉ Fumito Uwano
uwano@cs.okayama-u.ac.jp
Keiki Takadama
keiki@inf.uec.ac.jp

¹ Okayama University, 3-1-1, Tsushima-naka, Kita-ku, Okayama, Japan

² The University of Electro-Communications, 1-5-1, Chofugaoka, Chofu-shi, Tokyo, Japan

by modeling the opponent agent (self-other modeling: SOM). Zemzem and Tagina's method (Cooperative multi-agent learning approach based on the most recently modified transitions: CMRL-MRMT) makes agents learn cooperative action by copying the learning result of the advantageous agent [5, 12]. These methods utilize the information like the agents' states, actions, learning results, and more to achieve the cooperation. However, both methods cannot cooperate without communication.

On the other hand, some researchers aim to propose cooperative learning methods without communication. However, these methods are not complete, that is, they have to be under some conditions. For example, though Arai and Katia mentioned the noncommunicative method, the agent can observe other agent's state [6], and Shiraishi et al. extended that method, but the condition is remain required [7]. Uwano et al. add some direction to learning agents to cooperate each other without communication (Profit minimizing reinforcement learning: PMRL) [10], and they extended PMRL to perform in the dynamic environment (Profit minimizing reinforcement learning with oblivion of memory: PMRL-OM). However, PMRL-OM has two conditions for cooperation: the number of the goals equals to that of the agents and reward values in all goals are the same each other. These are very weak points because the agents' goals might be changed (e.g., changing transportation tasks in a storehouse) and importance (i.e., reward) of reaching a goal cannot be the same among all purposes (e.g., incomes/rewards of transporting tasks for heavy/valuable and light/valueless items are not the same each other.) Therefore, it is important to propose the cooperative-learning method without communication for the dynamism of the number of the goals and the reward values.

To tackle this issue, this paper proposes the two conditions unifying the two local information to select the appropriate purposes and proposes the noncommunicative MARL method based on the unified conditions to adapt the dynamic change of the rewards and the purposes. Concretely, one of the unified conditions is calculated the approximated number of time steps if the expected reward values are the same each other for all purposes, and the other is the minimum number of time steps divided by the reward value. The proposed method makes all agents learn to achieve the purposes in the order in which they have the minimum number of the condition values. After that, each agent learns cooperative policy by PMRL-OM as the previous method. Concluding the above, our contributions are shown as follows:

- Appropriate operation for agents with two purposes (e.g., storehouse robots advance both quantity and time to carry luggage) is established
- We designed new evaluation for agent's behavior to carry more valuable luggage with smaller time.

- The parameter can control to weight the evaluation whether the value or the time.

The remainder of this paper is organized in the following manner. “**Problem and Approach**” section explains a maze problem and the goal of this problem in this work. “**Baseline Methods**” section explains Q-learning and PMRL-OM, and the unified condition is explained and analyzed mathematically in “**Proposed Mechanism**” section. “**Experiment**” section describes the experiment and analyzes the obtained results, and finally, our conclusions are summarized in “**Conclusion**” section.

Problem and Approach

Problem

We employ route planning tasks in mazes whose number of goals and reward values change dynamically. Equation (1) shows the joint action A from actions of all agents $a_1 - a_n$ (n is the number of the agents.) The environment is translated by Eq. (2) from an arbitrary state S and the joint action A . This equation indicates that the joint action can always translate the current state to the next state (When the agents reach the goal, they cannot observe other states by any actions, that is, the goal states are absorbing states.) At the end, Eq. (3) shows a reward function which indicates a value of reward and the condition to get that, where r is the reward value, s_i and s_{-i} are the current states of the agents i and the other agents, and S_{goal} is a set of all goal states. This equation shows the agent acquires the reward when its observing state s_i is the goal state and the other agents have not observed that state.

$$A = a_1 \times a_2 \times \dots \times a_n \quad (1)$$

$$\tau : S \times A \times S \rightarrow 1 \quad (2)$$

$$R_i = \{r | s_i, s_{-i} \in S_{\text{goal}} \wedge s_i \neq s_{-i}\} \quad (3)$$

Figure 1 shows an example of grid world problems. The squares named “Agent A”, “Agent B”, “Goal X”, and “Goal

Fig. 1 Example of grid world

Y indicate the start states for the agents A and B and the goal states, that is, the states of the agents A and B s_A, s_B are the start states at the beginning of learning, and the goals X and Y are the goal states in the goal state set S_{goal} . The agents observe the own states s_A and s_B , select the actions a_A and a_B , and learn the own action by the acquired reward by the reward function (after that, the agents observe the next state translated by the state transition function, and repeat this cycle.)

In addition, this paper focuses on the dynamics for the environmental change as follows:

- Reward value change
- Goal number change

These dynamics influence the reward function and the state transition function for the goal state [Eq. (3) and S_{goal}]. Concretely, the value r is changed to other value, and the new states are inserted into the goal state set S_{goal} . Note that the environmental change happened once in a trial, and the proposed method aims to learn cooperative policy along to the environmental change.

Motivation and Approach

Our approach sets the discounted rewards according to Manhattan distance from the goals. From this approach, our motivation is that the reward, evaluation for learning, does not become the valuable of the result of the agent's behavior, but we mount information of distance into the reward. For example, Fig. 2 shows our approach and motivation for storehouse robots. In this figure, two robots carry some luggage to two shelves and the situation changes from left to right (i.e., a new shelf appears, the shelves move to other places and their capacities change.) The previous methods focus on the capacity as the reward value, but do not on the distance: Learning result can lead agents to reach a nearest goal. If the number of shelves is the same as that of agents, there is no problem. But actually the numbers are not the same each other like Fig. 2. In this situation, two robots have

to select two shelves among three, the previous methods select in order from only the capacity (reward). However, if only five pieces of luggage exist, do you think the robots need to store some of those to the farthest shelf in right side of Fig. 2? Our motivation is to solve this problem, and the proposed method estimates the new reward combined the reward and the distance (written “unified condition” in this paper). For example in Fig. 2, after the environmental change, if the shelf lost the luggage then the reward value increases along to the lost number, if the shelf moves far then the reward value decreases along to the distance, and if the shelf moves near then the reward value increases.

Baseline Methods

Q-Learning

Reinforcement learning (RL) [8] is a try-and-error method which aims at maximizing an acquired reward per a unit time. As its general framework, an RL agent interacts with an environment: it observes a state from the environment, selects an action, receives a reward from the environment as the result of that action and then learns from the reward. Note that this cycle from the observation to the next observation is called “step” in this paper. Among many RL methods, Q-learning [11] is a very popular RL method for a single-agent task. A Q-learning agent estimates state–action values (called Q -value) for the possible state–action pairs in the environment, i.e., the agent estimates an discounted expected reward that it will receive when its action a is executed in its state s . The agent learns to acquire a policy $\pi(s, a)$ to decide which action should be executed to maximize a received reward. Technically, the policy π can be composed of probabilities in selecting any action a in any state s and is calculated by $Q(s, a)$, $a \in A$ where $Q(s, a)$ is the Q -value when state and action become s and a , and A is a set of actions that includes the possible actions. To maximize a received reward, $Q(s, a)$ is updated as follows.

Fig. 2 Our motivation and approach

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r(s) + \gamma \max_{a' \in A} Q(s', a') - Q(s, a)], \quad (4)$$

where s' is the next state when a is executed in s , a' is the next action executed in the state s' , $r(s)$ is the reward received from the environment, $\max_{a' \in A} Q(s', a')$ is the largest Q -value when executing the action $a' \in A$ in the state s' . In addition to these variables, α is the learning rate, while γ is the discount factor. Precisely, α is the real number from 0 to 1 which indicates the learning speed, while γ is the real number from 0 to 1 which indicates how much the future rewards should be considered as important. Note that the current state s and the current action a is for the general agent, that is, the state and the action for the agent A are $s(A)$ and $a(A)$ as shown in the previous section.

Profit Minimizing Reinforcement Learning with Oblivion of Memory (PMRL-OM)

To learn multi-agent cooperation without communication in the dynamic change environment, Uwano et al. proposed Profit minimizing reinforcement learning with oblivion of memory (PMRL-OM) [9]. PMRL-OM is based on Q-learning and forces the cooperation by controlling the reward for dynamic change environments. By controlling the rewards, PMRL-OM controls all agents to reach the goals with the largest time steps among the goals they can receive the rewards. In the grid world problem like Fig. 1, all agents reach the farthest goals which they can receive the rewards. This is basic idea, that is, the advanced agent reaches the difficult goal to yield the easy goal to the other agents and the multi-agent system behaves cooperatively. The methods employed the internal reward and the goal value to realize the basic idea, and the next section explains those mechanisms.

Internal Reward

The internal reward is the reward inside of the agent, and the agent learns based on the internal reward. PMRL-OM estimates the internal reward for the agent to reach the appropriate goal with the largest goal value. In this section, we

assume the appropriate goal is the goal g and explain how to set the internal reward. The internal reward is calculated using Eq. (5) once every iteration. ir_g is the internal reward of the goal g , γ is the discount rate of Q-learning, g' is the certain goal among the goal set G which includes all goals, and $r_{g'}$ is the external reward in the goal g' . t_g and $t_{g'}$ are the minimum number of the steps until the agent has reached the goals g and g' . Note that the internal reward is calculated for only the appropriate goal g and those of the other goals are set by the external rewards. Equation (5) calculates discounted expected rewards as Q -values of actions to go to all goals in the agent's start and adds a constant value δ to the largest discount expected reward in the internal reward ir_g to reach the goal g .

$$ir_g = \max_{g' \in G, g' \neq g} r_{g'} \gamma^{t_{g'} - t_g} + \delta, \quad (5)$$

Goal Value

The goal value is the value for the agents to reach the farthest goals, and this value is converged to the minimum number of steps. The agent learns to reach the goal with the maximum goal value using the internal reward. Equation (6) is the goal value update functions. The goal value is updated once every iteration. t_g is the minimum number of the steps, and ξ is a constant value. Note that ξ is a positive integer value over than 0, and ξ indicates how much the current minimum number of steps are emphasized in the goal value. if ξ is small, the only present minimum numbers of steps are emphasized, and the more ξ is, the more past minimum numbers of steps are emphasized in the goal value, either. If the agent has received the reward, the goal value is updated by the top function; otherwise, it is updated by the bottom function. From these equations, all agents aim to reach the farthest goals among the goals can be reached by them.

$$\begin{cases} \text{bid}_g = \frac{\xi-1}{\xi} \text{bid}_g + \frac{t_g}{\xi} & \text{if some rewards are received} \\ \text{bid}_g = \frac{\xi-1}{\xi} \text{bid}_g + \frac{0}{\xi} & \text{otherwise} \end{cases} \quad (6)$$

Algorithm

Algorithm 1 Algorithm of PMRL-OM

Require: $t = \text{Array}([])$

```

1.  $g'$  is initialized by a random value
2. for  $iteration = 0$  to  $MaxIteration$  do
3.   while Not on the goal or  $step < MaxStep$  do
4.     State observation
5.     Action selection
6.     Receiving a reward  $r$ 
7.     for  $goal \in G$  do
8.       if  $goal = g'$  and the current state is in the goal  $g'$  then
9.          $ir_{goal} = \max_{g'' \in G, g'' \neq goal} r_{g''} \gamma^{t[g''].min() - t[goal].min()} + \delta$ 
10.      else
11.         $ir_{goal} = r$ 
12.      end if
13.    end for
14.    Updating Q-value by the internal reward
15.     $step++$ 
16.  end while
17.  if On the goal  $g$  and current step is smaller than before then
18.     $t.Add(g, step)$ 
19.  else
20.     $t.Add(g, MaxStep)$ 
21.  end if
22.  if  $t.Length > Cycle$  then
23.     $t.erase(t.begin())$ 
24.  end if
25.  if Receiving the reward then
26.     $bid_g = \frac{\xi-1}{\xi} bid_g + \frac{\beta t[g].min()}{\xi}$ 
27.  else
28.     $bid_g = \frac{\xi-1}{\xi} bid_g + \frac{0}{\xi}$ 
29.  end if
30.  Selecting a new goal  $g'$  with the largest goal value
31. end for

```

Algorithm 1 is the algorithm of PMRL-OM. In this algorithm, t is the array, including the tuple with the goal number g and the minimum numbers of steps until the agent reaches the goal in each iteration. At first, the goal number for the agent to reach the appropriate goal is initialized by a random value (line 1). The agent observes its own state, selects the action, receives a reward, estimates the internal reward from the reward and updates the Q -value from the internal reward (lines 3–13). As for the internal reward estimation, if the agent is in the goal g' , the internal reward is set to reach that; otherwise, the internal reward is set by the reward. The agent repeats this cycle for several times and stops this cycle if it has reached the goal or several steps are spent (lines 3 and 16). Thereafter, if the agent on the arbitrary goal g and the number of the current step is smaller than before, it stores this step into t (lines 17 and 18); otherwise, it stores the maximum number of steps into t (lines 19–21). After this process, if the length of t is over the threshold $Cycle$, the agent erases the first element in t (lines 22–24). At the

end, the agent updates the goal value bid_g (lines 25–29) and selects the new goal g' with the largest goal value (line 30). Since this process indicates the flow of PMRL-OM in one iteration, the agent goes to the next iteration and repeats the process. Note that $t[g]$ indicates the set of all steps for the goal g in the array t .

Proposed Mechanism

Standardized Step and Simple Condition

Under the environment with the reward change, the agents cannot determine the appropriate goal by the goal value because the goal value is calculated for the spent step to be the smallest. This improvement unifies the two conditions, the minimum number of steps and the reward, using Eq. (7). This value should be smaller for the better evaluation. In this equation, t_g is the minimum number of the steps for the

Fig. 3 Graphs of both conditions (left: standardized step, right: simple condition)

goal g , and t'_g is the standardized steps as the unified condition. r_g and r_{stan} are the reward of the goal g and the reward as the standard value, respectively. γ is the discount rate of Q-learning. Equation (7) approximates the minimum number of steps when the discounted expected reward is r_{stan} . Note that t'_g can be the positive and negative real value. This equation can be condition to help the agent to select the appropriate goals for cooperation. Concretely, the agents learn cooperative policy for the several goals from the smallest standardized step to the n -th smallest that among all goals, and the agents can obtain maximum gain per time.

$$t'_g = t_g - \log_{\gamma} \frac{r_{\text{stan}}}{r_g}, \quad (7)$$

where $t_g = \min t[g]$

In addition, this paper proposes another unified condition more simply shown in Eq. 8. This value should be smaller for the better evaluation, too. The condition value is calculated that the minimum number of steps is divided by the reward value, that is, the minimum number of steps should be smaller and the reward value should be larger.

$$C = \frac{t_g}{r_g} \quad (8)$$

Mathematical Analysis

To show the difference of the two conditions, the standardized step and the simple condition, we compare the standardized step with the simple condition. Figure 3 shows the behaviors for both conditions when the reward and the minimum number of steps are from 0 to 50. The x - and the y -axes indicate the reward and the number of steps, and the z -axis indicates the condition value. Both conditions show that when the smaller reward and the larger step of the situation are worse and vice versa. However, the gradients are different to each other, the surface of the standardized step

Fig. 4 $\frac{1}{r_g \ln \gamma}$ (where $r_g = 10$ and γ is 0.01 to 1)

Fig. 5 $\frac{1}{r_g \ln \gamma}$

is gentle, while that of the simple condition is steep. Equations (9) and (10) show the total derivative values of both conditions.

$$dt'_g = \frac{1}{r_g \ln \gamma} dr_g + dt_g \quad (9)$$

$$dC = -\frac{t_g}{r_g^2} dr_g + \frac{1}{r_g} dt_g \quad (10)$$

In Eq. (9), since r_{stan} is disappear in the derivative, r_{stan} can be set some values greater than 0. In addition, the smaller reward can emphasize the worse condition, while the larger reward does not emphasize the better condition. As for the minimum number of steps, the gradient is linear and the smaller number is better. This suggests that when the rewards are very large the standardized step evaluates only the minimum number of steps. On the other hand, in Eq. (10), the smaller reward very emphasizes the worse condition and the gradient of the minimum number of steps becomes smaller along to the smaller reward in this condition. In addition, the smaller step makes the gradient of the reward be gentle. This suggests that when the rewards are small, the simple condition evaluates the smaller minimum number of steps is better. From the above analysis, the standardized step is good condition to evaluate without some bias for the minimum number of steps and the reward as the unified condition.

In addition, the gradient can be changed by the parameter γ for the standardized step, unlike the simple condition. Figure 4 shows the gradient $\frac{1}{r_g \ln \gamma}$ whose r_g is 10. The gradient decreases the influence of the reward: since the reward value is easily larger than the number of the steps, this calculation decreases the difference of the range of the reward and the step. Furthermore, Fig. 5 shows the gradient $\frac{1}{r_g \ln \gamma}$ where the reward r_g is 0 to 50 and the parameter γ is 0.01 to 1. If the parameter γ is less than 0.5, the standardized step is influenced by only the step. On the other hand, if that is greater than or equal to 0.5, the reward and the step influences the standardized step with the minus and plus direction, respectively: it is clear that the larger reward and the smaller step are better. The reward value and the number of the step are changed by the environment, e.g., the large step in the large field. However, from Fig. 5, the gradient is 0 for almost all reward values, that is, if the agent has to select one among two goals with large rewards, considers only the numbers of the steps. The standardized step makes agents' learning priorities mainly in order from the largest reward, and for the

number of the step if the reward value is small. In this paper, γ is 0.9. Since the gradient $\frac{1}{r_g \ln \gamma}$ is approximately -1 by this γ , the gradients of the standardized step, reward and step, are balanced like $dt'_g = -dr_g + dt_g$. Therefore, $\gamma = 0.9$ is suitable for integrating the reward and the step.

Algorithm

Algorithm 2 shows the algorithm of PMRL-OM with the standardized step. Note that, for the simple condition, the equation should be changed to that of the simple condition in the process of calculating the unified condition value of the algorithm. In this algorithm, t is the array, including the tuple with the goal number g and the minimum numbers of steps until the agent reaches the goal in each iteration. At first, the goal number for the agent to reach the appropriate goal is initialized by a random value and the array of the standardized steps st is initialized by *MaxStep* (lines 1 and 2). The agent observes its own state, selects the action, receives a reward, estimates the internal reward from the reward and updates the Q -value from the internal reward (lines 5–15). As for the internal reward estimation, if the agent is in the goal g' , the internal reward is set to reach that; otherwise, the internal reward is set by the reward. The agent repeats this cycle for several times and stops this cycle if it has reached the goal or several steps are spent (lines 4 and 17). Thereafter, if the agent on the arbitrary goal g and the number of the current step are smaller than before, it stores this step into t (lines 18 and 19); otherwise, it stores the maximum number of steps into t (lines 20–22). After this process, if the length of t is over the threshold *Cycle*, the agent erases the first element in t (lines 23–25), and the agent updates the goal value bid_g (lines 26–30). After that, the agent calculates the standardized steps (lines 31–33). At the end, the agent selects the new goal g' with the largest goal value among the goals from the smallest st to the *AgentNumber*-th smallest st (line 34). Since this process indicates the flow of PMRL-OM in one iteration, the agent goes to the next iteration and repeats the process. Note that $t[g]$ indicates the set of all steps for the goal g in the array t .

Fig. 6 Grid world problems in both cases

Fig. 7 Eight grid world problems in Case (3) **Fig. 8** Result of Case (1) **Fig. 9** Result of Case (2)

Algorithm 2 Algorithm of the improved PMRL-OM

Require: $t = \text{Array}()$

1. g' is initialized by a random value
2. Standardized step st is initialized by $MaxStep$
3. **for** $iteration = 0$ to $MaxIteration$ **do**
4. **while** Not on the goal or $step < MaxStep$ **do**
5. State observation
6. Action selection
7. Receiving a reward r
8. **for** $goal \in G$ **do**
9. **if** $goal = g'$ and the current state is in the goal g' **then**
10. $ir_{goal} = \max_{g'' \in G, g'' \neq goal} r_{g''} \gamma^{t[g''].min()} - t[goal].min() + \delta$
11. **else**
12. $ir_{goal} = r$
13. **end if**
14. **end for**
15. Updating Q-value by the internal reward
16. $step++$
17. **end while**
18. **if** On the goal g and current step is smaller than before **then**
19. $t.Add(g, step)$
20. **else**
21. $t.Add(g, MaxStep)$
22. **end if**
23. **if** $t.Length > Cycle$ **then**
24. $t.erase(t.begin())$
25. **end if**
26. **if** Receiving the reward **then**
27. $bid_g = \frac{\xi-1}{\xi} bid_g + \frac{\beta t[g].min()}{\xi}$
28. **else**
29. $bid_g = \frac{\xi-1}{\xi} bid_g + \frac{0}{\xi}$
30. **end if**
31. **for** $goal \in G$ **do**
32. $st[goal] = t.min() - \log_{\gamma} \frac{r_{stan}}{r_g}$
33. **end for**
34. Selecting a new goal g' with the largest goal value among the goals from the smallest st to the $AgentNumber$ -th smallest st
35. **end for**

Table 1 Parameters

Iterations	50,000
Maximum steps	100
Initial Q -value	0
Learning rate α	0.1
Discount rate γ	0.9
Constant δ	10
Memory range $Cycle$	5000
Constant ξ	500
Discount rate β	0.9
Standard reward r_{stan}	10

Table 2 The evaluation values

Case	Optimal value	None	Standardized step	Simple condition
Case (1)	2.86, 4.00	0.10, 0.10	2.86, 4.00	2.86, 4.00
Case (2)	2.86, 7.14	0.10, 2.93	2.86, 7.14	2.86, 7.14

Experiment

Experimental Design and Setting

To investigate the effectiveness of the unified conditions in PMRL-OM, we verified that the PMRL-OM is applicable to dynamic changes in the grid world problems with the dynamic change of the rewards and the goals. We employ the grid worlds with the three cases as follows. The number of the goals and the goal positions are changed after a certain iteration in Case (1). The rewards in the goals are changed after a certain iteration in Case (2). Both changes exist in Case (3). We test the performance of the improved PMRL-OM by comparing the improved PMRL-OM with the standardized step and the simple condition with the normal PMRL-OM in each case. Concretely, Fig. 6 shows the grid world problems for both cases. The top and the bottom indicate the grid worlds for each case, and the left and the right sides indicate the grid worlds before and after the environmental change. The iteration 25000 is the borderline between two grid worlds in this experiment. In the grid world, the squares named A and B are the starts for the two agents (called the agents A and B), and the squares with some numbers are the goals and the numbers are the reward values. Since the reward values are not changed in Case (1), the improved PMRL-OM with the two conditions performs well, while the normal PMRL-OM performs bad. On the other hand, the grid worlds in Case (2) can be changed as the number of goals and the reward values, and the PMRL-OM with the standardized step performs the best. In Case (3), there are 8 kinds of grid worlds shown in Fig. 7. The borderlines are the iterations 15,000 and 30,000 in the grid world 2, and the borderline is the iteration 25000 in the other grid worlds.

- Case (1): the number of goals change
- Case (2): the rewards change
- Case (3): both changes exist

Table 3 The evaluation values in Case (3)

Grid world	None			Standardized step			Simple condition		
	Before	First	Second	Before	First	Second	Before	First	Second
Grid world 1	0.70	0.48		2.86	5.83		2.86	5.83	
Grid world 2	0.78	1.05	2.11	3.07	4.43	1.89	3.07	5.83	1.92
Grid world 3	0.86	6.05		2.86	5.46		2.86	5.46	
Grid world 4	0.15	4.42		2.65	3.94		4.20	3.92	
Grid world 5	2.94	3.98		3.44	4.47		3.45	3.86	
Grid world 6	0.30	0.81		12.1	0.79		12.1	0.86	
Grid world 7	5.69	7.41		8.39	5.07		11.6	6.42	
Grid world 8	0.63	3.80		2.74	8.44		2.74	8.48	

Table 4 The calculated condition values in Case (2)

	Before		After		
	Goal (6,1)	Goal (8,3)	Goal (5,2)	Goal (6,1)	Goal (8,3)
<i>Standard</i>					
Agent A	6	8	0.15	− 0.58	− 2.43
Agent B	4	6	− 1.85	− 2.58	− 4.43
<i>Simple</i>					
Agent A	6	8	0.27	0.30	0.27
Agent B	4	6	0.13	0.20	0.20

The evaluate criterion is Eq. (11). The total value of the rewards received by all agents is divided by the minimum number of steps when all agents have reached the goals in this equation, that is, this equation indicates how many gains all agents obtain per time.

$$\text{evaluate} = \frac{\sum_i r_i}{\max_i t_i} \quad (11)$$

The experiments were determined by the number of trials with 30 random seeds. Table 1 summarizes the parameters. The learning iterations and the steps are limited to 50,000 and 100 (lines 1 and 2). The initialized Q -value is 0, and the learning rate α and the discounted rate γ of Q-learning is 0.1 and 0.9, respectively (lines 3–5). The constant value δ in is 10 (line 6). For only parameters of the PMRL-OM, the memory range $Cycle$ is 5000, the constant ξ is 500, and the discount rate β is 0.9 (lines 7–9). At the end, the standard reward r_{stan} for the standardized step is 10 (line 10).

Experimental Result

Figures 8 and 9 show the results of each case. In each result in the figures, the vertical and the horizontal axes indicate the evaluation value of Eq. 11 and the number of iterations. The round, the square and the triangle marked lines are the

Fig. 10 Agent A's Q-table of standardized step in Case (2)

Fig. 15 Agent A's Q-table of normal PMRL-OM in Case (2)

Fig. 11 Agent B's Q-table of standardized step in Case (2)

Fig. 12 Agent A's Q-table of simple condition in Case (2)

Fig. 13 Agent B's Q-table of simple condition in Case (2)

Fig. 14 Agent A's Q-table of normal PMRL-OM in Case (2)

results of PMRL-OM, PMRL-OM with the standardized step and the simple condition, respectively. The vertical lines are the error bars. Note that this result is the median value

of the results with 30 seeds. The higher evaluation value is better in these results. As for Fig. 8, PMRL-OM performs worse than that with the two conditions and PMRL-OM with the conditions performs well. Since the error bars for the PMRL-OM with the conditions do not exist at the last half of the iterations, the agents can learn to reach the appropriate goals stably. On the other hand, in Fig. 9, PMRL-OM with the standardized step performs the best than only PMRL-OM and that with the simple condition. However, the results of all methods cannot converge to some values. Note that when we execute Brunner–Munzel test between the 30 results of PMRL-OM and that with the standardized steps, the p value is less than $2.2e-16$ in Case (1) and $2.749e-10$ in Case (2), that is, the results of PMRL-OM with the standardized step are better than those of PMRL-OM statistically.

Table 2 shows the optimal evaluation values and the median of the evaluation values of PMRL-OM and that with two conditions in the grid worlds before and after the dynamic change. The row indicates each case, while the columns indicate the kinds of case, the optimal evaluation values, the final evaluation values of PMRL-OM and that with the standardized step and the simple condition in order from the left. In the columns of the evaluation values, the left and the right values are the evaluation values before and after the environmental change, respectively. From this table, in Cases (1) and (2), the evaluation values of PMRL-OM with the standardized step are the same as the optimal values. On the other hand, the evaluation values of PMRL-OM with the simple condition are optimal in Case (1), but the evaluation values in Case (2) are optimal only before the dynamic change. As for PMRL-OM, the evaluation values are not optimal before and after the dynamic change in both cases.

Table 3 shows the evaluation values of PMRL-OM and that with the two conditions at the iteration just before the dynamic change in each grid world of Case (3). The row indicates each grid world, while the columns indicate the kinds of methods, PMRL-OM, that with the standardized step, and that with the simple condition. In each column, there are the evaluation values before the dynamic change, after the change and after the second change in order from the left. The bold numbers indicate the largest values. From this result, the evaluation values of the two conditions are

Table 5 The calculated condition values in the grid world 5

	Before				
	Goal (1,1)	Goal (3,1)	Goal (5,2)	Goal (7,1)	Goal (8,2)
<i>Standard</i>					
Agent A	8.58	8.85	1.15	8	− 2.433
Agent B	10.6	6.85	− 0.85	6.0	− 4.43
<i>Simple</i>					
Agent A	0.4	0.67	0.33	0.8	0.27
Agent B	0.8	0.33	0.2	0.6	0.2
	After				
	Goal (4,2)	Goal (4,3)	Goal (6,1)		
<i>Standard</i>					
Agent A	0.15	5.0	− 5.43		
Agent B	− 1.85	1.0	− 5.42		
<i>Simple</i>					
Agent A	0.27	0.5	0.17		
Agent B	0.13	0.1	0.17

Fig. 16 The averaged result on the grid world 2 in Case (3) **Fig. 17** The maximum result on the grid world 2 in Case (3)

better than those of PMRL-OM; in particular, those of the simple condition are the best.

Discussion

From the results, the conditions, the standardized step and the simple condition, perform better than those of PMRL-OM in the dynamic change environment of the goals and the rewards. We discuss the detailed performance of the conditions.

Effectiveness of the Two Conditions

PMRL-OM cannot adapt the dynamic change of the reward and the goal because it makes all agents aim to reach the

farthest goals, that is, the agents A and B reach the goal (6,1) and (8,3) in Cases (1) and (2), whereas the conditions limit the agents to reach some goals and lead them to the appropriate goals. In the grid world of Case (1), since all reward values are same with each other, the two conditions are calculated along to the minimum number of steps. On the other hand, in that of Case (2), Table 4 shows the condition values of the standardized step and the simple condition in the grid world of Case (2). The top table has the condition values of the standardized step, while the bottom table has the condition values of the simple condition. In each table, the left and the right sides indicate the condition values before and after the dynamic change. In each side, the rows and the columns indicate the agent and the goal, respectively. Note that the tuple after the word “Goal” shows the location, e.g., (x, y)

Fig. 18 Agent A's Q-table of standardized step on the grid world 2 in Case (3)

Fig. 19 Agent B's Q-table of standardized step on the grid world 2 in Case (3)

Fig. 20 The averaged result on the grid world 3 in Case (3)

indicates the x th row and y th column as the coordinate of the goal. The italic cell indicates the goals where the agent should reach by following the conditions. From this table, the standardized step can indicate the appropriate goals, while the simple condition cannot indicate like that, i.e., the agents A and B should reach the goals (6,1) and (8,3) after the dynamic change for the largest evaluation value. However, PMRL-OM can cover the mistake of the simple condition; concretely, the agent B can keep reaching the goal (8,3) to prevent from the agent A's reaching that goal. In this grid world, since the agent A has never reached the goal (8,3), the simple condition of the goal (8,3) becomes large (if the agent cannot the goal, the minimum number of steps for the goal is the maximum number of steps.) From the result of that, the agent A can reach the goals (5,2) and (6,1) with the

Fig. 21 The maximum result on the grid world 3 in Case (3)

Fig. 22 Agent A's Q-table of standardized step on the grid world 3 in Case (3)

Fig. 23 Agent B's Q-table of standardized step on the grid world 3 in Case (3)

Fig. 24 Agent A's Q-table of PMRL-OM on the grid world 3 in Case (3)

simple condition. The agents aim to reach the farthest goals, that is, the goals (6,1) and (8,3) for the agent A and B, and the evaluation values of the simple condition can be the best in Case (2). However, the standardized step is more suitable

Fig. 25 Agent B's Q-table of PMRL-OM on the grid world 3 in Case (3)

Fig. 26 The averaged result on the grid world 4 in Case (3)

Fig. 27 The maximum result on the grid world 4 in Case (3)

than the simple condition. In addition, the simple condition cannot indicate the difference among the goals like Fig. 3.

Figures 10, 11, 12, 13, 14 and 15 show the Q-table of each agent in Case (2). The top, the middle and the bottom figures indicate the Q-tables of the agents of PMRL-OM with the standardized step, that with the simple condition and original PMRL-OM, respectively. The left and the right sides are the Q-tables of the agents A and B, respectively. In each figure, the white, the light-colored and the dark-colored squares indicate the starts, the environment states and the

Fig. 28 Agent A's Q-table of standardized step on the grid world 4 in Case (3)

Fig. 29 Agent B's Q-table of standardized step on the grid world 4 in Case (3)

Fig. 30 Agent A's Q-table of PMRL-OM on the grid world 4 in Case (3)

Fig. 31 Agent B's Q-table of PMRL-OM on the grid world 4 in Case (3)

goals, respectively. The black arrows from the squares represent the actions, and the thickness of the arrows indicates how much the Q -value of the action. The number nearby the arrow shows the Q -value. From these figures, the agents A and B of PMRL-OM with the two conditions can estimate the appropriate Q -values to reach the goals (6,1) and (8,3), respectively. However, the agents A and B of PMRL-OM estimate the Q -values to reach the goal (8,3). This is bad because the two agents reach the same goal by following

the Q -value. From the figures, the above discussion is true and the two conditions can derive the appropriate Q -value.

Difference Between the Two Conditions

the different results of the standardized step and the simple condition in some grid worlds in Case (3), we analyze the difference of the performances of the conditions. Table 5 shows the standardized steps and the simple condition values of the grid world 5 in Case (3). The top and the bottom tables indicate the condition values before and after the dynamic change. In this grid world, the agents A and B should reach the goals (5,2) and (8,2) before the dynamic change, and they should reach the goals (4,2) and (6,1) after the dynamic change, respectively. From this table, the standardized step can indicate the appropriate goals to the agents, whereas

the simple condition cannot indicate that. This difference derives that the standardized step performs better than the simple condition in the grid world 5 in Table 3. From this result, it is clear that the standardized step can unify the two conditions. However, since there are many similar numbers of steps to reach the goals in Case (3), the simple condition sensitive for the reward value performs better than the standardized step.

Limit of the Two Conditions

Analyzing the result of the grid worlds in Tab. 3, we discuss the limit of the conditions.

In the grid world 2, the results of both conditions are under that of the previous method at the end. Figures 16

Fig. 32 The averaged result on the grid world 6 in Case (3)

Fig. 33 The maximum result on the grid world 6 in Case (3)

Table 6 The calculated condition values in the grid world 7

	Before						
	(1,1)	(2,3)	(4,3)	(5,2)	(6,3)	(7,3)	(8,3)
<i>Standard</i>							
Agent A	- 7.16	- 6.16	5	- 0.85	- 3.58	- 2.58	- 5.43
Agent B	- 6.16	- 5.16	6	0.15	- 2.58	- 3.58	- 6.43
<i>Simple</i>							
Agent A	0.15	0.18	0.5	0.2	0.15	0.2	0.17
Agent B	0.18	0.2	0.6	0.27	0.2	0.15	0.13
	After						
	(4,2)	(4,3)	(6,1)				
<i>Standard</i>							
Agent A	0.15	5.0	- 7.42				
Agent B	1.15	6.0	- 6.43				
<i>Simple</i>							
Agent A	0.5	0.27	0.1				
Agent B	0.6	0.33	0.13

Fig. 34 Agent A's Q-table of standardized step on the grid world 6 in Case (3)

Fig. 35 Agent B's Q-table of standardized step on the grid world 6 in Case (3)

they have to make agents learn more. Note that the results of the grid world 7 are the same as this case, but the condition calculations have a mistake. Table 6 shows both calculated conditions in the grid world 7. The agents have to cooperate among the goal located at (6,3) and (7,3) before the environmental change, and (4,2) and (6,1) after that. However, both conditions evaluate other goals. As for the results of that, the agents A and B do not reach the optimal goals with both conditions. However, the worst case is that the agents cooperate among the goal located at (4,3) and (5,2) before the environmental change, and (4,2) and (4,3) after that.

Since the result of the grid world 5 is wrote above, the analysis of this world is omitted. In the grid world 6, the results of the simple condition are the best, that of the unified condition are the best before the environmental change, and that of PMRL-OM are the worst. Figures 32 and 33 show the average and the maximum steps on the grid world 6 in Case (3), respectively. These results show the results of the three methods are the same each other. On the other hands, Figs. 34, 35, 36, 37, 38 and 39 show the Q-tables of the agents A and B with the seed 0 on the grid world 6 in Case (3) after the environmental change, respectively. The top, the middle and the bottom indicate the Q-tables of PMRL-OM with the unified and the simple conditions, and PMRL-OM, respectively. The three methods have the same route for all agents: the performances of the three methods are the same each other. However, the Q -values of PMRL-OM are less than those of both conditions: both conditions make the agents' progress more in the own learning to reach the goal than PMRL-OM. From this result, PMRL-OM performs worse than both conditions in the other seeds. Note that the grid world 8 is the same as this case.

Conclusion

This paper proposes a noncommunicative learning method to optimally operate multi-agent system with dynamic change, in particular, purposes and rewards. Concretely, this paper proposes the two unified conditions, the standardized step and the simple condition, combined of the number of steps and the rewards, and the proposed method enables the agents to learn cooperative behavior based on the condition. The standardized step is calculated the minimum number of steps if the expected reward values are the same each other for all goals, and the simple condition is calculated the minimum number of steps is divided by the reward value. The proposed method makes all agents learn to achieve the purposes in the order in which they have the minimum number of the condition values. After that, each agent learns cooperative

policy by PMRL as the previous method. This paper analyzes the performance of the two conditions and derives that the standardized steps can be combined both evaluations with almost same weight, whereas the simple condition is sensitive for the reward value, that is, the unified condition can help the agents to select the appropriate purposes among them with the small difference in terms of the two evaluations. In this paper, we test empirically the performances of PMRL-OM with the above two conditions by comparing with the PMRL-OM in three cases of grid world problems whose goal locations and reward values are changed dynamically. The results of this derive that the unified conditions performs well than the PMRL-OM without some condition in the grid world problems. In particular, the standardized step estimates the appropriate purposes accurately than the simple condition.

Strengths

The proposed conditions make agents learn to reach the goal with larger rewards and smaller steps than PMRL-OM. In particular, the standardized step, one of the conditions, evaluates the goals to reach ones with the large reward value per time appropriately. Furthermore, the conditions can avoid making agents to reach the goal to acquire the minimum reward with many spent time. Comparing the two conditions, the standardized step estimates the goal priority effectively than the simple condition. For example, on the grid world 5 in Case (3), the standardized step estimates the optimal goal priority for agents, while the simple condition estimates wrong.

Weaknesses

The proposed conditions have two weak points: required number of iterations and incomplete optimization. As for the former, both conditions require many learning iterations to make PMRL-OM apply to the environmental change as states and rewards. Concretely, from the environmental results, though the agents reach the appropriate goal for minimum step if they explore the goals and learn the own actions appropriately, they actually reach the goals through the detour routes and reach the inappropriate goals with some trials.

As for the latter, both conditions cannot enable agents to get the maximum reward value per time in some situations. That is because each agent calculates the proposed conditions for itself, that is, each agent reaches the near goals from the own start point. This paper focuses on summation of reward values all agents acquired become largest per time.

If the agent's own condition is not the same as the purpose for all agents, the proposed conditions do not perform the best (but better). Concretely, on the grid world 7 in Case (3), both conditions cannot estimate the optimal goal priority for agents to reach the goal with the maximum reward sum per time.

Future Work

There are three directions for the future works: proposing an optimal condition, proposing theoretical methods and applying to the real-world problem. For the first direction, the proposed conditions cannot estimate the goal priority optimally in some cases, e.g., the grid world 7 in Case (3). That reason is the conditions' evaluating locally: each agent learns cooperative actions among near goals which have larger reward value for itself, but not for all agents. To tackle this problem, the conditions have to include the information of the agents' locations. Concretely, we propose one rule to the environment that the agents can observe each other if they are in the same state and extend the conditions based on the information which agents have reached the goal at first to enable the agents to calculate new conditions from the reward value, the distance from the own location to the goals and from other's location to the goals.

For the second direction, the proposed conditions estimate the goal priority appropriately. However, this is shown experimentally, that is, they might estimate the worst in some cases. To tackle this problem, we establish the conditions' capability theoretically. Concretely, in this paper's route planning task, the worst case occurs when the arbitrary agent reaches the arbitrary goal. Hence, we should prove that both conditions make the agent avoid the worst goal. We suppose that they can because all agents avoid to reach the own farthest goals for themselves by both conditions, and the worst goal has to be one of these farthest goals. We have to prove that in the future.

For the third direction, we applied PMRL-OM with the conditions to the route planning tasks. This is supposed to apply it to the transportation problem. Therefore, we apply the PMRL-OM to real-world transportation problems as storehouse robots. In fact, Honig proposed the route planning method based on graph theory for Amazon's storehouse robots [3]. However, this method requires the complete information for the storehouse and the environment is stable. Since this situation cannot be occurred in the real world, our method based on a little information performs better than that method in the real-world storehouse robots operation problem.

Acknowledgements This work was supported by JSPS KAKENHI Grant Number JP17J08724.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Albrecht S, Stone P. Autonomous agents modelling other agents: a comprehensive survey and open problems. *Artif Intell.* 2018;258:66–95.
2. Godoy J, Karamouzas I, Guy S, Gini M. Implicit coordination in crowded multi-agent navigation. 2016. <https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12334>. Accessed May 2019.
3. Hönig W, Kiesel S, Tinka A, Durham JW, Ayanian N. Conflict-based search with optimal task assignment. In: *Proceedings of the 17th international conference on autonomous agents and multi-agent systems, international foundation for autonomous agents and multiagent systems, AAMAS '18*, 2018;pp 757–765
4. Liu Y, Liu L, Chen W. Intelligent traffic light control using distributed multi-agent q learning. In: *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*, 2017;pp 1–8. <https://doi.org/10.1109/ITSC.2017.8317730>
5. Raileanu R, Denton E, Szlam A, Fergus R. Modeling others using oneself in multi-agent reinforcement learning. Tech. rep., 2018. <http://proceedings.mlr.press/v80/raileanu18a/raileanu18a.pdf>
6. Sachiyo A, Katia S. Effective learning approach for planning and scheduling in multi-agent domain. In: *6th international conference on simulation of adaptive behavior*, 2000;pp 507–516
7. Shiraishi D, Miyazaki K, Kobayashi H. Proposal and evaluation of detour path suppression method in ps reinforcement learning. *SICE J Control Meas Syst Integr.* 2019;12(5):190–8.
8. Sutton RS, Barto AG. *Introduction to reinforcement learning*. 1st ed. Cambridge: MIT Press; 1998.
9. Uwano F, Takadama K. Utilizing observed information for non-communication multi-agent reinforcement learning toward cooperation in dynamic environment. *SICE J Control Meas Syst Integr.* 2019;12(5):199–208. <https://doi.org/10.9746/jcmsi.12.199>.
10. Uwano F, Tatebe N, Tajima Y, Nakata M, Kovacs T, Takadama K. Multi-agent cooperation based on reinforcement learning with internal reward in maze problem. *SICE J Control Meas Syst Integr.* 2018;11(4):321–30. <https://doi.org/10.9746/jcmsi.11.321>.
11. Watkins CJ (1989) *Learning from delayed rewards*. Ph.D. thesis, King's College
12. Zemzem W, Tagina M. Cooperative multi-agent learning in a large dynamic environment. In: *Torra V, Narukawa T, editors. Modeling decisions for artificial intelligence*. Cham: Springer; 2015. p. 155–66.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.