This study discusses important factors for zero communication, multi-agent cooperation by comparing different modified reinforcement learning methods. The two learning methods used for comparison were assigned different goal selections for multi-agent cooperation tasks. The first method is called Profit Minimizing Reinforcement Learning (PMRL); it forces agents to learn how to reach the farthest goal, and then the agent closest to the goal is directed to the goal. The second method is called Yielding Action Reinforcement Learning (YARL); it forces agents to learn through a Q-learning process, and if the agents have a conflict, the agent that is closest to the goal learns to reach the next closest goal. To compare the two methods, we designed experiments by adjusting the following maze factors: (1) the location of the start point and goal; (2) the number of agents; and (3) the size of the maze. The intensive simulations performed on the maze problem for the agent cooperation task revealed that the two methods successfully enabled the agents to exhibit cooperative behavior, even if the size of the maze and the number of agents change. The PMRL mechanism always enables the agents to learn cooperative behavior, whereas the YARL mechanism makes the agents learn cooperative behavior over a small number of learning iterations. In zero communication, multi-agent cooperation, it is important that only agents that have a conflict cooperate with each other.

Keywords: multi-agent system, reinforcement learning, internal reward, coooperation

1. Introduction

Multi-Agent Reinforcement Learning (MARL) is a useful approach for executing multi-agent cooperation tasks, such as multi-robot cooperation and traffic signal control [1–6]. However, MARL has difficulty with deriving good performance because the behavior of the agents are too complex to allow for cooperation with other agents. In particular, the cooperative behavior of many agents are barely derived. To address this issue, most of the conventional methods, such as research conducted by Tan [7], encourage agents to cooperate with each other by acquiring the information of other agent through their communication. Such information is very useful for cooperation among the agents, but it goes without saying that the agents are not guaranteed to acquire all of the information needed to cooperate with each other through communication alone. Therefore, it is important to explore methods based on zero communication to achieve multi-agent cooperation. Cooperation without communication among agents is difficult to achieve because the agents do not understand how to cooperate with each other [8, 9]. This is attributed to the fact that the behavior of the agents affects the behavior of other agents, and vice versa.

To address this issue, Uwano proposed a zero communication reinforcement learning method for multi-agent cooperation; this method is referred to as “PMRL” [10]. PMRL is a theoretical method used to allow agents to cooperate with each other in a maze problem involving the experiment for two agents. However, this method has one limitation. The performance of PMRL is not clear when the number of agents is larger than two. This suggests that performance, an important factor for multi-agent cooperation, is unknown in situations where the number of agents is greater than two. To address this issue, we propose another zero communication reinforcement learning method known as Yielding Action Reinforcement Learning (YARL). Using the YARL method, if agents have a conflict, either agent can yield the reward to another agent. This study discusses the factor for the cooperation among the several number of the agents through three experiments: the location of the start point and end goal; the number of the agents, and the size of the maze. We experimentally tested the PMRL and YARL methods in ten mazes using two, three, and five agents. Each maze had different start and goal locations, and both $3 \times 8$ and $5 \times 12$ grid mazes were used.

The remainder of this study is organized as follows. Section 2 explains Q-learning as a reinforcement learning method and the maze problem that was employed by Uwano. In Section 3, the PMRL and YARL methods are introduced and their related architecture, mechanisms, and algorithms are discussed. Section 4 presents the experiment and analysis of the obtained results. Finally, the conclusion of our study is presented in Section 5.
2. Background

2.1. Q-Learning

Reinforcement Learning (RL) [11] is a trial-and-error method that aims at maximizing an acquired reward over a certain unit of time. As its general framework, an RL agent interacts with an environment, observes a state from the environment, selects an action, and then receives a reward as the result of executing that action. Among the many RL methods available, Q-learning [12] is commonly used for single-agent tasks. The Q-learning agent estimates state-action values (referred to as the Q-value $Q(s,a)$) for the possible state-action pairs in the environment, i.e., the agent estimates a discounted expected reward that will be received when its action, $a$, is executed in a state, $s$. Then, the agent learns to acquire a policy $\pi(s,a)$, the probability to select each action in each state, to decide which action should be executed to maximize a received reward. Technically, the policy can be composed of the probabilities associated with selecting any action, $a$, in any state, $s$, and is calculated by $Q(s,a)$, $a \in A$, where $A$ is a set of actions that includes possible actions. To maximize a received reward, $Q(s,a)$ is updated as follows.

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

\hspace{1cm} \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots (1)$$

where $s'$ is the next state, $a'$ is the next action executed in the state $s'$, $r$ is the reward received from the environment, and $\max_{a' \in A} Q(s',a')$ is the largest Q-value when the action, $a' \in A$, is executed in the state, $s'$. In addition to these variables, $\alpha$ is the learning rate and $\gamma$ is the discount factor. Specifically, $\alpha$ is a real number from 0-1 that indicates the learning speed, and $\gamma$ is a real number from 0-1 that indicates how important the future rewards should be considered.

2.2. Dilemma Maze Problem

Uwano employed a dilemma maze problem to validate that his method can generate cooperative behavior between agents. The maze problem can validate whether the agent can change its behavior through the applied learning method or not. The maze has several states and several different kinds of states (i.e., start and goal). The agents depart from the start and continue to observe the states until they reach the goal. If the agents reach the goal, they can acquire the reward. During this cycle, the agents learn to reach the goal to acquire maximum gains over a unit of time.

In a maze problem, if the number of agents is two or more, a dilemma occurs. For example, Fig. 1 shows a maze problem with two agents. Agent A and Agent B indicate the starting positions for each agent, and Goal X and Goal Y indicate the goal positions. Because the agents learn to acquire maximum gains over a unit of time, they will select the goal closest to them. Therefore, Agent A and Agent B choose to aim for Goal X. This situation is called a “conflict.” To resolve the conflict, either agent has to learn to reach Goal Y on its own.

This change in strategy is called “cooperation” and a difference between the purpose of one agent and a group of agents is called a “dilemma.” In addition, when the agents reach different goals in a minimum number of steps, this is called an “optimal situation.” In this study, a step indicates one cycle between state observation and reward receipt, and the minimum step indicates the minimum number of steps taken until the agent reaches the goal. Note that if the number of agents is two or more, the minimum step indicates the minimum number of steps taken until all agents have reached a goal.

3. Zero Communication MARL Methods

To realize cooperation among the agents in the dilemma maze problem, Uwano proposed two learning algorithms, PMRL and YARL.

3.1. Architecture

The agent used in the previous two methods is different from the agent used in Q-learning. Fig. 2 presents the architecture of the Q-learning agent. In this figure, the agent stores the Q-value, the minimum steps, and the goal values as $bidAX$ and $bidAY$. These are real numbers that indicate priority to reach the optimal goal from all goals. The agents used in PMRL and YARL learn to reach the goal with the maximum goal value (the goal value learning method is explained in Section 3.2.1). Furthermore, the agent goes through the six processes presented in this figure: state observation, action selection, Q-value updating, goal selection, and internal reward setting (the details of these processes are explained in Section 3.3). The items inside a normal line area are the same as those used in Q-learning, whereas the items inside a dotted line area, the goal value, only exist in the PMRL agent. There are two new mechanisms introduced in PMRL and YARL, goal selection and internal reward setting. There are many Q-
values for all states and actions, many minimum steps and many goal values for all goals.

3.2. Mechanisms

In PMRL and YARL, the agents select the goal for cooperation, and then they set the internal reward provided when the selected goal is reached. The goal selection mechanism is different in each method. In this section, we introduce the goal selection mechanisms of PMRL and YARL, along with the internal reward settings and their associated characteristics.

3.2.1. Goal Selection

- PMRL

In order for the agents to select their appropriate goals through cooperation, they estimate the goal value. Fig. 3 shows the agent cooperation process using PMRL. In this maze, Agent A and Agent B learn to reach Goal Y, then Agent A learns to reach Goal X under the influence of Agent B. The graph of each agent indicates the goal value belonging to each agent. This figure shows that the value of Goal Y for Agent A is low, and therefore, Agent A starts learning to aim for Goal X because its value is larger than that of Goal Y. The following procedure describes how the agents select their own goals based on their individual goal values, and how the goal values are updated based on the minimum step value.

1. Selecting a goal

The agent selects the goal that has the largest goal value of all the available goals. Considering the goal of Agent A (represented by Goal), the following equation is satisfied.

\[ \text{Goal} = \arg \max_{g} \text{bid}_g \]  

(2)

In this equation, \( g \) represents the goal. After goal selection, the agent estimates the internal reward earned upon reaching the selected goal. Note that the agent generally selects the goal with the largest goal value, but sometimes it selects a goal randomly, introducing a small probability for the goal values to be updated evenly.

2. Updating goal value

Note that the agents do not utilize the minimum step of other agents in Eqs. (3) and (4). Instead, the agents employ information on whether the minimum step towards their goals is larger or smaller than those of the other agents. Specifically, if Agent A does not observe one of the other agents when heading for Goal Y, Agent A recognizes that it will reach Goal Y faster than the other agents. This means that \( t_{AY} = \min_{a} t_{atY} \), where \( a \) indicates one of the agents. However, if Agent A observes one of the other agents when heading toward Goal Y, Agent A recognizes that the other agent will reach Goal Y faster than Agent A, meaning \( t_{AY} \neq \min_{a} t_{atY} \). This suggests that the agents can correctly update the goal value using Eqs. (3) and (4) without acquiring the minimum step of other agents through communication.

\[ \text{bid}_a = \frac{n-1}{n} \text{bid}_a + \frac{1}{n} t_{AY} \left( t_{AY} = \min_{a} t_{atY} \right) \]  

(3)

\[ \text{bid}_a = \frac{n-1}{n} \text{bid}_a + \frac{0}{n} t_{AY} \left( t_{AY} \neq \min_{a} t_{atY} \right) \]  

(4)

Eqs. (3) and (4) indicate the average of the minimum steps and the average of those sometimes multiplied by zero, respectively. By applying infinite time, Eqs. (12) and (6) are derived from the above equations. From these equations, the agents can resolve any conflict and cooperate with each other because the difference in the goal values among all the agents always exists.

\[ \lim_{n \to \infty} \text{bid}_a = t_{AY} \left( t_{AY} = \min_{a} t_{atY} \right) \]  

(5)

\[ \lim_{n \to \infty} \text{bid}_a = 0 \left( t_{AY} \neq \min_{a} t_{atY} \right) \]  

(6)

Eqs. (3) and (4) are based on the following equation:

\[ \text{bid}_n = \frac{n-1}{n} \text{bid}_{n-1} + \frac{1}{n} t_{n} \]  

(7)

In this equation, \( n \) indicates the number of updates. \( \text{bid}_n \) indicates the goal value in \( n \) number of updates, and \( t_{i} \) indicates the minimum step after updating \( n \) times. This equation can be derived as follows:

\[ \text{bid}_n = \frac{t_{1}}{n} + \frac{t_{2}}{n} + \cdots + \frac{t_{n-1}}{n} + \frac{t_{n}}{n} = \frac{1}{n} \sum_{i=1}^{n} t_{i} \]  

(8)

From this equation, the goal value (i.e., \( \text{bid}_n \)) is calculated as the average of the minimum step value. When the agent reaches the same goal (i.e., almost all \( t_{i} (i = 1, \ldots, n) \) are the values around the minimum step to the goal), the following equation is derived.

\[ \text{bid}_n = \frac{1}{n} \sum_{i=1}^{n} t_{i} \]  

(9)

\[ \frac{1}{n} \left( (t+c_{1}) + (t+c_{2}) + \cdots + (t+c_{n}) \right) \]  

(10)

\[ = t + \frac{c}{n} \]  

(11)
The agents learn via Q-learning. If the agents experience a conflict, they begin to cooperate with each other.

In YARL, initially, the agents learn by Q-learning, as described in process 1. As a result, if the agents have a conflict in which they reach the same goal, they learn to cooperate, as described in process 2. In this study, the nearest goal is determined by the minimum step of the agents. The agent recognizes the goal closest to it as having the lowest minimum step of all the available goals. If the agents still have a conflict after these processes are executed, they learn to cooperate with each other by following process 2 again. Fig. 4 shows cooperation between two agents using the YARL approach. In this figure, two of the same mazes include two agents, A and B, who learn to reach two goals, X and Y. In this figure, the upper graphic represents process 1, and the lower graphic represents process 2. Agent A and Agent B learn to reach Goal X in the upper portion of the maze, but they experience a conflict (process 1). Agent B is close to the goal and learns to reach Goal Y in order to yield Goal X to Agent A in the lower graphic (process 2). YARL can create cooperation by utilizing the RL results.

### Algorithm 1 PMRL Algorithm

**Require:** Agents become in start positions

- \( t^i_0 = \max \text{step} (i = [0, \text{AgentNumber}], x = [0, \text{GoalNumber}]) \)
- \( T_{ix} = \max \text{step} (i = [0, \text{AgentNumber}], x = [0, \text{GoalNumber}]) \)
- \( g_i \in G(i = [0, \text{AgentNumber}]) \)
- \( v_{gi}(i = [0, \text{AgentNumber}]) \)

1. for iteration = 0 to MaxIteration do
   2. for All agents reach the goals or step = 0 to MaxStep do
      3. Agents observe their states
      4. Agents choose actions
      5. The agents which don’t reach the goal update Q-value
      6. if Agent i has reached goal x then
         7. \( T_{ix} = \text{step} \)
         8. end if
      9. end for
   10. for \( i = 0 \) to AgentNumber, \( x = 0 \) to GoalNumber do
      11. if \( T_{ix} < t^i_1 \) then
          12. \( t^i_1 = T_{ix} \)
          13. end if
      14. end for
   15. Agents select the goal \( g_i(i = 0, 1, \ldots, \text{AgentNumber} - 1) \)
   16. if \( g_i \) isn’t selected by others then
      17. \( v_{gi} \) is updated by \( v_{gi} = v_{gi} + \frac{t^i_0}{\text{iteration}} \)
      18. else if \( t^i_1 \) is smaller than that of others select \( g_i \)
      19. \( v_{gi} \) is updated by \( v_{gi} = v_{gi} + \frac{t^i_0}{\text{iteration}} \)
      20. else
      21. \( v_{gi} \) is updated by \( v_{gi} = v_{gi} + \frac{t^i_0}{\text{iteration}} \)
      22. end if
   23. Agents estimate internal reward
   24. Agents update Q-value by the internal reward
   25. end for

3.2.2. Internal Reward Design

To reach the goal selected by the methods described in Section 3.2.1, the internal reward is employed for PMRL and YARL. Each agent has set internal rewards for all goals. The internal reward is calculated based on the reward that the agent acquires and the minimum step value. The agent can learn to reach any goal based on the internal reward setting.

Figure 5 shows how the internal reward is designed for cooperation among agents. In this figure, \( t_{AX} \) and \( t_{AY} \) indicate the minimum steps from the start, B, to the goals, X and Y. The star mark represents the turning position to select whether Agent B should reach Goal X or Y, the directional arrows from the turning position to each...
Algorithm 2 YARL Algorithm.

Require: Agents become in start position
\[ t^i_x = \text{MaxStep}(i = [0, \text{AgentNumber}), x = [0, \text{GoalNumber}) \]
\[ T_{ix} = \text{MaxStep}(i = [0, \text{AgentNumber}), x = [0, \text{GoalNumber}) \]
\[ g_t \in G(i = [0, \text{AgentNumber})) \]
\[ C^i_t (i = [0, \text{AgentNumber}), x = [0, \text{GoalNumber}) \]
1. for iteration = 0 to MaxIteration do
2. for All agents reach the goals or step = 0 to MaxStep do
3. Agents observe their states
4. Agents choose actions
5. The agents which don’t reach the goal update Q-value
6. if Agent i has reached goal x then
7. \[ T_{ix} = \text{step} \]
8. if Agent i has a conflict in goal x then
9. \[ C^i_t = C^i_{t+1} \]
10. end if
11. end if
12. end for
13. if i = 0 to AgentNumber do
14. if \[ T_{ix} < t^i_x \] for goal x which agent i reach then
15. \[ t^i_x = T_{ix} \]
16. end if
17. end for
18. if iteration%Cycle = 0 then
19. if \[ \frac{C^i_{tx}}{C^i_{ty}} > \text{ProbConflict} \] (agent i, j and goal x) then
20. if Agent i reached goal x faster than Agent j then
21. Agent j selects the nearest goal without x
22. end if
23. end if
24. end if
25. Agents estimate internal reward
26. Agents update Q-value by the internal reward
27. end for

Goal State Selections for Cooperation Among Agents with RL

In such a situation, \( ir_X \) and \( ir_Y \) should be set as \( r \) and \( r + \delta(>0) \), respectively. \( \delta \) is employed to ensure that cooperation is established among the agents. Although Agent B in Fig. 1 should reach Goal Y through cooperation with Agent A, Agent B cannot always reach Goal Y in the case of no \( \delta \) because the Q-values needed to reach both goals are the same value (i.e., the agent reaches Goal Y with 50%). To solve this problem, \( \delta \) is added to the following equations, where \( \gamma \) is a constant value larger than zero. If \( ir_Y \) is larger than \( r + \delta \), the agent can always reach Goal Y. In Fig. 5, the internal rewards of agents A and B, \( ir_{AX}, ir_{AY}, ir_{BX} \) and \( ir_{BY} \) are calculated by the following equations:

\[
ir_{AX} = \gamma^{AX-t_{AX}}r + \delta \\
ir_{AY} = r \\
ir_{BX} = r \\
ir_{BY} = \gamma^{BY-t_{BY}}r + \delta
\]

The internal rewards are calculated by Eqs. (13)–(16) for the situation presented in Fig. 5. If each agent learns to reach another goal, the internal rewards of Agent A are calculated in the same manner as those of Agent B, and vice versa.

In cases involving two or more agents, the agents set the internal rewards for all goals based on Eqs. (13) and (16). For example, consider a maze problem involving a certain number of agents and five goals, V, W, X, Y, and Z. One of the agents, Agent A sets the internal reward for reaching Goal V based on the maximum difference in the minimum step between goal V and the other goals (i.e., \( t_{AV} - t_{BV}, t_{AX} - t_{BX}, t_{AY} - t_{BY}, t_{AZ} - t_{BZ} \)). This means that the agent can set the internal reward by itself using a comparison of the difference in the minimum steps it requires to reach one goal compared to other goals. In other words, the agent does not need information on any of the other agents to make this determination.

3.3. Algorithms

Before explaining the PMRL and YARL algorithms, we must first explain the following variables employed in the algorithms: \( t^i_x, T_{ij}, g_t, v_g \), and \( ir_t \). First, \( t^i_x \) indicates the
minimum step of agent $i$ to goal $x$, and $T_{ix}$ indicates the temporary variable of the minimum step of agent $j$ to goal $x$ ($T_{ix}$ does not matter which agent has). Agent $i$ updates $t_{x}^{i}$ or $t_{y}^{i}$ by itself when its minimum step toward goal $x$ or $y$ changes from that of agent $j$ (agent $j$ updates $t_{x}^{j}, t_{y}^{j}, t_{x}^{j}, t_{y}^{j}$ in the same manner as agent $i$). The remaining variables, $g_{i}$ and $v_{g_{i}}$, respectively indicate the identification number of the goal that agent $i$ selects and the goal value of $g_{i}$ both of which are only used for PMRL. $g_{i}$ in YARL is the variable that indicates the goal. Finally, $ir_{i}$ indicates the internal reward of agent $i$ as mentioned in Section 3.2.2.

The variables used in both algorithms, iteration, MaxIteration, MaxStep, AgentNumber and GoalNumber, indicate the current iteration, the maximum iterations, the maximum steps, the number of agents, and the number of goals, respectively.

### 3.3.1. PMRL Algorithm

In the PMRL algorithm (Algorithm 1), all agents observe the states, select actions, and update $Q$-values (lines 3, 4, and 5). If agent $i$ reaches goal $x$, the current step is temporarily stored as $T_{ix}$ (lines 6, 7, and 8). After all agents reach their goals (or the step exceeds MaxStep), agent $i$ updates $t_{x}^{i}$ by $T_{ix}$ as the minimum step if $T_{ix}$ is smallest (lines 10–14). Agent $i$ selects goal $g_{i}$ to set the goal value (line 15), and $v_{g_{i}}$ is updated according to $g_{i}$ (lines 16–22). Agent $i$ estimates the internal reward by Eq. (13) to reach the appropriate goal for cooperation (line 23). The $Q$-value of the action needed to reach the goal is updated with the internal reward by Eq. (17) (line 24).

As the basic principle, the agent learns to reach the goal according to the “ordinary” reward provided by Q-learning when reaching the “not-selected” goal, whereas the agent learns it according to the “internal” reward provided by PMRL when reaching the “selected” goal. Note that this principle does not change when the agent reaches the goal for the first time or when it reaches a previously visited goal.

\[
Q(s,a) \leftarrow Q(s,a) + \alpha [ir_{AS} + \delta + \gamma \max_{d' \in A} Q(s',d') - Q(s,a)]
\]

### 3.3.2. YARL Algorithm

In the YARL algorithm (Algorithm 2), all agents observe the states, select actions, and update $Q$-values (lines 3, 4, and 5). If agent $i$ reaches the goal $x$, the current step is temporarily stored as $T_{ix}$ (lines 6 and 7), and if other agents reach the same goal as agent $i$, a value of one is added to the conflict count, $C_{ix}$ (lines 8, 9, and 10). After all agents reach their goals (or the step exceeds MaxStep), the agent updates $t_{x}^{i}$ by $T_{ix}$ as the minimum step if $T_{ix}$ is the smallest (lines from 13 to 17). Then, the agents check whether the agents have a conflict or not once every several iterations, Cycle. If $C_{ix}$ and $C_{j}$ get over the threshold, $ProbConFLICT$, for the same goal $x$, the nearest agent to the conflict goal learns to reach the next nearest goal to yield the goal to other agents (lines from 18 to 24). The agent estimates the internal reward by Eq. (13) to reach the appropriate goal for cooperation (line 25). The $Q$-value of the action needed to reach the goal is updated with the internal reward by Eq. (17) (line 26).

### 4. Experiment

#### 4.1. Experimental Design and Setting

To compare the PMRL and YARL methods, we employ maze problems to conduct the following two experiments:

- **Experiment 1:** The robustness of the methods is investigated in terms of the number of agents in the $3 \times 8$ grid maze (i.e., 2, 3, and 5 agents).
- **Experiment 2:** The scalability of the proposed methods is investigated by applying them into the large $5 \times 12$ grid maze (1.5 times larger than the $3 \times 8$ maze).

##### 4.1.1. Maze of Experiment 1

For Experiment 1, several cases are explored. All of the agents in Cases (2-1) and (3-1) do not have to learn to cooperate with each other to reach the goals closest to them (i.e., all agents can reach the correct goals without the proposed methods; this means that only Q-learning enables all agents to reach the correct goals), while all agents in Cases (2-2) and (3-3) have to learn to cooperate with each other. In Case (3-2) in particular, two agents have to learn to cooperate with each other while one agent does not. Note that cases utilizing five agents are not employed because they can be exchanged for different combinations of cases utilizing two and three agents. In Experiment 1, there are 10 mazes that utilize different start and goal positions. Fig. 6 shows the sample maze in Experiment 1. In these mazes, A, B, and C indicate the starting position of agents A, B, and C, and X, Y, and Z indicate the goals. For the five agents, there are five types of mazes used for cooperation.

- **Maze Problem involving two agents**
  - Case (2-1): Each agent has to learn on its own
  - Case (2-2): Two agents have to cooperate with each other

- **Maze Problem involving three agents**
  - Case (3-1): Each agent has to learn on its own
  - Case (3-2): Two agents have to cooperate with each other, while another agent learns for itself
  - Case (3-3): Three agents have to cooperate with each other

##### 4.1.2. Maze of Experiment 2

For Experiment 2, we employ mazes that are 1.5 times larger than the mazes used in, Case (2-2), Case (3-3) and the case involving five agents. Fig. 7 shows that the $5 \times 12$ grid maze is in the same situation as the Case (2-2) maze. Specifically, there are three mazes, 1.5 times of Figs. 6(b), 6(e), and 8.
4.1.3. Evaluation Criteria and Parameters

This study evaluates the steps taken before the agents reach a goal as the evaluation criterion. The total number of experiments is determined by the number of trials (e.g., 300 trials with 30 different seeds in 10 kinds of mazes for each case in experiment 1, and 90 trials with 30 different seeds in 3 mazes in experiment 2).

The Q-learning parameters are summarized in Table 1. In this table, the learning iterations and step counts are limited to 50,000 and 100, respectively, as the threshold in experiments 1 and 2. We initialize the Q-values of all states as zero. The parameters, $\alpha$ and $\gamma$, are set to 0.1 and 0.9, respectively. Note that PMRL and YARL employ $\varepsilon$-greedy selection during the learning phase, while they employ greedy selection during the evaluation phase. Specifically, the agents select their actions according to the $\varepsilon$-greedy selection method using the $\varepsilon = 0.7$ value, and they evaluate the learning result according to the greedy selection method. We set $\varepsilon = 0.7$ in the learning phase because the agents have to explore a large space to find the minimum steps of all combinations of the agents and the goals. The ordinary (external) reward is set to 10. The constant $\delta$ is set as 10, while the random goal selection, $\eta$, is set as 15% for PMRL. The learning cycle, $Cycle$, is set to 5,000 and the threshold, $ProbConflict$, is set to 0.8 for YARL.

Table 1. Parameters.

<table>
<thead>
<tr>
<th></th>
<th>Experiment 1</th>
<th></th>
<th>Experiment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Q-learning</td>
<td>PMRL</td>
<td>YARL</td>
</tr>
<tr>
<td>iterations</td>
<td>50000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maximum steps</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>initial Q-value</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>learning rate $\alpha$</td>
<td>0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>discount rate $\gamma$</td>
<td>0.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>action selection</td>
<td>$\varepsilon$-greedy, $\varepsilon = 0.7$</td>
<td></td>
<td>-action selection</td>
</tr>
<tr>
<td>reward</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>constant $\delta$</td>
<td>none</td>
<td>10</td>
<td>none</td>
</tr>
<tr>
<td>random goal selection $\eta$</td>
<td>none</td>
<td>15%</td>
<td>none</td>
</tr>
<tr>
<td>Cycle</td>
<td>none</td>
<td>5000</td>
<td>none</td>
</tr>
<tr>
<td>ProbConflict</td>
<td>none</td>
<td>0.8</td>
<td>none</td>
</tr>
</tbody>
</table>

4.2. Experimental Results

4.2.1. Experiment 1

Figure 9 shows the steps until the agents reach the goal in the mazes presented in Figs. 6 and 8; as the typical one of 10 results for each case. The vertical axis indicates the step, while the horizontal axis indicates the learning iteration. The line with square markers indicates the minimum step, and other three lines in Fig. 9 indicate the results of Q-learning, PMRL, and YARL. If the step is 100, this situation suggests that the agents cannot cooperate with each other. As shown in Fig. 9(a), the Q-learning and YARL
steps are the minimum step, and while the PMRL step becomes low, it does not reach the minimum step value. In Fig. 9(b), the Q-learning step is 100 for almost all iterations, while those of PMRL and YARL become the minimum step. Figs. 9(d)–9(f) present the results for the three agents. In Fig. 9(d), the Q-learning and YARL steps are the minimum step, and while the PMRL step decreases, it does not become the minimum step. In Fig. 9(e), the Q-learning step is 100, while the PMRL and YARL steps converge to a low step. However, the YARL step converges to the minimum step, while the PMRL step does not. In Fig. 9(f), the steps of all methods are the same as those in Fig. 9(e). However, the steps of both methods converge to the minimum step, unlike Fig. 9(e). Fig. 9(c) shows the result of the five agents. The Q-learning and YARL steps are the same as those presented in Fig. 9(e), and the PMRL step is approximately 50.

4.2.2. Experiment 2

Figure 10 shows the step until the agents reach the goal in the large maze in Case (2-2), Case (3-3), and the case involving five agents. The vertical axis indicates the step, and the horizontal axis indicates the learning iteration. The line with square markers indicates the minimum step and the other three lines in Fig. 10 indicate the results of Q-learning, PMRL, and YARL. For the Q-learning and YARL methods, the results are the same as those presented in Fig. 9. For PMRL, the results are the same as those presented in Fig. 9 for Case (2-2) and Case (3-3), while the result is not converged, but that is converged to several step in the situation of five agents.

4.3. Characteristic Results

Figure 11 presents results that are not the same as those presented in Subsection 4.2.1. The left side of this figure represents a two agent case. Figs. 11(a) and 11(d)
but not the minimum step. Specifically, the YARL step is 100, while the steps of PMRL and YARL are low, and does not converge, the PMRL step is quite low but not the minimum step, and the YARL step converges to around the minimum step. In \( \text{Fig. 9(c)} \), the Q-learning step is not 100, and the steps of both methods are quite low. However, the YARL step is not the minimum step, while that of PMRL is the minimum step. The center section of \( \text{Fig. 11} \) presents results that are different from those presented in Section 4.2.1 for the three agent case. Figs. 11(b) and 11(e) present the results of Maze 8 in Case (3-2) and Maze 6 in Case (3-3), respectively. In both figures, the Q-learning step is 100, and while the steps of PMRL and YARL are quite low, they are not the minimum steps. The right side of \( \text{Fig. 11} \) shows that the results are not the same as those in Subsection 4.2.1 for the five agent case. In \( \text{Fig. 11(c)} \), the Q-learning and YARL step values are 100 for almost all iterations, while that of PMRL converges to around the minimum step. In \( \text{Fig. 11(f)} \), the Q-learning step is 100, while the steps of PMRL and YARL are low, but not the minimum step. Specifically, the YARL step is converged near the minimum step, while that of PMRL is approximately 60.

4.4. Discussion

From the experimental results, the agents can resolve conflicts using both methods, but the results differ depending on the number agents involved. In addition, comparing \( \text{Fig. 9(b)} \) to \( \text{Fig. 10(a)} \), \( \text{Fig. 9(f)} \) to \( \text{Fig. 10(b)} \), and \( \text{Fig. 9(c)} \) to \( \text{Fig. 10(c)} \), the results for the 3×8 grid maze are the same as those for the maze that is 1.5 times larger. This suggests that the cooperation among the agents is influenced not by the maze size, but by the number of agents. Furthermore, the agents can not cooperate with each other to achieve the minimum step. For example, the agents cannot reach the goals at the minimum step value in \( \text{Fig. 9(c)} \) for PMRL, and the agents cannot cooperate with each other in \( \text{Fig. 11(c)} \) for YARL. We discuss the reasons for these results in the next sections.

4.4.1. PMRL

The behavior of the agents cannot be optimal in Cases (2-1), (3-1), (3-2) and (5) (i.e., from Figs. 9(a), 9(d), and
That is because the PMRL method, which makes agents cooperate with each other, cannot make them learn greedy (i.e., Q-learning), in Case (2-1), (3-1) and (3-2), while PMRL requires a lot of learning iterations than 50,000 for cooperation in Case (5). In Fig. 6(a), agents A and B learn to reach goals Y and X, respectively. In Fig. 6(c), agents A, B, and C learn not to reach the goals X, Y, and Z, which is the optimal combination. Fig. 6(d) presents the same situation as Fig. 6(c). This suggests that PMRL can encourage cooperation, but this is not optimal for Cases (2-1), (3-1), and (3-2). Specifically, PMRL step is 8 in Figs. 9(a) and 6, and 5 in Figs. 9(d) and 9(e), respectively. As for Case (5), Fig. 12 presented the goal value of each combination of agent and goal. The error bars in this figure indicate a standard deviation. In this figure, if all agents reach the goal by following the goal values in Fig. 12, this situation is the most optimal for cooperation among the agents. However, the error bars for all agents are several numbers. This is because the goal values of all agents are not converged, and the agents require many learning iterations.

The step of the characteristic results for PMRL is not the minimum step, as demonstrated in Figs. 11(a), 11(b), 11(e), and 11(f). There are two potential reasons for this: the agents require more than 50,000 learning iterations, or the agents made mistakes during the goal selection process. In addition, Fig. 13 shows the maze for characteristic results. The characters from A to E represent each agent, and the characters from V to Z represent each goal. Fig. 14 represents the goal values in Maze 5 of Case (2-2). The vertical axis indicates the goal value, while the horizontal axis indicates the combinations of the agents and the goals. From this figure, the goal values of Agent A are the same values with each other. That is why the goal values of Agent B cannot be converged, then the step cannot converge the minimum step in Maze 5 of Case (2-2). Fig. 15 indicates the goal values in Maze 8 of Case (3-2). The axes are the same as Fig. 14. In this situation, agents A-C have to select the goals X, Y, Z, and Y for the optimal situation, respectively. However, in this figure, agent C selects goal Z because Agent B selects Goal Y and Z by following the same goal values of the goals with each other. This situation is the same as that in Maze 5 of Case (2-2). Fig. 16 indicates the goal val-
characteristic results, the YARL steps in processes change occurs every 5,000 iterations. For the to select the goal to reach with the minimum step with the optimal process. Note that a process in Cases (2-1) and (3-1), one process in Cases (2-2) and (3-2), two processes in Case (3-3) and four processes in Case (5) to select the goal to reach with the minimum step. In Fig. 9, all YARL steps are converged to the minimum step with the optimal process. Note that a process change occurs every 5,000 iterations. For the characteristic results, the YARL steps in Figs. 11(d), 11(b), and 11(f) did not converge to the minimum step, while the YARL step in Fig. 11(c) is 100. Fig. 18 shows Maze 8 of Cases (2-2) and (3-2), and Maze 4 of Case (5) and the goal selections in those mazes. In Figs. 18(a)–18(c), the vertical axis indicates the goal name, while the horizontal axis indicates the learning iterations. From these figures, the agents cannot select the optimal goal by following the YARL algorithm. For example, Agent A and Agent B with YARL select the Goal X and Goal Y in Fig. 18(a). However, each agent has to select another goal. This suggests that YARL cannot completely lead the optimal situations for multi-agent system. In the results presented in Fig. 11(c), it can be seen that the agents cannot cooperate with each other. That is because the agents made mistakes in determining the conflict. Figs. 13(c) and 19 show Maze 3 of Case (5) and the goal selection in that maze. In Fig. 19, the vertical axis indicates the goal name, while the horizontal axis indicates the learning iterations. Ideally, in this maze, agents A–E reach goal V, V or W, W, Y or Z and Z, respectively, then agents A, B, D, and E determine the conflict and select other goals at every 5,000 iterations. However, the agents C and E selects goal V and Y, respectively, while agent D does not determine the conflict at every 5,000 iterations. These results suggest that Cycle for YARL is more learning iterations for 5,000 in order for the agents to converge the Q-value.

Fig. 16. Goal value in Maze 6 of Case (3-3).

Fig. 17. Goal value in Maze 4 of Case (5) in 200,000 learning iterations.

Fig. 18. Goal selection with YARL in mazes 8 and 4 of Case (2-2), (3-2), and (5).
4.4.3. Important Factor for Cooperation

From the results for PMRL and YARL, it is clear that PMRL can always enable the agents to cooperate with each other. However, this method requires many learning iterations. Although YARL can lead the optimal situation for the cooperation, this method cannot always lead cooperation. These reasons suggest that the goal value and the estimation method are effective for the cooperation among the agents, but it is also important to revise the results of Q-learning. In PMRL, if the agents select one goal, the goal value of the agents without the nearest agent from the goal becomes smaller than before. That is why the goal values of all agents for the goal are different from each other, and why the agents can select a goal that the other agents do not select. Note that if the goal values of one agent for different goals are the same, the agents can learn to cooperate with other agents; and (3) several agents have to learn to cooperate with other agents, and others have to learn with Q-learning. This research utilized 10 types of mazes for each class. However, this research employed 5×12 and 9×9 grid mazes. These mazes were classified into the above class (2) for 2, 3, and 5 agents. This research applied the agents to these mazes. The experimental results revealed the following: (a) PMRL and YARL enable the agents to cooperate with each other even if the number of agents and the size of the maze are changed; (b) the PMRL mechanism can always make the agents cooperate with each other, whereas the YARL mechanism can make the agents cooperate with each other with a small number of learning iterations; and (c) it is important in zero communication multi-agent cooperation that only agents that have a conflict cooperate with each other.

The proposed methods can promote cooperation in many situations. However, there is no method for guaranteeing cooperation in all situations. We proposed a new method to tackle this issue. Specifically, we proposed a method that combined PMRL with YARL. From the results of this research, because either method can always enable the agents to cooperate with each other, it can be suggested that the method that combined PMRL with YARL is effective. Furthermore, because these methods were proposed for multi-agent cooperation in a maze problem, we would generalize PMRL and YARL and apply these methods to several tasks or applications in other contexts, such as the pursuit game problem, puddle world problem, or transportation problem. Specifically, we have to determine whether the tasks or applications satisfy the following two conditions designed in the maze problem: (1) the learning is terminated when all agents reach the goals; (2) when one agent reaches the goal, the agent cannot move from the goal (i.e., absorbing Markov chain). This means that the two methods can be applied to many kinds of tasks or applications that satisfy these conditions. In addition to this issue, further careful qualifications and justifications are needed to generalize our results.

5. Conclusion

This study compared two learning methods for multi-agent cooperation without inter-agent communication. The intention was to discuss the important factors for zero communication cooperation among any agents in any maze. As one of these methods, PMRL, encourages agents to learn to reach the farthest goal, which contributes to solving any conflicts that may arise among the agents. The second method proposed in this study, YARL, enables the agent closest to a goal to learn to reach the next closest goal.

For this comparison, we employed two experiments where we varied the number of agents and the size of maze. Specifically, this study classified scenarios into three cooperation situations: (1) all agents have to learn with Q-learning; (2) all agents have to learn to cooperate with other agents; and (3) several agents have to learn to cooperate with other agents, and others have to learn with Q-learning. From the results for PMRL and YARL, it is clear that these justifications are needed to generalize our results.
Acknowledgements
This work was supported by JSPS KAKENHI Grant Number JP17J08724

References: