Autonomous Learning of Reward Distribution in Not100 Game

Katsunari Shibata(shibata@cc.oita-u.ac.jp), Tsutomu Masaki & Masanori Sugisaka

Dept. of Electrical & Electronics Engineering, Oita University, 700 Dannoharu, Oita 870-1192, Japan.

Abstract

In this paper, autonomous learning of reward distribution in multi-agent reinforcement learning was applied to the 4 player game named "not100". In this game, more shrewd tactics to cooperate with the other agents is required for each agent than the other tasks that the learning was applied previously. The reward distribution ratio after learning was varied among simulation runs. However, the validity of the average non-uniform reward distribution ratio was examined in some ways. The three agents with higher win probability after learning cooperated mutually, while strong cooperation was not observed in some cases when the agents learned with a fixed distribution ratio.

1 Introduction

In multi-agent systems, since it is difficult to know the policy to solve a given task in advance, autonomous learning such as reinforcement learning is useful. However, it is one of the difficult problems to decide "reward distribution among agents" that affects to generate cooperative behaviors. There are some methods proposed already, but the reward distribution ratio is decided in advance. For example, one agent gets the whole reward, or the reward is distributed uniformly to all the agents. There is also the method that the agent who contributes directly to get a reward gets a part of the reward, and the rest of it is distributed to the other agents uniformly[1]. However, since appropriate distribution depends deeply on a given task, enough knowledge about the task is required to determine the distribution. Accordingly there is much possibility that this spoils the effectiveness of reinforcement learning, which is useful when the knowledge about the given task is not enough.

Then the method has been proposed that the agent learns the reward distribution ratio to the other agents together with the learning of actions[2]. The learning is based on the principle that by the distribution of the reward, the other agents become to help him, and finally he gets more reward or gets the reward earlier. This method was applied to a simple two-agent and three-agent competitive problems. It was shown that (1) the state that no agents get any rewards by a continuous conflict can be avoided, (2) the distribution ratio becomes small to the agent whose help is not needed, and (3) not only the ratio but also the change of the ratio affects the action learning. In the three-agent problem, the tactics of an agent was observed such that he disturbs the goal of another agent who gave him less reward than the other one.

In this paper, the reward distribution learning is applied to a different type of problem, and the effectiveness in wide area is verified. The problem has different properties as (1) more shrewd tactics is required, (2) some agent always can get reward in a finite time, (3) the time to get the reward is not considered, and (4) not only one agent can get reward. The problem is the four-player game named "Not 100 game". In this game, since one player always loses the game when the other three players cooperate mutually, shrewd tactics is necessary. It was verified whether cooperation with the other agents emerges, and whether the distribution ratio obtained by the learning is valid or not.

2 Learning of Reward Distribution

The reward distribution is learned together with the action, but the time scale is different between them. The action is learned at each time step, while the reward distribution ratio is fixed for some trials, and is updated according to the total reward obtained during the period when the ratio is fixed. Concretely, $dist_{ji}$ that is the reward distribution ratio from agent i to agent j is changed using random numbers, and is fixed for one cycle that is defined as N trials. Since the distribution ratio should always satisfy

$$\sum_{j=1}^{A} dist_{ji} = 1.0, \tag{1}$$

the change of the ratio $\Delta dist_{ji}$ is calculated as

$$\Delta dist_{ji} = rnd_{ji} - rnd_{(j+1)\%A,i}, \tag{2}$$

where A: the number of agents and rnd: a random number. When $dist$ becomes larger than 1.0 or less than 0.0, it is set to be 1.0 or 0.0 respectively. The difference is distributed uniformly to the ratio to the others. In order to remove the effect in the transition period, the total reward R in the latter half of the cycle is calculated using the reward r_i of agent i as

$$R_i = \sum_{n=N/2+1}^{N} \sum_{i=1}^{A} dist_{ji} r_i(n) \tag{3}$$

and is evaluated. The learning is so simple that when R is larger than the previous value, the distribution ratio is set as the default value, and otherwise, the distribution ratio is restored to the previous value.

Here, each agent acts sequentially, and the state transition is deterministic. The action learning is based on Q-learning. Since the time to get the reward is not necessary to be considered in the problem in this paper, the discount factor γ is set to be 1.0. As an example, the learning of the agent $j = 0$ is shown in the followings. The state evaluation $V_j(s_j(t+1))$ just after his action is calculated from the possibility P_{fin} that the game finishes before his next tern, expected reward in that case $\bar{r}_{fin}$, and the expected maximum Q-value $maxQ_j$ at his next tern as

$$\begin{aligned} V_j(s_j(t+1)) = {} & P_{fin_j}(s_j(t+1))\bar{r}_{fin_j}(s_j(t+1)) \\ & +(1-P_{fin_j}(s_j(t+1)))maxQ_j(s_j(t+1)). \quad (4) \end{aligned}$$

Each term on the right hand side is calculated as

$$\begin{aligned} P_{fin_j}(s_j(t+1)) \quad & \leftarrow \quad (1-\alpha)P_{fin_j}(s_j(t+1)) + \alpha \\ & \text{if the game finishes before his next tern} \\ & \leftarrow \quad (1-\alpha)P_{fin_j}(s_j(t+1)) \\ & \text{otherwise,} \qquad (5) \end{aligned}$$

$$\begin{aligned} \bar{r}_{fin_j}(s_j(t+1)) \leftarrow {} & (1-\alpha)\bar{r}_{fin_j}(s_j(t+1)) \\ & +\alpha\sum_{i=0}^{A} dist_{ji}r_i(t+k) \\ & \text{if the game finishes at } t{+}k, \qquad (6) \end{aligned}$$

$$\begin{aligned} maxQ_j(s_j(t+1)) \leftarrow {} & (1-\alpha)maxQ_j(s_j(t+1)) \\ & +\alpha max_k(Q_j(s_j(t+A), a_k) \qquad (7) \end{aligned}$$

where α: a learning constant. Q value is learned using V as

$$\begin{aligned} Q_j(s_j(t), a(t)) \leftarrow {} & (1-\alpha)Q_j(s_j(t), a(t)) \\ & +\alpha\sum_{i=0}^{A} dist_{ji}r_i(t+1) + V_j(s_j(t+1)) \qquad (8) \end{aligned}$$

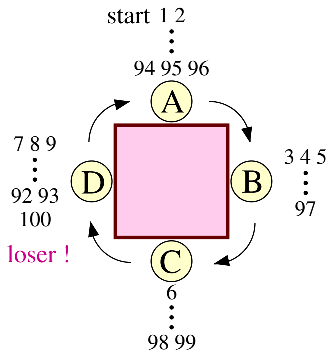


Figure 1: Not 100 game

1. 4 players count sequentially from 0
2. Each one counts 1, 2, or 3 numbers at one time.
   (If the present number is 90, the next player can count until 91, 92, or 93.)
3. The player who counts 100 loses the game.
   (If the present number is 99, the next player has to count 100 and becomes a loser automatically.)

Figure 2: The rule of "Not 100 Game"

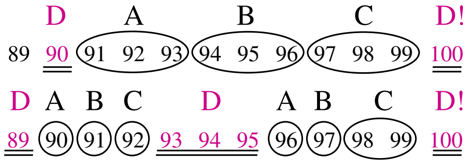


Figure 3: The reason why the agent always loses the game when the other 3 agents cooperate mutually

3 Not100 Game

Here, not 100 game is introduced. As shown in Fig. 1, 4 players sit at a table, each player counts within 3 numbers sequentially, and the player who counts 100 loses the game. Fig. 2 shows the rule of this game.

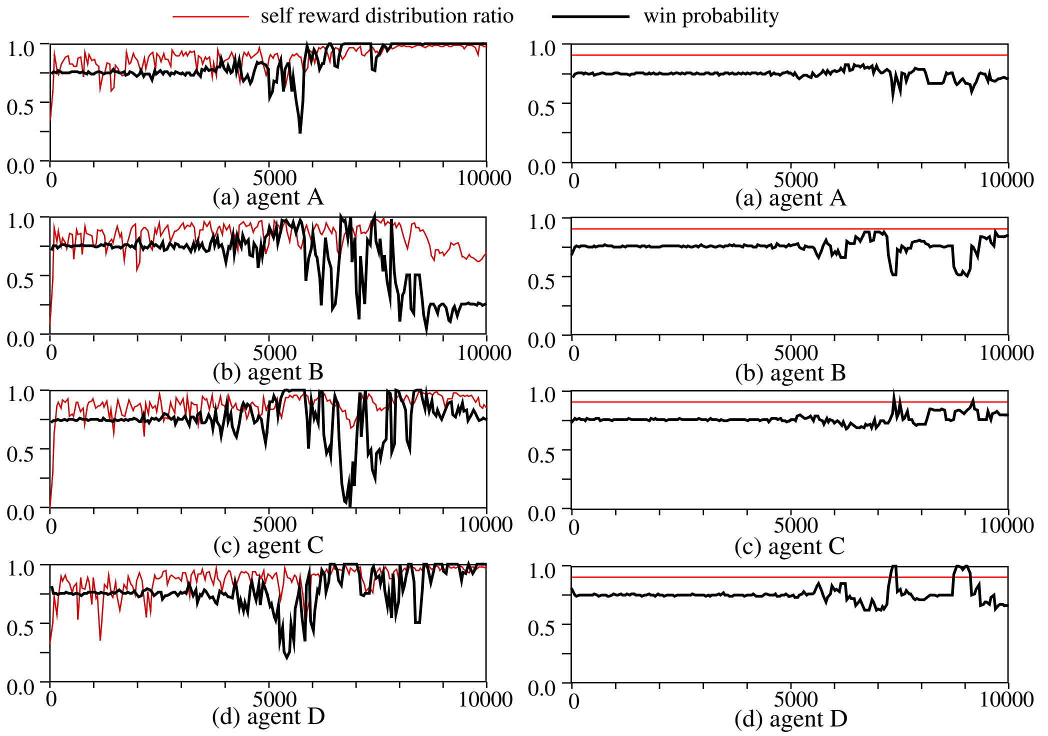
In this game, one player cannot win the game when the other 3 players cooperate mutually. Accordingly, it becomes important how to get the help of the other players to win the game. Actually, the game is interesting at the point that the human relation between players and the character of each player can be peeped.

Fig. 3 shows two examples of the game processes. Here, it is supposed that the player A, B, and C cooperate mutually. If the player D counts 97, 98 or 99, one of the others has to count 100, and the player D can win the game. If he counts one of the numbers from 90 to 96, since one player can count one, two or three numbers at one time, he has to count 100 at the next tern as shown in the upper process in Fig. 3. However, when he counts 89, he also cannot count 97, 98, or 99 at the next tern as shown in the lower process in Fig. 3, he has to count 100. At the case of Fig. 1, the loser depends on whether the player C counts only 98 or two numbers of 98 and 99.

In this paper, to make the computation time short, "Not 30 game" is employed on behalf of "Not 100 game". To all the agents except for the agent who counts 30, the reward 1.0 is given. In the previous tasks to which the authors applied the learning, it is clear that the reward distribution is profitable for any of the agents, because no reward is given when the conflict state happens. However, in this problem, three agents always get the reward in a finite time, and the appropriate distribution is not clear.

4 Simulation

One cycle is defined as $400(= N)$ trials, and the distribution ratio is updated at every cycle. 10000 cycles are done in one simulation. The initial distribution ratio is decided randomly with the condition that each



(a) agent A

(b) agent B

(c) agent C

(d) agent D

(A) when reward distribution ratio was learned

(a) agent A

(b) agent B

(c) agent C

(d) agent D

(B) when reward distribution ratio was fixed at 0.91 for itself and 0.03 for the others

Figure 4: Change of each agent's win-probability and reward self-distribution ratio

one is positive and the total is 1.0. The range of the random number added to the distribution ratio is reduced from 0.1 to 0.01 linearly in log scale in 9000 cycles. The first count agent is decided randomly at each trial. The Boltzmann selection is employed for the action selection after normalizing to make the maximum Q value become 1.0. The temperature is also reduced from 1.0 to 0.1 linearly in log scale in 8000 cycles.

Fig. 4(a) shows the change of the self reward distribution ratio and the win probability for each of 4 agents. In the early stage of the learning, since the temperature was high, the win probability of every agent was almost 0.75. The self-distribution ratio became around the value from 0.8 to 0.9 soon even if the initial ratio is decided randomly. The win probability began to fluctuate from around 5000 cycle, but the way of change varied depending on the simulation run. The distribution ratio of the agent whose win probability is large is apt to be large.

For comparison, Fig. 4(b) shows the result when the distribution ratio is fixed at 0.91 for himself and 0.03 for the others. The fluctuation of the win probability is smaller. The reason can be thought that in the learning case, the small change of the distribution ratio sometimes influences the win probability.

Next, the validity of the distribution ratio is verified. Table 1 shows the average ratio after learning over 100 simulation runs. It is seen that the self-distribution ratio of the agent whose win probability is large becomes large. Even though the ratio varied actually, it is also seen that depending on the simulation run, the ratio to the next agent is slightly smaller than the other agent. The reason can be thought that the help of the next agent contributes the win probability less than the other agents.

Table 1: The reward distribution ratio after learning.

	the agent whom the reward is distributed to			
	myself	next agent	opposite agent	previous agent
the agents with *win_prob*>0.9	0.961	0.011	0.013	0.014
all agents	0.895	0.037	0.033	0.035

Furthermore, the ratio of one agent was fixed, and the win probability, total reward, and the average self-distribution ratio of the other agents were observed. Fig. 5 shows them as a function of the fixed ratio. Each of them is the average over 10 simulation runs. When the fixed ratio is small, the win probability is 1.0 except for the case of 0.0. It is suddenly decreasing as the fixed ratio becomes larger around 0.9. The total reward is the maximum when the fixed ratio is around 0.8. The acquired self-distribution ratio after learning as in Fig. 1 is larger than 0.8, but is not different so much. It is interesting that the self-distribution ratio of the other agents also becomes larger when the fixed ratio is larger than 0.8. This is because even

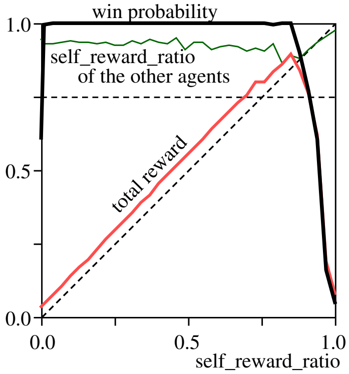
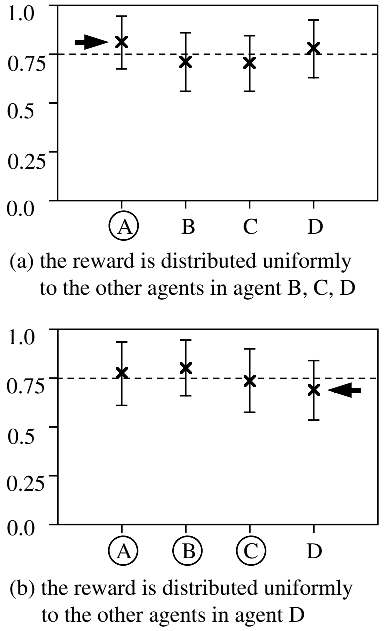


Figure 5: The win probability, total reward, and self-distribution ratio of the other agents as a function of fixed self-distribution ratio.



(a) the reward is distributed uniformly to the other agents in agent B, C, D

(b) the reward is distributed uniformly to the other agents in agent D

Figure 6: Effect of non-uniform reward distribution. The alphabet with circle indicates the agent with non-uniform reward distribution.

though the other agents makes their self-distribution ratio large, they can win the game.

The validity of the non-uniform distribution to the other agents as shown in Table 1 is verified. Fig. 6 (a) shows the win probability when the ratio of the agent A is fixed as Table 1 and that of the others is fixed such that the self-distribution is the same but the rest is distributed uniformly to the others. The win probability of the agent A is larger than the others. As shown in Fig. 6(b), when only the agent D distributed uniformly to the others, the win probability of the agent D became smaller. From these results, the obtained weighted ratio is supposed to be valid.

Table 2: Change of win probability by the replace of one agent between two groups.

		max	Group A (receiving)			
			learning	selfish 1.0-0.0	0.91 -0.03	uniform 0.25-0.25
	min		0.238	0.578	0.592	0.711
Group B (sending the strongest agent)	learning	1.000		0.504	0.511	0.898
	selfish	0.921	0.158		0.568	0.932
	0.91 -0.03	0.889	0.175	0.554		0.920
	uniform	0.784	0.096	0.302	0.345	

Finally, the learning was performed in 4 groups, and the win probability was observed when the weakest agent was replaced by the strongest agent in the other group. In one group, all the agents learned the ratio. In other groups, the ratio of every agent was fixed. The fixed ratio varied among three groups as in Table 2. The result are shown in Table 2. When the weakest agent in the learning group was replaced, the win probability of the newcomer agent is only around 0.1, while the win probability of the strongest agent in the learning group is more than 0.5 in every other group. However, the strongest agent in the selfish group or the group of (0.91-0.03) ratio could win more often than the strongest agent in the learning group. The result can be interpreted that in the learning group, an appropriate cooperation strategy is obtained by the agents, it is hard for the other group agent to win. However, since the cooperation strategy is not effective in the other groups, the strongest agent in the learning group could not win very much.

5 Conclusion

The reward distribution learning was applied to “Not 100 game”. The weighted reward distribution was observed, and the validity was examined in some ways. The cooperation could be observed when the distribution ratio was learned, while it could not be observed when the distribution ratio is fixed as to distribute the reward to the other agents uniformly.

Acknowledgment

A part of this research was supported by Grants-in-Aid for Scientific Research of the Ministry of Education, Culture, Sports, Science and Technology of Japan (#13780295)

References

[1] Shirakawa, H., et al., “Experimental Study on Emergence of Cooperative Action Using Reinforcement Learning”, *Proc. of the 5-th Intelligent Systems Symp.*, pp. 119–124, 1998. in Japanese

[2] Shibata, K. and Ito, K., “Autonomous Learning of Reward Distribution for Each Agent in Multi-Agent Reinforcement Learning”, *Intelligent Autonomous Systems*, **6**, pp. 495–502, 2000.