Coordination Structures Generated by Deep Reinforcement Learning in Distributed Task Executions

Extended Abstract

Yuki Miyashita
Computer Science and Communications Engineering
Waseda University
Tokyo
y.miyashita@isl.cs.waseda.ac.jp

Toshiharu Sugawara
Computer Science and Communications Engineering
Waseda University
Tokyo
sugawara@waseda.jp

ABSTRACT

We investigate the coordination structures generated by deep Q-network (DQN) in a distributed task execution. Cooperation and coordination are the crucial issues in multi-agent systems, and very sophisticated design or learning is required in order to achieve effective structures or regimes of coordination. In this paper, we show the results that agents establish the division of labor in a bottom-up manner by determining their implicit responsible area when input structure for DQN is constituted by their own observation and absolute location.

KEYWORDS

Multi-agent deep reinforcement learning; Coordination; Cooperation; Divisional cooperation

ACM Reference Format:
Yuki Miyashita and Toshiharu Sugawara. 2019. Coordination Structures Generated by Deep Reinforcement Learning in Distributed Task Executions. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 3 pages.

1 INTRODUCTION

Cooperation and coordination for improving overall efficiency in multi-agent systems (MAS) is an important issue. However, the appropriate strategic regime of multiple agents for cooperation is influenced by a variety of factors such as task structures, frequency of task occurrence, and environmental characteristics, which makes it a challenging issue.

Deep reinforcement learning (DRL) has produced many successful results in fields such as robotics [1, 3] and games [4, 6, 7]. Recently, some studies apply DRL to MAS so as to accomplish cooperation and coordination in MAS [2, 5, 8]. However, it is still not clear how the observations of agents and data input to the DQN affects the generated coordination structures in multi-agent deep reinforcement learning (MA-DRN).

Our contribution is to examine what kinds of strategic cooperative behaviors emerge in multiple agents by changing the observable view of each agent as the inputs to the DQNs. We use the *distributed task execution game*,which is similar to the *treasure hunting game*. Although this game is simple, it is appropriate to understand the characteristics of emerging coordination regimes. In our model, agents have their own DQNs and, to examine how input structure which contains agent observation and absolute location affects the results of coordination, we change the agent's observable range size to compare the effect of observation.

Our experimental results show that regardless of the observable size, agents were able to establish the division of labor in a bottom-up manner, in the sense that they determined individual areas that each agent would be responsible for.

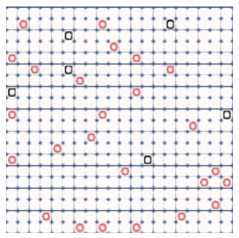
Figure 1: Example of problem environment.

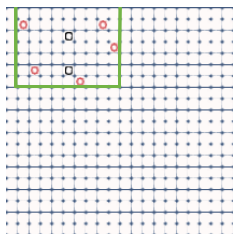
Figure 2: Agent's observable state when $V = 4$

2 PROBLEM FORMULATION

We consider a multi-agent problem called the *distributed task execution game* in which tasks continuously appear somewhere in an environment at a certain rate and multiple agents move around to select and execute the tasks concurrently. A problem environment is, as shown in Fig. 1, a $N \times N$ lattice in which black squares are agents and red circles are tasks they have to execute. The possible actions are one of $A = \{up, right, down, left\}$. If an agent moves onto a cell containing a task, it executes the task and receives a positive reward r. Then, the task disappears and a new task is placed on another cell.

An agent can observe the limited local area (a limited range of observation is specified by the observable range size V) whose center is itself and can know the absolute locations. Then, it composes its observed local information and the abtract entire map, as shown in Fig. 2 ($V = 4$), and inputs to the own DQN. Since agents only can observe inside of its range (the green square is the observable range in Fig. 2), the unobservable regions are set to be blank. When the environment transit to next state, agent can receive a reward if it executs the task. Then, agents have to select and take

Yuki Miyashita works with Shimizu Corporation.
This work is partly supported by KAKENHI (17KT0044).

Table 1: Experimental parameters.

Parameter	Value
Size of environment N	20
No. of agents n	6
Reward r	1
Epoch length H	200
Sum of epochs F	60,000

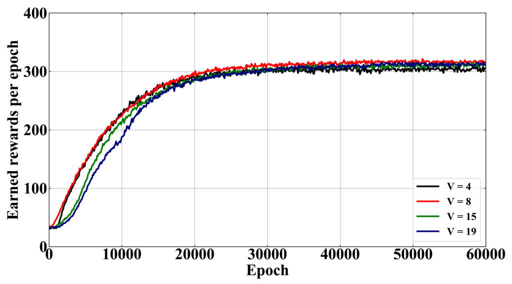

Figure 3: Earned rewards per epoch (relative views, m = 25).

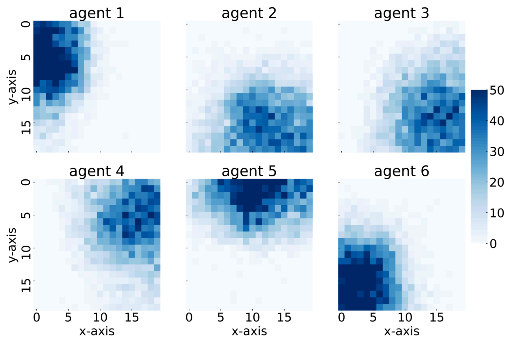

Figure 4: Locations of executed tasks (relative views).

the action only on the basis of observed local states. Because we consider MA-DRL, the agents individually learn the Q-values (or their policies) in order to improve the performance, i.e., agents autonomously identify appropriate coordinated/cooperative behaviors to obtain more rewards using their own DQNs. The architecture of the neural network for DQN is composed of convolutional network layers, max pooling layers, and fully connected network (FCN) layers.

We introduce discrete time $t \geq 0$, and initially, n agents and m tasks are scattered in the environment. Agents carry on actions until an epoch of the game end (if $t \geq H$, an epoch of the game end),and we iterate this game for F epochs, where F is also a positive integer. The objective of the agents is to maximize the number of rewards they receive, so they learn which action will result in higher rewards every time by using the ε-greedy strategy with decay and experience replay. When agents update their own network parameter, they calculate action value error at the mean squared loss function, and we adapt RMSprop [9] as optimizer of loss function.

3 EXPERIMENTAL RESULTS

We experimentally compare the performances (i.e., the total rewards earned by all agents) and analyze the coordination behavior by changing their observable range size V. The parameters for these experiments are listed in Table 1. The results of the earned rewards per epoch from 1 to 60,000 epochs (12, 000, 000 time steps) are plotted in Fig. 3, where each plot is the average value of the earned rewards every 100 epochs when V = 4, 8, 15, and 19. Note that agents with V = 19 can observe the entire environment correctly. These results clearly indicate that the total earned rewards increased along with epochs in all cases, but their performances were almost identical regardless of the observable range sizes (agents with V = 8 exhibited the best performance).

To analyze the structure of coordinated behavior, we investigate where each agent was working in the environment. We counted the number of tasks that each agent executed in individual cells between 55,000 and 60,000 epochs when V = 8 and visualized these data using heat maps. This is shown in Fig. 4, where the darker blue cells indicate that the corresponding agent executed more tasks, and white and faint blue cells indicate that it seldom or never executed tasks, respectively. We can see that the movements of all agents were localized and tended to execute tasks in specific regions; it seems that they form a division of labor by segmentation in a bottom-up manner. We call these segmented regions where mainly just one agent is moving around the *responsible regions*. When multiple agents try to do the same task, only one agent can earn the reward and the attempt of the other agents go to waste. Therefore, they generate their working regions to avoid such conflicts. In the earlier epochs (around 10,000 epochs), agents already started to execute only in specific regions, but the regions are unclear and indistinct. Then, they gradually formed shapes and the locations were stable. Note that we also confirmed that when V = 4, 15, and 19, similar divisional cooperation appeared, and specific regions are more specified when V is the larger.

4 CONCLUSION

Our experimental results indicated that agents were able to learn different coordination strategies. In our game, redundant or useless actions are caused by conflicts, meaning that multiple agents target the same tasks, so agents attempted to identify strategies to reduce such conflicts. Therefore, agents established divisional cooperation on the basis of locational segmentation. In contract, when we conducted experiments that agents only had their observation (didn't get absolute location) , agent didn't establish divisional cooperation on the basis of locational segmentation. However the earned total reward were higher than those in Fig. 3 because agent can formed flexible coordination in which agents targeted tasks by pay attention to location of other agents to avoid conflicts.

We would like to extend our environments, agents, and games for our future work. For example, we will explore situations where the environment has an obstacle, task generation is biased in a certain area, tasks have structures that should be done cooperatively with a number of different agents, and agents have their specialties.

REFERENCES

[1] Smruti Amarjyoti. 2017. Deep Reinforcement Learning for Robotic Manipulation - The state of the art. *CoRR* abs/1701.08878 (2017). arXiv:1701.08878 http://arxiv.org/abs/1701.08878

[2] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. 2017. Cooperative Multi-agent Control Using Deep Reinforcement Learning. In *Autonomous Agents and Multiagent Systems (AAMAS workshop)*, Gita Sukthankar and Juan A. Rodriguez-Aguilar (Eds.). Springer International Publishing, Cham, 66–83.

[3] Maximilian Hüttenrauch, Adrian Sosic, and Gerhard Neumann. 2017. Guided Deep Reinforcement Learning for Swarm Systems. *CoRR* abs/1709.06011 (2017). arXiv:1709.06011 http://arxiv.org/abs/1709.06011

[4] Guillaume Lample and Devendra Singh Chaplot. 2017. Playing FPS Games with Deep Reinforcement Learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.* 2140–2146. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14456

[5] Joel Z. Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. 2017. Multi-agent Reinforcement Learning in Sequential Social Dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS '17)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 464–473. http://dl.acm.org/citation.cfm?id=3091125.3091194

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR* abs/1312.5602 (2013). arXiv:1312.5602 http://arxiv.org/abs/1312.5602

[7] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[8] Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. 2018. Lenient Multi-Agent Deep Reinforcement Learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '18)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 443–451. http://dl.acm.org/citation.cfm?id=3237383.3237451

[9] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4, 2 (2012), 26–31.