Off-Beat Multi-Agent Reinforcement Learning

Extended Abstract

Wei Qiu
Nanyang Technological University
Singapore
qiuw0008@e.ntu.edu.sg

Weixun Wang
Tianjin University
Tianjin, China
wxwang@tju.edu.cn

Rundong Wang
Nanyang Technological University
Singapore
rundong001@e.ntu.edu.sg

Bo An
Nanyang Technological University
Singapore
boan@ntu.edu.sg

Yujing Hu
NetEase Fuxi AI Lab
Hangzhou, China
huyujing@corp.netease.com

Svetlana Obraztsova
Nanyang Technological University
Singapore
lana@ntu.edu.sg

Zinovi Rabinovich
Nanyang Technological University
Singapore
zinovi@ntu.edu.sg

Jianye Hao
Tianjin University
Tianjin, China
jianye.hao@tju.edu.cn

Yingfeng Chen
NetEase Fuxi AI Lab
Hangzhou, China
chenyingfeng1@corp.netease.com

Changjie Fan
NetEase Fuxi AI Lab
Hangzhou, China
fanchangjie@corp.netease.com

ABSTRACT

We investigate cooperative multi-agent reinforcement learning in environments with off-beat actions, *i.e.*, all actions have execution durations. During execution durations, the environmental changes are not synchronised with action executions. To learn efficient multi-agent coordination in environments with off-beat actions, we propose a novel reward redistribution method built on our novel graph-based episodic memory. We name our solution method as LeGEM. Empirical results on stag-hunter game show that it significantly boosts multi-agent coordination.

KEYWORDS

multi-agent coordination; multi-agent reinforcement learning

ACM Reference Format:
Wei Qiu, Weixun Wang, Rundong Wang, Bo An, Yujing Hu, Svetlana Obraztsova, Zinovi Rabinovich, Jianye Hao, Yingfeng Chen, and Changjie Fan. 2023. Off-Beat Multi-Agent Reinforcement Learning: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

Despite the recent successes of multi-agent reinforcement learning (MARL) in autonomous systems [1, 12] and real-time strategy (RTS) video games [10], learning effective multi-agent coordination in environments with off-beat actions remains challenging for MARL. Many cooperative MARL methods [3, 4, 6, 7] fail to learn efficient multi-agent coordination in environments where action durations are caused by off-beat actions. The main reason is TD-learning [8] fails when displaced rewards caused by action durations are used in training. To this end, we propose a novel reward redistribution method built on our novel graph-based episodic memory. We name our method as LeGEM. Empirical results on stag-hunter game show that it significantly boosts multi-agent coordination in environments with off-beat actions and achieves leading performance.

2 PRELIMINARIES

MARL aims to learn optimal policies for all the agents in the team. With TD-learning and a global Q value proxy Q^{tot} for the optimal Q^*, $\{Q_i\}_{i=1}^{N}$ are optimized via minimizing the loss [2, 11]: $\theta^* = \arg\min_{\theta^*} \mathcal{L}(\theta) := \mathbb{E}_{D' \sim \mathcal{D}}[(y_t^{\text{tot}} - Q_\theta^{\text{tot}}(\boldsymbol{s}_t, \boldsymbol{u}_t))^2]$, where $y_t^{\text{tot}} = r_t + \gamma \max_{\boldsymbol{u}'} Q_{\bar{\theta}}^{\text{tot}}(\boldsymbol{s}_{t+1}, \boldsymbol{u}')$ and θ is the parameters of the agents. $\bar{\theta}$ is the parameter of the target Q^{tot} and is periodically copied from θ. D' is a sample from the replay buffer $\mathcal{D}$.

3 METHODOLOGY

3.1 Temporal Recency Reward Redistribution

To learning efficient multi-agent coordination in environment with off-beat actions for MARL methods. We redistribute rewards to agents' pivot timesteps (we will introduce the method for searching agent's povit timesteps in the following subsection). The *pivot timestep* of each agent is the timestep when the off-beat action was executed and later triggered the reward.

The timestep to which the reward should be distributed is called the *final pivot timestep.* We denote the *final pivot timestep* at timestep t as e_t. For a shared reward at timestep t, each agent's pivot timestep

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom. © 2023 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

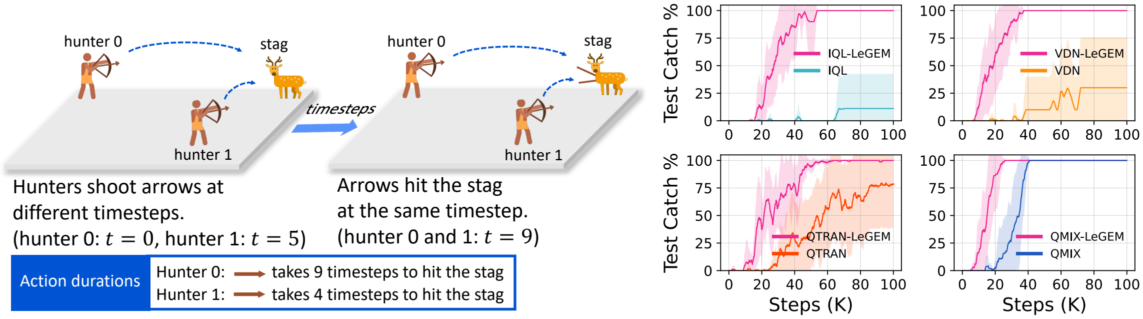

Figure 1: An illustrative scenario: two-agent stag-hunter game, where two agents (hunters) have only partial observations, different durations of the shoot action, and cannot communicate. The goal is to catch the stag and they are rewarded when their shots hit the stag at the same time. Both agents can see the stag. As the shoot action durations of the two agents are different, to catch the stag, the two agents should shoot the arrow at different timesteps given the distances. For hunter 0, the timestep to shoot the arrow is 0, while for hunter 1, it is 5. At timestep 9, the two arrows will hit the stag and all hunters will receive a positive shared reward. Though the scenario is easy for human beings, it is hard for MARL agents due to the action duration. Experiment results: in this scenario, the optimal policy for agent 0 is to shoot the arrow at timestep 0 while the optimal policy for agent 1 is to shoot the arrow at timestep 5. Such an asynchronous property of OBMAS motivates agents to learn tacit policies. The curves show that VDN and IQL fail to learn coordination policies even in this simple scenario. With our LeGEM, MARL methods gain enhanced performance as well as improved sample efficiency.

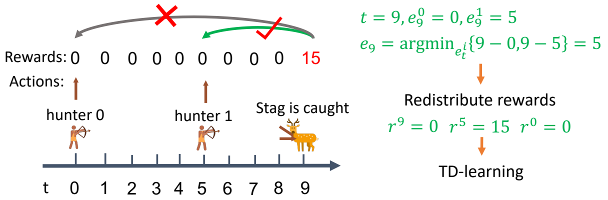
Rewards: 0 0 0 0 0 0 0 0 0 15
Actions:
hunter 0
hunter 1
Stag is caught
t 0 1 2 3 4 5 6 7 8 9
$t = 9, e_9^0 = 0, e_9^1 = 5$
$e_9 = \text{argmin}_{e_t^i}\{9 - 0, 9 - 5\} = 5$
Redistribute rewards
$r^9 = 0 \quad r^5 = 15 \quad r^0 = 0$
TD-learning

Figure 2: Reward redistribution for the example in Fig. 1.

e_t^i can be different. We can get e_t via proximity:

$$e_t = \arg\min_{e_t^i}\{t - e_t^i\}_{i=1}^N \qquad (1)$$

In the example of Fig. 1, the final pivot timestep e_9 for timestep 9 is $e_{t=9} = 5$. Then, we can utilize it to update the reward in each transition $(\boldsymbol{s}^{e_t}, \tilde{\boldsymbol{u}}^{e_t}, r^{e_t}, \boldsymbol{s}^{e_t+1})$:

$$\hat{r}^{e_t} = \mathbb{1}(e_t \geq t) \cdot r^{e_t} + \mathbb{1}(e_t < t) \cdot r^t, \qquad (2)$$

where $\mathbb{1}(\cdot)$ is the indicator function. r^{e_t} is replaced with $\hat{r}^{e_t}$. This update rule is conducted iteratively from $t = 0$ to $t = T - 1$. To stabilize learning and circumvent the overestimation of the TD target, r^t is also updated after Eqn. 2 via $(1 - \mathbb{1}(e_t < t) \cdot (1 - \beta)) \cdot r^t$. Therefore, we can utilize updated transitions with newly replaced rewards for MARL training. We also present an example in Fig. 2 to illustrate the workflow of our reward redistribution method.

3.2 Episodic Memory and Searching Method

The reward redistribution method introduced in Sec. 3.1 relies on certain structures to searching the pivot timestep e_t^i for agent i. We propose our novel episodic memory (EM). During training, each agent i collects its individual trajectories τ_i. We then define τ_i of agent i as $\tau_i = [(o_i^0, \tilde{u}_i^0, r^0), \cdots, (o_i^{T-1}, \tilde{u}_i^{T-1}, r^{T-1})]$, where T is the length of the trajectory and the triplet $(o_i^t, \tilde{u}_i^t, r^t)$ represents the observation, action and reward of timestep t. r^t is globally shared between agents.

Graphs and Sub-Graph. Each agent's episodic memory (EM) has T graphs categorized by the length of the episode. Each graph consists of many sub-graphs that are categorized by the episode return. We define the graph of agent i's EM as a directed graph $\phi_i^t \in \Phi_i$ where Φ_i is the set of graphs of agent i and ϕ_i^t is the t-th graph of Φ_i, $t \in \{0, \cdots, T-1\}$. To model an agent's behaviour explicitly and make the trajectories easy to represent, we create T graphs for each agent and let $\Phi_i = \{\phi_i^t\}_{t=0}^{T-1}$ where T is the maximum depth of all graphs and the maximum length of the episode as well. The maximum level of ϕ_i^t is $t+1$. The graph consists of nodes that are connected by edges. Each node contains key, visit count and pointers connecting the precursors (nodes at the previous level) and the successors (nodes at the next level). Besides the ϕ_i^t, we define the sub-graph set of ϕ_i^t as $\Phi_i^{t,\Omega} = \{\phi_i^{t,\omega}\}_{\omega=0}^{\Omega-1}$ by using the discretized episode return and there are Ω sub-graphs. $\phi_i^{t,\omega}$ is the ω-th sub-graph whose episode return is the ω-th item in $[0, \cdots, \boldsymbol{r}^{t,i}]$ where $\boldsymbol{r}^{t,i}$ is the maximum discretized episode return of ϕ_i^t. We define the key $(o_i^t, \tilde{u}_i^t)$ using agent i's o_i^t and $\tilde{u}_i^t$ at timestep t. The visit count of the node indicates the total number of visits made by agent i to the node. The initial value of the visit count is 1.

Updating Graphs. Given τ_i of length T, if the node is already in the graph at level t, we then increase the visit count by 1. Otherwise, we create a new node for level t of the graph and update its pointers. Meanwhile, sub-graphs will be also created and updated.

Searching method. We propose our search schemes for our reward redistribution method. For agent i, given τ_i, the corresponding graph is $\phi_i^l = \Phi_i[l]$ ($l = \texttt{length}(\tau_i)\texttt{-}1$) and $\phi_i^{l,\omega} = \Phi_i^{l,\Omega}[\omega]$, and episode return is $\boldsymbol{r}^{l,i}$. Agent i searches from the node (the key is $(o_i^t, \tilde{u}_i^t)$ and $o_i^t \in \tau_i$, $u_i^t \in \tau_i$) at level t in sub-graph $\phi_i^{l,\omega}$ to find the pivot timestep e_t^i for r^t. Searching ends when the pattern of decreasing or increasing visit count ends and the corresponding level is the candidate pivot timestep.

4 EXPERIMENTS AND CONCLUSION

We conduct experiments on stag-hunt game as shown in Fig. 1. We select QMIX [4], VDN [7], IQL [9] and Qtran [6] as baselines. We implement our method on PyMARL [5] and use 10 random seeds to train each method on the testbed. Results show that with our method LeGEM, MARL methods gain enhanced performance.

REFERENCES

[1] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. 2012. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics* 9, 1 (2012), 427–438.

[2] 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), 529–533.

[3] Frans A Oliehoek, Christopher Amato, et al. 2016. *A Concise Introduction to Decentralized POMDPs*. Vol. 1. Springer.

[4] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*. 4295–4304.

[5] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philiph H. S. Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. *CoRR* abs/1902.04043 (2019).

[6] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. 5887–5896.

[7] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296* (2017).

[8] Richard Stuart Sutton. 1984. *Temporal credit assignment in reinforcement learning*. Ph.D. Dissertation. University of Massachusetts Amherst.

[9] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. 2017. Multiagent cooperation and competition with deep reinforcement learning. *PLoS ONE* 12, 4 (2017).

[10] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575, 7782 (2019), 350–354.

[11] Christopher JCH Watkins and Peter Dayan. 1992. Q-Learning. *Machine Learning* 8, 3-4 (1992), 279–292.

[12] Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, et al. 2020. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. *arXiv preprint arXiv:2010.09776* (2020).