Multi-Agent Hierarchical Reinforcement Learning with Dynamic Termination

Extended Abstract

Dongge Han, Wendelin Boehmer, Michael Wooldridge, Alex Rogers
University of Oxford
Oxford, United Kingdom
dongge.han@cs.ox.ac.uk,wendelin.boehmer@cs.ox.ac.uk,michael.wooldridge@cs.ox.ac.uk,alex.rogers@cs.ox.ac.uk

ABSTRACT

In a multi-agent system, an agent's optimal policy will typically depend on the policies of other agents. Predicting the behaviours of others, and responding promptly to changes in such behaviours, is therefore a key issue in multi-agent systems research. One obvious possibility is for each agent to broadcast their current intention, for example, the currently executed option in a hierarchical RL framework. However, this approach results in inflexible agents when options have an extended duration. While adjusting the executed option at each step improves flexibility from a single-agent perspective, frequent changes in options can induce inconsistency between an agent's actual behaviour and its broadcasted intention. In order to balance flexibility and predictability, we propose a dynamic termination Bellman equation that allows the agents to flexibly terminate their options.

KEYWORDS

Multi-agent Learning, Hierarchical Reinforcement Learning

ACM Reference Format:
Dongge Han, Wendelin Boehmer, Michael Wooldridge, Alex Rogers. 2019. Multi-Agent Hierarchical Reinforcement Learning with Dynamic Termination. 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

Many important real-world tasks [13], such as taxi coordination [5], supply chain management [2], and distributed sensing [4] are multi-agent by nature. Despite the success of single-agent reinforcement learning (RL) [7, 10], multi-agent RL has remained an open challenge. For example, from one agent's perspective the environment dynamics are non-stationary as they can be affected by the others' actions [9]. An optimal policy will thus typically depend on the policies of other agents, and hence it is essential that an agent learns the behaviours of others. One possible solution is to let each agent model and broadcast its intention, in order to indicate its subsequent behaviours [1]. Fortunately, hierarchical RL provides a simple solution for modeling agents' intentions by allowing them to use *options*, which are subgoals that an agent aims to achieve in a finite horizon. Makar et al. [6] proposed *multiagent hierarchical RL*, where agents can coordinate by broadcasting their current option to the others. However, despite the advantage brought by using options, there can be a delay in an agent's response to changes in others' behaviours and in the environment, due to the temporally-extended nature of options. A potential solution to improving agents' flexibility towards changes is to terminate options prematurely. This has been studied previously to address the problem of *imperfect options* in single-agent settings [11]. However, this approach may no longer prove advantageous in a multi-agent scenario: if an agent frequently switches its option, the option that is broadcasted may become inconsistent with its subsequent behaviour. This poses a dilemma unique to multi-agent systems: insufficient termination of options results in agents' delayed response to changes, while excessive terminations makes an agent's behaviour unpredictable.

We propose a dynamic termination Bellman equation, which allows an agent to flexibly terminate its current option. This approach balances flexibility and predictability, combining the advantages of both. The intuitive way of modelling dynamic termination is to keep an additional controller which decides whether to terminate the current option at each step. In this paper, we incorporate the controller as an additional option. In this way, the Q-value of the newly introduced option is associated consistently with the Q-values of the original options, and our model introduces negligible additional complexity to the original model.

2 BACKGROUND

The Options Framework [11] introduces *options* as temporally extended actions. An option o is defined as a triple $\langle \mathcal{I}^o, \beta^o, \pi^o \rangle$, where $\mathcal{I}^o \subseteq \mathcal{S}$ is the initiation set and $\beta^o : \mathcal{S} \to [0, 1]$ is the option termination condition. $\pi^o : \mathcal{S} \to \mathcal{A}$ is a deterministic option policy that selects primitive actions to achieve the target of the option. On reaching the termination condition in state s', an agent can select a new option from the set $O(s') := \{o \mid s' \in \mathcal{I}^o\}$. A Semi-Markov Decision Process (SMDP) [11] defines the optimal Q-value:

$$Q(s_t, o_t) \quad = \quad \mathbb{E}\Big[\sum_{\tau=0}^{k-1} \gamma^\tau r_{t+\tau} + \gamma^k \max_{o' \in O(s_{t+k})} Q(s_{t+k}, o')\Big], \qquad (1)$$

where k refers to the number of steps until the termination condition $\beta^{o_t}(s_{t+k}) = 1$ is fulfilled. Options are usually pre-trained and have to cover a large range of tasks, without being able to solve any one task perfectly. To address this imperfect options problem, Sutton et al. [11] have shown that premature termination at each step to select the next best option can improve performance significantly. Harutyunyan et al. [3] further addressed the adverse effect of early termination on exploration, by executing a separate *exploration policy* during training, which follows the selected option, but randomly terminates with a fixed probability ρ.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

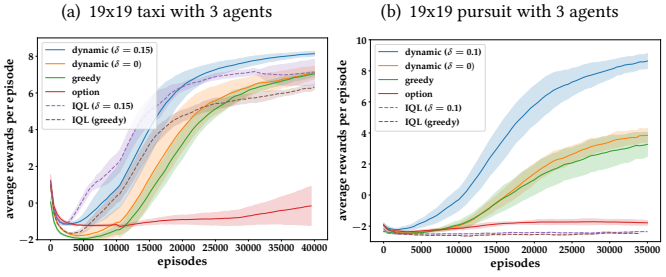

Figure 1: Results from the taxi and pursuit tasks. Each point per 500 episodes shows average validation result over 100 episodes. Shaded areas are standard deviation across 5 seeds.

Agents		taxi		2 agent pursuit		3 agent pursuit	
		19x19	*25x25*	*16x16 (r=1)*	*19x19 (r=1)*	*10x10 (r=1)*	*16x16 (r=2)*
Dynamic Termination	$\delta = 0.1$	**7.89**	**5.75**	**10.24**	**9.30**	**6.71**	**10.38**
	$\delta = 0$	6.58	3.28	6.73	4.07	5.53	6.54
Greedy Termination		6.62	3.23	7.36	3.74	5.89	6.40
Option Termination		-0.32	-0.94	5.47	-1.82	-3.77	5.20
IQL	$\delta = 0.1$	7.11	5.09	-1.57	-2.29	-1.62	-0.59
	greedy	6.08	2.79	-2.12	-2.49	-2.13	-0.42

Table 1: Average reward after training for Taxi and Pursuit. NxN denotes grid-world size, k agent pursuit is the required number of agents for capture, and r is the capture range.

3 DELAYED COMMUNICATION

A straight forward application of decentralized multi-agent approaches like *independent Q-learning* (IQL) [12] to the Options framework [11] would yield agents that make decisions independent of each other: every agent would estimate the Q-value of its available options with Equation 1. Here other agents are treated as stationary parts of the environment, which can lead to unstable training when those agents change their policy. The best way to avoid this instability would be learn the Q-value w.r.t. the joint option of all agents $\mathbf{o}_t := (o_t^1, \ldots, o_t^n)$, i.e. $Q_{\text{joint}}(s_t, \mathbf{o}_t)$ [8]. We propose to use a *delayed communication channel* over which agents signal the new option they switched to after each termination. This approach allows each agent j access to all other agents' options of the previous time step $\mathbf{o}_{t-1}^{-j} := (o_{t-1}^1, \ldots, o_{t-1}^{j-1}, o_{t-1}^{j+1}, \ldots, o_{t-1}^n)$. Agents can approximate the joint Q-value by conditioning on this information, that is, by choosing options o_t^j that maximize the *delayed Q-value* $Q^j(s_t, \mathbf{o}_{t-1}^{-j}, o_t^j)$. In this way, we can avoid maximization over joint options and reduce potentially costly communication.

In our decentralized multi-agent options model, agent j selects an option according to $Q^j(s_t, \mathbf{o}_{t-1}^{-j}, o^j)$, which is defined as:

$$\begin{aligned} Q^j(s_t, \mathbf{o}_{t-1}^{-j}, o^j) &:= \mathbb{E}\left[r_t + \gamma U^j(s_{t+1}, \mathbf{o}_t^{-j}, o^j)\right] \\ U^j(s_{t+1}, \mathbf{o}_t^{-j}, o^j) &:= \left(1 - \beta^{o^j}(s_{t+1})\right) Q^j(s_{t+1}, \mathbf{o}_t^{-j}, o^j) \\ &\quad + \beta^{o^j}(s_{t+1}) \max_{o'^j \in O^j} Q^j(s_{t+1}, \mathbf{o}_t^{-j}, o'^j)\,. \end{aligned} \tag{2}$$

Note that if $\mathbf{o}_t^{-j} = \mathbf{o}_{t-1}^{-j}$, that is, if no *other* agent changes their behaviour, the approximation of $Q_{\text{joint}}(s_t, \mathbf{o}_t)$ is perfect. In line with Intra-Option Learning [11], we update the Q-values of all options o^j that would have executed the same action a_t^j as the actually executed option o_t^j, which vastly improves sample efficiency.

4 DYNAMIC OPTION TERMINATION

Although greedy termination has been shown to improve performance of single agents with imperfect options [3], the agent's behaviour will become less predictable for others. In particular agents that utilize the delayed Q-value of Equation 2 will make sub-optimal decisions whenever another agent terminates. To increase the predictability of agents, while allowing them to terminate flexibly when the task demands it, we propose to put a price δ on the decision to terminate the current option. Option termination is therefore no longer hard-coded, but becomes part of the agent's policy. This can be represented by an additional option $o^j = T$ for agent j to terminate. Note that, unlike in the Options framework, we do not need a termination function $\beta^{o^j}(s_t)$ for each option o^j. It is enough to compare the value of the previous option $Q^j(s_t, \mathbf{o}_{t-1}^{-j}, o_{t-1}^j)$ with the value of choosing a new option $Q^j(s_t, \mathbf{o}_{t-1}^{-j}, T)$, as shown in our novel *dynamic termination Bellman equation* :

$$Q^j(s_t, \mathbf{o}_{t-1}^{-j}, o^j) := \begin{cases} \mathbb{E}\left[r_t + \gamma \max\limits_{o'^j \in \{o^j, T\}} Q^j(s_{t+1}, \mathbf{o}_t^{-j}, o'^j)\right] & , o^j \neq T \\ \max\limits_{o'^j \in O^j} Q^j(s_t, \mathbf{o}_{t-1}^{-j}, o'^j) - \delta & , o^j = T \end{cases}. \tag{3}$$

Similarly to Equation 2, Equation 3 allows intra-option learning and can be applied to all options o^j that would have selected the same action as the executed option o_t^j. Note that the termination option T can always be updated, as it does not depend on the transition.

5 EXPERIMENTS AND RESULTS

We test our approach on standard multi-agent taxi pickup and pursuit tasks. In each episode, m passengers (preys) are randomly distributed in a NxN grid-world. The agents will receive a full observation of the grid-world, and (except for IQL agents) also the last executed options of other agents. In the taxi pickup task, an agent receives a reward of 1 by occupying the same grid as the passenger. In the pursuit task, k agents need to occupy grids next to a prey to capture it and each participating agent will receive a reward of 1. In both tasks, each step incurs a cost of -0.01, and options are defined as going towards equidistant landmarks of distance $L = 3$. We compare our dynamic terminating agent with four baseline methods: the option termination agent using the original termination conditions, greedy termination (as introduced in Sec. 2), and IQL agents with dynamic or greedy termination schemes. All agents are trained off-policy using the exploration policy with random termination probability $\rho = 0.5$ and we share the parameters among the decentralized agents.

Figure 1(a) shows the results from the taxi pickup task. The option termination agent fails due to its inflexibility to switch options. In contrast, our dynamic ($\delta = 0.15$) agent is highly flexible. Moreover compared with greedy and IQL, its high predictability indeed helps the agents to interpret others' intentions and better distribute their target passengers. Figure 1(b) shows the results on the pursuit task, where at least two agents need to surround a prey within capture range = 1. Seen from the IQL agents' low performance, option broadcasting and interpreting others' behaviours are crucial to this task. Our dynamic termination agent ($\delta = 0.1$) significantly outperforms all other agents. Compared with the greedy agents, we can conclude that predictability significantly helps our dynamic agents to stay committed and succeed in cooperation. Finally, we present the performance of all agents across different tasks and varying parameters in Table 1.

REFERENCES

[1] Jakob Foerster, Ioannis Alexandros Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*. 2137–2145.

[2] Mihalis Giannakis and Michalis Louis. 2016. A multi-agent based system with big data processing for enhanced supply chain agility. *Journal of Enterprise Information Management* 29, 5 (2016), 706–727.

[3] Anna Harutyunyan, Peter Vrancx, Pierre-Luc Bacon, Doina Precup, and Ann Nowe. 2017. Learning with options that terminate off-policy. *arXiv preprint arXiv:1711.03817* (2017).

[4] Victor Lesser, Charles L Ortiz Jr, and Milind Tambe. 2012. *Distributed sensor networks: A multiagent perspective*. Vol. 9. Springer Science & Business Media.

[5] Kaixiang Lin, Renyu Zhao, Zhe Xu, and Jiayu Zhou. 2018. Efficient Large-Scale Fleet Management via Multi-Agent Deep Reinforcement Learning. *arXiv preprint arXiv:1802.06444* (2018).

[6] Rajbala Makar, Sridhar Mahadevan, and Mohammad Ghavamzadeh. 2001. Hierarchical multi-agent reinforcement learning. In *Proceedings of the fifth international conference on Autonomous agents*. ACM, 246–253.

[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] Martin Riedmiller and Daniel Withopf. 2005. Effective Methods for Reinforcement Learning in Large Multi-Agent Domains (Leistungsfähige Verfahren für das Reinforcement Lernen in komplexen Multi-Agenten-Umgebungen). *it-Information Technology* 47, 5 (2005), 241–249.

[9] Peter Stone and Manuela Veloso. 2000. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots* 8, 3 (2000), 345–383.

[10] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

[11] Richard S Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence* 112, 1-2 (1999), 181–211.

[12] Ming Tan. 1998. Readings in Agents. Chapter Multi-agent Reinforcement Learning: Independent vs. Cooperative Agents, 487–494.

[13] Michael Wooldridge. 2009. *An introduction to multiagent systems*. John Wiley & Sons.