Reinforcement Learning with Option Machines (EA)

Floris den Hengst[1 2], Vincent François-Lavet[2], Mark Hoogendoor[2], and Frank van Harmelen[2]

[1]ING Bank Floris.den.Hengst@ing.com
[2]Vrije Universiteit Amsterdam, {F.den.Hengst,Vincent.FrancoisLavet, M.Hoogendoorn, Frank.van.Harmelen}@vu.nl

1 Introduction

Sample efficiency remains an open challenge in RL and prevents adoption in many real-world settings [5,8]. Recently, approaches that combine low-level RL with high-level symbolic instructions have become popular as this plays both to their strengths. RL is used to learn primitive actions in ground states whereas symbolic approaches are used to reason over subgoals, groups of states and multi-timestep actions. Existing approaches, however, typically require instructions to fully define the task at hand, i.e. that high returns are *always* obtained if the instructions are followed. Such rich instructions, however, may be hard to attain in practice.

We target a setting in which an agent is to optimize an environment reward with the help of *underspecified* instructions. These instructions convey some information about solving the task, but they do not not guarantee a high environment reward. In [9], we propose and evaluate a framework for sample-efficient RL with these instructions.

Our framework provides interleaving, looping and partial ordering of multi-timestep actions know as options, in contrast to some existing extensions to options [16,1,4]. It abstracts over actions, in contrast to some work on RL and transition systems [14,10,11]. Other work assumes a complete and possibly hard to obtain task specification, and/or do not use *named* options and are less reusable and interpretable [15,6,12,2,19,20,3,7].

2 The Option Machine Framework

We address problems formalized as Markov decision processes (MDPs) with the option framework [17,18]. This framework introduces a temporal abstraction over actions, in which the agent selects a 'primitive' action or 'multi-step' action at each time step. We combine options with transition systems known as Finite State Transducers (FSTs). FSTs are transition systems that map strings of inputs to strings of output. An FST can be specified in a temporal logic and then converted to a FST with existing tools [13].

Our approach uses high-level instructions composed of *options* and *events*. Events are formalized as a set of atomic propositions AP^I. We assume that some function $L : S \to 2^{AP^I}$ for detecting these events in MDP states is available as a handcrafted or pretrained component. Options are actions that take multiple time steps and can be reused across tasks. In our framework, all options are *named*, i.e. associated with an atomic proposition $ap \in AP^O$. The instructions in this work define which options are permissible based on the detected events.

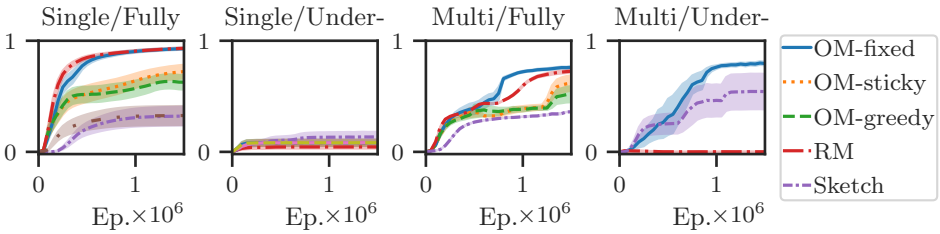

Fig. 1: Rewards in the single- and multi-task settings on fully and underspecified environments.

		Option Machines		
Environment	Sketch	Fixed	Greedy	Sticky
Fully specified	0.03	**0.90**	0.74	0.73
Underspecified	0.09	**0.60**	N/A	N/A

Table 1: Zero-shot results on 1K test episodes.

In particular, at each time point t with events $l_t := L(s_t)$, the FST transitions to a new state q_t based on its current state q_{t-1} and events l_t. Upon this transition, the FST outputs a subset $AP_t^O \subseteq AP^O$ of all options. This subset is interpreted as the set of permissible options at time point t. The agent then selects one of these and executes its policy to select an atomic action. We compare ways of selecting from the permissible options and two of these propagate learned knowledge to the symbolic layer. The two-level control structure generates trajectories by executing different policies at different time steps and introduces the problem of properly attributing subsequences to options during learning. We use FST state information to address this problem in the proposed learning algorithm by assigning the entire subsequence generated while visiting a single FST state to the latest active policy for any subsequence in which a particular FST state was visited. Additionally, we use FST state information to define a shaping reward similar to the Reward Machines (RM) approach [3].

3 Evaluation and Discussion

We evaluate the proposed approach in two benchmark environments that have one key difference: every trajectory that satisfies the specification yields high return in the 'fully specified' environment whereas in the 'underspecified' environment only some trajectories do so. We evaluate the approach in single-task, multi-task and zero-shot settings in these environments. Our approach significantly outperforms policy sketches [1] in all settings and RMs in the multi-task setting, see Figure 1 and Table 1. Shaping rewards were only supplied in the fully specified environment as we found that shaping rewards should not be used in the underspecified case. These results indicate that with the OM framework, agents are capable of learning named long-term behaviors based on task specifications and environment reward only. This enables reuse within and across tasks, facilitates zero-shot generalisation if solutions to subtasks are known and enables the inspection of the agents' decision-making process at an interpretable level of abstraction.

References

1. Andreas, J., Klein, D., Levine, S.: Modular multitask reinforcement learning with policy sketches. In: ICML. pp. 166–175. PMLR (2017)
2. Brafman, R., De Giacomo, G., Patrizi, F.: Ltlf/ldlf non-markovian rewards. In: AAAI. vol. 32. AAAI (2018)
3. Camacho, A., Icarte, R., Klassen, T., Valenzano, R., McIlraith, S.: Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In: IJCAI. pp. 6065–6073 (2019)
4. Dietterich, T.: Hierarchical reinforcement learning with the maxq value function decomposition. JAIR **13**, 227–303 (2000)
5. Dulac-Arnold, G., Mankowitz, D., Hester, T.: Challenges of real-world reinforcement learning. arXiv:1904.12901 (2019)
6. Fu, J., Topcu, U.: Probably approximately correct mdp learning and control with temporal logic constraints. In: Robotics: Science and Systems. No. 10 (2014)
7. den Hengst, F., François-Lavet, V., Hoogendoorn, M., van Harmelen, F.: Planning for potential: efficient safe reinforcement learning. Machine Learning pp. 1–20 (2022)
8. den Hengst, F., Grua, E., El Hassouni, A., Hoogendoorn, M.: Reinforcement learning for personalization: A systematic literature review. Data Science **3**(2), 107–147 (2020)
9. den Hengst, F., Francois-Lavet, V., Hoogendoorn, M., van Harmelen, F.: Reinforcement learning with option machines. In: Raedt, L.D. (ed.) Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22. pp. 2909–2915. International Joint Conferences on Artificial Intelligence Organization (7 2022). https://doi.org/10.24963/ijcai.2022/403, https://doi.org/10.24963/ijcai.2022/403, main Track
10. Leonetti, M., Iocchi, L., Patrizi, F.: Automatic generation and learning of finite-state controllers. In: AIMSA. pp. 135–144. Springer (2012)
11. Leonetti, M., Iocchi, L., Stone, P.: A synthesis of automated planning and reinforcement learning for efficient, robust decision-making. Artificial Intelligence **241**, 103–130 (2016)
12. Li, X., Vasile, C., Belta, C.: Reinforcement learning with temporal logic rewards. In: IROS. pp. 3834–3839. IEEE/RSJ (2017)
13. Michaud, T., Colange, M.: Reactive synthesis from ltl specification with spot. In: 7th Workshop on Synthesis, SYNT@ CAV (2018)
14. Parr, R., Russell, S.: Reinforcement learning with rhierarchies of machines. NIPS pp. 1043–1049 (1998)
15. Sadigh, D., Kim, E., Coogan, S., Sastry, S., Seshia, S.: A learning based approach to control synthesis of markov decision processes for linear temporal logic specifications. In: CDC. pp. 1091–1096. IEEE (2014)
16. Singh, S.: Transfer of learning by composing solutions of elemental sequential tasks. Machine Learning **8**(3), 323–339 (1992)
17. Sutton, R., Barto, A.: Reinforcement learning: An introduction. MIT press (2018)
18. Sutton, R., Precup, D., Singh, S.: Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence **112**(1-2), 181–211 (1999)
19. Toro Icarte, R., Klassen, T., Valenzano, R., McIlraith, S.: Teaching multiple tasks to an rl agent using ltl. In: AAMAS. pp. 452–461 (2018)
20. Toro Icarte, R., Klassen, T., Valenzano, R., McIlraith, S.: Using reward machines for high-level task specification and decomposition in reinforcement learning. In: ICML. vol. 35, pp. 2107–2116. PMLR (2018)