Hierarchical Reinforcement Learning for Robots in POMDPs with Option-Observation Initiation Sets

Denis Steckelmacher Hélène Plisnier
Diederik M. Roijers
Ann Nowé

Vrije Universiteit Brussel, Brussels, Belgium

1 INTRODUCTION

Many robotic reinforcement learning problems have a hierarchical nature, and often exhibit some degree of partial observability. While hierarchy and partial observability are usually tackled separately (for instance by combining recurrent neural networks and options), we show on a complex robotic task that addressing both problems simultaneously is simpler and more efficient. More specifically, we build on the well-known Options framework, widely used on complex but fully-observable tasks, and address partial observability by making the initiation set of options conditional on the previously-executed option. Options with such Option-Observation Initiation Sets (OOIs) are at least as expressive as Finite State Controllers, a state-of-the-art approach for learning in POMDPs, as proven in [1].

Our experiment emphasizes the three main properties of Options with OOIs: complex tasks in POMDPs can be solved to expert level, designing OOIs require only minimal effort, and options with OOIs are more than three times more sample-efficient than a recurrent neural network over options, thereby largely offsetting the time needed to design OOIs when physical robots are used.

2 RELATED WORK

Two significant challenges in reinforcement learning are complex long-running tasks and partial observability. Options address the first challenge by factoring a complex task into simpler sub-tasks [2]. The agent learns a top-level policy that repeatedly selects options, that in turn execute sequences of actions before returning [3]. The second challenge, partial observability, is addressed by maintaining a belief of what the agent thinks the full state is [4], storing information in an external memory for later reuse [5, 6], or using recurrent neural networks (RNNs) to allow information to flow between time-steps [7].

Combined solutions to the above two challenges have recently been designed for planning [8], but solutions for learning algorithms are not yet ideal. HQ-Learning decomposes a task into a sequence of fully-observable subtasks [9], which precludes cyclic tasks from being solved. Using recurrent neural networks in options and for the top-level policy [10] addresses both challenges, but brings in the design complexity and approximate memory of RNNs [11, 12]. OOIs allow challenging hierarchical partially observable tasks to be solved, even if they are repetitive, while keeping the top-level and option policies memoryless, thus easy to learn and to implement.

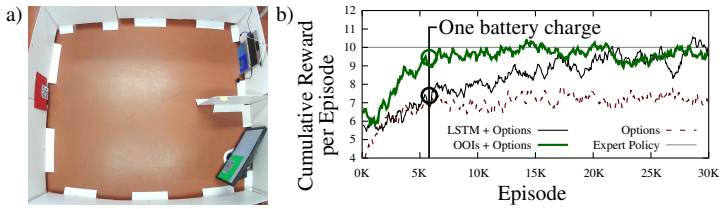

Figure 1: a) Experimental setup. b) Cumulative reward per episode for different agents (*OOIs + Options* is ours). Our agent matches the expert policy after 8K episodes, while LSTM over options needs more than 25K episodes.

3 OPTION–OBSERVATION INITIATION SETS

Descriptions of partially observable tasks in natural language often contain allusions at sub-tasks that must be sequenced or cycled through, possibly with branches. This is easily mapped to a policy over options and sets of options that may or may not follow each other (an example is given in the next section). Our main contribution, Option-Observation Initiation Sets (OOIs), allow to describe which options may follow which ones. We define the initiation set I_ω of option ω so that the set $\mathcal{O}_t$ of options available at time t depends on the observation x_t and previously-executed option ω_{t-1}:

$$I_\omega \subseteq \Omega \times (O \cup \{\emptyset\})$$
$$\mathcal{O}_t \equiv \{\omega \in O : (x_t, \omega_{t-1}) \in I_\omega\}$$

with $\omega_0 = \emptyset$, Ω the set of observations and O the set of options. $\mathcal{O}_t$ allows the agent to condition the option selected at time t on the one that has just terminated, even if the top-level policy does not observe ω_{t-1}. The top-level and option policies remain memoryless, which facilitates learning and simplifies the design of option policies, as shown in the next section.

4 EXPERIMENT

A Khepera III robot has to gather objects from two terminals, separated by a wall, and to bring them to the root, as shown in Figure 1 (a). Objects have to be gathered one by one from a terminal until it becomes empty. When a terminal is emptied, the other one is automatically refilled with 2 to 4 objects (selected at random). The robot therefore has to alternatively gather objects from both terminals. The episode finishes after the terminals have been emptied 2 or 3 times. The root is marked by a paper QR-code encoding 1. Each terminal has a screen displaying its color and a 1 QR-code when full (reward of +2 when visited by the robot), 2 when empty (reward of -2). A wireless camera, mounted on top of the robot, detects color blobs and can read QR-codes from a short distance. This makes the environment partially observable, as the robot cannot observe the state of the terminals from the root, when it needs to decide where to go next.

12 fixed options allow the robot to move towards colored objects, 4 options per color (red, green or blue). Having several options sharing the same policy allows them to encode memory, without encoding the solution of the problem in the option set. For instance, the agent learns to go back to the root using one option when the blue terminal is empty, another one when it is full. At the root, the option that just terminated, combined with OOIs, allows the agent to select its next destination based on past observations [1]. The robot learns a policy over these options that solves the task, which shows that it is able to leverage the OOIs to encode memory (see Figure 1, b). An LSTM over the same 12 options, but without OOIs, needs three times more time before learning a policy of comparable quality. On an actual Khepera robot, learning with OOIs takes about 4 hours, which corresponds to a single battery charge, while an LSTM over options takes more than 12 hours to learn.

REFERENCES

[1] Denis Steckelmacher, Diederik M. Roijers, Anna Harutyunyan, Peter Vrancx, and Ann Nowé. Reinforcement learning in POMDPs with memoryless options and option-observation initiation sets. *CoRR*, abs/1708.06551, 2017.

[2] Doina Precup. *Temporal Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts, 2000.

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

[4] A Cassandra, L Kaelbling, and M Littman. Acting optimaly in partially observable stochastic domains. In *Proceedings of the 12th AAAI National Conference on Artificial Intelligence*, volume 2, pages 1023–1028, 1994.

[5] Leonid Peshkin, Nicolas Meuleau, and Leslie Kaelbling. Learning policies with external memory. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 307–314, 1999.

[6] Wojciech Zaremba and Ilya Sutskever. Reinforcement Learning Neural Turing Machines. *CoRR*, abs/1505.00521, 2015.

[7] Bram Bakker. Reinforcement learning with long short-term memory. In *Advances in Neural Information Processing Systems (NIPS)*, volume 14, 2001.

[8] Ruijie He, Emma Brunskill, and Nicholas Roy. Efficient planning under uncertainty with macro-actions. *Journal of Artificial Intelligence Research (JAIR)*, 40:523–570, 2011.

[9] Marco Wiering and Jürgen Schmidhuber. HQ-Learning. *Adaptive Behavior*, 6(2):219–246, 1997.

[10] Mohan Sridharan, Jeremy Wyatt, and Richard Dearden. Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs. *Artificial Intelligence*, 174:704–725, 2010.

[11] Rafal Józefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2342–2350, 2015.

[12] Sepp Hochreiter and Jurgen Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1–32, 1997.