Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning

Richard S. Sutton, Doina Precup, Satinder Singh
Motivation

- Learning, planning, and representing knowledge at **multiple levels of temporal abstraction** are longstanding challenges for AI.
- Many real-world decision-making problems admit hierarchical temporal structures:
  - Example: planning for a trip
  - Enable simple and efficient planning
- This paper: how to automate the ability to plan and work flexibly with multiple time scales?
This paper

- Temporal abstraction within the framework of RL and MDP using **options**
  - Enable **temporally extended actions** and planning with **temporally abstract knowledge**

- Benefits
  - MDPs + options = semi-MDPs: standard results for SMDPs apply!
  - Knowledge transfer: use domain knowledge to define options, solutions to sub-goals can be reused
  - Possibly more efficient learning and planning
MDPs
- At each time step $t = 0, 1, \ldots$
  - Perceive state of environment $s_t \in S$
  - Select an action $a_t \in A$
  - One-step state-transition probability $p_{s,s'} = P(s_{t+1} = s'|s_t = s, a_t = a)$
  - At $t + 1$, receive reward $r_{t+1}$ and observe the new state $s_{t+1}$
- The goal is to learn a Markov policy $\pi : S \times A \rightarrow [0, 1]$ that maximizes the expected discounted future rewards from each state:
  $$V^\pi(s) = E[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots | s_t = s, \pi]$$

Semi-MDPs
- State transitions and control selections at discrete times, but the time between successive control choices is variable
- Allows for temporally extended courses of actions and Markovian at the level of decision points
- However, temporally extended actions are treated as indivisible and unknown units
Options

- Goal: generalize primitive actions to include temporally extended courses of actions with internally divisible units
- An option \((I, \pi, \beta)\) has three components:
  - A policy \(\pi : S \times A \rightarrow [0, 1]\)
  - A termination condition \(\beta : S^+ \rightarrow [0, 1]\)
  - An initiation set \(I \subseteq S\)
- If option \((I, \pi, \beta)\) is taken at \(s \in I\), then actions are selected according to \(\pi\) until the option terminates stochastically according to \(\beta\)
- **Markov option**: within an option, policies and termination conditions depend on the current state
- **Semi-Markov option**: policies and termination conditions may depend on all prior event since the option was initiated
MDP + Options = Semi-MDP!

**Theorem:** For any MDP and any set of options defined on that MDP, the decision process that selects only among those options and executing each to termination is an semi-MDP

**Implications:**
- This relationship among MDPs, options, and semi-MDPs provides a basis for the theory of planning and learning methods with options
- i.e. MDPs + Options are more flexible compared to conventional semi-MDP, but standard results for semi-MDPs can be applied to analyze MDPs with options
Semi-MDP Dynamics
Semi-MDP Dynamics

- From $A$ to $O$
Semi-MDP Dynamics

- From $A$ to $O$
- From one-step to (stochastic) $k$-step
Semi-MDP Dynamics

- From $A$ to $O$
- From one-step to (stochastic) $k$-step

$$r_s^o = E \left\{ r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{k-1} r_{t+k} \mid \mathcal{E}(o, s, t) \right\}$$

$$p_{ss'}^o = \sum_{k=1}^{\infty} p(s', k) \gamma^k$$
Semi-MDP Infrastructure - this looks familiar...

$$V^\mu(s) = E \{ r_{t+1} + \cdots + \gamma^{k-1}r_{t+k} + \gamma^k V^\mu(s_{t+k}) \mid \mathcal{E}(\mu, s, t) \}$$

(where $k$ is the duration of the first option selected by $\mu$)

$$= \sum_{o \in \mathcal{O}_s} \mu(s, o) \left[ r_s^o + \sum_{s'} p_{ss'}^o V^\mu(s') \right],$$
V^\mu(s) = E\{r_{t+1} + \cdots + \gamma^{k-1}r_{t+k} + \gamma^k V^\mu(s_{t+k}) \mid \mathcal{E}(\mu, s, t)\}

(where k is the duration of the first option selected by \mu)

= \sum_{o \in \mathcal{O}_s} \mu(s, o) \left[ r_s^o + \sum_{s'} p_{ss'}^o V^\mu(s') \right],

V^*_o(s) \overset{\text{def}}{=} \max_{o \in \mathcal{O}_s} \left[ r_s^o + \sum_{s'} p_{ss'}^o V^*_o(s') \right]
Semi-MDP Infrastructure - this looks familiar...

\[ V^\mu(s) = E\{r_{t+1} + \cdots + \gamma^{k-1}r_{t+k} + \gamma^k V^\mu(s_{t+k}) \mid \mathcal{E}(\mu, s, t)\} \]

(where \( k \) is the duration of the first option selected by \( \mu \))

\[ = \sum_{o \in O_s} \mu(s, o) \left[ r_s^o + \sum_{s'} p_{ss'}^o V^\mu(s') \right], \]

\[ V^*_O(s) \stackrel{\text{def}}{=} \max_{o \in O_s} \left[ r_s^o + \sum_{s'} p_{ss'}^o V^*_O(s') \right] \]

Allows for planning & learning analogously to in MDPs!
Example of one option’s policy:

4 stochastic primitive actions
- up
- left (fails 33% of the time)
- right
- down

8 multi-step options (to each room’s 2 hallways)
Between MDPs and Semi-MDPs...

- Interrupting options
- Intra-option model / value learning
- Subgoals

Open up the black-box when Option is Markov!
I. Interrupting options

- Don’t have to follow options to termination!
- At time t, if continue with o:

\[ Q^\mu(s_t, o) \]

If select new option:

\[ V^\mu(s_t) = \sum_{o'} \mu(s_t, o') Q^\mu(s_t, o') \]

- Policy \( \mu \rightarrow \mu' \)  Interrupted Policy

- For all s,

\[ V^{\mu'}(s) \geq V^\mu(s) \]
Landmark example
II. Intra-option **model** learning

Given \( o = (I, \pi, \beta) \), learn model \( r_s^o, p_{s,s'}^o \).

**Intra-option value** learning

Given \( o = (I, \pi, \beta) \), \( r_s^o, p_{s,s'}^o \), learn value function \( Q_\mathcal{O}^*(s, o) \).

- Take an action, update estimates for all **consistent** options.
# SMDP-Learning vs. Intra-option Learning

<table>
<thead>
<tr>
<th></th>
<th>SMDP</th>
<th>Intra-option Learning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Update only when option terminates</td>
<td>Update after each action (Learn from fragments of experience)</td>
<td></td>
</tr>
<tr>
<td>Update 1 option at a time</td>
<td>Update all options consistent with current action (off-policy, can learn never-selected options)</td>
<td></td>
</tr>
<tr>
<td>Semi-Markov options</td>
<td>Only Markov options</td>
<td></td>
</tr>
</tbody>
</table>
III. Learning options for subgoals

- Can we learn the policy that determines an option?
  - Yes: add terminal subgoal rewards
  - Perform Q-learning to adapt policies towards achieving subgoals
  - Subgoals + rewards must still be given
Conclusion

● **Strengths**
  ○ General framework for reinforcement learning at different levels of temporal abstraction
  ○ Mimics real-world setting of sub-tasks and sub-goals
  ○ Same formulations and algorithms apply across levels
  ○ “Efficiency” in planning

● **Weaknesses**
  ○ Domain knowledge required to formalize options/subgoals
  ○ Options may not generalize well across environments
  ○ Might necessitate a small state-action space
Questions + Discussion

● How does the temporal abstraction framework relate to meta-learning?
● Can you imagine environments for which this framework cannot be applied in a straightforward way, or for which adopting this framework might be disadvantageous?
  ○ What if the state that we observe is a noisy version of the actual state? Are options still useful in the partially-observable setting?
● Hierarchical abstraction for both state space and action space?
● Possible extensions for intra-option learning:
  ○ Use reweighting to learn about inconsistent options?
  ○ Concept of consistency between option and action for stochastic options?