Computer Engineering Department

Hierarchical RL

Mohammad Hossein Rohban, Ph.D.

Spring 2025

Courtesy: Most of slides are adopted from CS 330 Stanford.

What is Temporal Abstraction? - Motivating Example

- Consider an activity such as cooking
  - **High-level**: Choose a recipe, make grocery List
  - **Medium-level**: get a pot, put ingredients in the Pot, stir until smooth
  - **Low-level**: wrist and arm movement, muscle Contraction
- All have to be seamlessly integrated.

Temporal Abstraction in AI

- Temporal Abstractions is not specific to RL, it has a long root in AI.
- It has been shown to:
  - Generate shorter plans
  - Reduce the complexity of choosing actions
  - Provide robustness against model misspecification
  - Allow taking shortcuts in the environment
  - Improves interpretability

Advantages in Complex RL Tasks

Advantages to planning

- Need to generate shorter plans
- Improves robustness to model errors
- Might need to look at fewer states, since the abstract actions have pre-defined termination conditions
- Discretize the action space in continuous problems

Advantages to learning

- Improves exploration (can travel in larger leaps)
- Gives a natural way of using a single stream of data to learn many things (off-policy learning)

Advantages to interpretability

- Focusing attention: Sub-plans ignore a lot of information - Improves readability of both models and resulting plans - Reduces the problem size

Procedural, Temporally Abstract knowledge: Options

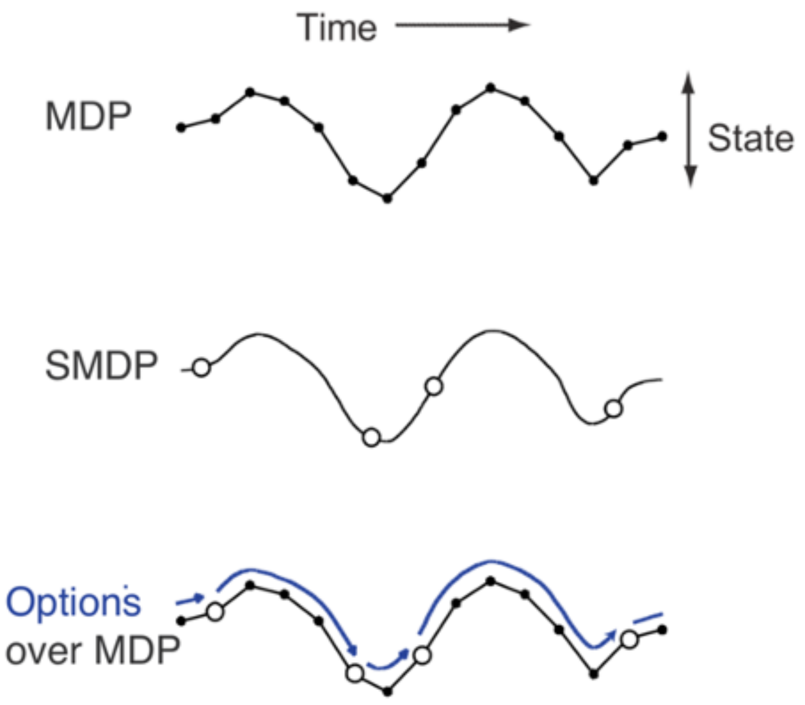
- Generalize actions to include temporally extended courses of actions.
- An option $\omega = (I, \pi, \beta)$ has three components:
  - An initiation set $I \subseteq S$
  - A terminations condition $\beta : S \rightarrow [0,1]$
  - A policy $\pi: S \times \mathcal{A} \rightarrow [0,1]$
- If the option (I, π, β) is taken at $s \in I$, then actions are selected according to π until the option terminates stochastically according to β.

Options - Example

- **Robot Navigation**: If there is no obstacle in front (I), go forward (π) until you get too close to another object (β).

- **Open-the-door**:
  - I: all states in which a closed door is within reach
  - π: pre-defined controller for reaching, grasping, and turning the door knob
  - β: terminate when the door is open

Decision-Making with Options: Semi-MDPs

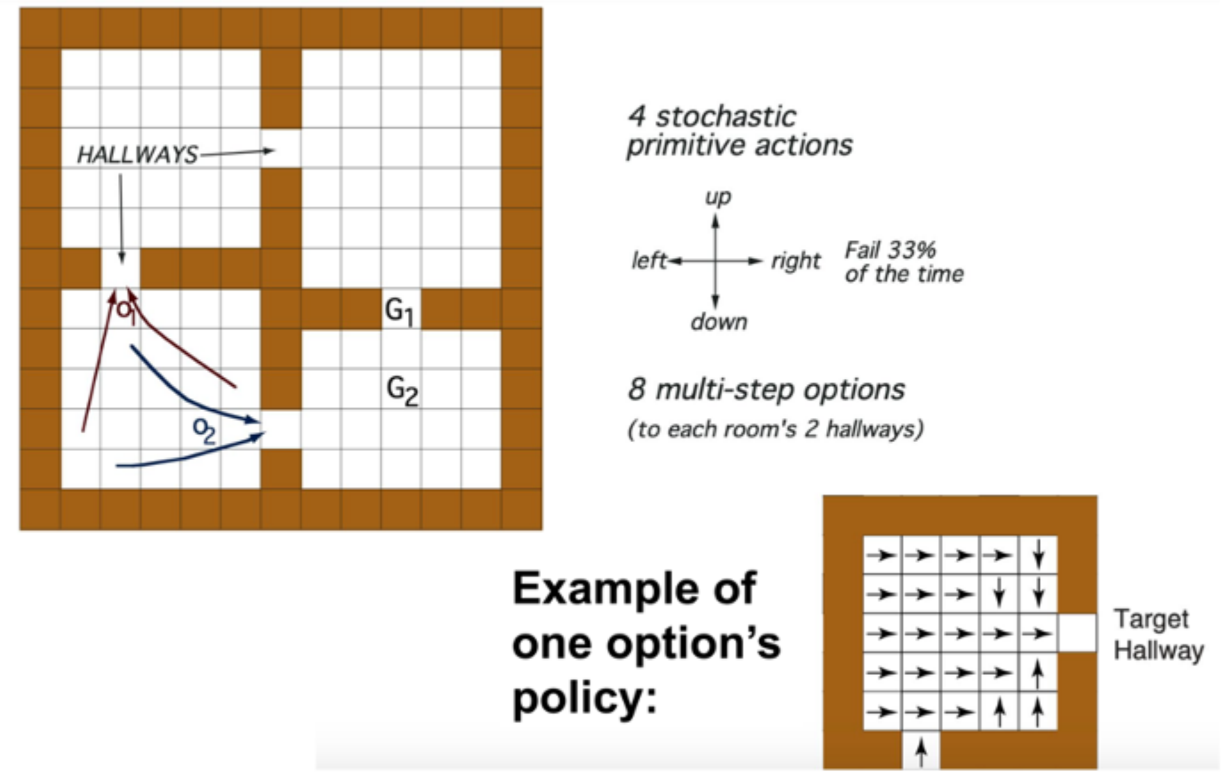


Discrete time
Homogeneous discount

Continuous time
Discrete events
Interval-dependent discount

Discrete time
Overlaid discrete events
Interval-dependent discount

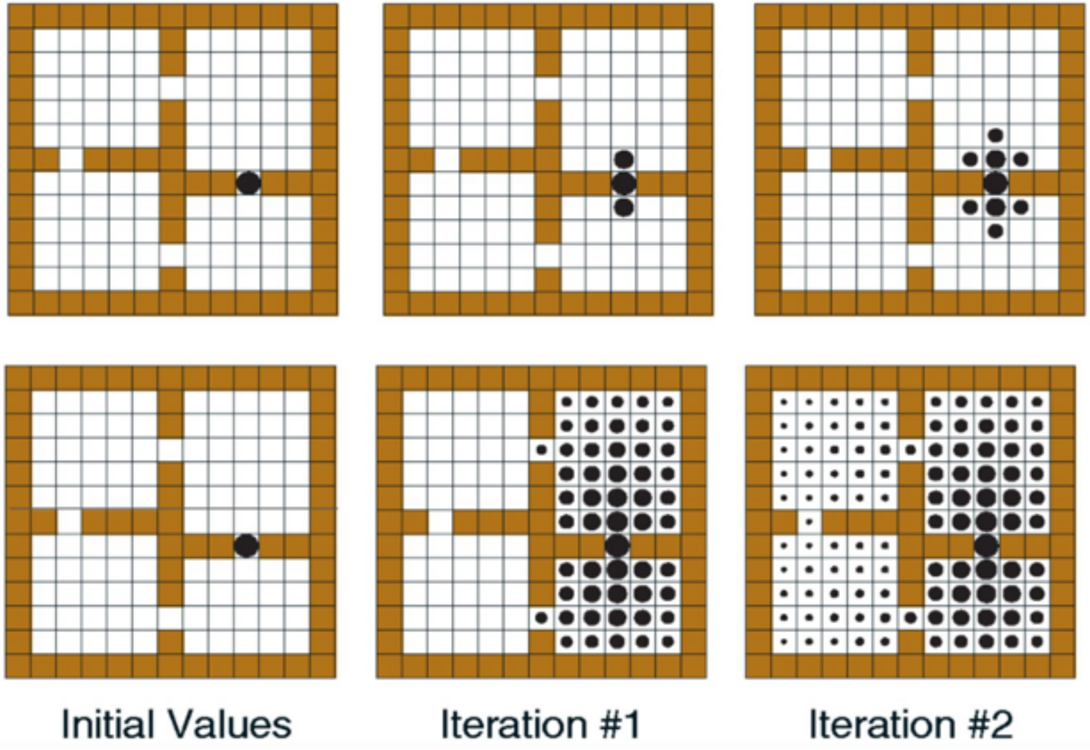
Example: Navigation



Example: Navigation

With cell-to-cell primitive actions

With room-to-room options



Option-critic

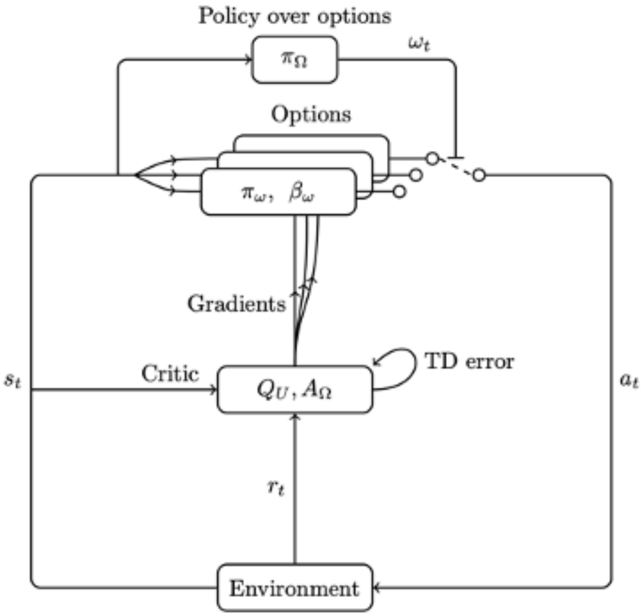
$$Q_\Omega(s,\omega) = \sum_a \pi_{\omega,\theta}(a \mid s)\, Q_U(s,\omega,a)$$

$$Q_U(s,\omega,a) = r(s,a) + \gamma \sum_{s'} \mathrm{P}(s' \mid s,a)\, U(\omega,s')$$

$$U(\omega,s') = (1-\beta_{\omega,\vartheta}(s'))Q_\Omega(s',\omega) + \beta_{\omega,\vartheta}(s')V_\Omega(s')$$

Option-critic

Architecture



Algorithm

Algorithm 1: Option-critic with tabular intra-option Q-learning

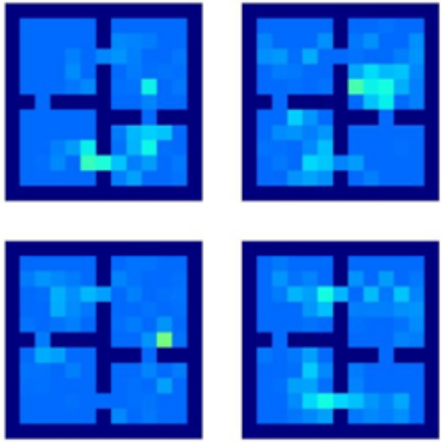
$s \leftarrow s_0$
Choose ω according to an ϵ-soft policy over options $\pi_\Omega(s)$
repeat
  Choose a according to $\pi_{\omega,\theta}(a \mid s)$
  Take action a in s, observe s', r

  1. Options evaluation:
  $\delta \leftarrow r - Q_U(s, \omega, a)$
  if *s' is non-terminal* **then**
    $\delta \leftarrow \delta + \gamma(1 - \beta_{\omega,\vartheta}(s'))Q_\Omega(s', \omega) + \gamma\beta_{\omega,\vartheta}(s') \max_{\bar{\omega}} Q_\Omega(s', \bar{\omega})$
  end
  $Q_U(s, \omega, a) \leftarrow Q_U(s, \omega, a) + \alpha\delta$

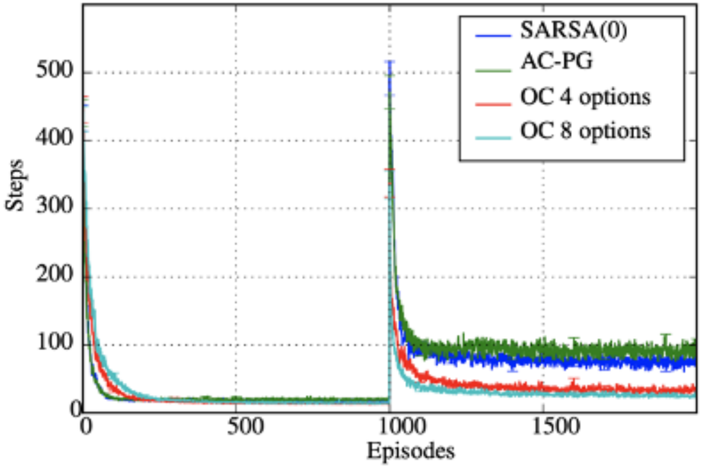
  2. Options improvement:
  $\theta \leftarrow \theta + \alpha_\theta \frac{\partial \log \pi_{\omega,\theta}(a \mid s)}{\partial \theta} Q_U(s, \omega, a)$
  $\vartheta \leftarrow \vartheta - \alpha_\vartheta \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} (Q_\Omega(s', \omega) - V_\Omega(s'))$

  if *$\beta_{\omega,\vartheta}$ terminates in s'* **then**
  choose new ω according to ϵ-soft$(\pi_\Omega(s'))$
  $s \leftarrow s'$
until *s' is terminal*

Experiment 1: Four-Room Navigation

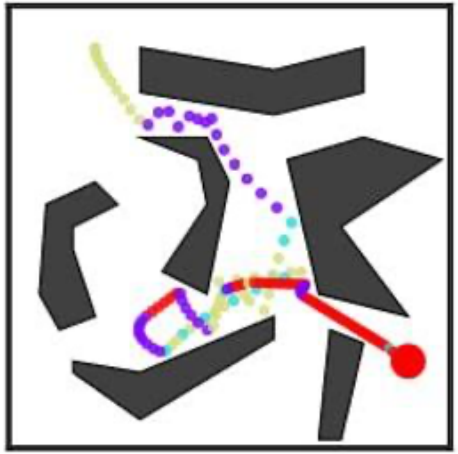


Termination probabilities for the option-critic agent learning with 4 options. The darkest color represents the walls in the environment while lighter colors encode higher termination probabilities.

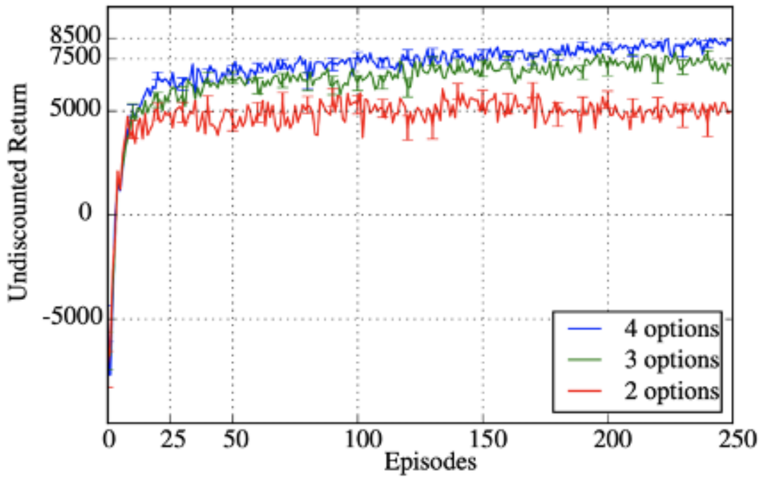


After a 1000 episodes, the goal location in the four-room domain is moved randomly. Option-critic ("OC") recovers faster than the primitive actor-critic ("AC-PG") and SARSA(0). Each line is averaged over 350 runs.

Experiment 2: Pinball



Pinball: Sample trajectory of the solution found after 250 episodes of training using 4 options All options (color-coded) are used by the policy over options in successful trajectories. The initial state is in the top left corner and the goal is in the bottom right one (red circle).



Learning curves in the Pinball domain.