Hierarchical Reinforcement Learning : The Option Framework

Orso Forghieri

Erwan Le Pennec (CMAP)

Hind Castel (SAMOVAR, Telecom SudParis)

Emmanuel Hyon (LIP6, Sorbonne Université)

August 10, 2024

Table of Contents

Introduction

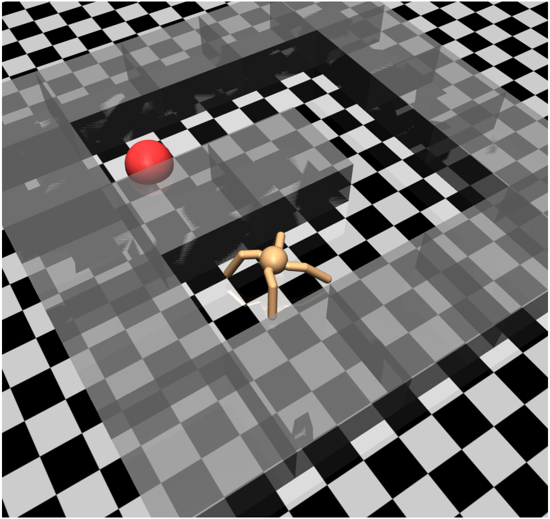

Figure 1: An example of Hierarchical Environment : Ant Maze

Introduction

Figure 2: Richard Sutton (DeepMind), Doina Precup (McGill University) and Satinder Singh (DeepMind)

Introduction [Hutsebaut-Buysse et al., 2022]

Timeline of HRL : :

1. 1999 : Option framework [Sutton et al., 1999]
2. 1997 – 2006 : Problem-specific models (HAM, MAXQ, HEXQ)
3. 2009 – 2018 Goal conditional, options, Deep HRL (Option-Critic, Hiro, DIAYN)

Table of Contents

Table of Contents

Markov Decision Process

Definition (Markov Decision Process)

A Markov Decision Process $\mathcal{M} = (S, A, T, R)$ is given by :

- The state space S
- The action space A
- The transition function $T(s, a, s') = \mathbb{P}[s_{t+1} = s' | s_t = a_t, a_t = a]$
- The reward function $R(s, a) = \mathbb{E}[r_{t+1} | s_t = s, a_t = a]$

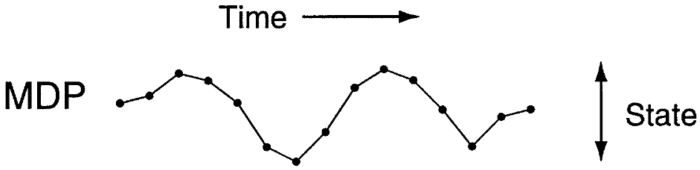

Figure 3: Trajectory of a Markov Decision Process

Temporal Abstraction [Sutton et al., 1999]

Definition (Option)

An option $\omega =< I_\omega, \pi_\omega, \beta_\omega > :$

- can be initialized when $s \in I_\omega \subset S$
- is executed following π_ω
- can be early stopped with probability $\beta_\omega(s)$ along execution

Temporal Abstraction

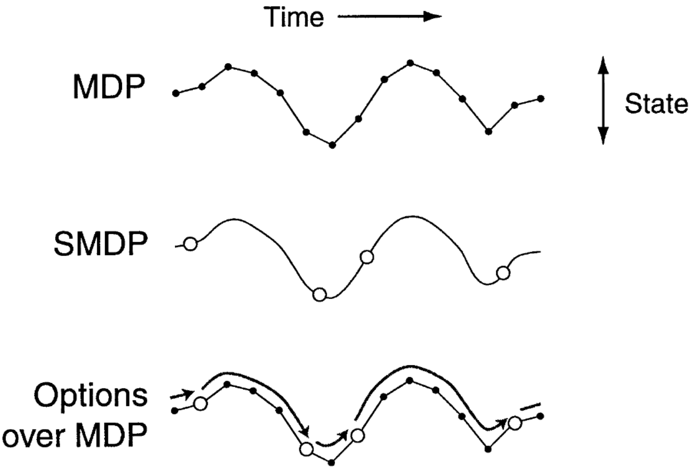

Figure 4: Semi-Markov Decision Process [Sutton et al., 1999]

Remarks on options

For an option o :

- Its policy π_o can be trained relatively to its pseudo-reward r_o [Pateria et al., 2021]
- The early stopping probability can also be trained [Bacon et al., 2017, Xu et al., 2018]

Options : a two-level approach

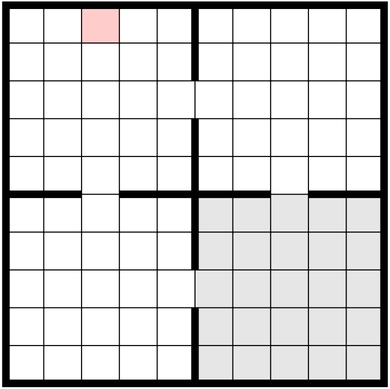

Figure 5: Four rooms example

- $S = [\![0; 9]\!]^2$, $A = 4$
- $\mathbb{P}[s' = (x, y+1) | s = (x, y), a = "N"] = 0.8$

Options : a two-level approach

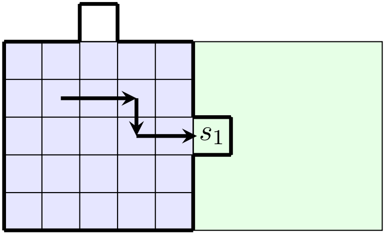

Figure 6: An example of option

Example of State Abstraction

Definition (State Abstraction)

A State Abstraction is a function $\phi : S \mapsto S_\phi$ that maps each $s \in S$ to an abstract state s_ϕ.

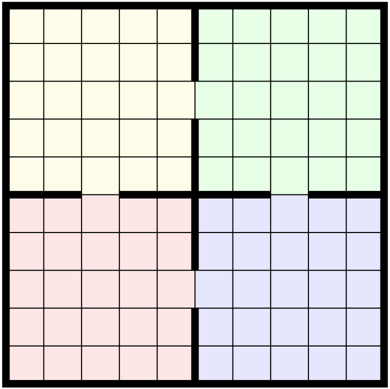

Figure 7: Four rooms example

Remarks on State Abstraction

State Abstraction

- is complementary to options (learned options can be adapted to a given state abstraction) [Pateria et al., 2021]
- implies an approximation on value function detailed in [Abel et al., 2016]

Optimal Bellman Equations

Optimal Bellman Equations are the following

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma V^*(s') \right)$$

with $T(s,a,s') = \mathbb{P}[s_{t+1} = s' | s_t = s, a_t = a]$

$$Q^*(s,a) = \sum_{s'} T(s,a,s') \left(R(s,a,s') + \gamma \max_{a'} Q^*(s',a') \right)$$

Optimal Bellman Equations for Options

Optimal Bellman Equations for Options are the following :

$$V^*(s) = \max_a \sum_{s',N} T(s,a,s',N) \left(R(s,a,s',N) + \gamma^N V^*(s') \right)$$

with $T(s,a,s',N) = \mathbb{P}\left[s_{t+N} = s' | s_t = s, a_t = a\right]$
and $R(s,a,s',N) = \mathbb{E}\left[\sum_{n=0}^{N-1} \gamma^n r_{t+n} \mid s_t = s, a_t = a, s_{t+N} = s'\right]$

$$Q^*(s,a) = \sum_{s',N} T(s,a,s',N) \left(R(s,a,s',N) + \gamma^N \max_{a'} Q^*(s',a') \right)$$

Table of Contents

Temporal Difference Error

Definition (Temporal Difference Error)

Given a step (s_t, a_t, r_t, s_{t+1}) and an approximation V of the value function, we define the temporal difference error by :

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$

$\rightarrow$ Two approaches using the TD-Error :

- Update an approximation of Q (Q-Learning)
- Update an approximation of π and Q (Actor-Critic)

Q-Learning vs Policy Gradient

Theorem (Policy Gradient Theorem)

Approximating the policy by π_θ and given the objective

$$J(\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right]$$

its gradient relatively to θ can be written

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t, s_t) G_t$$

Actor-Critic algorithm

→ **Actor-Critic** with neural networks π_θ and Q_w uses the update

$$\theta \leftarrow \theta + \alpha_\theta \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t, s_t) Q_w(s_t, a_t) \qquad (\pi_\theta)$$

$$w \leftarrow w + \alpha_w \delta_t \nabla_w Q_w(s_t, a_t) \qquad (Q_w)$$

Actor-Critic Architecture

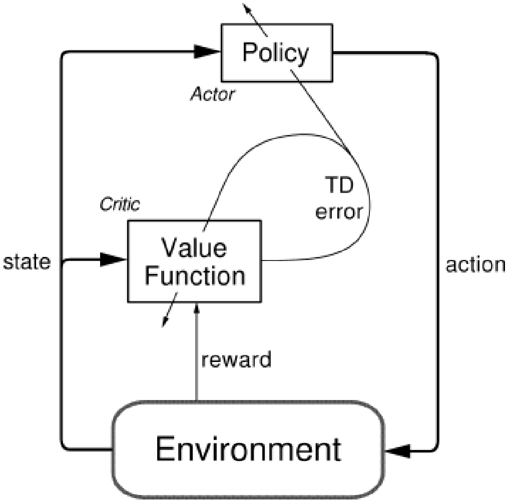

Figure 8: Actor-Critic structure [Sutton and Barto, 2018]

Option-Critic

The main idea of Option-Critic is to gradient relatively to :

- Each policy $\pi_\omega, \omega \in \mathcal{O}$
- Each termination probability $\beta_\omega, \omega \in \mathcal{O}$

applying formulas similar to Actor-Critic.

Option-Critic [Bacon et al., 2017]

Theorem (Intra-Option Policy Gradient Theorem, Policy)

Gradient of $\mathbb{E}\ Q_\Omega(s,\omega)$ *relatively to* θ *is given by :*

$$\nabla_\theta \mathbb{E}\ Q_\Omega(s,\omega) = \sum_{s,\omega} \mu_\Omega\left(s,\omega \mid s_0,\omega_0\right) \sum_a \frac{\partial \pi_{\omega,\theta}(a \mid s)}{\partial \theta} Q_U\left(s,\omega,a\right)$$

where

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

with

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

and with

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

Option-Critic [Bacon et al., 2017]

Theorem (Intra-Option Policy Gradient Theorem, Termination)

Gradient of $\mathbb{E}\ Q_\Omega(s,\omega)$ *relatively to* ϑ *is given by :*

$$\nabla_\vartheta \mathbb{E}\ Q_\Omega(s,\omega) = -\sum_{s',\omega} \mu_\Omega(s',\omega \mid s_i 1, \omega_0) \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial \vartheta} A_\Omega(s',\omega)$$

where

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

with

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

and with

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

Option-Critic Architecture [Bacon et al., 2017]

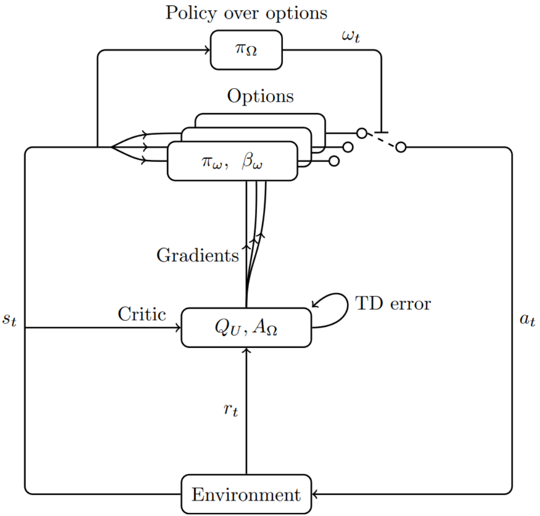

Figure 9: Option-Critic structure

Table of Contents

Option-Critic [Bacon et al., 2017]

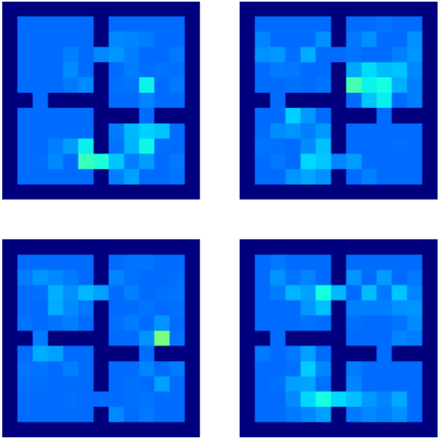

Figure 10: Option termination probability (ligher = greater) on 4 rooms example, 4 options [Bacon et al., 2017]

HIerarchical Reinforcement learning with Off-policy correction (HIRO) [Nachum et al., 2018]

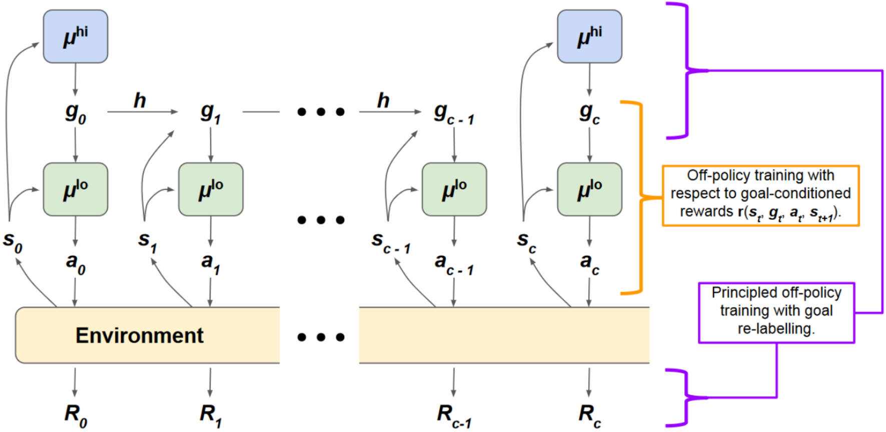

Figure 11: Design of HIRO algorithm [Nachum et al., 2018]

Test of HIRO

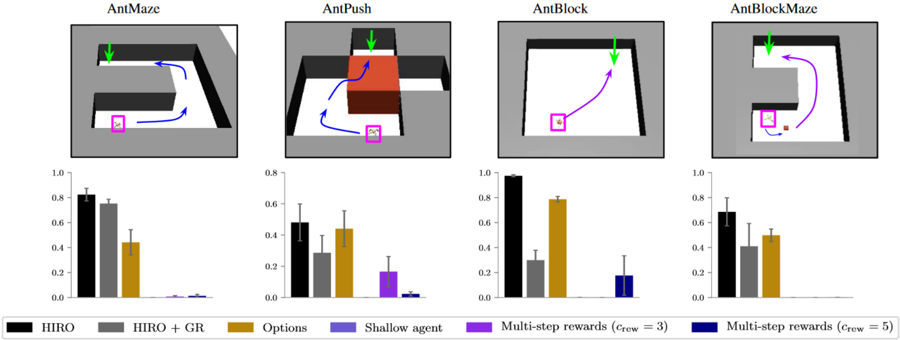

Figure 12: Application of HIRO algorithm to Ant tasks [Nachum et al., 2019]

Achievements of HRL [Hutsebaut-Buysse et al., 2022]

HRL state of the art provides :

- Formal context for options learning
- Performant exploration of complex environments
- Problem specific algorithms

Current research is focused on

- Applications of RL techniques to HRL (Robustness, exploration, Deep RL)
- Transposable skills
- Sample efficient learning
- Goal encoding

Personal Work

- Application and properties of State Abstraction from [Dean and Lin, 1995]
- Bibliography on HRL
- Exploration of Deep HRL methods

Table of Contents

Conclusion

HRL covers :

- Temporal and spatial abstraction
- Deep Option and subgoals Learning

Three main areas of research :

- Sample efficient option learning
- Goal encoding and transfer of skills
- Application of State Abstraction

Abel, D., Hershkowitz, D., and Littman, M. (2016).
Near optimal behavior via approximate state abstraction.
In *International Conference on Machine Learning*, pages 2915–2923. PMLR.

Bacon, P.-L., Harb, J., and Precup, D. (2017).
The option-critic architecture.
In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Dean, T. and Lin, S.-H. (1995).
Decomposition techniques for planning in stochastic domains.
In *IJCAI*, volume 2, page 3. Citeseer.

Hutsebaut-Buysse, M., Mets, K., and Latré, S. (2022).
Hierarchical reinforcement learning: A survey and open research challenges.
Machine Learning and Knowledge Extraction, 4(1):172–221.

Nachum, O., Gu, S. S., Lee, H., and Levine, S. (2018).
Data-efficient hierarchical reinforcement learning.

Advances in neural information processing systems, 31.

Nachum, O., Tang, H., Lu, X., Gu, S., Lee, H., and Levine, S. (2019).
Why does hierarchy (sometimes) work so well in reinforcement learning?
arXiv preprint arXiv:1909.10618.

Pateria, S., Subagdja, B., Tan, A.-h., and Quek, C. (2021).
Hierarchical reinforcement learning: A comprehensive survey.
ACM Computing Surveys (CSUR), 54(5):1–35.

Sutton, R. S. and Barto, A. G. (2018).
Reinforcement learning: An introduction.
MIT press.

Sutton, R. S., Precup, D., and Singh, S. (1999).
Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning.
Artificial intelligence, 112(1-2):181–211.

Xu, Z., van Hasselt, H. P., and Silver, D. (2018).

Meta-gradient reinforcement learning.
Advances in neural information processing systems, 31.