Globally Optimal Hierarchical Reinforcement Learning for Linearly-Solvable Markov Decision Processes

Guillermo Infante, Anders Jonsson, Vicenç Gómez

AAAI 2022

Overview

Introduction

Introduction

- Hierachical Reinforcement Learning aims to make learning more efficient by exploiting of large problems (Wen et al., 2020).
- Novel approach to HRL using Linearly-solvable Markov Decision Processes (LMDPs) by combining the value function for subtasks induced by partitions on the state space.
- LMDPs are a computationally efficient way to model sequential decision problems (Todorov, 2006).
- Our method presented here retrieves the globally optimal value function (Dietterich, 2000).

Background

Background - LMDPs (i)

An LMDP (Kappen et al., 2012; Todorov, 2006) is as a tuple $\mathcal{L} = \langle \mathcal{S}, \mathcal{T}, \mathcal{P}, \mathcal{R}, \mathcal{J} \rangle$, where:

- $\mathcal{S}$ is a discrete set of non-terminal states.
- $\mathcal{T}$ is a set of terminal states.
- We use $\mathcal{S}^+ = \mathcal{S} \cup \mathcal{T}$ to denote the full set of states.
- $\mathcal{P} : \mathcal{S} \to \Delta(\mathcal{S}^+)$ is an uncontrolled transition function.
- $\mathcal{R} : \mathcal{S} \to \mathbb{R}$ is a reward function for non-terminal states, assumed to be non-negative.
- $\mathcal{J} : \mathcal{T} \to \mathbb{R}$ is a reward function for terminal states, assumed to be non-negative.

The learning agent follows a policy $\pi : \mathcal{S} \to \Delta(\mathcal{S}^+)$, which is a conditional probability distribution over next states $\pi(\cdot|s_t)$.

Background - LMDPs (ii)

- At time-step t, the agent observes s_t and receives a reward

$$\mathcal{R}(s_t, \pi) = \mathcal{R}(s_t) - \lambda \cdot \mathrm{KL}(\pi(\cdot|s_t) \| \mathcal{P}(\cdot|s_t)).$$

- The aim of the agent is to maximize

$$v^\pi(s) = \mathbb{E}\left[\sum_{t=1}^{T-1} \mathcal{R}(S_t, \pi) + \mathcal{J}(S_T) \,\middle|\, S_1 = s\right].$$

- We obtain the following Bellman optimality equation

$$\begin{aligned}\frac{1}{\lambda} v(s) &= \frac{1}{\lambda} \max_\pi \left[\mathcal{R}(s, \pi) + \mathbb{E}_{s' \sim \pi(\cdot|s)} v(s')\right] \\ &= \frac{1}{\lambda} \mathcal{R}(s) + \max_\pi \mathbb{E}_{s' \sim \pi(\cdot|s)} \left[\frac{1}{\lambda} v(s') - \log \frac{\pi(s'|s)}{\mathcal{P}(s'|s)}\right] \quad (\forall s).\end{aligned}$$

Background - LMDPs (iii) - Eigenvector setting

- Taking the exponential transformation $z(s) = e^{v(s)/\lambda}$ for each $s \in \mathcal{S}^+$ leads to Bellman equations that are linear in z

$$z(s) = e^{\mathcal{R}(s)/\lambda} \sum_{s'} \mathcal{P}(s'|s) z(s').$$

- If $\mathcal{P}$ and $\mathcal{R}$ are known, this problem can be formulated as an eigenvector problem

$$\mathbf{z} = RP\mathbf{z}^+ \text{ where } R = \mathrm{diag}(e^{\mathcal{R}(\cdot)/\lambda})$$

where initially $\mathbf{z} = \mathbf{1}$.

Background - LMDPs (iii) - Online setting

- Alternatively, there exists an online (corrected) update rule

$$\hat{z}(s_t) \leftarrow (1-\alpha_t)\hat{z}(s_t) + \alpha_t e^{r_t/\lambda}\hat{z}(s_{t+1})\frac{\mathcal{P}(s_{t+1}|s_t)}{\hat{\pi}(s_{t+1}|s_t)}.$$

- Given a z, policies are derived using

$$\pi(s'|s) = \frac{\mathcal{P}(s'|s)z(s')}{\sum_{s''}\mathcal{P}(s''|s)z(s'')}.$$

- This process is equivalent to a probabilistic inference task (Kappen et al., 2012).

Background - LMDPs (iv) - Compositionality

- Let $\{\mathcal{L}_1, \ldots, \mathcal{L}_n\}$ be a collection of LMDPs.
- Each LMDP $\mathcal{L}_i$ only differs in the reward structure of each terminal state $\mathsf{t} \in \mathcal{T}$ (i.e. $\mathcal{J}_i(\mathsf{t})$).
- Let us consider a new LMDP $\mathcal{L}$ whose (exponential) reward structure at terminal states can be expressed as follows (Todorov, 2009)

$$e^{\mathcal{J}(\mathsf{t})/\lambda} = z(\mathsf{t}) = \sum_{k=1}^{n} w_k e^{\mathcal{J}_k(\mathsf{t})/\lambda}.$$

- Since the Bellman equation is linear in z, then the optimal value function of any $s \in \mathcal{S}$ satisfies the same equation above

$$z(s) = \sum_{k=1}^{n} w_k z_k(s)$$

Hierarchical LMDPs

Hierarchical LMDPs (i)

- Hierarchical decomposition inspired by the work of Wen et al. (Wen et al., 2020).
- Given an LMDP $\mathcal{L}$, its state space $\mathcal{S}$ is partitioned into L subsets $\{\mathcal{S}_i\}_{i=1}^{L}$.
- Each such subset $\mathcal{S}_i$ induces a subtask, represented by an LMDP $\mathcal{L}_i = \langle \mathcal{S}_i, \mathcal{T}_i, \mathcal{P}_i, \mathcal{R}_i, \mathcal{J}_i \rangle$:
  1. The set of non-terminal states is $\mathcal{S}_i$.
  2. The set of terminal states $\mathcal{T}_i$ includes all states in $\mathcal{S}^+ \setminus \mathcal{S}_i$ that are reachable in one step from a state in $\mathcal{S}_i$.
  3. $\mathcal{P}_i : \mathcal{S}_i \to \mathcal{S}_i^+$ and $\mathcal{R}_i : \mathcal{S}_i \to \mathbb{R}$ are restrictions of $\mathcal{P}$ and $\mathcal{R}$ to $\mathcal{S}_i$.
  4. Reward at $\tau \in \mathcal{T}_i$ equals $\mathcal{J}_i(\tau) = \mathcal{J}(\tau)$ if $\tau \in \mathcal{T}$, and $\mathcal{J}_i(\tau) = \hat{v}(\tau)$ otherwise, where $\hat{v}(\tau)$ is the estimated value in $\mathcal{L}$ of the non-terminal state $\tau \in \mathcal{S} \setminus \mathcal{S}_i$.

Hierarchical LMDPs (ii)

Definition
Two subtasks $\mathcal{L}_i$ and $\mathcal{L}_j$ are equivalent if there exists a bijection $f : \mathcal{S}_i \to \mathcal{S}_j$ such that the transition probabilities and rewards of non-terminal states are equivalent through f.

- We define a set of *exit states* $\mathcal{E} = \cup_{i=1}^{L} \mathcal{T}_i$
- We also use $\mathcal{E}_i = \mathcal{E} \cap \mathcal{S}_i$ to denote the set of (non-terminal) exit states in the subtask $\mathcal{L}_i$.
- A set of equivalence classes $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_C\}$, $C \leq L$, i.e. a partition of the set of subtasks $\{\mathcal{L}_1, \ldots, \mathcal{L}_L\}$ such that all subtasks in a given partition are equivalent.
- We represent a single subtask $\mathcal{L}_j = \langle \mathcal{S}_j, \mathcal{T}_j, \mathcal{P}_j, \mathcal{R}_j, \mathcal{J}_j \rangle$ per equivalence class $\mathcal{C}_j \in \mathcal{C}$.

Hierarchical LMDPs (iii) - Illustration

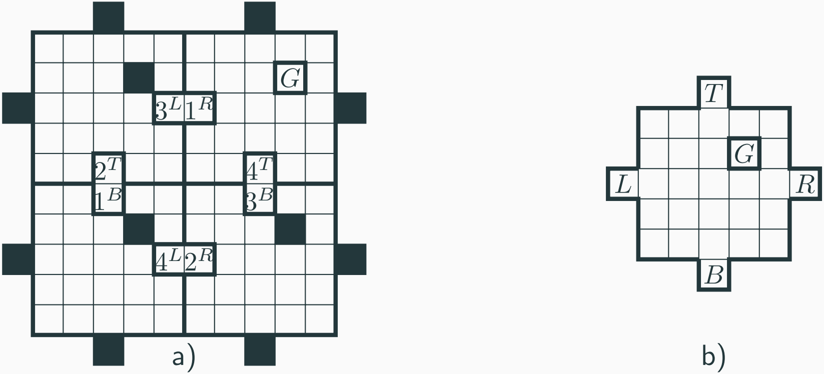


Figure 1: a) A 4-room LMDP, with all exit states highlighted; b) a single subtask with 5 terminal states G, L, R, T, B that is equivalent to all 4 room subtasks.

Hierarchical LMDPs (iv) - Subtask compositionality (i)

- For a subtask $\mathcal{L}_j = \langle \mathcal{S}_j, \mathcal{T}_j, \mathcal{P}_j, \mathcal{R}_j, \mathcal{J}_j \rangle$ as defined previously, consider its terminal set $\mathcal{T}_j = \{\tau_1, \ldots, \tau_n\}$.
- We define n base LMDPs $\mathcal{L}_j^1, \ldots, \mathcal{L}_j^n$, where each base LMDP $\mathcal{L}_j^k$ only differ in the reward of terminal states $\mathcal{J}_j^k$.
- Concretely, $z_j^k(\tau) = 1$ if $\tau = \tau_k$, and $z_j^k(\tau) = 0$ otherwise. This corresponds to an actual reward of $\mathcal{J}_j^k(\tau) = 0$ for $\tau = \tau_k$, and $\mathcal{J}_j^k(\tau) = -\infty$ otherwise.
- Thus, we can solve these base LMDPs to obtain $z_j^1, \ldots, z_j^n$.
- Having an estimate $\hat{z}(s)$ for each $t \in \mathcal{T}_j$, then thanks to compositionality, the estimated value of each non-terminal state $s \in \mathcal{S}_i$ is

$$\hat{z}(s) = \hat{z}(\tau_1) z_j^1(s) + \cdots + \hat{z}(\tau_n) z_j^n(s) \;\; \forall s \in \mathcal{S}_i, \forall \mathcal{L}_i \in \mathcal{C}_j.$$

Hierarchical LMDPs (v) - Subtask compositionality (ii)

- For all subtaks, terminal states $\tau_1 \dots \tau_n$ are by definition in $\mathcal{E}$.
- Having access to a value estimate $\hat{z}_{\mathcal{E}} : \mathcal{E} \to \mathbb{R}$ and the base LMDPs value functions $z_j^1, \dots, z_j^n$ is enough to express the value estimate of every other state without learning.
- No need to store an explicit estimate $\hat{z}(s)$.
- Once we have solved the base LMDPs, there is no need to solve again each individual subtask. That is why we represent a single subtask $\mathcal{L}_j$ for each equivalence class $\mathcal{C}_j$.

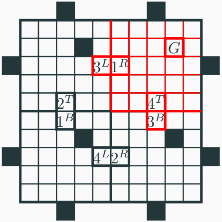
Hierarchical LMDPs (vi) - Subtask compositionality (iii)

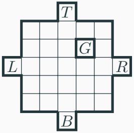
Remark

If $\hat{z}(s)$ is optimal for $s \in \mathcal{E}$, then $\hat{z}(s)$ for $s \in \mathcal{S}_i$ will also be optimal. Thanks to compositionality we have

$$\hat{z}(s) = \hat{z}(3^L) * z_L(s) + \hat{z}(3^B) * z_B(s) + \hat{z}(G) * z_G(s)$$

For any state s in $\mathcal{S}_i$ represented in red. Thus, if $\hat{z} = z^*$ for terminal states, then it will be optimal for the interior states as well.





Algorithms

Eigenvector algorithm (i)

- We can reformulate the system of equations yielded by

$$\hat{z}(s) = \hat{z}(\tau_1) z_j^1(s) + \cdots + \hat{z}(\tau_n) z_j^n(s) \;\; \forall s \in \mathcal{S}_i, \forall \mathcal{L}_i \in \mathcal{C}_j.$$

  to be defined only in the states $s \in \mathcal{E}$.
- Thus, we define

$$\mathbf{z}_{\mathcal{E}} = G \mathbf{z}_{\mathcal{E}}.$$

- G contains the value of the base LMDPs, while $\mathbf{z}_{\mathcal{E}}$ is initialized with value 1 for all $s \in \mathcal{E}$.
- No need to keep an estimate of the interior states in the higher level, the values for states in $\mathcal{E}$ are sufficient.

Eigenvector algorithm (ii) - Convergence proof

Lemma (1)
If the reward of each terminal state $t \in \mathcal{T}_i$ equals its optimal value in $\mathcal{L}$, i.e. $z_i(t) = z(t)$, the optimal value of each non-terminal state $s \in \mathcal{S}_i$ equals its optimal value in $\mathcal{L}$, i.e. $z_i(s) = z(s)$.

Lemma (2)
The solution to $\mathbf{z}_{\mathcal{E}} = G\mathbf{z}_{\mathcal{E}}$ is unique.

Lemma (3)
For each subtask $\mathcal{L}_i$ and state $s \in \mathcal{S}_i^+$, it holds that

$$z_i^1(s) + \cdots + z_i^n(s) \leq 1$$

Online algorithm (i)

- We keep an estimate of the base value functions $\hat{z}_j^1, \ldots, \hat{z}_j^n$ for each $\mathcal{L}_j$.
- A single transition is enough to update the value functions of all base LMDPs associated with $\mathcal{L}_j$ by using intra-task learning (Kaelbling, 1993).
- The update rule for the states in the exit set becomes

$$\hat{z}_{\mathcal{E}}(s) \leftarrow (1-\alpha_\ell)\hat{z}_{\mathcal{E}}(s) + \alpha_\ell[\hat{z}_{\mathcal{E}}(t_1)\hat{z}_j^1(s) + \cdots + \hat{z}_{\mathcal{E}}(t_n)\hat{z}_j^n(s)].$$

- Estimates at any level are learned in an episodic fashion.
- When to update states in $\mathcal{E}$ is still a question to answer (next slide).

Online algorithm (ii)

We propose the following alternatives:

V_1: Update the value of an exit state $s \in \mathcal{E}_i$ each time we take a transition from s.

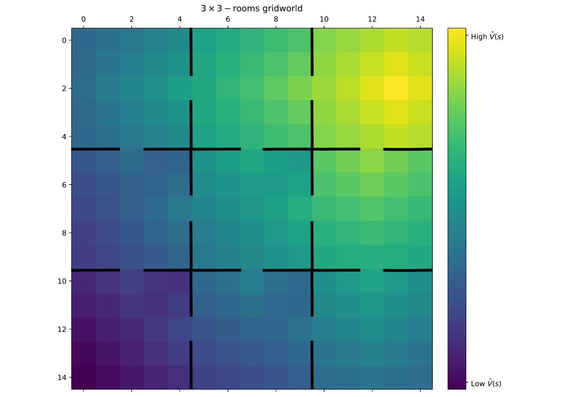
V_2: When we reach a terminal state of the subtask $\mathcal{L}_i$, update the values of all exit states in $\mathcal{E}_i$.

V_3: When we reach a terminal state of the subtask $\mathcal{L}_i$, update the values of all exit states in $\mathcal{E}_i$ and all exit states of subtasks in the equivalence class $\mathcal{C}_j$ of $\mathcal{L}_i$.

Experiments and results

Experiments - Rooms domain

- We varied the size of the rooms as well as the number of rooms.



Results - Rooms domain

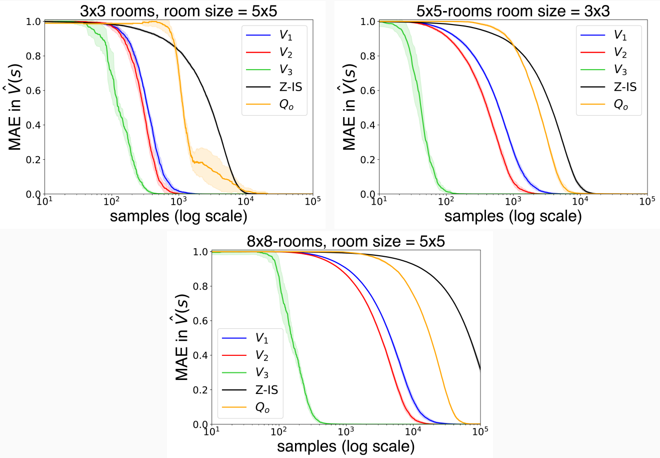
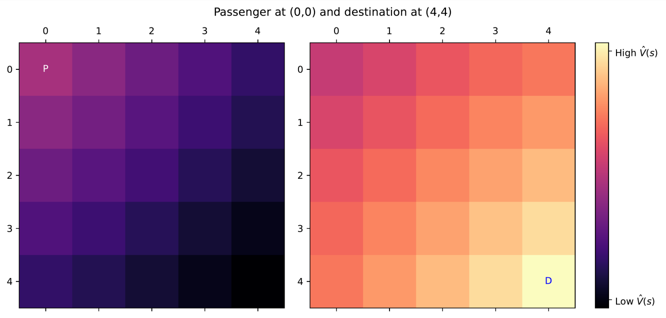


Figure 2: MAE over time for 3×3 (left), 5×5 (middle) and 8×8 (right) room instances.

Experiments - Taxi domain

- A passenger is located at one of the four corners and he must be carried to a certain corner (excluding the pickup location).
- Base LMDPs here are going to each of the corners.
- Sometimes there exist natural equivalence classes in facotres MDPs



Results - Taxi domain

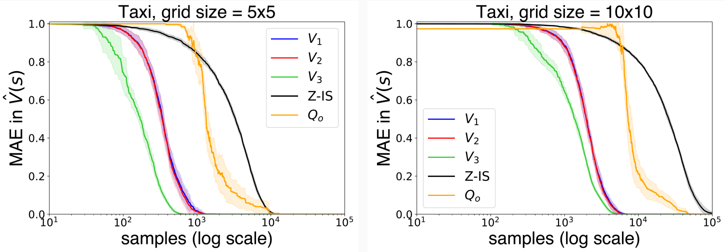


Figure 3: MAE over time for 5×5 (left) and 10×10 (right) grids of Taxi domain.

Contributions and Conclusion

Contributions

- We define a novel scheme based on compositionality for solving subtasks.
- The subtasks decomposition is at the level of the value function, thus our approach does not suffer from non-stationarity in the online setting.
- Under mild assumptions, our method converges to the optimal value function.
- Unlike the options setting, we can retrieve the optimal value function for the full state space using the value estimates for the exit states.

Conclusion

- Unlike typical HRL approaches, we no longer need a high-level policy that selects among subgoals (e.g. non-stationarity in SMDPs).
- Instead, we are able to retrieve the optimal value function for each state thanks to the decomposition of the value function in terms of the values of base LMDPs.

References

Dietterich, T. G. (2000). Hierarchical reinforcement learning with the MAXQ value function decomposition. *J. Artif. Intell. Res.*, 13:227–303.

Kaelbling, L. P. (1993). Learning to Achieve Goals. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1094–1099.

Kappen, H. J., Gómez, V., and Opper, M. (2012). Optimal control as a graphical model inference problem. *Mach. Learn.*, 87(2):159–182.

Todorov, E. (2006). Linearly-solvable Markov decision problems. *Advances in Neural Information Processing Systems (NIPS)*, pages 1369–1376.

Todorov, E. (2009). Compositionality of optimal control laws. *Advances in Neural Information Processing Systems (NIPS)*, pages 1856–1864.

Wen, Z., Precup, D., Ibrahimi, M., Barreto, A., Van Roy, B., and Singh, S. (2020). On Efficiency in Hierarchical Reinforcement Learning. In *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS)*.