ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230290

Theoretical Remarks on Feudal Hierarchies and Reinforcement Learning

Diogo S. Carvalho[a,b;*], **Francisco S. Melo**[a,b] **and Pedro A. Santos**[a,b]

[a]INESC-ID, Lisbon, Portugal
[b]Instituto Superior Técnico, University of Lisbon, Lisbon, Portugal
ORCiD ID: Diogo S. Carvalho https://orcid.org/0000-0003-3008-7322

Abstract. Hierarchical reinforcement learning is an increasingly demanded resource for learning to make sequential decisions towards long term goals. Feudal hierarchies are among the most deployed frameworks. However, there are few theoretical results for hierarchical structures. In this work, we formalize the common two-level feudal hierarchy as two Markov decision processes, with the one on the high level being dependent on the policy executed at the low level. Despite the non-stationarity raised by the dependency, we show that each of the processes presents stable behavior. We then build on the first result to show that, regardless of the convergent learning algorithm used for the low level, convergence of both prediction and control algorithms at the high-level is guaranteed. Our results contribute with theoretical support for the use of feudal hierarchies in combination with standard reinforcement learning methods at each level.

1 Introduction

The field of hierarchical reinforcement learning is receiving growing attention [29]. While the most well-known frameworks that give rise and exploit hierarchies on sequential decision making problems remain virtually unchanged [3, 36, 4, 28], their use is both expanding and diversifying [40, 15, 2]. As an example that attests for its power, hierarchies and reinforcement learning, combined with expert demonstrations, set the state of the art on the game of Minecraft [7, 20, 33, 24]. Hierarchical reinforcement learning is also piercing through the multi-agent field [23, 14, 1]. Hierarchies seduce planning and reinforcement learning theorists and practitioners by claiming to allow for temporal abstraction and easing long-term credit assignment and exploration. As a side product, a suitably constructed hierarchical policy allows for flexible extension and replacement of the decision blocks that naturally emerge at each level of the hierarchy.

Feudal hierarchies were the first hierarchical structure proposed for sequential decision making with reinforcement learning [3]. As the name may suggest, an agent on a higher level sets goals for an agent on a lower level to achieve. While learning a hierarchical policy at both levels simultaneously, an added difficulty of non-stationarity arises [27, 16, 12]. As a practical example of this non-stationarity, suppose that, in the beginning of training, a high-level agent issues a suitable goal for a low-level agent; if the low-level agent does not achieve its goal, the high-level agent's decision is not assigned its due credit. To learn each level of a hierarchical policy using standard reinforcement learning methods is to underlyingly assume that, on such level, a Markov decision process (MDP) is present. The assumption is both intuitive and convenient. However, it is not clear which hierarchies sustain a stable behavior of each MDP. In this work, we bridge theory and application of hierarchical reinforcement learning by (i) sculpting the Markovian properties of feudal hierarchies, formalizing the hierarchy as coupled uneven Markov decision processes; (ii) presenting sufficient conditions for their stability under an assumption on the stopping time of low-level episodes; (iii) proving the convergent behavior of learning algorithms for prediction and control of hierarchical policies, resorting to the previous stability result. Specifically, we prove that the TD(0) and Q-learning algorithms at the high-level converge with probability 1 (w.p. 1) if the low-level policy is converging. The assumption is weak and only requires infinitely often state-action visitation.

* Corresponding Author. Email: diogo.s.carvalho@tecnico.ulisboa.pt.

2 Background

A Markov decision process [30] formalizes the interaction of an agent and its environment through a tuple $(\mathcal{X}, \mathcal{A}, P, r, \gamma)$, with $\mathcal{X}$ a finite state space; $\mathcal{A}$ a finite action space; P a transition probability tensor with $[P]_{a,x,x'}$ the probability that state x' follows from the execution of action a in state x, $P(x' \mid x, a)$; $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ the expectation of the reward, which has bounded variance; γ in $[0, 1)$ a discount factor. A policy $\pi : \mathcal{X} \times \mathcal{A} \to [0, 1]$ maps states and actions to probabilities.

Solving a Markov decision problem means learning a policy π that, for every state and action, has maximal value. By value we mean $V_\pi : \mathcal{X} \to \mathbb{R}$ such that

$$V_\pi(x) \coloneqq \mathbb{E}_\pi\big[r(x_0, a_0) + \gamma \sum_{t=0}^{\infty} \gamma^t r(x_{t+1}, a_{t+1}) \mid x_0 = x\big],$$

where we use the policy as a subscript to the expectation to denote that $a_t \sim \pi(x_t, \cdot)$ and $x_{t+1} \sim P(\cdot \mid x_t, a_t)$. Algorithms such as TD$(0)$ [38] solve the *prediction* problem of computing V_π. We can also define the action value function $Q_\pi : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ as

$$Q_\pi(x, a) \coloneqq \mathbb{E}_\pi\big[r(x, a) + \gamma V_\pi(x')\big],$$

with $x' \sim P(\cdot \mid x, a)$. A greedy policy with respect to an action value function Q uniformly selects, at each state x, actions from the set $\arg\max_a Q(x, a)$. It is an established fact [30] that an optimal policy π^* exists and has maximal action value $Q^* \coloneqq Q_{\pi^*}$ verifying

$$Q^*(x, a) = \mathbb{E}\big[r(x, a) + \gamma \max_{a'} Q^*(x', a')\big],$$

where the subscript π was dropped from the expectation since the expression holds no randomness regarding actions. Conversely, the greedy policy with respect to Q^* is optimal. The problem of computing Q^* or π^* is called the *control* problem.

The mentioned fact that, from Q^*, we can trivially obtain an optimal policy π^* legitimizes the existence of value-based algorithms, that do not explicitly improve the policy, and rather construct a sequence of value functions $\{Q_t\}_{t\in\mathbb{N}_0}$ converging to Q^*.[1] Q-learning [42] is the bedrock of most value-based methods [26, 39, 41, 8]. Contrarily, policy-based methods hold on the policy gradient theorem [35] to construct a sequence of policies $\{\pi_t\}_{t\in\mathbb{N}_0}$ such that $\pi_t \to \pi^*$. The actor-critic architecture [13] is the general form of most policy-based algorithms [19, 25, 31, 6, 31]. In the actor-critic architecture, the actor consists of a policy and the critic consists of a value function. In the previous sense, solving both prediction and control problems is necessary in the class of policy-based methods.

3 Feudal hierarchies

In a feudal hierarchy, a task (represented as an MDP) is addressed by several agents operating at different levels: higher-level agents act by selecting goals for lower-level agents; the lower-level agents, in turn, act to achieve the goals set by the higher-level agent. As our first contribution, we formalize each level as a Markov decision process.

We refer to the original (non-hierarchical) MDP as a *shallow* Markov decision process and we use superscripts h and l to denote the *high-level* and *low-level* components of a two-level hierarchy, respectively. Consider a shallow Markov decision process $\mathcal{M} \coloneqq (\mathcal{X}, \mathcal{A}, P, r, \gamma)$, as described in Section 2. We thus define a two-level hierarchy as a pair $(\mathcal{M}^h, \mathcal{M}^l)$, with

$$\begin{cases} \mathcal{M}^h(\pi^l) \coloneqq \big(\mathcal{X}, \Omega, P^h(\pi^l), r^h(\pi^l), \gamma^h\big); \\ \mathcal{M}^l \coloneqq (\mathcal{X}\times\Omega, \mathcal{A}, P^l, r^l, 1) \end{cases},$$

where, adding to the notation already established for Markov decision processes, we used Ω to denote the action space of the high-level MDP and, simultaneously, one of the components of the state space of the low-level MDP. We also call Ω the *goal* space. We focus on finite goal spaces. However, the formalism extends to continuous goal spaces, including embeddings of Ω as previously considered in applications: the former was used in [27]; the latter in [40].

On the high-level process $\mathcal{M}^h$, we note the dependency of the high-level transition probabilities P^h and reward r^h on the policy π^l of the agent at the low level. On the low-level process $\mathcal{M}^l$, the low-level reward function $r^l : (\mathcal{X}\times\Omega)\times\mathcal{A} \to \mathbb{R}$ can technically be any. We also observe that, even though we set the discount factor on $\mathcal{M}^l$ as 1, and thus do not discount the low-level rewards, we will very soon formalize a stopping time for each low-level episode, parameterized by the discount factor γ, through Assumption 1 . The assumption is equivalent, regarding the optimization objective and corresponding solution, to considering long term discounted rewards while ensuring that every low-level episode eventually terminates.

We now provide further detail on each non-trivial component of both $\mathcal{M}^h$ and $\mathcal{M}^l$, starting with the latter. We refer to the decision-maker in the MDP $\mathcal{M}^h$ as the high-level *sub-agent* and the decision maker in the MDP $\mathcal{M}^l$ as the low-level sub-agent.

[1] Unless otherwise noted or clear from context, we consider throughout the document pointwise convergence of functions, such as value functions and policies, and the topology induced by the Euclidean metric over $\mathbb{R}$. If the topological space is strictly contained in $\mathbb{R}$, we consider the usual sup-space induced topology. We also use $f \to f^*$ to be read as *f converges to f^**.

Low-level Markov decision process We start by describing the dynamics of the low-level MDP. After the high-level sub-agent sets a goal $\omega \in \Omega$, the low-level sub-agent observes the current state of the environment, x_t, and the current goal, ω. A low-level episode takes place at the low-level MDP, $\mathcal{M}^l$, for τ time steps. We assume that τ is a stopping time random variable following a geometric distribution parameterized by γ.

Assumption 1. *The stopping time of a low-level episode, τ, is a random variable supported in $\mathbb{N}_0$ with probability mass function $\xi(t) \coloneqq P(\tau = t) = (1-\gamma)\gamma^t$.*

The value of γ determines the "urgency" of the stopping time—larger values of γ yield shorter episodes. Regarding the optimization objective of the MDP, Assumption 1 is equivalent to considering a discount factor of γ without stopping the episode. However, Assumption 1 ensures that every low-level episode eventually terminates and a new decision may take place at the high-level once more. Additionally, even though many practical implementations assume that the stopping time τ takes a specific value T w.p. 1, our assumption is not far, in the sense that our results would hold up to fixed temporal shifts of T and $\gamma \lesssim 1$, while still allowing for infinite exploration of low-level state-action pairs.

On the low level, for each goal, the dynamics of the original Markov decision process $\mathcal{M}$ are preserved, so we define

$$[P^l]_{a,(x,\omega),(x',\omega')} \coloneqq \begin{cases} [P]_{a,x,x'}, & \text{if } \omega = \omega' \\ 0, & \text{if } \omega \neq \omega' \end{cases}.$$

Intuitively, it is possible to think of the low-level MDP as "aggregating" a set of MDPs, one for each goal $\omega \in \Omega$, that share the same transition dynamics.

Contrarily to the transition structure P^l, the reward structure r^l is not necessarily inherited from the original process $\mathcal{M}$ and, in practical applications, the architect of a hierarchy can design any r^l structure that is meaningful in the sense of high-level goal achievement. It should, however (i) lead the low-level agent towards achieving the goal $\omega \in \Omega$ set at the high-level; (ii) restrict the low-level agent to achieving its goal ω, i.e., agnostically to the long-term plan of the high-level agent. Such structure over the low-level reward is, therefore, amenable to previously implemented hierarchies [40, 27, 3]. We respectively denote $V^l_{\pi^l}$ and $Q^l_{\pi^l}$ the value and action value functions of a policy π^l at the low level.

High-level Markov decision process After the high-level sub-agent issues a goal $\omega \in \Omega$, the low-level sub-agent executes its policy π^l until the stopping time. Consider

$$[P^\omega_{\pi^l}]_{x,x'} \coloneqq \sum_a \pi^l\big((x,\omega),a\big)P(x' \mid x,a) \tag{1}$$

the transition dynamics of the Markov reward process induced by the policy π^l on M^l. For each goal ω, $[P^h(\pi^l)]_{\omega,x,x'}$ is the probability that, at the stopping time, the low-level sub-agent is at state (x',ω). Formally,

$$\big[P^h(\pi^l)\big]_{\omega,\cdot,\cdot} \coloneqq \mathbb{E}[P^{\omega\,\tau}_{\pi^l}],$$

where we are taking the expectation with respect to the stopping time τ, appearing as the power of the Markov matrix $P^\omega_{\pi^l}$.

Regarding the high-level rewards, for each goal ω and low-level policy π^l, they equal the expected discounted rewards, until the stopping time, of the original Markov decision process which is obtained

by following policy π^l. The high-level agent is rewarded by the environment. Formalizing once more,

$$r^h(\pi^l)(x,\omega) \coloneqq \mathbb{E}_{\pi^l}\Big[\sum_{t=0}^{\tau} r(x_t,a_t) \mid x_0 = x\Big].$$

The expectation is taken with respect to the stopping time τ and states and actions according to π^l.

We finally argue for the choice a discount factor of $\mathcal{M}^h$ as $\gamma^{\frac{1}{\gamma}}$, so that, on average, the sum of rewards is discounted by the same factor γ as in the original process $\mathcal{M}$, even though such assumption is not necessary to establish the results below. On a different but related context, the same suggestion is made in [22].

Before moving on to the theoretical results on the stability of feudal hierarchies and asymptotic behaviour of reinforcement learning over them, we present an example of a Markov decision process and, especially, its decomposition as a feudal hierarchical process. The example could be skipped without loss of comprehension but serves the purpose of providing intuition over the just proposed formal definition of the feudal hierarchical process.

3.1 Example of a feudal hierarchical process

Suppose our shallow reinforcement learning agent, corresponding to the original Markov decision process $\mathcal{M}$, is an ant set on a 7×7 maze grid world depicted in Figure 1.[2] Even though cells are numbered, we omit the numbers from the figure to avoid visual overhead. The goal state of the environment is the southwestern-most cell on the grid. The ant receives a positive unit reward at the goal state and null otherwise, describing the reward structure r. Adding some difficulty to the ant's navigation task, the grid world is set on a windy place. At each time step, the wind blows *north*, *south*, *east* or *west* with uniform equal probability. The pairs composed of the ant's position and the wind direction form the state space $\mathcal{X}$. The ant is not strong enough to move in a contrary direction to the wind and, therefore, the ant can either stand at its current position or let itself move in the direction of the wind. Both actions are successful unless the wind blows the ant toward a wall cell, in which case the ant stays in the same cell, forming the transition probabilities P. The action space $\mathcal{A}$ is then composed of the actions *stand* and *move*.

Now we move on to describing a feudal hierarchical structure over the Markov decision process presented. We start with the high-level $\mathcal{M}^h$. At the high-level, the agent has the same state space $\mathcal{X}$ as in the shallow environment, observing both the position of the ant and wind direction. It can then set a goal for the low level to achieve. Such goals are to move *north*, *south*, *east* or *west*, forming the goal action space Ω. The high-level observes a new state and outputs a new goal after the low-level episode terminates. The number of low-level time steps between high-level observations is random with a geometric distribution of $\gamma \in [0,1)$.The transition probabilities depend on the policy operating at the low level as well as the termination of the low-level episode. The reward received by the high-level is the sum, over the length of the low-level episode, of rewards it would have received in the shallow environment, r. If the high-level discount factor is $\gamma^{\frac{1}{\gamma}}$, on average, the rewards of the high-level are discounted by the same factor as in the shallow Markov decision process, γ.

Finally, we focus on the low-level process $\mathcal{M}^l$. The state of the agent at the low level is the triple of ant's position, wind direction and goal set by the high-level agent. Its action space is equal to $\mathcal{A}$. When the low level takes an action, the transitions happen as in the shallow environment, modelled by P^l, regardless of the goal. The low-level agent receives a positive unit reward if it moves in the direction of the goal, a negative unit reward if it moves in a direction away from the goal, and a null reward if its position remains unchanged, forming r^l.

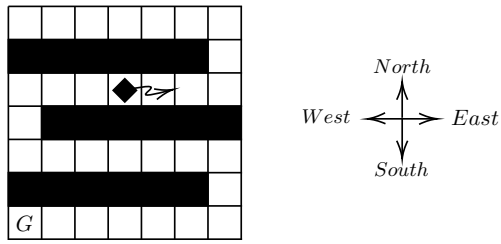

Figure 1: Depiction of the windy ant maze grid world. The goal state is marked as G. The position of the ant is marked with the diamond. The wind direction is indicated as a bent arrow. At the current time step, wind is blowing *east*. The optimal action in the shallow environment is therefore to *stand*. In the hierarchical environment, the optimal goal to be set by the agent at the high-level is *west*. With the current wind direction, for such goal, the optimal action at the low level is to *stand*, which give a null reward.

4 Stability of feudal hierarchies

In the remainder of the document, we respectively denote $V^l_{\pi^l}$ and $Q^l_{\pi^l}$ the value and action value functions of a policy π^l at the low level and respectively denote by $V^h_{\pi^h}(\pi^l)$ and $Q^h_{\pi^h}(\pi^l)$ the value and action value functions of a policy π^h at the high-level MDP $\mathcal{M}(\pi^l)$. We present our first result, on the stability of a two-level feudal hierarchy as formalized above.

Theorem 1 (Stability of feudal hierarchies)**.** *Consider the Markov decision processes $\mathcal{M}$, $\mathcal{M}^h$ and $\mathcal{M}^l$, and a converging sequence of low-level policies $\pi^l_t \to \pi$, $t \in \mathbb{N}$. Under Assumption 1 , we have:*

1. $P^h(\pi^l_t) \to P^h(\pi)$;
2. $r^h(\pi^l_t) \to r^h(\pi)$;
3. $V^h_{\pi^h}(\pi^l_t) \to V^h_{\pi^h}(\pi)$;
4. $Q^h_{\pi^h}(\pi^l_t) \to Q^h_{\pi^h}(\pi)$,

for each high-level policy π^h.

Proof. Before proving assertions 1 through 4, we recall that, in metric spaces, if an element $f_t \to f^*$, $G(f_t) \to G(f^*)$ if and only if G is a continuous mapping. Therefore, the theorem equivalently states that, for every high-level policy π^h, the functions P^h, r^h, $V^h_{\pi^h}$ and $Q^h_{\pi^h}$ are continuous on the low-level policies π^l. With the previous remark in mind, we start proving each assertion of the theorem.

1. We show that the transition probabilities $P^h(\pi^l)$ vary continuously with π^l for each ω. First note that the transition dynamics Markov reward process induced by the low-level policy on $\mathcal{M}^l$, $P^\omega_{\pi^l}$, computed through Equation (1), is a continuous function of π^l. Now recall that

$$\big[P^h(\pi^l)\big]_{\omega,\cdot,\cdot} = \mathbb{E}[P^{\omega\,\tau}_{\pi^l}].$$

[2] The example is inspired by the Ant Maze environment from [5], which was also adapted in [27], and by the Windy Maze environment in [18].

Finally note that

$$\begin{aligned}\mathbb{E}[P_{\pi^l}^{\omega\,\tau}] &= \sum_{\tau=0}^{\infty}\xi(\tau)P_{\pi^l}^{\omega\,\tau}\\ &= \sum_{\tau=0}^{\infty}(1-\gamma)\gamma^{\tau}P_{\pi^l}^{\omega\,\tau}\\ &= (1-\gamma)(I-\gamma P_{\pi^l}^{\omega})^{-1}.\end{aligned}$$

The inverse is well defined and continuous.

2. Recall that, for all $\omega \in \Omega$,

$$r^h(\pi^l)(x,\omega) = \mathbb{E}_{\pi^l|\omega}\big[\sum_{t=0}^{\tau} r(x_t,a_t) \mid x_0 = x\big].$$

After defining the real vector

$$[r_{\pi^l}^{\omega}]_x \coloneqq \sum_a \pi^l\big((x,\omega),a\big)r(x,a),$$

we write the expected sum of discounted rewards

$$\begin{aligned}&\mathbb{E}_{\pi^l|\omega}\big[\sum_{t=0}^{\tau} r(x_t,a_t) \mid x_0 = x\big]\\ &\quad= \Big[\sum_{\tau=0}^{\infty}\xi(\tau)\sum_{t=0}^{\tau}\gamma^t P_{\pi^l}^{\omega\,t} r_{\pi^l}^{\omega}\Big]_x\\ &\quad= \Big[\sum_{\tau=0}^{\infty}\xi(\tau)(I-\gamma P_{\pi^l}^{\omega})^{-1}\cdot\\ &\qquad\cdot\big(I-(\gamma P_{\pi^l}^{\omega})^{\tau+1}\big)r_{\pi^l}^{\omega}\Big]_x\\ &\quad= \Big[(I-\gamma P_{\pi^l}^{\omega})^{-1}\cdot\\ &\qquad\cdot\big(I-(1-\gamma)(I-\gamma^2 P_{\pi^l}^{\omega})^{-1}\cdot\\ &\qquad\cdot\gamma P_{\pi^l}^{\omega}\big)r_{\pi^l}^{\omega}\Big]_x.\end{aligned}$$

The conclusion follows again from continuity.

3. The value function $V_{\pi^h}^h(\pi^l) : \mathcal{X} \to \mathbb{R}$ is defined as

$$V_{\pi^h}^h(\pi^l)(x) = \mathbb{E}_{\pi^h}\big[\sum_{t=0}^{\infty}(\gamma^h)^t r^h(\pi^l)(x_t,\omega_t) \mid x_0 = x\big],$$

where $\omega_t \sim \pi^h(x_t,\cdot)$ and $x_{t+1} \sim P^h(\pi^l)(\cdot \mid x_t,\omega_t)$.
We observe that

$$V_{\pi^h}^h(\pi^l) = \big(I-\gamma^h P^h(\pi^l)\big)^{-1} r^h(\pi^l)$$

and that the expression varies continuously with π^l, by 1 and 2.

4. Building on assertion 3, we observe that

$$Q_{\pi^h}^h(\pi^l)(x,\omega) = \mathbb{E}\big[r^h(\pi^l)(x,\omega) + \gamma V_{\pi^h}^h(\pi^l)(x')\big]$$

continuously maps π^l at every state-action pair.

□

For a final remark, consider that we say a sequence of Markov decision processes is converging if both the transition and reward functions are converging. In such sense, Theorem 1 guarantees that, if the low-level policy is converging, then the high-level Markov decision process is also converging.

5 Convergence of feudal hierarchical reinforcement learning

In this section, we focus on evaluating and learning feudal hierarchical policies. In our setting, the low level learns using any learning algorithm at any learning rate, as long as it converges to the optimal policy of the low-level Markov decision process and the high-level uses TD(0) for prediction and Q-learning for control. We show that, even though the algorithms operate over a non-stationary Markov decision process, convergence still holds w.p. 1.

Before we move to the theorem, we introduce some notation. $r_{t+1}(x,a)$ denotes a reward sample distributed according to $r^h(\pi_t^l)(x,a)$; $y_t(x,\omega)$ a next state sample distributed according to $P^h(\pi_t^l)(x,\omega)$; $V_{\pi^h}^{*,h}$ denotes the value function of the high-level policy π^h when the low-level policy is $\pi^{*,l}$ and $Q_t^{*,h}$ denotes the optimal action value function of the high-level Markov decision process $\mathcal{M}^h(\pi_t^l)$, where π_t^l is the low-level policy at time step t.

Theorem 2 (Convergence of feudal hierarchical reinforcement learning)**.** *Consider the Markov decision processes $\mathcal{M}$, $\mathcal{M}^h$ and $\mathcal{M}^l$. Consider a sequence of low-level policies $\pi_t^l, t \in \mathbb{N}$, converging to the optimal low-level policy $\pi^{*,l}$.*

1. *Suppose that the high-level learning rates $\alpha_t : \mathcal{X} \to [0,1]$ are such that, for all x,*

$$\begin{cases}\sum_t \alpha_t(x) = \infty & w.p.1;\\ \sum_t \alpha_t^2(x) < \infty & w.p.1.\end{cases}$$

Then, the TD(0) algorithm on the high level, given by

$$\begin{aligned}V_{t+1}^h(x) = V_t^h(x) &+\\ &+ \alpha_t(x)\Big(r_{t+1}(x,\omega) + \gamma^h V_t^h\big(y_t(x,\omega)\big) - V_t^h(x)\Big),\end{aligned}$$

converges to $V_{\pi_h}^{*,h}$ w.p. 1 for any high-level policy π^h.

2. *We assume the high-level learning rates $\alpha_t : \mathcal{X}\times\Omega \to [0,1]$ are such that, for all (x,ω),*

$$\begin{cases}\sum_t \alpha_t(x,\omega) = \infty & w.p.1;\\ \sum_t \alpha_t^2(x,\omega) < \infty & w.p.1.\end{cases}$$

Then, the Q-learning algorithm on the high level, given by

$$\begin{aligned}Q_{t+1}^h(x,\omega) = Q_t^h(x,\omega) &+\\ &+ \alpha_t(x,\omega)\Big(r_{t+1}(x,\omega) + \gamma^h \max_{\omega'} Q_t^h\big(y_t(x,\omega),\omega'\big) -\\ &- Q_t^h(x,\omega)\Big),\end{aligned}$$

converges to $Q^{*,h}$ w.p. 1..

Proof. We start with recalling a convergence result for stochastic processes [32]. Consider the random iteration

$$\Delta_t(z) = \big(1-\alpha_t(z)\big)\Delta_t(z) + \alpha_t(z)F_t(z),$$

where $z \in \mathcal{Z}$ and $\mathcal{Z}$ is finite. Consider also the sequence of σ-fields $\mathcal{F}_t$ such that $\big(\alpha_t(z), \Delta_t(z), F_t(z)\big) \in \mathcal{F}_t$. The sequence $\{\Delta_t\}$ converges to 0 w.p. 1 under the following conditions, for each z in $\mathcal{Z}$:

1. $\|\mathbb{E}[F_t(z) \mid \mathcal{F}_t]\| \leq \gamma\|\Delta_t\| + c_t, c_t \to 0$ w.p. 1 ;
2. $var(F_t(z) \mid \mathcal{F}_t) \leq K(1+\|\Delta_t\|)^2$.

Note that the assumptions on the learning rates require, in particular, infinitely often state action visitation. We refer to the original manuscript [32] for clarifying details over the learning rates.

Once we prove convergence of Algorithm 2, for control, convergence of Algorithm 1 for prediction follows as a particular case: the high-level policy induces a Markov reward process over $\mathcal{M}^h$, for each low-level policy π_t^l, which is a particular case of a Markov decision process, with a singleton action space; in a Markov reward process, TD(0) updates are equivalent to the ones of Q-learning. Therefore, to establish Theorem 2, we must only verify conditions 1 and 2 hold for Algorithm 2.

Consider the stochastic processes

$$\begin{cases} \Delta_t(x,\omega) = Q_t^h(x,\omega) - Q^{*,h}(x,\omega) \\ F_t(x,\omega) = r_{t+1}(x,\omega) + \\ \quad +\gamma^h \max_{a'} Q_t^h\big(y_t(x,\omega),\omega'\big) - Q^{*,h}(x,\omega). \end{cases}$$

1. After focusing on the definition of $F_t(x,\omega)$, we observe that

$$\begin{aligned} &\mathbb{E}[F_t(x,\omega) \mid \mathcal{F}_t] \\ &\quad = \mathbb{E}\Big[r_{t+1}(x,\omega) + \gamma^h \max_{\omega'} Q_t^h\big(y_t(x,\omega),\omega'\big) - \\ &\qquad - {Q_t}^{*,h}(x,\omega) \mid \mathcal{F}_t\Big] + \\ &\qquad + \mathbb{E}\big[Q_t^{*,h}(x,\omega) - Q^{*,h}(x,\omega) \mid \mathcal{F}_t\big]. \end{aligned}$$

   From Theorem 1 we know the second term vanishes. For the first term we use the theorem again and also the contractiveness of the Bellman operator to conclude the assertion:

$$\begin{aligned} &\Big\|\mathbb{E}\Big[r_{t+1}(x,\omega) + \gamma^h \max_{\omega'} Q_t^h\big(y_t(x,\omega),\omega'\big) - Q_t^{*,h}(x,\omega) \mid \mathcal{F}_t\Big]\Big\| \\ &\qquad \leq \gamma^h \big\|Q_t^h - Q_t^{*,h}\big\| \\ &\qquad \leq \gamma^h \big\|\Delta_t\big\| + \gamma^h \big\|Q^{*,h} - Q_t^{*,h}\big\|. \end{aligned}$$

2. We must show that, for each t, the variance of $F_t(x,\omega)$ is bounded by a linear combination of 1, $\|Q_t^h(x,\omega)\|$ and $\|{Q_t^h}^2(x,\omega)\|$. We call for attention on the definition of $F_t(x,\omega)$ once more. Now we recall that the reward function has bounded variance (see Section 2). Additionally, the contribution of the term $Q^{*,h}$ to the variance is additive, since it is independent from Q_t^h given the history $\mathcal{F}_t$. Finally, the dependence of $F_t(x,\omega)$ on $Q_t^h(x,\omega)$ is, at most, linear. We conclude the result. The present proof, on a bound of the variance of the updates, closely resembles one from [32].

□

While Theorem 2 assumes the limit of the low-level policies is the optimal low-level policy, the convergence result would still hold if the limiting policy was any, π, not necessarily optimal. The limits of the value function would be, in that case, with respect to π. The result requires the low-level policies to be converging to the optimal and infinitely often state action visitation at the high-level. Numerous learning algorithms are known to converge in the tabular case. Examples of those algorithms are Q-learning [11], SARSA [37] and actor-critic variations [13], requiring as sufficient the GLIE hypothesis [32] of being *greedy in the limit with infinite exploration*. Suppose that the low-level policy being learned is itself a GLIE policy. Then, if at the high-level actions are chosen from the Q-values estimates according to Boltzmann [35] or decreasing ϵ-greedy [34] distributions, greed and exploration at the low level are not harmed.

6 Discussion

Extension to n-level hierarchies In this work, we established that (i) the commonly employed two-level feudal hierarchy for reinforcement learning is stable, in the sense that, when the low-level policy is converging, the high-level Markov decision process is also converging and that (ii) standard tabular reinforcement learning algorithms converge on non-stationary but convergent Markov decision processes. Putting the two results together, we conclude that reinforcement learning converges in two-level feudal hierarchies. Nevertheless, our conclusion does extend to any n-level feudal hierarchy. To see (i), we can fix a low-level policy π^l and define an extra higher-level hierarchy on top of $\mathcal{M}^h(\pi^l)$ the same way we did for two-level hierarchies; we can then use the base case proven in the text and conclude the induction step. To see (ii), no further proof is required. Even though two-level hierarchies are significantly more employed, three-level hierarchies also appeared in the literature [16].

Comparison with previous results Our work is not the first that looks at the asymptotic behaviour of reinforcement learning on non-stationary Markov decision processes [17, 9, 21, 10]. We establish a convergence result for prediction and control on the high level of a feudal hierarchy if the low level is convergent. Equivalently, we prove convergence of TD(0) and Q-learning on convergent sequences of Markov decision processes. The control result is not, in general, a particular case of the one from [21], as optimal Q-values are not independent of the history of the process throughout the interaction. In fact, the optimal Q-values are time dependent, even though they converge to a time-independent solution. Additionally, we do not impose ergodicity of the underlying processes. For similar reasons, the prediction result is not a particular case of the one in [10].

7 Conclusion

Our work formalizes a two-level feudal hierarchy as two layers of Markov decision processes that preserve the structure of the underlying environment. We then show that, despite non-stationarity of the high-level with respect to the policy operating at the low level, the dependency is smooth in the continuity sense. Finally, we prove the convergent behavior of a feudal hierarchical learning algorithm.

Future work should build on the convergence results for the case where other learning algorithms, such as actor-critic ones, encompassing both prediction and control steps, are used at the high-level. It should also be worth considering function approximation at the high-level, even though most shallow reinforcement learning algorithms are known not to necessarily converge in such setting. Finally, while our setting is purposely very general, adding assumptions on the learning rates could allow considerations on the ones to use at high and low levels in order to achieve improved convergence rates.

Acknowledgements

This work was partially supported by the Portuguese Fundação para a Ciência e a Tecnologia (FCT) under INESC-ID multi-annual funding (UIDB/50021/2020), RELEvaNT (PTDC/CCICOM/5060/2021), SLICE (PTDC/CCI-COM/30787/2017) and CRAI (C628696807-00454142 (IAPMEI/PRR)). The research was also partially supported by TAILOR, a project funded by EU Horizon 2020 research and innovation program under GA No. 952215. Diogo S. Carvalho acknowledges the FCT PhD grant (2020.05360.BD).

References

[1] Sanjeevan Ahilan and Peter Dayan, 'Feudal multi-agent hierarchies for cooperative reinforcement learning', *arXiv preprint arXiv:1901.08492*, (2019).

[2] Pierre-Luc Bacon, Jean Harb, and Doina Precup, 'The option-critic architecture', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, (2017).

[3] Peter Dayan and Geoffrey E Hinton, 'Feudal reinforcement learning', in *Advances in Neural Information Processing Systems*, eds., S. Hanson, J. Cowan, and C. Giles, volume 5. Morgan-Kaufmann, (1993).

[4] Thomas G Dietterich et al., 'The maxq method for hierarchical reinforcement learning.', in *ICML*, volume 98, pp. 118–126. Citeseer, (1998).

[5] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel, 'Benchmarking deep reinforcement learning for continuous control', in *International conference on machine learning*, pp. 1329–1338. PMLR, (2016).

[6] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al., 'Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures', in *International Conference on Machine Learning*, pp. 1407–1416. PMLR, (2018).

[7] William H Guss, Brandon Houghton, Nicholay Topin, Phillip Wang, Cayden Codel, Manuela Veloso, and Ruslan Salakhutdinov, 'Minerl: A large-scale dataset of minecraft demonstrations', *arXiv preprint arXiv:1907.13440*, (2019).

[8] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver, 'Rainbow: Combining improvements in deep reinforcement learning', in *Thirty-second AAAI conference on artificial intelligence*, (2018).

[9] Marcus Hutter, 'Extreme state aggregation beyond markov decision processes', *Theoretical Computer Science*, **650**, 73–91, (2016).

[10] Marcus Hutter, Samuel Yang-Zhao, and Sultan J Majeed, 'Conditions on features for temporal difference-like methods to converge', *arXiv preprint arXiv:1905.11702*, (2019).

[11] Tommi Jaakkola, Michael I Jordan, and Satinder P Singh, 'On the convergence of stochastic iterative dynamic programming algorithms', *Neural computation*, **6**(6), 1185–1201, (1994).

[12] YiDing Jiang, Shixiang (Shane) Gu, Kevin P Murphy, and Chelsea Finn, 'Language as an abstraction for hierarchical deep reinforcement learning', in *Advances in Neural Information Processing Systems*, eds., H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, volume 32. Curran Associates, Inc., (2019).

[13] Vijay R Konda and John N Tsitsiklis, 'Actor-critic algorithms', in *Advances in neural information processing systems*, pp. 1008–1014, (2000).

[14] Xiangyu Kong, Bo Xin, Fangchen Liu, and Yizhou Wang, 'Revisiting the master-slave architecture in multi-agent deep reinforcement learning', *arXiv preprint arXiv:1712.07305*, (2017).

[15] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum, 'Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation', *Advances in neural information processing systems*, **29**, 3675–3683, (2016).

[16] Andrew Levy, George Konidaris, Robert Platt, and Kate Saenko, 'Learning multi-level hierarchies with hindsight', *Proceedings of the 7th International Conference on Learning Representations*, (2017).

[17] Lihong Li, Thomas J Walsh, and Michael L Littman, 'Towards a unified theory of state abstraction for mdps.', *ISAIM*, **4**, 5, (2006).

[18] Eric Liang, Richard Liaw, Robert Nishihara, Philipp Moritz, Roy Fox, Ken Goldberg, Joseph E. Gonzalez, Michael I. Jordan, and Ion Stoica, 'RLlib: Abstractions for distributed reinforcement learning', in *International Conference on Machine Learning (ICML)*, (2018).

[19] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra, 'Continuous control with deep reinforcement learning', *arXiv preprint arXiv:1509.02971*, (2015).

[20] Zichuan Lin, Junyou Li, Jianing Shi, Deheng Ye, Qiang Fu, and Wei Yang, 'Juewu-mc: Playing minecraft with sample-efficient hierarchical reinforcement learning', *arXiv preprint arXiv:2112.04907*, (2021).

[21] Sultan Javed Majeed and Marcus Hutter, 'On q-learning convergence for non-markov decision processes.', in *IJCAI*, pp. 2546–2552, (2018).

[22] Sultan Javed Majeed and Marcus Hutter, 'Exact reduction of huge action spaces in general reinforcement learning', *arXiv preprint arXiv:2012.10200*, (2020).

[23] Rajbala Makar, Sridhar Mahadevan, and Mohammad Ghavamzadeh, 'Hierarchical multi-agent reinforcement learning', in *Proceedings of the fifth international conference on Autonomous agents*, pp. 246–253, (2001).

[24] Hangyu Mao, Chao Wang, Xiaotian Hao, Yihuan Mao, Yiming Lu, Chengjie Wu, Jianye Hao, Dong Li, and Pingzhong Tang, 'Seihai: A sample-efficient hierarchical ai for the minerl competition', in *International Conference on Distributed Artificial Intelligence*, pp. 38–51. Springer, (2021).

[25] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu, 'Asynchronous methods for deep reinforcement learning', in *International conference on machine learning*, pp. 1928–1937. PMLR, (2016).

[26] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al., 'Human-level control through deep reinforcement learning', *nature*, **518**(7540), 529–533, (2015).

[27] Ofir Nachum, Shixiang (Shane) Gu, Honglak Lee, and Sergey Levine, 'Data-efficient hierarchical reinforcement learning', in *Advances in Neural Information Processing Systems*, eds., S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, volume 31. Curran Associates, Inc., (2018).

[28] Ronald Parr and Stuart Russell, 'Reinforcement learning with hierarchies of machines', *Advances in neural information processing systems*, 1043–1049, (1998).

[29] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek, 'Hierarchical reinforcement learning: A comprehensive survey', *ACM Computing Surveys (CSUR)*, **54**(5), 1–35, (2021).

[30] Martin L Puterman, *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2014.

[31] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov, 'Proximal policy optimization algorithms', *arXiv preprint arXiv:1707.06347*, (2017).

[32] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári, 'Convergence results for single-step on-policy reinforcement-learning algorithms', *Machine learning*, **38**(3), 287–308, (2000).

[33] Alexey Skrynnik, Aleksey Staroverov, Ermek Aitygulov, Kirill Aksenov, Vasilii Davydov, and Aleksandr I Panov, 'Forgetful experience replay in hierarchical reinforcement learning from demonstrations', *arXiv preprint arXiv:2006.09939*, (2020).

[34] Richard S Sutton, 'Generalization in reinforcement learning: Successful examples using sparse coarse coding', *Advances in neural information processing systems*, 1038–1044, (1996).

[35] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour, 'Policy gradient methods for reinforcement learning with function approximation', in *Advances in neural information processing systems*, pp. 1057–1063, (2000).

[36] Richard S Sutton, Doina Precup, and Satinder Singh, 'Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning', *Artificial intelligence*, **112**(1-2), 181–211, (1999).

[37] Csaba Szepesvári, *Static and dynamic aspects of optimal sequential decision making*, Ph.D. dissertation, Bolyai Institute, 1998.

[38] Gerald Tesauro et al., 'Temporal difference learning and td-gammon', *Communications of the ACM*, **38**(3), 58–68, (1995).

[39] Hado Van Hasselt, Arthur Guez, and David Silver, 'Deep reinforcement learning with double q-learning', in *Proceedings of the AAAI conference on artificial intelligence*, volume 30, (2016).

[40] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu, 'Feudal networks for hierarchical reinforcement learning', in *International Conference on Machine Learning*, pp. 3540–3549. PMLR, (2017).

[41] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas, 'Dueling network architectures for deep reinforcement learning', in *International conference on machine learning*, pp. 1995–2003. PMLR, (2016).

[42] Christopher JCH Watkins and Peter Dayan, 'Q-learning', *Machine learning*, **8**(3-4), 279–292, (1992).