# Near instance-optimal PAC Reinforcement Learning in deterministic MDPs

Andrea Tirinzoni[3], Aymen Al Marjani[1,2], Emilie Kaufmann[2],

[1]UMPA, ENS de Lyon
[2]Inria Lille, Equipe SCOOL
[3]META AI, Paris

Journées de Statistiques
16 Juin 2022

# Motivation

# Markov Decision Process

1. $H$: Horizon.
2. $(\mathcal{S}_h, \mathcal{A}_h)_{h\in[H]}$: Finite state and action spaces.
3. The agent interacts *sequentially* with the environment within *episodes*. At each episode, the agent starts at some fixed state $s_1 \in \mathcal{S}_1$.
4. Then for every $h \in [H]$:
   - The agent is at some state $s_h \in \mathcal{S}_h$,
   - chooses to play action $a_h \in \mathcal{A}_h$,
   - receives reward $R(s_h, a_h) \sim q_h(.|s_h, a_h)$,
   - makes transition to the next state $s_{h+1} = f_h(s_h, a_h)$. Given $(s_h, a_h)$, $s_{h+1}$ is deterministic.

# The PAC RL problem

We define a policy as $\pi = \{\pi_h\}_{h \in [H]}$, where $\pi_h : \mathcal{S} \to \mathcal{A}$.

# The PAC RL problem

We define a policy as $\pi = \{\pi_h\}_{h\in[H]}$, where $\pi_h : \mathcal{S} \to \mathcal{A}$.
Each policy $\pi$ is characterized by its value function:

$$V_1^{\pi} = \mathbb{E}_{q,\pi}\left[ \sum_{h=0}^{H} R(s_h^{\pi}, a_h^{\pi}) \middle| \, s_1 \right].$$

The optimal value function is defined as: $V_1^{\star} = \max_{\pi} V_1^{\pi}$

# The PAC RL problem

We define a policy as $\pi = \{\pi_h\}_{h\in[H]}$, where $\pi_h : \mathcal{S} \to \mathcal{A}$.
Each policy $\pi$ is characterized by its value function:

$$V_1^{\pi} = \mathbb{E}_{q,\pi}\left[\sum_{h=0}^{H} R(s_h^{\pi}, a_h^{\pi}) \,\middle|\, s_1\right].$$

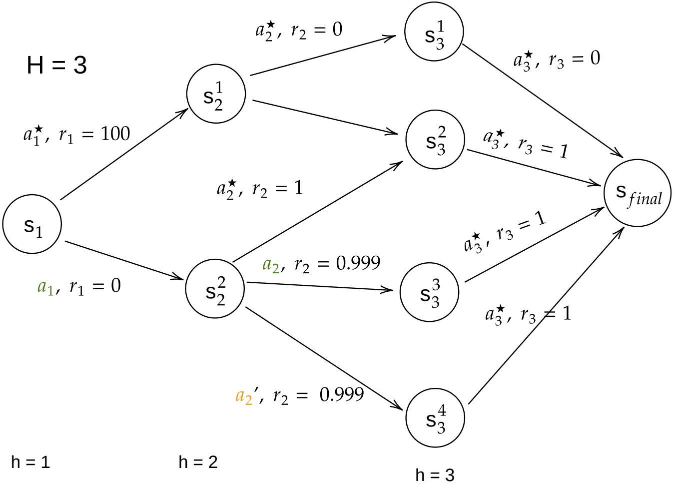
The optimal value function is defined as: $V_1^{\star} = \max_{\pi} V_1^{\pi}$

**The PAC RL Problem:** The transitions $f_h$ are known, but the rewards $(q_h)_{h\in[H]}$ are unknown. Given parameters $(\varepsilon, \delta)$, interact with the environment for some number of episodes $\tau_\delta$, until you find a policy $\widehat{\pi}$ such that:

$$\mathbb{P}(V_1^{\widehat{\pi}} \geq V_1^{\star} - \varepsilon) \geq 1 - \delta.$$

using minimum number of Episodes!

# The learning problem illustrated



# Previous results

- Define the action-value function:

$$Q_h^\star(s,a) = \mathbb{E}_{q,\pi^\star}\left[\sum_{\ell=h}^{H} R(s_\ell, a_\ell) \middle| s_h = s, a_h = s\right],$$
$$V_h^\star(s) = \max_{a \in_h} Q_h^\star(s,a).$$

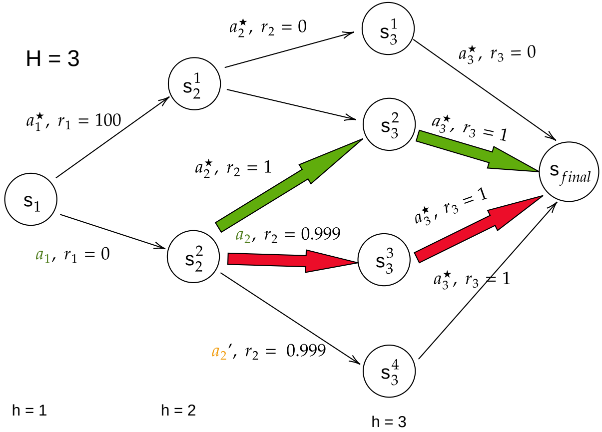
- Define the value gaps:

$$\Delta_h(s,a) = V_h^\star(s) - Q_h^\star(s,a)$$

- (Wagenmaker et al. 2021) propose an algorithm for PAC RL whose sample complexity is roughly

$$\widetilde{\mathcal{O}}\left(\sum_{(s,a,h)} \frac{H^2 \log(1/\delta)}{\max(\Delta_h(s,a), \varepsilon)^2}\right)$$

# Value gaps? Seriously ?

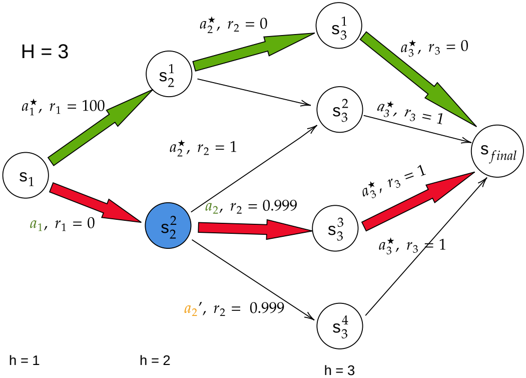



$\Delta_2(s_2^2, a_2) = 10^{-3} \Longrightarrow$ Relying on value gaps to detect suboptimality of $(s_2^2, a_2)$ will take many episodes !

# Beyond value gaps: Return gaps !

We propose a *return* gaps:

$$\overline{\Delta}_h(s,a) = V_1^\star - \max_{\pi\text{: goes through (h,s,a)}} V_1^\pi.$$




$\overline{\Delta}_2(s_2^2, a_2) = 98 \Longrightarrow$ We can detect suboptimality of $(s_2^2, a_2)$ earlier !

# Hoeffding bounds

- Assuming the reward distributions are $\sigma^2$-subgaussian, we can define high probability upper and lower confidence bounds on the value of any policy:

$$\overline{V}_h^{t,\pi}(s) = \sum_{\ell=h}^{H} \left( \hat{r}_\ell^t(s_\ell^\pi, a_\ell^\pi) + \sqrt{\frac{\log\left(\frac{e(t+1)SAH}{\delta}\right)}{2n_\ell^t(s_\ell^\pi, a_\ell^\pi)}} \right),$$

$$\underline{V}_h^{t,\pi}(s) = \sum_{\ell=h}^{H} \left( \hat{r}_\ell^t(s_\ell^\pi, a_\ell^\pi) - \sqrt{\frac{\log\left(\frac{e(t+1)SAH}{\delta}\right)}{2n_\ell^t(s_\ell^\pi, a_\ell^\pi)}} \right)$$

# Algorithm: Elimination rule and stopping rule

**Algorithm 1** Elimination-based PAC RL (EPRL) for deterministic MDPs

**Input:** deterministic MDP (without reward) $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \{f_h\}_{h\in[H]}, s_1, H)$, $\varepsilon$, $\delta$
Initialize $\mathcal{A}_h^0(s) \leftarrow \mathcal{A}_h(s)$ for all $h \in [H], s \in \mathcal{S}_h$
Set $n_h^0(s,a) \leftarrow 0$ for all $h \in [H], s \in \mathcal{S}_h, a \in \mathcal{A}_h(s)$
**for** $t = 1, \ldots$ **do**
    Play $\pi^t \leftarrow \textsc{MaxCoverage}()$
    Update statistics $n_h^t(s,a), \hat{r}_h^t(s,a)$
    $\mathcal{A}_h^t(s) \leftarrow \mathcal{A}_h^{t-1}(s) \cap \left\{a \in \mathcal{A} : \max_{\pi\in\Pi_{s,a,h}\cap\Pi^{t-1}} \overline{V}_1^{t,\pi}(s_1) \geq \max_{\pi\in\Pi} \underline{V}_1^{t,\pi}(s_1)\right\}$
    **if** $\max_{\pi\in\Pi^t}\left(\overline{V}_1^{\pi,t}(s_1) - \underline{V}_1^{\pi,t}(s_1)\right) \leq \varepsilon$ or $\forall h \in [H], s \in \mathcal{S}_h : |\mathcal{A}_h^t(s)| \leq 1$ **then**
        Stop and recommend $\widehat{\pi} \in \arg\max_{\pi\in\Pi^t} \overline{V}_1^{\pi,t}(s_1)$
    **end if**
**end for**

# Algorithm: Sampling rule

**Algorithm 2** Elimination-based PAC RL (EPRL) for deterministic MDPs

**Input:** deterministic MDP (without reward) $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \{f_h\}_{h\in[H]}, s_1, H)$, $\varepsilon$, $\delta$
Initialize $\mathcal{A}_h^0(s) \leftarrow \mathcal{A}_h(s)$ for all $h \in [H], s \in \mathcal{S}_h$
Set $n_h^0(s,a) \leftarrow 0$ for all $h \in [H], s \in \mathcal{S}_h, a \in \mathcal{A}_h(s)$
**for** $t = 1, \dots$ **do**
    Play $\pi^t \leftarrow \textsc{MaxCoverage}()$
    Update statistics $n_h^t(s,a), \hat{r}_h^t(s,a)$
    Do eliminations.
    Check if stopping rule is triggered.
**end for**

**function** $\textsc{MaxCoverage}()$
    Let $k_t \leftarrow \min_{h,s,a} n_h^{t-1}(s,a) + 1$ and $\bar{t}_{k_t} \leftarrow \inf_{l\in\mathbb{N}}\{l : k_l = k_t\}$
        **return** $\pi^t \leftarrow \arg\max_{\pi\in\Pi} \sum_{h=1}^{H} \mathbb{1}\left(a_h^\pi \in \mathcal{A}_h^{\bar{t}_{k_t}-1}(s_h^\pi), n_h^{t-1}(s_h^\pi, a_h^\pi) < k_t\right)$

# Main Results

## Theorems 1 and 3 (Tirinzoni, AL Marjani, Kaufmann" 22)

EPRL is $(\varepsilon, \delta)$-PAC. Moreover, with probability at least $1 - \delta$, its sample complexity bounded by:

$$\tau_\delta = \widetilde{\mathcal{O}}\left(\varphi^\star\left(\left[\frac{H^2 \log(1/\delta)}{\max(\overline{\Delta}_h(s,a), \varepsilon)^2}\right]_{s,a,h}\right)\right),$$
$$\leq \widetilde{\mathcal{O}}\left(\sum_{s,a,h} \frac{H^2 \log(1/\delta)}{\max(\overline{\Delta}_h(s,a), \varepsilon)^2}\right).$$

Furthermore, any $(\varepsilon, \delta)$-PAC algorithm must have a sample complexity at least:

$$\mathbb{E}[\tau_\delta] = \Omega\left(\varphi^\star\left(\left(\frac{\log(1/\delta)}{\max(\overline{\Delta}_h(s,a), \varepsilon)^2}\right)_{s,a,h}\right)\right).$$

where the $\widetilde{\mathcal{O}}$ hides universal constants (not that large, trust me) and log factors.

# Conclusion and perspectives

1. We can detect and eliminate suboptimal (state,action) pair very early by looking at the full trajectory and not only what happens after.
2. Combining this simple elimination rule with clever exploration, we can achieve near optimal sample complexity !
3. Extensions to stochastic transitions ?

# Thanks !