Online Reinforcement Learning: PAC Exploration in Discounted MDPs

Mohammad Sadegh Talebi
m.shahi@di.ku.dk
Department of Computer Science

MDP Classification Based on Horizon

Three classes of MDPs based on the horizon N:

- Discounted MDPs
- Finite-horizon MDPs
- Average-reward MDPs

From MDPs to RL

- In RL, we consider the same interaction model as in MDPs, but assume that P and R are unknown.
- The agent wishes to maximize her collected (discounted) rewards
- An optimal policy, or a near-optimal one, must be learnt but the available information is the history of experience

$$h_t = (s_1, a_1, r_1, \ldots, s_{t-1}, a_{t-1}, r_{t-1}, s_t)$$

RL Problems Based on N

A classification of RL problems based on (task) horizon N:

- Discounted MDPs $\Longrightarrow$ Discounted RL problems
- Finite-horizon MDPs $\Longrightarrow$ Episodic RL problems (with a fixed episode length)
  (not covered in OReL 2024)
- Average-reward MDPs $\Longrightarrow$ Average-reward RL problems

Discounted RL

- The agent interacts with a discounted MDP

$$M = (\mathcal{S}, \mathcal{A}, \underbrace{P, R}_{\text{unknown}}, \gamma)$$

- The interaction proceeds for an arbitrary number of time steps without any reset.
- The initial state is chosen by Nature.
- Objective: to learn a policy solving

$$\max_{\pi \in \Pi^{\mathsf{SD}}} V^\pi(s) = \mathbb{E}^\pi\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r_t \Big| s_1 = s\Big]$$

for any initial state $s \in \mathcal{S}$.

RL Settings

Common taxonomies of RL settings:

- **Off-policy** vs. **On-policy**
  - In off-policy, data is collected using some behavior policy (logging policy). Hence, the learned policy does not influence data collection.
  - Whereas in on-policy, actions are taken according to the learned policy.
- **Offline (Batch)** vs. **Online**
  - Offline RL works with pre-collected data using some behavior policy, whereas in online RL data is collected along the way.
  - Both aim to find a near-optimal policy using as few samples as possible.
  - They look at different performance metrics.
  - Offline RL closely resembles supervised ML.

Offline-vs-online taxonomy appears more relevant in practice as well as the recent literature.

RL: Design Approaches

Three main approaches to algorithm design in RL:

- Model-Based: Consists in maintaining an approximate MDP model through estimating R and P, and deriving a value function from the *approximate* MDP.
  - Examples: UCB1, UCRL2.
- Model-Free: Directly learns a value function (without estimating R and P), and derives a policy from it.
  - Examples: TD, variants of Q-Learning, DQN.
- Policy Search: Directly searches in the space of policies.
  - Example: Policy Gradient, PPO.

More recent terminology: **Model-based** vs. **Valued-based** vs. **Policy-based**

Online Discounted RL: Setting and Performance Metrics

Recap

Online Discounted RL. An **agent** interacts with a discounted MDP $M = (\mathcal{S}, \mathcal{A}, P, R, \gamma)$ for some (potentially unbounded) rounds without any reset

At each time step $t = 1, 2, \ldots$:

- The agent observes the current state s_t and takes an action $a_t \in \mathcal{A}$
- M decides a reward $r_t \sim R(s_t, a_t)$ and a next state $s_{t+1} \sim P(\cdot|s_t, a_t)$
- The agent receives r_t (any time in step t before start of $t+1$)

M is unknown (beyond $\mathcal{S}$ and $\mathcal{A}$), and the goal is to maximize $\sum_{t=1}^{\infty} \gamma^{t-1} r_t$ (in expectation) using collected experience (history):

$$h_t = (s_1, a_1, r_1, \ldots, s_{t-1}, a_{t-1}, r_{t-1}, s_t)$$

Need to balance exploration and exploitation.

Online RL: Performance Metrics

- Many offline algorithms can be made online with some tricks (e.g., `QL`).
- But will they explore well?

> For online RL, we need performance metrics to measure the quality of exploration-exploitation tradeoff.

Online RL: Performance Measures

The performance of a learning algorithm $\mathbb{A}$ can be measured through:

- **Convergence:** Whether $\mathbb{A}$ converges to an optimal (or near-optimal) policy.
- **PAC Sample Complexity:** The number of steps where the value of the current policy output by $\mathbb{A}$ is not near-optimal with high-probability.
- **Regret:** The amount of reward lost due to choosing sub-optimal actions by $\mathbb{A}$.

In fact these metrics measure how exploration-exploitation tradeoff is implemented.

More precise definitions to follow.

Sample Complexity of Exploration

Consider an RL algorithm $\mathbb{A}$, and $h_t = (s_1, a_1, r_1, \ldots, s_{t-1}, a_{t-1}, r_{t-1}, s_t)$ a history of $\mathbb{A}$, with $a_t \sim \pi_t(\cdot|s_t)$. I.e., π_t is the learned policy at time t.

A notion of sample complexity introduced by (Kakade, 2003):

Sample Complexity of Exploration

For input $\varepsilon > 0$, time step t is bad if π_t is not ε-optimal for the current state s_t:

$$V^{\pi_t}(s_t) < V^\star(s_t) - \varepsilon \quad \Longrightarrow \quad t \text{ is } \varepsilon\text{-bad}$$

The **sample complexity of exploration** of $\mathbb{A}$ is the total number of ε-bad time steps over the entire trajectory:

$$\sum_{t=1}^{\infty} \mathbb{I}\{V^{\pi_t}(s_t) < V^\star(s_t) - \varepsilon\}$$

- It measures the number of mistakes along the *whole trajectory*.
- Sample complexity can be used as a relevant performance measure in discounted RL problems.

PAC-MDP Algorithms

We are interested in RL algorithms, whose sample complexities are controlled by some functions that are not too large as a function of relevant parameters $S, A, \varepsilon, \delta$ and γ.

PAC-MDP Algorithm

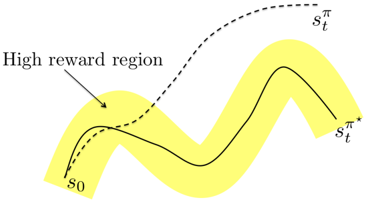
An algorithm $\mathbb{A}$ is called (ε, δ)-**PAC-MDP** if for any ε and δ, the sample complexity of $\mathbb{A}$ is upper bounded w.p. $\geq 1 - \delta$ by some polynomial in

$$S,\ A,\ \frac{1}{\varepsilon},\ \frac{1}{\delta},\ \text{and}\ \frac{1}{1-\gamma}.$$

PAC-MDP $\equiv$ Probably Approximately Correct in MDPs

An Important Remark

Defining Sample Complexity w.r.t. the trajectory of the algorithm is not always meaningful:



$$\mathbb{P}\Big(V^{\pi_t}(\boldsymbol{s_t}) \geq V^\star(\boldsymbol{s_t}) - \varepsilon\Big) \geq 1-\delta$$

Due to exploration, we can end up in states with very low rewards, and being optimal from there may not mean much. A more meaningful criterion would be

$$\mathbb{P}\Big(V^{\pi_t}(\boldsymbol{s_t}) \geq V^\star(\boldsymbol{s_t^{\pi^\star}}) - \varepsilon\Big) \geq 1-\delta$$

where $s_t^{\pi^\star}$ is the state on the trajectory of an optimal agent.

Q-Learning: Online RL

`QL` (`Q-Learning`) for online RL (via OFU):

- Q **Update:**

$$Q_{t+1}(s,a) = \begin{cases} Q_t(s,a) + \alpha_t\Big(r_t + \gamma \max_{b\in\mathcal{A}} Q_t(s_{t+1},b) - Q_t(s,a)\Big) & (s,a) = (s_t,a_t) \\ Q_t(s,a) & \text{else.} \end{cases}$$

- **Action Selection:** trust your current Q_t but use a bit of exploration. Hence, take

$$a_t \sim \pi_t(\cdot|s_t; Q_t)$$

where $\pi_t(\cdot|s_t;Q_t)$ depends on Q_t but uses some exploration too.

Q-Learning: Online RL

Examples of $\pi_t(Q_t)$ with (built-in) exploration device:

- E.g., ε-greedy policy (for some $\varepsilon > 0$)

$$\pi_{\varepsilon\text{-greedy}}(s) = \begin{cases} \operatorname{argmax}_a Q_t(s,a) & \text{w.p. } 1-\varepsilon \\ \text{sample uniformly at random from } \mathcal{A} & \text{w.p. } \varepsilon \end{cases}$$

- E.g., Boltzmann's policy (a.k.a. softmax):

$$\text{at state } s\text{, select action } a \in \mathcal{A} \text{ w.p. } \quad \frac{e^{\eta Q_t(s,a)}}{\sum_{b\in\mathcal{A}} e^{\eta Q_t(s,b)}}$$

  where $\eta > 0$ is a parameter controlling exploration.

These balance exploration-exploitation. But what can be said about the quality of exploration-exploitation?

Such `QL` variants converge to $\pi^\star$ (and play it often). Yet, not sufficient to make `QL` PAC-MDP.

OFU Principle

Optimism in the Face of Uncertainty (OFU)

- A well-known principle in balancing exploration-exploitation in bandits and online RL dating back to (Lai & Robbins, 1985).
- Also known as the Optimism principle

The OFU Principle: In an uncertain world, suppose that the environment is the best possible (in terms of rewards)!

- If the chosen action is optimal $\Longrightarrow$ no penalty
- If sub-optimal $\Longrightarrow$ reducing uncertainty

Optimism in the Face of Uncertainty (OFU)

In bandits, OFU prescribes replacing unknown mean rewards by their corresponding high-probability UCBs. the most prominent example is the `UCB` algorithm.

In MDPs, different implementations exist depending on the approach

- **In model-based:** Select the best candidate environment (among all plausible models/MDPs), i.e. the one leading to the highest possible value function.
- **In model-free:** When updating the Q-function, be optimistic. Initialize all Q-values to their highest possible value and use "reward + exploration bonus" instead of "reward" alone.

This lecture: two OFU-based PAC-MDP algorithms (`UCB-QL`, `MBIE`).

PAC-MDP Algorithms Exist

Some PAC-MDP algorithms:

- Kakade (2003) defined the notion of sample complexity of exploration.
- Some model-based algorithms include:
  - `Rmax` (Brafman & Tennenholtz, 2002), one of the earliest PAC-MDP algorithms.
  - `MBIE` (Strehl & Littman, 2008), `UCRL`γ (Lattimore & Hutter, 2014),
- `Delayed Q-Learning` (Strehl et al., 2006) is the first model-free PAC-MDP algorithm.
- `UCB-QL` (Dong et al., 2020) is a recent model-free PAC-MDP algorithm.

This lecture: `UCB-QL` and `MBIE`, and worst-case lower bound.

`UCB-QL`: UCB + Q-Learning

UCB-QL

UCB-QL is a recent model-free PAC-MDP algorithm presented and analyzed in (Dong et al., 2020).

We present UCB-QL and investigates its theoretical and empirical sample complexity.

- It is model-free and maintains Q functions.
- It has a Q-update resembling the one in QL –hence the name.
- Its main departure from QL (and its variants for off-policy RL) is use of UCB-type exploration to maintain optimism –hence the name (again).

Recap: UCB

Recall UCB in a K-armed bandit (coinciding with an MDP with a single state and K actions):

$$a_t \in \arg\max_{a \in [K]} \mathtt{UCB}_t(a) := \left(\widehat{\mu}_t(a) + \sqrt{\frac{3\log(t)}{2N_t(a)}}\right)$$

We may view it (with some compromise of math rigour) as follows:

$$\widehat{\mu}_{t+1}(a_t) = \left(1 - \frac{1}{N_t(a_t)+1}\right)\widehat{\mu}_t(a_t) + \frac{r_t}{N_t(a_t)+1} \quad \text{(similar to Q-learning update)}$$

$$a_t \in \underset{a}{\operatorname{argmax}} \left(\widehat{\mu}_t(a) + \underbrace{\sqrt{\frac{3\log(t)}{2N_t(a)}}}_{\text{promotion term}}\right)$$

Namely, normalized $Q(a) \equiv \mu(a)$

Recap: UCB

Why not, as in UCB, directly adding an exploration bonus to the Q function?

- To incorporate optimism in Q function, we are interested in having

$$Q_t(s,a) + \square\sqrt{\frac{\log(t)}{N_t(s,a)}}$$

  Not just for action selection but also we want Q_t memorizes this.

- To guarantee optimism, $\square$ should be scaled to be, at least, in the possible range of Q, namely $\frac{R_{\max}}{1-\gamma}$. Indeed, it must be scaled as $\left(\frac{R_{\max}}{1-\gamma}\right)^{3/2}$.
- This is more complicated than UCB, because the notion of value function $Q(s,a)$ is much more complex than a simple mean reward $\mu(a)$.
- A proposal for `QL`-type update + exploration:

$$Q_{t+1}(s_t,a_t) = (1-\alpha_t)Q_t(s_t,a_t) + \alpha_t\left(r_t + \square\sqrt{\frac{\log(t)}{N_t(s_t,a_t)}} + \gamma \max_{a'\in\mathcal{A}} Q_t(s_{t+1},a')\right)$$

UCB-QL

UCB-QL maintains two Q-functions:

- Optimistic Q-function $Q \in \mathbb{R}^{S\times A}$
- Historical minimum Q-function $\widehat{Q} \in \mathbb{R}^{S\times A}$.

The update is performed on Q but actions are taken greedily w.r.t. $\widehat{Q}$. More precisely, at each t

- We update Q using "reward + bonus" $r_t + b_{N_t(s_t,a_t)}$:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha_{N_t(s_t,a_t)}\Big[r_t + \underbrace{b_{N_t(s_t,a_t)}}_{\text{bonus}} + \gamma \max_a \widehat{Q}(s_{t+1},a) - Q(s_t,a_t)\Big]$$

  where for some large enough parameter H (see next slides), we define

$$\alpha_k = \frac{H+1}{H+k}, \quad b_k = \frac{1}{1-\gamma}\sqrt{\frac{32H}{k}\log\frac{SA(k+1)(k+2)}{\delta}}$$

- Then we update $\widehat{Q}$: $\widehat{Q}(s_t,a_t) \leftarrow \min\Big\{\widehat{Q}(s_t,a_t), Q(s_t,a_t)\Big\}$

UCB-QL

- **input:** ε, δ
- **initialization:** For all (s, a),
  - $N(s, a) = 1$
  - $\widehat{Q}(s, a) = Q(s, a) = \frac{R_{\max}}{1-\gamma}$
- **for** $t = 1, 2, \dots$
  - Take $a_t \in \text{argmax}_a \widehat{Q}(s_t, a)$
  - Receive $r_t \sim R(s_t, a_t)$ and $s_{t+1} \sim P(\cdot|s_t, a_t)$
  - Update Q:

    $$Q(s_t, a_t) \leftarrow (1 - \alpha_k)Q(s_t, a_t) + \alpha_k \left[r_t + b_k + \gamma \max_a \widehat{Q}(s_{t+1}, a)\right]$$

    where $k = N(s_t, a_t)$.
  - Update $\widehat{Q}$: $\widehat{Q}(s_t, a_t) \leftarrow \min\left\{\widehat{Q}(s_t, a_t), Q(s_t, a_t)\right\}$
  - $N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$.

See next slide for H, b_k, and α_k.

UCB-QL: Parameters

Recall $k = N_t(s,a)$. Choose

$$\alpha_k = \frac{H+1}{H+k}$$
$$b_k = \frac{1}{1-\gamma}\sqrt{\frac{32H}{k}\log\frac{SAk^2}{\delta}}$$

for some fictitious horizon number $H := H(\gamma, \varepsilon)$.

One can set H to the effective horizon:

$$H = H_{\text{eff}} := \frac{-1}{1-\gamma}\log(\varepsilon(1-\gamma))$$

Then

$$H = H_{\text{eff}} \quad \Longrightarrow \quad b_k = b_{N_t(s,a)} = \widetilde{\mathcal{O}}\left(\sqrt{\frac{H_{\text{eff}}^3}{N_t(s,a)}}\right) = \widetilde{\mathcal{O}}\left(\frac{1}{(1-\gamma)^{3/2}\sqrt{N_t(s,a)}}\right)$$

UCB-QL: Sample Complexity

Sample complexity of UCB-QL in *any* discounted MDP with S states and A actions (i.e., worst-case bound):

Theorem (Sample Complexity of UCB-QL)

For any $\varepsilon > 0$, $\delta \in (0,1)$, *the sample complexity of UCB-QL is bounded by*

$$\widetilde{\mathcal{O}}\left(\frac{SA}{\varepsilon^2(1-\gamma)^7}\log\frac{1}{\delta}\right), \quad \textit{w.p.} \ \geq 1-\delta,$$

where $\widetilde{\mathcal{O}}(\cdot)$ *hides poly-logarithmic terms in* SA, ε^{-1}, *and* $\frac{1}{1-\gamma}$.

$\Longrightarrow$ UCB-QL is PAC-MDP. More precisely:

$$\mathbb{P}\left\{\sum_{t=1}^{\infty} \underbrace{\mathbb{I}\{V^\star(s_t) - V^{\pi_t}(s_t) > \varepsilon\}}_{t \text{ is } \varepsilon\text{-bad}} > \widetilde{\mathcal{O}}\left(\frac{SA}{\varepsilon^2(1-\gamma)^7}\log\frac{1}{\delta}\right)\right\} < \delta,$$

UCB-QL: Proof Idea

The (complicated) proof lies on the following facts:

- Implementing optimism: $\widehat{Q}_t \geq Q^\star$ for all t w.h.p. In particular,

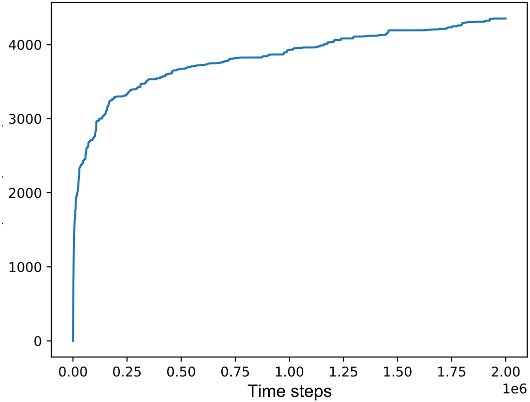
$$\begin{aligned}\widehat{Q}_t(s_t, a_t) &\geq \widehat{Q}_t(s_t, \pi^\star(s_t)) \qquad \text{(by algorithm design)}\\ &\geq Q^\star(s_t, \pi^\star(s_t)) \qquad \text{(by optimism)}\\ &= V^\star(s_t) \qquad \text{(by definition of } Q^\star\text{)}\end{aligned}$$

- So to count ε-bad steps, one can upper bound steps where

$$\widehat{Q}_t(s_t, a_t) - Q^\star(s_t, a_t) > \varepsilon$$

- Carefully chosen H and α_k guarantee that $\widehat{Q}_t$ is not overly optimistic.

Numerical Experiments



The number of ε-steps for a single run of UCB-QL in 5-state RiverSwim ($\gamma = 0.9$, $\varepsilon = 0.1$, $\delta = 0.05$).

MBIE: Model-Based Interval Estimation

OFU: Model-Based

`MBIE` (Strehl & Littman, 2008) is a model-based PAC-MDP algorithm designed based on OFU.

Model-based recipe for the optimism principle (OFU):

- **Step 1:** Maintains a set of plausible MDPs (models) (i.e., consistent with history h_t). This can be done by defining high-probability confidence sets for R and P, and forming a corresponding set of MDPs.
- **Step 2:** Choose an optimistic model (among all models) and an optimistic policy leading to the highest value.

Step 1: Confidence Sets - Empirical MDP

For any $t \geq 1$, define

- $N_t(s,a,s')$: number of visits, up to t, to (s,a) followed by a visit to s'

$$N_t(s,a,s') = \sum_{i=1}^{t-1} \mathbb{I}\{s_i = s, a_i = a, s_{i+1} = s'\}$$

- $N_t(s,a)$: number of visits, up to t, to (s,a)

$$N_t(s,a) = \sum_{s' \in \mathcal{S}} N_t(s,a,s')$$

Empirical Estimator for P:

$$\forall s' \in \mathcal{S}: \quad \widehat{P}_t(s'|s,a) = \begin{cases} \frac{N_t(s,a,s')}{N_t(s,a)} & \text{if } N_t(s,a) > 0 \\ \frac{1}{S} & \text{otherwise} \end{cases}$$

Empirical Estimator for R:

$$\widehat{R}_t(s,a) = \frac{1}{N_t(s,a)} \sum_{i=1}^{t-1} r_i \mathbb{I}\{s_i = s, a_i = a\}$$

Empirical MDP

The empirical MDP:

$$\widehat{M}_t = (\mathcal{S}, \mathcal{A}, \widehat{P}_t, \widehat{R}_t, \gamma)$$

Why not only using $\widehat{M}_t$. I.e., finding the optimal policy in $\widehat{\pi}_t^\star$ and taking $a_t = \widehat{\pi}_t^\star(s_t)$ each step.

$\rightarrow$ No exploration-exploitation tradeoff. Will not lead to a PAC-MDP algorithm.

Step 1: Confidence Sets

$\delta \in (0,1)$ is given.

Confidence Set for R:

- Define a confidence set for $R(s,a)$ as

$$C_{s,a} = \left\{ \lambda \in [0,1] : |\widehat{R}_t(s,a) - \lambda| \leq \beta_{N_t(s,a)} \right\}$$

  for some suitable function $\beta_{N_t(s,a)}$.

- For example, using Hoeffding's inequality (combined with Laplace's methods):

$$\beta_n = \sqrt{\tfrac{1}{2n}(1+\tfrac{1}{n}) \log \tfrac{SA\sqrt{n+1}}{\delta}}, \quad n \in \mathbb{N}.$$

$$\mathbb{P}\Big(\forall t \geq 1,\ \forall (s,a) :\ R(s,a) \in C_{s,a}\Big) \geq 1 - \delta$$

Step 1: Confidence Sets

$\delta \in (0,1)$ is given.

Confidence Set for P:

- Define a confidence set for $P(\cdot|s,a)$ as

$$C'_{s,a} = \left\{ q \in \Delta(\mathcal{S}) : \left\| \widehat{P}_t(\cdot|s,a) - q \right\|_1 \leq \beta'_{N_t(s,a)} \right\}$$

  for some suitable function $\beta'_{N_t(s,a)}$.

- For example, using Weissman's inequality (combined with Laplace's methods):

$$\beta'_n = \sqrt{\tfrac{2}{n}\left(1+\tfrac{1}{n}\right) \log \tfrac{SA(2^S-2)\sqrt{n+1}}{\delta}}$$

$$\mathbb{P}\Big(\forall t \geq 1, \forall (s,a) : \ P(\cdot|s,a) \in C'_{s,a}\Big) \geq 1-\delta$$

Step 1: Set of Models

Confidence sets $\{C_{s,a}, C'_{s,a}\}_{s\in\mathcal{S},a\in\mathcal{A}}$ yield a set of models (i.e., MDPs) consistent with the history $h_t = (s_1, a_1, r_1, \ldots, s_{t-1}, a_{t-1}, r_{t-1}, s_t)$:

$$\mathcal{M}_t = \Big\{ M' = (\mathcal{S}, \mathcal{A}, P', R', \gamma) : \quad P'(\cdot|s,a) \in C'_{s,a} \text{ and } R'(s,a) \in C_{s,a}, \ \forall s, a \Big\}$$

- $\mathcal{M}_t$ collects *all MDPs* that could be a candidate for the true Model M (in view of h_t).
- Moreover, M is trapped in $\mathcal{M}_t$ with high probability, simultaneously for all t:

$$\mathbb{P}(\forall t \geq 1 : M \in \mathcal{M}_t) \geq 1 - 2\delta$$

Step 2: Planning

Step 2: Planning. To implement OFU, we wish to find

$$\pi_t \in \arg \max_{M' \in \mathcal{M}_t} \max_{\pi \in \Pi^{\mathsf{SD}}} V^{\pi}_{M'}$$

and then we choose $a_t = \pi_t(s_t)$.

Alternatively, by Bellman's optimality equation, we wish to find $\widetilde{Q}(s,a)$ satisfying: For all (s,a),

$$\widetilde{Q}(s,a) = \max_{R'(s,a) \in C_{s,a}} R'(s,a) + \gamma \max_{P'(\cdot|s,a) \in C'_{s,a}} \sum_x P'(x|s,a) \max_{a'} \widetilde{Q}(x,a')$$

where $\widetilde{Q}(s,a)$ is indeed the optimal Q-function of $\mathcal{M}_t$.

Step 2: Planning

$$\widetilde{Q}(s,a) = \max_{R'(s,a)\in C_{s,a}} R'(s,a) + \gamma \max_{P'(\cdot|s,a)\in C'_{s,a}} \sum_x P'(x|s,a) \max_{a'} \widetilde{Q}(x,a')$$

Compared to optimality equations for MDPs, we have two extra maximizations.

- The one in blue admits a closed-form solution:

$$\max_{R'(s,a)\in C_{s,a}} R'(s,a) = \widehat{R}_t(s,a) + \beta_{N_t(s,a)}$$

- No closed-form solution to the second. However, for a fixed $u \in \mathbb{R}^{S\times A}$, the problem

$$\max_{p\in C'(s,a)} \sum_x p(x) \max_{a'} u(x,a')$$

  can be solve using a simple procedure thanks to the shape of $C'_{s,a}$.

The second optimization problem can be efficiently solved using `Extended Value Iteration (EVI)`.

MBIE

- **input:** ε, δ
- **initialization:** For all (s, a),
  - $N(s, a) = 0$
  - $\widetilde{Q}(s, a) = \frac{R_{\max}}{1-\gamma}$
- **for** $t = 1, 2, \dots$
  - Compute estimates $\widehat{P}_t$ and $\widehat{R}_t$
  - Find $\widetilde{Q}$ by solving Bellman's equation for $\mathcal{M}_t$ using `EVI`
  - Choose $a_t \in \mathrm{argmax}_a \, \widetilde{Q}(s_t, a)$
  - Receive reward $r_t \sim R(s_t, a_t)$ and next-state $s_{t+1} \sim P(\cdot|s_t, a_t)$
  - Update $N(s_t, a_t) \leftarrow N(s_t, a_t) + 1$.

MBIE: EVI

- **input:** ε
- **initialization:** Select $\widetilde{Q}_0 \in \mathbb{R}^{S\times A}$ arbitrarily. Set $n = -1$.
- **repeat:**
  - Increment n
  - Compute, for each (s, a),

$$R'(s,a) = \widehat{R}_t(s,a) + \beta_{N(s,a)}$$

$$P'(\cdot|s,a) \in \operatorname{argmax}\left\{ \sum_{x\in\mathcal{S}} q(x) \max_{a'} \widetilde{Q}_n(x,a') : q \in C'_{s,a} \right\}$$

  - Update, for each (s, a),

$$\widetilde{Q}_{n+1}(s,a) = R'(s,a) + \gamma \sum_{x\in\mathcal{S}} P'(x|s,a) \max_{a'} \widetilde{Q}_n(x,a')$$

  until $\|\widetilde{Q}_{n+1} - \widetilde{Q}_n\|_\infty < \frac{\varepsilon(1-\gamma)}{2\gamma}$
- **output:** $\widetilde{Q}_n$

MBIE: EVI

Algorithm for solving

$$\max_{q \in C'_{s,a}} \sum_{x \in \mathcal{S}} q(x)u(x)$$

Index $\mathcal{S} = \{s_1, s_2, \dots, s_S\}$, and assume w.l.o.g. that

$$u(s_1) \geq u(s_2) \geq \ldots \geq u(s_S)$$

- **initialization:** $q = \widehat{P}_t(\cdot|s,a)$
- Set $q(s_1) = \widehat{P}_t(s_1|s,a) + \frac{1}{2}\beta'_{N_t(s,a)}$
- $\ell = S$
- **while:** $\sum_{x \in \mathcal{S}} q(x) > 1$
  - Set $q(s_\ell) = \max\left\{0, 1 - \sum_{x \neq s_\ell} q(x)\right\}$
  - Decrement ℓ
- **output:** q

MBIE: Sample Complexity

Sample complexity of MBIE in any discounted MDP with S states and A actions:

Theorem (Sample Complexity of MBIE)

For any $\varepsilon > 0$, $\delta \in (0,1)$, the sample complexity of MBIE is bounded by

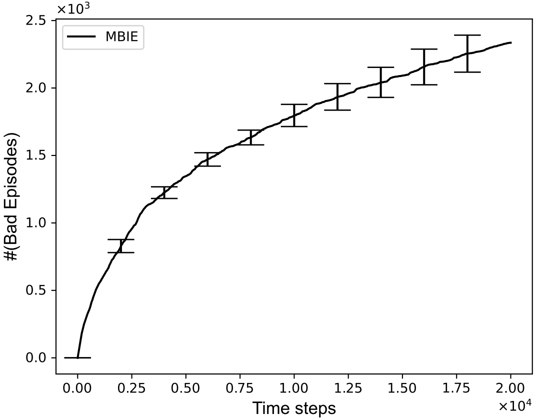
$$\widetilde{\mathcal{O}}\left(\frac{S^2 A}{\varepsilon^3 (1-\gamma)^6} \log\left(\frac{1}{\delta}\right)\right), \quad \textit{w.p.} \ \geq 1-\delta,$$

where $\widetilde{\mathcal{O}}(\cdot)$ hides poly-logarithmic terms in SA, ε^{-1}, and $\frac{1}{1-\gamma}$.

$\Longrightarrow$ MBIE is PAC-MDP. More precisely:

$$\mathbb{P}\left\{ \sum_{t=1}^{\infty} \underbrace{\mathbb{I}\{V^\star(s_t) - V^{\pi_t}(s_t) > \varepsilon\}}_{t \text{ is } \varepsilon\text{-bad}} > \widetilde{O}\left(\frac{S^2 A}{\varepsilon^3 (1-\gamma)^6} \log\frac{1}{\delta}\right) \right\} < \delta,$$

Numerical Experiments



The number of ε-steps under `MBIE` in RiverSwim ($\gamma = 0.93$, $\varepsilon = 0.12$, $\delta = 0.05$).

Worst-Case Lower Bound on Sample Complexity

Worst-Case Lower Bound

How good is the sample complexity bound of `UCB-QL`? Could it be improved?

To answer these, we need to derive lower bounds on sample complexity.

- Problem-dependent lower bound
- Worst-case lower bound

Worst-Case Lower Bound

The following lower bound on sample complexity is due to (Lattimore & Hutter, 2014).

Theorem (Worst-Case Lower Bound)

Let $S \geq 4$, A, γ, δ, and ε, with $\varepsilon(1-\gamma)$ being sufficiently small. For any learning algorithm $\mathbb{A}$, there exists a discounted MDP M with S states, A actions, and discount factor γ such that with probability at least δ, the number of ε-bad steps of $\mathbb{A}$ is larger than

$$c_1 \cdot \frac{SA}{\varepsilon^2(1-\gamma)^3} \log\left(\frac{c_2 S}{\delta}\right)$$

for some universal constants $c_1, c_2 > 0$. Namely, w.p. higher than δ,

$$\sum_{t=1}^{\infty} \mathbb{I}\{V^{\mathbb{A}_t}(s_t) < V^\star(s_t) - \varepsilon\} \geq c_1 \cdot \frac{SA}{\varepsilon^2(1-\gamma)^3} \log\left(\frac{c_2 S}{\delta}\right)$$

- The theorem asserts a fundamental performance limit on sample complexity which no algorithm can beat.

Worst-Case Lower Bound

$$\underbrace{\Omega\left(\frac{SA}{\varepsilon^2(1-\gamma)^3}\log\left(\frac{S}{\delta}\right)\right)}_{\text{worst-case LB}} \quad \text{vs.} \quad \underbrace{\widetilde{\mathcal{O}}\left(\frac{SA}{\varepsilon^2(1-\gamma)^7}\log\left(\frac{SA}{\delta}\right)\right)}_{\texttt{UCB-QL}\text{ UB}}$$

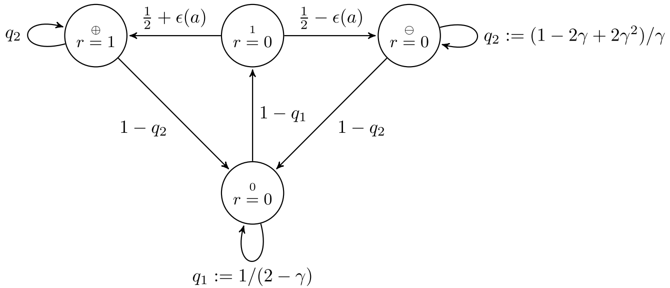
The sample complexity of `UCB-QL`

- Has optimal dependence on S, A, and ε, δ (ignoring poly-log factors).
- Could be improved by a factor of $1/(1-\gamma)^4$.
- `UCRL`γ (Lattimore & Hutter, 2014), a variant of `UCRL2` for discounted MDPs, achieves:

$$\widetilde{\mathcal{O}}\left(\frac{S^2 A}{\varepsilon^2(1-\gamma)^3}\log\left(\frac{SA}{\delta}\right)\right)$$

- This gap was closed in 2021.

Worst-Case MDP: $S = 4$



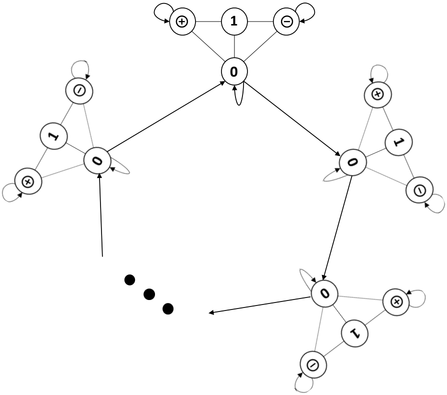
A family of worst-case 4-state MDPs (Lattimore & Hutter, 2014):

- $\mathcal{S} = \{\oplus, \ominus, 1, 0\}$ and A actions per each state.
- All actions have identical rewards, and the rewarding state is $+$.
- $\varepsilon(a^\star) = 16\varepsilon(1 - \gamma)$ for some $a = a^\star$, and $\varepsilon(a) = 0$ for $a \neq a^\star$.
- $s = \oplus, \ominus$ are highly abosorbing. $s = 0$ traps the agent for around $\frac{1}{1-\gamma}$ steps (in expectation).

Worst-Case MDP: $S > 4$

A worst-case instance for $S > 4$ can be constructed by chaining $S/4$ of the previous 4-state MDPs together

- State 0 of k-th one transits with a very small probability to state 0 of the $(k+1)$-th.
- q_1 must be slightly modified too; see (Lattimore & Hutter, 2014).



References

- Sham Kakade, "On the sample complexity of reinforcement learning," *PhD Thesis, University of London, University College London*, 2003.
- Ronen Brafman and Moshe Tennenholtz, "R-max -A general polynomial time algorithm for near-optimal reinforcement learning," *Journal of Machine Learning Research*, 2002.
- Alexander Strehl et al., "PAC model-free reinforcement learning" *International Conference on Machine Learning*, 2006.
- Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael Jordan, "Is Q-learning provably efficient?" *Advances in Neural Information Processing Systems 31*, 2018.
- Kefan Dong, Yuanhao Wang, Xiaoyu Chen, and Liwei Wang, "Q-learning with UCB exploration is sample efficient for infinite-horizon MDP" *International Conference on Learning Representations*, 2020.
- Tor Lattimore and Marcus Hutter, "Near-optimal PAC bounds for discounted MDPs" *Theoretical Computer Science*, 2014.
- Eyal Even-Dar and Yishay Mansour, "Convergence of optimistic and incremental Q-learning," *Advances in Neural Information Processing Systems 14*, 2001.