Lipschitz Lifelong Reinforcement Learning

Erwan Lecarpentier [1] David Abel [2] Kavosh Asadi [2]
Yuu Jinnai [2] Emmanuel Rachelson [1] Michael L. Littman [2]
[1] ISAE-SUPAERO, Université de Toulouse, [2] Brown University

erwan.lecarpentier@isae-supaero.fr

July 10, 2019

Reinforcement Learning

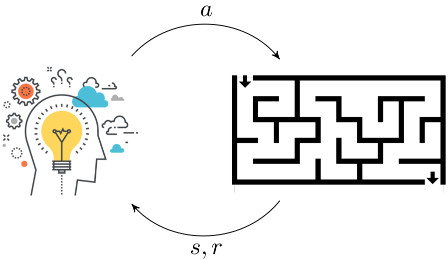
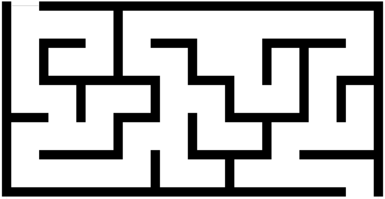


Figure: Reinforcement Learning framework

Reinforcement Learning

Markov Decision Process (MDP)

An MDP is a 4-tuple $\{\mathcal{S}, \mathcal{A}, T, R\}$:

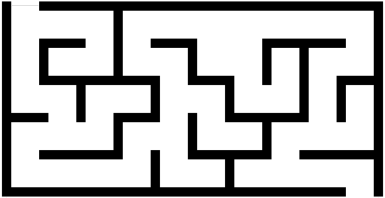


Reinforcement Learning

Markov Decision Process (MDP)

An MDP is a 4-tuple $\{\mathcal{S}, \mathcal{A}, T, R\}$:

- $\mathcal{S}$ is a state space; (x, y)

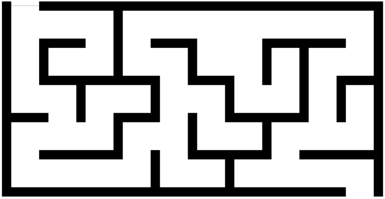


Reinforcement Learning

Markov Decision Process (MDP)

An MDP is a 4-tuple $\{\mathcal{S}, \mathcal{A}, T, R\}$:

- $\mathcal{S}$ is a state space; (x, y)
- $\mathcal{A}$ is an action space; $\{\uparrow, \rightarrow, \downarrow, \leftarrow\}$

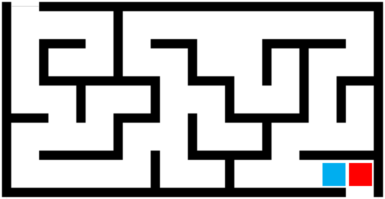


Reinforcement Learning

Markov Decision Process (MDP)

An MDP is a 4-tuple $\{\mathcal{S}, \mathcal{A}, T, R\}$:

- $\mathcal{S}$ is a state space; (x, y)
- $\mathcal{A}$ is an action space; $\{\uparrow, \rightarrow, \downarrow, \leftarrow\}$
- $T^{a}_{s,s'} = \mathbf{Pr}(s' \mid s, a)$; $T^{\leftarrow}_{{\color{red}\blacksquare},{\color{cyan}\blacksquare}} = 0.9,\ T^{\uparrow}_{{\color{red}\blacksquare},{\color{cyan}\blacksquare}} = 0.1$

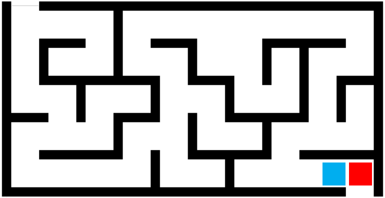


Reinforcement Learning

Markov Decision Process (MDP)

An MDP is a 4-tuple $\{\mathcal{S}, \mathcal{A}, T, R\}$:

- $\mathcal{S}$ is a state space; $\color{green}(x, y)$
- $\mathcal{A}$ is an action space; $\color{green}\{\uparrow, \rightarrow, \downarrow, \leftarrow\}$
- $T^{a}_{s,s'} = \mathbf{Pr}(s' \mid s, a)$; $\color{green}T^{\leftarrow}_{\color{red}\blacksquare,\color{cyan}\blacksquare} = 0.9,\ T^{\uparrow}_{\color{red}\blacksquare,\color{cyan}\blacksquare} = 0.1$
- R^{a}_{s} is a reward function; $\color{green}R^{\downarrow}_{\color{red}\blacksquare} = 1,\ R^{\leftarrow}_{\color{red}\blacksquare} = 0$



Reinforcement Learning

What is "solving an MDP"?

Reinforcement Learning

What is "solving an MDP"?

Definition 1

Policy: $\pi : s \mapsto a$

Reinforcement Learning

What is "solving an MDP"?

Definition 1

Policy: $\pi : s \mapsto a$

Definition 2

Value function:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{s_t}^{a_t} \mid s_0 = s, s_{t+1} \sim T_{s_t,\cdot}^{a_t}, a_t = \pi(s_t)\right]$$

Reinforcement Learning

What is "solving an MDP"?

Definition 1

Policy: $\pi : s \mapsto a$

Definition 2

Value function:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{s_t}^{a_t} \mid s_0 = s, s_{t+1} \sim T_{s_t,\cdot}^{a_t}, a_t = \pi(s_t)\right]$$

Definition 3

Optimal policy: $\pi^* = \arg\max_{\pi} V^{\pi}(s), \forall s \in \mathcal{S}$

Reinforcement Learning

Some more definitions:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{s_t}^{a_t} \mid s_0 = s, s_{t+1} \sim T_{s_t,\cdot}^{a_t}, a_t = \pi(s_t)\right]$$

Definition 4

Q-Value function:

$$Q^\pi(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{s_t}^{a_t} \mid s_0 = s, a_0 = a, s_{t+1} \sim T_{s_t,\cdot}^{a_t}, a_t = \pi(s_t)\right]$$

Reinforcement Learning

Some more definitions:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{s_t}^{a_t} \mid s_0 = s, s_{t+1} \sim T_{s_t,\cdot}^{a_t}, a_t = \pi(s_t)\right]$$

Definition 4

Q-Value function:

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_{s_t}^{a_t} \mid s_0 = s, a_0 = a, s_{t+1} \sim T_{s_t,\cdot}^{a_t}, a_t = \pi(s_t)\right]$$

Definition 5

Optimal value functions:

$$V^*(s) = \max_{\pi} V^{\pi}(s) \qquad Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a)$$

Machine Learning big picture

Supervised Learning

$$\hat{F}(x_i) = \hat{y}_i$$

Unsupervised Learning

$$\hat{F}(x_i)$$

Reinforcement Learning

Optimality Bellman equation:

$$\hat{Q}(s_i, a_i) = R_{s_i}^{a_i} + \gamma \mathbb{E}_{s' \sim T_{s_i \cdot}^{a_i}} \left[\max_{a'} \hat{Q}(s', a') \right]$$

R-MAX: an example of learning agent

R-MAX: an example of learning agent

- Optimistic initialization: $\hat{R}^a_s = R_{\max}$, $\hat{T}^a_{s,s} = 1, \forall s, a \in \mathcal{S} \times \mathcal{A}$;

R-MAX: an example of learning agent

- Optimistic initialization: $\hat{R}_s^a = R_{\max}, \hat{T}_{s,s}^a = 1, \forall s, a \in \mathcal{S} \times \mathcal{A}$;
- Learn the true model online $(\hat{R}, \hat{T}) \to (R, T)$;

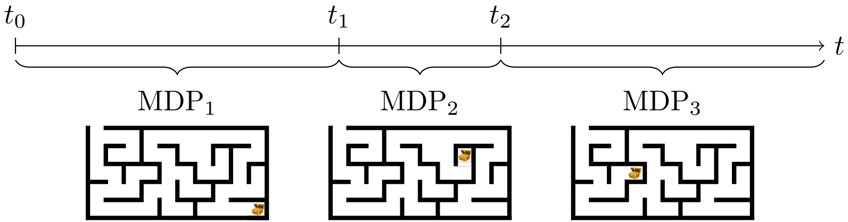
R-MAX: an example of learning agent

- Optimistic initialization: $\hat{R}_s^a = R_{\max}, \hat{T}_{s,s}^a = 1, \forall s, a \in \mathcal{S} \times \mathcal{A}$;
- Learn the true model online $(\hat{R}, \hat{T}) \rightarrow (R, T)$;
- Find an ϵ-optimal policy with high probability in polynomial time;

R-MAX: an example of learning agent

- Optimistic initialization: $\hat{R}_s^a = R_{\max}, \hat{T}_{s,s}^a = 1, \forall s, a \in \mathcal{S} \times \mathcal{A}$;
- Learn the true model online $(\hat{R}, \hat{T}) \to (R, T)$;
- Find an ϵ-optimal policy with high probability in polynomial time;
- One of the only algorithms with a guaranteed convergence rate.

Lifelong Reinforcement Learning



Transfer

How can we leverage the knowledge acquired during interactions with previous MDPs to speed-up the resolution of the current task?

Take home message

Contributions

- Theoretical study of the Lipschitz **Continuity of V^* and Q^* in the MDP space**;
- Proposal of a **practical, non-negative, transfer method** based on a local distance between MDPs;
- Proposal and study of a **PAC-MDP algorithm** applying this transfer method in the Lifelong RL setting.

Notation

$$M = \langle \mathcal{S}, \mathcal{A}, R, T \rangle \text{ new MDP}$$

$$\bar{M} = \langle \mathcal{S}, \mathcal{A}, \bar{R}, \bar{T} \rangle \text{ explored MDP}$$

Idea

The closer two MDPs, the closer their optimal value functions.

$\downarrow$

Can we do value transfer with that?

Idea

The closer two MDPs, the closer their optimal value functions.

$\downarrow$

Can we do value transfer with that?

$$|Q^*_M(s,a) - Q^*_{\bar{M}}(s,a)| \leq d_{\mathcal{M}}(M, \bar{M})$$

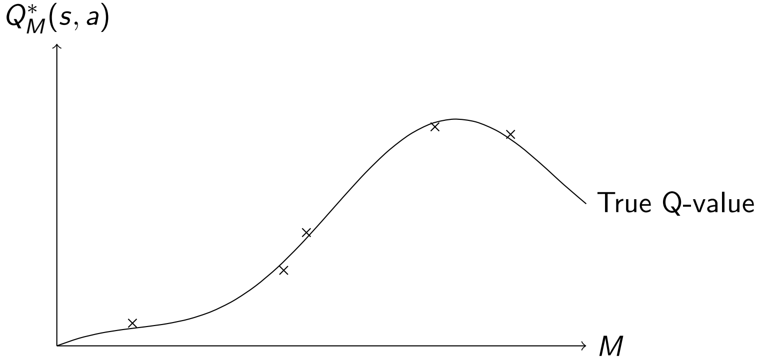
$$\downarrow$$

$$Q^*_M(s,a) \leq U(s,a)$$

$$U(s,a) := Q^*_{\bar{M}}(s,a) + d_{\mathcal{M}}(M, \bar{M})$$

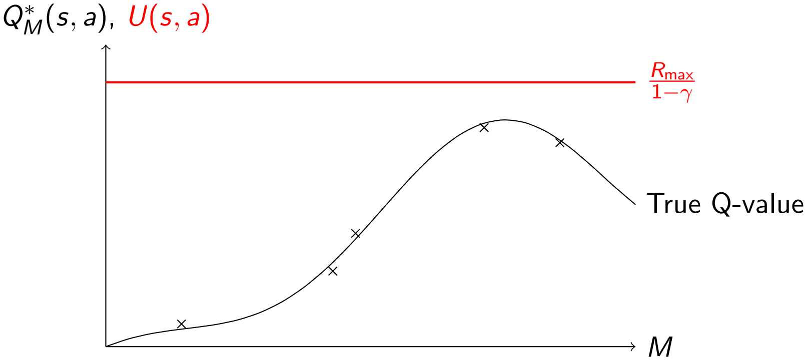
Idea

$M = \langle \mathcal{S}, \mathcal{A}, R, T \rangle \in \mathcal{M}$
$s, a \in \mathcal{S} \times \mathcal{A}$



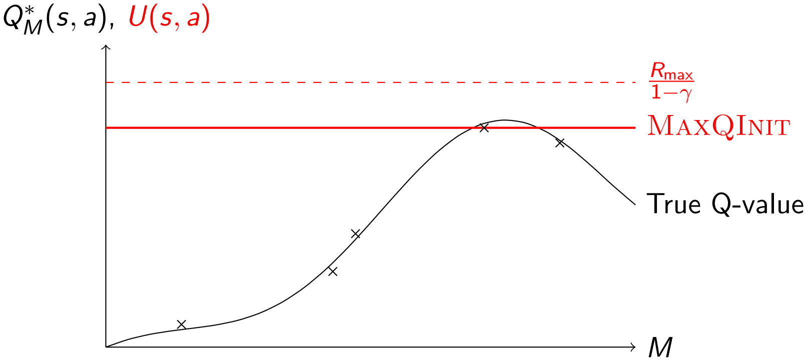
Idea

$$M = \langle \mathcal{S}, \mathcal{A}, R, T \rangle \in \mathcal{M}$$
$$s, a \in \mathcal{S} \times \mathcal{A}$$



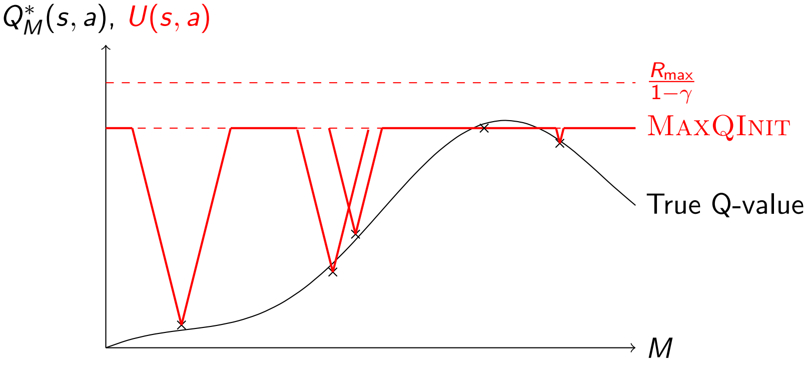
Idea

$M = \langle \mathcal{S}, \mathcal{A}, R, T \rangle \in \mathcal{M}$
$s, a \in \mathcal{S} \times \mathcal{A}$



Idea

$$M = \langle \mathcal{S}, \mathcal{A}, R, T \rangle \in \mathcal{M}$$
$$s, a \in \mathcal{S} \times \mathcal{A}$$



Lipschitz bounds

Local model pseudo-metric

Definition 1 Local pseudo-metric between two models

For two MDPs $M = \langle \mathcal{S}, \mathcal{A}, R, T \rangle$ and $\bar{M} = \langle \mathcal{S}, \mathcal{A}, \bar{R}, \bar{T} \rangle$, we define the distance between their models at $(s, a) \in \mathcal{S} \times \mathcal{A}$ as:

$$D_f \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right) (s, a) = |R_s^a - \bar{R}_s^a| + \sum_{s'} f(s') |T_{ss'}^a - \bar{T}_{ss'}^a|$$

defined for any function $f : \mathcal{S} \to \mathbb{R}^+$

Local continuity

Theorem 1 Local continuity

For any two MDPs M and $\bar{M}$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$|Q^*_M(s, a) - Q^*_{\bar{M}}(s, a)| \leq d^{\bar{M}}_M(s, a)$$

where $d^{\bar{M}}_M$ is defined with the following fixed-point equation:

$$d^{\bar{M}}_M(s, a) = D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s, a) + \gamma \sum_{s'} T^a_{ss'} \max_{a'} d^{\bar{M}}_M(s', a')$$

Local continuity

Theorem 1 Local continuity

For any two MDPs M and $\bar{M}$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$|Q^*_M(s, a) - Q^*_{\bar{M}}(s, a)| \leq d^{\bar{M}}_M(s, a)$$

where $d^{\bar{M}}_M$ is defined with the following fixed-point equation:

$$d^{\bar{M}}_M(s, a) = D_{\gamma V^*_{\bar{M}}} \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right)(s, a) + \gamma \sum_{s'} T^a_{ss'} \max_{a'} d^{\bar{M}}_M(s', a')$$

Remarks:

1. Selected $f : s \mapsto \gamma V^*_{\bar{M}}(s)$ for the local model pseudo-metric
2. **Local** Lipschitz continuity of the optimal Q-function
3. $d^{\bar{M}}_M(s, a)$ is asymmetric
4. $d^{\bar{M}}_M(s, a)$ can be computed with dynamic programming

Global continuity

Corollary 1 Global continuity

For any two MDPs M and $\bar{M}$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$|Q^*_M(s, a) - Q^*_{\bar{M}}(s, a)| \leq d^{\bar{M}}_M$$

$$d^{\bar{M}}_M := \frac{1}{1-\gamma} \max_{s,a} \left[D_{\gamma V^*_{\bar{M}}} \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right) (s, a) \right]$$

Global continuity

Corollary 1 Global continuity

For any two MDPs M and $\bar{M}$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$,

$$|Q^*_M(s, a) - Q^*_{\bar{M}}(s, a)| \leq d_M^{\bar{M}}$$

$$d_M^{\bar{M}} := \frac{1}{1-\gamma} \max_{s,a} \left[D_{\gamma V^*_{\bar{M}}} \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right) (s, a) \right]$$

Remarks:

1. **Global** Lipschitz continuity of the optimal Q-function
2. Little practical use because learning the maximum local model pseudo-metric is as difficult as learning the new MDP M.

Lipschitz bound

Definition 2: Lipschitz bound

Given two MDPs M and $\bar{M}$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, the **Lipschitz bound on Q^*_M induced by $Q^*_{\bar{M}}$** is defined by:

$$U_{\bar{M}}(s, a) := Q^*_{\bar{M}}(s, a) + \min \left[d_M^{\bar{M}}(s, a), d_{\bar{M}}^{M}(s, a) \right]$$

Lipschitz bound

Definition 2: Lipschitz bound

Given two MDPs M and $\bar{M}$, for all $(s, a) \in \mathcal{S} \times \mathcal{A}$, the **Lipschitz bound on Q^*_M induced by $Q^*_{\bar{M}}$** is defined by:

$$U_{\bar{M}}(s, a) := Q^*_{\bar{M}}(s, a) + \min \left[d^{\bar{M}}_M(s, a), d^M_{\bar{M}}(s, a) \right]$$

Obviously we have:

$$Q^*_M(s, a) \leq U_{\bar{M}}(s, a)$$

How to upper-bound $U_{\bar{M}}(s, a)$?

Upper-bound on Lipschitz bound

Lipschitz bound:

$$U_{\bar{M}}(s,a) = Q^*_{\bar{M}}(s,a) + \min\left[d^{\bar{M}}_{M}(s,a), d^{M}_{\bar{M}}(s,a)\right]$$

$$d^{\bar{M}}_{M}(s,a) = D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s,a) + \gamma \sum_{s'} T^a_{ss'} \max_{a'} d^{\bar{M}}_{M}(s',a')$$

Upper-bound on Lipschitz bound

Lipschitz bound:

$$U_{\bar{M}}(s,a) = Q^*_{\bar{M}}(s,a) + \min\left[d_M^{\bar{M}}(s,a), d_{\bar{M}}^{M}(s,a)\right]$$

$$d_M^{\bar{M}}(s,a) = D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s,a) + \gamma \sum_{s'} T^a_{ss'} \max_{a'} d_M^{\bar{M}}(s',a')$$

- Known upper-bound

Upper-bound on Lipschitz bound

Lipschitz bound:

$$U_{\bar{M}}(s,a) = Q^*_{\bar{M}}(s,a) + \min\left[d^{\bar{M}}_{M}(s,a), d^{M}_{\bar{M}}(s,a)\right]$$

$$d^{\bar{M}}_{M}(s,a) = D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s,a) + \gamma \sum_{s'} T^a_{ss'} \max_{a'} d^{\bar{M}}_{M}(s',a')$$

- Known upper-bound
- Maximization over the unknown model(s)

Upper-bound on Lipschitz bound

Lipschitz bound:

$$U_{\bar{M}}(s,a) = Q^*_{\bar{M}}(s,a) + \min\left[d^{\bar{M}}_{M}(s,a), d^{M}_{\bar{M}}(s,a)\right]$$

$$d^{\bar{M}}_{M}(s,a) = D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s,a) + \gamma \sum_{s'} T^{a}_{ss'} \max_{a'} d^{\bar{M}}_{M}(s',a')$$

- Known upper-bound
- Maximization over the unknown model(s)
- Maximize over s' if unknown

Upper-bound on Lipschitz bound

Lipschitz bound:

$$U_{\bar{M}}(s,a) = Q^*_{\bar{M}}(s,a) + \min\left[d^{\bar{M}}_{M}(s,a), d^{M}_{\bar{M}}(s,a)\right]$$

$$d^{\bar{M}}_{M}(s,a) = D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s,a) + \gamma \sum_{s'} T^{a}_{ss'} \max_{a'} d^{\bar{M}}_{M}(s',a')$$

- Known upper-bound
- Maximization over the unknown model(s)
- Maximize over s' if unknown

Notation: from now on $U_{\bar{M}}(s,a)$ refers to the upper-bound on the Lipschitz upper-bound.

Improved upper-bound

Notation: $K :=$ set of known state-action pairs in current MDP.

Improved upper-bound

Notation: $K :=$ set of known state-action pairs in current MDP.

Definition 3 Improved upper-bound

Given a set of Lipschitz bounds $\left\{U_{\bar{M}_1}, U_{\bar{M}_2}, \cdots\right\}$, the improved upper-bound on the R-MAX bound is defined by:

$$U(s,a) = \begin{cases} R_s^a + \gamma \sum_{s'} T_{ss'}^a \max_{a'} U(s',a') & \text{if } (s,a) \in K \\ \min\left[\frac{R_{\max}}{1-\gamma}, U_{\bar{M}_1}(s,a), U_{\bar{M}_2}(s,a), \cdots\right] & \text{else} \end{cases} \quad (1)$$

Improved upper-bound

Notation: $K :=$ set of known state-action pairs in current MDP.

Definition 3 Improved upper-bound

Given a set of Lipschitz bounds $\left\{ U_{\bar{M}_1}, U_{\bar{M}_2}, \cdots \right\}$, the improved upper-bound on the R-MAX bound is defined by:

$$U(s,a) = \begin{cases} R_s^a + \gamma \sum_{s'} T_{ss'}^a \max_{a'} U(s', a') & \text{if } (s,a) \in K \\ \min\left[\frac{R_{\max}}{1-\gamma}, U_{\bar{M}_1}(s,a), U_{\bar{M}_2}(s,a), \cdots \right] & \text{else} \end{cases} \quad (1)$$

Remarks:

1. Can be computed with dynamic programming
2. Influence of the Lipschitz bounds is propagated even for known state-action pairs.

Lipschitz R-MAX algorithm

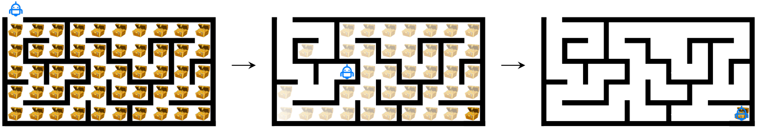
Algorithm 1 Lipschitz R-MAX algorithm

for each sampled MDP **do**
  for $t = 1, 2, \cdots$ **do**
    s current state
    $a = \arg\max_{a'} U(s, a')$
    Observe reward r and next state s'
    if enough observations for (s, a) **then**
      Update model at (s, a)
      for each known MDP $\bar{M}$ **do**
        Update $U_{\bar{M}}$ # Dynamic Programming
      Update U with Equation 1   # Dynamic Programming
  Save learned model

Lipschitz R-MAX algorithm

R-MAX:

Lipschitz R-MAX:



Lipschitz R-MAX analysis

Property 1 Sample complexity

With probability $1 - \delta$, Lipschitz R-MAX algorithm achieves an ϵ-optimal return in the MDP M for all but

$$\mathcal{O}\left(\frac{|\{s, a \in \mathcal{S} \times \mathcal{A} \mid U(s, a) \geq V^*_M(s) - \epsilon\}|}{\epsilon^3 (1 - \gamma)^3} \right)$$

time-steps, with U defined in Equation 1.

Lipschitz R-MAX analysis

Property 1 Sample complexity

With probability $1-\delta$, Lipschitz R-MAX algorithm achieves an ϵ-optimal return in the MDP M for all but

$$\mathcal{O}\left(\frac{|\{s, a \in \mathcal{S} \times \mathcal{A} \mid U(s,a) \geq V^*_M(s) - \epsilon\}|}{\epsilon^3(1-\gamma)^3}\right)$$

time-steps, with U defined in Equation 1.

Property 2 Computational complexity

The total computation complexity of Lipschitz R-MAX is

$$\mathcal{O}\left(B + \frac{S^2A^2(S+\ln(A))(N+1)}{(1-\gamma)} \ln \frac{1}{\epsilon(1-\gamma)}\right)$$

with B the number of time steps, ϵ the precision of the value iteration algorithm and N the memory size.

Improving Lipschitz R-MAX

Issue: upper-bounds on local distances $D_{\gamma V^*_{\bar{M}}}\left(\langle R, T\rangle, \langle \bar{R}, \bar{T}\rangle\right)(s, a)$ can lead to poor Lipschitz upper-bounds $U_{\bar{M}}$.

Improving Lipschitz R-MAX

1) Assuming close models

Maximum model distance Hypothesis

$$D_{\max} \triangleq \max_{s,a,M,\bar{M} \in \mathcal{S}\times\mathcal{A}\times\mathcal{M}^2} \left(D_{\gamma V^*_{\bar{M}}} \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right) (s, a) \right)$$

Improving Lipschitz R-MAX

1) Assuming close models

Maximum model distance Hypothesis

$$D_{\max} \triangleq \max_{s,a,M,\bar{M} \in \mathcal{S} \times \mathcal{A} \times \mathcal{M}^2} \left(D_{\gamma V^*_{\bar{M}}} \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right) (s,a) \right)$$

2) Evaluating the local distances

Theorem 2 Maximum local distance estimation

Introduce the following local model distance estimator:

$$\hat{D}_{\max}(s,a) \triangleq \max_{M,\bar{M} \in \hat{\mathcal{M}}^2} \left(D_{\gamma V^*_{\bar{M}}} \left(\langle R, T \rangle, \langle \bar{R}, \bar{T} \rangle \right) (s,a) \right)$$

After sampling m MDPs, the probability of successful estimation is:

$$Pr(\hat{D}_{\max}(s,a) \geq D_{\max}(s,a)) \geq 1 - 2(1 - p_{\min})^m + (1 - 2p_{\min})^m$$

where $p_{\min} = \min_{M \in \mathcal{M}} Pr(M)$ is a lower bound on the sampling probability of an MDP.

Experiments

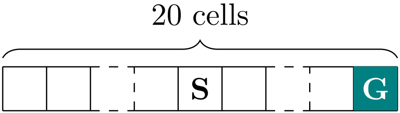
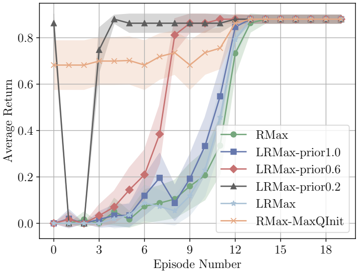
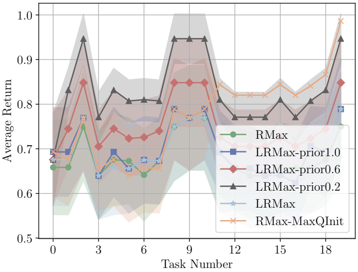


Figure: 1D corridor, reward is sampled in [0.8, 1]





Experiments

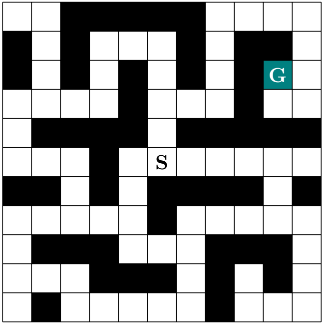
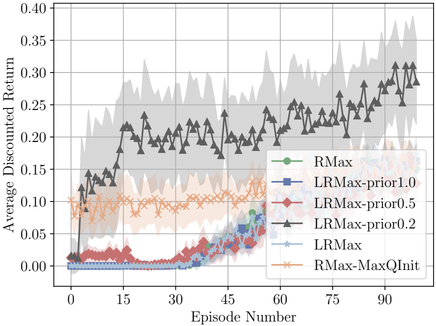


Figure: Maze A), slip probability is sampled in $[0, 0.1]$



Experiments

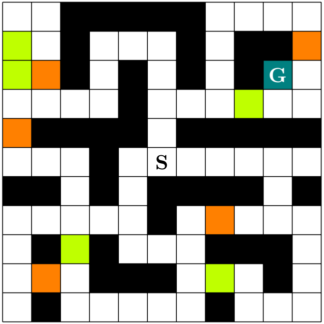
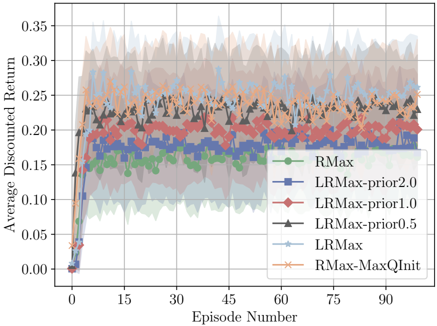


Figure: Maze B), activated walls are either the green or the orange ones.



Conclusion

Contributions

- Theoretical study of the Lipschitz **Continuity of V^* and Q^* in the MDP space**;
- Proposal of a **practical, non-negative, transfer method** based on a local distance between MDPs;
- Proposal and study of a **PAC-MDP algorithm** applying this transfer method in the Lifelong RL setting.