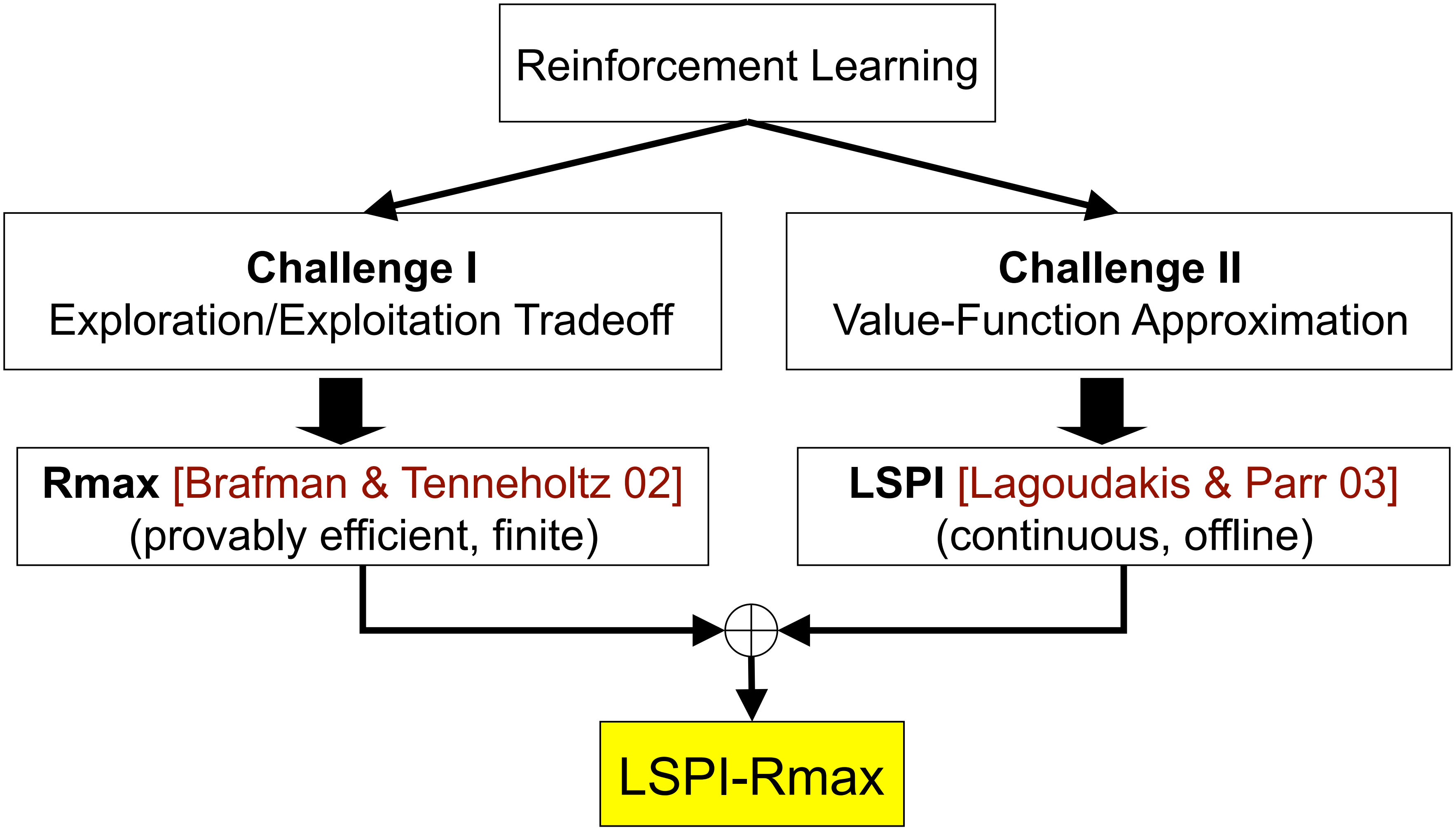
Online Exploration in Least-Squares Policy Iteration

Lihong Li, Michael L. Littman, and Christopher R. Mansley

Rutgers Laboratory for Real-Life Reinforcement Learning (RL^3)

Contributions



Outline

- Introduction
  - LSPI
  - Rmax
- LSPI-Rmax
- Experiments
- Conclusions

Basic Terminology

- Markov decision process
  - States: S
  - Actions: A
  - Reward function: $-1 \leq R(s,a) \leq 1$
  - Transition probabilities: $T(s'|s,a)$
  - Discount factor: $0 < \gamma < 1$
- Optimal value function: $Q^*(s,a)$
- Optimal policy: $\pi^*(s) = \arg\max_a Q^*(s,a)$
- Approximate $Q^*(s,a)$

Linear Function Approximation

$$Q(s,a) = \sum_{i=1}^{k} w_i \phi_i(s,a) = w \cdot \phi(s,a)$$

- Features: $\phi_i(s,a)$
  - A.k.a. "basis functions", and predefined
- Weights: w_i
  - Measures contributions of ϕ_i to approximating Q^*
- Learning = finding w such that: $w \cdot \phi(s,a) \approx Q^*(s,a)$

LSPI [Lagoudakis & Parr 03]

$\pi \leftarrow \pi'$

Initialize π

Evaluate π: compute w

Improve π: $\pi'(s) = \mathrm{argmax}_a \, w \cdot \phi(s,a)$

Given samples: $D = \{(s_1, a_1, r_1, s'_1), \ldots, (s_m, a_m, r_m, s'_m)\}$

Approx. Bellman Eqn.: $w \cdot \phi(s_i, a_i) \approx r_i + \gamma w \cdot \phi(s'_i, \pi(s'_i)), \ \forall i$

LSTDQ sets up a least-squares problem

and computes: $w = \mathbf{A}^{-1}\mathbf{b}$

$$\mathbf{A} = \sum_{i=1}^{m} \phi(s_i, a_i)\big(\phi(s_i, a_i) - \gamma\phi(s'_i, \pi(s'_i))\big)^{\mathrm{T}}, \ \mathbf{b} = \sum_{i=1}^{m} \phi(s_i, a_i) r_i$$

But, LSPI does not specify how to collect samples *D*:
a fundamental challenge in online reinforcement learning

An agent only collects samples in states it visits…

Given samples: $D = \{(s_1, a_1, r_1, s'_1), \ldots, (s_m, a_m, r_m, s'_m)\}$

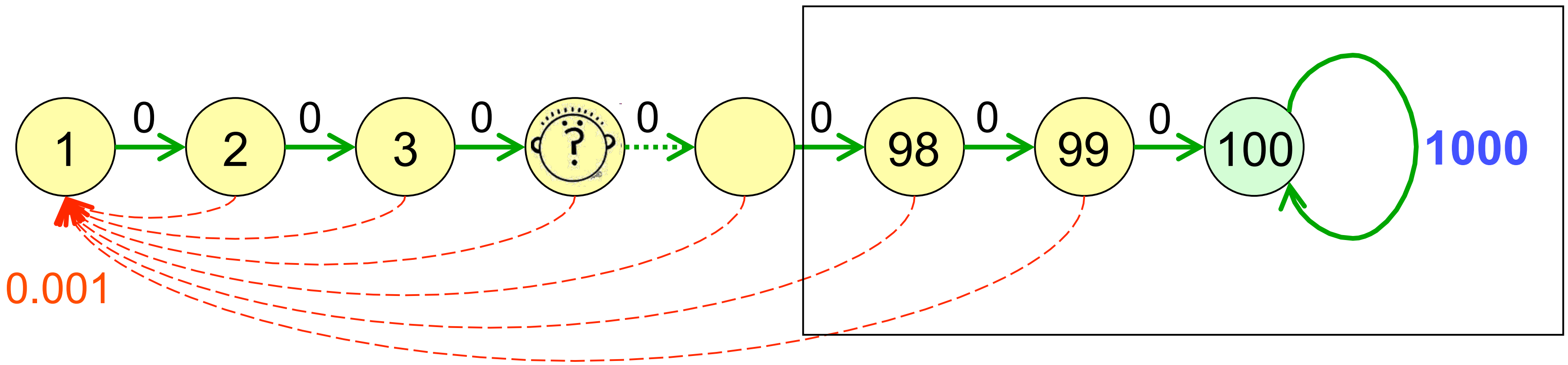
Approx. Bellman Eqn.: $w \cdot \phi(s_i, a_i) \approx r_i + \gamma w \cdot \phi(s'_i, \pi(s'_i)), \ \forall i$

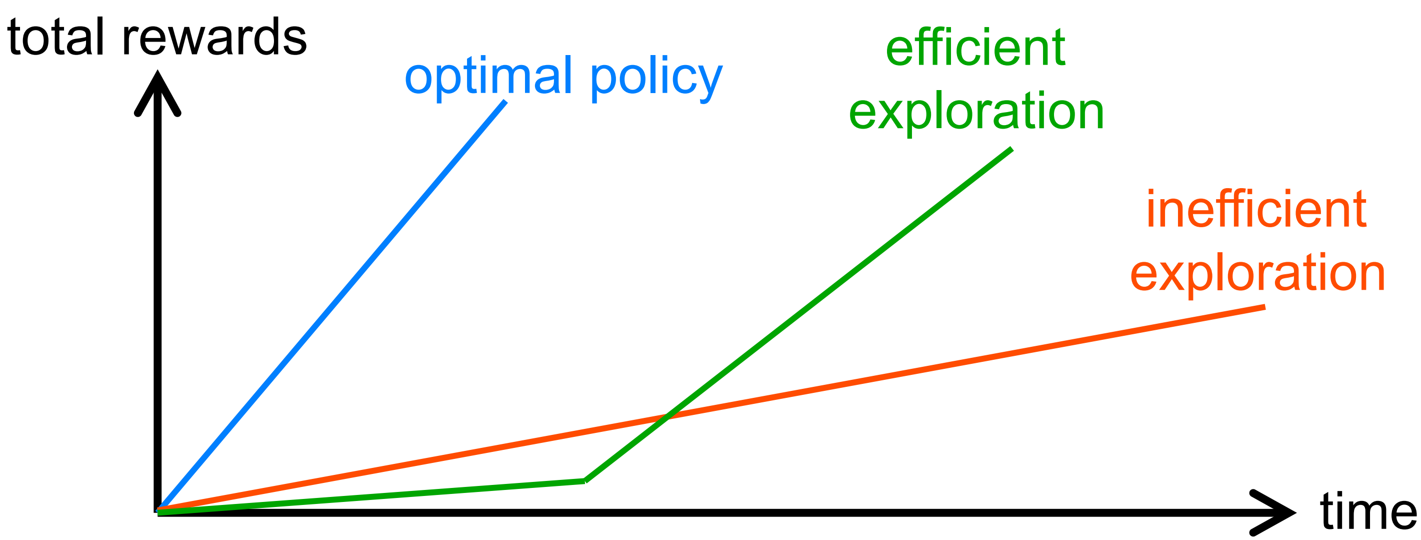
LSTDQ sets up a least-squares problem

and computes: $w = \mathbf{A}^{-1}\mathbf{b}$

$$\mathbf{A} = \sum_{i=1}^{m} \phi(s_i, a_i)\big(\phi(s_i, a_i) - \gamma\phi(s'_i, \pi(s'_i))\big)^{\mathrm{T}}, \ \mathbf{b} = \sum_{i=1}^{m} \phi(s_i, a_i) r_i$$

Exploration/Exploitation Tradeoff

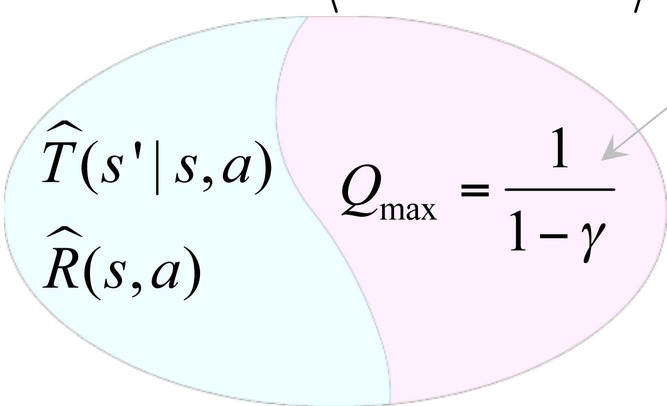




Rmax [Brafman & Tenenholtz 02]

- Rmax is for finite-state, finite-action MDPs
- Learns *T* and *R* by counting/averaging
- In s_t, takes optimal action in $\widehat{M}_{\text{Known}}$

$$\widehat{M}_{\text{Known}} = \left\langle S, A, \widehat{T}, \widehat{R}, \gamma \right\rangle$$



Known state-actions S x A Unknown state-actions

"Optimism in the face of uncertainty"

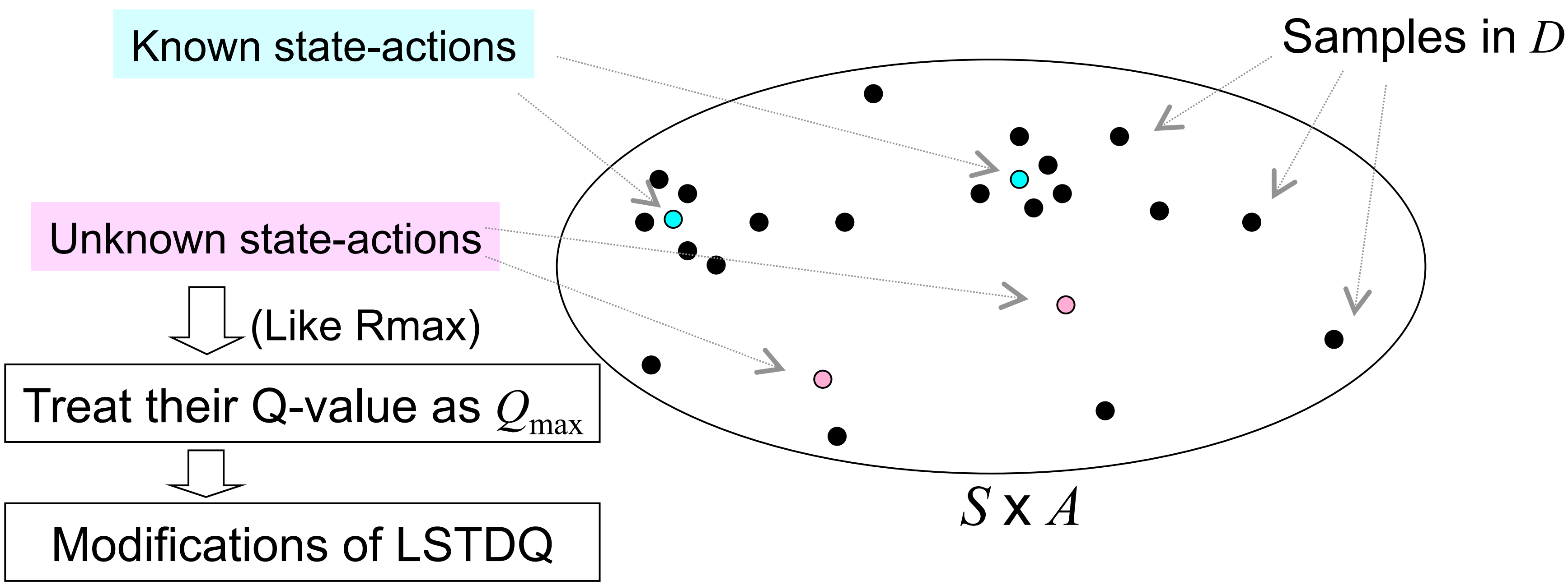
➢**Either:** explore "unknown" region

➢**Or:** exploit "known" region

<u>Thm</u>: Rmax is provably efficient

LSPI-Rmax

- Similar to LSPI
- But distinguishes known/unknown (s,a):



LSTDQ-Rmax

Given samples: $D = \{(s_1, a_1, r_1, s_1'), \ldots, (s_m, a_m, r_m, s_m')\}$

Treat $Q(s,a) = \dfrac{1}{1-\gamma}$ if (s, a) is unknown:

$E.g.,$ if (s_i, a_i) is unknown,

change (s_i, a_i, r_i, s_i') to $(s_i, a_i, Q_{\max}, \square)$ and

$$A = \cdots + \phi(s_i, a_i)\phi(s_i, a_i)^{\mathrm{T}} + \cdots$$

$$b = \cdots + \phi(s_i, a_i) Q_{\max} + \cdots$$

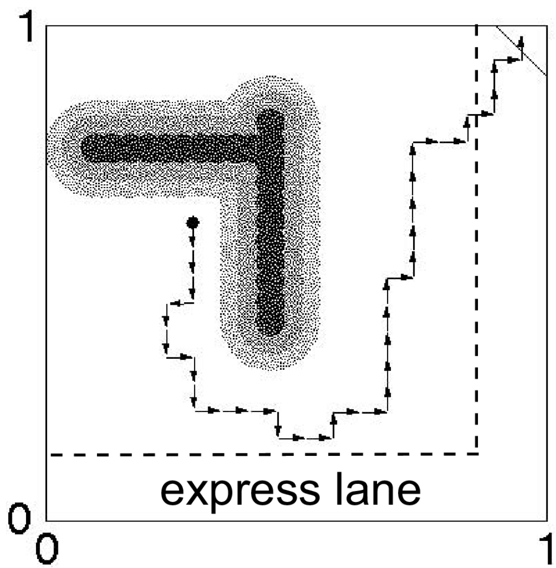
Similarly for (s_i', a).

LSPI-Rmax for Online RL

- D = empty set
- Initialize w
- **for** t = 1, 2, 3, …
  - Take greedy action: a_t = argmax$_a$ $w \cdot \phi(s_t, a)$
  - D = D U $\{(s_t, a_t, r_t, s_{t+1})\}$
  - Run LSPI using LSTDQ-Rmax

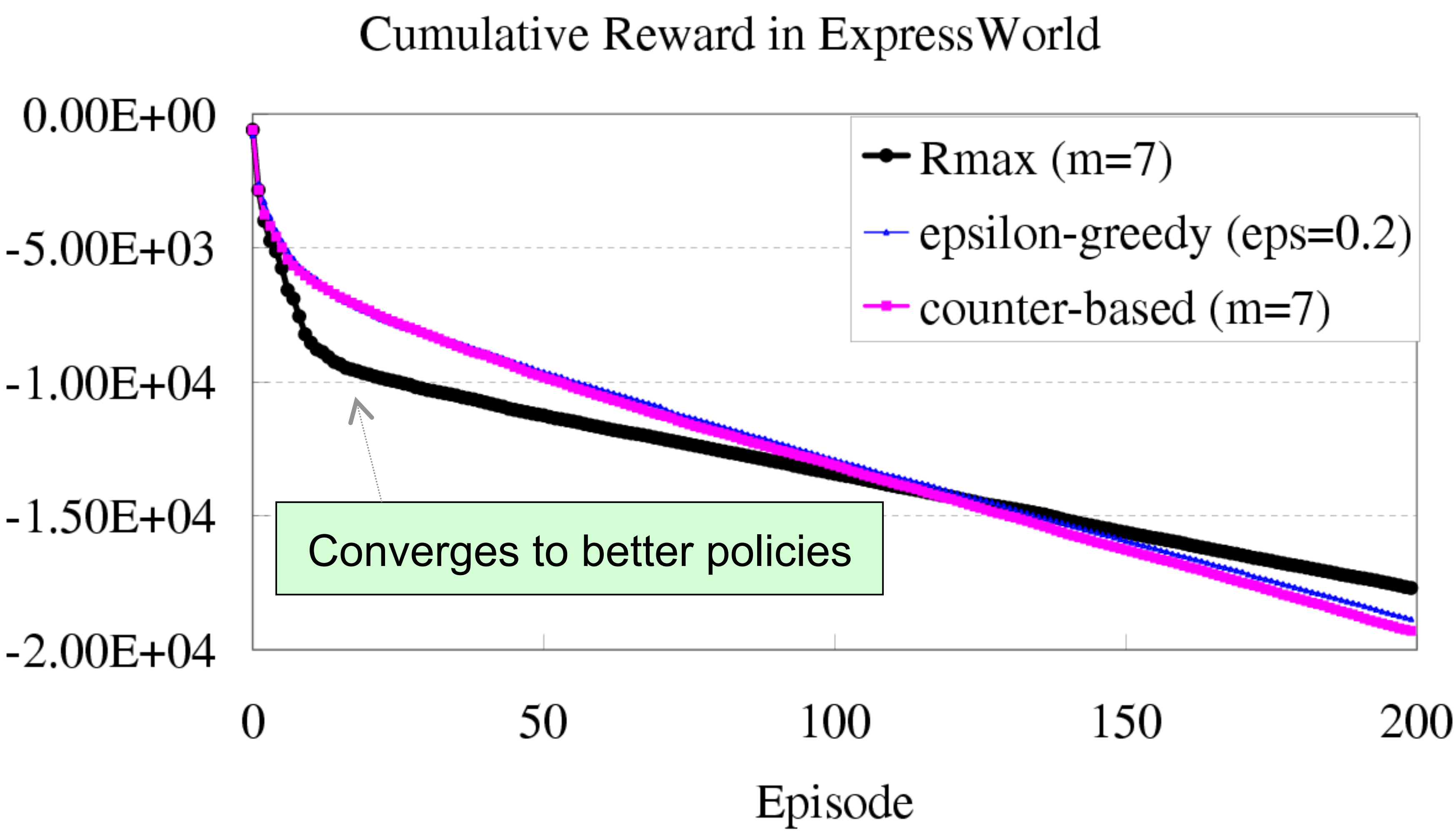
Experiments

- Problems
  - MountainCar
  - Bicycle
  - Continuous Combination Lock
  - **ExpressWorld** (a variant of PuddleWorld)



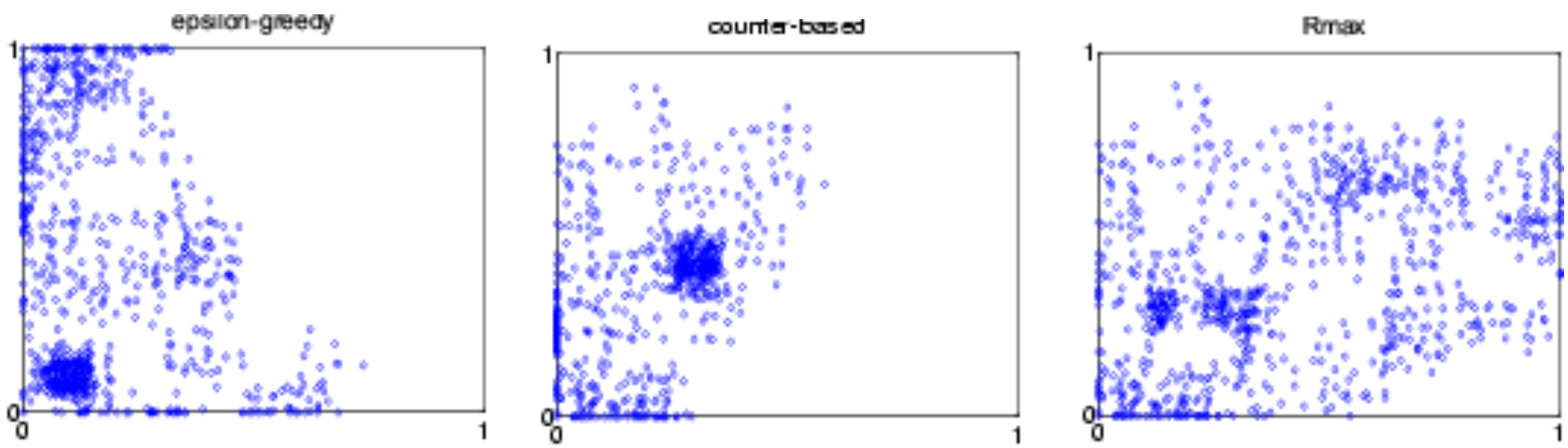
Four actions
Stochastic transitions
Reward:
  -1 reward per step
  -0.5 reward per step in “expresslane”
  penalty for stepping into puddles
Random start states

Various Exploration Rules with LSPI



A Closer Look

States visited in the first 3 episodes:

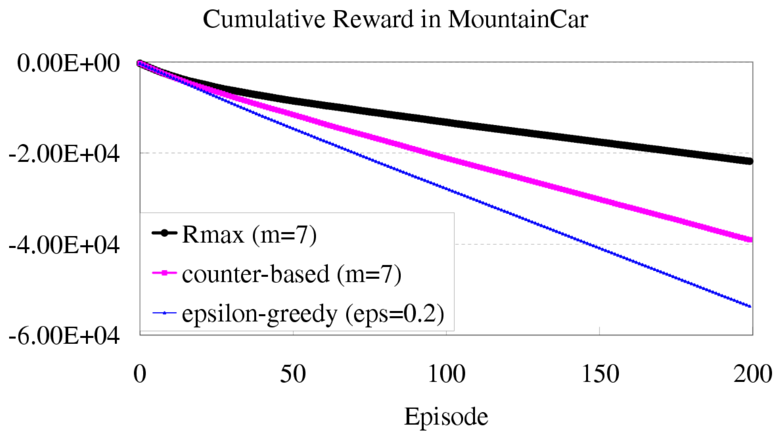


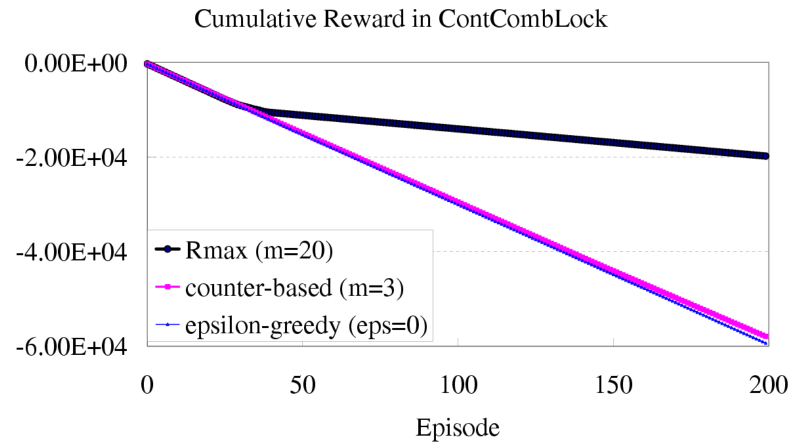
Inefficient exploration

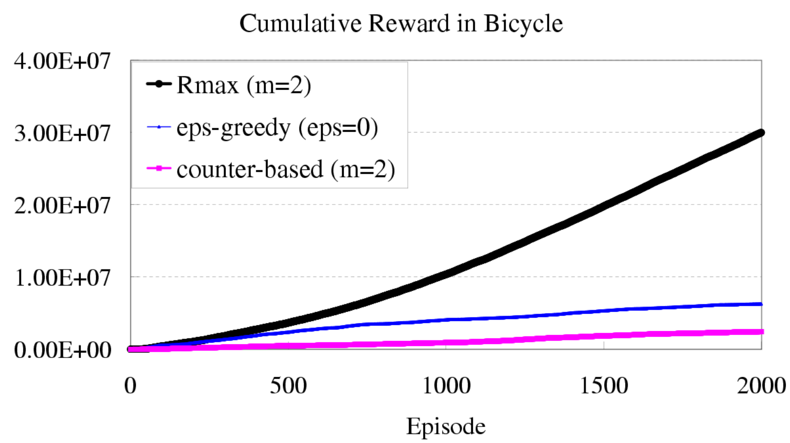
Efficient exploration

Help discovery of goal and expresslane

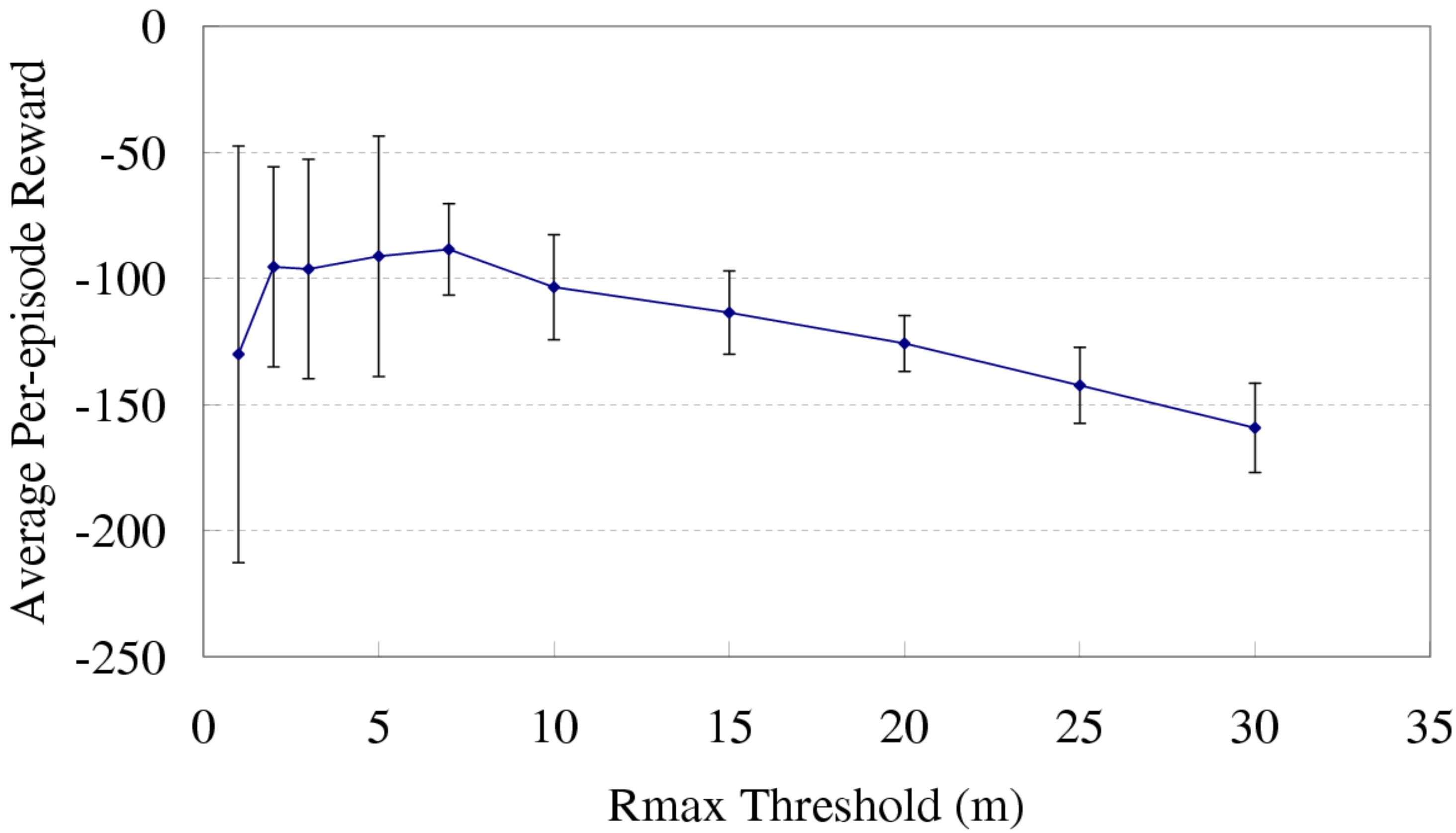
More Experiments







Effect of Rmax Threshold



Conclusions

- We proposed LSPI-Rmax
  - LSPI + Rmax
  - encourages active exploration
  - with linear function approximation
- Future directions
  - Similar idea applied to Gaussian process RL
  - Comparison to model-based RL

Where are features from?

- Hand-crafted features
  - expert knowledge required
  - expensive and error prone
- Generic features
  - RBF, CMAC, polynomial, etc.
  - may not always work well
- Automatic feature selection using
  - Bellman error [Parr et al. 07]
  - spectral graph analysis [Mahadevan & Maggioni 07]
  - TD approximation [Li & Williams & Balakrishnan 09]
  - L_1 Regularization for LSPI [Kolter & Ng 09]

LSPI-Rmax *vs.* MBRL

- Model-based RL (e.g., Rmax)
  - Learns an MDP model
  - Computes policy with the approximate model
  - Can use function approx. in model learning
    - Rmax w/ many compact representations [Li 09]

- LSPI-Rmax is model-free RL
  - Avoids expensive “planning” step
  - Has weaker theoretical guarantees