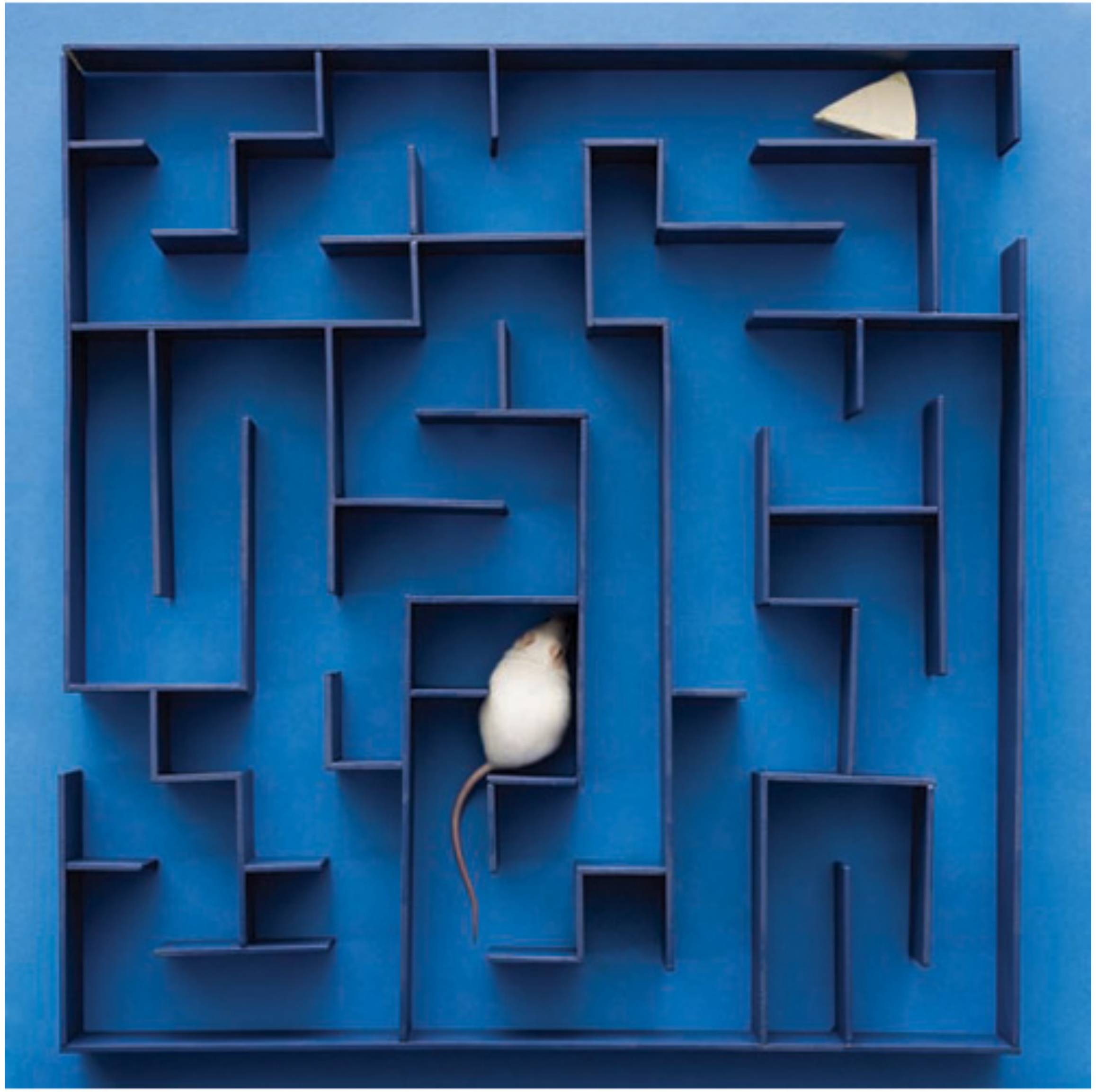
Reinforcement Learning with Rich Observations

Alekh Agarwal
Microsoft Research NYC

Joint work with Nan Jiang, Akshay Krishnamurthy, John Langford and Robert Schapire

What is Reinforcement Learning?

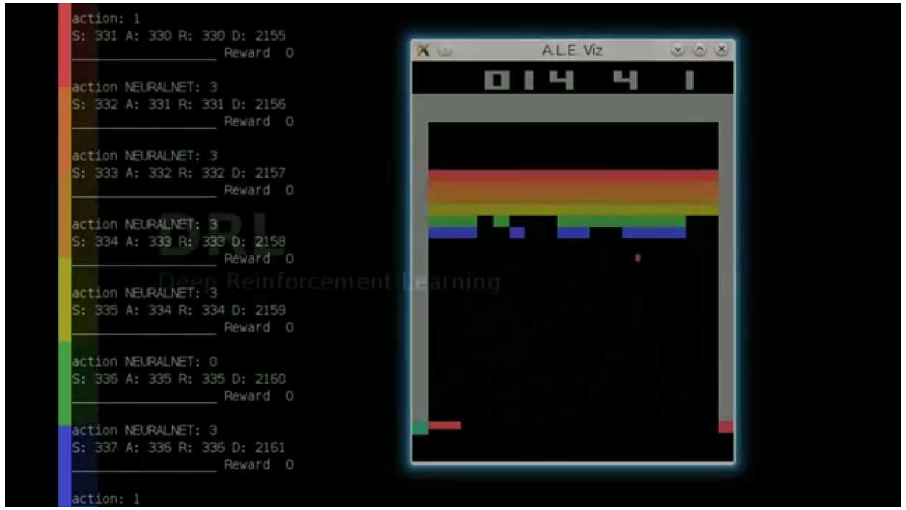


How to Learn?

Practice

Powerful modeling, simple exploration

e.g.: Atari Deep Reinforcement Learning



Theory

Sophisticated exploration in small-state MDPs

e.g. E^3, R-MAX algorithms

Limited theory for rich observations

Goal

Develop Reinforcement Learning approaches guaranteed to learn an optimal policy with a small number of samples despite rich observations.

Our Results

Model	PAC Guarantees
Small-state MDPs	Known
Structured large-state MDPs	New
Reactive POMDPs	Extended
Reactive PSRs	New
LQR (continuous actions)	Known

Our Results

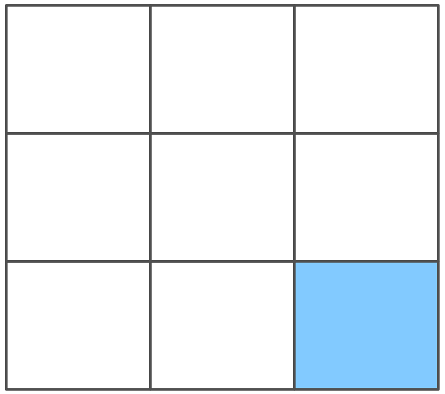
Model	PAC Guarantees
Small-state MDPs	**Known**
Structured large-state MDPs	**New**
Reactive POMDPs	**Extended**
Reactive PSRs	**New**
LQR (continuous actions)	**Known**

Key ideas

- New measure of the hardness of exploration
- Algorithm with sample complexity scaling with this measure
- Applications in several RL settings

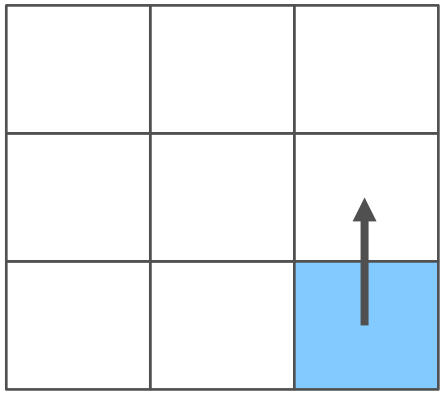
Model

Markov Decision Processes (MDPs)



$$x_1 \sim \Gamma_1$$

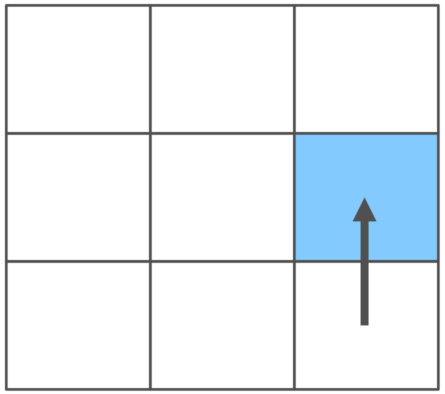
Markov Decision Processes (MDPs)



$x_1 \sim \Gamma_1$

Take action a_1, Observe $r_1(a_1)$

Markov Decision Processes (MDPs)



- Episodic: H actions in a trajectory
- Layered: Distinct states at each level
- Markovian: x_h only depends on (x_{h-1}, a_{h-1}), r_h on (x_h, a_h)

$x_1 \sim \Gamma_1$

Take action a_1, Observe $r_1(a_1)$

New state $x_2 \sim \Gamma(x_1, a_1)$

Goal of Learning

- Maximize long-term reward

$$\sum_{h=1}^{H} r_h(a_h)$$

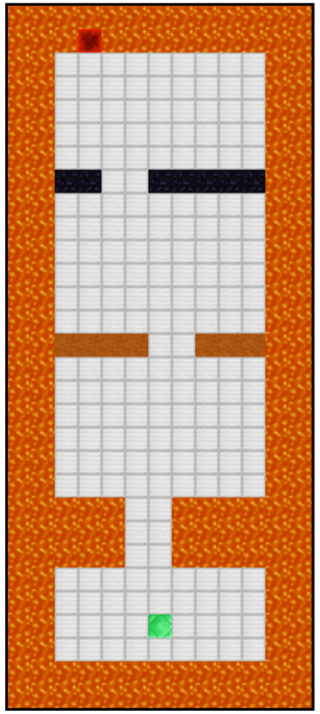
Goal of Learning

- Maximize long-term reward…using policies

$$\sum_{h=1}^{H} r_h(\pi(x_h))$$

- Policies are mappings from states to actions

Example: Navigation in a toy setting



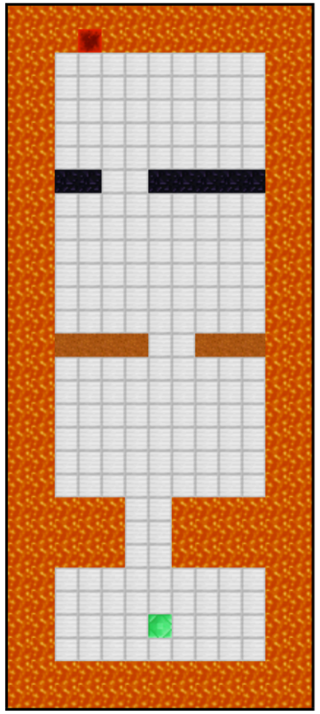
Robotic agent navigating in real-world (left)

States: Position in a grid

Actions: Forward/Back/Left/Right

Reward: 1 on reaching target, -100 for dying

Example: Navigation in a real setting



Robotic agent navigating in real-world (right)

States: Camera view in front of the robot

Actions: Forward/Back/Left/Right

Reward: 1 on reaching target, -100 for dying

Example: Web Search

- User comes with an intent
- Issues a query to the search engine
- Receives ranked list of results
- Issues another query
- …

States: Query, info on user

Actions: Search results

Reward: 1 when user finds a satisfactory result

Existing results

- Learn ϵ-optimal policy using $poly\left(|X|, A, H, \frac{1}{\epsilon}\right)$ samples
- Small number of states necessary for learning

Lower bound

There is an MDP with $|A|^H$ states where finding an ϵ-optimal policy requires $\Omega\left(\frac{|A|^H}{\epsilon^2}\right)$ trajectories.

Intuition: Embed a bandit problem with $|A|^H$ arms.

Compact $\mathcal{F}$ not sufficient for generalization in RL

Gathering the right data has large sample complexity

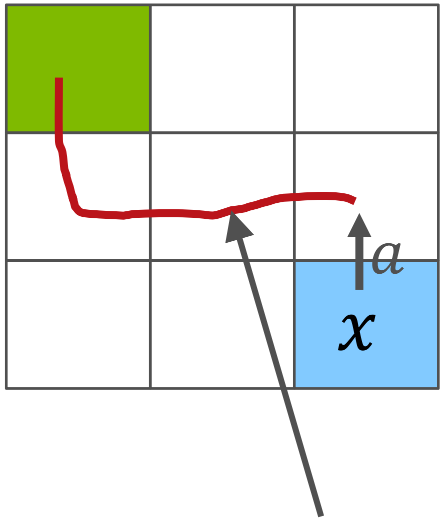
Large-state MDPs

- Too many “unique” states in real-world tasks
- Cannot reason separately for each state
- Need information sharing between similar states
  - aka generalization
- Typically done via value-function approximation

Function Approximation

Optimal value function

- Optimal value function Q^*
  - Maps (x, a) pair to a long-term reward
  - Take action a in state x and follow the optimal policy thereafter
  - Removes the need to reason over multiple decisions



$Q^*(x, a)$ = length of shortest path after taking a in x

- Optimal policy $\pi^*(x) = \text{argmax}_a Q^*(x, a)$

Value of a policy

- Value: Long-term reward on following a policy

$$V(\pi) = \mathrm{E}_{x \sim \Gamma_1}[V(x, \pi)], \text{ where}$$

Distribution of initial state

$$V(x, \pi) = \mathrm{E}_{r \sim D_x}\left[r(\pi(x)) + \mathrm{E}_{x' \sim \Gamma(x, \pi(x))} V(x', \pi)\right]$$

Distribution of next state

Instantaneous reward

Optimal value function

- Optimal value function: Best reward from each state

$$V^\star = \mathrm{E}_{x\sim\Gamma_1}[V^\star(x)], \text{ where}$$

Distribution of initial state

$$V^\star(x) = \max_a \mathrm{E}_{r\sim D_x}\left[r(a) + \mathrm{E}_{x'\sim\Gamma(x,a)} V^\star(x')\right]$$

Optimal action

Instantaneous reward

Distribution of next state

Optimal value function

- Optimal value function: Best reward from each state

$$V^\star = \mathrm{E}_{s\sim\Gamma_1}[V^\star(x)], \text{ where}$$

$$V^\star(x) = \max_a \underbrace{\mathrm{E}_{r\sim D_x}\left[r(a) + \mathrm{E}_{x'\sim\Gamma(x,a)} V^\star(x')\right]}_{Q^\star(x,a)}$$

Optimal policy: $\pi^\star(x) = \operatorname{argmax}_a Q^\star(x,a)$

Function approximation

Given a class $\mathcal{F} : X \times A \to \mathrm{R}$, find a good approximation to $Q^\star$, assuming $Q^\star \in \mathcal{F}$

Associated greedy policy: $\pi_f = \mathrm{argmax}_a\, f(x, a)$

- Key intuition: Use a class $\mathcal{F}$ that generalizes well in supervised learning
  - Consider $\mathcal{F}$ of small VC-dimension/Rademacher complexity/finite size…

A solution sketch

- Start with an initial guess $f_1 \in \mathcal{F}$ for $Q^\star$
- Act according to f_1, collect trajectories

  $(x_1, a_1, r_1, \ldots, x_H, a_H, r_H)$ where $a_h = \pi_{f_1}(x_h)$
- Use the trajectories to obtain a better estimate $f_2 \in \mathcal{F}$
- Repeat

A solution sketch

- Start with an initial guess $f_1 \in \mathcal{F}$ for $Q^\star$
- Act according to f_1, collect trajectories

  $(x_1, a_1, r_1, \ldots, x_H, a_H, r_H)$ where $a_h = \pi_{f_1}(x_h)$
- Use the trajectories to obtain a better estimate $\boldsymbol{f_2} \in \mathcal{F}$
- Repeat

Our Setting

Bellman Equations

$$Q^\star(x,a) = \mathrm{E}_{r\sim D_x}\left[r(a) + \mathrm{E}_{x'\sim\Gamma(x,a\,)} V^\star(x')\right]$$

- Take a at current step, act optimally thereafter

Bellman Equations

$$Q^\star(x,a) = \mathrm{E}_{r\sim D_x}\left[r(a) + \mathrm{E}_{x'\sim\Gamma(x,a\,)} \max_{a'} Q^\star(x',a')\right]$$

- Take a at current step, act optimally thereafter

Bellman Equations

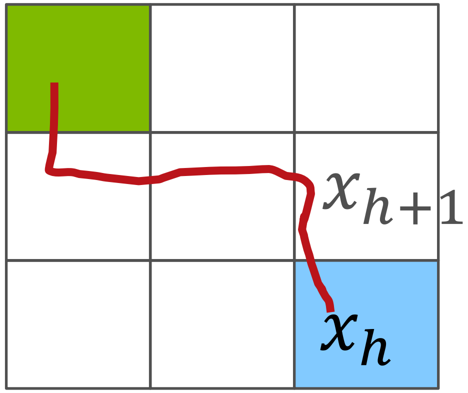
$$Q^\star(x,a) = \mathrm{E}_{r\sim D_x}\left[r(a) + \mathrm{E}_{x'\sim\Gamma(x,a)} Q^\star(x', \pi^\star(x'))\right]$$

- Take a at current step, act optimally thereafter
- Holds for each x, hence any distribution over x

Bellman Equations

$$\varepsilon(f, \pi, h) = \mathrm{E}[f(x_h, a_h) - r_h \ - f(x_{h+1}, a_{h+1})],$$

where $a_1, \ldots a_{h-1} \sim \pi$ and a_h, a_{h+1} according to π_f



Bellman Equations

$$\varepsilon(f,\pi,h) = \mathrm{E}[f(x_h,a_h) - r_h \; - f(x_{h+1},a_{h+1})],$$

where $a_1, \ldots a_{h-1} \sim \pi$ and a_h, a_{h+1} according to π_f

Standard result: $\varepsilon(Q^\star,\pi,h) = 0$, for all π

Gives a test for checking if $f \approx Q^\star$

Using Bellman Equations

- Given candidate $f \in \mathcal{F}$, check $\varepsilon(f, \pi, h)$ for all π, h
- Reject f if $\varepsilon(f, \pi, h) \gg 0$ for any π, h
- Restrict to $\pi = \pi_g$ for $g \in \mathcal{F}$

Validity condition

Challenge: Computing $\boldsymbol{\varepsilon(f, \pi, h)}$ requires samples from $\boldsymbol{\pi}$

For all $\boldsymbol{\pi_g}$ requires $\boldsymbol{O}(|\mathcal{F}|)$ samples!

Key challenges

- Too many functions in any interesting $\mathcal{F}$
- Data based on one f might not prove sub-optimality for others
- Need to collect the right data
- Like bandits, but with exponentially many arms!

Bellman factorization and rank

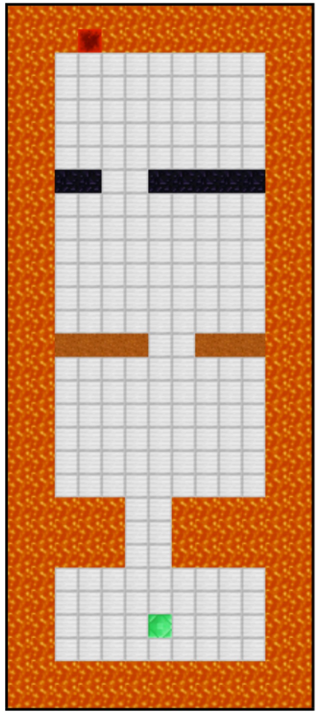
- Consider the $|\mathcal{F}| \times |\mathcal{F}|$ matrix:

$$\varepsilon(\mathcal{F}, h)_{f,g} = \varepsilon(f, \pi_g, h)$$

Bellman rank of an MDP is the rank of $\boldsymbol{\varepsilon}(\mathcal{F}, \boldsymbol{h})$

- Bounded by number of states
- Bounded by rank of transition matrix $\boldsymbol{\Gamma}$
- Bounded by number of “hidden” states

Example: Navigation in a real setting



Robotic agent navigating in real-world (right)

States: Camera view in front of the robot

Transitions determined by grid-view (left)

Bellman rank bounded by size of grid!

Example: Web Search

- User comes with an intent
- Issues a query to the search engine
- Receives ranked list of results
- Issues another query
- …

States: Query, info on user

Transitions often depend on user intent

Bellman rank bounded by number of possible intents (topics)

Summary so far

- Given function approximators $f \in \mathcal{F}$
- Want to find an f such that
  - f is valid
  - f yields a good policy, that is $V(\pi_f)$ is large
- Algorithm intuition: Low Bellman rank gives concise basis for checking validity (exploration)
- Challenge: We do not know the basis, just its existence

Our Algorithm

Reminder…

- $Q^\star$ is valid for state distribution under any policy π
- $Q^\star$ captures the optimal value:

$$
\begin{aligned}
V(\pi^\star) &= \mathrm{E}_{x\sim\Gamma_1} \max_a [Q^\star(x,a)] \\
&= \mathrm{E}_{x\sim\Gamma_1} Q^\star(x, \pi^\star(x))
\end{aligned}
$$

Optimism Led Iterative Value-function Elimination (OLIVE)

- $\mathcal{F}_0 = \mathcal{F}$
- For t=1,2,…
  - Choose f_t to maximize $\hat{V} = \mathbf{E}_{\boldsymbol{x}\sim\Gamma_1}[\boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{\pi}_{\boldsymbol{f}}(\boldsymbol{x})\right)]$

    Optimism under uncertainty, guess for $V(\pi^\star)$ if $f = Q^\star$

  - Collect trajectories using $\pi_t = \pi_{f_t}$
  - If $V(\pi_t) \geq \hat{V} - \epsilon$
    - Return π_t

    Checking our optimistic belief

  - Reject all f with large $\varepsilon(f, \pi_t, h)$ for any h
  - Set $\mathcal{F}_t$ to be the set of surviving f

    Prune the possible solutions

PAC Guarantee

Suppose $Q^\star \in \mathcal{F}$, and the Bellman rank is at most M. OLIVE returns a policy π satisfying $V(\pi) \geq V(\pi^\star) - \epsilon$ and with probability $1 - \delta$, the number of trajectories needed is at most

$$O\left(\frac{M^2 H^3 |A|^2 \log(|\mathcal{F}|/\delta)}{\epsilon^2}\right)$$

Implications

- Retains sample-efficiency for small-state MDPs
  - albeit better results exist here
- New results for several settings
  - Low-rank MDPs
  - Reactive POMDPs
  - Reactive PSRs
  - …
- Unifying treatment for sample-efficient RL

Proof intuition (correctness)

Algorithm always retains $Q^\star$, terminates when:

$$\boldsymbol{V(\pi_t) \geq \widehat{V} - \epsilon \geq V^\star - \epsilon}$$

- Either f found in first step is near-optimal
- Or, we will reject it
- Shows correctness

Proof intuition (sample efficiency)

- Bellman error matrix has low rank
- Each elimination step decreases rank by 1 if we check for $\varepsilon(f, \pi, h) = 0$
- Extension to noisy checking: Ellipsoidal argument
  - Reduce the volume of “Bellman error vectors” by constant fraction each time

Extensions

- Do not require $Q^\star \in \mathcal{F}$
  - Find the valid f with largest $V(\pi_f)$
- Adapt to the knowledge of M
- Allow errors in Bellman factorization and validity
- Allow infinite classes $\mathcal{F}$ with low VC-like dimensions

Wrapping up

- New structural condition for efficient exploration in RL
- First sample-complexity results in a broad setup called Contextual Decision Processes
  - Unifying treatment for several RL models
- Algorithm robust to modeling assumptions
- Key open problem: Computational efficiency

Thank You!

Details at: https://arxiv.org/abs/1610.09512