Lecture 13: Fast RL Part III

Emma Brunskill

CS234 Reinforcement Learning

Spring 2024

With a few slides from David Silver.
The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right figure).

Select all that are true.

1. Sample 3 params: 0.1, 0.5, 0.3. These are more likely to come from the Beta(1,2) distribution than Beta(1,1).
2. Sample 3 params: 0.2, 0.5, 0.8. These are more likely to come from the Beta(1,1) distribution than Beta(1,2).
3. It is impossible that the true Bernoulli parameter is 0 if the prior is Beta(1,1).
4. Not sure

The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right). The true parameters are arm 1 $\theta_1 = 0.4$ & arm 2 $\theta_2 = 0.6$. Thompson sampling = TS

1. TS could sample $\theta = 0.5$ (arm 1) and $\theta = 0.55$ (arm 2).
2. For the sampled thetas (0.5,0.55) TS is optimistic with respect to the true arm parameters for all arms.
3. For the sampled thetas (0.5,0.55) TS will choose the true optimal arm for this round.
4. Not sure
The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right). Select all that are true.

1. Sample 3 params: 0.1, 0.5, 0.3. These are more likely to come from the Beta(1,2) distribution than Beta(1,1).
2. Sample 3 params: 0.2, 0.5, 0.8. These are more likely to come from the Beta(1,1) distribution than Beta(1,2).
3. It is impossible that the true Bernoulli parameter is 0 if the prior is Beta(1,1).
4. Not sure

The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right). The true parameters are arm 1 $\theta_1 = 0.4$ & arm 2 $\theta_2 = 0.6$. Thompson sampling = TS

1. TS could sample $\theta = 0.5$ (arm 1) and $\theta = 0.55$ (arm 2).
2. For the sampled thetas (0.5, 0.55) TS is optimistic with respect to the true arm parameters for all arms.
3. For the sampled thetas (0.5, 0.55) TS will choose the true optimal arm for this round.
4. Not sure
Last time: Fast Learning (Bayesian bandits to MDPs)

This time: Fast Learning III (MDPs)

Next time: Monte Carlo Tree Search
Over these 3 lectures will consider 2 settings, multiple frameworks, and approaches

Settings: Bandits (single decisions), MDPs

Frameworks: evaluation criteria for formally assessing the quality of a RL algorithm. So far seen empirical evaluations, asymptotic convergence, regret, probably approximately correct

Approaches: Classes of algorithms for achieving particular evaluation criteria in a certain set. So far for exploration seen: greedy, $\epsilon-$greedy, optimism, Thompson sampling, for multi-armed bandits

Goal: fast, efficient RL for large, complex domains.
Table of Contents

1. MDPs
2. Bayesian MDPs
3. Generalization and Exploration
4. Summary
5. Exploration for Multi-Task RL
Very similar set of frameworks and approaches are relevant for fast learning in reinforcement learning

**Frameworks**
- Regret
- Bayesian regret
- Probably approximately correct (PAC)

**Approaches**
- Optimism under uncertainty
- Probability matching / Thompson sampling

**Framework:** Probably approximately correct
Model-Based Interval Estimation with Exploration Bonus (MBIE-EB)

(Strehl and Littman, J of Computer & Sciences 2008)

1: Given $\epsilon$, $\delta$, $m$

2: $\beta = \frac{1}{1-\gamma} \sqrt{0.5 \ln(2 |S||A|m/\delta)}$

3: $n_{sas}(s, a, s') = 0, \forall s \in S, a \in A, s' \in S$

4: $rc(s, a) = 0, n_{sa}(s, a) = 0, \tilde{Q}(s, a) = 1/(1 - \gamma), \forall s \in S, a \in A$

5: $t = 0, s_t = s_{init}$

6: loop

7: $a_t = \arg \max_{a \in A} \tilde{Q}(s_t, a)$

8: Observe reward $r_t$ and state $s_{t+1}$

9: $n_{sa}(s_t, a_t) = n_{sa}(s_t, a_t) + 1, n_{sas}(s_t, a_t, s_{t+1}) = n_{sas}(s_t, a_t, s_{t+1}) + 1$

10: $rc(s_t, a_t) = \frac{rc(s_t, a_t)(n_{sa}(s_t, a_t) - 1) + r_t}{n_{sa}(s_t, a_t)}$

11: $\hat{R}(s_t, a_t) = rc(s_t, a_t)$ and $\hat{T}(s'|s_t, a_t) = \frac{n_{sas}(s_t, a_t, s')}{{n}_{sa}(s_t, a_t)}, \forall s' \in S$

12: while not converged do

13: $\tilde{Q}(s, a) = \hat{R}(s, a) + \gamma \sum_{s'} \hat{T}(s'|s, a) \max_{a'} \tilde{Q}(s', a) + \frac{\beta}{\sqrt{n_{sa}(s, a)}}, \forall s \in S, a \in A$

14: end while

15: end loop
For a given $\epsilon$ and $\delta$, a RL algorithm $A$ is PAC if on all but $N$ steps, the action selected by algorithm $A$ on time step $t$, $a_t$, is $\epsilon$-close to the optimal action, where $N$ is a polynomial function of $(|S|, |A|, \frac{1}{1-\gamma}, \frac{1}{\epsilon}, \frac{1}{\delta})$.

Is this true for all algorithms?
Theorem 2. Suppose that $\epsilon$ and $\delta$ are two real numbers between 0 and 1 and $M = \langle S, A, T, R, \gamma \rangle$ is any MDP. There exists an input $m = m(\frac{1}{\epsilon}, \frac{1}{\delta})$, satisfying $m(\frac{1}{\epsilon}, \frac{1}{\delta}) = O(\frac{|S|}{\epsilon^2(1-\gamma)^4} + \frac{1}{\epsilon^2(1-\gamma)^4} \ln \frac{|S||A|}{\epsilon(1-\gamma)^\delta})$, and $\beta = (1/(1-\gamma)) \sqrt{\ln(2|S||A|m/\delta)}/2$ such that if MBIE-EB is executed on MDP $M$, then the following holds. Let $A_t$ denote MBIE-EB’s policy at time $t$ and $s_t$ denote the state at time $t$. With probability at least $1 - \delta$, $V_{M}^{A_t}(s_t) \geq V_{M}^*(s_t) - \epsilon$ is true for all but $O(\frac{|S||A|}{\epsilon^3(1-\gamma)^6} (|S| + \ln \frac{|S||A|}{\epsilon(1-\gamma)^\delta}) \ln \frac{1}{\delta} \ln \frac{1}{\epsilon(1-\gamma)})$ timesteps $t$. 
One of the key ideas: Simulation Lemma

- Bound error in value function due to error in dynamics & reward models

---

1 Covered in problem sessions: https://web.stanford.edu/class/cs234/sessions/CS234_Win23_ProblemSession2.pdf
**Bayesian bandits** exploit prior knowledge of rewards, $p[\mathcal{R}]$

- They compute posterior distribution of rewards $p[\mathcal{R} | h_t]$, where $h_t = (a_1, r_1, \ldots, a_{t-1}, r_{t-1})$

- Use posterior to guide exploration
  - Upper confidence bounds (Bayesian UCB)
  - Probability matching (Thompson Sampling)

- Better performance if prior knowledge is accurate
Consider a bandit problem where the reward of an arm is a binary outcome \( \{0, 1\} \) sampled from a Bernoulli with parameter \( \theta \).

- E.g. Advertisement click through rate, patient treatment succeeds/fails, ...

The Beta distribution \( \text{Beta}(\alpha, \beta) \) is conjugate for the Bernoulli distribution

\[
p(\theta | \alpha, \beta) = \theta^{\alpha-1}(1 - \theta)^{\beta-1} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}
\]

where \( \Gamma(x) \) is the Gamma function.

- Assume the prior over \( \theta \) is a \( \text{Beta}(\alpha, \beta) \) as above

- Then after observed a reward \( r \in \{0, 1\} \) then updated posterior over \( \theta \) is \( \text{Beta}(r + \alpha, 1 - r + \beta) \)
Thompson Sampling for Bandits

1: Initialize prior over each arm \( a \), \( p(\mathcal{R}_a) \)
2: \textbf{loop}
3: For each arm \( a \) \textbf{sample} a reward distribution \( \mathcal{R}_a \) from posterior
4: Compute action-value function \( Q(a) = \mathbb{E}[\mathcal{R}_a] \)
5: \( a_t = \arg \max_{a \in A} Q(a) \)
6: Observe reward \( r \)
7: Update posterior \( p(\mathcal{R}_a | r) \) using Bayes law
8: \textbf{end loop}
Bayesian Model-Based RL

- Maintain posterior distribution over MDP models
- Estimate both transition and rewards, \( p[\mathcal{P}, \mathcal{R} \mid h_t] \), where \( h_t = (s_1, a_1, r_1, \ldots, s_t) \) is the history
- Use posterior to guide exploration
  - Upper confidence bounds (Bayesian UCB)
  - Probability matching (Thompson sampling)
Thompson Sampling: Model-Based RL

- Thompson sampling implements probability matching

\[
\pi(s, a \mid h_t) = \mathbb{P}[ Q(s, a) \geq Q(s, a'), \forall a' \neq a \mid h_t ]
\]

\[
= \mathbb{E}_{\mathcal{P}, \mathcal{R} \mid h_t} \left[ 1(a = \arg \max_{a \in \mathcal{A}} Q(s, a)) \right]
\]

- Use Bayes law to compute posterior distribution \( p[\mathcal{P}, \mathcal{R} \mid h_t] \)
- **Sample** an MDP \( \mathcal{P}, \mathcal{R} \) from posterior
- Solve MDP using favorite planning algorithm to get \( Q^*(s, a) \)
- Select optimal action for sample MDP, \( a_t = \arg \max_{a \in \mathcal{A}} Q^*(s_t, a) \)
Posterior Sampling for Reinforcement Learning (PSRL). Osband, Russo, Van Roy (NeurIPS 2013)

1: Initialize prior over dynamics and reward models for each \((s, a), p(R_{as}), p(T(s'|s, a))\)
2: Initialize state \(s_0\)
3: for \(k \in 1:K\), number of episodes do
4: Sample a MDP \(M\):
5: for each \((s, a)\) pair do
6: Sample a dynamics model \(T(s'|s, a)\)
7: Sample a reward model \(R(s, a)\)
8: end for
9: Compute \(Q^*_M\), optimal value for MDP \(M\)
10: for \(t \in 1:H\) do
11: \(a_t = \arg \max_{a \in A} Q^*_M(s_t, a)\)
12: Observe reward \(r_t\) and next state \(s_{t+1}\)
13: end for
14: Update posterior \(p(R_{as_t}|r_t), p(T(s'|s_t, a_t)|s_{t+1})\) using Bayes rule
15: end for
Check Your Understanding: Fast RL III

- Strategic exploration in MDPs (select all):
  1. Doesn’t really matter because the distribution of data is independent of the policy followed
  2. Can involve using optimism with respect to both the possible dynamics and reward models in order to compute an optimistic Q function
  3. Is known as PAC if the number of time steps on which a less than near optimal decision is made is guaranteed to be less than an exponential function of the problem domain parameters (state space cardinality, etc).
  4. Not sure

- In Thompson sampling for tabular MDPs in the shown algorithm:
  1. TS samples the reward model parameters and could use the empirical average for the dynamics model parameters and obtain the same performance
  2. Can perform MDP planning everytime the posterior is updated
  3. Always has the same computational cost each step as Q-learning
  4. Not sure
Strategic exploration in MDPs (select all):
1. Doesn’t really matter because the distribution of data is independent of the policy followed
2. Can involve using optimism with respect to both the possible dynamics and reward models in order to compute an optimistic Q function
3. Is known as PAC if the number of time steps on which a less than near optimal decision is made is guaranteed to be less than an exponential function of the problem domain parameters (state space cardinality, etc).
4. Not sure

In Thompson sampling for tabular MDPs in the shown algorithm:
1. TS samples the reward model parameters and could use the empirical average for the dynamics model parameters and obtain the same performance.
2. Can perform MDP planning everytime the posterior is updated.
3. Always has the same computational cost each step as Q-learning.
4. Not sure
1: Initialize prior over dynamics and reward models for each (s, a), p(R_{as}), p(T(s'|s, a))
2: Initialize state s_0
3: for k ∈ 1:K, number of episodes do
   4: Sample a MDP M:
      5: for each (s, a) pair do
         6: Sample a dynamics model T(s'|s, a)
         7: Sample a reward model R(s, a)
      8: end for
   9: Compute Q^*_M, optimal value for MDP M
10: for t ∈ 1:H do
    11: a_t = arg max_{a ∈ A} Q^*_M(s_t, a)
    12: Observe reward r_t and next state s_{t+1}
13: end for
14: Update posterior p(R_{a_t s_t}|r_t), p(T(s'|s_t, a_t)|s_{t+1}) using Bayes rule
15: end for

https://www.youtube.com/watch?v=xjGK-wm0PkI
Table of Contents

1. MDPs
2. Bayesian MDPs
3. Generalization and Exploration
4. Summary
5. Exploration for Multi-Task RL
Active area of ongoing research: combine generalization & strategic exploration

Many approaches are grounded by principles outlined here
  - Optimism under uncertainty
  - Thompson sampling
Generalization and Strategic Exploration

- Active area of ongoing research: combine generalization & strategic exploration
- Many approaches are grounded by principles outlined here
  - Optimism under uncertainty
  - Thompson sampling
- These issues are important for large state spaces and large action spaces, in bandits and Markov decision processes
- Rest of today: brief discussion of contextual bandits, then MDPs
Multi-armed bandit is a tuple of \((\mathcal{A}, \mathcal{R})\), where \(\mathcal{A}\) : known set of \(m\) actions (arms)

- \(\mathcal{R}^a(r) = \mathbb{P}[r | a]\) is an unknown probability distribution over rewards
- At each step \(t\) the agent selects an action \(a_t \in \mathcal{A}\)
- The environment generates a reward \(r_t \sim \mathcal{R}^{a_t}\)
- Goal: Maximize cumulative reward \(\sum_{\tau=1}^{t} r_{\tau}\) / minimize total regret

Contextual bandits: context/state space \(\mathcal{S}\) and action space \(\mathcal{A}\)

- \(\mathcal{R}^{a,s}(r) = \mathbb{P}[r | a, s]\) is an unknown probability distribution over rewards, for a particular state and action
- If the state and/or action space is very large, it is common to use a function to represent the relationship between the input state and action and the output rewards
Benefits of Generalization: Bandits vs Contextual Multiarmed Bandits:

- $k$ is the number of arms, y-axis is the regret. [Figure is Figure 19.1, Lattimore and Szepesvari, Bandit Algorithms]
Contextual Multiarmed Bandits

- Contextual bandits: context/state space $S$ and action space $A$
- $\mathcal{R}^{a,s}(r) = \mathbb{P}[r \mid a, s]$ is an unknown probability distribution over rewards, for a particular state and action
- If the state and/or action space is very large, it is common to use a function to represent the relationship between the input state and action and the output rewards
- Common to model reward as a linear function of input features $\phi(s, a)$
- $r = \theta \phi(s, a) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$
Disjoint Linear Contextual Multi-armed Bandits

- Assumes that each arm $a$ has its own $\theta_a$ parameter
- $r(s, a) = \theta_a \phi(s) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$

Check your understanding: can $r = \theta \phi(s, a) + \epsilon$ represent a disjoint linear model?
$r = \theta \phi(s, a) + \epsilon$

Previously we used Hoeffding’s inequality to represent uncertainty over a scalar reward.

We would like to now represent uncertainty over $r$ through uncertainty over $\theta$ (check your understanding: why is this sufficient to capture uncertainty over $r$?)

Requires us to compute an uncertainty set over a vector $\theta$.

This can be done in a computationally tractable way, see e.g. A Contextual-Bandit Approach to Personalized News Article Recommendation, WWW 2010 or Chapter 19 in Lattimore and Szepesvari.)
Active area of ongoing research: combine generalization & strategic exploration

Many approaches are grounded by principles outlined here
  - Optimism under uncertainty
  - Thompson sampling

These issues are important for large state spaces and large action spaces, in bandits and Markov decision processes

Rest of today: brief discussion of contextual bandits, then MDPs
Generalization and Optimism

- Recall MBIE-EB algorithm for finite state and action domains
- What needs to be modified for continuous / extremely large state and/or action spaces?
Model-Based Interval Estimation with Exploration Bonus (MBIE-EB)

(Strehl and Littman, J of Computer & Sciences 2008)

1: Given $\epsilon, \delta, m$
2: $\beta = \frac{1}{1-\gamma} \sqrt{0.5 \ln(2 |S||A|m/\delta)}$
3: $n_{sas}(s, a, s') = 0, \forall s \in S, a \in A, s' \in S$
4: $rc(s, a) = 0, n_{sa}(s, a) = 0, \tilde{Q}(s, a) = 1/(1 - \gamma), \forall s \in S, a \in A$
5: $t = 0, s_t = s_{init}$
6: loop
7: $a_t = \arg \max_{a \in A} \tilde{Q}(s_t, a)$
8: Observe reward $r_t$ and state $s_{t+1}$
9: $n_{sa}(s_t, a_t) = n_{sa}(s_t, a_t) + 1, n_{sas}(s_t, a_t, s_{t+1}) = n_{sas}(s_t, a_t, s_{t+1}) + 1$
10: $rc(s_t, a_t) = \frac{rc(s_t, a_t)(n_{sa}(s_t, a_t) - 1) + r_t}{n_{sa}(s_t, a_t)}$
11: $\hat{R}(s_t, a_t) = rc(s_t, a_t)$ and $\hat{T}(s'|s_t, a_t) = \frac{n_{sas}(s_t, a_t, s')}{n_{sa}(s_t, a_t)}, \forall s' \in S$
12: while not converged do
13: $\tilde{Q}(s, a) = \hat{R}(s, a) + \gamma \sum_{s'} \hat{T}(s'|s, a) \max_{a'} \tilde{Q}(s', a) + \frac{\beta}{\sqrt{n_{sa}(s, a)}}, \forall s \in S, a \in A$
14: end while
15: end loop
Recall MBIE-EB algorithm for finite state and action domains

What needs to be modified for continuous / extremely large state and/or action spaces?

Estimating uncertainty
  - Counts of \((s,a)\) and \((s,a,s')\) tuples are not useful if we expect only to encounter any state once
Recall: Value Function Approximation with Control

- For Q-learning use a TD target $r + \gamma \max_{a'} \hat{Q}(s', a'; w)$ which leverages the max of the current function approximation value

$$\Delta w = \alpha (r(s) + \gamma \max_{a'} \hat{Q}(s', a'; w) - \hat{Q}(s, a; w)) \nabla_w \hat{Q}(s, a; w)$$

- Modify to:

$$\Delta w = \alpha (r(s) + r_{bonus}(s, a) + \gamma \max_{a'} \hat{Q}(s', a'; w) - \hat{Q}(s, a; w)) \nabla_w \hat{Q}(s, a; w)$$
Recall: Value Function Approximation with Control

- For Q-learning use a TD target \( r + \gamma \max_{a'} \hat{Q}(s', a'; w) \) which leverages the max of the current function approximation value

\[
\Delta w = \alpha \left( r(s) + r_{\text{bonus}}(s, a) + \gamma \max_{a'} \hat{Q}(s', a'; w) - \hat{Q}(s, a; w) \right) \nabla_w \hat{Q}(s, a; w)
\]

- \( r_{\text{bonus}}(s, a) \) should reflect uncertainty about future reward from \((s, a)\)

- Approaches for deep RL that make an estimate of visits / density of visits include: Bellemare et al. NIPS 2016; Ostrovski et al. ICML 2017; Tang et al. NIPS 2017

- Note: bonus terms are computed at time of visit. During episodic replay can become outdated.
Benefits of Strategic Exploration: Montezuma’s revenge

Figure: Bellemare et al. "Unifying Count-Based Exploration and Intrinsic Motivation"

- https://www.youtube.com/watch?v=ToSe_CUG0F4
- Enormously better than standard DQN with $\epsilon$-greedy approach
Leveraging Bayesian perspective has also inspired some approaches.

One approach: Thompson sampling over representation & parameters (Mandel, Liu, Brunskill, Popovic IJCAI 2016)
Generalization and Strategic Exploration: Thompson Sampling

- For scaling up to very large domains, again useful to consider model-free approaches
- Non-trivial: would like to be able to sample from a posterior over possible $Q^*$
- Bootstrapped DQN (Osband et al. NIPS 2016)
  - Train $C$ DQN agents using bootstrapped samples
  - When acting, choose action with highest $Q$ value over any of the $C$ agents
  - Some performance gain, not as effective as reward bonus approaches
Leveraging Bayesian perspective has also inspired some approaches

One approach: Thompson sampling over representation & parameters (Mandel, Liu, Brunskill, Popovic IJCAI 2016)

For scaling up to very large domains, again useful to consider model-free approaches

Non-trivial: would like to be able to sample from a posterior over possible $Q^*$

Bootstrapped DQN (Osband et al. NIPS 2016)

Efficient Exploration through Bayesian Deep Q-Networks (Azizzadenesheli, Anandkumar, NeurIPS workshop 2017)

- Use deep neural network
- On last layer use Bayesian linear regression
- Be optimistic with respect to the resulting posterior
- Very simple, empirically much better than just doing linear regression on last layer or bootstrapped DQN, not as good as reward bonuses in some cases
Ultimately often want agents that can learn and before across many tasks.

Can we have agents that learn to explore?

DREAM (Liu et al. NeurIPS 2022) was one example

Decision Pretrained Transformer (Lee, Xie, Pacchiano, Chandak, Finn, Nachum and Brunskill NeurIPS 2023) is another
Decision-Pretrained Transformer for Meta RL

- Key idea: Training to predict $a^*$ mimics Thompson Sampling but can capture a much richer set of priors

Lee, Xie et al. NeurIPS 2023
Can Learn and Leverage (Unknown) Task Structure To Significantly Accelerate Exploration

- Key idea: Training to predict $a^*$ mimics Thompson Sampling but can capture a much richer set of priors

Lee, Xie et al. NeurIPS 2023
1 MDPs

2 Bayesian MDPs

3 Generalization and Exploration

4 Summary

5 Exploration for Multi-Task RL
Summary: What You Are Expected to Know

- Define the tension of exploration and exploitation in RL and why this does not arise in supervised or unsupervised learning
- Be able to define and compare different criteria for "good" performance (empirical, convergence, asymptotic, regret, PAC)
- Be able to map algorithms discussed in detail in class to the performance criteria they satisfy
- Understand the UCB proof sketch
- For those of you doing default project: be able to implement UCB and TS for linear contextual bandit. See e.g. A Contextual-Bandit Approach to Personalized News Article Recommendation, WWW 2010 or Chapter 19 in Lattimore and Szepesvari)
Class Structure

- Last time: Fast Learning (Bayesian bandits to MDPs)
- **This time: Fast Learning III (MDPs)**
- Next time: Monte Carlo Tree Search
Theoretical Results

- Discussed regret bounds for bandits, & PAC bounds for tabular MDPs
- Now exist tight (in dominant term) minimax results for regret and PAC for tabular MDPs
  - Azar, Mohammad Gheshlaghi, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. ICML 2017 (regret)
  - Dann, C., Li, L., Wei, W., and Brunskill, E. Policy certificates: Towards accountable reinforcement learning. ICML 2019 (PAC)
- Also exist instance-dependence bounds for tabular MDPs, e.g.:
  - Zanette and Brunskill. Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds. ICML 2019
Do there exist strong theoretical bounds for RL with function approximation?

Active area of recent work

- Many others, including our work (lead by Andrea Zanette), and Mengdi Wang’s lab.

Active area: quantifying features of the domain that correspond to hardness

Eluder dimension (Russo and Van Roy), Bellman rank (Jiang et al), ..
Table of Contents

1. MDPs
2. Bayesian MDPs
3. Generalization and Exploration
4. Summary
5. Exploration for Multi-Task RL
Exploration Across Tasks

- **DREAM**
- Active area of recent work
  - Many others, including our work (lead by Andrea Zanette), and Mengdi Wang’s lab.
- Active area: quantifying features of the domain that correspond to hardness
- Eluder dimension (Russo and Van Roy), Bellman rank (Jiang et al), ..