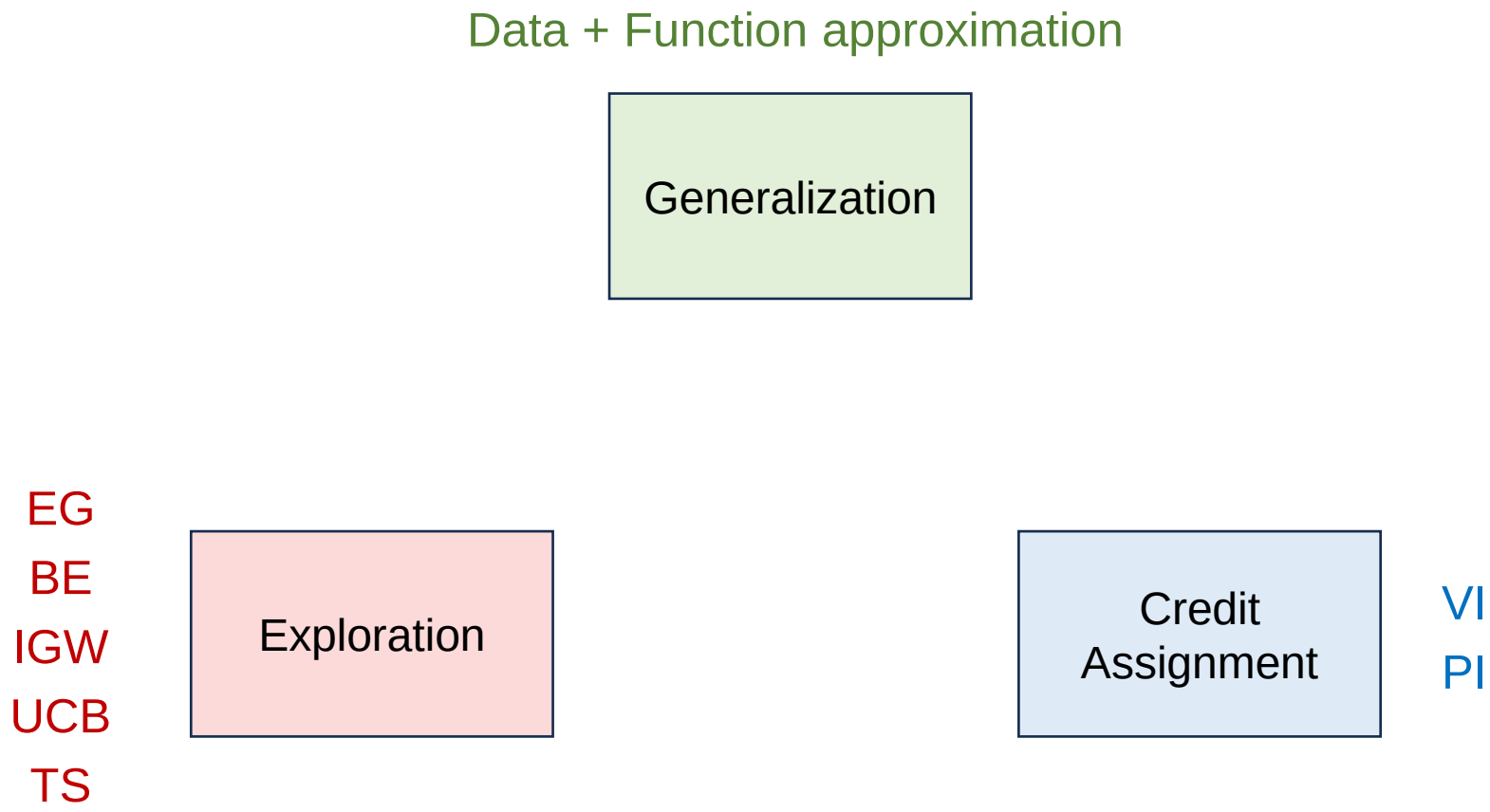
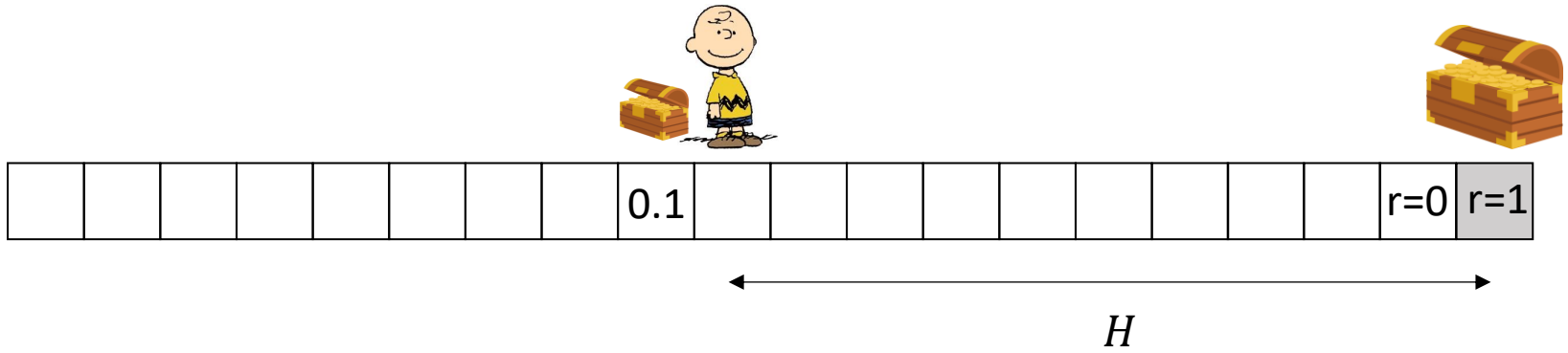
Exploration in MDPs

Chen-Yu Wei

We have addressed all 3 main challenges in online RL



We have addressed all 3 main challenges in online RL (?)



Environment:

- Fixed-horizon MDP with episode length H
- Initial state at 0
- A single rewarding state at state H
- Actions: Go LEFT or RIGHT

Suppose we perform DQN with ϵ-greedy with random initialization
⇒ On average, we need 2^H episodes to see the reward
(before that, we won't make any meaningful update and will just do random walk around state 0)

Regret Analysis for MDPs?

- We have done regret analysis for several bandit algorithms:
  - Regression oracle + (ϵ-greedy or inverse gap weighting)
  - UCB
  - EXP3

- We did not really establish regret bounds for MDPs
  - Partially – DQN under 2 assumptions: the data in replay buffer is exploratory and Bellman completeness
  - Not for policy-iteration-based algorithms

Regret Analysis for MDPs?

$$\mathbb{E}_{s\sim\rho}\left[V^{\pi^\star}(s)\right] - \mathbb{E}_{s\sim\rho}[V^{\pi}(s)]$$

1

$$= \sum_{s,a} d_\rho^\pi(s)\left(\pi^\star(a|s) - \pi(a|s)\right)Q^\star(s,a) = \sum_{s,a} d_\rho^\pi(s,a)\left(V^\star(s) - Q^\star(s,a)\right)$$

For VI-based algorithm (approximating $Q^\star$)

Approximating $Q^\star(s,a)$ requires the replay buffer to cover **wide range of** state-actions.

2

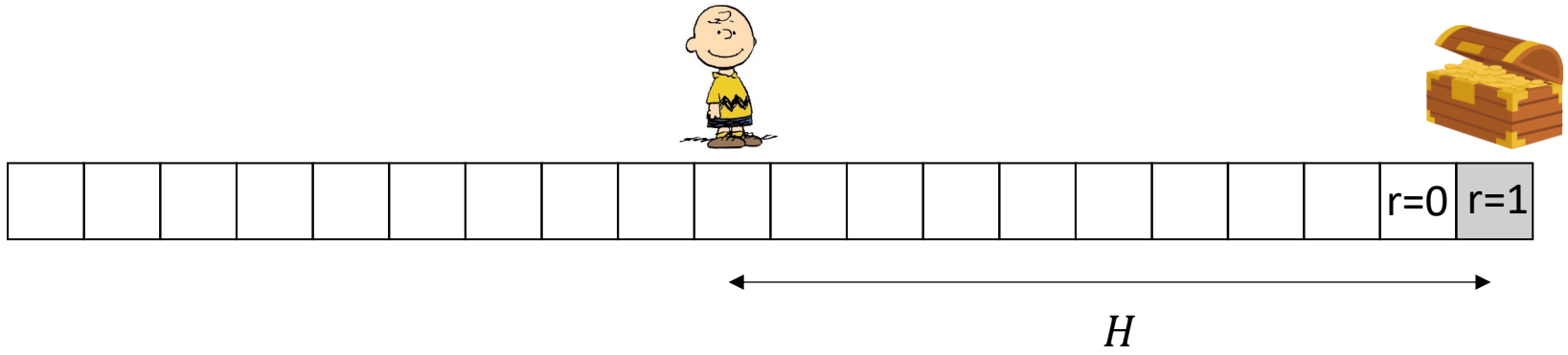
$$= \sum_{s,a} d_\rho^{\pi^\star}(s)\left(\pi^\star(a|s) - \pi(a|s)\right)Q^\pi(s,a) = \sum_{s,a} d_\rho^{\pi^\star}(s,a)\left(Q^\pi(s,a) - V^\pi(s)\right)$$

For PI-based algorithm (approximating Q^π)

Approximating $Q^\pi(s,a)$ only requires state-actions generated from current policy

But…

Regret Analysis for MDPs?



$$\sum_{s,a} d_\rho^\pi(s,a)\,(V^\star(s) - Q^\star(s,a))$$

$$\sum_{s,a} d_\rho^{\pi^\star}(s,a)\,(Q^\pi(s,a) - V^\pi(s))$$

large

$\exists s,a.$ $d_\rho^{\pi^\star}(s,a)$ large $Q^{\pi^\star}(s,a) - V^{\pi^\star}(s,a)$ large

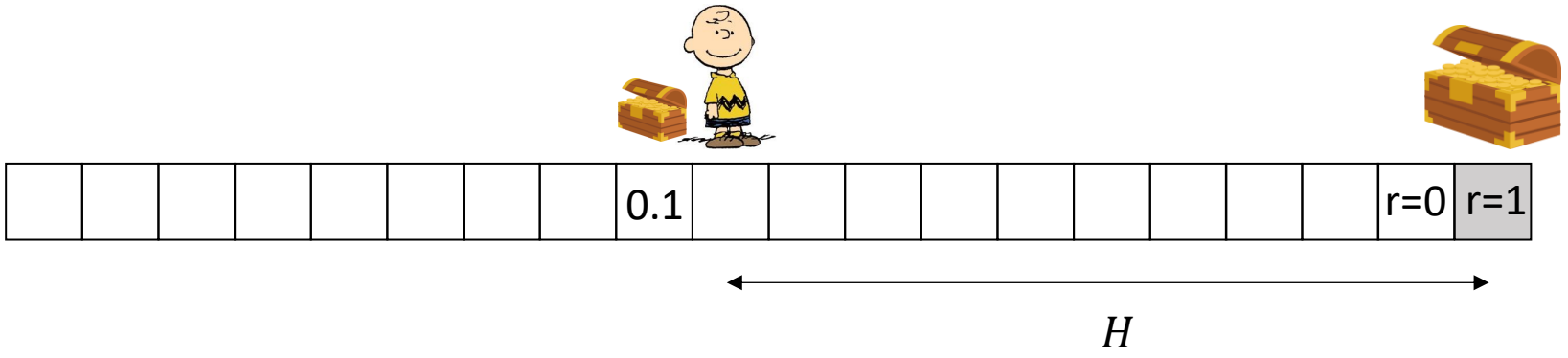
$d_\rho^{\pi_k}(s,a)$

PI-based algorithm only tries to make small $\sum_{s,a} d_\rho^{\pi_k}(s,a)\,(Q^\pi(s,a) - V^\pi(s))$ small.

It can only quickly find optimal policy when $d_\rho^{\pi_k} \approx d_\rho^{\pi^\star}$

Insufficiency of algorithms we have discussed for MDPs

- Lack of **exploration over the state space** (we need **deep exploration**)
- This issue is particularly critical if
  - Local reward does not provide any information (Sparse reward)
  - Local reward provide misleading information



- Solution
  - Try to make the data (i.e., state-action) distribution close to $d^{\pi^\star}$
  - Try to visit as many states as possible

Exploration Bonus (Optimism Principle)

- We have discussed this idea for action exploration – UCB.

Upper Confidence Bound

$$a_t = \text{argmax}_a \quad \hat{R}_t(a) + \sqrt{\frac{2\log(2/\delta)}{N_t(a)}}$$

$\hat{R}_t(a)$ = the empirical mean of arm a up to time $t-1$.
$N_t(a)$ = the number of times we draw arm a up to time $t-1$.

Exploration Bonus (Optimism Principle)

$$a_t = \text{argmax}_a \quad \hat{R}_t(a) + \sqrt{\frac{2\log(2/\delta)}{N_t(a)}} \overset{\gamma}{=} \tilde{R}_t(a)$$

$$\sum_t \sqrt{\frac{1}{N_t(a_t)}} \leq \sqrt{AT}$$

$$\text{Regret} = \sum_t \left(R(a^*) - R(a_t)\right)$$

$$= \sum_t \left(\tilde{R}(a^*) - \tilde{R}(a_t)\right) + \sum_t \left(R(a^*) - \tilde{R}(a^*)\right) + \sum_t \left(\tilde{R}(a_t) - R(a_t)\right)$$

bonus

① $\tilde{R}_t(a) + b_t(a) \geq R(a)$

≤ 0 $\qquad$ ≤ 0

$\leq \sqrt{\frac{1}{N_t(a_t)}}$

② $\sum_t b_t(a_t) \leq \text{sub-linear}(T)$

Exploration Bonus for MDPs

UCB Value Iteration (UCBVI) (finite state - actn)

For episode $1, 2, \ldots, T$:

$\tilde{Q}_{H+1}(s,a) = 0 \quad \forall s, a$

For step $H, H-1, \ldots, 1$:

$$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + H\sqrt{\frac{2\log(2/\delta)}{N_t(s,a)}} \quad \forall s, a$$

Receive $s_1 \sim \rho$

For step $1, 2, \ldots, H$:

Take action $a_h = \text{argmax}_a \tilde{Q}_h(s_h, a)$

Receive $r_h = R(s_h, a_h) + \text{noise}$, $s_{h+1} \sim P(\cdot|s_h, a_h)$

Exploration Bonus for MDPs

$$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + H\sqrt{\frac{2\log(2/\delta)}{N_t(s,a)}} \quad \forall s,a$$

$\tilde{Q}_h(s,a) \geq Q_h^*(s,a)$ $\forall s,a,h$ w.h.p.

$\tilde{Q}_H(s,a) = \hat{R}(s,a) + H\sqrt{\frac{2\log(1/\delta)}{N_t(s,a)}} \geq Q_H^*(s,a) = R(s,a)$

for $h<H$

$\tilde{Q}_h(s,a) = \hat{R}(s,a) + \sum_{s'} \hat{p}(s'|s,a)\, \tilde{V}_{h+1}(s') \geq \hat{R}(s,a) + \sum_{s'} \hat{p}(s'|s,a)\, V_{h+1}^*(s') + H\sqrt{\frac{2\log(1/\delta)}{N_t(s,a)}}$

$+ H\sqrt{\frac{2\log(1/\delta)}{N_t(s,a)}} \geq R(s,a) + \sum_{s'} p(s'|s,a)\, V_{h+1}^{\pi}(s') = Q_h^*(s,a)$

Exploration Bonus for MDPs

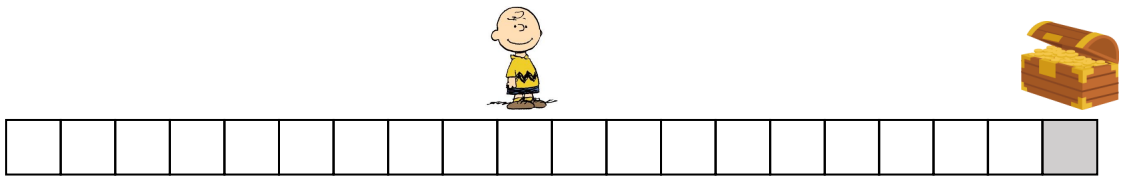
Theorem. Regret Bound of UCBVI

UCBVI ensures with high probability,

$$\text{Regret} = \sum_{t=1}^{T} \left(V^{\star}(s_{t,1}) - V^{\pi_t}(s_{t,1}) \right) \lesssim H\sqrt{SAT}.$$

$$\frac{1}{T}\sum_{t=1}^{T}\left(V^{*}(s_{t,1}) - V^{\pi_t}(s_{t,1})\right) \lesssim \sqrt{\frac{H^2SA}{T}} \le \varepsilon$$

$$\Rightarrow T \gtrsim \frac{H^2SA}{\varepsilon^2}$$



Improving the required number of episodes from 2^H to $\text{poly}(H)$

Jaksch, Ortner, Auer. Near-Optimal Regret Bounds for Reinforcement Learning. 2010.

Azar, Osband, Munos. Minimax Regret Bounds for Reinforcement Learning. 2017.

Thompson Sampling
(Posterior Sampling)

$H_t = (a_1, r_1, a_2, r_2, \cdots, a_{t-1}, r_{t-1})$

Bayesian interpretation:
Assume the reward mean $(\theta(1), \ldots, \theta(A))$ is drawn from a Gaussian distribution (prior distribution).
Then the **posterior distribution** is

$$P(\theta(a)|\mathcal{H}_t) = \mathcal{N}\left(\hat{R}_t(a), \frac{1}{N_t(a)}\right)$$

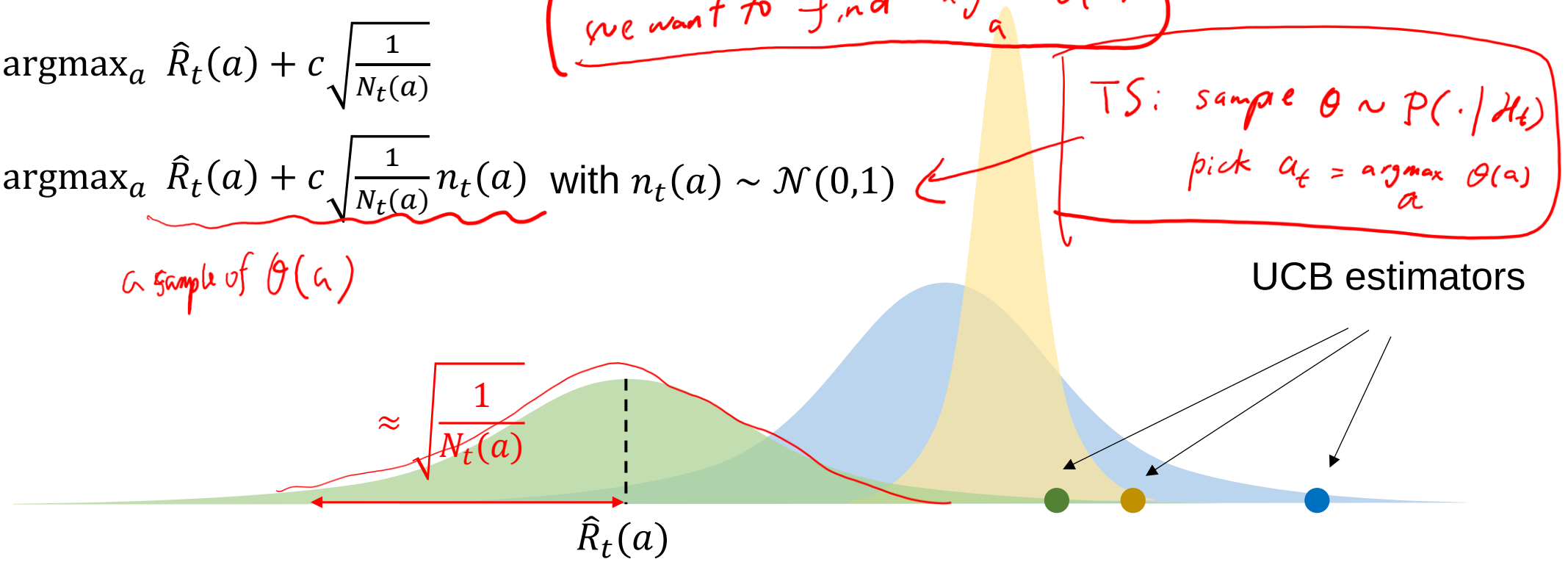
we want to find $\operatorname{argmax}_a \theta(a)$

UCB: $a_t \approx \text{argmax}_a \ \hat{R}_t(a) + c\sqrt{\frac{1}{N_t(a)}}$

TS: $a_t \approx \text{argmax}_a \ \hat{R}_t(a) + c\sqrt{\frac{1}{N_t(a)}}\, n_t(a)$ with $n_t(a) \sim \mathcal{N}(0,1)$

a sample of $\theta(a)$

TS: sample $\theta \sim P(\cdot|H_t)$
pick $a_t = \operatorname{argmax}_a \theta(a)$



Randomized Exploration for MDPs

Randomized Value Iteration

For episode $1, 2, \ldots, T$:

$\tilde{Q}_{H+1}(s,a) = 0 \quad \forall s, a$

For step $H, H-1, \ldots, 1$:

$$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + H \sqrt{\frac{2\log(2/\delta)}{N_t(s,a)}} \underbrace{n_t(s,a)}_{\sim \mathcal{N}(0,1)}$$

Receive $s_1 \sim \rho$

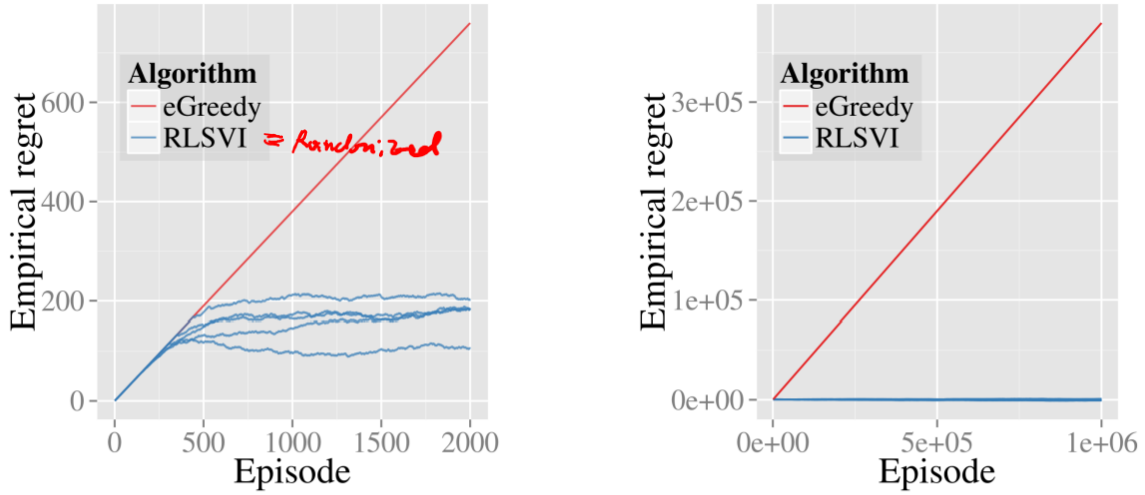
For step $1, 2, \ldots, H$:

Take action $a_h = \text{argmax}_a \, \tilde{Q}_h(s_h, a)$

Receive $r_h = R(s_h, a_h) + \text{noise}, \quad s_{h+1} \sim P(\cdot\,|s_h, a_h)$

Osband, Van Roy, Wen. Generalization and Exploration via Randomized Value Functions. 2014.

Randomized Exploration for MDPs



(a) First 2000 episodes (b) First 10^6 episodes

Figure 2. Efficient exploration on a 50-chain

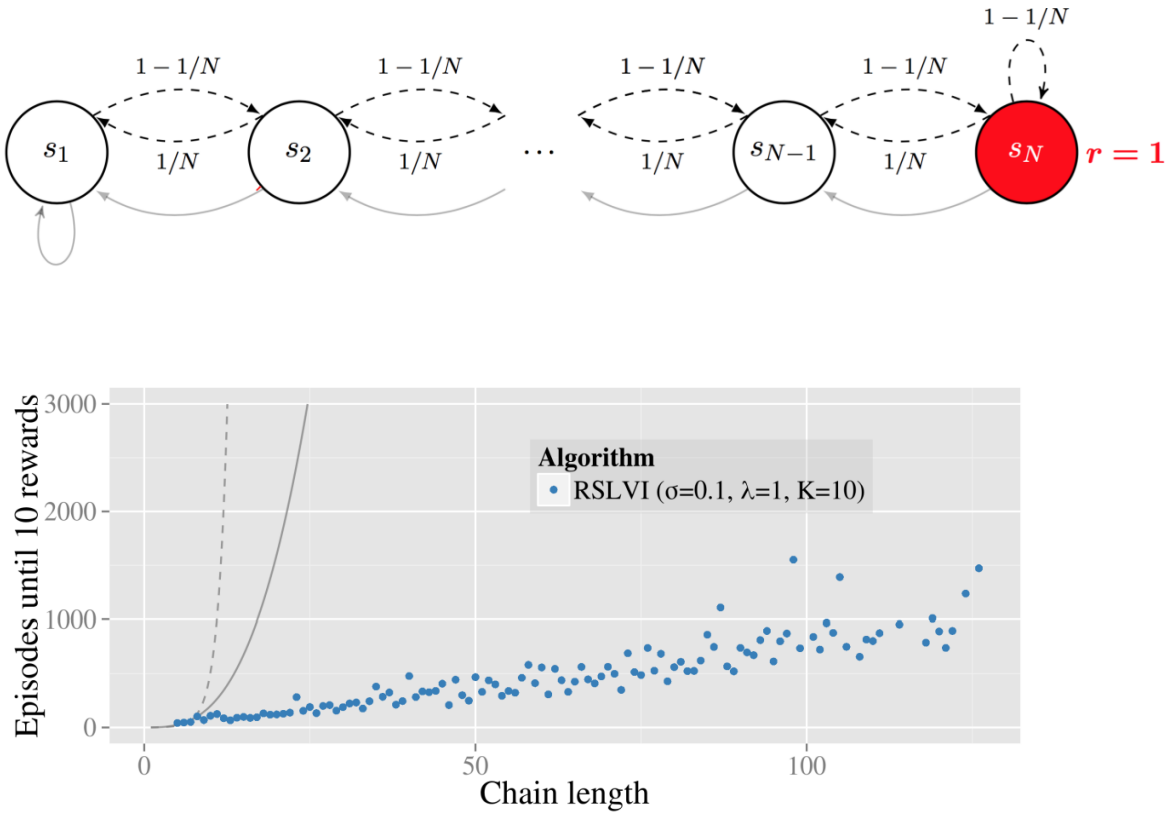


Figure 3. RLSVI learning time against chain length.

Osband, Van Roy, Wen. Generalization and Exploration via Randomized Value Functions. 2014.

Common Approaches of Exploration

- Optimistic Exploration
  - Upper Confidence Bound
- Randomized Exploration
  - Thompson Sampling (Posterior Sampling)
- Information-Directed Exploration

Information-Directed Exploration (1/3)

Another Bayesian approach – like Thompson sampling.

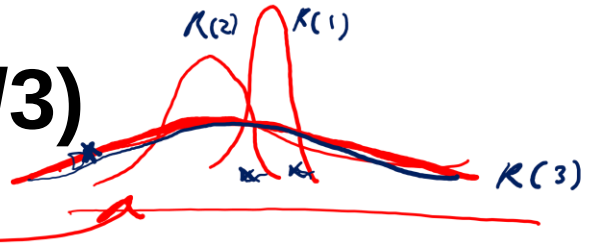
Assume the parameter of the world (e.g., the mean reward of the arms) is drawn as $\theta \sim P_{\text{prior}}$

After observing history $\mathcal{H}_t = (a_1, r_1, a_2, r_2, \ldots, a_{t-1}, r_{t-1})$, we can calculate the posterior distribution of θ:

$$P(\theta|\mathcal{H}_t) = \frac{P(\mathcal{H}_t, \theta)}{P(\mathcal{H}_t)} = \frac{P(\mathcal{H}_t|\theta)P_{\text{prior}}(\theta)}{P(\mathcal{H}_t)} \propto P(\mathcal{H}_t|\theta)P_{\text{prior}}(\theta)$$

Key question: Based on the posterior estimation of the world $P(\theta|\mathcal{H}_t)$, what action should we pick next?

Information-Directed Exploration (2/3)



Thompson Sampling: Sample $\theta_t \sim P(\cdot\,|\mathcal{H}_t)$ and choose $a_t = a^\star(\theta_t) = \operatorname{argmax}_a \theta_t(a)$

The optimal action in the world of θ_t

Equivalently, execute $\pi(a) = \operatorname{argmax}_\pi \mathbb{E}_{\theta\sim P(\cdot|\mathcal{H}_t)}[\mathbb{I}\{a^\star(\theta) = a\}]$

Information-directed Sampling: Select an arm that tradeoffs **regret** and **information gain**

$$\text{Regret}_\theta(\pi) = \max_{a^\star} \theta(a^\star) - \theta(\pi)$$

$$\text{InfoGain}_\theta(\pi) = \mathbb{E}_{r\sim\theta(\pi)}[\text{KL}(P(\cdot\,|\mathcal{H}_t, \pi, r), P(\cdot\,|\mathcal{H}_t))]$$

How much will the posterior change after obtaining a new sample from π?

Execute $\pi = \operatorname{argmin}_\pi \mathbb{E}_{\theta\sim P(\cdot|\mathcal{H}_t)}\left[\text{Regret}_\theta(\pi) - \lambda\, \text{InfoGain}_\theta(\pi)\right]$

Russo and Van Roy. Learning to Optimize via Information-Directed Sampling. 2014.
Foster et al. The Statistical Complexity of Interactive Decision Making. 2021.

Information-Directed Exploration (3/3)

When is information-directed exploration better than optimistic / posterior exploration?

arm 1

arm 2

arm 3

Suppose we know there are two possible worlds, where the three arms follow $\{\text{Bernoulli}(0.5), \text{Bernoulli}(0.6), 0.4\}$ or $\{\text{Bernoulli}(0.6), \text{Bernoulli}(0.5), 0.3\}$

$\Rightarrow$ Although we know arm 3 is definitely not the best arm, we still want to sample it (once), so we can easily tell which world we're in.

Exploration in Large State Spaces with Function Approximation

UCB / TS with Given State-Action Features

Suppose for any (s, a), we have access to a feature vector $\phi(s, a) \in \mathbb{R}^d$.

Then instead of counting the #visits to every state-action, we can evaluate the **~~novelty~~ of the feature**.

$$\Lambda_t = \sum_{i<t} \sum_{h=1}^{H} \phi(s_{ih}, a_{ih}) \phi(s_{ih}, a_{ih})^T$$

LSVI-UCB	$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'\|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + c \cdot \sqrt{\phi(s,a)\Lambda_t^{-1}\phi(s,a)}$

Jin et al. Provably efficient reinforcement learning with linear function approximation. 2019.

RLSVI	$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'\|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + c \cdot \mathcal{N}(0, \phi(s,a)\Lambda_t^{-1}\phi(s,a))$

Zanette et al. Frequentist Regret Bounds for Randomized Least-Squares Value Iteration. 2019.

How to Adapt These Ideas to General Cases?

Ideas from UCB:

or $\frac{1}{N(s,a)}$ (theoretically better for deterministic environment)

1. $\tilde{R}(s,a) = \hat{R}(s,a) + \frac{1}{\sqrt{N(s,a)}}$ where $N(s,a) \approx$ Amount of prior visit to (s,a)
2. $\tilde{R}(s,a) = \hat{R}(s,a) + e(s,a)$ where $e(s,a) \approx$ Prediction error on $\hat{R}(s,a)$ and $\hat{P}(\cdot\,|s,a)$

Ideas from TS:

3. $\tilde{R}(s,a) = \hat{R}(s,a) +$ noise whose variance scales with the uncertainty of $\hat{R}(s,a)$ and $\hat{P}(\cdot\,|s,a)$

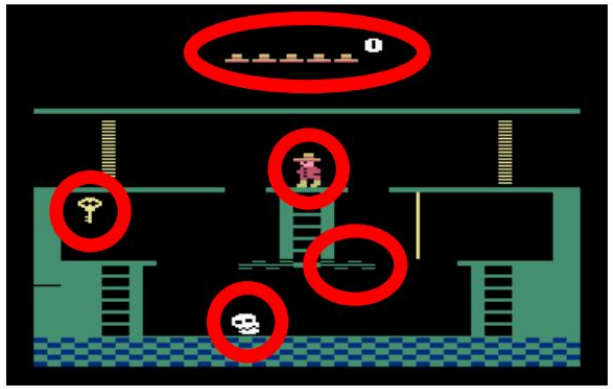
Ideas from Information-directed Sampling:

4. $\tilde{R}(s,a) = \hat{R}(s,a) + \lambda \underbrace{\mathrm{KL}\big(\mathcal{P}(\cdot\,|\mathcal{H}_t, s, a, s'), \mathcal{P}(\cdot\,|\mathcal{H}_t)\big)}_{\text{Information gain}}$

After all these, just perform standard RL algorithm over $\tilde{R}$.

1. Bonus from the Number of Prior Visits

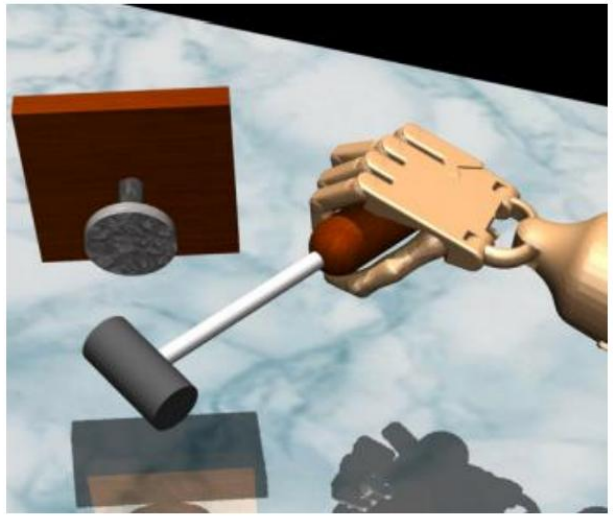
Pseudo Count



idea: fit a density model $p_\theta(\mathbf{s})$ (or $p_\theta(\mathbf{s}, \mathbf{a})$)

$p_\theta(\mathbf{s})$ might be high even for a new $\mathbf{s}$
if $\mathbf{s}$ is similar to previously seen states

can we use $p_\theta(\mathbf{s})$ to get a "pseudo-count"?



if we have small MDPs
the true probability is:

$$P(\mathbf{s}) = \frac{N(\mathbf{s})}{n}$$

count

probability/density

total states visited

after we see $\mathbf{s}$, we have:

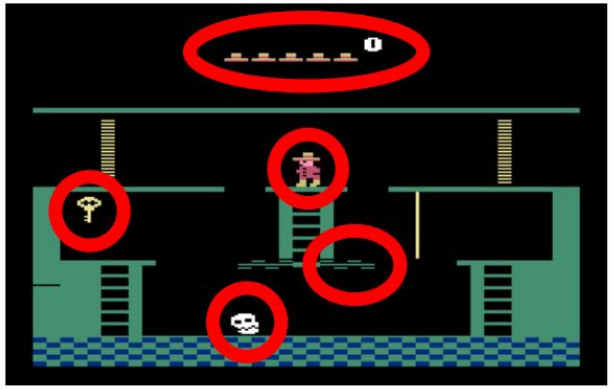
$$P'(\mathbf{s}) = \frac{N(\mathbf{s}) + 1}{n + 1}$$

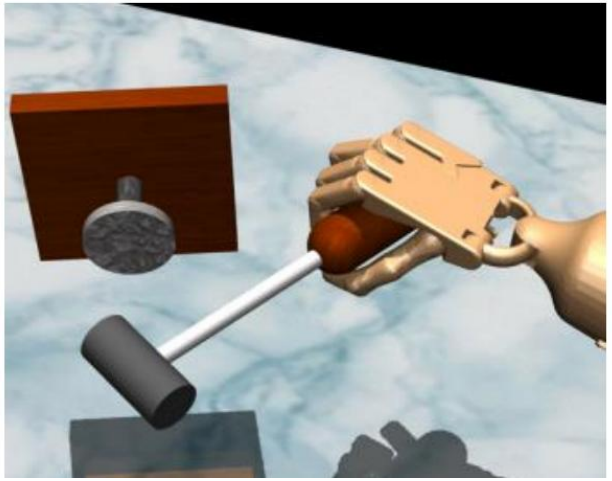
can we get $p_\theta(\mathbf{s})$ and $p_{\theta'}(\mathbf{s})$ to obey these equations?

Bellemare et al. Unifying Count-Based Exploration and Intrinsic Motivation. 2016

(Source: Sergey Levine's Deep RL)

Pseudo Count





fit model $p_\theta(\mathbf{s})$ to all states $\mathcal{D}$ seen so far

take a step i and observe $\mathbf{s}_i$

fit new model $p_{\theta'}(\mathbf{s})$ to $\mathcal{D} \cup \mathbf{s}_i$

use $p_\theta(\mathbf{s}_i)$ and $p_{\theta'}(\mathbf{s}_i)$ to estimate $\hat{N}(\mathbf{s})$

set $r_i^+ = r_i + \mathcal{B}(\hat{N}(\mathbf{s}))$ ← "pseudo-count"

$\hat{N}(s_i)$

how to get $\hat{N}(\mathbf{s})$? use the equations

$$p_\theta(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i)}{\hat{n}} \qquad p_{\theta'}(\mathbf{s}_i) = \frac{\hat{N}(\mathbf{s}_i)+1}{\hat{n}+1}$$

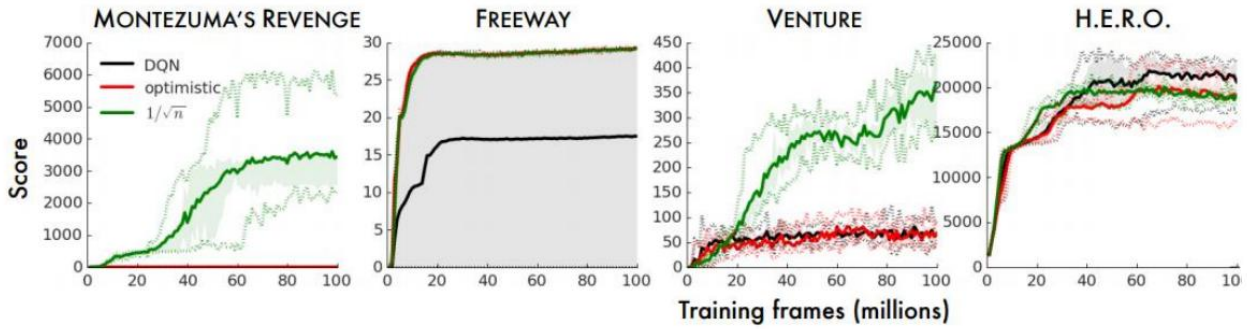
two equations and two unknowns!

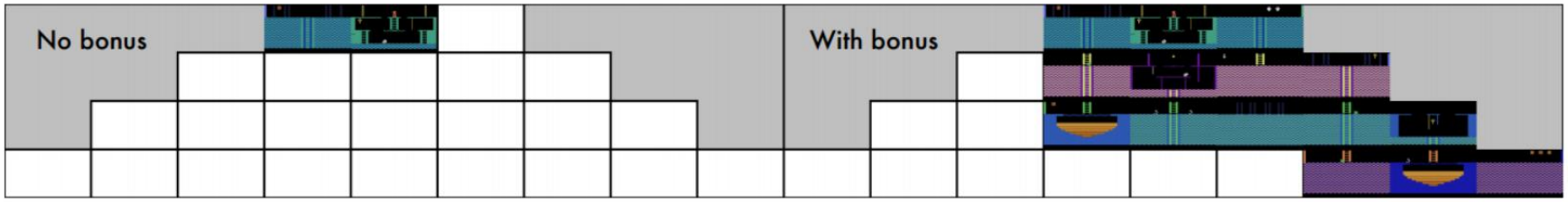
$$\hat{N}(\mathbf{s}_i) = \hat{n} p_\theta(\mathbf{s}_i) \qquad \hat{n} = \frac{1 - p_{\theta'}(\mathbf{s}_i)}{p_{\theta'}(\mathbf{s}_i) - p_\theta(\mathbf{s}_i)} p_\theta(\mathbf{s}_i)$$

Bellemare et al. Unifying Count-Based Exploration and Intrinsic Motivation. 2016

(Source: Sergey Levine's Deep RL)

Pseudo Count





Bellemare et al. Unifying Count-Based Exploration and Intrinsic Motivation. 2016

(Source: Sergey Levine's Deep RL)

Hash

What if we still count states, but in a different space?

idea: compress $\mathbf{s}$ into a k-bit code via $\phi(\mathbf{s})$, then count $N(\phi(\mathbf{s}))$

shorter codes = more hash collisions

similar states get the same hash? maybe

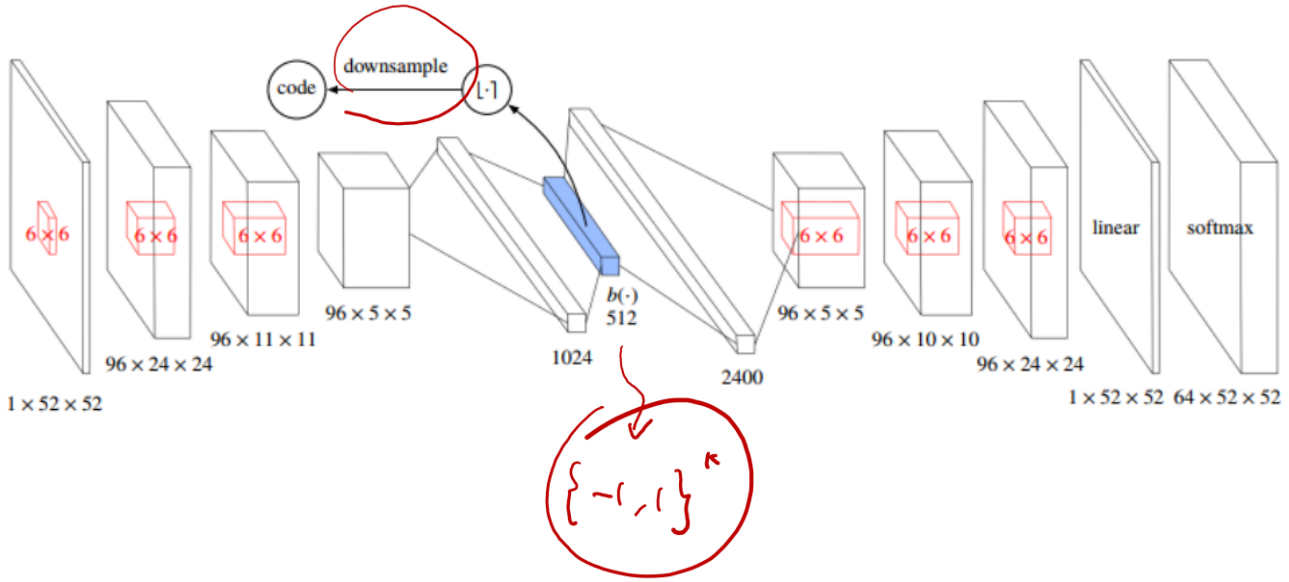
$$\phi(s) = \text{sgn}(Ag(s)) \in \{-1, 1\}^k, \tag{2}$$

where $g : \mathcal{S} \to \mathbb{R}^D$ is an optional preprocessing function and A is a $k \times D$ matrix with i.i.d. entries drawn from a standard Gaussian distribution $\mathcal{N}(0, 1)$. The value for k controls the granularity: higher values lead to fewer collisions and are thus more likely to distinguish states.

#Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. 2017.

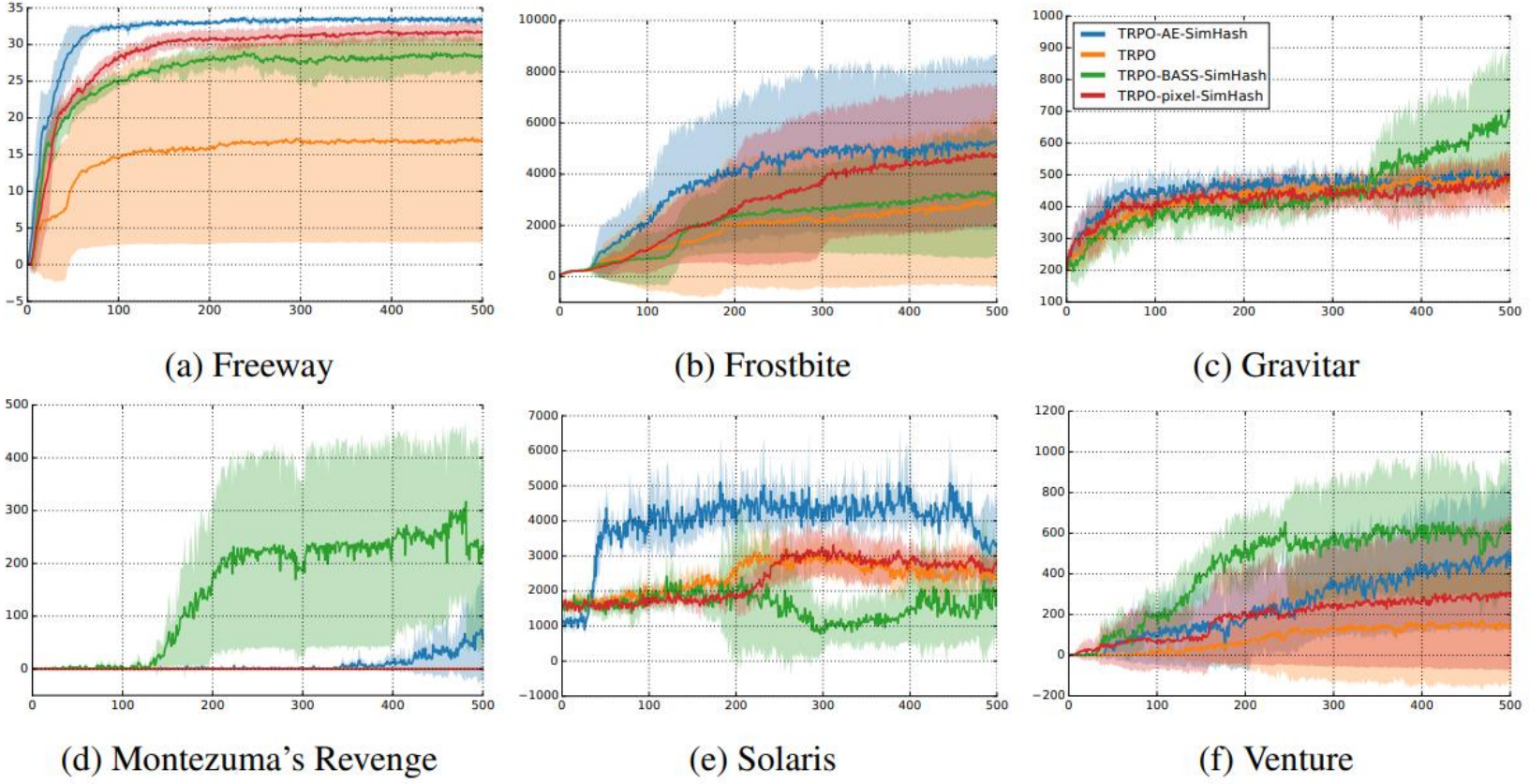
Hash

improve the odds by *learning* a compression:



#Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. 2017.

Hash



(a) Freeway (b) Frostbite (c) Gravitar

(d) Montezuma's Revenge (e) Solaris (f) Venture

#Exploration: A Study of Count-Based Exploration for Deep Reinforcement Learning. 2017.

2. Bonus from Prediction Error

Bonus from Prediction Error

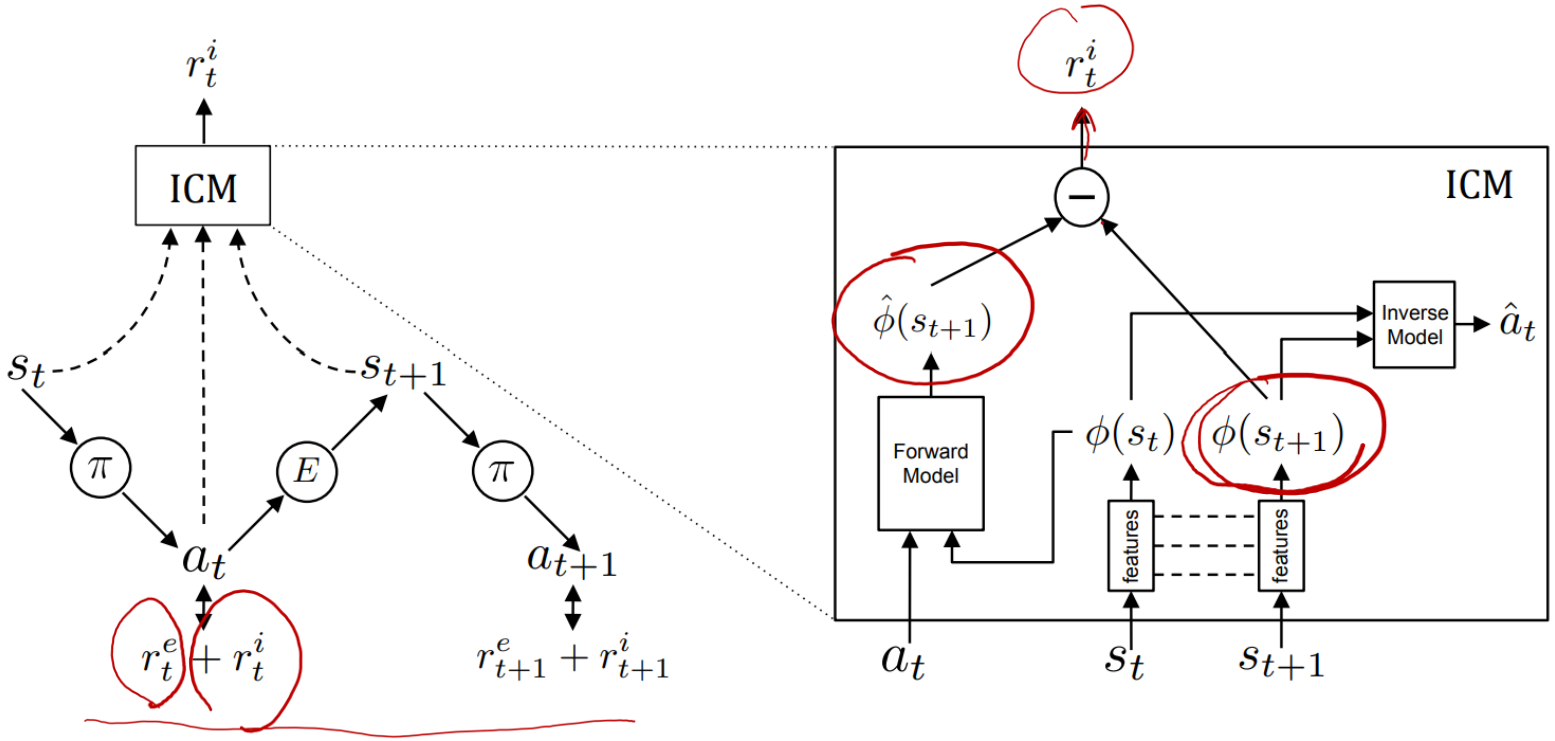
Ideally, we would like to estimate $\left\|\hat{P}(\cdot\,|s,a) - P(\cdot\,|s,a)\right\|$ and set it as bonus.

However, we don't know the ground-truth transition, so the best we can do is try to **predict the next state**.

There are some issues if we naively do this:

1. For stochastic environments where transitions are random, we will never have small prediction error for next state.
2. For many environments, some part of the state is uncontrollable by the learner (e.g., movement of the clouds in the background).

Intrinsic Curiosity Module



- Task 1: Given s_t and s_{t+1}, predict a_t: make the feature ϕ capture action-related dynamics
- Task 2: Given $\phi(s_t)$ and a_t, predict $\phi(s_{t+1})$

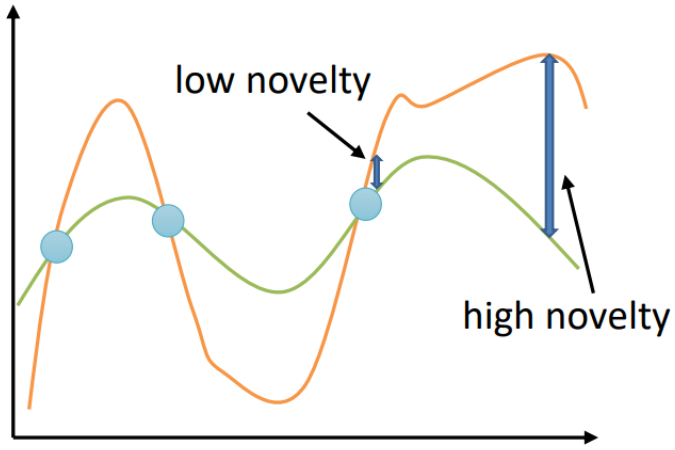
Pathak et al. Curiosity-driven Exploration by Self-supervised Prediction. 2017.

Random Network Distillation

let's say we have some **target** function $f^\star(\mathbf{s}, \mathbf{a})$

given our buffer $\mathcal{D} = \{(\mathbf{s}_i, \mathbf{a}_i)\}$, fit $\hat{f}_\theta(\mathbf{s}, \mathbf{a})$

use $\mathcal{E}(\mathbf{s}, \mathbf{a}) = ||\hat{f}_\theta(\mathbf{s}, \mathbf{a}) - f^\star(\mathbf{s}, \mathbf{a})||^2$ as bonus



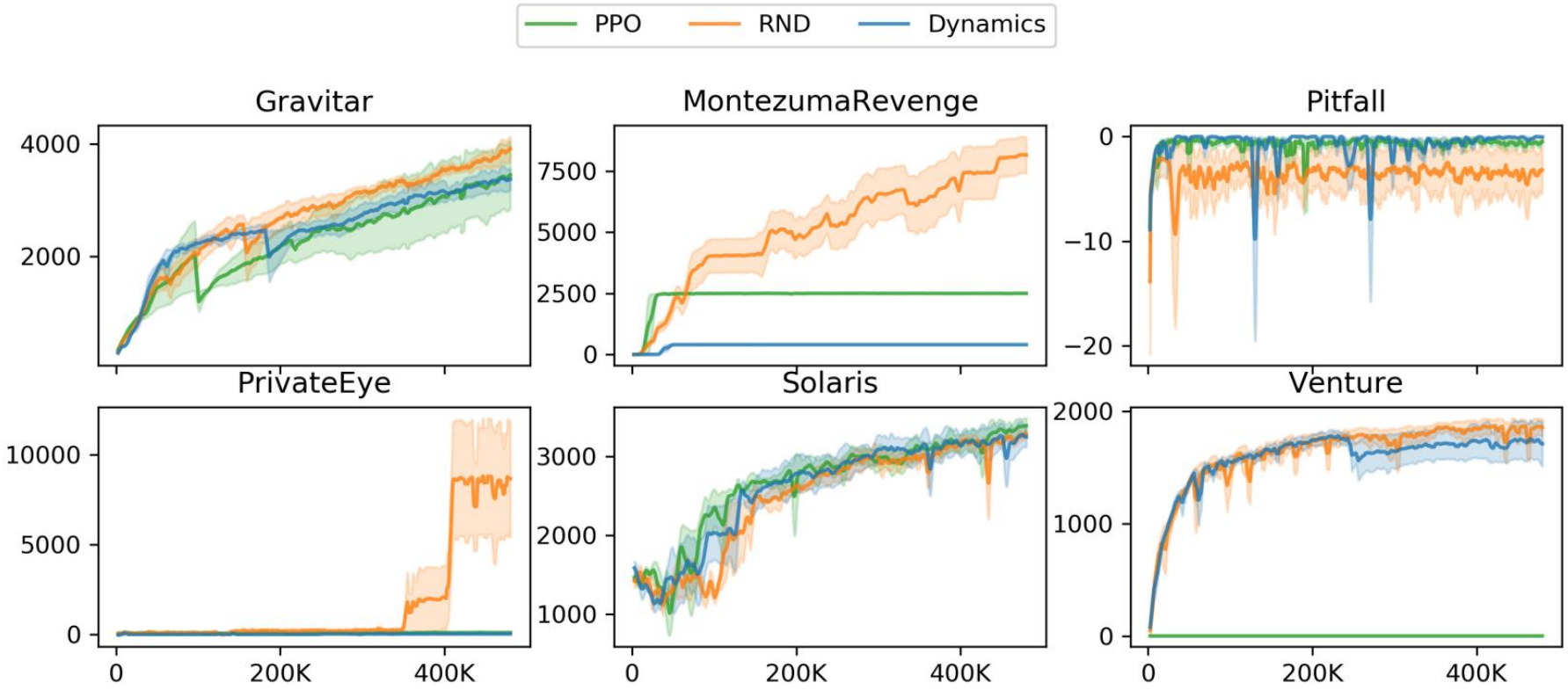
what should we use for $f^\star(\mathbf{s}, \mathbf{a})$?

one common choice: set $f^\star(\mathbf{s}, \mathbf{a}) = \mathbf{s}'$ – i.e., next state prediction

even simpler: $f^\star(\mathbf{s}, \mathbf{a}) = f_\phi(\mathbf{s}, \mathbf{a})$, where ϕ is a *random* parameter vector

Burda et al. Exploration by Random Network Distillation. 2018.

Random Network Distillation



Burda et al. Exploration by Random Network Distillation. 2018.

3. Thompson Sampling

Recall: Randomized Value Iteration

Randomized Value Iteration

Draw a value fun $\tilde{Q}$ from $\mathbb{P}(\cdot|H_t)$

For episode $1, 2, \ldots, T$:

VI with

$\tilde{Q}_{H+1}(s,a) = 0 \quad \forall s, a$

For step $H, H-1, \ldots, 1$:

$$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + H \sqrt{\frac{2\log(2/\delta)}{N_t(s,a)}} \underbrace{n_t(s,a)}_{\sim \mathcal{N}(0,1)}$$

Receive $s_1 \sim \rho$

For step $1, 2, \ldots, H$:

Take action $a_h = \text{argmax}_a \tilde{Q}_h(s_h, a)$

Receive $r_h = R(s_h, a_h) + \text{noise}, \quad s_{h+1} \sim P(\cdot|s_h, a_h)$

Osband, Van Roy, Wen. Generalization and Exploration via Randomized Value Functions. 2014.

Recall: Randomized Value Iteration

$$\tilde{Q}_h(s,a) \triangleq \hat{R}(s,a) + \sum_{s'} \hat{P}(s'|s,a) \max_{a'} \tilde{Q}_{h+1}(s',a') + n_t(s,a) \qquad n_t(s,a) \sim \mathcal{N}\left(0, \frac{1}{N_t(s,a)}\right)$$

Adapting this idea to DQN:

$$\theta = \operatorname*{argmin}_{\theta} \sum_{(s,a,r,s') \in \mathcal{B}} \left(r + \max_{a'} Q_{\bar{\theta}}(s',a') + n_t(s,a) - Q_\theta(s,a) \right)^2 \qquad (*)$$

Notice that difference noise gives different θ.

Direct generalization from Randomized VI (not easy to implement)

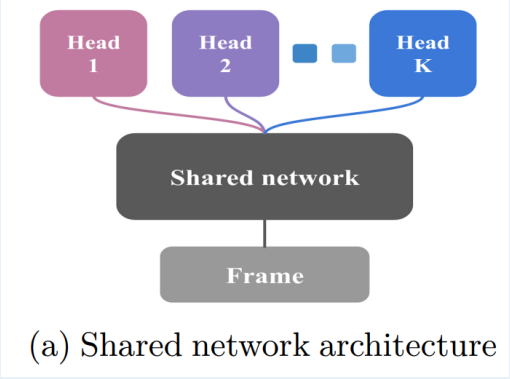
Θ = Space of θ's

In each episode, sample a $\theta \in \Theta$ with the distribution following (*), and execute $\pi(s) = \operatorname*{argmax}_a Q_\theta(s,a)$

Bootstrapped DQN

Osband et al. Deep Exploration via Bootstrapped DQN. 2016.
Osband et al. Randomized Prior Functions for Deep Reinforcement Learning. 2018.

Randomly initialize K instances of DQN $\theta_1, \dots, \theta_K$
(each θ_i has their own target network $\bar{\theta}_i$ and replay buffer $\mathcal{B}_i$).



(a) Shared network architecture

For each episode:

- Randomly sample $i \sim \text{Unif}\{1,2,\dots,K\}$
- Execute $\pi(s) = \max_a Q_{\theta_i}(s,a)$ in the whole episode.
- Randomly place the obtained (s,a,r,s') in some/all replay buffers.
- Update all DQN parameters.

Bootstrapped DQN

Osband et al. Deep Exploration via Bootstrapped DQN. 2016.

Osband et al. Randomized Prior Functions for Deep Reinforcement Learning. 2018.

Some intuitions:

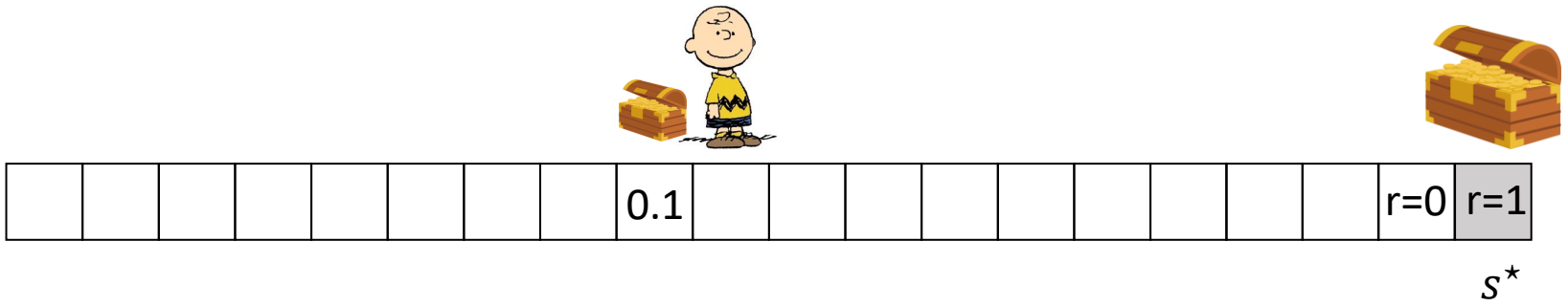
- The random initialization makes $Q_{\theta_1}(s,a), \ldots, Q_{\theta_K}(s,a)$ all very different. We can view them as associated with different initial noise $n_1(s,a)$. $\sim N(0,1)$
- Over the course of training, for (s,a)'s that are more often visited, their effective magnitude of $n_t(s,a)$ decreases (because we train those DQNs without adding more noise).
- For (s,a)'s that are not often visited, their effective magnitude of $n_t(s,a)$ remains high.
- **Why does this perform deep exploration?** For a particular state s, if $\max_a Q_{\theta_i}(s,a)$ is initialized high but has not been visited many times before, the training of θ_i will propagate this high value to other state and encourage the learner to reach s from other states.

Bootstrapped DQN

Osband et al. Deep Exploration via Bootstrapped DQN. 2016.

Osband et al. Randomized Prior Functions for Deep Reinforcement Learning. 2018.

- In the toy example, as long as **one of the K DQNs** initializes $s^\star$ (or some states close to it) with a high value, then it can help the learner explore to $s^\star$.
- In this example, roughly we need $K = \mathrm{O}(\text{number of states})$ to achieve this effect.

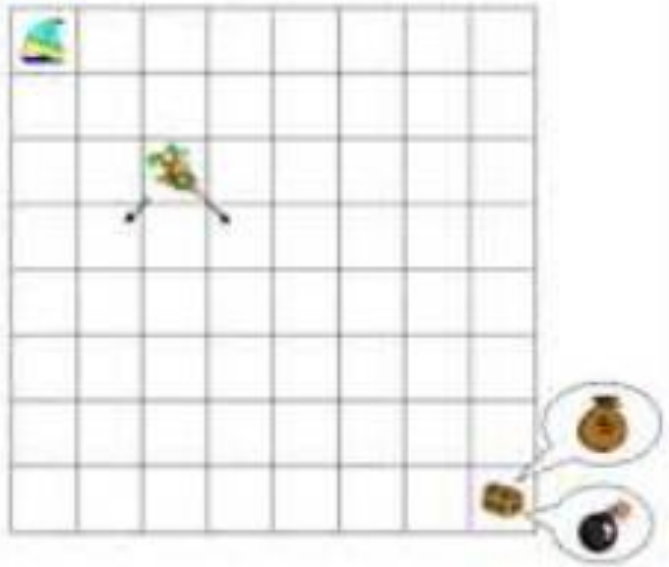


Bootstrapped DQN

Osband et al. Deep Exploration via Bootstrapped DQN. 2016.
Osband et al. Randomized Prior Functions for Deep Reinforcement Learning. 2018.

"Deep Sea" Exploration

- **Stylized "chain" domain testing "deep exploration":**
  - *State = N x N grid, observations 1-hot.*
  - *Start in top left cell, fall one row each step.*
  - *Actions {0,1} map to left/right in each cell.*
  - ***"left" has reward = 0, "right" has reward = -0.1/N***
  - ***... but if you make it to bottom right you get +1.***
- Only one policy (out of more than 2^N) positive return.
- **ε-greedy / Boltzmann / policy gradient / are useless.**

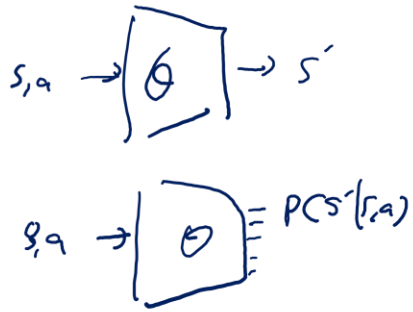


4. Bonus from Information Gain

Estimating the Information Gain

Suppose that we model the world (e.g., state transition) as

$$p_\theta(s'|s,a)$$



In order to calculate the information gain $\mathrm{KL}\big(\mathcal{P}(\theta|\mathcal{H},s,a,s'),\mathcal{P}(\theta|\mathcal{H})\big)$ we have to calculate $\mathcal{P}(\theta|\mathcal{H}) = \dfrac{P(\theta)P(\mathcal{H}|\theta)}{P(\mathcal{H})}$, $P(\mathcal{H}) = \int_\theta P_{\text{prior}}(\theta)\,P(\mathcal{H}|\theta)$

This is a well-known hard problem, and the way to do it is by introducing another parameterized model $q_\phi(\theta)$ to approximate $\mathcal{P}(\theta|\mathcal{H})$.

① $q_\phi(\theta) \approx P(\theta|\mathcal{H})$

p_θ and q_ϕ is trained by maximizing the **variational lower bound**: ② MLE or P_0

$$\mathbb{E}_{\theta\sim q_\phi}[\log p_\theta(\mathcal{H})] - \mathrm{KL}\big(q_\phi(\theta),\mathcal{P}_{\text{prior}}(\theta)\big)$$

Variational Information Maximizing Exploration (VIME)

For $k = 1,2, \ldots$

For $i = 1,2, \ldots N$:

Sample a_i, and observe the reward r_i and next state s_{i+1}

Estimate one information gain $\mathrm{KL}\left(q_{\phi'}(\theta), q_{\phi}(\theta)\right)$

Construct modified reward $r_i' = r_i + \mathrm{KL}\left(q_{\phi'}(\theta), q_{\phi}(\theta)\right)$

$q_\phi \approx$ — before

$q_{\phi'} \approx$ posterior after seeing (s_i, a_i, r_i, s_{i+1})

Use dataset $\{(s_i, a_i, r_i', s_i')\}$ to update the policy

Update p_θ and q_ϕ by maximizing

$$\mathbb{E}_{\theta \sim q_\phi}[\log p_\theta(\mathcal{H})] - \mathrm{KL}\big(q_\phi(\theta), \mathcal{P}_{\mathrm{prior}}(\theta)\big)$$

Houthooft et al. VIME: Variational Information Maximizing Exploration. 2017.