Explicit Planning for Efficient Exploration in Reinforcement Learning

Liangpeng Zhang[1] (L.Zhang.7@pgr.bham.ac.uk), Ke Tang[2](tangk3@sustc.edu.cn), Xin Yao[2,1] (xiny@sustc.edu.cn)
[1]CERCIA, School of Computer Science, University of Birmingham, U.K. [2]Shenzhen Key Laboratory of Computational Intelligence, University Key Laboratory of Evolving Intelligent Systems of Guangdong Province, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China

ABSTRACT

- Systematic exploration strategies: R-MAX, MBIE, UCRL, their variants, etc. are essentially **heuristic-based**
  - Choose actions greedily w.r.t. some predefined heuristics
  - When heuristics do not match MDP's property well, excessive exploration can happen, reducing learning efficiency
- We propose that **explicit planning for exploration** helps
  - Treat exploration as a two-part procedure:
    1. Evaluate how much data is needed at each state-action pair (= specify an *exploration demand matrix*)
    2. Actually explore and collect data (= fulfil the demand)
  - The second step can be explicitly planned by our Value Iteration for Exploration Cost (VIEC) algorithm
- To show how explicit planning helps, exploration behaviours of ε-greedy, R-MAX, MBIE, and the optimal exploration scheme in **tower MDPs** are analysed and compared
  - Systematic strategies (heuristics): $O(n^2md)$ or $O(n^2m + nmd)$
  - Optimal exploration scheme: $O(nmd)$
  - n = #states, m = #actions, d = demand at each state-action pair
- Exploration behaviour analysis also shows that existing systematic strategies are weak to
  - **Distance traps**: uncertainty being wrongly diminished
  - **Reward traps**: irrelevant rewards that mislead exploration

DEMAND MATRIX

- To guarantee the quality of policy, systematic exploration strategies usually use Hoeffding's or Chernoff's inequalities to evaluate how much data should be collected at each (s, a)
  - Leads to PAC property (sample complexity bounds) or regret bounds
  - R-MAX needs $O(\frac{1}{\varepsilon^2(1-\gamma)^4}(n + \ln\frac{nm}{\delta}))$ data at every (s, a) to be (ε, δ)-PAC [Kakade, 2003; Strehl et al., 2009]
  - MBIE needs $O(\frac{1}{\varepsilon^2(1-\gamma)^4}(n + \ln\frac{nm}{\varepsilon(1-\gamma)\delta}))$ [Strehl and Littman, 2008]
- In practice, obtaining a sufficiently good policy do not need that much data, so people manually tune the exploration parameters to control how much data to be collected
- We call such specifications of data requirement as **(exploration) demands**
  - **Demand matrix** D: entry $D[s, a] = k$ means at least k data should be collected at (s, a) in the following exploration activity
  - For R-MAX with (ε, δ)-PAC property, the elements of the initial demand matrix D_o is uniformly set to $D_o[s, a] = k_0$ for all (s, a) where $k_0 = O(\frac{1}{\varepsilon^2(1-\gamma)^4}(n + \ln\frac{nm}{\delta}))$
  - MBIE & UCRL: although they seem to decide the demand "on the fly" by using confidence intervals (CIs), the frequentist CIs themselves work as *pre-data* analysis, which means that the demand is *already fixed before learning* given the parameter settings (i.e. does not change "on the fly", which is why they can be confidence procedures and thus have performance guarantees)
  - If you already have some prior knowledge to MDP, you can directly specify the demand matrix to reduce unnecessary exploration
  - Simple strategies such as ε-greedy do not have fixed demand matrix and rely on pure luck to find useful information → low efficiency

PLANNING FOR EXPLORATION

- Exploration can be regarded as a two-step procedure:
  - Step1: Specify the exploration demand
  - Step2: Fulfil the demand by collecting data through MDP interaction
  - (optional) jump to Step1 with updated information
- Systematic exploration strategies excel at Step1, but Step2 is treated with less deliberation. That's where *planning* comes in
- Current demand D_t reduces by 1 at (S_t, A_t) after (S_t, A_t) is executed (unless it is already 0) while other elements remain unchanged, i.e.

$$D_{t+1}[s,a] = \begin{cases} \max\{0, D_t[s,a] - 1\}, & (s,a) = (S_t, A_t) \\ D_t[s,a], & \text{otherwise} \end{cases}$$

Such operation is written as $D_{t+1} = H(D_t; S_t, A_t)$
- An **exploration scheme** ψ is a mapping from demand-state pairs to actions, i.e. $\psi(D; s) = a$ indicates action a should be taken when at state s and current demand is D
- **Exploration cost** $C^{\psi}(D; s, a)$ is the expected #steps needed for D to become 0 in an MDP interaction process starting from (s, a) and following exploration scheme ψ
- The planning for exploration problem is an *augmented undiscounted MDP* with:
  - (Augmented) state space $\mathcal{D} \times \mathcal{S}$, action space $\mathcal{A}$
  - Transition $\Pr(D', s', a | D, s) = \begin{cases} P(s'|s,a), & D' = H(D; s, a) \\ 0, & \text{otherwise} \end{cases}$
  - Each step yields cost 1 when $D_t \neq \mathbf{0}$, and cost 0 after $D_t = \mathbf{0}$
- Thus Bellman equation for exploration cost is

$$C^{\psi}(D; s, a) = \begin{cases} 1 + \sum_{s' \in \mathcal{S}} P(s'|s,a) C^{\psi}(D'; s', \psi(D', s')), & D \neq \mathbf{0} \\ 0, & D = \mathbf{0} \end{cases}$$

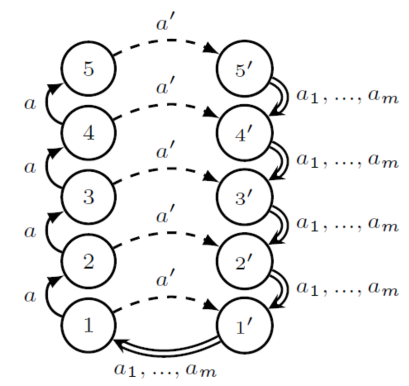
where $D' = H(D; s, a)$.
- We want less exploration cost, so optimal scheme ψ^* has the least $C^{\psi}(D; s, a)$ at every demand-state-action tuple $(D; s, a)$
- **Bellman optimality equation for exploration cost** is

$$C^*(D; s, a) = \begin{cases} 1 + \sum_{s' \in \mathcal{S}} P(s'|s,a) \min_{a' \in \mathcal{A}} C^*(D'; s', a'), & D \neq \mathbf{0} \\ 0, & D = \mathbf{0} \end{cases}$$

where $D' = H(D; s, a)$.
- The optimal exploration scheme ψ^* can be computed by our **Value Iteration for Exploration Cost (VIEC)** algorithm which solves the above equation through a modified Value Iteration process (see paper for detail)
  - Since the demand space $\mathcal{D}$ is exponential in the number of states $|\mathcal{S}|$, the computation of ψ^* is expensive
  - However, most of $\mathcal{D}$ becomes irrelevant after the initial D_0 is given
  - So it should be possible to significantly speed up the computation with techniques such as prioritised sweeping (left to future work)
- VIEC needs to know transition probabilities P to compute ψ^*. When P is unavailable, we can use the estimated transition $\hat{P}$ instead, following an iterative process:
  - Compute exploration scheme $\psi = \mathrm{VIEC}(D, \hat{P})$
  - Collect data by following ψ
  - Update $\hat{P}$ from collected data, jump to the first step with updated $\hat{P}$

TOWER MDP: WHERE HEURISTICS FAIL

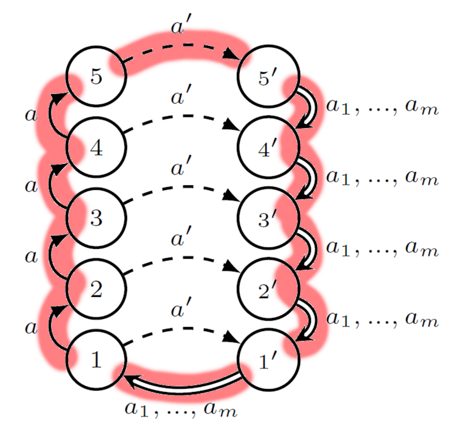
- We use tower MDPs to analyse when and how heuristics fail and planning helps
- A tower MDP of height $h = 5$:



- Upward states $\{s_1, \ldots, s_h\}$, downward states $\{s'_1, \ldots, s'_h\}$
- Taking action a at $s_i \in \{s_1, \ldots, s_{h-1}\}$ goes to s_{i+1} with pr=1
- Taking action a' at $s_i \in \{s_1, \ldots, s_h\}$ goes to s'_i with pr=1
- Each s'_i is an m-armed bandit with unknown reward distributions and leads to s'_{i-1} (or s_1 if $i = 1$)
- Initial demand D_0: uniformly set to a positive interger d for all m-armed bandits, and set to 0 for all (s_i, a) and (s_i, a') due to no uncertainty there

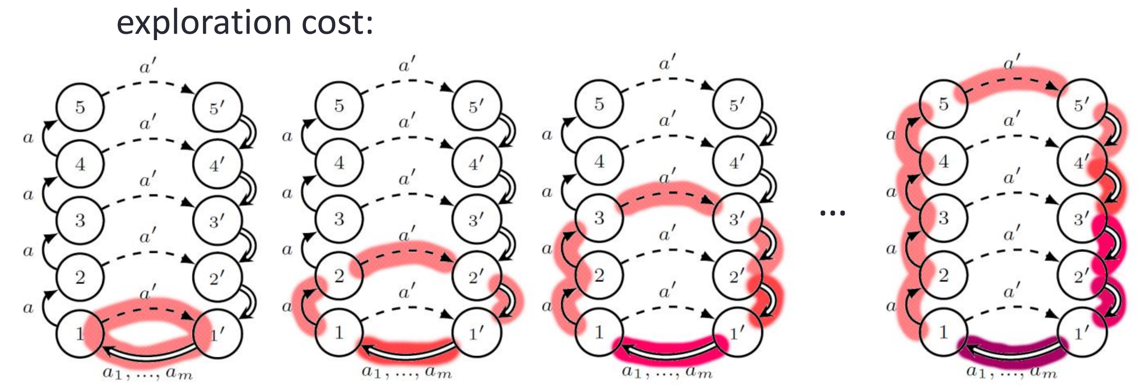
TOWER MDP: OPTIMAL SCHEME

- Optimal scheme: take the marked path $m \times d$ times, each time select a bandit arm with positive demand at the corresponding downward state
- Total exploration cost is $2hmd = \boldsymbol{\Theta(nmd)}$.



TOWER MDP: R-MAX

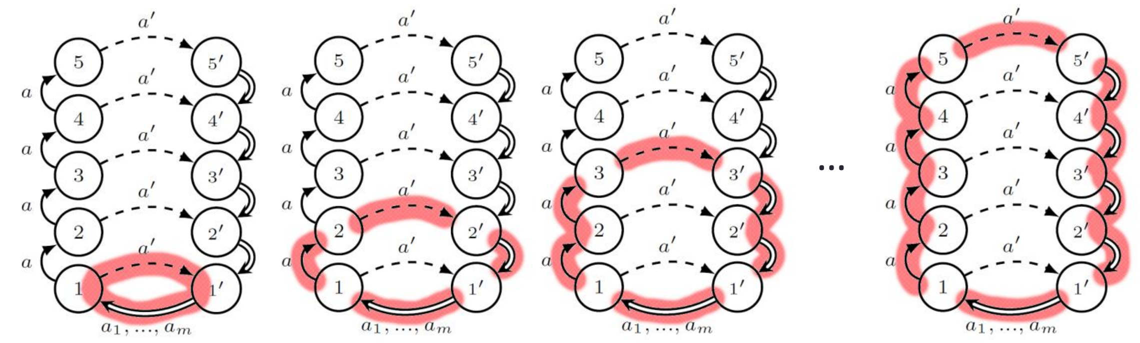
- R-MAX chooses action with highest $\tilde{Q}(s, a)$ which is computed using a modified Bellman equation where "unknown" (s, a) (the ones with positive demand) have $\tilde{Q}(s, a) = \frac{R_{\max}}{1-\gamma}$, which encourages exploration to such (s, a).
- **Distance trap**: due to such design, "uncertainty" is discounted when passed to other states, resulting in R-MAX tending to prioritise the "unknown" state-actions that are near the current state
- In tower MDPs, this results in R-MAX being strongly attracted by the closest "unknown" bandits, greatly increasing the total exploration cost:



- Total exploration cost: $2md + 4md + \cdots + (2h)md = h(h+1)md = \boldsymbol{\Theta(n^2md)}$.

TOWER MDP: MBIE

- MBIE chooses action with highest $\tilde{Q}(s, a)$ which is computed using Bellman equation with modified $\tilde{P}$ and/or $\tilde{R}$ that are the upper bounds of confidence intervals of $\hat{P}$ and $\hat{R}$
- Like R-MAX, MBIE discounts "uncertainty" with distance and thus is weak to the distance traps
- However, the "uncertainty" (represented by $\frac{\beta}{\sqrt{N(s,a)}}$ terms in CIs) diminishes as the number of data $N(s, a)$ increases, which makes MBIE being absorbed less than R-MAX by a nearby "unknown" bandit in tower MDPs:



- In the best case (being trapped by each arm only once), the total exploration cost of MBIE is $(2 + 4 + 6 + \cdots + 2h)m + 2hm(d-1) = \boldsymbol{\Theta(n^2m + nmd)}$
- **Reward trap**: Further, if all bandits give positive rewards, MBIE can be attracted more often than above by the lower-level bandits due to these rewards being considered in $\tilde{Q}$
  - Thus its actual performance will be between $\Theta(n^2m + nmd)$ and $\Theta(n^2md)$
  - Remark: reader may argue that rewards are not always "traps" because they can be designed to guide exploration. However, practice tells us designing a reward function that can properly represent the learning target is already difficult (otherwise we don't need Inverse RL), so designing a reward function that can *both* represent the learning target *and* guide exploration is very difficult. If you try training a Super Mario agent you in this way, you will see Mario ignoring the goal and just trying to get coins infinitely.

CONCLUSION & FUTURE WORK

- Existing systematic strategies are good at specifying the exploration demands, but are weak at fulfilling them
  - Prone to distance & reward traps
- Explicit planning for exploration helps fulfil the exploration demand more efficiently
  - It avoids unnecessary revisits to already explored states that are caused by greedily following predefined heuristics
  - The planning problem can be described as augmented MDPs
  - Optimal exploration scheme exists and can be found by solving Bellman optimality equation for exploration cost
  - Our VIEC algorithm can solve it, though with a high computational cost
- Future work:
  - Fast approximation to VIEC, since the augmented state space is too large and difficult to iterate over
  - Try to classify common MDPs by their dynamic properties, and find out which heuristics are helpful for each class – they can still be helpful when the properties of heuristics and MDP matches