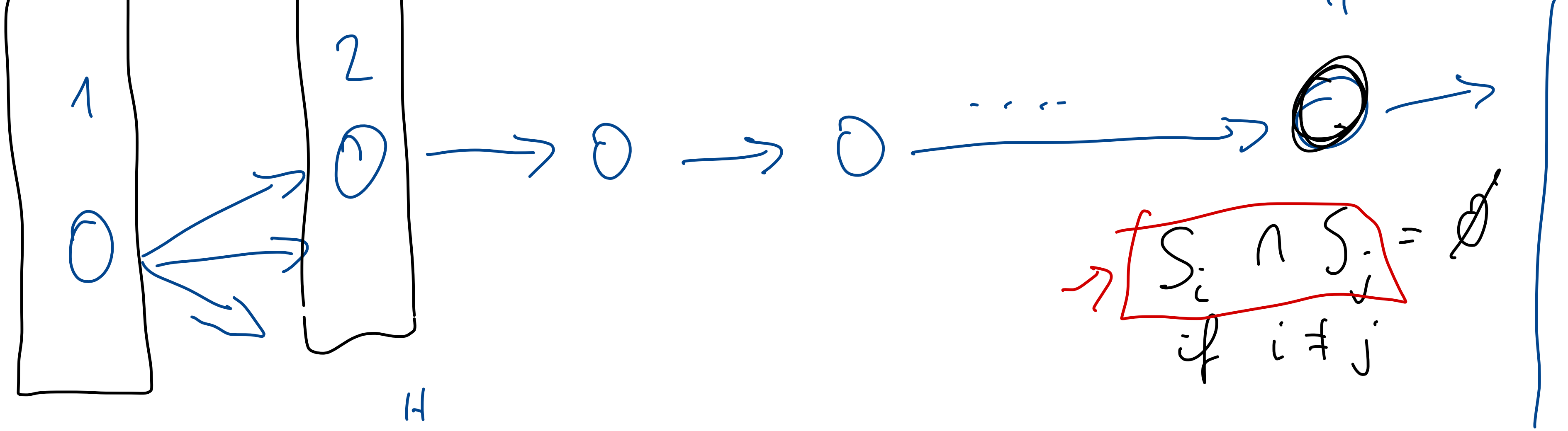
Lecture 10
Planning under q^* realizability

The planner will be given a feature map ϕ_h for every stage $0 \leq h \leq H-1$ such that $\boxed{\phi_h} : \boxed{\mathcal{S}_h} \times \mathcal{A} \to \mathbb{R}^d$. The realizability assumption means that

S_1 S_2

$$\inf_{\theta \in \mathbb{R}^d} \max_{0 \leq h \leq H-1} \|\Phi_h \theta - q_h^*\|_\infty = 0 \,. \tag{1}$$



$S_i \cap S_j = \emptyset$ if $i \neq j$

$q^*_{h_\pi} = \max_\pi E_{R_i \sim \pi} \sum_{t=h}^{H} R_t$

$s \in S_H$

$q(s,a) = r_{sa}$

$s \in S_i \to s' \in S_{i+1}$

Theorem (worst-case query-cost is exponential under q^*-realizability): For any d, H large enough and any online planner $\mathcal{P}$ that is $\boxed{9/128}$-sound for the H-horizon planning problem, there exists a triplet (M, s_0, ϕ) where M is a finite MDP with random rewards taking values in $\boxed{[0,1]}$ and deterministic transitions, s_0 is a state of this MDP and ϕ is a d-dimensional feature-map such that (1) holds for the optimal action-value function $q^* = (q_h^*)_{0 \le h \le H-1}$ and the expected number of queries q that $\mathcal{P}$ uses when interconnected with (M, s_0, ϕ) satisfies

$$\boxed{q = e^{\Omega(d \wedge H)}}$$

$H \sim \frac{1}{1-\gamma}$

(Optimistic Constraint Propagation)

Design principles:

- signal-to-noise ratio of almost all actions must be low
- this must hold for all stages that are easy to reach (eg initial state, random last-stage state, etc.)
- => random last-stage state q^* must be tiny

For simplicity there is only ever 1 reward during an episode so q^*=this reward

- Take ~exp(d) many JL vectors such that $|\langle a,b\rangle| \le 1/4$ and $\langle a,a\rangle=1$
- Each JL vector corresponds to an action
- a^* always optimal
- if a^* played, $r(s,a^*)=q^*(s,a^*)$, transition to exit lane
- if a^* never played, reward only in the last stage: $r(s,a)=q^*(s,a)$
    - but this is ~exp(-H) tiny!

To "solve" MDP (get a delta-sound planner for some const delta):
- either find a^* (needle-in-a-haystack, takes ~exp(d) steps)
- or learn from final-stage rewards (low SNR, takes ~exp(H) steps)

To get final-stage reward that small,

introduce penalty for every suboptimal action.

If a_1, a_2, …, a_n were previous actions to get to s:

q*(s,a)=penalty(a_1, a_2)*penalty(a_2, a_3)*…*penalty(a_n, a)*penalty(a, a*)

phi(s,a)=[1, JL(a)/2 * penalty(a_1, a_2)*penalty(a_2, a_3)*…*penalty(a_n, a)~~*penalty(a, a*)~~]

theta*=[1, JL(a*)]

$$f(s,a) = \frac{1}{2}\left(\prod_{i=1}^{n-1} \text{penalty}(a_i, a_{i+1})\right) \cdot [1, JL(a)]$$

where

- penalty(x, y) = (<JL(x), JL(y)> + 1) ~~*~~ 2
- ie. remap JL vectors’ inner products to [0, 1] (easy linear OP)
    - each penalty factor above is <=5/8, unless a_i=a* because disallow repeated action
    - => q* exponentially decreases
- observe penalty(a*, a*)=1
    - => pulling a* in next action always gives q*(s,a) reward (consistency)

$$q^* = m \, E \sum R$$

$$\begin{tikzcd} \bigcirc \arrow[d, "a"] \\ s \end{tikzcd}$$

$$q^*(s, a) = 0$$

Extension 1: do we need so many actions?

A: NO! https://arxiv.org/abs/2110.02195

TensorPlan and the Few Actions Lower Bound for Planning in MDPs under Linear Realizability of Optimal Value Functions

Gellért Weisz
DeepMind, London, UK
University College London, London, UK

Csaba Szepesvári
DeepMind, London, UK
University of Alberta, Edmonton, Canada

András György
DeepMind, London, UK
Imperial College London, London, UK

- Only d actions
- d in exp(d) lower bound replaced by p:=d^1/4:
    - because q* will now be a 4th-order polynomial in p
    - ie. linear in d
- Main idea:
    - Replace JL vectors with corners of a p-dimensional hypercube
        - WHP inner products of randomly picked corners small
    - Split the selection of a corner (previously, the action) into p “rounds” steps
    - Intricate rules to ensure:
        - close corners cannot be selected in consecutive rounds
        - pi* greedily moves the corner selection close to theta*

Extension 2: online planning vs online RL

Is it harder if you cannot plan?

A: YES! Exponentially so, at least when you also assume suboptimality gap.

Planning can be solved in poly() queries with this assumption (how?)

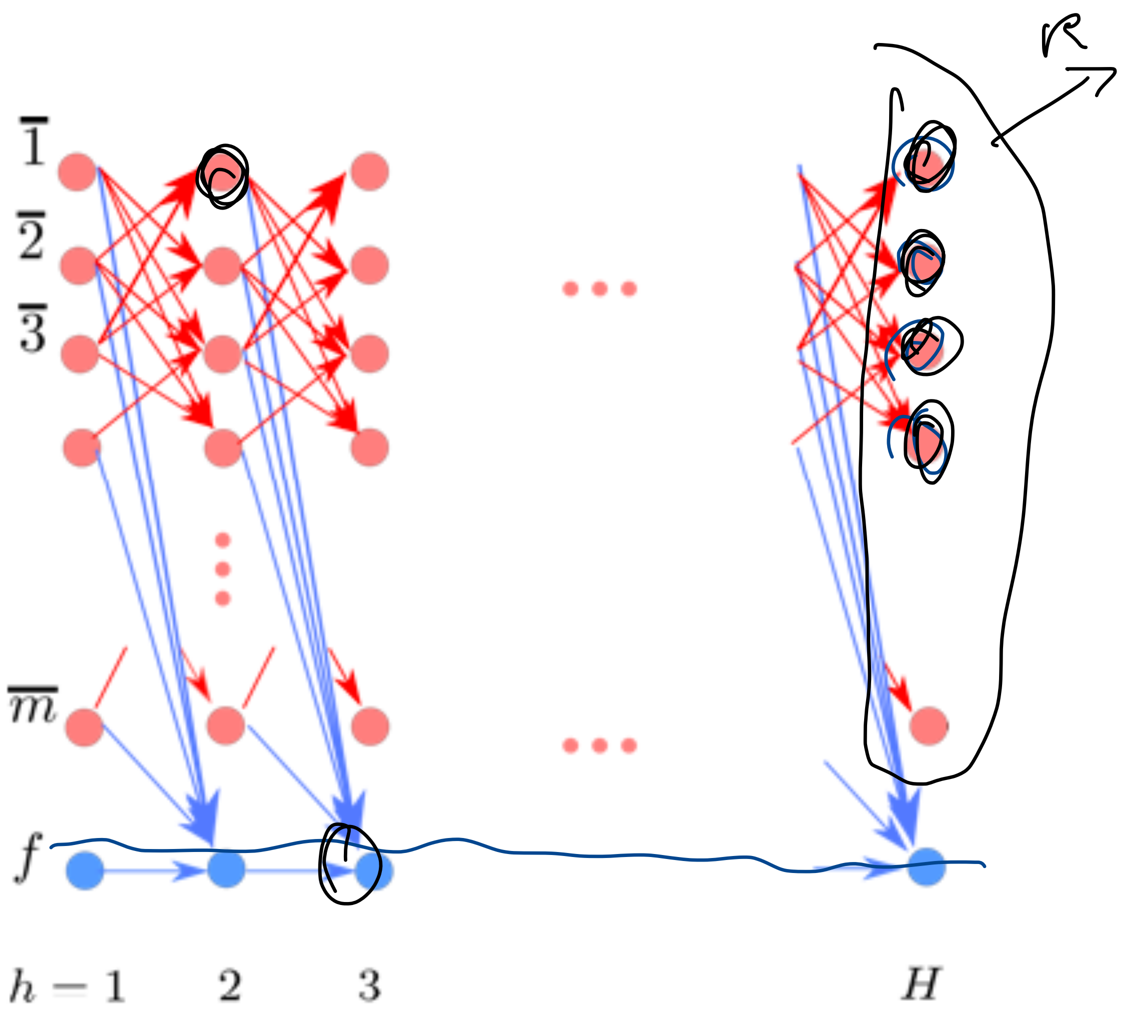
Assumption 2 (Minimum Gap). For any state $s \in \mathcal{S}$, $a \in \mathcal{A}$, the suboptimality gap is defined as $\Delta_h(s,a) := V_h^*(s) - Q_h^*(s,a)$. We assume that $\min_{h\in[H],s\in\mathcal{S},a\in\mathcal{A}} \{\Delta_h(s,a) : \Delta_h(s,a) > 0\} \geq \Delta_{\min}$.

An Exponential Lower Bound for Linearly-Realizable MDPs with Constant Suboptimality Gap

Yuanhao Wang* Ruosong Wang† Sham M. Kakade‡

Same effect as downscaling with penalty factor:

- transition to exit lane with probability corresponding to penalty



Easy with planning: keep replaying until red transition

Hard with online RL: even though last-stage rewards remain large, WHP cannot get there

s, a