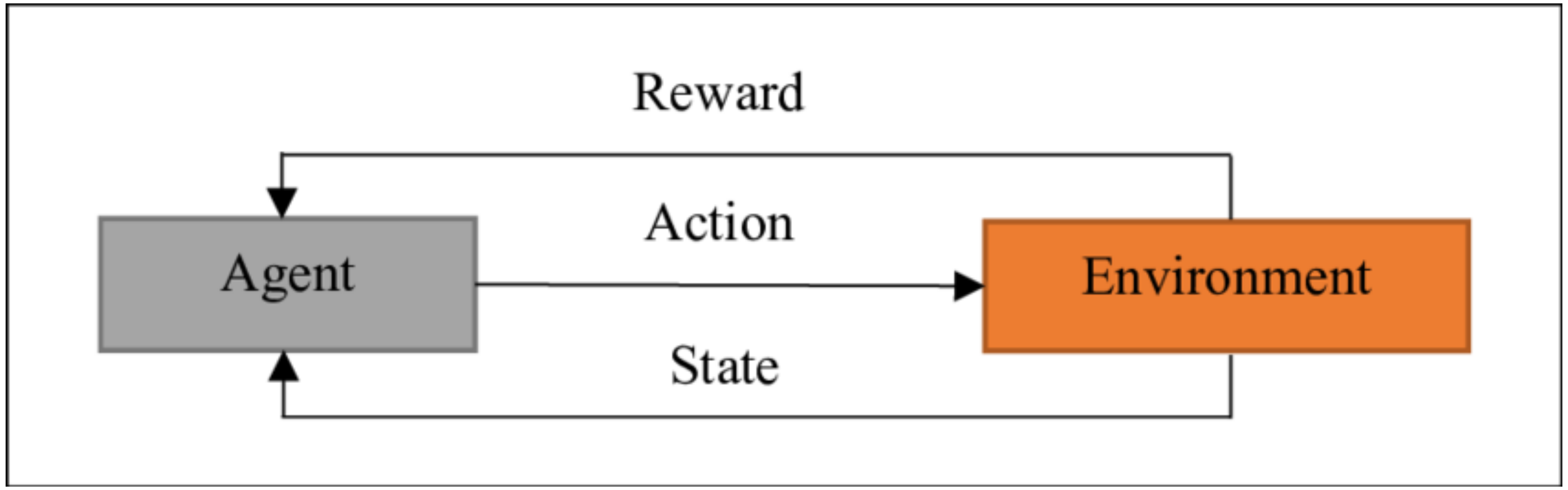
Reinforcement learning

RL Refresh

- No need for large volume of human-curated data (labels)
- Learning during operation (or something close to it like a simulation)
- Example: controls of a high DoF humanoid robot



Humanoid robot control

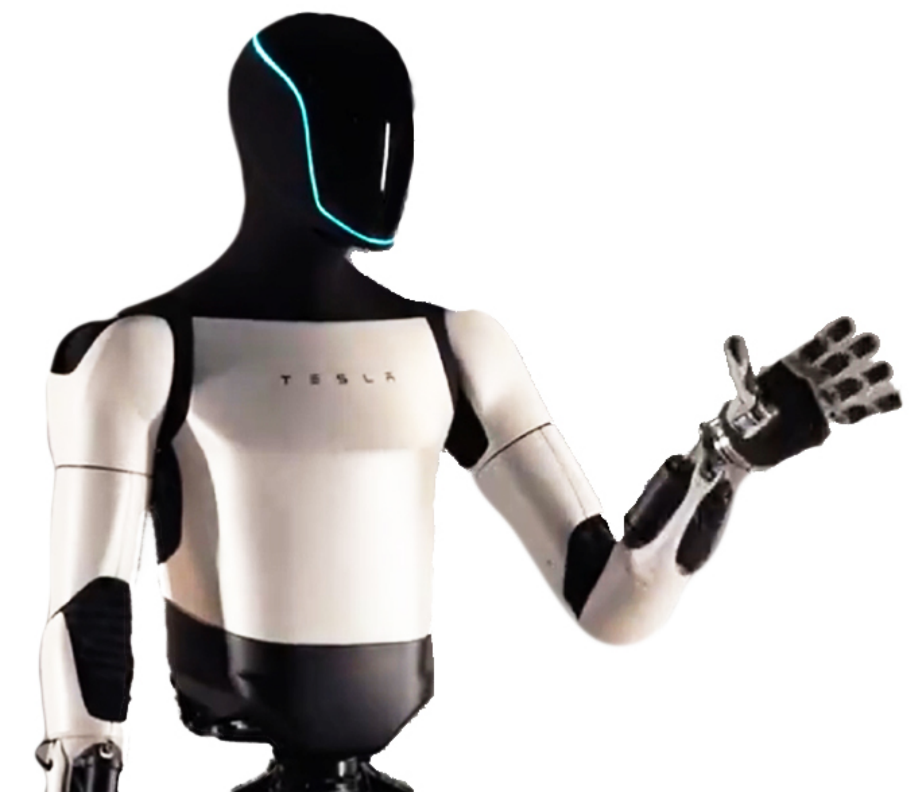
Problem: given a camera image and the dynamics/kinematics of a humanoid robot, control 16 different joints (16 dim vector of motor torques)

Supervised learning

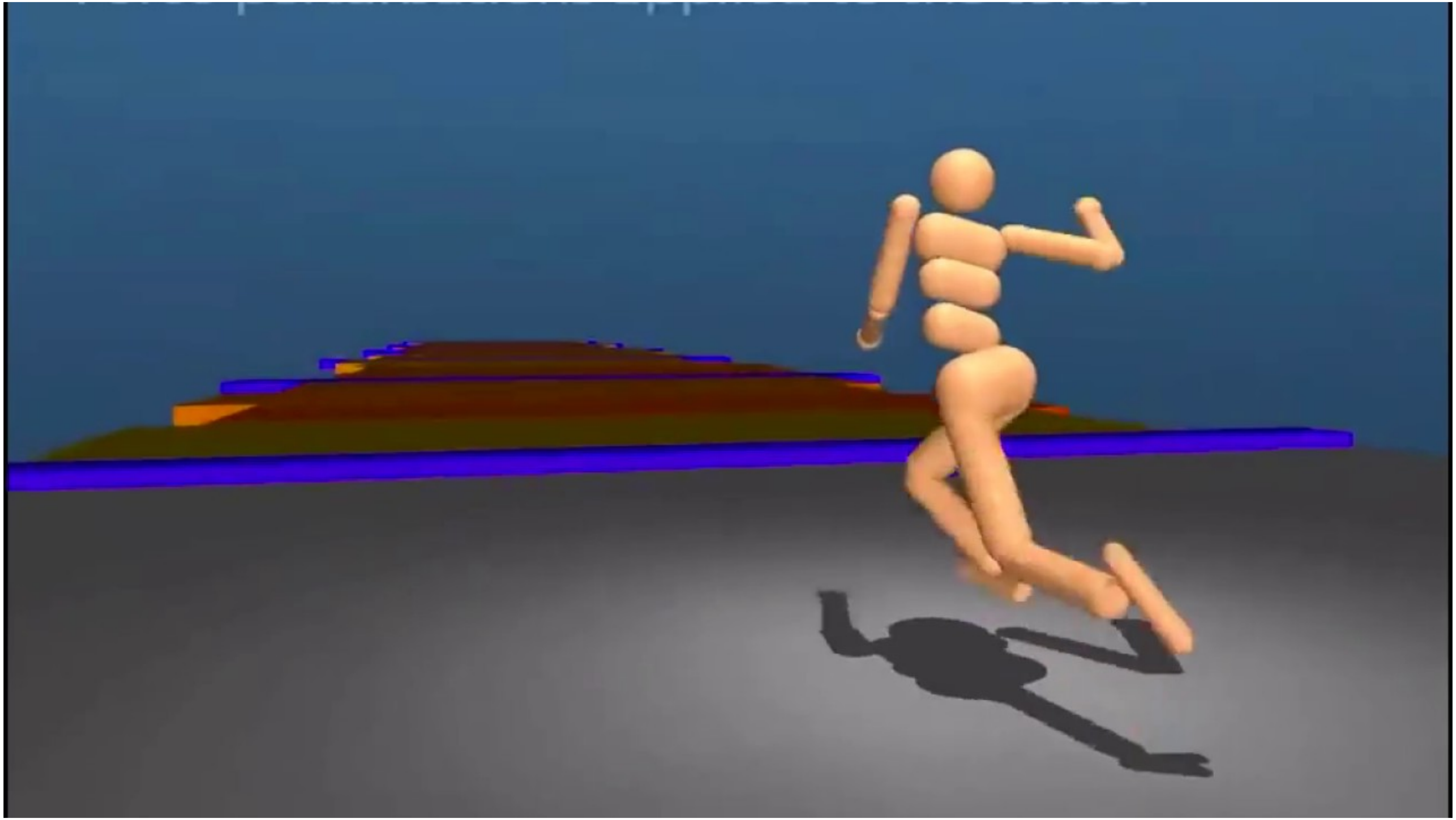
- (numerous) human operators
- 1000s of hours of walking
- Operators themselves must know how to react and stabilize the robot

Reinforcement learning

- -(roll^2 + pitch^2) in a simulator



Humanoid robot control

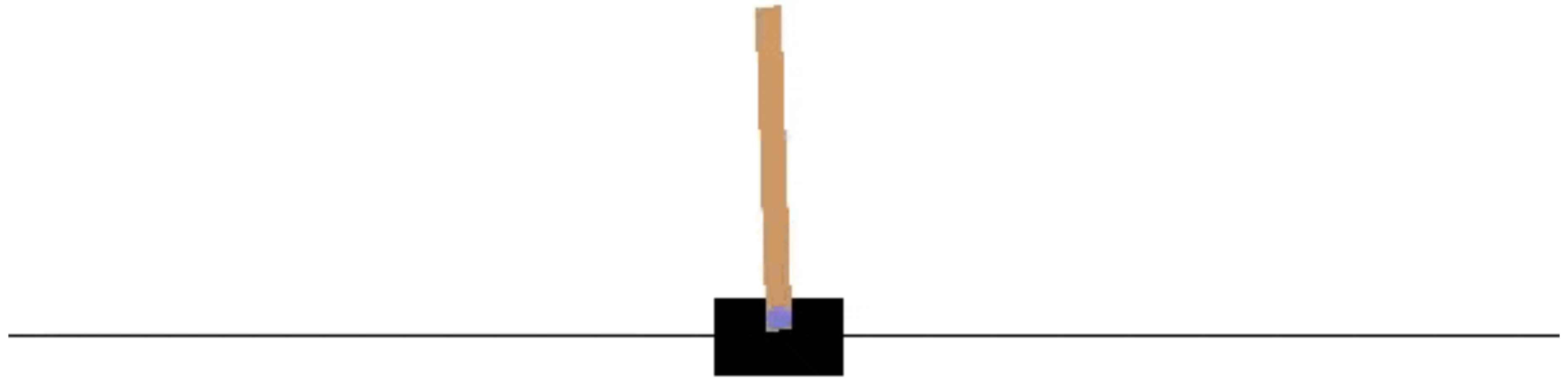


RL Definitions

- The process is formalized as an MDP: $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t, \mathbf{x}_{t-1}, \mathbf{u}_{t-1}, \ldots, \mathbf{x}_0, \mathbf{u}_0) = p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t)$
- $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t)$ is the world/process model (e.g. bicycle model of a car)
- State space $X \subset \mathbb{R}^n$, control space $U \subset \mathbb{R}^m$
- Reward $r : X \times X \times U \rightarrow \mathbb{R}$
- Discount $\gamma \in (0,1)$
- Policy: $\pi : X \rightarrow U$, either probabilistic (distribution $\pi_t(\mathbf{u}_t \mid \mathbf{x}_t)$) or deterministic
- Value function of a policy: $V^\pi : X \rightarrow \mathbb{R}$
- Action-value function of a policy: $Q^\pi : X \times U \rightarrow \mathbb{R}$
- Advantage function of a policy: $A^\pi : X \times U \rightarrow \mathbb{R}$

Why do we need the discount factor?

- The task being solved is either episodic or continuous
- Episodic: explicit end (we find the exit from a maze, checkmate in chess)
- Continuous: task never ends - balancing a pole on a cart
- If we just summed the rewards, it would go to infinity in such cases
- By multiplication with γ we create a finite sum



Functions

- Action-value function can be computed with Bellman's equation as

$$Q^{\pi}(\mathbf{x}, \mathbf{u}) = \int_{\mathbf{x}'} p(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}) \left[r(\mathbf{x}', \mathbf{x}, \mathbf{u}) + \gamma V^{\pi}(\mathbf{x}') \right] \, d\mathbf{x}'$$

- Similarly for the value function

$$V^{\pi}(\mathbf{x}) = \int_{\mathbf{u}} \pi_t(\mathbf{u} \mid \mathbf{x}) \int_{\mathbf{x}'} p(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}) \left[r(\mathbf{x}', \mathbf{x}, \mathbf{u}) + \gamma V^{\pi}(\mathbf{x}') \right] \, d\mathbf{x}' \, d\mathbf{u} = \int_{\mathbf{u}} \pi_t(\mathbf{u} \mid \mathbf{x}) \, Q^{\pi}(\mathbf{x}, \mathbf{u}) \, d\mathbf{u}$$

- Advantage function is just

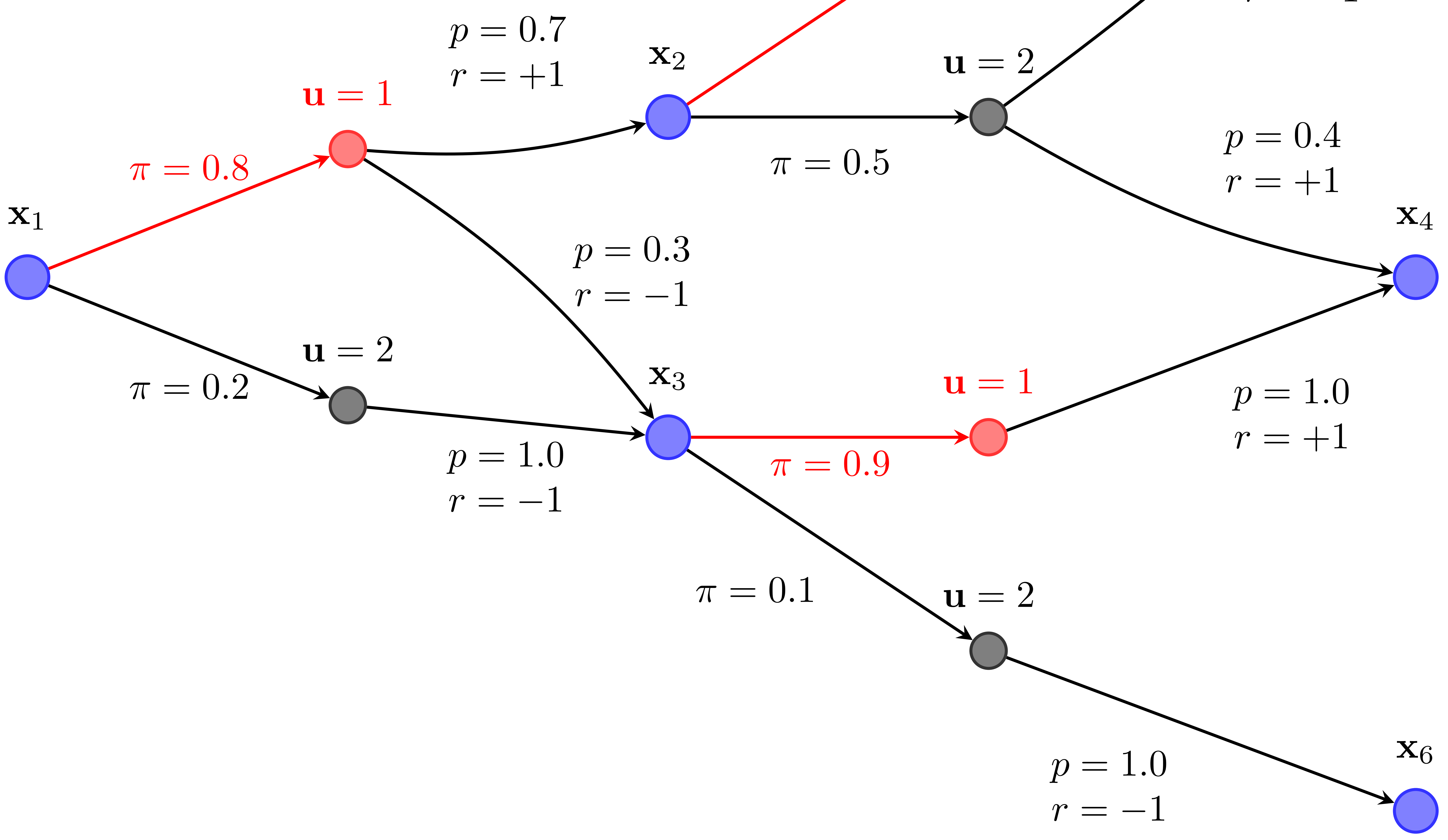
$$A^{\pi}(\mathbf{x}, \mathbf{u}) = Q^{\pi}(\mathbf{x}, \mathbf{u}) - V^{\pi}(\mathbf{x})$$

- If either actions or states are discrete (finite sets), replace integral with sum

$$Q^{\pi}(\mathbf{x},\mathbf{u}) = \int_{\mathbf{x}'} p(\mathbf{x}' \mid \mathbf{x},\mathbf{u})\left[r(\mathbf{x}',\mathbf{x},\mathbf{u}) + \gamma V^{\pi}(\mathbf{x}')\right]\, d\mathbf{x}'$$

$$V^{\pi}(\mathbf{x}) = \int_{\mathbf{u}} \pi_t(\mathbf{u} \mid \mathbf{x})\, Q^{\pi}(\mathbf{x},\mathbf{u})\, d\mathbf{u}$$

$$A^{\pi}(\mathbf{x},\mathbf{u}) = Q^{\pi}(\mathbf{x},\mathbf{u}) - V^{\pi}(\mathbf{x})$$



For this RL MDP, find:

1. $V^{\pi}(\mathbf{x}_2)$
2. $V^{\pi}(\mathbf{x}_3)$
3. $Q^{\pi}(\mathbf{x}_1, \mathbf{u} = \{1,2\})$
4. $V^{\pi}(\mathbf{x}_1)$
5. $A^{\pi}(\mathbf{x}_1, \mathbf{u} = \{1,2\})$

$\gamma = 0.8$

Results

1. $V^{\pi}(\mathbf{x}_2) = 0.5[1.0(-1)] + 0.5[0.6(-1) + 0.4(1)] = -0.6$
2. $V^{\pi}(\mathbf{x}_3 = 0.9[1.0(1)] + 0.1[1.0(-1)]$ = 0.8
3. $Q^{\pi}(\mathbf{x}_1, \mathbf{u} = 1) = 0.7[1 + 0.8V^{\pi}(\mathbf{x}_2)] + 0.3[-1 + 0.8V^{\pi}(\mathbf{x}_3)]$ = 0.256
4. $Q^{\pi}(\mathbf{x}_1, \mathbf{u} = 2) = 1.0[-1 + 0.8V^{\pi}(\mathbf{x}_3)]$ = -0.36
5. $V^{\pi}(\mathbf{x}_1) = 0.8Q^{\pi}(\mathbf{x}_1, \mathbf{u} = 1) + 0.2Q^{\pi}(\mathbf{x}_1, \mathbf{u} = 2)$ = 0.1328
6. $A^{\pi}(\mathbf{x}_1, \mathbf{u} = 1) = Q^{\pi}(\mathbf{x}_1, \mathbf{u} = 1) - V^{\pi}(\mathbf{x}_1)$ = 0.1232
7. $A^{\pi}(\mathbf{x}_1, \mathbf{u} = 2) = Q^{\pi}(\mathbf{x}_1, \mathbf{u} = 2) - V^{\pi}(\mathbf{x}_1)$ =-0.4928

Return

- Probability of a **trajectory**

$$p(\tau \mid \pi) = p(\mathbf{x}_0)\prod_{t=0}^{T} p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t)\pi(\mathbf{u}_t \mid \mathbf{x}_t)$$

- Total discounted trajectory **return**

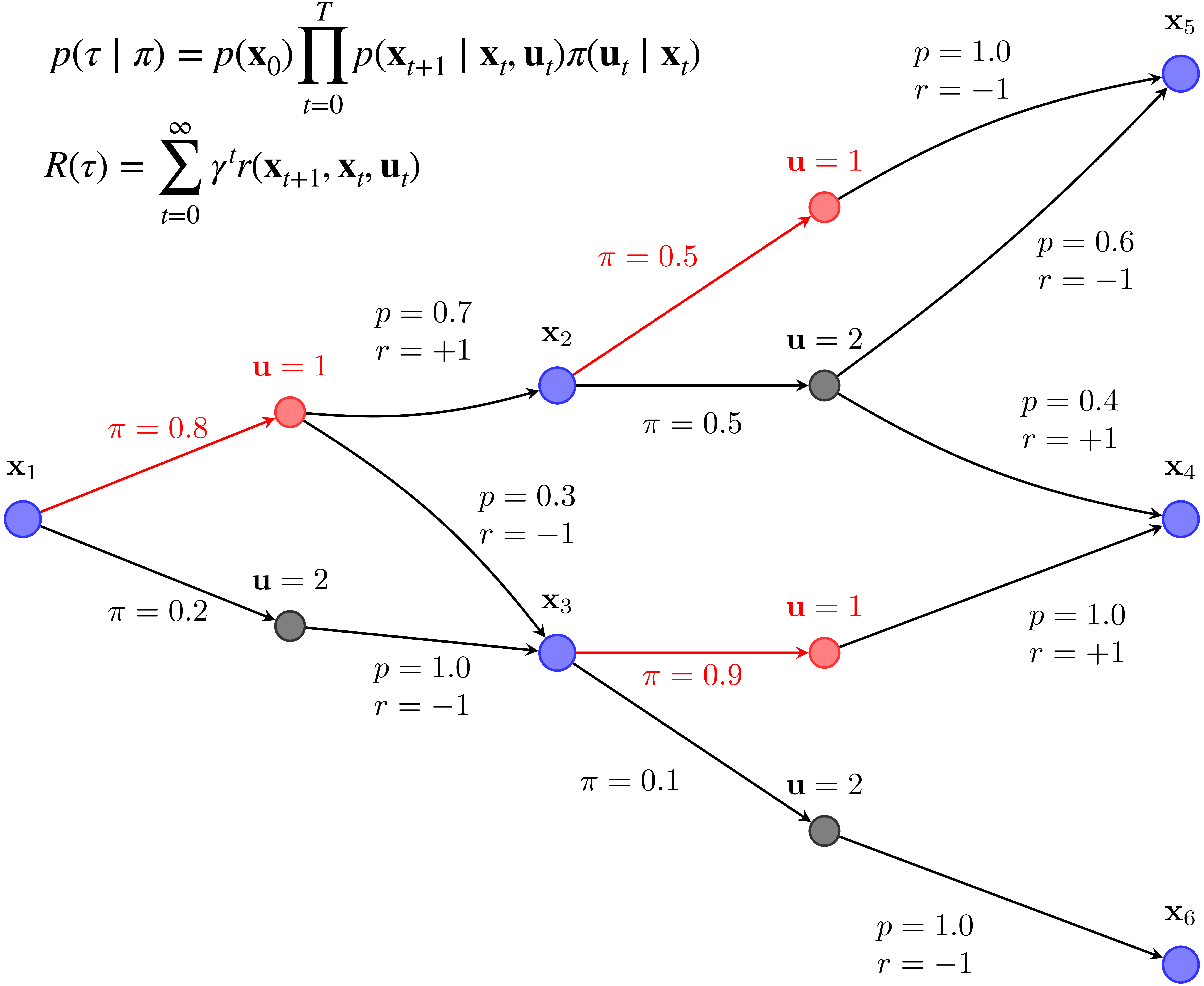
$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(\mathbf{x}_{t+1}, \mathbf{x}_t, \mathbf{u}_t)$$ - this is always a sum because time is discrete

- Expected total return of a **policy**

$$J(\pi) = \int_{\tau} p(\tau \mid \pi) R(\tau)\, d\tau = \mathbb{E}_{\tau \sim \pi}[R(\tau)]$$

$$p(\tau \mid \pi) = p(\mathbf{x}_0) \prod_{t=0}^{T} p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t) \pi(\mathbf{u}_t \mid \mathbf{x}_t)$$

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(\mathbf{x}_{t+1}, \mathbf{x}_t, \mathbf{u}_t)$$



Let's say we have a trajectory τ:

0. $\mathbf{x}_1, \mathbf{u} = 1$
1. $\mathbf{x}_2, \mathbf{u} = 2$
2. $\mathbf{x}_4$

find

- $R(\tau)$
- $p(\tau \mid \pi)$

assume $\gamma = 0.8$

Results

Let's say we have a trajectory τ:

0. $\mathbf{x}_1, \mathbf{u} = 1$
1. $\mathbf{x}_2, \mathbf{u} = 2$
2. $\mathbf{x}_4$

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r(\mathbf{x}_{t+1}, \mathbf{x}_t, \mathbf{u}_t) = r(\mathbf{x}_2, \mathbf{x}_1, \mathbf{u} = 1) + \gamma r(\mathbf{x}_4, \mathbf{x}_2, \mathbf{u} = 2) = 1 + 0.8(+1) = 1.8$$

We have just 1 starting state -> its probability is always 1

$$p(\tau \mid \pi) = p(\mathbf{x}_0) \prod_{t=0}^{T} p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t) \pi(\mathbf{u}_t \mid \mathbf{x}_t) =$$

$$= \prod_{t=0}^{T} p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{u}_t) \pi(\mathbf{u}_t \mid \mathbf{x}_t) = (0.8 \cdot 0.7) \cdot (0.5 \cdot 0.4) = 0.112$$

Optimal policy and functions

- Optimal policy is defined as

$$\pi^* = \arg\max_{\pi} J(\pi)$$

- Optimal policy has the optimal value function

$$V^*(\mathbf{x}) = \max_{\mathbf{u}} \int_{\mathbf{x}'} p(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}) \left[r(\mathbf{x}', \mathbf{x}, \mathbf{u}) + \gamma V^*(\mathbf{x}') \right] \, d\mathbf{x}' = \max_{\mathbf{u}} \; Q^*(\mathbf{x}, \mathbf{u})$$

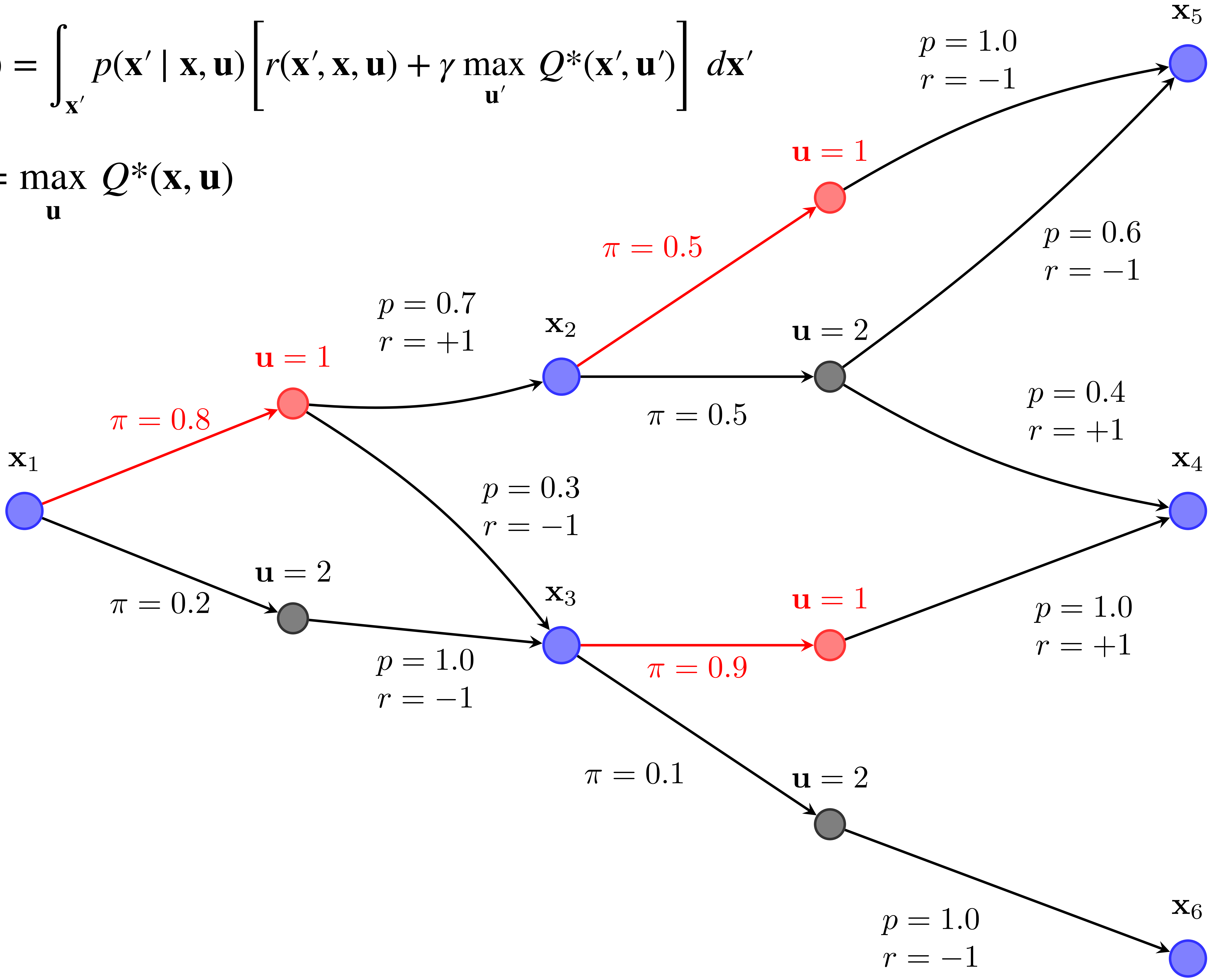
- Also

$$Q^*(\mathbf{x}, \mathbf{u}) = \int_{\mathbf{x}'} p(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}) \left[r(\mathbf{x}', \mathbf{x}, \mathbf{u}) + \gamma \max_{\mathbf{u}'} \; Q^*(\mathbf{x}', \mathbf{u}') \right] \, d\mathbf{x}'$$

- It is greedy with respect to the rewards

$$Q^*(\mathbf{x},\mathbf{u}) = \int_{\mathbf{x}'} p(\mathbf{x}' \mid \mathbf{x},\mathbf{u}) \left[r(\mathbf{x}',\mathbf{x},\mathbf{u}) + \gamma \max_{\mathbf{u}'} Q^*(\mathbf{x}',\mathbf{u}') \right] d\mathbf{x}'$$

$$V^*(\mathbf{x}) = \max_{\mathbf{u}} Q^*(\mathbf{x},\mathbf{u})$$



For this RL MDP, find:

1. Q^{π^*}
2. V^{π^*}
3. π^*

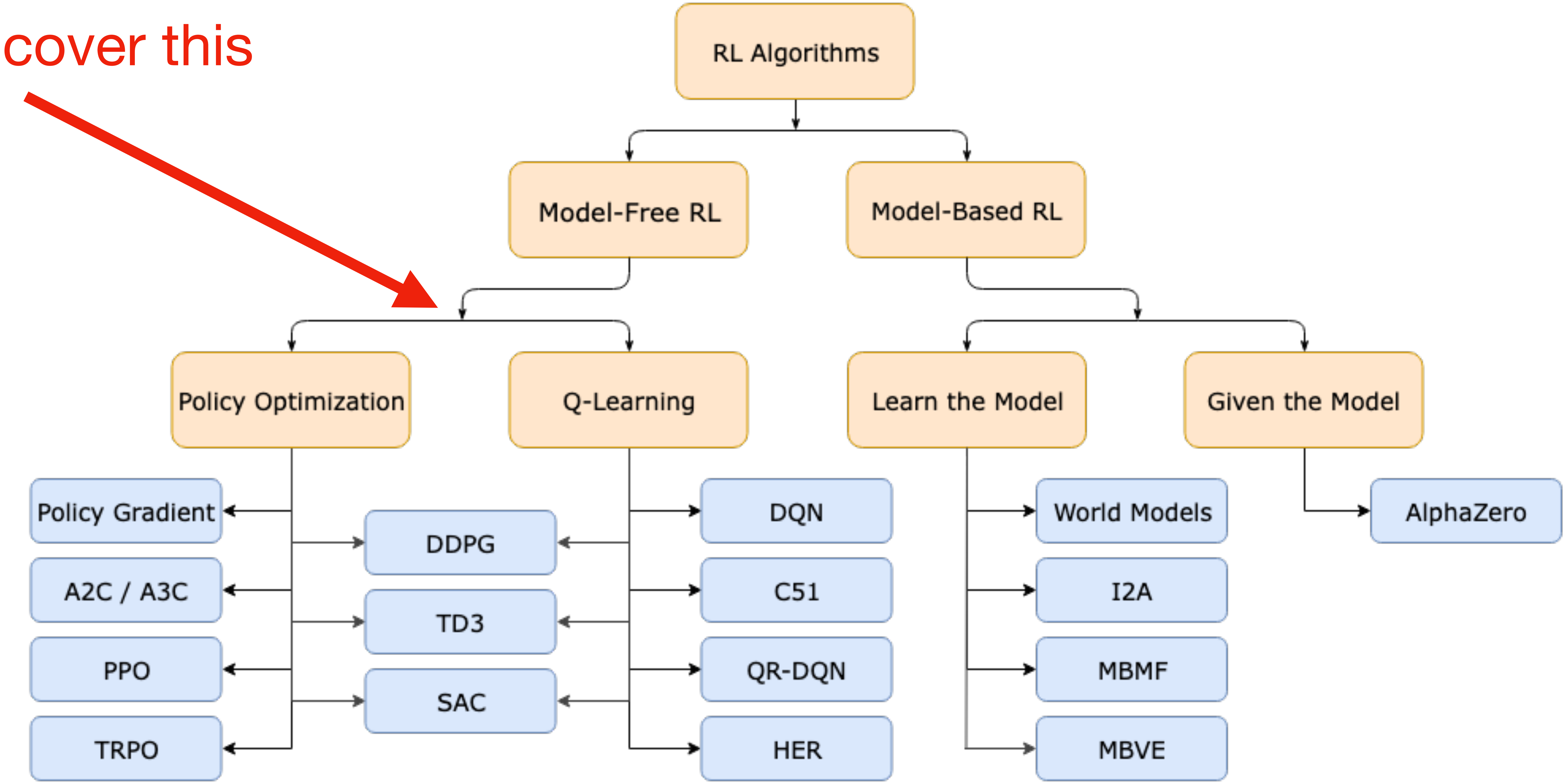
assume $\gamma = 0.8$

Results

1. $Q^{\pi^*}(\mathbf{x}_2, \mathbf{u} = 1) = 1.0 \cdot (-1 + 0.8 \cdot 0) = -1$
2. $Q^{\pi^*}(\mathbf{x}_2, \mathbf{u} = 2) = 0.4 \cdot (1 + 0.8 \cdot 0) + 0.6 \cdot (-1 + 0.8 \cdot 0) = -0.2$
3. $Q^{\pi^*}(\mathbf{x}_3, \mathbf{u} = 1) = 1.0 \cdot (1 + 0.8 \cdot 0) = 1$
4. $Q^{\pi^*}(\mathbf{x}_3, \mathbf{u} = 2) = 1.0 \cdot (-1 + 0.8 \cdot 0) = -1$
5. $Q^{\pi^*}(\mathbf{x}_1, \mathbf{u} = 1) = 0.7 \cdot (1 + 0.8 \cdot (-0.2)) + 0.3 \cdot (-1 + 0.8 \cdot 1) = 0.528$
6. $Q^{\pi^*}(\mathbf{x}_1, \mathbf{u} = 2) = 1.0 \cdot (-1 + 0.8 \cdot 1) = -0.2$
7. $V^{\pi^*}(\mathbf{x}_2) = -0.2$
8. $V^{\pi^*}(\mathbf{x}_3) = 1$
9. $V^{\pi^*}(\mathbf{x}_1) = 0.528$
10. $\pi^*(\mathbf{x}_i) = \begin{cases} \mathbf{u} = 1 & \text{if } i = 1 \\ \mathbf{u} = 2 & \text{if } i = 2 \\ \mathbf{u} = 1 & \text{if } i = 3 \end{cases}$

Intro into deep RL

We will cover this



https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html#citations-below

On-policy, off-policy

2 main ways to learn the policy π

On-policy

- Direct learning
- Uses π for exploration and learning
- Directly maximize J

Off-policy

- The learned policy is different from the one used to gather data about the environment
- We construct π^* by learning Q^*
- Minimize loss derived from Bellman
- More efficient

Policy gradient

- Expected total return of a policy parameterized by θ

$$J(\pi_\theta) = \int_\tau p(\tau \mid \pi_\theta) R(\tau)\, d\tau$$

- **We want to maximize it**

$$\max_\theta J(\pi_\theta)$$

- Solve easily with gradient ascent -> we must compute the gradient

Policy gradient

Helpers
$$\frac{\partial \log p(\tau|\pi_\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\left[\log p(\mathbf{x}_0) + \sum_{t=0}^{T}\log(p(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{u}_t)) + \sum_{t=0}^{T}\log(\pi_\theta(\mathbf{u}_t|\mathbf{x_t}))\right]$$

$$\frac{\partial \log p(x\,|\,\theta)}{\partial\theta} = \frac{1}{p(x\,|\,\theta)}\frac{\partial p(x\,|\,\theta)}{\partial\theta} \Rightarrow \frac{\partial p(x\,|\,\theta)}{\partial\theta} = p(x\,|\,\theta)\frac{\partial \log p(x\,|\,\theta)}{\partial\theta}$$

$$\frac{\partial J(\pi_\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\int_\tau p(\tau\mid\pi_\theta)R(\tau)\,d\tau = \ldots$$

Policy gradient

Helpers

$$\frac{\partial \log p(\tau|\pi_\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\left[\log p(\mathbf{x}_0) + \sum_{t=0}^{T}\log(p(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{u}_t)) + \sum_{t=0}^{T}\log(\pi_\theta(\mathbf{u}_t|\mathbf{x}_t))\right]$$

$$\frac{\partial \log p(x\,|\,\theta)}{\partial\theta} = \frac{1}{p(x\,|\,\theta)}\frac{\partial p(x\,|\,\theta)}{\partial\theta} \Rightarrow \frac{\partial p(x\,|\,\theta)}{\partial\theta} = p(x\,|\,\theta)\frac{\partial \log p(x\,|\,\theta)}{\partial\theta}$$

$$\frac{\partial J(\pi_\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\int_\tau p(\tau\mid\pi_\theta)R(\tau)\,d\tau = \int_\tau \frac{\partial p(\tau\mid\pi_\theta)}{\partial\theta}R(\tau)\,d\tau =$$

$$= \int_\tau p(\tau\mid\pi_\theta)\frac{\partial \log p(\tau\mid\pi_\theta)}{\partial\theta}R(\tau)\,d\tau = \boxed{\mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\frac{\partial \log \pi_\theta(\mathbf{u}_t\mid\mathbf{x}_t)}{\partial\theta}R(\tau)\right]}$$

Policy gradient

We derived the policy gradient explicitly as the expected value of a random variable

-> we can approximate it with **sample mean** ($\bar{x} = \frac{1}{N}\sum_i x_i$)

the expression that we actually compute in code then becomes

$$\frac{\partial J(\pi_\theta)}{\partial \theta} \approx \frac{1}{|\mathcal{T}|} \sum_{\tau \in \mathcal{T}} \sum_{t=0}^{T} \frac{\partial \log \pi_\theta(\mathbf{u}_t \mid \mathbf{x}_t)}{\partial \theta} R(\tau)$$

where $\mathcal{T}$ is a set of gathered trajectories

Rewards-to-go

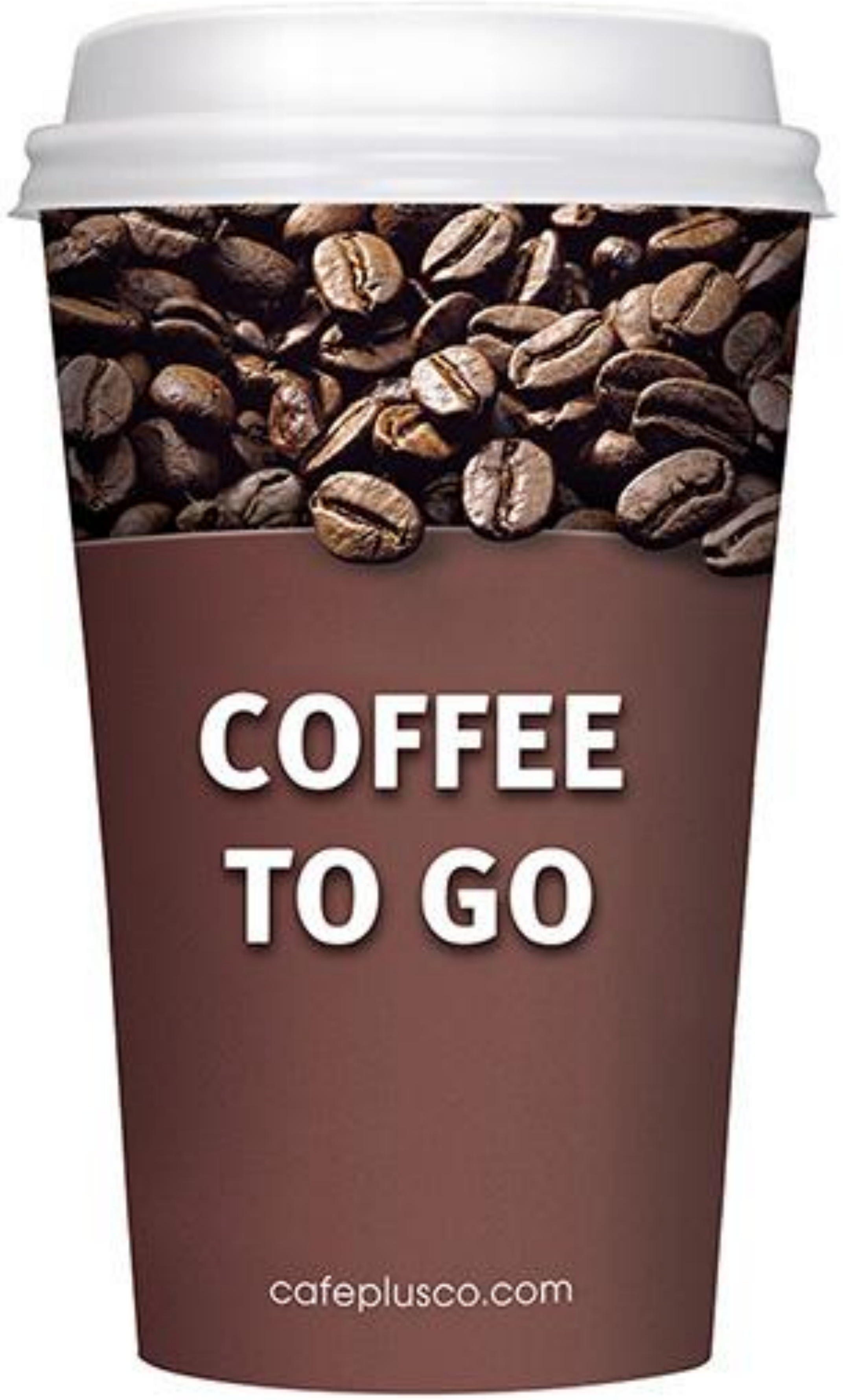
- In policy gradient we multiply the log-prob-derivative with total return
- It does not make sense to include the rewards obtained before executing an action
- We go from

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\frac{\partial \log \pi_\theta(\mathbf{u}_t \mid \mathbf{x}_t)}{\partial\theta}R(\tau)\right],\ R(\tau)=\sum_{t=0}^{\infty}\gamma^t r(\mathbf{x}_{t+1},\mathbf{x}_t,\mathbf{u}_t)$$

- to

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\frac{\partial \log \pi_\theta(\mathbf{u}_t \mid \mathbf{x}_t)}{\partial\theta}\sum_{t'=t}^{T}\gamma^{t'} r(\mathbf{x}_{t'+1},\mathbf{x}'_t,\mathbf{u}'_t)\right]$$

- this can be proven exactly with EGLP lemma
- we lower the amount of noise in our sample means (justified by EGLP lemma)



Actor and critic

- In the policy gradient, the reward-to-go (partial return) can be replaced by different indicators that don't change the expected value
- We can replace it by the on-policy value or advantage function (Q^π, A^π)
- This function has to be somehow computed next to the policy
- We train 2 deep nets
- **Actor:** the policy π itself
- **Critic:** a function approximating Q^π, V^π or A^π

Advantage actor critic (A2C)

- **A synchronous deep RL algorithm**
- **Actor:** the policy π itself
- **Critic:** a function approximating V^{π}
- We minimize policy gradient with advantage indicator:

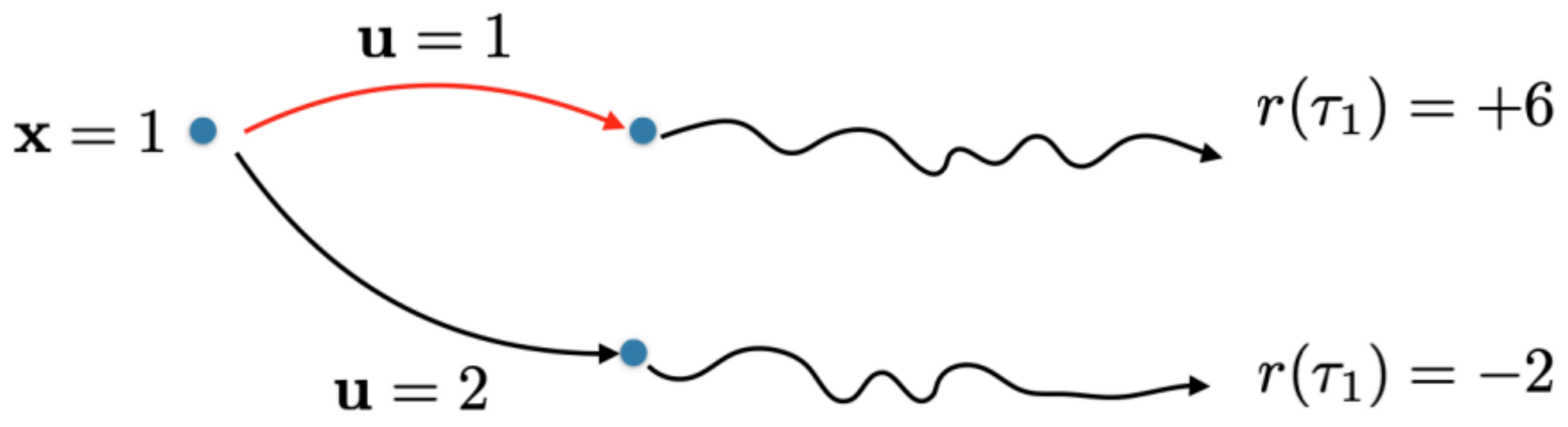
$$\mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} \frac{\partial \log \pi_\theta(\mathbf{u}_t \mid \mathbf{x}_t)}{\partial \theta} A^{\pi_\theta}(\mathbf{x}_t, \mathbf{u}_t) \right]$$

- Recall $A^{\pi}(\mathbf{x}, \mathbf{u}) = Q^{\pi}(\mathbf{x}, \mathbf{u}) - V^{\pi}(\mathbf{x})$

Exam problem

$$\pi_\theta(\mathbf{u}|\mathbf{x}) = \begin{cases} \sigma(\theta\mathbf{x}+1) & \text{if } \mathbf{u}=1 \\ 1-\sigma(\theta\mathbf{x}+1) & \text{if } \mathbf{u}=2 \end{cases}$$

$\theta = -1$



Find

$$A(\mathbf{u}=1,\mathbf{x}=1) = A^\pi(\mathbf{u}=1,\mathbf{x}=1) = Q^\pi(\mathbf{u}=1,\mathbf{x}=1) - V^\pi(\mathbf{x}=1) = 6 - (0.5\cdot 6 + 0.5\cdot(-2)) = 4$$

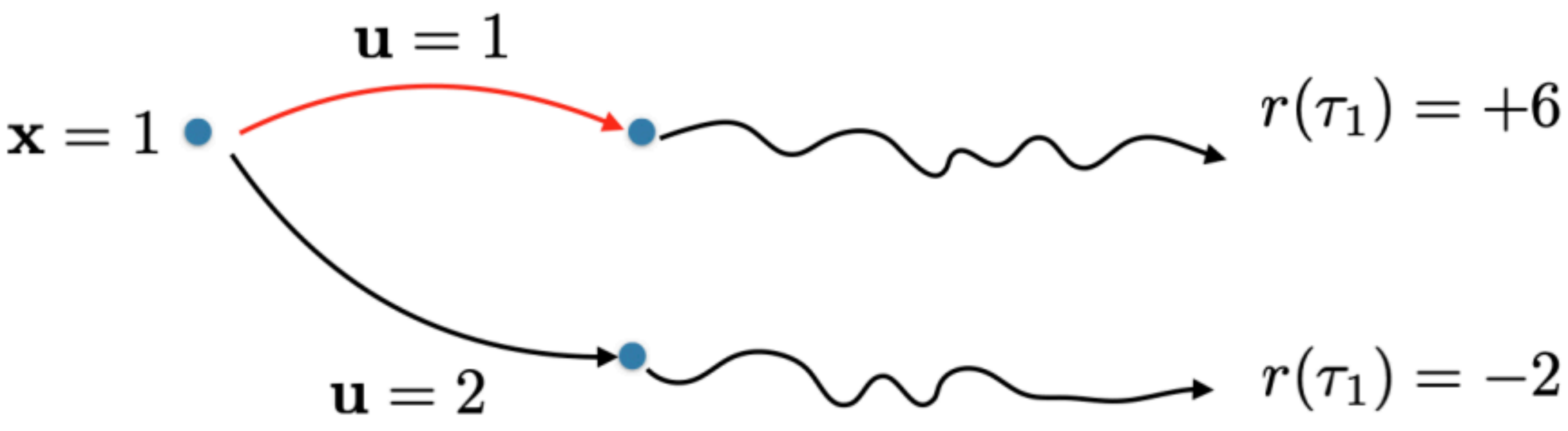
$$A(\mathbf{u}=2,\mathbf{x}=1) = A^\pi(\mathbf{u}=2,\mathbf{x}=1) = Q^\pi(\mathbf{u}=2,\mathbf{x}=1) - V^\pi(\mathbf{x}=1) = (-2) - (0.5\cdot 6 + 0.5\cdot(-2)) = -4$$

$$\tau = [\mathbf{x}_1 = 1, \mathbf{u}_1 = 1, \ldots].$$

$$\left.\frac{\partial \log \pi_\theta(\mathbf{u}|\mathbf{x})}{\partial\theta}\right|_{\substack{\mathbf{x}=\mathbf{x}_1\\ \mathbf{u}=\mathbf{u}_1}} \cdot A(\mathbf{u}=\mathbf{u}_1,\mathbf{x}=\mathbf{x}_1) =$$

Exam problem

$$\pi_\theta(\mathbf{u}|\mathbf{x}) = \begin{cases} \sigma(\theta\mathbf{x}+1) & \text{if } \mathbf{u}=1 \\ 1-\sigma(\theta\mathbf{x}+1) & \text{if } \mathbf{u}=2 \end{cases}$$



$\theta = -1$

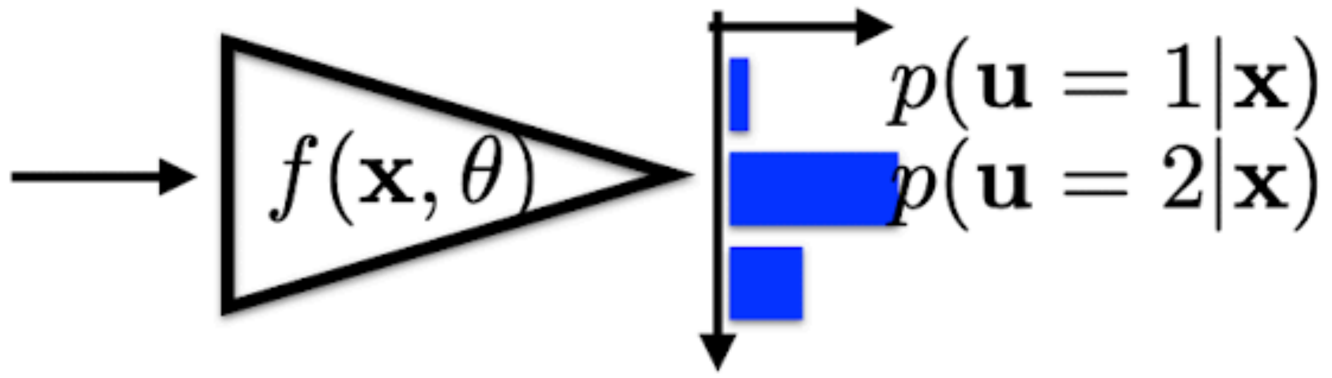
$$\tau = [\mathbf{x}_1 = 1, \mathbf{u}_1 = 1, \ldots].$$

$$\left.\frac{\partial \log \pi_\theta(\mathbf{u}|\mathbf{x})}{\partial \theta}\right|_{\substack{\mathbf{x}=\mathbf{x}_1 \\ \mathbf{u}=\mathbf{u}_1}} \cdot A(\mathbf{u}=\mathbf{u}_1, \mathbf{x}=\mathbf{x}_1) =$$

$$= (1-\sigma(\theta+1)) \cdot A^\pi(\mathbf{u}=1, \mathbf{x}=1) = (1-\sigma(\theta+1)) \cdot 4 = (1-\sigma(-1+1)) \cdot 4 = 2$$

$$\frac{\partial \log \pi_\theta(\mathbf{u}=1\,|\,\mathbf{x}=1)}{\partial \theta} = \frac{\partial \log \sigma(\theta+1)}{\partial \theta} = \frac{1}{\sigma(\theta+1)} \cdot \sigma(\theta+1)(1-\sigma(\theta+1)) = 1-\sigma(\theta+1)$$

Advantage actor critic (A2C)



1. Initialize policy $\pi_\theta(\mathbf{u}|\mathbf{x})$, $V_\omega(\mathbf{x})$
2. Collect trajectories τ with policy $\pi_\theta(\mathbf{u}|\mathbf{x})$
3. **Critic:** Update value function to predict observed values: $V_\omega(\mathbf{x}) \leftarrow r + \gamma V_\omega(\mathbf{x}')$

   $$\mathcal{L}_{\text{critic}}(\omega) = \Big(\underbrace{r + \gamma V_\omega(\mathbf{x}') - V_\omega(\mathbf{x})}_{A_\omega}\Big)^2$$

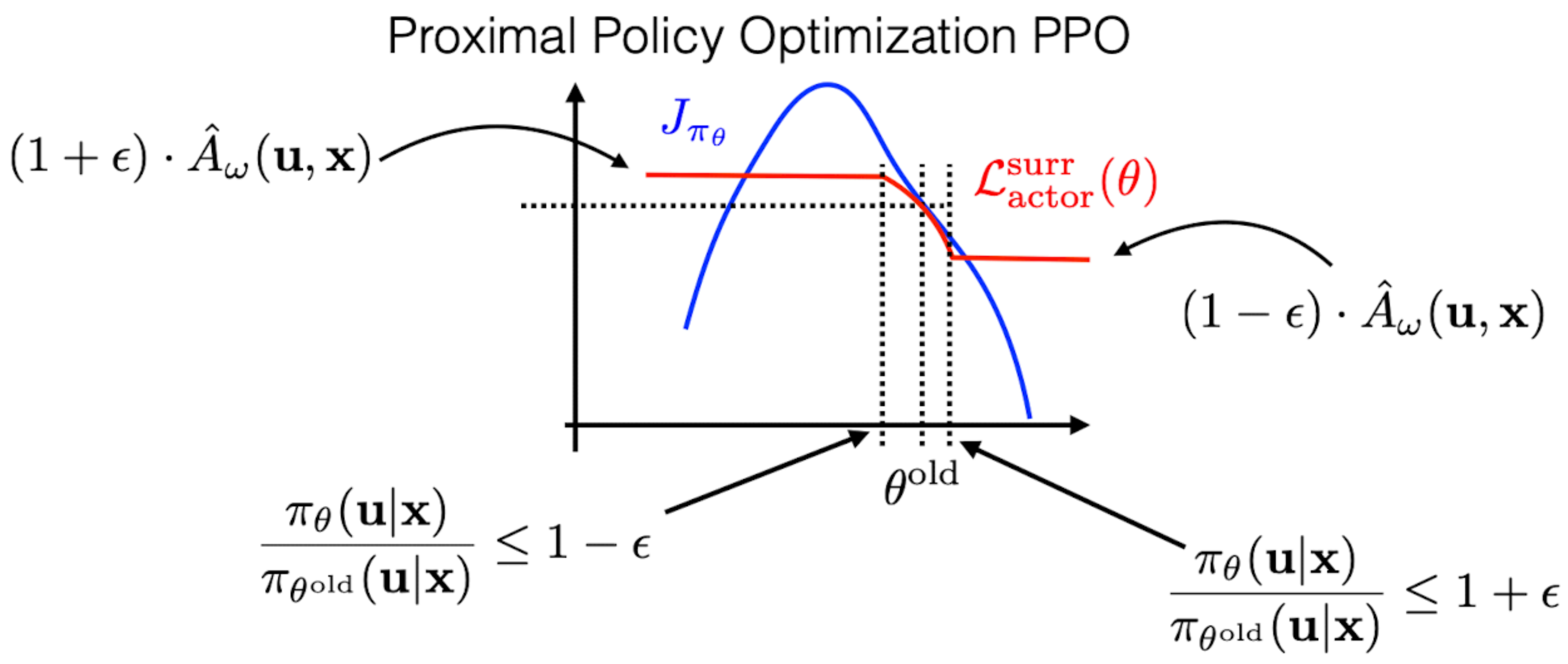
4. **Actor:** Update policy by policy gradient:

   $$\mathcal{L}_{\text{actor}}(\theta) = \sum_{(\mathbf{u},\mathbf{x},\mathbf{x}')\in\tau} \log \pi_\theta(\mathbf{u}|\mathbf{x}) \cdot \underbrace{\Big(r + \gamma V_\omega(\mathbf{x}') - V_\omega(\mathbf{x})\Big)}_{A_\omega = Q - V}$$

   $$\theta := \theta + \alpha \frac{\partial \mathcal{L}_{\text{actor}}(\theta)}{\partial \theta} \qquad \omega := \omega + \beta \frac{\partial \mathcal{L}_{\text{critic}}(\omega)}{\partial \omega}$$

5. Repeat from 2

Proximal policy optimization (PPO)



$$\arg\max_\theta \;\; \mathbb{E}_{\tau\sim\pi_\theta}\left[\min\left\{\frac{\pi_\theta(\mathbf{u}|\mathbf{x})}{\pi_{\theta^{\text{old}}}(\mathbf{u}|\mathbf{x})}\cdot\hat{A}_\omega(\mathbf{u},\mathbf{x}),\;\; \text{clip}(\frac{\pi_\theta(\mathbf{u}|\mathbf{x})}{\pi_{\theta^{\text{old}}}(\mathbf{u}|\mathbf{x})},1-\epsilon,1+\epsilon)\cdot\hat{A}_\omega(\mathbf{u},\mathbf{x})\right\}\right]$$

tl;dr: **Stabilize learning** by limiting the change in probability by a hard threshold -> limit gradients by thresholding in the **action probability space. Helps avoid catastrophic forgetting.**