Reinforcement Learning

CSE 473: Introduction to Artificial Intelligence



Megan Hazen(mh75@cs)
cs.uw.edu/473
acknowledgements

Welcome!
What's something you had to learn by doing?

Administrivia

ANNOUNCEMENTS

- Test 1 Friday
  - **Bring pen/pencil, ID**
  - **8.5x11" note sheet**

ASSIGNMENTS

- Project 1 due **Tonight**
- Homework 2 due next week (10.30)

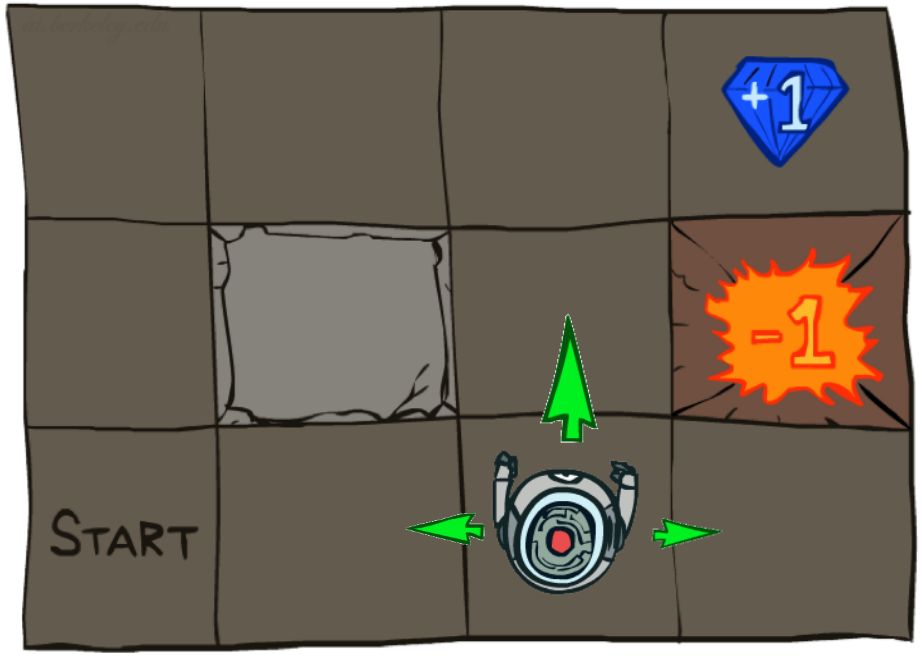


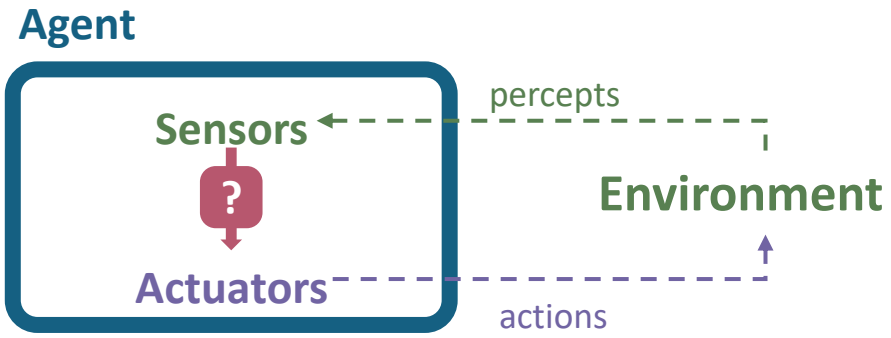
Reinforcement Learning Problems

FOUNDATIONS

- Still a Markov Decision Process, with:
  - States $s \in S$
  - Actions $a \in A$
  - Transition model $T(s, a, s')$

    $$T(s, a, s') = P(s'|s, a)$$
  - Reward function $R(s, a, s')$
  - Start state s_0
  - Utility function $U(s)$
- But:
  - $\boldsymbol{T}$ and $\boldsymbol{R}$ are unknown.
  - Now must explore to learn best policy

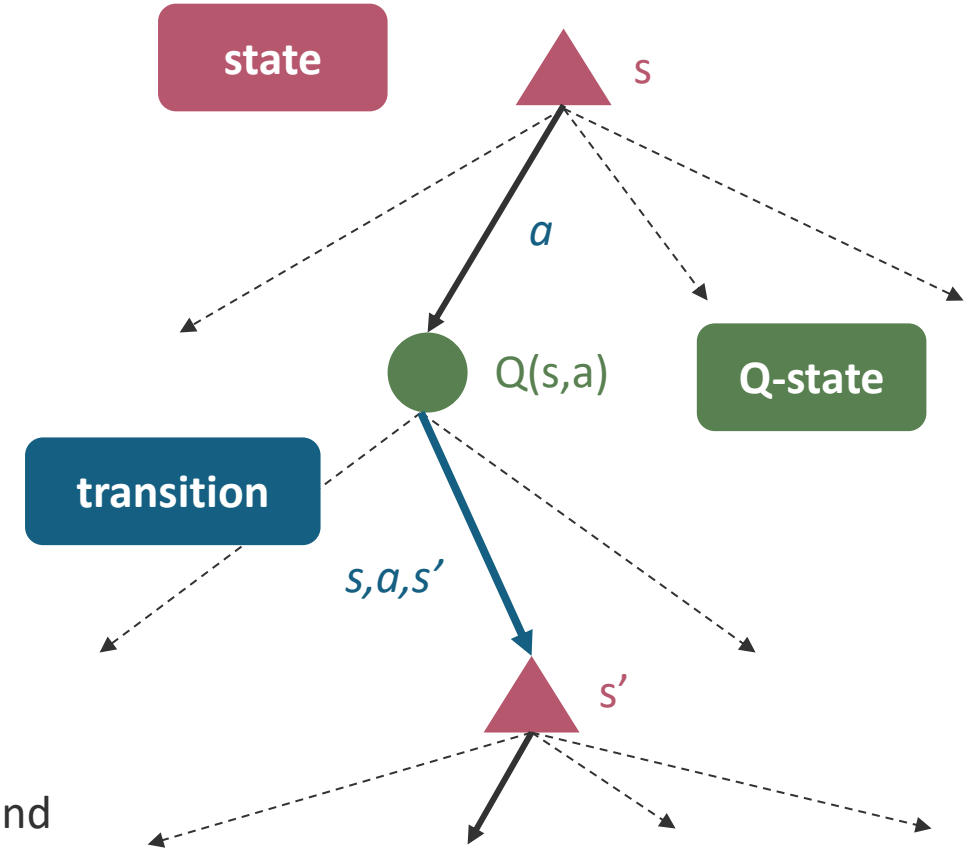


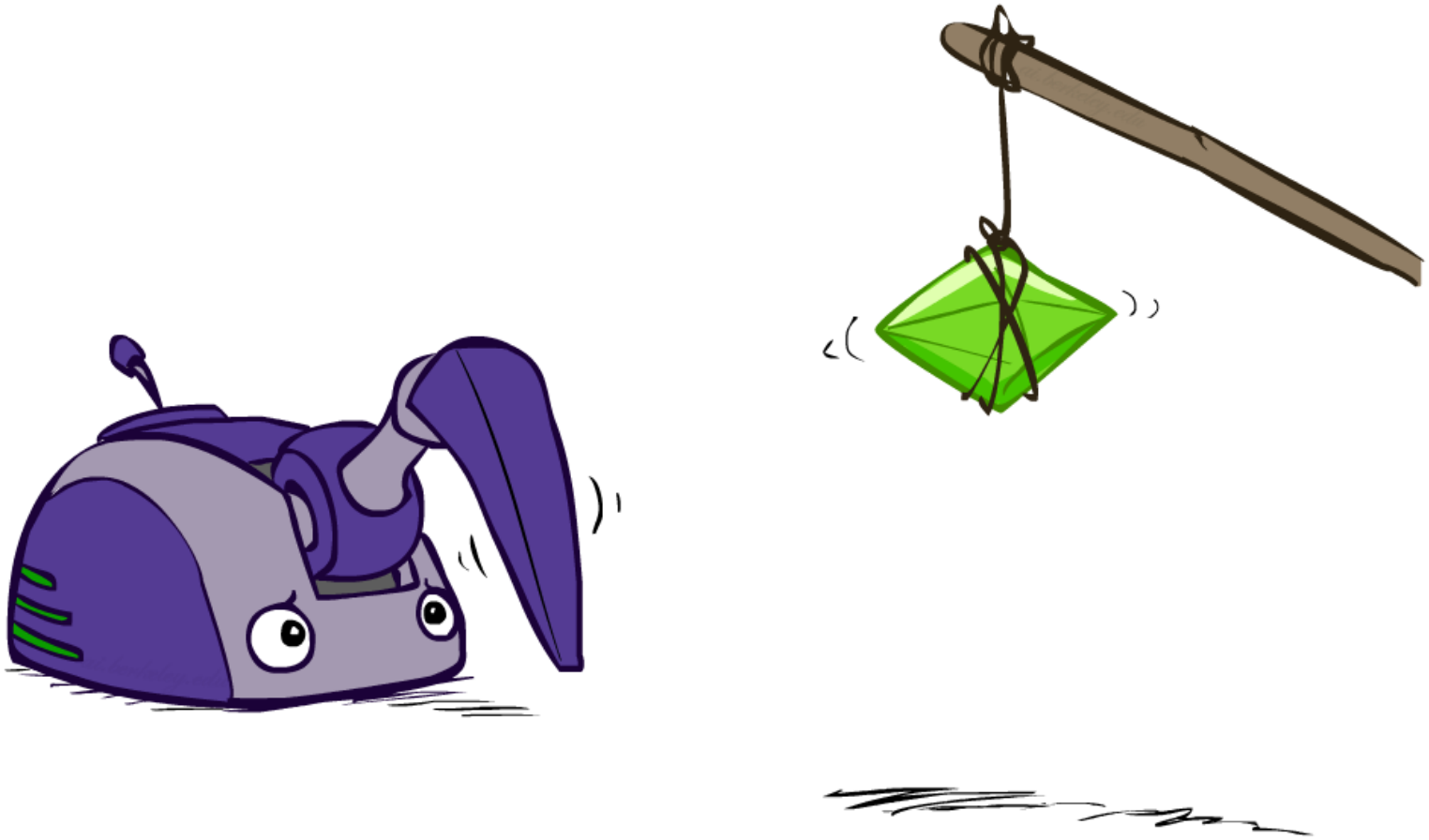


Recap: MDP Search Trees

FOUNDATIONS

- Expected utility (value) of policy $\boldsymbol{\pi}$ at $\boldsymbol{s_0}$:
  - $U^{\pi}(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t), s'_t)]$
- Optimal Policy $\boldsymbol{\pi^*}$
  - $\pi^* = \text{argmax}_{\pi} U^{\pi}(s)$
- Value of a State $\boldsymbol{U^*(s)}$
  - Expected utility starting at state $\boldsymbol{s}$ and acting optimally
- Q-Value $\boldsymbol{Q(s,a)}$
  - $Q^*(s,a)$ is the expected utility of taking action $\boldsymbol{a}$ from state $\boldsymbol{s}$ and acting optimally thereafter
  - $U^*(s) = \max_a Q^*(s,a)$

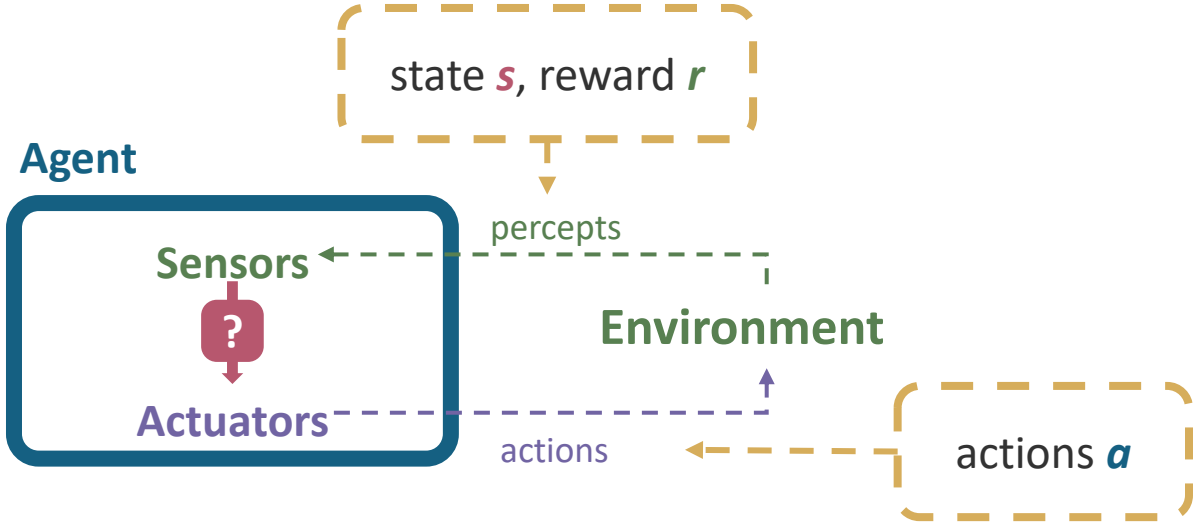




Reinforcement Learning: Learning by Doing

Reinforcement Learning Approaches

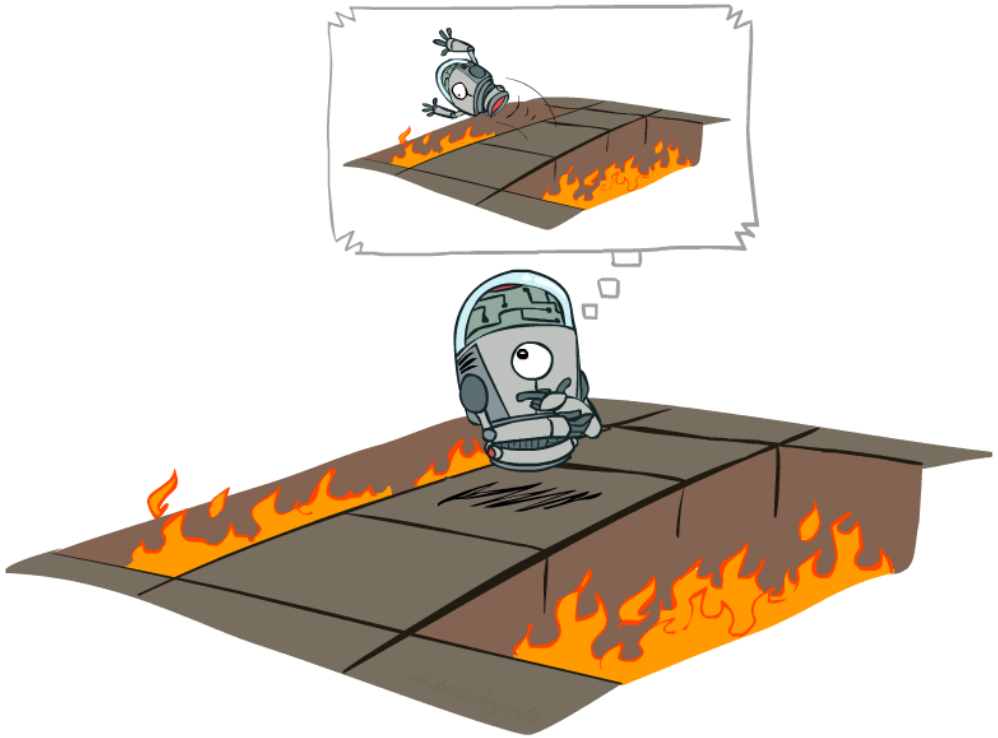
FOUNDATIONS



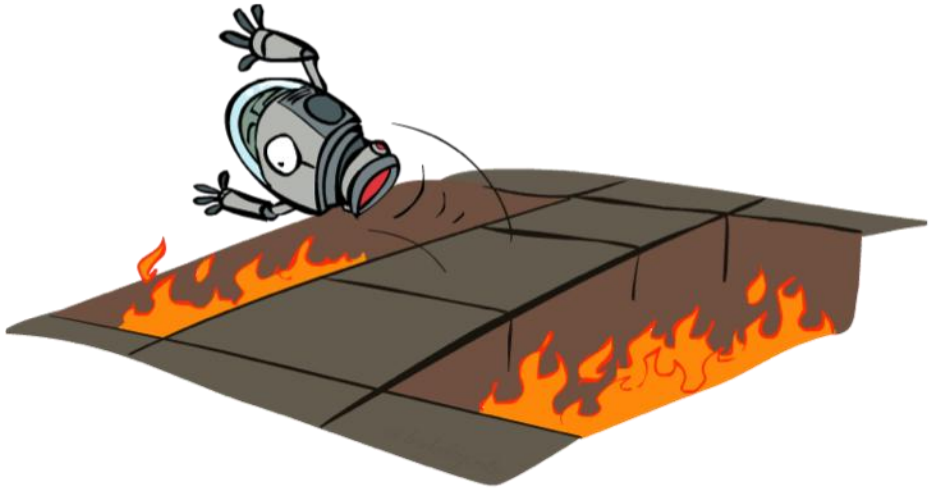
- **Exploration**: try unknown actions to gain information
- **Exploitation**: eventually, you have to use what you know
- **Sampling**: repeat many times to get good estimates
- **Generalization**: what you learn in one state may apply to others, too

Offline vs. Online Learning

CONTEXT



Offline Solution (MDPs)

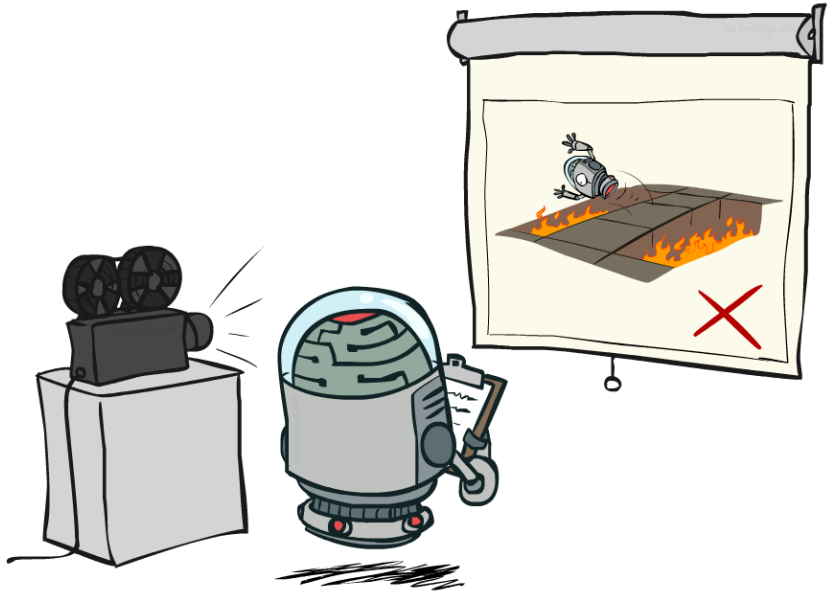


Online Learning (RL)

Passive Reinforcement Learning

FOUNDATIONS

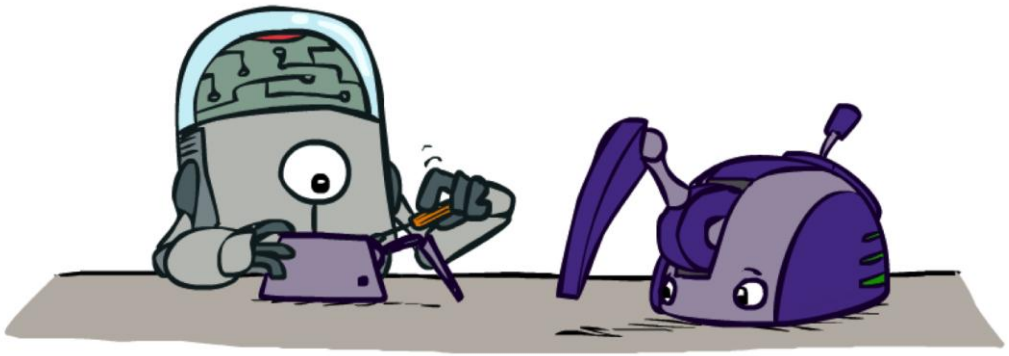
- Simplified Task: **Policy Evaluation**
  - Input: fixed policy $\pi(s)$
  - Don't know $T(s, a, s') = P(s'|s, a)$
  - Don't know $R(s, a, s)$
  - Goal: *learn the state values*
- In this case:
  - Learner is "along for the ride"
  - No choice about what actions to take
  - Just execute the policy and learn from experience
  - But it's *not* offline planning! Agent actually takes actions in the world

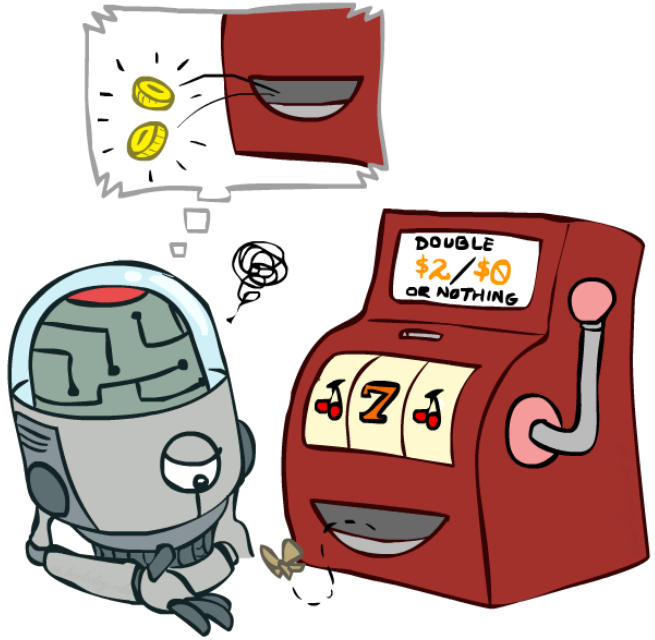


Questions?

Passive Reinforcement Learning – How?

CONTEXT





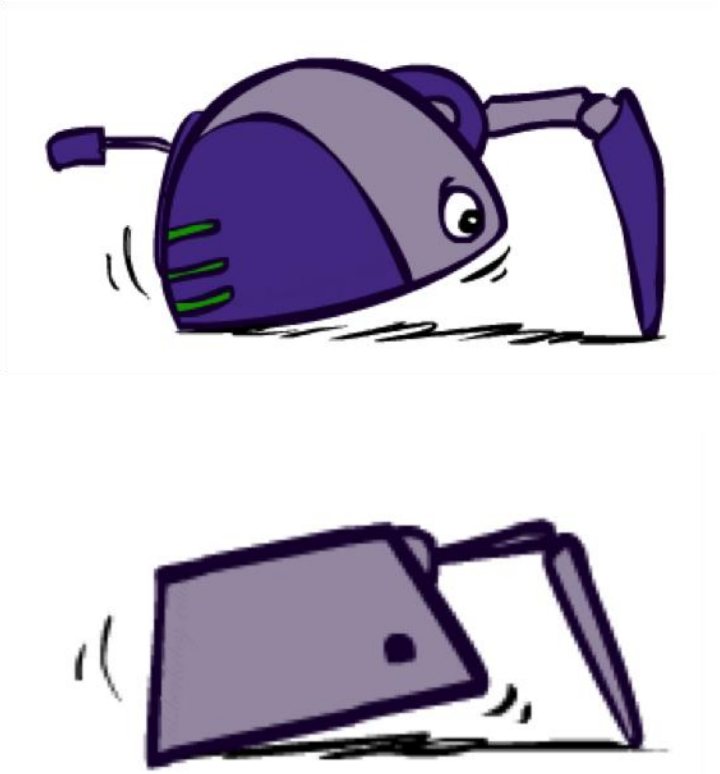
Model-Based Learning

Model-Free Learning

Model-Based Learning

FOUNDATIONS

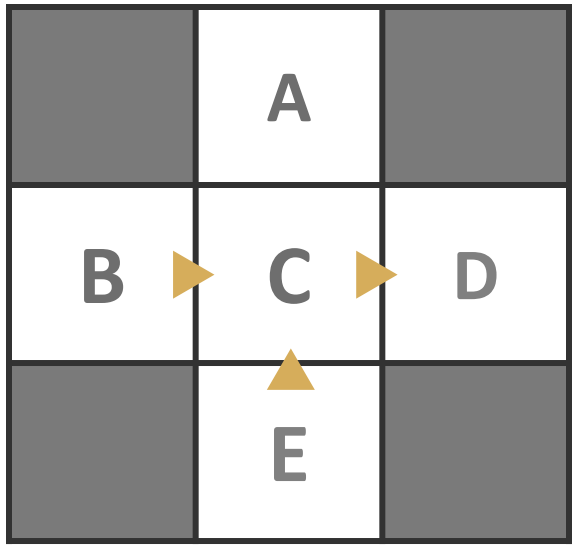
- Idea:
  - Learn approximate model based on experiences
  - Solve for values *as if* the learned model were correct
- **Step 1**: Learn Empirical MDP Model
  - Count outcomes s' for each s, a
  - Normalize to give estimate of $\widehat{T}(s, a, s')$
  - Discover each $\widehat{R}(s, a, s')$ when we experience (s, a, s')
- **Step 2**: Solve Learned MDP
  - For example, using value iteration



Model-Based Learning Example

EXAMPLE

Input Policy π



Assume: $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Learned Model

$\hat{T}(s, a, s')$

T(B, east, C) = 1.00
T(C, east, D) = 0.75
T(C, east, A) = 0.25
T(E, north, C) … …

$\hat{R}(s, a, s')$

R(B, east, C) = -1
R(C, east, D) = -1
R(D, exit, x) = +10
R(E, north, C) … …

Model-Based Learning: Pros and Cons

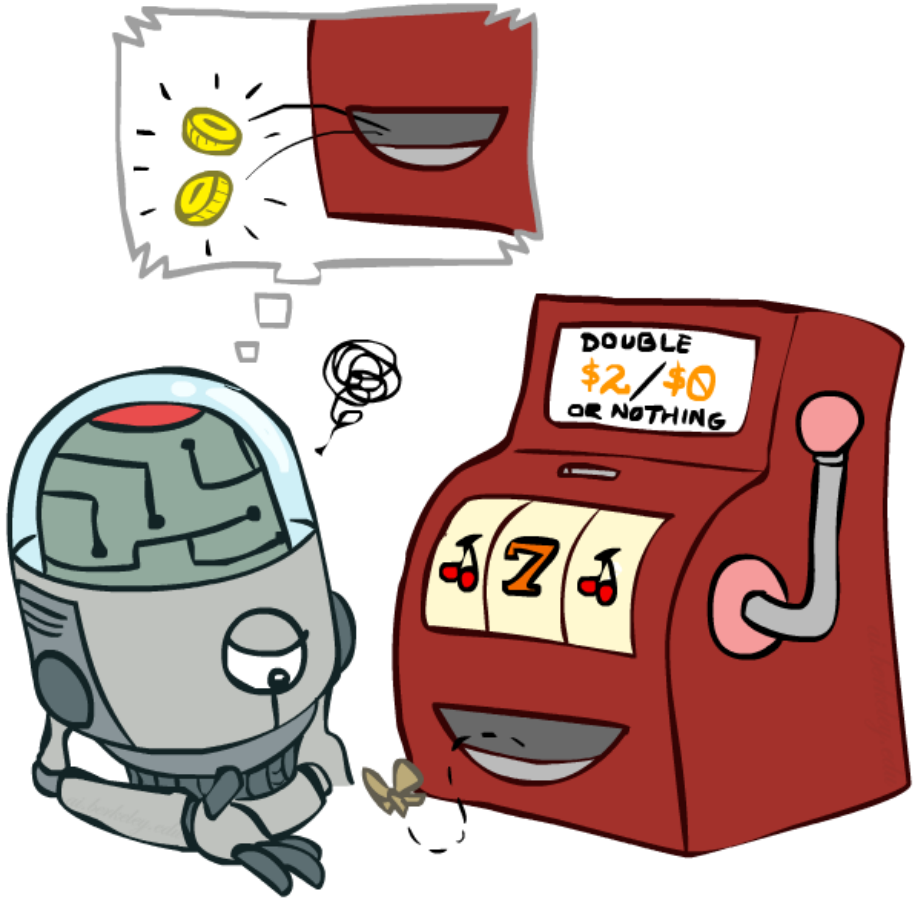
CONTEXT

- **Advantage**
  - Makes efficient use of experiences
- **Disadvantage**
  - May not scale to large state spaces
    - Learns to model one state-action pair at a time (this is fixable)
    - Cannot solve MDP for very large $|S|$

Questions?

Model-Free Learning

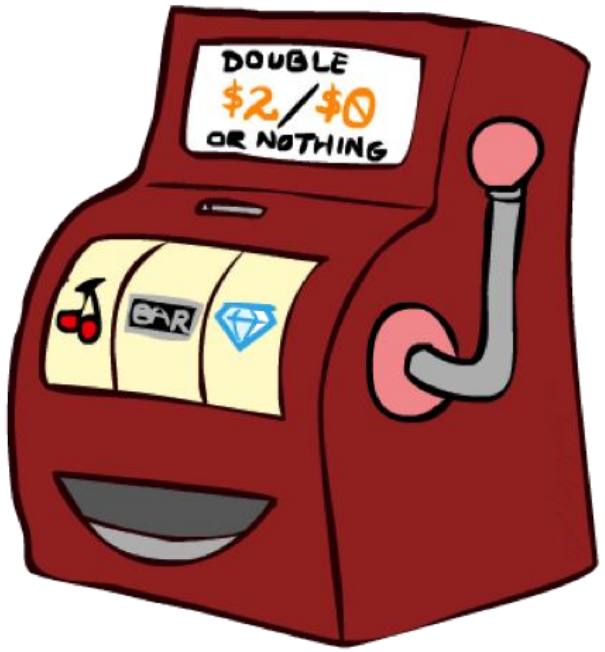
CONTEXT



Direct Evaluation

FOUNDATIONS

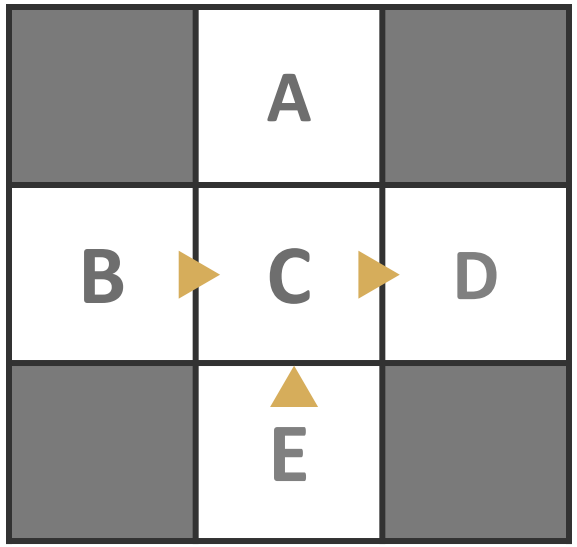
- **Goal**: Compute values for each state under π
- Idea: Average together observed sample values
  - Act according to policy π
  - Every time agent visits a state, write down the actual discounted sum of rewards
    - Must go backwards through episodes to do this
  - Average samples



Direct Evaluation Example (1)

EXAMPLE

Input Policy π



Assume: $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

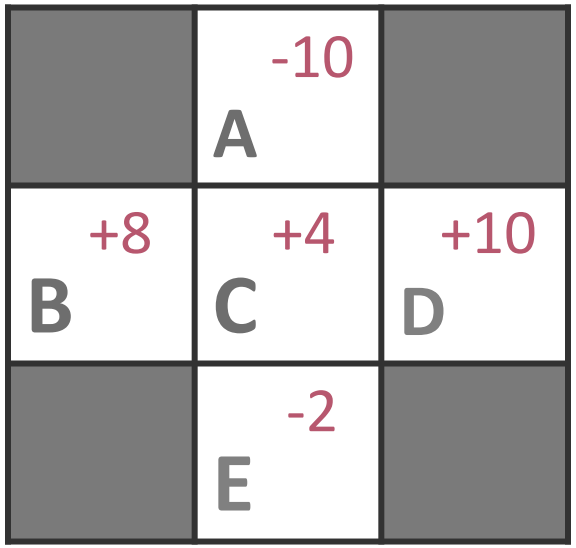
Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Output Values

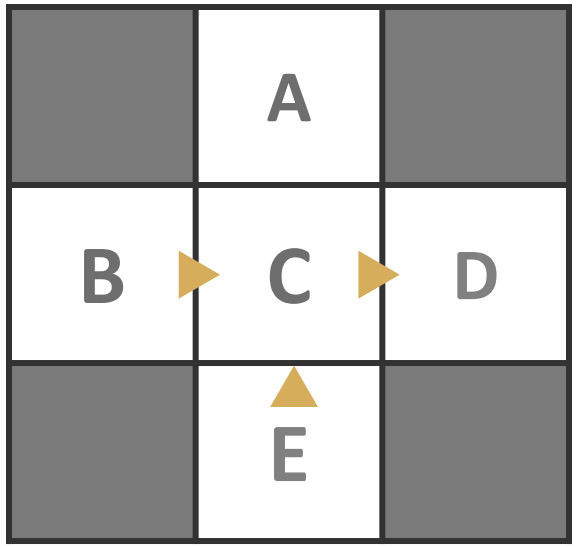


$U^{\pi}(s)$ = average(returns(s))

Direct Evaluation Example (2)

EXAMPLE

Input Policy π



Assume: $\gamma = 1$

Observed Episodes (Training)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

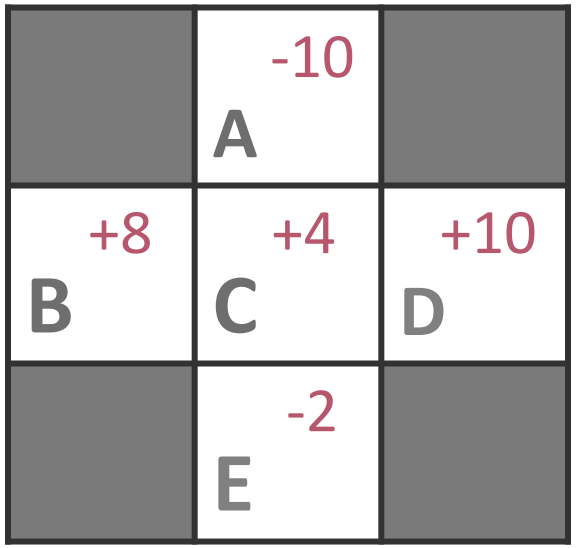
Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

Output Values



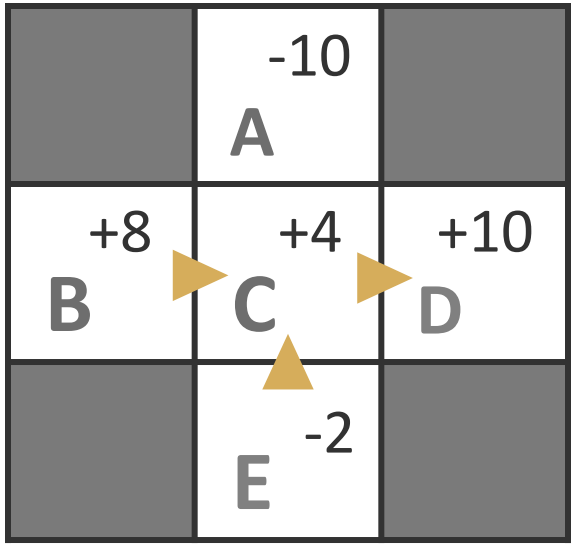
$U^{\pi}(D) = 3/3 * 10 = 10$
$U^{\pi}(A) = 1/1 * -10 = -10$
$U^{\pi}(B) = 2/2 * (-1 + -1 + 10) = 8$
$U^{\pi}(C) = 3/4 * (-1 + 10) +$
$1/4 (-1 + - 10) = 4$
$U^{\pi}(E) = 1/2 * (-1 + -1 + 10) +$
$1/2 (-1 + -1 + - 10) = -2$

Problems with Direct Evaluation

CONTEXT

- **Advantages**
  - Easy to understand
  - Doesn't require any knowledge of T, R
  - Eventually computes the correct average values using only sampled transitions
- **Disadvantages**
  - Wastes information about state connections
  - Each state must be learned separately
  - Takes a long time to learn

Output Values



If B and E both go to C under this policy, how can their values be different?

Questions?

Why Not Use Policy Evaluation?

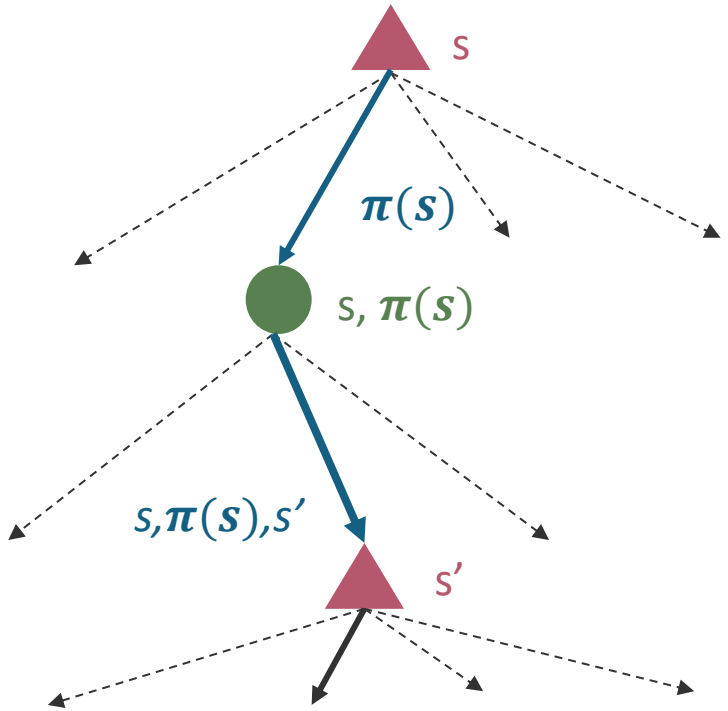
CONTEXT

- Simplified Bellman updates calculate ***U*** for a fixed policy
  - Each round, replace *U* with a one-step look-ahead layer over *U*

$$U_0^{\pi}(s) = 0$$

$$U_{k+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s') \left[R(s, \pi(s), s') + \gamma U_k^{\pi}(s')\right]$$

  - This approach fully exploits connections between states
  - But we need ***T*** and ***R***
- How can we do this update without knowing ***T*** or ***R***?



Sample-Based Policy Evaluation?

FOUNDATIONS

Instead of taking the expectation, just average over observations

$$U^{\pi}_{k+1}(s) \leftarrow \sum_{s'} \frac{1}{n} \left[R(s, \pi(s), s') + \gamma U^{\pi}_{k}(s')\right]$$

- Idea: Take samples of outcome ***s'*** and average

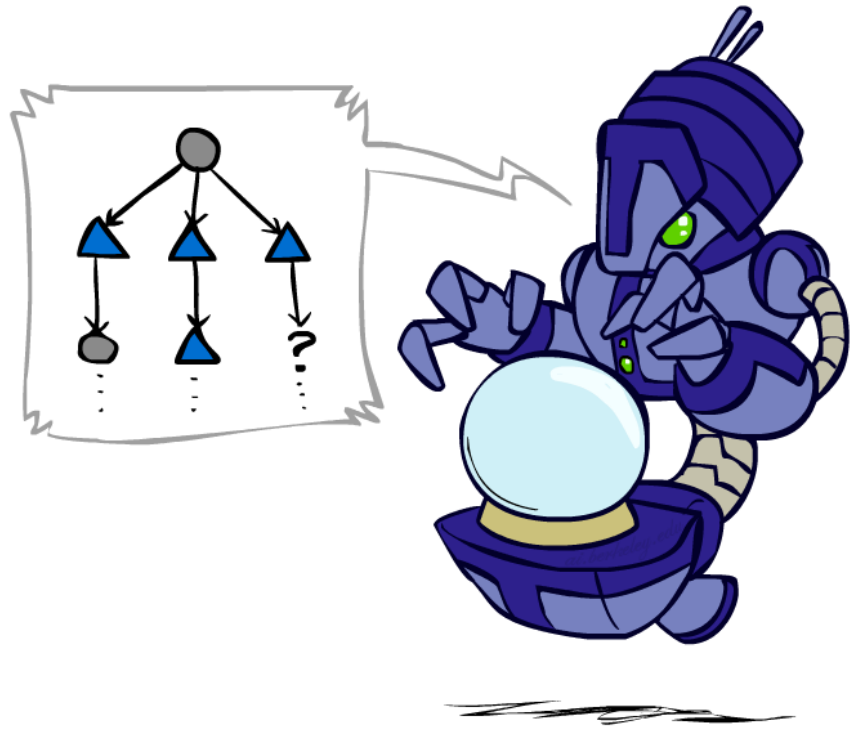
$$\text{sample}_1 = R(s, \pi(s), s'_1) + \gamma U^{\pi}_{k}(s'_1)$$

$$\text{sample}_2 = R(s, \pi(s), s'_2) + \gamma U^{\pi}_{k}(s'_2)$$

...

$$\text{sample}_n = R(s, \pi(s), s'_n) + \gamma U^{\pi}_{k}(s'_n)$$

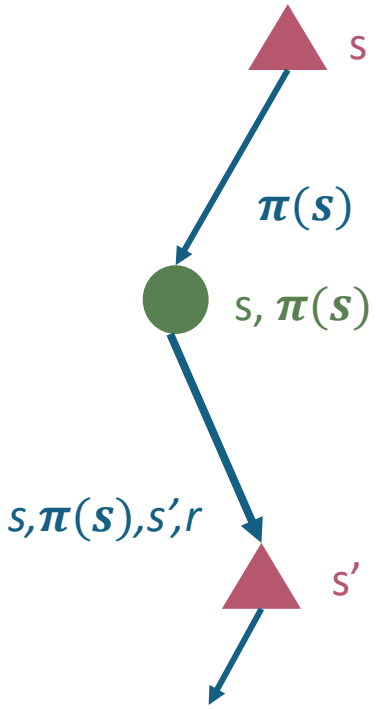
$$U^{\pi}_{k+1}(s) \leftarrow \frac{1}{n} \sum_{i=1}^{n} \text{sample}_i$$



Temporal Difference Learning

FOUNDATIONS

- Idea: Learn from every experience!
  - Update $\boldsymbol{U(s)}$ each time we experience a transition $\boldsymbol{(s, a, s', r)}$
  - Likely outcomes $\boldsymbol{s'}$ will contribute updates more often
- Temporal Difference Learning
  - Policy is still fixed, still doing evaluation
  - Move $U(s)$ toward value of whichever successor s' occurs (running average)
  - Sample of $\boldsymbol{U(s)}$: $\text{sample} = R(s, \pi(s), s') + \gamma U^{\pi}(s')$
  - Update to $\boldsymbol{U(s)}$: $U^{\pi}(s) \leftarrow (1 - \alpha)U^{\pi}(s) + \alpha \cdot \text{sample}$



Exponential Moving Average

DEFINITION

- Running interpolation update

$$\bar{x}_t = (1 - \alpha) \cdot \bar{x}_{t-1} + \alpha \cdot x_t$$

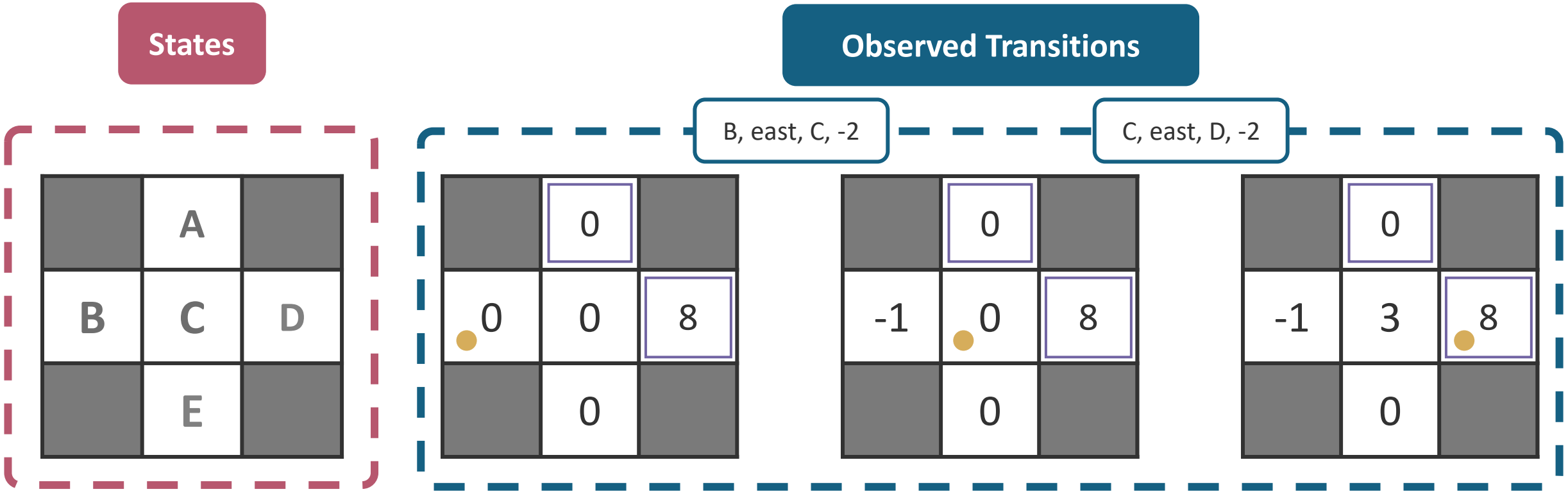
  - Makes recent samples more important
  - Forgets about the past (distant values were wrong anyway)

- Decreasing **learning rate α** can give converging averages

TDL Example (1)

Assume: $\gamma = 1$, $\alpha = 0.5$

EXAMPLE



$$U^{\pi}(s) \leftarrow (1-\alpha)U^{\pi}(s) + \alpha[R(s,\pi(s),s') + \gamma U^{\pi}(s')]$$

TDL Example (2)

EXAMPLE

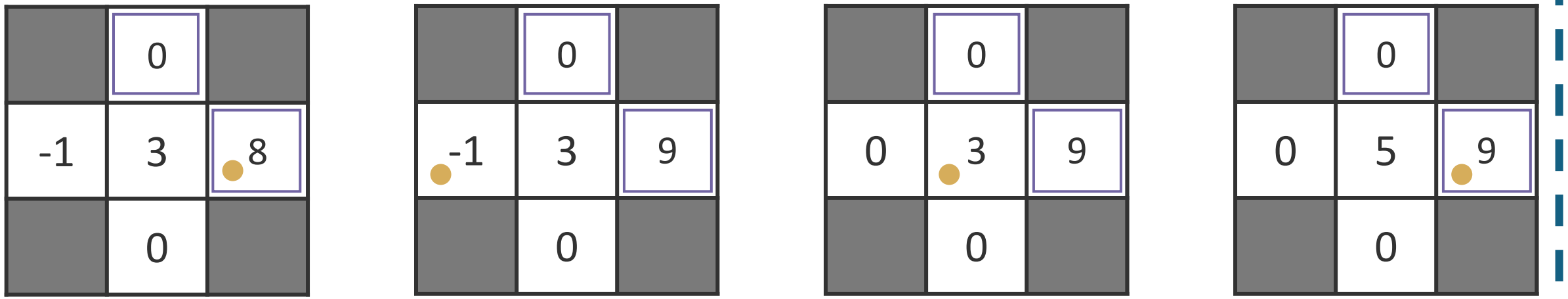
Assume: $\gamma = 1$, $\alpha = 0.5$

Observed Transitions

D, exit, , +10

B, east, C, -2

C, east, D, -2



$U^{\pi}(D) \leftarrow (1/2)U^{\pi}(D) + 1/2\,[+10]$
$\leftarrow 4+5$
$\leftarrow 9$

$U^{\pi}(B) \leftarrow (1/2)U^{\pi}(B) + 1/2\,[-2 + U^{\pi}(C)]$
$\leftarrow -1/2+1.5$
$\leftarrow 0$

$U^{\pi}(C) \leftarrow (1/2)U^{\pi}(C) + 1/2\,[-2 + U^{\pi}(D)]$
$\leftarrow (1/2)3 + 1/2\,[-2 + 9]$
$\leftarrow 5$

Problems with TD Value Learning

CONTEXT

TD Value Learning performs model-free policy evaluation

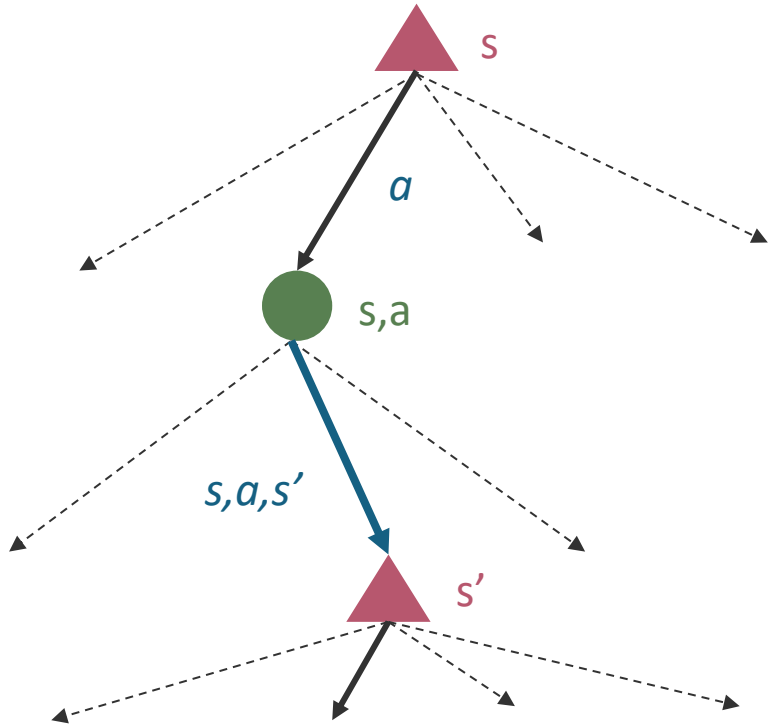
- Mimics Bellman updates with running average samples

- However, if we want to turn values into policy, we're stuck:

$$\pi(s) = \text{argmax}_a Q(s,a)$$

$$Q(s,a) = \sum_{s'} T(s,a,s')[R(s,a,s') + \gamma U(s')]$$

- Idea: Learn **Q-values**, not utilities
  - Makes action selection model-free, too!



Questions?

that's it for today!

SUMMARY

- Reinforcement learning involves learning by doing
- Model-based vs. model-free approaches

UPCOMING

- Q-Learning
- More Reinforcement Learning

REMINDERS

- **Test 1 Friday**
- Do **Homework 2** (due 10/30)
- Do **Project 2** (due 11/2)