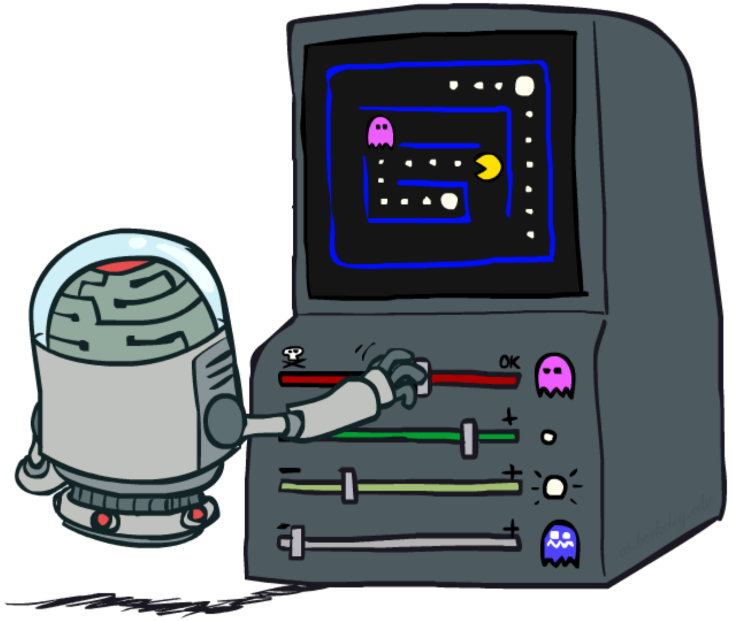
COMS W4701: Artificial Intelligence

Lecture 12: Active Reinforcement Learning



Instructor: Tony Dear

*Lecture materials derived from UC Berkeley's AI course at ai.berkeley.edu

Announcements

- Check HW2 grades and solutions
- Regrade requests by next Wednesday

- Regular recitations tomorrow (topic: RL)
- Lecture next Tuesday: Generalizations of RL, topics review for midterm

- Midterm review session: Example problems for midterm
- Next Wednesday 10/16, 5:30pm in CSB 451

Today

- Passive TD learning
- Multi-armed bandits
- Exploration vs exploitation
- Q-learning

Passive Reinforcement Learning

- Don't know underlying model
- Given a fixed policy, find corresponding values
- Learning is based on a set of **trials** and **observations**

- Model-based: Estimate and update MDP model (adaptive dynamic programming)

- Model-free: Do away with MDP model entirely
- Direct estimation: Accumulate estimates of expected utilities per trial
- **Temporal-difference learning**: Utility estimates persist and update each time a transition is observed based on samples

Temporal-Difference Learning

- Direct utility estimation ignores underlying model and MDP relationship
- ADP respects the MDP, but learning and updating model can get clunky

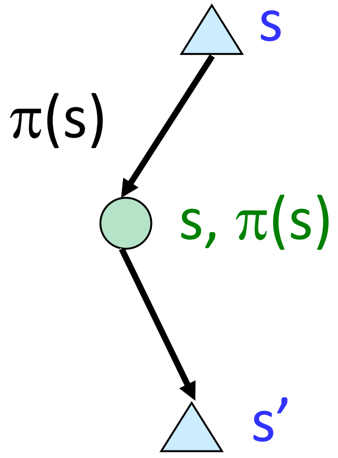
- **TD learning**: Treat each *transition* (s, a, s') as a *sample* for $V^{\pi}(s)$
- Keep track of V^{π} so far and update $V^{\pi}(s)$ with each transition

$$sample = R(s, \pi(s), s') + \gamma V^{\pi}(s')$$

α: **learning rate**

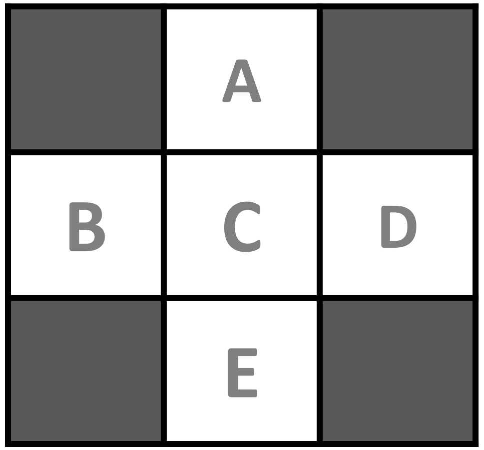
$$V^{\pi}(s) \leftarrow (1-\alpha)V^{\pi}(s) + (\alpha)sample$$

$$V^{\pi}(s) \leftarrow V^{\pi}(s) + \alpha(sample - V^{\pi}(s))$$



Example: TD Learning

States

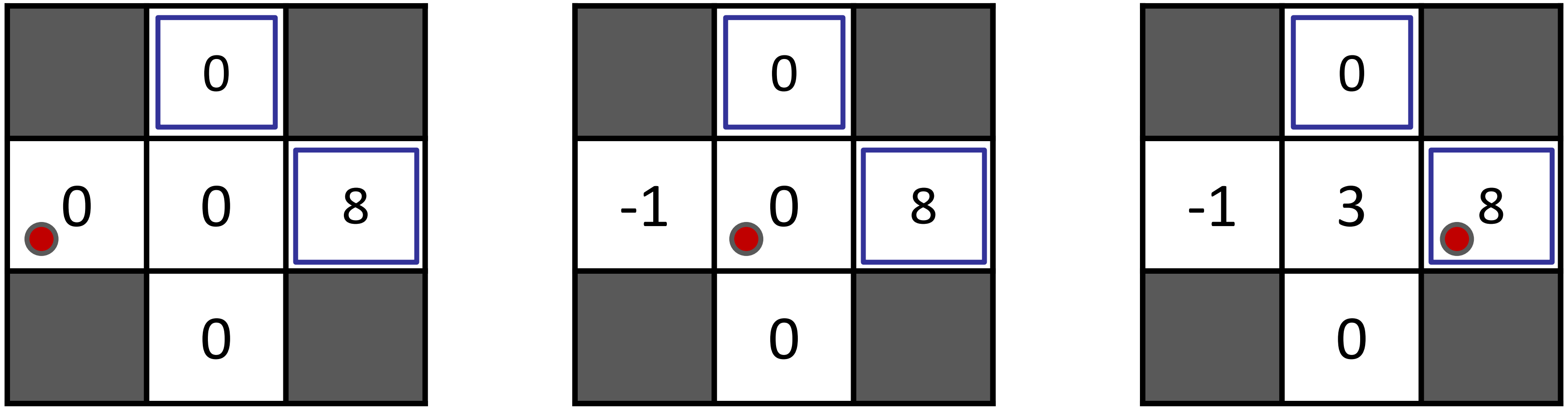


Assume: γ = 1, α = 1/2

Observed Transitions

B, east, C, -2

C, east, D, -2



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

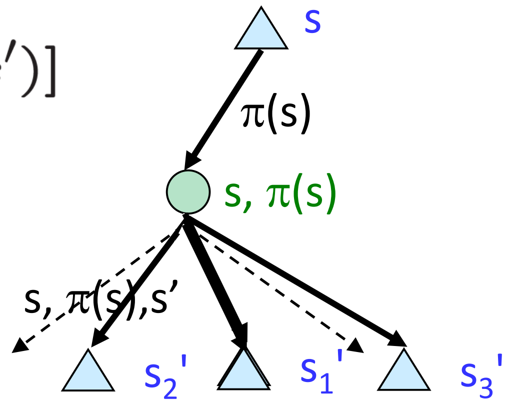
$$V^{\pi}(B) \leftarrow 0.5V^{\pi}(B) + 0.5\big(-2 + 1V^{\pi}(C)\big) = 0.5(0) + 0.5\big(-2 + 1(0)\big) = -1$$

$$V^{\pi}(C) \leftarrow 0.5V^{\pi}(C) + 0.5\big(-2 + 1V^{\pi}(D)\big) = 0.5(0) + 0.5\big(-2 + 1(8)\big) = 3$$

Model-Based vs TD Learning

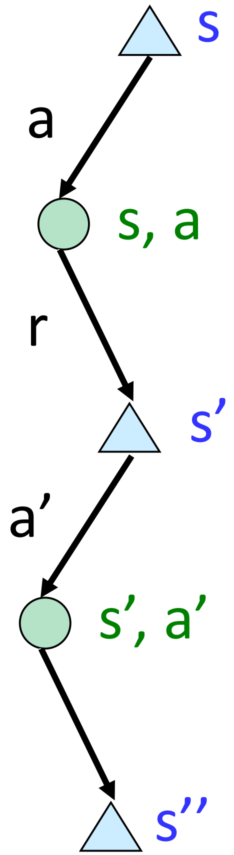
- Both preserve the underlying MDP model

$$V_{k+1}^{\pi}(s) \leftarrow \sum_{s'} T(s,\pi(s),s')[R(s,\pi(s),s') + \gamma V_k^{\pi}(s')]$$

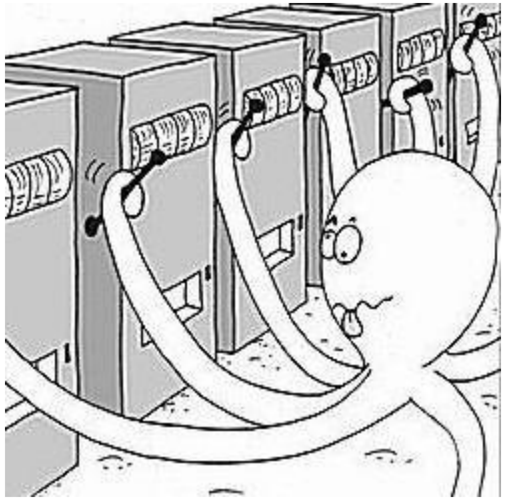


- ADP averages over **all** possible outcomes from every state, weighted by transition probabilities
- Same as policy evaluation

- TD only averages for each observed state
- Converges more slowly and unpredictably, but simpler than ADP



Multi-Armed Bandits

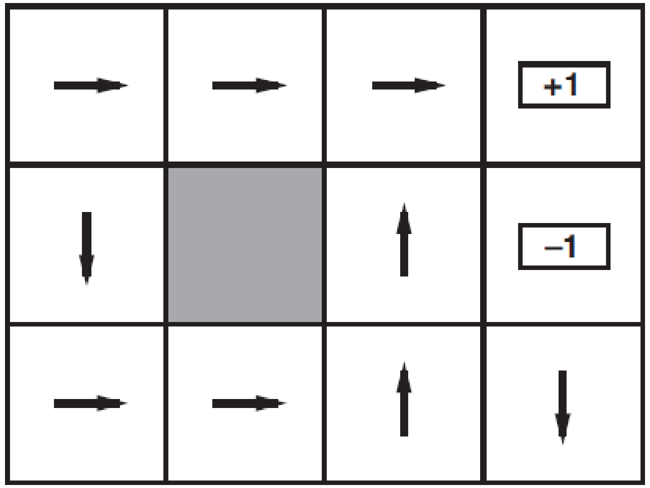


- Problem: Don’t know environment or optimal policy
- Still want to maximize rewards

- Tradeoff between **exploration** and **exploitation**
  - Gather more information or maximize best rewards so far?
  - How to determine when model is good enough?

- Applications: Resource allocation for maximizing productivity, clinical trials to explore different treatments, financial portfolio design

Active Reinforcement Learning



- Bandits are a special case of active reinforcement learning
- Unknown transitions, rewards, policy, values
- Goal: Still want to maximize expected utility

- Suppose we have value estimates from ADP or TD learning
- Then we can extract the best policy for these values

$$\hat{\pi}(s) = \mathrm{argmax}_a \sum_{s'} \hat{T}(s,a,s')[\hat{R}(s,a,s') + \gamma \hat{V}(s')]$$

- Is $\hat{\pi}$ the best policy overall? Probably not since we're relying on estimates!
- Need to explore; take non-optimal actions and update the model

Exploration Functions

- Simplest exploration scheme: ***ε*-greedy**
- Follow policy but perform random action with probability ε

- **Exploration function**: Prioritize less visited states
- Replace value estimates $\hat{V}$ with $f(\hat{V}, N(s,a))$
- $N(s,a)$ counts number of times that (s,a) has been visited

- Suppose we want to visit all states at least N_e times
- One option: *Replace* $\hat{V}$ with optimistic estimate R^+ $\quad f(u,n) = \begin{cases} R^+ \text{ if } n < N_e \\ u \text{ otherwise} \end{cases}$

- Alternatively: *Inflate* $\hat{V}$ with a bonus that decreases over time $\quad f(u,n) = u + N_e/n$

Q-Value Iteration

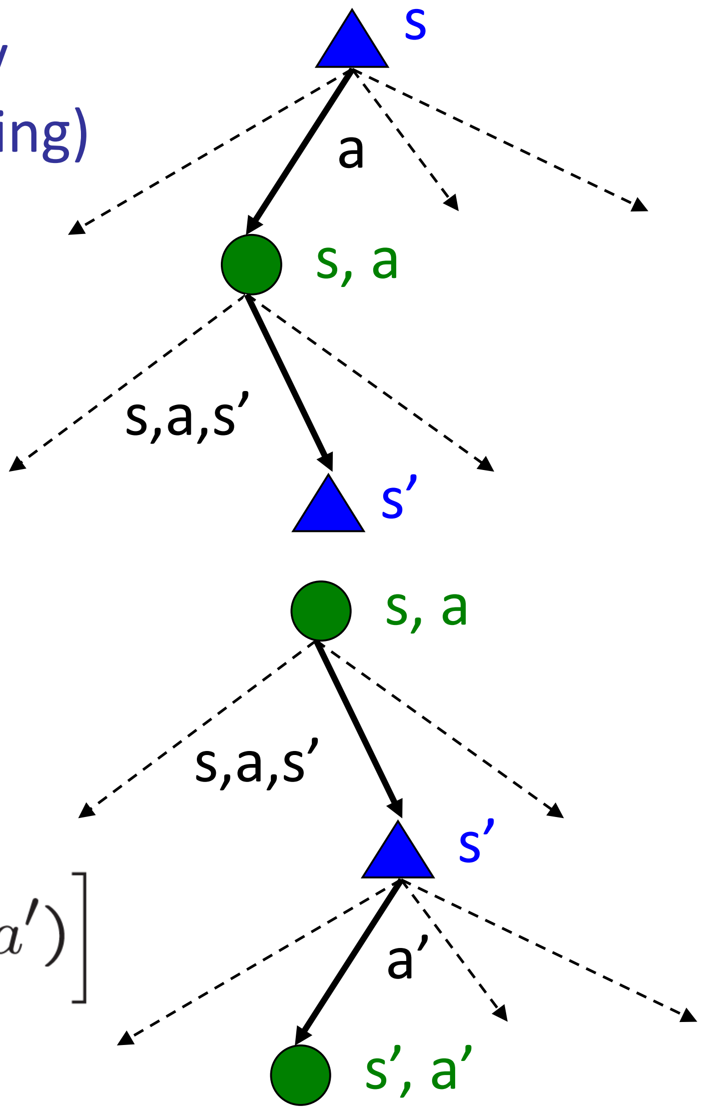
- We're doing an awful lot of work to extract a new policy every time we update our model (ADP) or value estimates (TD learning)

- How do we extract a policy? Argmax over Q-values!
- Why not just compute and keep Q-values around?

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_k(s')\right]$$

$$\Downarrow$$

$$Q_{k+1}(s,a) \leftarrow \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma \max_{a'} Q_k(s',a')\right]$$



Q-Learning

- We can use samples to simulate Bellman update for Q-values instead of state values

$$Q_{k+1}(s,a) \leftarrow \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$$

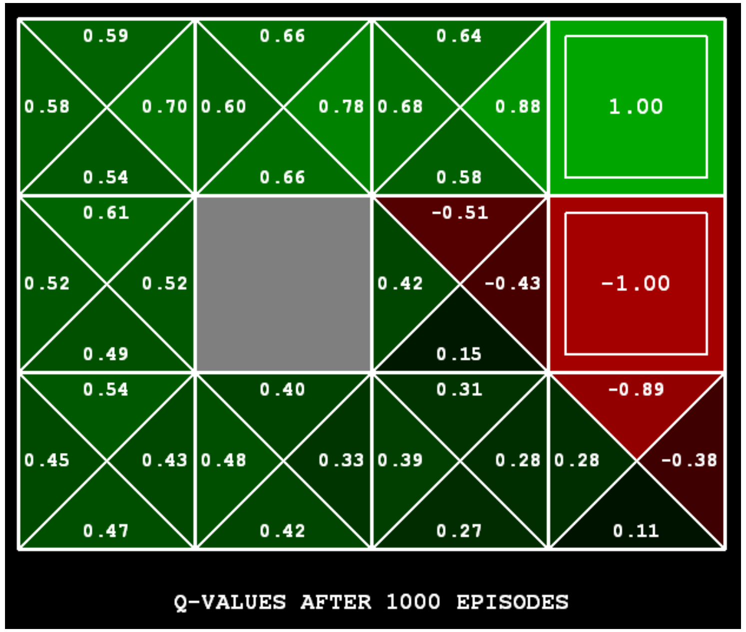
- If we take TD approach, estimate Q using running average

$$sample = r + \gamma \max_{a'} Q(s',a')$$

α: **learning rate**

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(sample)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha(sample - Q(s,a))$$



- Action selection if using exploration function:

$$a = \underset{a'}{\operatorname{argmax}} f(Q(s',a'), N(s',a'))$$

Example: Q-Learning

- Observed transitions:

  B, east, C, -2

  C, north, A, -2

- Which one is exploration and which one is exploitation?
- What updates occur for Q-learning?

$$sample = r + \gamma \max_{a'} Q(s', a')$$

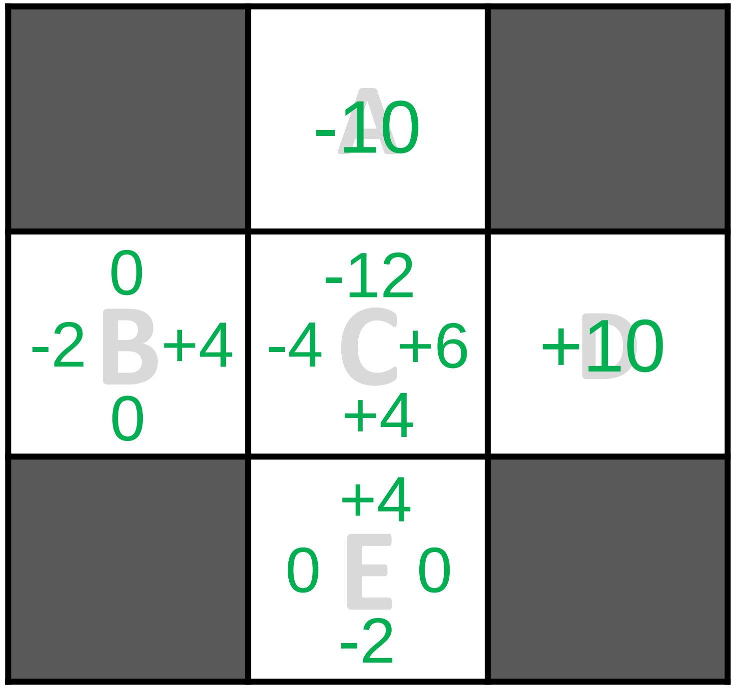
$$Q(s,a) \leftarrow Q(s,a) + \alpha(sample - Q(s,a))$$

$$sample_1 = -2 + 0.8 \max(-12, -4, 4, 6) = 2.8$$

$$sample_2 = -2 + 0.8 \max(-10) = -10$$

$$Q(B, east) = 4 + 0.5(2.8 - 4) = 3.4$$

$$Q(C, north) = -12 + 0.5(-10 - (-12)) = -11$$



$\gamma = 0.8$ $\alpha = 0.5$

Parameters for Q-Learning

- Q-learning updates are off-policy: may not reflect actions taken if exploring

$$sample = r + \gamma \max_{a'} Q(s', a')$$

- Can still converge to optimal policy!

- **Learning rate α**: Convergence guaranteed if α decreases to 0 over time
  - In practice, a constant rate, e.g. $\alpha = 0.1$, is sufficient
- **Exploration rate ε**: Can be constant, can decrease over time depending on context

- **Discount factor γ**: Determines significance of future rewards to agent
  - In practice, learning is faster if starting with lower γ and then increasing over time
- **Initial conditions Q_0**: Inflated initial values can encourage exploration

Summary: MDPs and RL

Known MDP: Offline Dynamic Programming

Goal	Technique
Evaluate fixed policy π	Policy evaluation
Find optimal $\pi^*, V^*, (Q^*)$	Value / policy iteration

Unknown MDP: Model-Based

Goal	Technique
Evaluate fixed policy π	ADP / policy evaluation
Find optimal $\pi^*, V^*, (Q^*)$	ADP / policy exploration

Unknown MDP: Model-Free

Goal	Technique
Evaluate fixed policy π	TD Value Learning
Find optimal $\pi^*, V^*, (Q^*)$	Q-learning