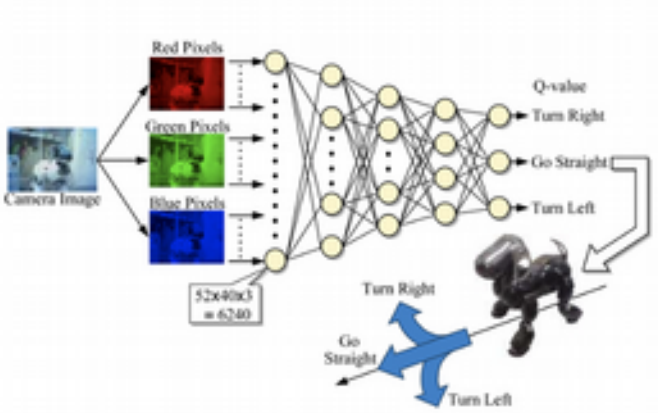
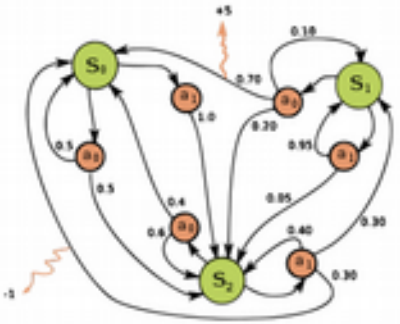
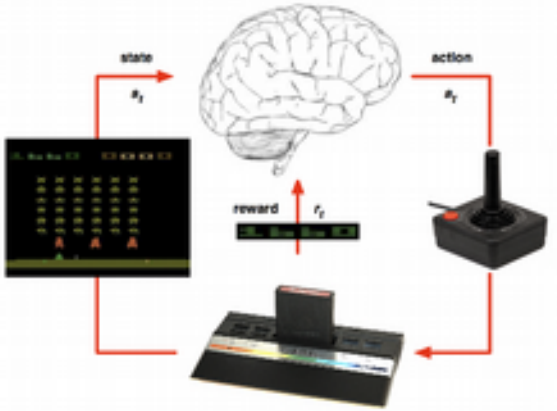
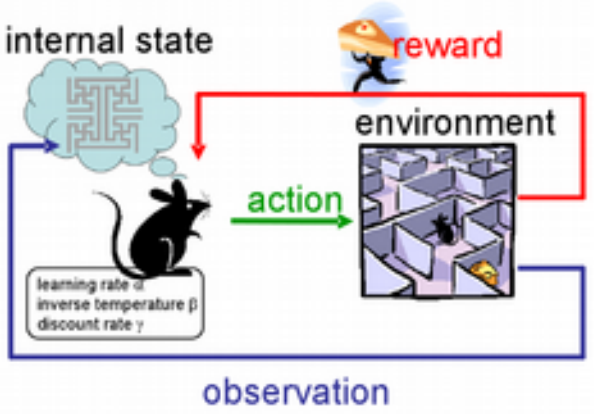
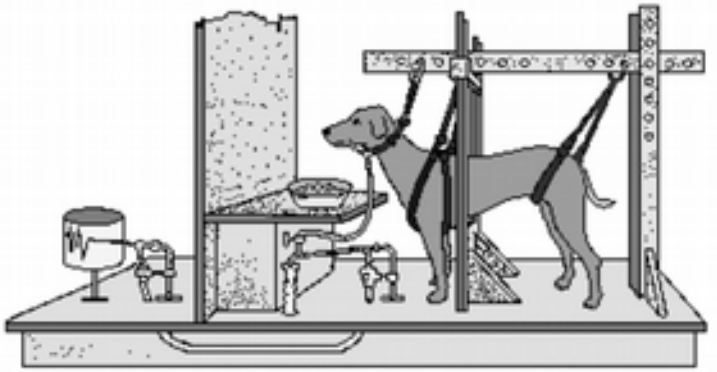
COMP 138: Reinforcement Learning











Instructor: Jivko Sinapov
Webpage: https://www.eecs.tufts.edu/~jsinapov/teaching/comp150_RL_Fall2020/

Announcements

Announcements

- Homework 2 is out

Reading Assignment

- Chapters 6 of Sutton and Barto

Research Article Topics

- Transfer learning
- Learning with human demonstrations and/or advice
- Approximating q-functions with neural networks

Reading Assignment

- Chapter 6 of Sutton and Barto
- Matthew E. Taylor, Peter Stone, and Yaxin Liu. **Transfer Learning via Inter-Task Mappings for Temporal Difference Learning**. Journal of Machine Learning Research, 8(1):2125-2167, 2007.

Discussion Comments

“How can MDP deal with problems where some of the states are hidden since most of the real-world problems have hidden information?”

- Tung

Discussion Comments

“How can we know what is the most “optimal” policy for a game or an environment if there are multiple policies given the same amount of reward?”

- Tung

Discussion Comments

“I’d like to know more about how the discount factor plays an important role in the efficient training of the agent to learn a task. How different values of the discount factor influence the time needed for the agent to learn the task. I have observed that a value close to 1 is preferred, but how would the learning change if this value is decreased?”

– Yash

Discussion Comments

“As far as I understand, a robot made to solve a real-world problem such as delivering mail over the air would “trigger” a new state and consult its policy with some fixed frequency (or whenever a drastic change happens). I am wondering if it is common for the robot to adjust that frequency. In case of severe weather and strong wind, for instance, the agent might want to consult its policy more often than it would when the wind stays still. Would adjusting the frequency with which a new state is triggered be a kind of action that the agent can make?”

– Sasha

Discussion:
What makes a good project?

Project Breakout

- Talk to 4-5 people in the class
- Note: their names, their interests, do they have ideas for project or not, do they anticipate using RL in their research?

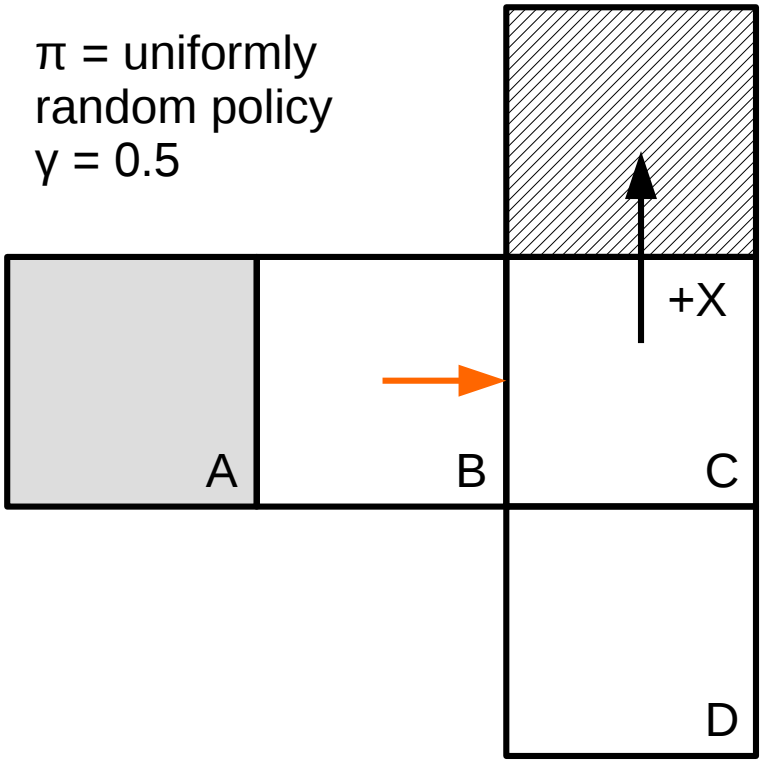
Dynamic Programming

Policy Evaluation

Iterative policy evaluation

Input π, the policy to be evaluated
Initialize an array $V(s) = 0$, for all $s \in \mathcal{S}^+$
Repeat
    $\Delta \leftarrow 0$
    For each $s \in \mathcal{S}$:
        $v \leftarrow V(s)$
        $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r \,|\, s, a)\big[r + \gamma V(s')\big]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)
Output $V \approx v_\pi$

π = uniformly random policy
γ = 0.5

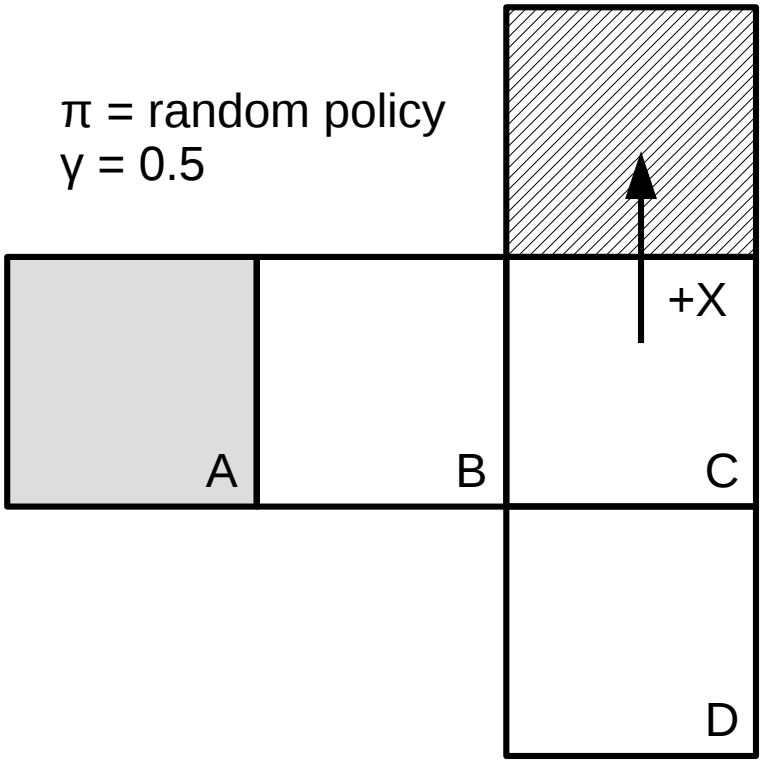


	V_0	V_1	V_2	V_3	V_4	V_5
A	0	0	0	X/24		
B	0	0	X/12			
C	0	X/3	3X/8			
D	0	X/6	3X/16			

$$
\begin{aligned}
v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \Big[r + \gamma v_k(s') \Big]
\end{aligned}
$$

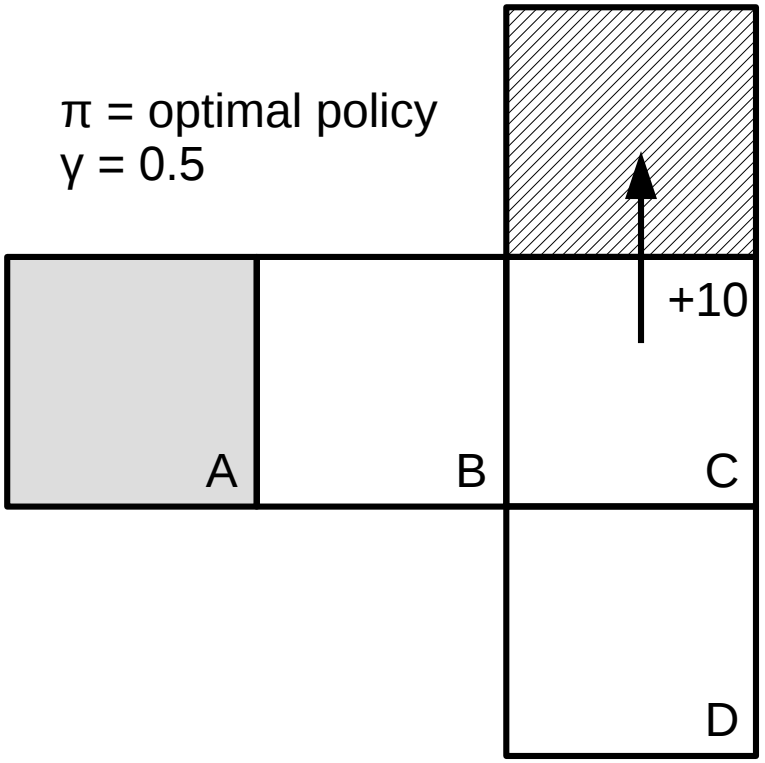
Working area: 3x/16

π = random policy
γ = 0.5



	V_0	V_1	V_2	V_3	V_4	V_5
A	0	0	0			
B	0	0	X/12			
C	0	X/3				
D	0	X/6				

$$
\begin{aligned}
v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) \Big[r + \gamma v_k(s') \Big]
\end{aligned}
$$



	V_0	V_1	V_2	V_3	V_4	V_5
A	0					
B	0					
C	0					
D	0					

At each state, the agent has 1 or more actions allowing it to move to neighboring states. Moving in the direction of a wall is not allowed

$$
\begin{aligned}
v_{k+1}(s) &\doteq \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s', r \mid s, a) \Big[r + \gamma v_k(s') \Big]
\end{aligned}
$$

WORKING TEXT AREA:

Policy Improvement

- Main idea: if for a particular state *s*, we can do better than following the current policy by taking a different action, then the current policy is not optimal and changing it to follow the different action at state *s* improves it

Policy Iteration

- evaluate → improve → evaluate → improve → …..

Value Iteration

- Main idea:
  - Do one sweep of policy evaluation under the current greedy policy
  - Repeat until values stop changing (relative to some small Δ)

Policy iteration (using iterative policy evaluation)

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
   $\Delta \leftarrow 0$
   For each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s)) \big[r + \gamma V(s')\big]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a) \big[r + \gamma V(s')\big]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Project Breakout: Finding a Partner

THE END