HW6 - MDP Recitation
CIS521 - Artificial Intelligence

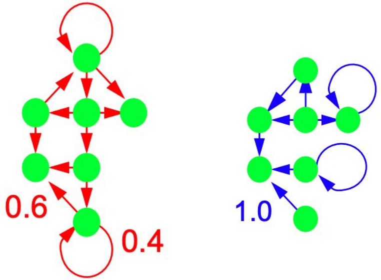
Oct. 21, 2022

Penn Engineering

Markov Decision Process

Markov Decision Process

- A set of possible world states S
- A set of possible actions A
- A real valued reward function R(s, a, s')
  - Reward based on taking action a, moving from s to s'
- A description T(s, a, s') of each action's effects in each state (transition function)
  - Represents the distribution of P(s' | s, a)



Penn Engineering

Markov Property

- The effects of an action taken in a state depend only on that state and not on the prior history.
- R(s, a, s’)
- P(s’ | s, a)

How to solve such a problem?

Goal (typically)

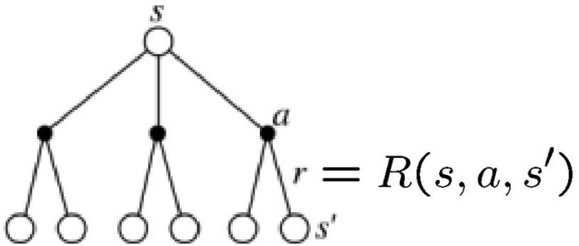
- Finding a best policy that can maximize expected sum of rewards
- Best policy: A set of best actions at different states

V: State-value

Bellman Equation of V state-value function:

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \sum_{s' \in \mathcal{S}} P(s'|s,a)[R(s,a,s') + \gamma V^\pi(s')]$$

Backup Diagram:



Optimal policy and optimal state-value function:

$$V^*(s) := \max_\pi V^\pi(s) = V^{\pi^*}(s), \qquad \forall s \in \mathcal{S}$$

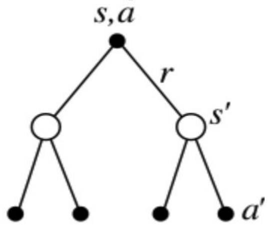
Q: Action-value

Bellman Equation of the Q Action-Value function:

$$Q^\pi(s,a) = \sum_{s' \in \mathcal{S}} P(s'|s,a) \left[R(s,a,s') + \gamma \sum_{a' \in \mathcal{A}} \pi(a'|s') Q^\pi(s',a') \right]$$

Proof: similar to the proof of the Bellman Equation of V state-value function.

Backup Diagram:



Similarly, the **optimal action-value function:**

$$Q^*(s,a) := \max_\pi Q^\pi(s,a)$$

Relations with each other

Q from V:

$$Q^{\pi}(s,a) = \sum_{s' \in \mathcal{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V^{\pi}(s')\right]$$

V from Q:

$$V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) Q^{\pi}(s,a)$$

Relations with each other

Important Properties:

$$Q^*(s,a) = \mathbb{E}\left[r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a\right]$$

$$V^*(s) = \max_{a \in \mathcal{A}} Q^*(s,a)$$

$$Q^*(s,a) = \sum_{s' \in \mathcal{S}} P(s'|s,a)\left[R(s,a,s') + \gamma V^*(s')\right]$$

Penn Engineering

Greedy Policy for V

Definition: Greedy policy for a given $Q(s,a)$ function:

$$\pi(s,a) = \begin{cases} 1, & \text{if } a = \arg\max_a Q(s,a) \\ 0, & \text{otherwise;} \end{cases}$$

Equivalently, (Greedy policy for a given V(s) function):

$$\pi(s,a) = \begin{cases} 1, & \text{if } a = \arg\max_a P(s' \mid s,a)(R(s,a,s') + \gamma V(s')) \\ 0, & \text{otherwise;} \end{cases}$$

Greedy optimal policy

Theorem: A greedy optimal policy from the optimal Value function:

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} \left[R(s,a,s') + \gamma \sum_{s' \in \mathcal{S}} P(s'|s,a) V^*(s') \right]$$

HW6: Value iteration

For each state:

- Find best action
- Update state value to be the best q value given the best action

HW6: Policy iteration

- Start with an initial policy using the best actions obtained via the previous code
- Update state values based on the current policy
- Get best actions based on each state
- Compare Q values and update current policy
- Keep updating the state values until the difference between $V_{k+1}(s)$ and $V_k(s)$ reaches a difference of less than 1e-6

References:

https://www.cs.cmu.edu/~mgormley/courses/10601-s17/slides/lecture26-ri.pdf