CS 486 — Lecture 18: Markov Decision Processes, Part 2

1 Solving for the Optimal Policy

- $V^\pi(s)$ represents the expected utility of entering state s and following policy π after.
- $V^*(s)$ means the expected utility of entering state s and following optimal policy π^* after.
- What is the expected utility if I am in state s and take action a?

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)V^*(s')$$

- In state s, choose an action that maximizes my expected utility:

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

1.1 Bellman Equations

- We define V and Q as such:

$$V^*(s) = R(s) + \gamma \max_a Q^*(s,a)$$
$$Q^*(s,a) = \sum_{s'} P(s'|s,a)V^*(s')$$

- If we combine the equations, we get the Bellman equation:

$$V^*(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)V^*(s')$$

- We cannot solve the Bellman equation system efficiently.
- We can solve for the $V^*(s)$'s iteratively:
  1. Let $V_i(s)$ be the values for the i'th iteration. Start with arbitrary initial values for $V_0(s)$.
  2. As the i'th iteration, compute $V_{i+1}(s) = R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)V_i(s')$.
  3. Terminate when $\max_s |V_i(s) - V_{i+1}(s)|$ is small enough.
- If we apply the Bellman update infinitely often, V_i's are guaranteed to converge to the optimal values!

1.2 Value Iteration

- Each state accumulates negative rewards until the algorithm finds a point to the +1 goal state.
- We have two types of updates for $V^*(s)$:
  - Synchronous — store and use $V_i(s)$ to calculate $V_{i+1}(s)$.
  - Asynchronous — store and use $V_i(s)$ and update the values one at a time in any order.