COMP 4752
Computational Intelligence

Lecture 16

Dynamic Programming for MDP

Gridworld / Assignment 3 Explanation

Dynamic Programming

- Collection of algorithms that can compute optimal policies given a perfect model of the environment as an MDP
- DP's main idea: my value is related somehow to the value of my neighbours
- We will use DP to compute the value function, and use the value function to compute the policy

Policy (Value) Evaluation

- Let's compute the state-value function V^{π} for an arbitrary policy π
- We use the Bellman Equation

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

- π (s,a) = Probability of taking a in s under π

Policy Evaluation

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

- In our case, actions deterministically transition us from one state to another:

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

In Code?

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

```
V[s] = 0
for each legal action a at s:
    p = probability of choosing a at s (π)
    r = reward from doing a at s
    ns = apply a to s (next state)
    v[s] += p * (r + gamma * v[ns])
```

In Code?

$$V^{\pi}(s) = \sum_{a} \pi(s,a) \left[\mathcal{R}^{a}_{ss'} + \gamma V^{\pi}(s') \right]$$

```
for each state s in environment:
        v[s] = 0
        for each legal action a at s:
            p = probability of choosing a at s (π)
            r = reward from doing a at s
            ns = apply a to s (next state)
            v[s] += p * (r + gamma * v[ns])
```

Policy Update

- Taking which action at s has highest value

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

- We just computed this! If we want to know the value of doing action a at state s, look up the value of the resulting v[s’]

Policy Update

- Update the policy to choose the action that yields the highest value

$$\pi'(s) = \arg\max_{a} Q(s, a)$$

In Code

```
for each legal action a at s:
        s' = apply a to s (deterministic)
        value = v[s']
        if (value is new max)
            record a as max action
update policy with equiprobable choice of
selecting from all actions with max value
```

Example Policy Update

- 5 actions [a1, a2, a3, a4, a5] at state s
- Yield new states [s1, s2, s3, s4, s5]
- With values [10, 0, 10, 5, 10]
- Actions that maxed value = [a1, a3, a5]
- New policy = equiprobable a1, a3, a5
- NewPolicy[s] = [0.33, 0, 0.33, 0, 0.33]