Finding the Optimal Policy (I)

COMP4211

THE DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
計算機科學及工程學系

Optimal Policy

The task is to learn the optimal policy π^*

$$\pi^* \equiv \arg\max_\pi V^\pi(s), \quad \forall s$$

How to learn the optimal policy?

Simple Idea

Run through all possible policies. Select the best

What's the problem?

Optimal Action

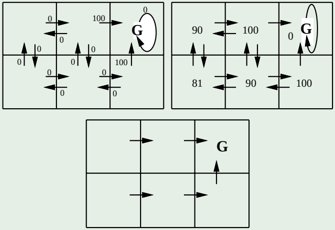
We can try to learn V^* (optimal state value function) first

$$V^*(s) = \max_\pi V^\pi(s)$$

- maximum discounted cumulative reward that the agent can obtain starting from start s

If you know V^*, what is the optimal action at state s?

Example (Problem; V^*)



- actions that appear best after a one-step search will be optimal

Bellman Optimality Condition

expected return for action a:

- immediate reward $r(s, a)$ plus the discounted value of V^* of the immediate successor state $\delta(s, a)$

$$r(s, a) + \gamma V^*(\delta(s, a))$$

(non-deterministic: $\sum_{s'} P(s, s', a)[R(s, s', a) + \gamma V^*(s')]$)

optimal policy

- take the action with maximum $r(s, a) + \gamma V^*(\delta(s, a))$

$$\max_a[r(s, a) + \gamma V^*(\delta(s, a))]$$

The value of a state under an optimal policy must equal the expected return for the best action from that state

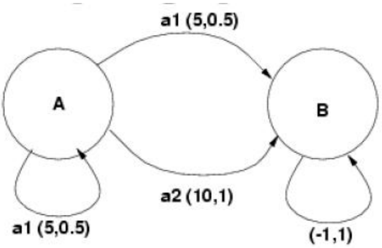
Bellman optimality condition

$$\boxed{V^*(s) = \max_a[r(s, a) + \gamma V^*(\delta(s, a))]}$$

(cf: Bellman condition:

$$V^\pi(s) = r(s, a) + \gamma V^\pi(\delta(s, a))$$

Example



$$V^*(A) = \max(5 + \gamma(0.5)(V^*(A) + V^*(B)), 10 + \gamma V^*(B))$$
$$V^*(B) = -1 + \gamma V^*(B)$$

For $\gamma = 0.5$, we get

$$V^*(B) = \frac{-1}{1-\gamma} = -2$$
$$V^*(A) = \max\left(\frac{5+(0.5)(0.5)(-2)}{1-(0.5)(0.5)}, 10 + (0.5)(-2)\right) = 9$$

How to find the optimal policy?

- policy iteration
- value iteration
- Q learning

Policy Improvement

Suppose we have computed V^π for a policy π (policy evaluation)

For a given state s, would it be better to do an action $a \neq \pi(s)$?

Value of taking action a in state s:

$$Q^\pi(s,a) = r(s,a) + \gamma V^\pi(\delta(s,a))$$

- action value function for policy π

better to switch to action a for state s if $Q^\pi(s,a) > V^\pi(s)$

- do this for all states to get a new policy π'

$$\begin{aligned} \pi'(s) &= \arg\max_a Q^\pi(s,a) \\ &= \arg\max_a r(s,a) + \gamma V^\pi(\delta(s,a)) \end{aligned}$$

Policy Iteration

$$\pi_0 \to V^{\pi_0} \to \pi_1 \to V^{\pi_1} \to \cdots \pi^* \to V^* \to \pi^*$$

policy evaluation policy improvement

Policy Iteration...

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

2. Policy Evaluation
   Repeat
   $\Delta \leftarrow 0$
   For each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s'} \mathcal{P}_{ss'}^{\pi(s)} \left[\mathcal{R}_{ss'}^{\pi(s)} + \gamma V(s')\right]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   $b \leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s'} \mathcal{P}_{ss'}^{a} \left[\mathcal{R}_{ss'}^{a} + \gamma V(s')\right]$
   If $b \neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop; else go to 2

Value Iteration

Recall

- Bellman optimality condition

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

- iterative policy evaluation

$$V_{k+1}(s) \leftarrow \sum_a \pi(s, a) \sum_{s'} P(s, s', a)[R(s, s', a) + \gamma V_k(s')]$$

Value iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s, s', a)[R(s, s', a) + \gamma V_k(s')]$$

- $V(s)$ converges to $V^*(s)$

Value Iteration...

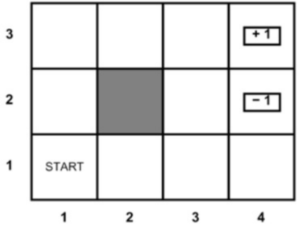
Initialize V arbitrarily, e.g.,$V(s) = 0$, for all $s \in \mathcal{S}^+$

Repeat
    $\Delta \leftarrow 0$
    For each $s \in \mathcal{S}$:
        $v \leftarrow V(s)$
        $V(s) \leftarrow \max_a \sum_{s'} \mathcal{P}^a_{ss'} [\mathcal{R}^a_{ss'} + \gamma V(s')]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

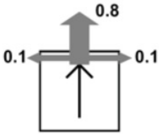
Output a deterministic policy, π, such that
    $\pi(s) = \arg\max_a \sum_{s'} \mathcal{P}^a_{ss'} \left[\mathcal{R}^a_{ss'} + \gamma V(s')\right]$

- value iteration works even if we randomly traverse the environment instead of looping through each state
  - but we must still visit each state infinitely often
- in practice, value iteration often requires less total time to find the optimal solution compared to policy iteration

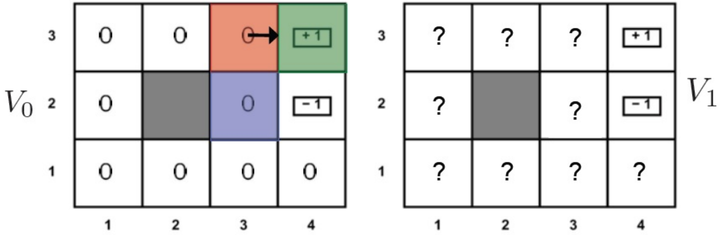
Example



- The agents actions do not always go as planned:
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10
  - If there is a wall in the direction the agent would have been taken, the agent stays put



Example...



- $\gamma = 0.9$, immediate reward $= 0$

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s, s', a)[R(s, s', a) + \gamma V_k(s')]$$

for the red grid $s = (3, 3)$, what is $V_1(s)$?

$$0.8 \times [0.0 + 0.9 \times 1.0] + 0.1 \times [0.0 + 0.9 \times 0.0] + 0.1 \times [0.0 + 0.9 \times 0.0]$$

Example...

V_1

3	0	0	0.72	+1
2	0		0	−1
1	0	0	0	0
	1	2	3	4

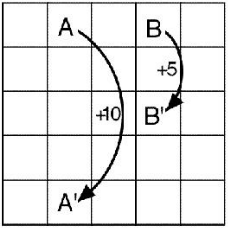
V_2

3	0	0.52	0.78	+1
2	0		0.43	−1
1	0	0	0	0
	1	2	3	4

- information propagates outward from terminal states and eventually all states have correct value estimates

Another Example

Problem; V^*; π^*



22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7