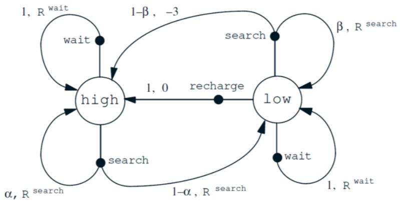


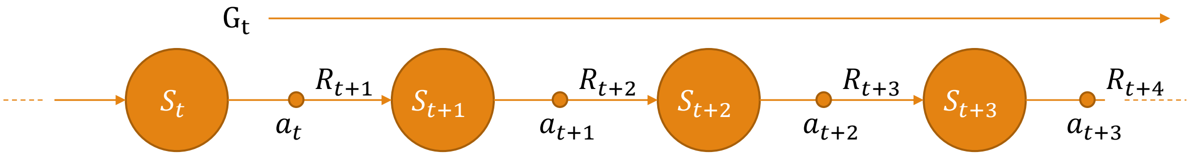
ECE 414/517 Reinforcement Learning

LECTURE 6: BELLMAN EQUATIONS

SEP. 12 2022

(ADAPTED FROM SLIDES BY DR. ITAMAR AREL)

Returns and state value function



$$\mathrm{G_t} = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{\{i=0\}}^{\infty} \gamma^i R_{t+i+1}$$

$$\mathrm{G_t} = R_{t+1} + \gamma \mathrm{G_{t+1}}$$

$$V^{\pi}(s) = E_{\pi}[\mathrm{G_t} | \mathrm{S_t} = \mathrm{s}]$$

Bellman Equations

Given the definitions for the value function and the return:

$$V^{\pi}(s) = E_{\pi}[\mathrm{G_t}|\mathrm{S_t} = \mathrm{s}]$$
$$\mathrm{G_t} = R_{t+1} + \gamma \mathrm{G_{t+1}}$$

We can derive the following:

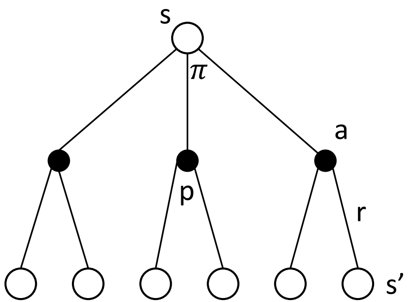
$$V^{\pi}(s) = E_{\pi}[R_{t+1} + \gamma G_{t+1}|\mathrm{S_t} = \mathrm{s}]$$
$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma V^{\pi}(s')]$$

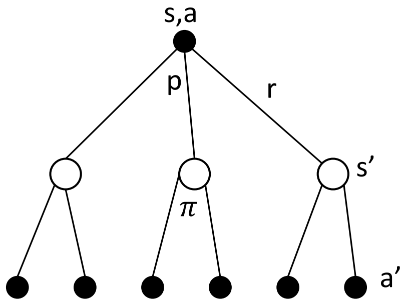
Can be solved as linear equations ($|s|$ equations with $|s|$ unknowns)

Backup Diagram

$$V^{\pi}(s) = \textstyle\sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma V^{\pi}(s')] = \displaystyle\sum_a \pi(a|s)\, Q^{\pi}(s,a)$$

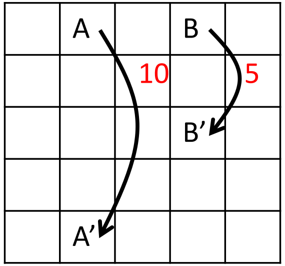
$$Q^{\pi}(s,a) = \sum_{s',r} p(s',r|s,a)\left[r + \gamma \sum_{a'} \pi(a'|s') Q^{\pi}(s',a')\right] = \sum_{s',r} p(s',r|s,a)[r + \gamma V^{\pi}(s')]$$





Grid World

1. An agent starts at a certain cell on a grid and at each time step can make one of 4 moves: **north, south, east, west.**
2. If a move is made which takes the agent off the grid, it does not move but is given a reward of -1
3. Two special grid cells which produce reward:
   - Grid cell A give a reward of 10 for any move and you always end up at A'
   - Grid cell B give a reward of 10 for any move and you always end up at B
4. All other moves give a zero reward
5. Calculate $V_\pi(s)$ for all states where $\pi = \{0.25, 0.25, 0.25, 0.25\}$ for all states
6. Set $\gamma = 0.9$



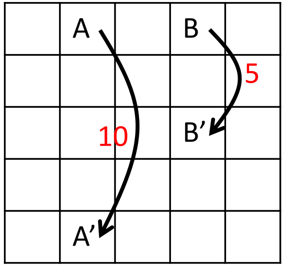
$$v^{\pi}(s) = \sum_{a} \pi(a,s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma V^{\pi}(s')\right]$$

$$v^{\pi}(0,0) = \overset{W}{0.25} \cdot \left[-1 + 0.9\, V^{\pi}(0,0)\right] + \overset{N}{0.25}\left[-1 + 0.9\, V^{\pi}(0,0)\right]$$
$$+ \underset{}{\overset{S}{0.25}} \cdot \left[0 + 0.9\, V^{\pi}(1,0)\right] + \overset{E}{0.25}\left[0 + 0.9\, V^{\pi}(0,1)\right]$$

$$v^{\pi}(0,1) = \overset{W}{0.25} \cdot \left[10 + 0.9\, V^{\pi}(4,1)\right] \times 4$$

Grid World

$$V^{\pi}(s) = \mathrm{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid S_t = s\right] = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma V^{\pi}(s')]$$



1. Show that the bellman equations hold for the middle square.

$$\begin{aligned} V(2,2) = &\ 0.\overset{w}{2}5[0 + 0.9 \cdot 0.74] \\ &+ 0.\overset{n}{2}5[0 + 0.9 \cdot 2.25] \\ &+ 0.\overset{e}{2}5[0 + 0.9 \cdot 0.30] \\ &+ 0.\overset{s}{2}5[0 + 0.9 \cdot -0.35] = 0.67 \end{aligned}$$

3.31	8.79	4.43	5.32	1.49
1.52	2.99	2.25	1.91	0.55
0.05	0.74	0.67	0.36	-0.40
-0.97	-0.44	-0.35	-0.59	-1.18
-1.86	-1.34	-1.23	-1.42	-1.97

Optimal Value Functions:

1. A policy π^* is defined to be better than or equal to a policy π, if its expected return is greater than or equal to that of π **for all states:**
$$\pi^* \geq \pi \quad \Leftrightarrow \quad V^{\pi^*}(s) \geq V^{\pi}(s) \quad \forall s \in S$$

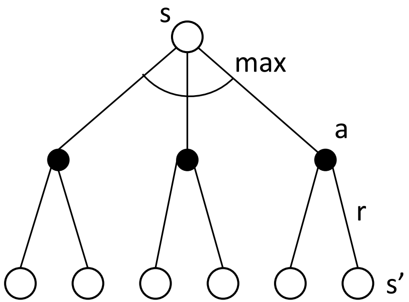
2. There is always **at least** one policy (a.k.a optimal policy) that is better than or equal to all other policies:
$$V^*(s) = \max_{\pi} V^{\pi}(s) \quad \forall s \in S$$

3. Optimal policies also share the same optimal action-value function, defined as:
$$Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a) \quad \forall s \in S, a \in A(s)$$

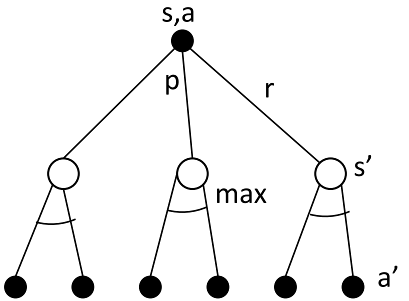
4. Since the action value is simply the return for taking action a in state s and thereafter following an optimal policy, we can write:
$$Q^*(s,a) = \mathrm{E}[\mathrm{r}_{t+1} + \gamma V^*(s_{t+1}) | s_t = s, a_t = a]$$

Bellman Optimality Equation

1. Since $V^*(s)$ is the value function for a policy, it must satisfy the Bellman Equation
2. When using the optimal policy we call the equation the **Bellman Optimality Equation**.
3. Intuitively, the Bellman Optimality Equation expresses the fact that the value of a state under an optimal policy must equal the expected return for the best action from that state:



$$V^*(s) = \max_a E[r_{t+1} + \gamma V^*(s_{t+1}) | S_t = s, A_t = a]$$
$$= \max_a \sum_{s',r} p(s', r | s, a)[r + \gamma V^*(s')]$$



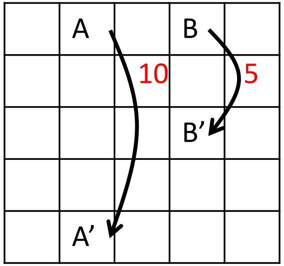
$$Q^*(s,a) = E[r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1}, a') | S_t = s, A_t = a]$$
$$= \sum_{s',r} p(s', r | s, a)[r + \gamma \max_{a'} Q^*(s', a')]$$

Bellman Optimality Equation

$$V^*(s) = \max_a E[r_{t+1} + \gamma V^*(s_{t+1}) | S_t = s, A_t = a]$$

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

$$V^*(0,0) = \max \begin{array}{c} w \\ n \\ e \\ s \end{array} \begin{bmatrix} -1 + 0.9 \cdot 21.98 \\ -1 + 0.9 \cdot 21.98 \\ 0 + 0.9 \cdot 24.42 \\ 0 + 0.9 \cdot 19.78 \end{bmatrix} = 21.98$$



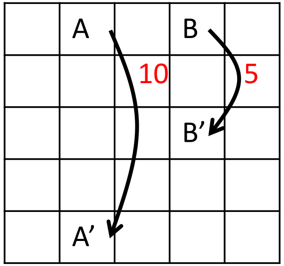
21.98	24.42	21.98	19.42	17.48
19.78	21.98	19.78	17.80	16.02
17.80	19.78	17.80	16.02	14.42
16.02	17.80	16.02	14.42	12.98
14.42	16.02	14.42	12.98	11.68

Bellman Optimality Equation

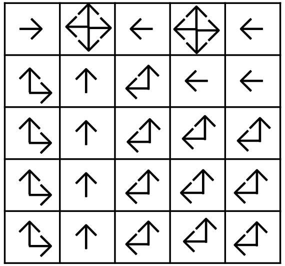
1. Why optimal state value functions are useful?
   - Any policy which is greedy with respect to V^*, is an optimal policy.
     - Why?
2. Therefore, given V^*, one-step-ahead search produces the long-term optimal actions. (This is a deterministic policy)

$$\pi^*(s) = \underset{a \in A}{argmax} \left[\sum_{s',r} p(s', r | s, a)[r + \gamma V^*(s')] \right]$$

3. E.g. back to the grid world:

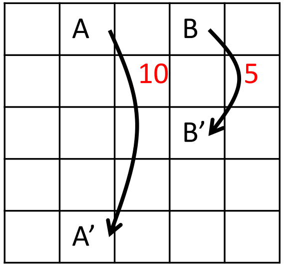


21.98	24.42	21.98	19.42	17.48
19.78	21.98	19.78	17.80	16.02
17.80	19.78	17.80	16.02	14.42
16.02	17.80	16.02	14.42	12.98
14.42	16.02	14.42	12.98	11.68

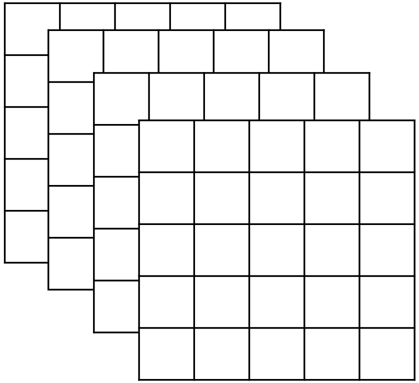


Bellman Optimality Equation

1. Given $Q^*(s,a)$, the agent does not even need to do a one-step-ahead search:
$$\pi^*(s) = \underset{a \in A}{\operatorname{argmax}}\, Q^*(s,a)$$
2. If the function is known, agent does not need to know anything about the dynamics of the environment
3. What are the implementation tradeoffs?



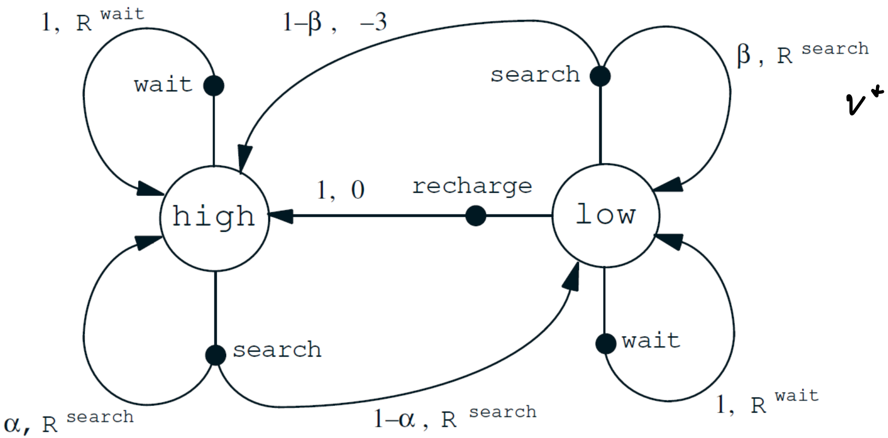
21.98	24.42	21.98	19.42	17.48
19.78	21.98	19.78	17.80	16.02
17.80	19.78	17.80	16.02	14.42
16.02	17.80	16.02	14.42	12.98
14.42	16.02	14.42	12.98	11.68



Bellman Optimality Equations Example

$$V^*(s) = \max_a E[r_{t+1} + \gamma V^*(s_{t+1}) | S_t = s, A_t = a]$$
$$= \max_a \sum_{s',r} p(s', r | s, a)[r + \gamma V^*(s')]$$

h: high
l: low
s: search
w: wait
re: recharge



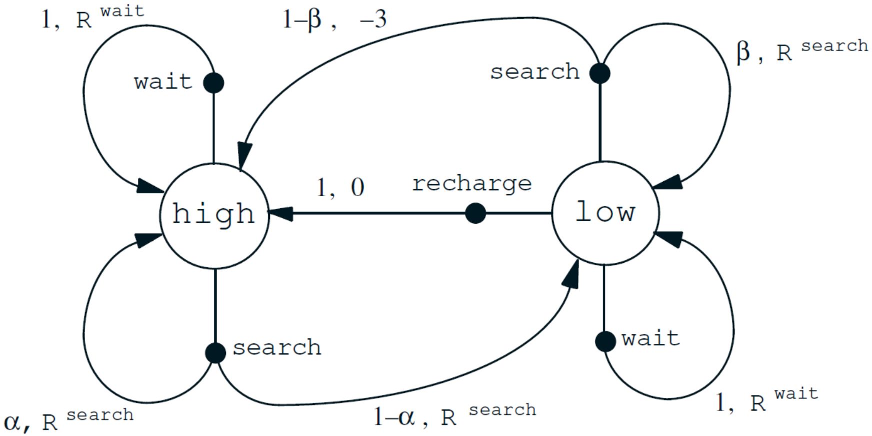
$$V^*(h) = \max_a \begin{cases} \alpha(R^{search} + \gamma V^*(h)) + (1-\alpha)(R^{search} + \gamma V^*(l)) \\ R^{wait} + \gamma V^*(h) \end{cases}$$

Bellman Optimality Equations Example

$$Q^*(s,a) = E[r_{t+1} + \gamma \max_{a'} Q^*(s_{t+1},a') | S_t = s, A_t = a]$$

$$= \sum_{s',r} p(s',r|s,a)\left[r + \gamma \max_{a'} Q^*(s',a')\right]$$

h: high
l: low
s: search
w: wait
re: recharge



Solving the Bellman Optimality Equation

1. Finding an optimal policy by solving the Bellman Optimality Equation (for example with dynamic programming) requires the following:
    - Accurate knowledge of the environment dynamics (environment model)
    - Enough space and time to do the computation
    - The Markov property
2. How much space and time do we need:
    - Polynomial in the number of states (will discuss in next chapter)
    - However, number of states can be extremely large (e.g. backgammon has about 10^{20} states)
3. We usually have to settle for approximations
4. Many RL methods can be understood as approximately solving the Bellman Optimality Equation.
    - Advantages of approximation?
        - learn more effectively
        - feature extraction can reduce noise
        - can address large scale problems

Solving the Bellman Optimality Equation

1. During the next few weeks we'll talk about techniques for solving the RL problem:
   - Dynamic Programming – well developed mathematically, solves the Bellman equations, but requires an accurate model of the environment.
   - Monte Carlo Methods – Do not require an exact model, but work only on episodic tasks since returns only calculated at the end.
   - Temporal Difference Learning – Do not require an exact model and can work on non episodic tasks.
     - More complex to analyze
     - Launched the revisiting of RL as a pragmatic framework (1988)
2. All methods differ in efficiency and speed of convergence to the optimal solution.

Summary

1. Agent environment interaction
   - states
   - actions
   - rewards
2. Policy:
   - $\pi(a|s)$
   - probability of selecting action in state
3. Return: (weighted) sum of future rewards
   - episodic vs. continuing tasks
4. Markov property
5. Markov Decision Process:
   - Transition probability
   - Expected reward
6. Value Functions
   - State value function for a policy
   - Action value function for a policy
7. Bellman Equations
8. Optimal value functions
9. Optimal Policies
10. The need for approximation