CSPB3202 Artificial Intelligence

MDP and RL

University of Colorado Boulder

Note: Some slides were adapted from the materials from Prof. Pieter Abbeel and Prof. Dan Klein at UC Berkeley, and Russell & Norvig's AI book.

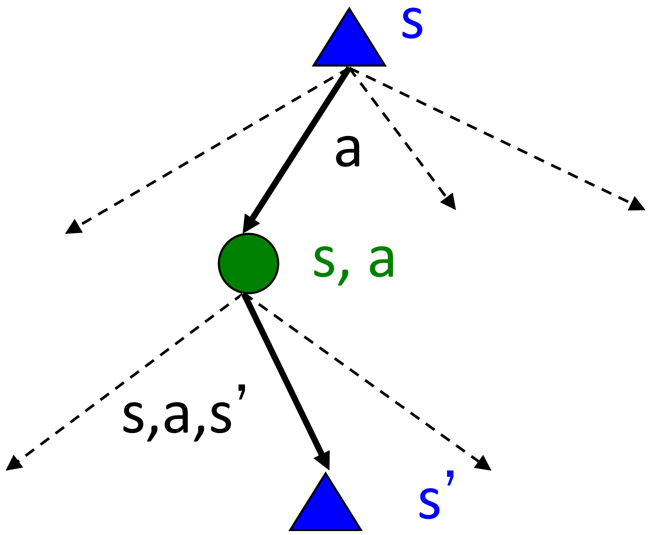
MDP: Non-deterministic Search

Recap: Defining MDPs

- Markov decision processes:
  - Set of states S A1 : finite states
  - Start state s_0
  - Set of actions A A2 : known
  - Transitions $P(s'|s,a)$ (or $T(s,a,s')$) → known
  - Rewards $R(s,a,s')$ (and discount γ) $R(s)$ +1 −1 living cost

- MDP quantities so far:
  - Policy = Choice of action for each state
  - Return (Utility) = sum of (discounted) rewards

$$G_t = \sum_{t=0}^{\infty} \gamma^t r_t$$



Solving MDP

Dynamic Programming

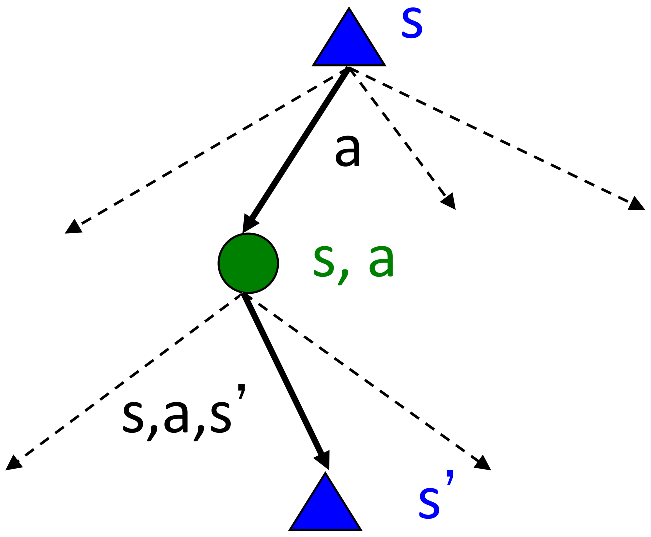
Finite MDP – All s known (fully observable)

– s are finite

– $P(s'|s,a)$ is known, perfect

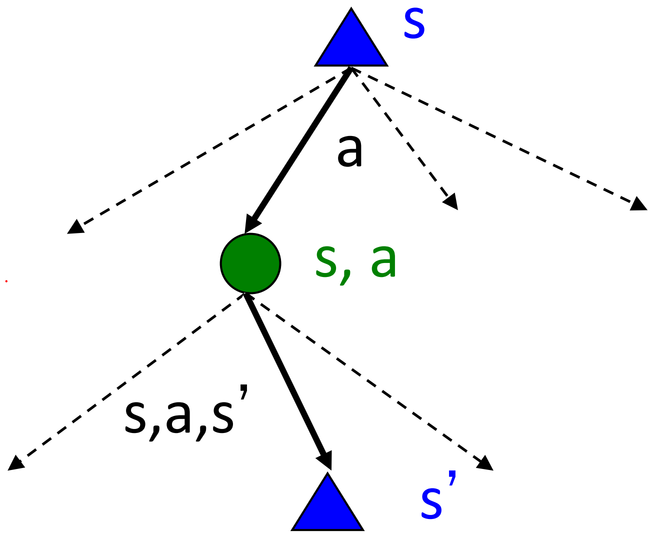
model of the world

Exactly solvable 10^{20} → Approximations



Value of a state

- Set of states S
- Set of actions A
- Transitions P(s’|s,a) (or T(s,a,s’))
- Rewards R(s,a,s’) (and discount γ)



G_t = Return = $\sum_{k=0} \gamma^k r_t = r_0(s) + \gamma \cdot r_1 + \gamma^2 r_2 + \cdots$

$V^\pi(s) = E_\pi[G_t \mid S_t = s]$: expected return in s

Value ft

π : policy $\Rightarrow$ $\pi(a \mid s)$ [a, ...]

action-value ft (Q-ft)

$Q^\pi(s,a) = E_\pi[G_t \mid S_t = s, A_t = a]$

$V^\pi(s) = \sum_a \pi(a \mid s) Q^\pi(s,a)$

$Q^\pi(s,a) = \sum_{s',r} P(s',r \mid s,a) V^\pi(s')$

Value Functions

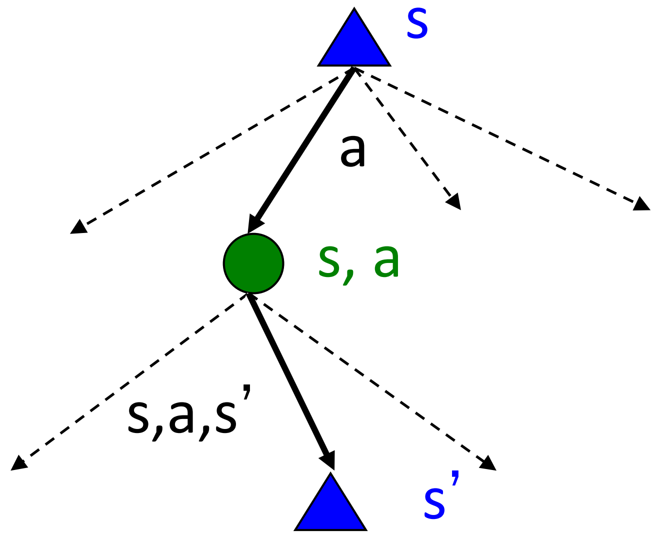
Value ft $\quad V_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$

Action-Value ft $\quad Q^\pi(s,a) = \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a]$

Return $\quad G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$

Policy $\quad \pi(a|s)$

$$\begin{cases} V_\pi(s) = \sum_a \pi(a|s)\, Q^\pi(s,a) \\ Q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)\, V_\pi(s') \end{cases}$$



Goal

Sequence of action $\quad$ soln

$\pi(a|s_0) \rightarrow s_t \quad \boxed{a_0, a_1, a_2 \cdots}$

maximize G_t

Bellman Equation

$$V_\pi(s) = \mathop{E}_{\pi}[G_t \mid s]$$

$$= \mathop{E}_{\pi}[r_t + \gamma G_{t+1} \mid s_t = s]$$

$$= R(s) + \gamma \mathop{E}_{\pi}[G_{t+1} \mid s_t = s]$$

$s \to s'$

$$\boxed{V_\pi(s) = R(s) + \gamma \sum_a \pi(a \mid s) \sum_{s'} P(s' \mid s, a) V_\pi(s')}$$

Bellman Eqn V_π

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} = r_t + G_{t+1}$$

$$= r_t + \underbrace{\boxed{\sum_{k=1}^{\infty} \gamma^k r_{t+k}}}_{G_{t+1}}$$

Bellman Equation

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

AIMA

$$U \quad V^{\pi}(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi(s)) V^{\pi}(s')$$

RL (by SB)

$$s \xrightarrow{r} s'$$

$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s',r} P(s',r|s,a) (r + \gamma V_{\pi}(s'))$$

Optimal Quantities

$V^{\pi}(s) = \sum_a \pi(a|s)\, q^{\pi}(s,a)$

- The value (utility) of a state s:

  $V^*(s)$ = expected utility starting in s and acting optimally

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

- The value (utility) of a q-state (s,a):

  $Q^*(s,a)$ = expected utility starting out having taken action a from state s and (thereafter) acting optimally

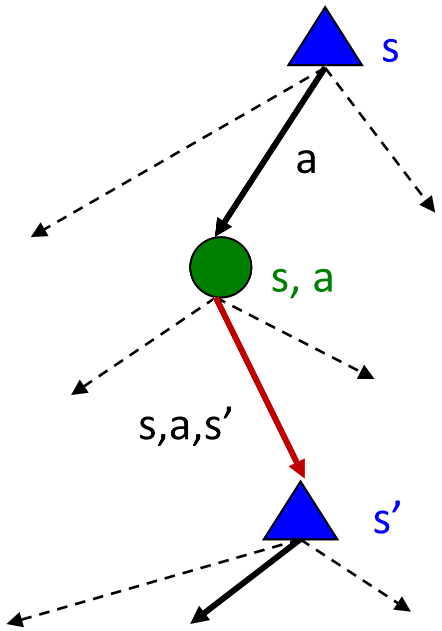
  $q^*(s,a) = \max_\pi q^\pi(s,a)$

- The optimal policy:

  $\pi^*(s)$ = optimal action from state s

  $\pi^*(s) = \arg\max_a \sum_{s'} P(s'|s,a)\, V_\pi(s')$

  $V^*(s) = \max_a q^*(s,a)$



s is a *state*

(s, a) is a *q-state*

(s,a,s') is a *transition*

Bellman Optimality Equation

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

$$V^* = \max_{\pi} V_{\pi}$$

$$q^* = \max_{\pi} q_{\pi}$$

$$V^* = \max_{a} q^*$$

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

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

Solving Bellman Equation

Dynamic Programming

1) Value Iteration

2) Policy Iteration ✓

Solving using Policy

init $\left(\begin{array}{l} V = 0 \\ \pi \to \text{random} \end{array} \right.$

1) evaluate policy $V(s)$

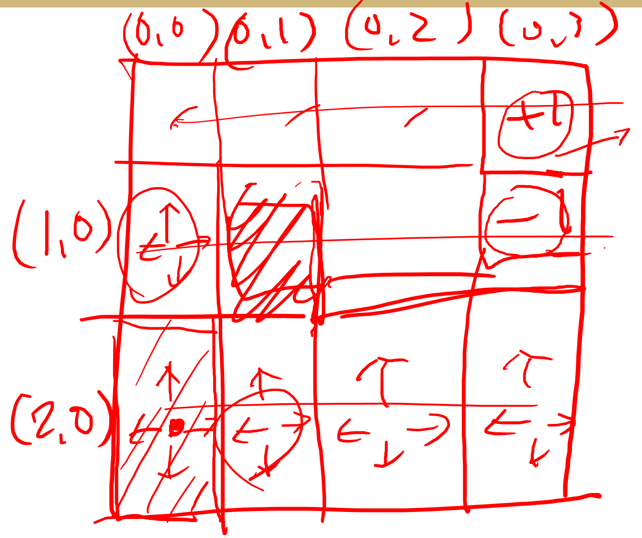
$$V^\pi = \sum_a \pi(a|s) \cdot \sum_{s',r} P(s',r|s,a)\left(r + \gamma V^\pi(s')\right)$$

0.25

2) Policy update

$$\max_a \sum_{s'} P(s'|s,a) V_\pi(s') \quad \sum_{s'} P(s'|s,\pi) V_\pi(s')$$

$$\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s'} P(s'|s,a) V_\pi(s')$$



Policy Iteration

Eval π

update π

Policy Iteration

Value Iteration

init $V = 0 \quad \forall s$

for i

$$V_{i+1}(s) \leftarrow r(s) + \gamma \max_a \sum_{s'} P(s' | s, a) V_i(s')$$

$$V_k \simeq V_{k-1}$$

$$V_k - V_{k-1} < \Delta$$

$$\pi(s) = \underset{a}{\operatorname{argmax}} \sum_{s'} P(s' | s, a) (r + \gamma V(s'))$$

Comparison

- Both value iteration and policy iteration compute the same thing (all optimal values)

- In value iteration:
  - Every iteration updates both the values and (implicitly) the policy
  - We don't track the policy, but taking the max over actions implicitly recomputes it

- In policy iteration:
  - We do several passes that update utilities with fixed policy (each pass is fast because we consider only one action, not all of them)
  - After the policy is evaluated, a new policy is chosen (slow like a value iteration pass)
  - The new policy will be better (or we're done)

- Both are dynamic programs for solving MDPs

Note on Dynamic Programming

Asynchronous DP

$$\boxed{|V_{i+1}(s) - V_i(s)|}$$

{ finite MDP
perfect model (T)

DP
- Policy Iter
- Value Iter

[$V(s) \Rightarrow$ Exp Return
$Q(s,a)$ "
π^*] Bellman Eqn