

DYNAMIC

PROGRAMMING

DYNAMIC PROGRAMMING

- Assume full knowledge of dynamics
- Two classes of problems:

① PREDICTION: - Inputs: Dynamics, cost, γ , policy π
- Output: Value function V^π

② CONTROL: - Input: Dynamics, cost, γ
- Outputs: Optimal Value function V^*
and optimal policy π^*

FINITE VS INFINITE HORIZONS

- Finite horizon:
 - theory is simpler
 - mostly used in optimal control
 - policy and Value depend on time
- Infinite horizon:
 - theory is more complex, but elegant
 - mostly used in RL
 - policy and Value are stationary
 - good approximation of problems with finite but long horizon

BELLMAN OPERATORS

- Bellman (expectation backup) operator T^π :

$$T^\pi(V(x)) = l(x, \pi(x)) + \gamma V(f(x, \pi(x)))$$

- Bellman optimality (backup) operator T :

$$(TV)(x) = \min_{u \in \mathcal{U}} l(x, u) + \gamma V(f(x, u))$$

PREDICTION: ITERATIVE POLICY EVALUATION

- Problem: evaluating a given policy π
- Solution: apply iteratively Bellman expectation operator:

$$V_{n+1}(x) = l(x, \pi(x)) + \gamma V_n(f(x, \pi(x))) \quad \forall x \in \mathcal{X}$$

- In short: $V_{n+1} = T^\pi(V_n)$
- Start from arbitrary estimate of Value function V_0
- Guaranteed to converge to V^π : $\lim_{n \rightarrow \infty} V_n = V^\pi$

When $\|V_{n+1} - V_n\| \leq \text{thr} \Rightarrow \text{STOP ALG.}$

Suppose $V_0(x) = 0 \quad \forall x \in \mathcal{X}$

$$V_1(x) = l(x, \pi(x)) + \gamma V_0(x') = l(x, \pi(x)) \quad \forall x \in \mathcal{X}$$

$$\begin{aligned} V_2(x) &= l(x, \pi(x)) + \gamma V_1(x') = \\ &= l(x, \pi(x)) + \gamma l(x', \pi(x')) \end{aligned}$$

$$\begin{aligned} V_3(x) &= l(x, \pi(x)) + \gamma V_2(x') = \\ &= l(x, \pi(x)) + \gamma [l(x', \pi(x')) + \gamma l(x'', \pi(x''))] = \\ &= l(x, \pi(x)) + \gamma l(x', \pi(x')) + \gamma^2 l(x'', \pi(x'')) \end{aligned}$$

CONTROL: POLICY ITERATION

- Problem: Finding optimal policy π^*
- Solution:
 - Start with arbitrary policy π_0
 - For $k = 0 \dots \infty$
 - Evaluate $\pi_k \Rightarrow V^{\pi_k}$
 - Improve policy acting greedily w.r.t. V^{π_k} :
$$\pi_{k+1}(x) = \arg \min_u l(x, u) + \gamma V^{\pi_k}(f(x, u))$$
- Always converges to π^* : $\lim_{k \rightarrow \infty} \pi_k = \pi^*$

MODIFIED POLICY ITERATION

- Evaluating exactly a policy can take many iterations
- What if we compute an approximation of V^{π_k} ?
 - Use m_k policy evaluation iterations to compute V^{π_k}
- Under some mild assumptions it converges to V^*
- For $m_k = \infty$ we recover Policy Iteration
- For $m_k = 1$ we get Value Iteration
 - Every time you update V you use a policy that's greedy w.r.t. V :

$$V(x) = l(x, \pi^k) + \gamma V^{k-1}(f(x, \pi^k)) \quad \text{with} \quad \pi^k(x) = \arg \min_u l(x, u) + \gamma V^{k-1}(f(x, u))$$

$$\Rightarrow V(x) = \min_u l(x, u) + \gamma V^{k-1}(f(x, u))$$

CONTROL: VALUE ITERATION

Algorithm:

- Start with arbitrary value function V_0
 - For $k = 0 \dots \infty$
 - For any $x \in \mathcal{X}$
 - $V_{k+1}(x) = \min_{u \in \mathcal{U}} l(x, u) + \gamma V_k(f(x, u))$
- $\left. \vphantom{\min_{u \in \mathcal{U}}} \right\} V_{k+1} = T(V_k)$
- V_k converges to V^* : $\lim_{k \rightarrow \infty} V_k = V^*$
 - Optimal policy π^* is computed at the end from V^* :

$$\pi^* = \arg \min_{u \in \mathcal{U}} l(x, u) + \gamma V^*(f(x, u))$$