Solving MDPs

Value Iteration

Repeated Bellman updates:

for all States s **do**
  $U(s) \leftarrow R(s)$
while unsatisfied **do**
  for each state s **do**
    $U'(s) \leftarrow R(s) + \gamma \max_a \sum_{s'} T(s,a,s') U(s')$
  $U \leftarrow U'$

Utility values are guaranteed to converge with enough updates
This equilibrium gives an optimal policy

When To Stop

Eventually updates have diminishing returns

$$||U_{i+1} - U_i|| = \text{max difference of } U \text{ between iterations}$$

$$||U_{i+1} - U_i|| < \varepsilon(1-\gamma)/\gamma \implies ||U_{i+1} - U^*|| < \varepsilon$$

Policy loss: the most utility an agent loses by following policy π_i instead of π^*
Policy is often optimal many iterations before U_i converges on optimal

Policy Iteration

U: utility for all states
π: action policy for all states

$\textsc{PolicyIter}(mdp)$

---

repeat
  $U \leftarrow \texttt{PolicyEval}(\pi, U, mdp)$
  $unchanged \leftarrow \text{true}$
  for all States s **do**
    $a^* \leftarrow \text{argmax}_{a \in A(s)} QVal(mdp, s, a, U)$
    if $QVal(mdp, s, a^*, U) > QVal(mdp, s, \pi[s], U)$ **then**
      $\pi[s] \leftarrow a^*$
      $unchanged \leftarrow \text{false}$
until $unchanged$
return π

Reinforcement Learning

Definition

Build a good policy based on experience only: (s, a, s', r)
Objective:

- **finite horizon**: $R(s_0) + R(s_1) + R(s_2)$, or
- **infinite horizon**: $R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) + \ldots$

Learning because we don't start with a model of the system

Solving MDPs

Reinforcement Learning

Definition

ADP

Sweeping

Adaptive Dynamic Programming

- Learn T and R functions through trials
- Easy approach to find rewards from each state, and transition probabilities
- Calculate π using MDP solution

Update utility and policy of all states until satisfied

Prioritized Sweeping

Focus on areas of model where large changes are expected

UPDATE(s, a, s', r)

- update model
- UPDATE(s)
- Do k times: update highest priority

UPDATE(s)

- update $U(s)$
- priority of s $\leftarrow 0$
- **for all** States s' that are predecessors of s **do**
  - priority of $s' \leftarrow \max(current, \max_a \delta \hat{T}(s', a, s))$