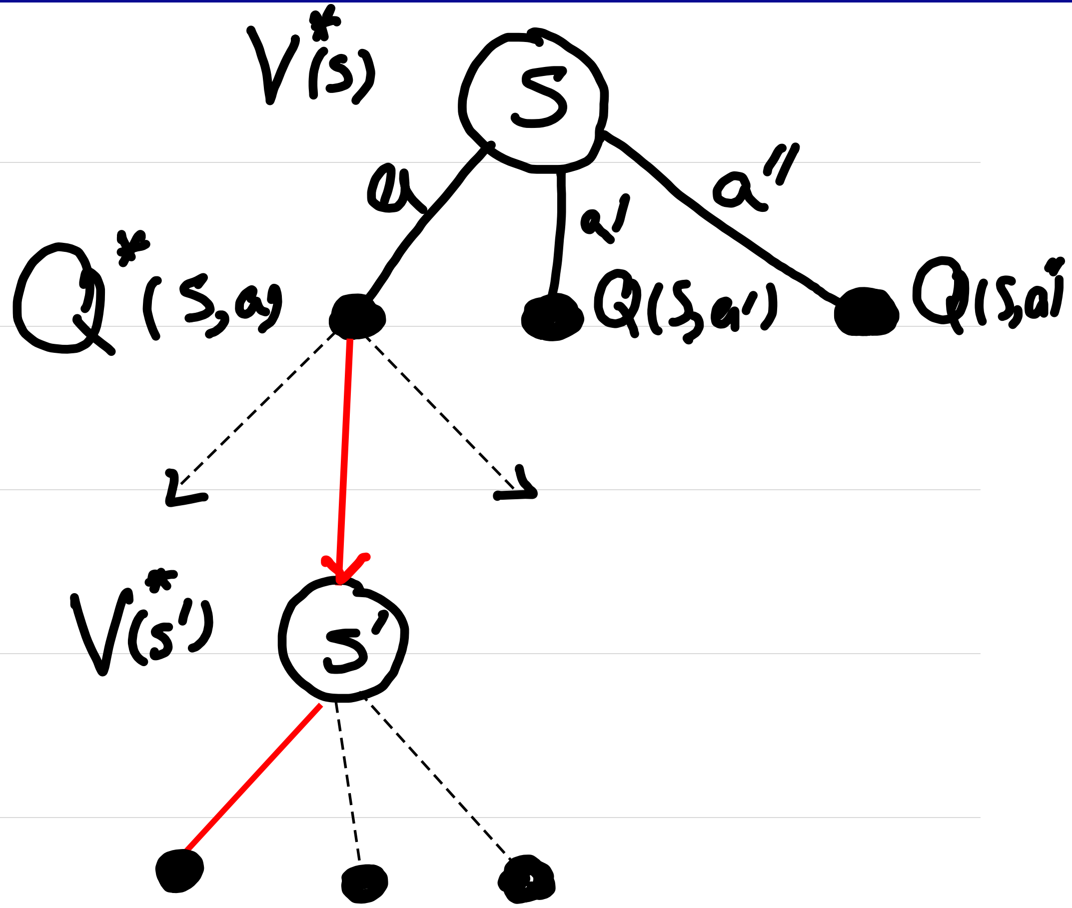
Welcome to Week 12: Solving MDP

- Outline:
  - Solving MDP: Computing the best policy
    - The problem with Value iteration
    - Policy based method: Policy Evaluation + policy Improvement

Problems with Value Iteration

- Value iteration repeats the Bellman updates:

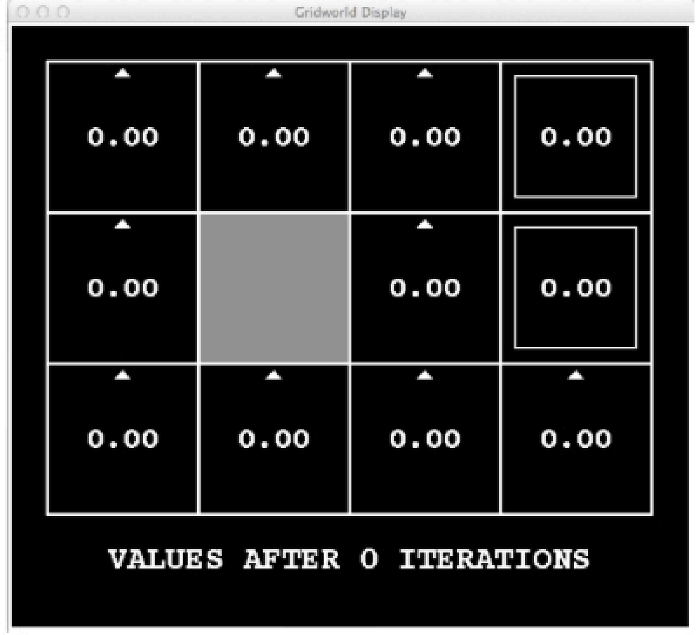
$$V_{k+1}(s) \leftarrow \max_a \sum_{s' \in S} T(s,a,s')\left[R(s,a,s') + \gamma V_k(s')\right]$$



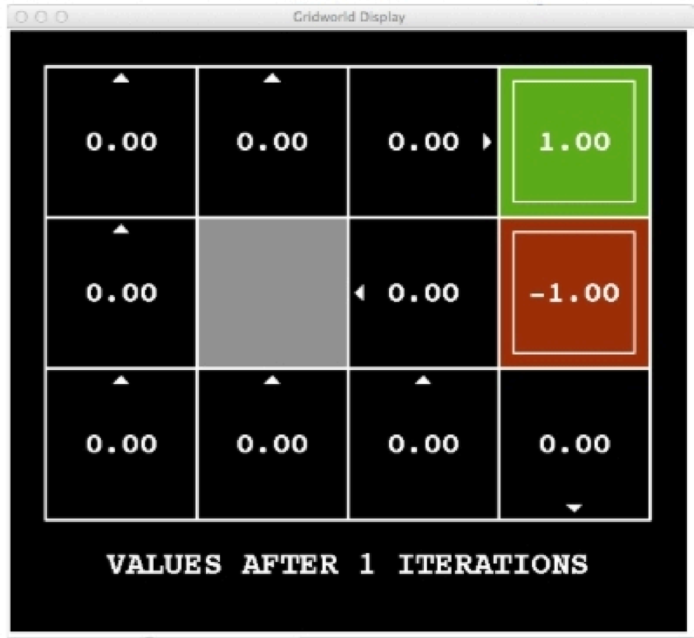
- Problem 1: It's slow. $O(|S|^2|A|)$ per iteration
- Problem 2: The max at each state rarely changes
  The policy converges long before the values
-
-
-

The Policy Converges Long Before the Values

K=0



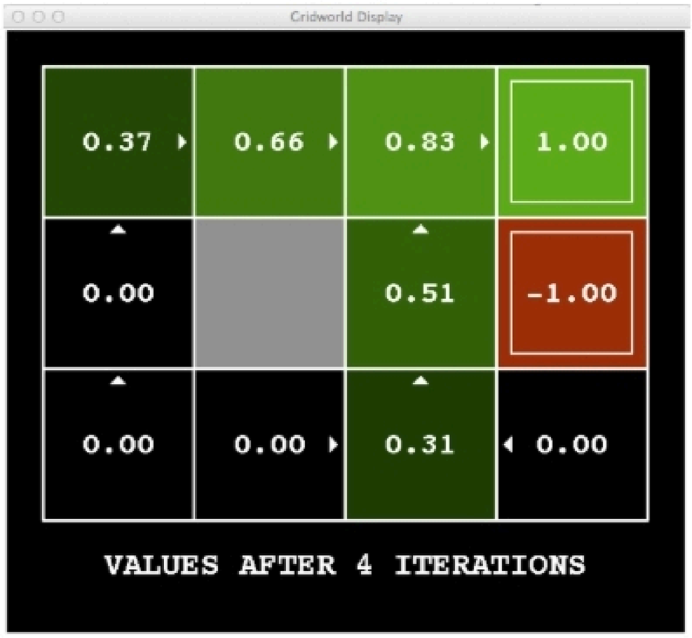
K=1



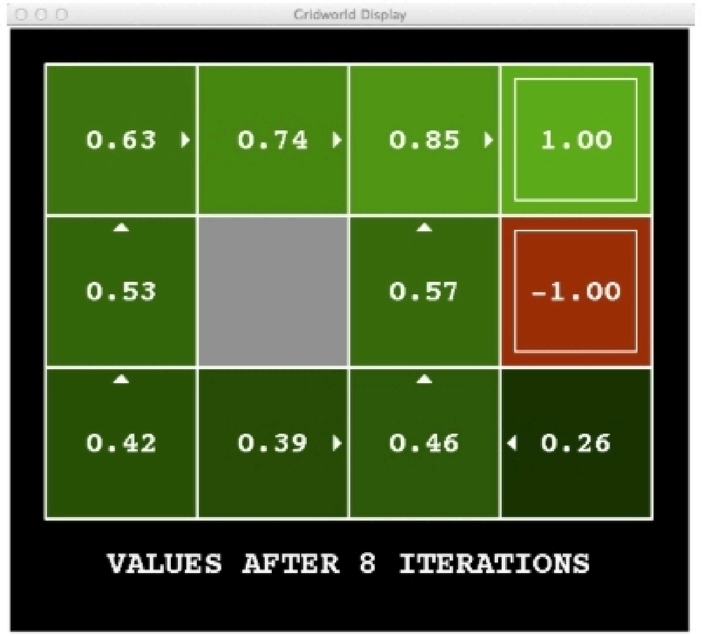
K=2

K=3

K=4

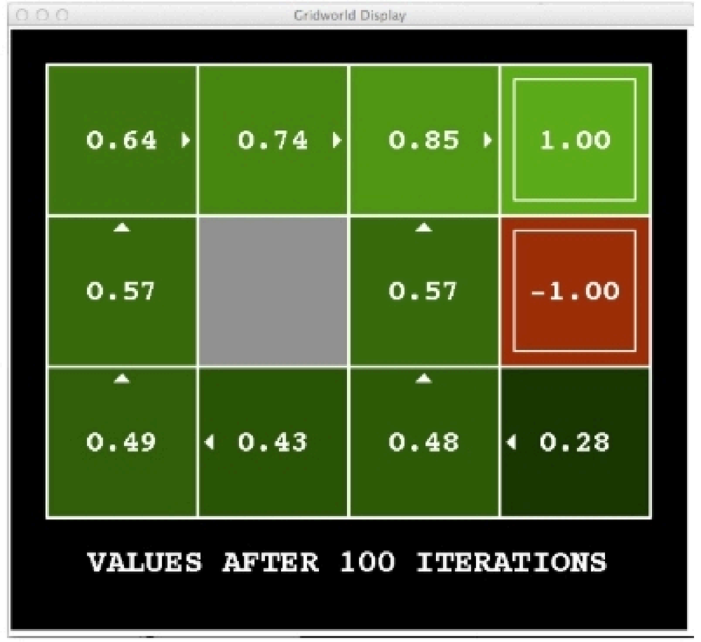


K=8



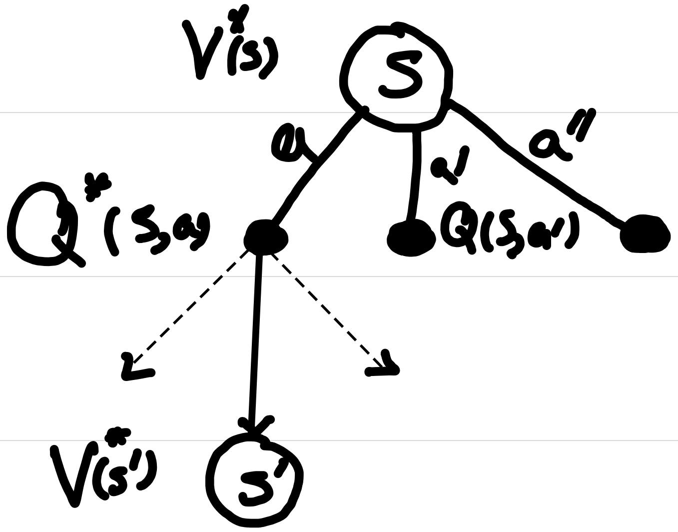
K=12

K=100



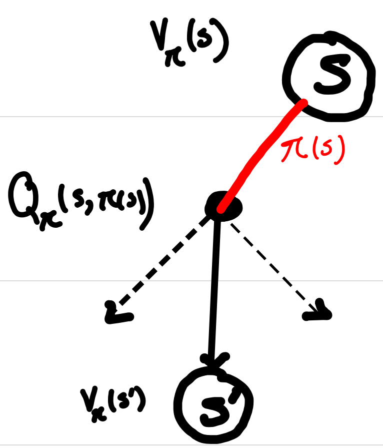
Policy Evaluation

- In Bellman Equation, we didn't have a fixed policy.
  - Hence, we had to max over all actions in the Bellman equation.



$$V^*(s) = \max_{a \in A} \sum_{s' \in S} T(s,a,s') \left[R(s,a,s') + \gamma V^*(s') \right]$$

- If we have a fixed policy π, what's the recursive relation of $V_\pi(s)$?



$$V_\pi(s) =$$

Policy Evaluation

- How do we compute the V's for a fixed policy π?
  - Idea 1: Turn recursive Bellman equations into updates (like value iteration)

    Initialize $V_{\pi,0}(s) = 0$, for all $s \in S$

    For $k=1$, until convergence:

    For all $s \in S$:

    $$V_{\pi,k+1}(s) \leftarrow \sum_{s' \in S} T(s,\pi(s),s')\left[R(s,\pi(s),s') + \gamma V_{\pi,k}(s')\right]$$

  - Computational complexity per iteration:
  - Idea 2: With fixed policy, we just have a linear system of equation. Solve with your favorite linear solver.

Policy Iteration

- By Combining "policy evaluation" and "policy extraction", we design a nice policy based method to find optimal policy
- Policy Iteration:
  - Step 1: Policy Evaluation; Calculate values for some fixed policy (not optimal values!) until Convergence.
  - Step 2: Policy Improvement; Given the Values from step 1, update policy using one-step look-ahead
  - Repeat steps until convergence.

- Policy Iteration:

Initialize $\pi_0(s)$ for all $s \in S$

For $i=1$, until Convergence:

Policy Evaluation

Initialize $V_{\pi_i,0}(s)=0$ for all $s \in S$

For $k=1$, until Convergence:

$$V_{\pi_i,k+1}(s) \leftarrow \sum_{s' \in S} T(s,\pi_i(s),s')\left[R(s,\pi_i(s),s') + \gamma V_{\pi_i,k}(s')\right]$$

Store the values after convergence as $V_{\pi_i}(s)$.

Policy Improvement

For all $s \in S$:

$$\pi_{i+1}(s) \leftarrow \arg\max_{a \in A} \sum_{s' \in S} T(s,a,s')\left[R(s,a,s') + \gamma V_{\pi_i}(s')\right]$$

Summary

- To compute the optimal values of an MDP, use
  - Value iteration
  - or, Policy iteration
- To compute values for a fixed policy, use
  - Policy evaluation
- To turn your values into a policy, use
  - policy extraction

Summary: They are All Just Variation of The Same Equation

- Optimal V and Q functions:

$$V^*(s) = \max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V^*(s')\right] \qquad \left(V^*(s) = \max_a Q^*(s,a)\right)$$

$$Q^*(s,a) = \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right]$$

- Value function for a fixed policy:

$$V_\pi(s) = \sum_{s'} T(s,\pi(s),s')\left[R(s,\pi(s),s') + \gamma V_\pi(s')\right]$$

- Policy π^* for V^* and Q^* function:

$$\pi^*(s) = \arg\max_a Q^*(s,a)$$

$$\pi^*(s) = \arg\max_a \sum_{s'} T(s,a,s')\left[R(s,a,s') + \gamma V^*(s')\right]$$

Comparison

- Both value iteration and Policy iteration Compute the same thing
- 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 no max.
  - After the policy is evaluated, a new policy is chosen
  - The new policy will be better (or we're done)

Both methods are dynamic Programs to Solve MDP's

-