Reinforcement Learning

Computer Engineering Department
Sharif University of Technology

Mohammad Hossein Rohban, Ph.D.

Hossein Hasani

Spring 2023

Courtesy: Some slides are adopted from CS 285 Berkeley, and CS 234 Stanford, and Pieter Abbeel's compact series on RL.

Value Iteration

- $V_0^*(s)$ = optimal value for state s when H=0
  - $V_0^*(s) = 0 \quad \forall s$
- $V_1^*(s)$ = optimal value for state s when H=1
  - $V_1^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_0^*(s'))$
- $V_2^*(s)$ = optimal value for state s when H=2
  - $V_2^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1^*(s'))$
- $V_k^*(s)$ = optimal value for state s when H = k
  - $V_k^*(s) = \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_{k-1}^*(s'))$

Value Iteration

Algorithm:

Start with $V_0^*(s) = 0$ for all s.

For k = 1, … , H:

For all states s in S:

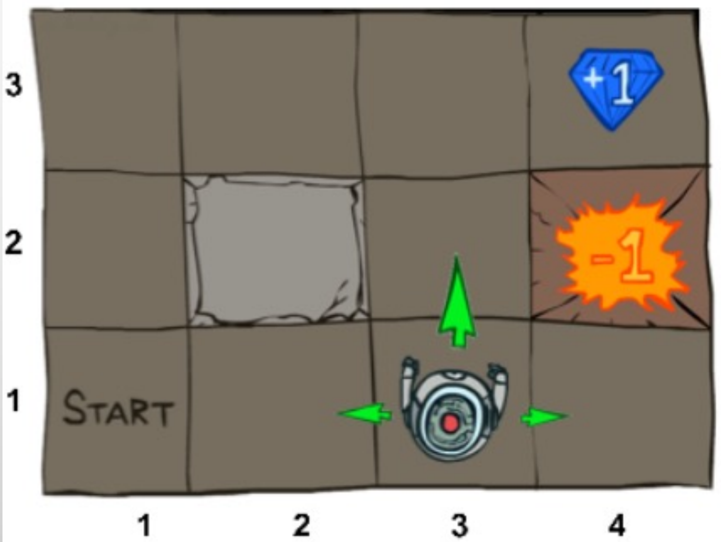
$$V_k^*(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)\left(R(s,a,s') + \gamma V_{k-1}^*(s')\right)$$

$$\pi_k^*(s) \leftarrow \arg\max_a \sum_{s'} P(s'|s,a)\left(R(s,a,s') + \gamma V_{k-1}^*(s')\right)$$

This is called a value update or Bellman update/back-up

Value Iteration

$$V_0(s) \leftarrow 0$$

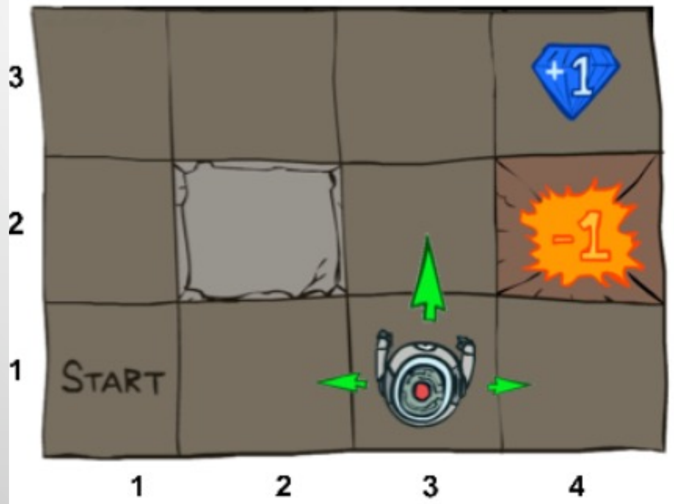


k = 0

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_1(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_0(s'))$$



Noise = 0.2
Discount = 0.9

k = 0

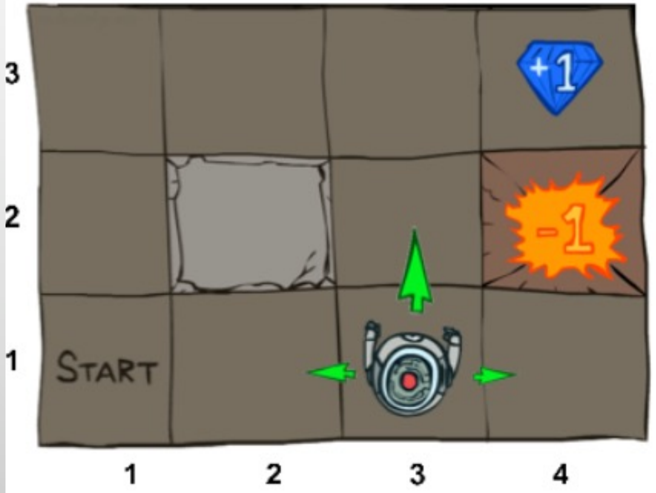
0.00	0.00	0.00	0.00
0.00		0.00	0.00
0.00	0.00	0.00	0.00

VALUES AFTER 0 ITERATIONS

Value Iteration

$$V_2(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1(s'))$$

k = 1



0.00	0.00	0.00	1.00
0.00		0.00	-1.00
0.00	0.00	0.00	0.00

VALUES AFTER 1 ITERATIONS

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_2(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_1(s'))$$

k = 2

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 3

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

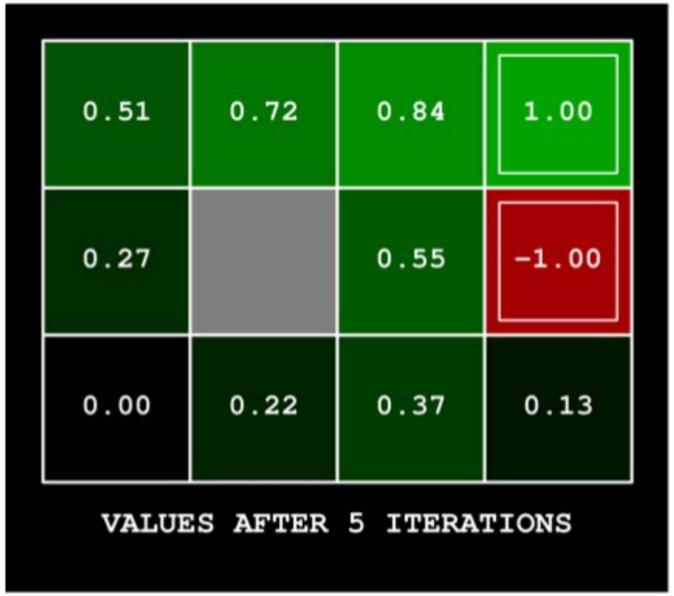
k = 4

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 5

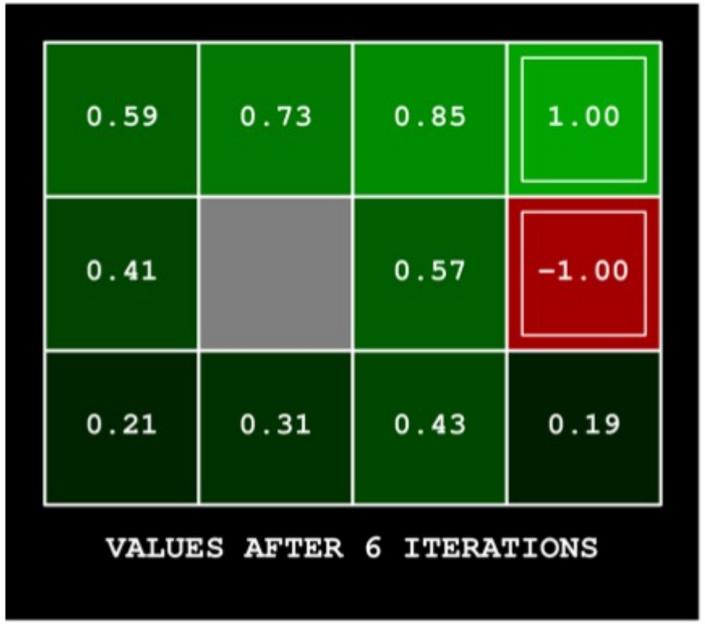


Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 6



Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)\left(R(s,a,s') + \gamma V_k(s')\right)$$

k = 7

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 8

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 9

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 10

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 11

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma V_k(s'))$$

k = 12

Noise = 0.2
Discount = 0.9

Value Iteration

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P(s'|s,a)\left(R(s,a,s') + \gamma V_k(s')\right)$$

k = 100

Noise = 0.2

Discount = 0.9

Value Iteration Convergence

Theorem. Value iteration converges. At convergence, we have found the optimal value function V* for the discounted infinite horizon problem, which satisfies the Bellman equations

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

Bellman Optimality Equation

Proof Sketch (special case)

- Assume $r \geq 0$
- $V_H(s)$ is a **bounded** and **increasing** sequence in H.
  - So it converges
- But $V_{H+1} = \max_a \sum_{s'} P(s'|s,a)[R(s,a,s') + \gamma V_H(s')]$ is a continuous function of $V_H(s)$.
- Taking limits of both sides yields the Bellman optimality equation.
- General case: Use **contraction mapping** idea (could be discussed at the recitation class)

Q-Values

- $Q^*(s,a)$ = expected utility starting in s, taking action a, and (thereafter) acting optimally

$$V^*(s) = \max_{a'} Q^*(s,a')$$

- Bellman Equation:

$$Q^*(s,a) = \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma \max_{a'} Q^*(s',a'))$$

- Q-value Iteration:

$$Q^*_{k+1}(s,a) \leftarrow \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma \max_{a'} Q^*_k(s',a'))$$

Q-Value Iteration

$$Q^*_{k+1}(s,a) \leftarrow \sum_{s'} P(s'|s,a)(R(s,a,s') + \gamma \max_{a'} Q^*_k(s',a'))$$

k = 100

Noise = 0.2

Discount = 0.9

Policy Evaluation

- Recall value iteration:

$$V_k^*(s) \leftarrow \max_a \sum_{s'} P(s'|s,a) \left(R(s,a,s') + \gamma V_{k-1}^*(s')\right)$$

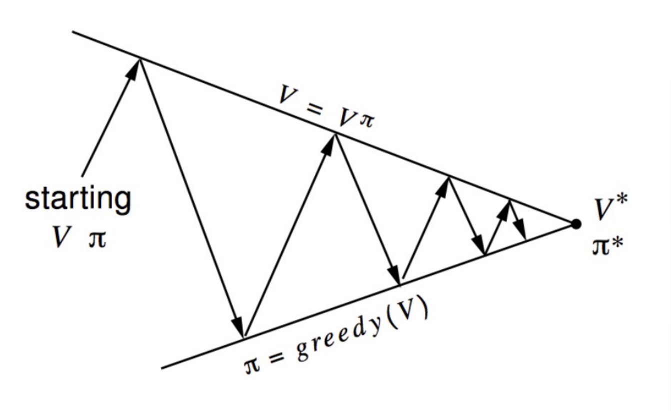
- Policy evaluation for a given $\pi(s)$:

$$V_k^\pi(s) \leftarrow \sum_{s'} P(s'|s,\pi(s))(R(s,\pi(s),s') + \gamma V_{k-1}^\pi(s))$$

At convergence:

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

Policy Iteration



- One iteration of policy iteration
  - Policy evaluation for current policy π_k :
    - Iterate until convergence

$$V_{i+1}^{\pi_k}(s) \leftarrow \sum_{s'} P(s'|s, \pi_k(s)) \left[R(s, \pi(s), s') + \gamma V_i^{\pi_k}(s')\right]$$

  - Policy improvement: find the best action according to one-step look-ahead

$$\pi_{k+1}(s) \leftarrow \arg\max_a \sum_{s'} P(s'|s, a) \left[R(s, a, s') + \gamma V^{\pi_k}(s')\right]$$

- Repeat until policy converges
  - At convergence: optimal policy; and converges faster than value iteration under some conditions

One-step look ahead improves the policy

- Consider an alternative policy $\pi_{(k+1)}^{(1)}(t,s)$ that takes the prescribed actions in $\pi_{k+1}(s)$ **only at time t = 0**; and stays the same as $\pi_k(s)$ in later times.
- The value function V(s) for this new policy is larger than or equal to V(s) for the original policy $\pi_k(s)$ for all s. Why?
- Now let $\pi_{(k+1)}^{(2)}(t,s)$, which takes the prescribed action in $\pi_{k+1}(s)$ **only at times t = 0 and t = 1**, and stays the same as $\pi_k(s)$ in later times.
- Similarly, V(s) gets improve for $\pi_{(k+1)}^{(2)}(t,s)$ compared to $\pi_{(k+1)}^{(1)}(t,s)$ for all s.
- Repeating this argument $\pi_{(k+1)}^{(\infty)}(t,s)$ becomes the same as $\pi_{k+1}(s)$.

Policy Iteration Guarantees

Policy Iteration iterates over:

- Policy evaluation
  - Iterate until convergence

$$V_{i+1}^{\pi_k}(s) \leftarrow \sum P(s'|s,\pi_k(s)) \left[R(s,\pi(s),s') + \gamma V_i^{\pi_k}(s')\right]$$

- Policy Improvement

$$\pi_{k+1}(s) \leftarrow \arg\max_a \sum_{s'} P(s'|s,a) \left[R(s,a,s') + \gamma V^{\pi_k}(s')\right]$$

Theorem. Policy iteration is guaranteed to converge and at convergence, the current policy and its value function are the optimal policy and the optimal value function!

Proof sketch:

(1) *Guarantee to converge*: In every step the policy improves. This means that a given policy can be encountered at most once. This means that after we have iterated as many times as there are different policies, i.e., (number actions)$^{\text{(number states)}}$, we must be done and hence have converged.

(2) *Optimal at convergence*: by definition of convergence, at convergence $\pi_{k+1}(s) = \pi_k(s)$ for all states s. This means $\forall s \ \ V^{\pi_k}(s) = \max_a \sum_{s'} T(s,a,s') \left[R(s,a,s') + \gamma V_i^{\pi_k}(s')\right]$

Hence V^{π_k} satisfies the Bellman equation, which means V^{π_k} is equal to the optimal value function V*.