

Markov Decision Processes

Chen-Yu Wei

Sequence of Actions



To win the game, the learner has to take a sequence of actions $a_1 \rightarrow a_2 \rightarrow \dots \rightarrow a_H$.

One option: view every sequence as a “meta-action”: $\bar{a} = (a_1, a_2, \dots, a_H)$

Drawback:

- The number of actions is exponential in horizon
- In stochastic environments, this does not leverage intermediate observations

Solution idea: dynamic programming

Interaction Protocol: Fixed-Horizon Case

For **episode** $t = 1, 2, \dots, T$:

For **step** $h = 1, 2, \dots, H$:

Learner observes an observation $x_{t,h}$

Learner chooses an action $a_{t,h}$

Learner receives instantaneous reward $r_{t,h}$

General case:

$$\mathbb{E}[r_{t,h}] = R(x_{t,1}, a_{t,1}, \dots, x_{t,h}, a_{t,h}), \quad x_{t,h+1} \sim P(\cdot \mid x_{t,1}, a_{t,1}, \dots, x_{t,h}, a_{t,h})$$

\Rightarrow Optimal decisions may depend on the entire history $\mathcal{H}_t = (x_{t,1}, a_{t,1}, \dots, x_{t,h})$

Interaction Protocol: Fixed-Horizon Case

For **episode** $t = 1, 2, \dots, T$:

For **step** $h = 1, 2, \dots, H$:

Learner observes an observation $x_{t,h}$

Learner chooses an action $a_{t,h}$

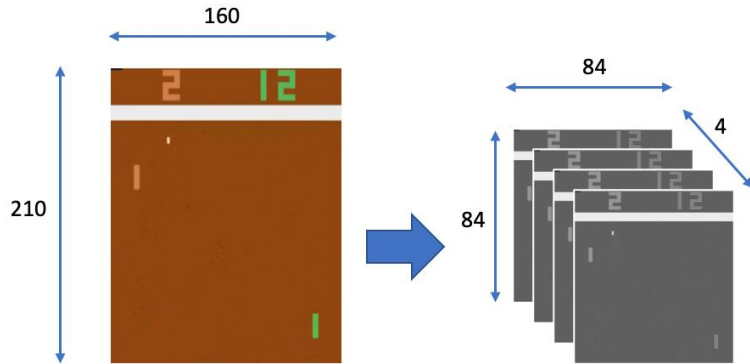
Learner receives instantaneous reward $r_{t,h}$

We assume that the history $\mathcal{H}_t = (x_{t,1}, a_{t,1}, \dots, x_{t,h})$ can be summarized as a **horizon-length-independent** representation $s_{t,h} = \Phi(x_{t,1}, a_{t,1}, \dots, x_{t,h}) \in \mathcal{S}$ so that

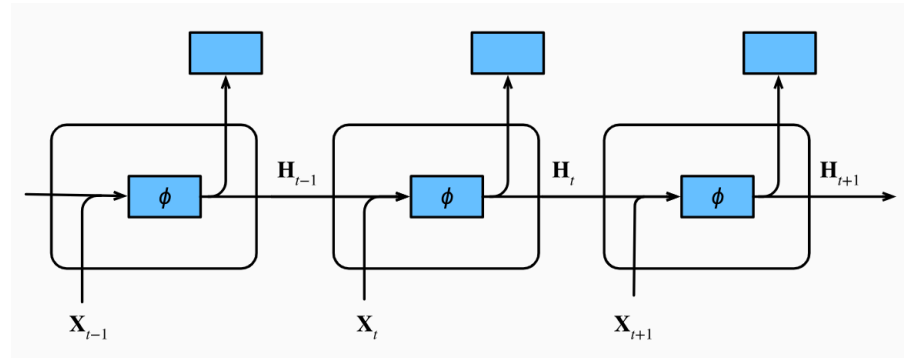
$$\mathbb{E}[r_{t,h}] = R(s_{t,h}, a_{t,h}), \quad x_{t,h+1} \sim P(\cdot \mid s_{t,h}, a_{t,h})$$

$s_{t,h}$ is called the “**state**” at the step h of episode t .

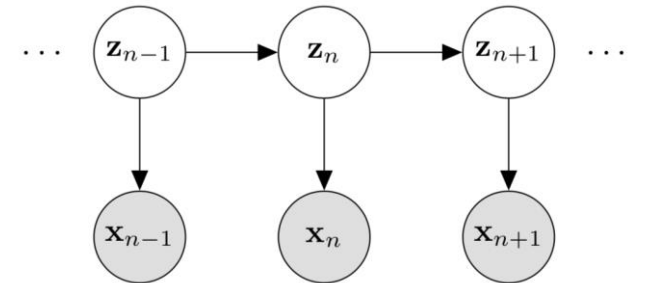
From Observations to States



Stacking recent observations



Recurrent neural network



Hidden Markov model

Interaction Protocol: Fixed-Horizon Case

For **episode** $t = 1, 2, \dots, T$:

For **step** $h = 1, 2, \dots, H$:

Environment reveals **state** $s_{t,h}$

Learner chooses an action $a_{t,h}$

Learner observes instantaneous reward $r_{t,h}$ with $\mathbb{E}[r_{t,h}] = R(s_{t,h}, a_{t,h})$

Next state is generated as $s_{t,h+1} \sim P(\cdot \mid s_{t,h}, a_{t,h})$

This is called the Markov decision process.

MDP as Contextual Bandits?

Viewing states as contexts, and viewing the problem as a contextual bandit problem with TH rounds (what's wrong?)

$$\begin{aligned}
 \text{Regret (Contextual bandit)} &= \sum_{t=1}^T \sum_{h=1}^H \underbrace{\max_a}_{\substack{\downarrow \\ (x_{t,1}, a_{t,1} \dots x_{t,h})}} R(s_{t,h}^*, a) - \sum_{t=1}^T \sum_{h=1}^H R(s_{t,h}, a_{t,h}) \\
 \text{Regret (MDP)} &= \sum_{t=1}^T \left[\sum_{h=1}^H R(s_{t,h}^*, a_{t,h}^*) \right] - \sum_{t=1}^T \sum_{h=1}^H R(s_{t,h}, a_{t,h})
 \end{aligned}$$

$$s_{t,1}^* = s_{t,1}$$

$$s_{t,h}^* \neq s_{t,h} \quad \text{for } h \geq 2$$

Formulations

- Interaction Protocol
 - Fixed-Horizon
 - Variable-Horizon (Goal-Oriented)
 - Infinite-Horizon
- Performance Metric
 - Total Reward
 - Average Reward
 - Discounted Reward
- Policy
 - History-dependent policy
 - Markov policy
 - Stationary policy

Horizon = Length of an episode

Interaction Protocols (1/3): Fixed-Horizon

Horizon length is a fixed number H

$h \leftarrow 1$

Observe initial state s_1

While $h \leq H$:

 Choose action a_h

 Observe reward r_h with $\mathbb{E}[r_h] = R(s_h, a_h)$

 Observe next state $s_{h+1} \sim P(\cdot | s_h, a_h)$

Examples: games with a fixed number of time

Interaction Protocols (2/3): Goal-Oriented

The learner interacts with the environment until reaching **terminal states** $\mathcal{T} \subset \mathcal{S}$

$h \leftarrow 1$

Observe initial state s_1

While $s_h \notin \mathcal{T}$:

 Choose action a_h

 Observe reward r_h with $\mathbb{E}[r_h] = R(s_h, a_h)$

 Observe next state $s_{h+1} \sim P(\cdot | s_h, a_h)$

$h \leftarrow h + 1$

Examples: video games, robotics tasks, personalized recommendations, etc.

Interaction Protocols (3/3): Infinite-Horizon

The learner continuously interacts with the environment

$h \leftarrow 1$

Observe initial state s_1

Loop forever:

Choose action a_h

Observe reward r_h with $\mathbb{E}[r_h] = R(s_h, a_h)$

Observe next state $s_{h+1} \sim P(\cdot | s_h, a_h)$

$h \leftarrow h + 1$

Examples: network management, inventory management

Formulations for Markov Decision Processes

- Interaction Protocol
 - Fixed-Horizon
 - Variable-Horizon (Goal-Oriented)
 - Infinite-Horizon
 - Performance Metric
 - Total Reward
 - Average Reward
 - Discounted Reward
 - Policy
 - History-dependent policy
 - Markov policy
 - Stationary policy
- } Episodic setting

Performance Metric

Total Reward (for episodic setting): $\sum_{h=1}^{\tau} r_h$ (τ : the step where the episode ends)

Average Reward (for infinite-horizon setting): $\lim_{T \rightarrow \infty} \frac{1}{T} \sum_{h=1}^T r_h$

Discounted Total Reward (for episodic or infinite-horizon): $\sum_{h=1}^{\tau} \gamma^{h-1} r_h \leq \frac{1}{1-\gamma}$ if $|r_h| \leq 1$.

τ : the step where the episode ends, or ∞ in the infinite-horizon case

$\gamma \in [0,1)$: discount factor

Interaction Protocols vs. Performance Metrics

Fixed-Horizon	“natural” objective ----->	Total Reward	
Goal-Oriented	----->	Total Reward	Could be unbonded
Infinite-horizon	----->	Average Reward	Could have constant change for an infinitesimal change in policy

Discounted Total Reward?

Focusing more on the **recent** reward

There is a potential mismatch between our ultimate goal and what we optimized.

Our Focus

In most of the following lectures, we focus on the **goal-oriented / infinite-horizon** setting with **discount total reward** as the performance metric.

Policy

A mapping from observations/contexts/states to (distribution over) actions

- Contextual bandits

$$a \sim \pi(\cdot \mid x) \quad (\text{randomized/stochastic})$$

$$\text{or } a = \pi(x) \quad (\text{deterministic})$$

- Multi-armed bandits

$$a \sim \pi$$

$$\text{or } a = a^*$$

Policy for MDPs

History-dependent Policy

$$a_h \sim \pi(\cdot \mid s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_h)$$
$$a_h = \pi(s_1, a_1, r_1, s_2, a_2, r_2, \dots, s_h)$$

Markov Policy

$$a_h \sim \pi(\cdot \mid s_h, h)$$
$$a_h = \pi(s_h, h)$$

← For **fixed-horizon + total reward** setting, there exists an optimal policy in this class

Stationary Policy

$$a_h \sim \pi(\cdot \mid s_h)$$
$$a_h = \pi(s_h)$$

← For **infinite-horizon/goal-oriented + discounted total reward** setting, there exists an optimal policy in this class

Fixed-Horizon + Total Reward

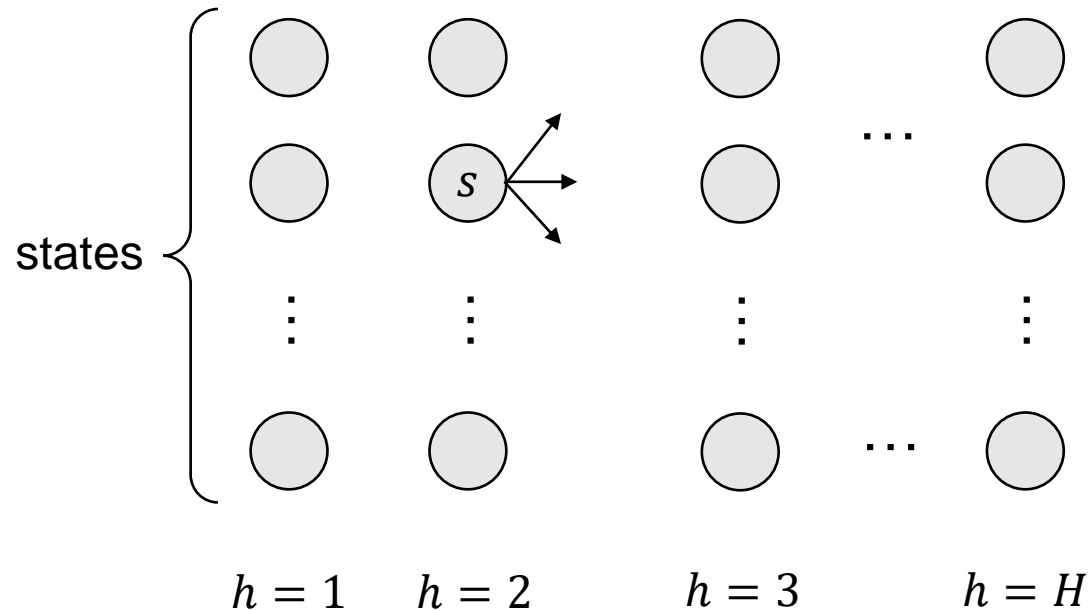
Dynamic Programming

Goal: Calculate the expected total reward of a policy

A (Markov) policy is a mapping from (state, step index) to action distribution, written as

$$\pi_h(\cdot | s) \in \Delta(\mathcal{A}) \quad \text{for } s \in \mathcal{S} \text{ and } h \in \{1, 2, \dots, H\}$$

Dynamic Programming



State transition: $P(s'|s, a)$

Reward: $R(s, a)$

$$V_h^\pi(s) = \mathbb{E} \left[\sum_{i=h}^H R(s_i, a_i) \mid s_h = s, a_i \sim \pi_i(\cdot | s_i) \forall i \geq h \right]$$

Key quantity: $V_h^\pi(s)$ = the expected total reward of policy π starting from state s at step h .

Backward calculation:

$$V_H^\pi(s) = \sum_a \pi_H(a|s) R(s, a) \quad \forall s$$

For $h = H - 1, \dots, 1$: for all s

$$V_h^\pi(s) = \sum_a \pi_h(a|s) \left(R(s, a) + \underbrace{\sum_{s'} P(s'|s, a) V_{h+1}^\pi(s')}_{\text{Expected total reward from step } h+1} \right)$$

Expected total reward from step $h+1$

Bellman Equation

$$Q_h^\pi(s, a) = \mathbb{E} \left[\sum_{i=h}^H R(s_i, a_i) \mid s_h = s, a_h = a, a_i \sim \pi_i(\cdot | s_i) \forall i \geq h+1 \right]$$

$$V_{H+1}^\pi(s) = 0$$

$$V_h^\pi(s) = \sum_a \pi_h(a|s) \underbrace{\left(R(s, a) + \sum_{s'} P(s'|s, a) V_{h+1}^\pi(s') \right)}_{Q_h^\pi(s, a)} \quad \text{for } h = H, \dots, 1$$

$$V_h^\pi(s) = \sum_{a \in \mathcal{A}} \pi_h(a|s) Q_h^\pi(s, a)$$

$$Q_h^\pi(s, a) = R(s, a) + \sum_{s'} P(s'|s, a) V_{h+1}^\pi(s')$$

Occupancy Measures

$$d_{\rho}^{\pi}(s) = \mathbb{E} \left[\sum_{h=1}^H \mathbb{I}\{s_h = s\} \mid \underline{s_1 \sim \rho, a_i \sim \pi_i(\cdot | s_i) \forall i \geq 1} \right]$$

$d_{\rho}^{\pi}(s)$: the expected number of times state s is visited, under policy π and initial state distribution ρ

$$\boxed{d_{\rho,h}^{\pi}(s) = \Pr(s_h = s \mid \text{---})} = \mathbb{E} \left[\mathbb{I}\{s_h = s\} \mid \text{---} \right], \quad d_{\rho}^{\pi}(s) = \sum_{h=1}^H d_{\rho,h}^{\pi}(s)$$

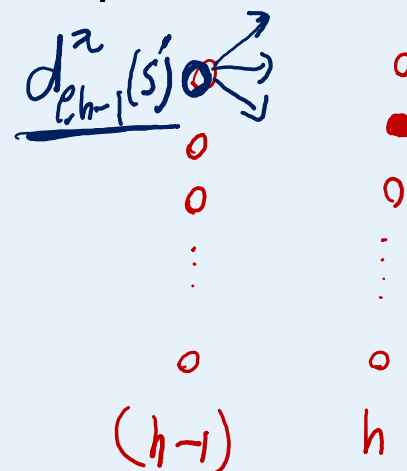
Key quantity: $d_{\rho,h}^{\pi}(s)$ = the probability of state s being visited **at step h** , under policy π and initial state distribution ρ

Forward calculation:

$$d_{\rho,1}^{\pi}(s) = \rho(s) \quad \forall s$$

For $h = 2, \dots, H$:

$$d_{\rho,h}^{\pi}(s) = \sum_{s'} d_{\rho,h-1}^{\pi}(s') \sum_{a'} \pi_{h-1}(a' | s') P(s | s', a') \quad \forall s$$



Reverse Bellman Equation

$$d_{\rho,1}^{\pi}(s) = \rho(s)$$

$$d_{\rho,h}^{\pi}(s) = \sum_{s',a'} d_{\rho,h-1}^{\pi}(s') \pi_{h-1}(a'|s') P(s|s',a') \quad \text{for } h = 2, \dots, H$$

$d_{\rho,h-1}^{\pi}(s', a') = \Pr(S_{h-1}=s', a_{h-1}=a' \mid s_1 \sim \rho, \pi)$

$$d_{\rho,h}^{\pi}(s) = \sum_{s',a'} d_{\rho,h-1}^{\pi}(s', a') P(s|s', a')$$

$$d_{\rho,h}^{\pi}(s, a) = d_{\rho,h}^{\pi}(s) \pi_h(a|s)$$

Dynamic Programming

$$V_h^*(s) = \max_{\pi} V_h^{\pi}(s)$$

Goal: Find the optimal policy

Key quantity: $V_h^*(s)$ = the optimal expected total reward starting from state s at step h .

Backward calculation:

$$V_H^*(s) = \max_a R(s, a) \quad \forall s$$

For $h = H - 1, \dots, 1$:

$$V_h^*(s) = \max_a \left(R(s, a) + \sum_{s'} P(s'|s, a) V_{h+1}^*(s') \right) \quad \forall s$$

} Value Iteration

$$\pi_h^*(s) = \operatorname{argmax}_a R(s, a) + \sum_{s'} P(s'|s, a) V_{h+1}^*(s')$$

Bellman Optimality Equation

$$V_{H+1}^*(s) = 0$$

$$V_h^*(s) = \max_a \left(\underbrace{R(s, a) + \sum_{s'} P(s'|s, a) V_{h+1}^*(s')}_{Q_h^*(s, a)} \right) \quad \text{for } h = H, \dots, 1$$

$$V_h^*(s) = \max_a Q_h^*(s, a)$$

$$Q_h^*(s, a) = R(s, a) + \sum_{s'} P(s'|s, a) V_{h+1}^*(s')$$

$$\pi_h^*(s) = \operatorname{argmax}_a Q_h^*(s, a)$$

Recap

$$V_h^\pi(s) = \sum_{a \in \mathcal{A}} \pi_h(a|s) Q_h^\pi(s, a)$$

$$Q_h^\pi(s, a) = R(s, a) + \sum_{s' \in \mathcal{S}} P(s'|s, a) V_{h+1}^\pi(s')$$

Bellman Equation
(Value Iteration for V^π)

$$d_{\rho, h}^\pi(s, a) = d_{\rho, h}^\pi(s) \pi_h(a|s)$$

$$d_{\rho, h}^\pi(s) = \sum_{s', a'} d_{\rho, h-1}^\pi(s', a') P(s|s', a')$$

Reverse Bellman Equation

$$V_h^*(s) = \max_a Q_h^*(s, a)$$

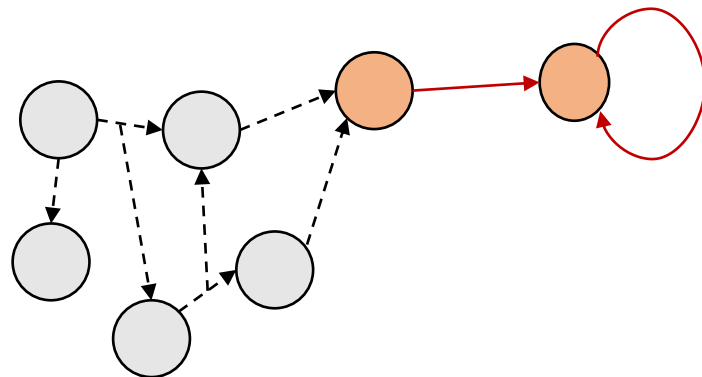
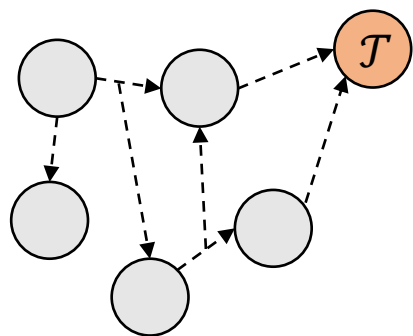
$$Q_h^*(s, a) = R(s, a) + \sum_{s' \in \mathcal{S}} P(s'|s, a) V_{h+1}^*(s')$$

Bellman Optimality Equation
(Value Iteration)

**Infinite-Horizon / Goal-Oriented +
Discounted Total Reward**

Equivalent Views

→ deterministic and zero-reward

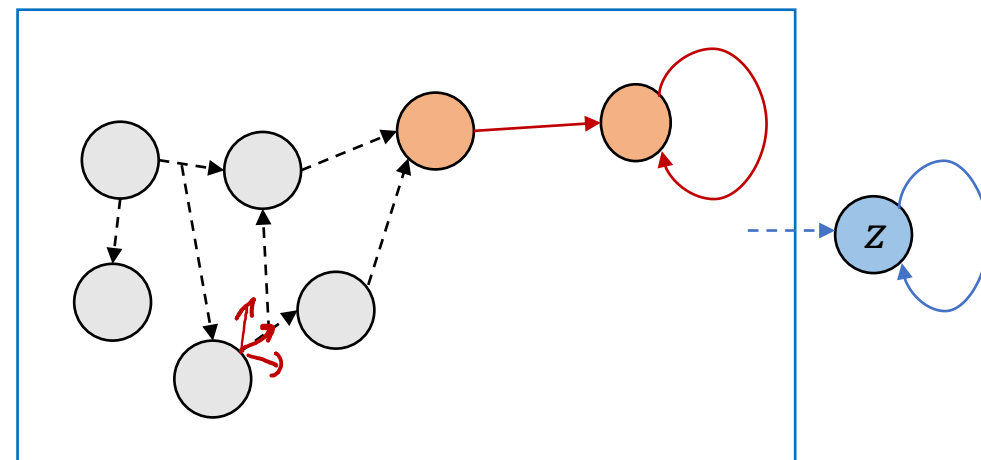


Converting goal-oriented to infinite-horizon

$$\mathbb{E}^{\text{new}} \left[\sum_{h=1}^{\infty} \gamma^{h-1} r_h \right] = \mathbb{E}^{\text{old}} \left[\sum_{h=1}^{\tau} \gamma^{h-1} r_h \right]$$

$$p^{\text{new}}(s'|s,a) = \gamma p^{\text{old}}(s'|s,a), \quad p^{\text{new}}(z|s,a) = 1-\gamma$$

Scale down all transitions by a factor of γ and add probability $1 - \gamma$ transitioning to z



Converting discounted total reward to total reward

$$\mathbb{E}^{\text{new}} \left[\sum_{h=1}^{\infty} r_h \right] = \mathbb{E}^{\text{old}} \left[\sum_{h=1}^{\infty} \gamma^{h-1} r_h \right]$$

Prob of staying in original MDP at step h

Dynamic Programming

$$V(s) = \lim_{H \rightarrow \infty} \mathbb{E} \left[\sum_{h=1}^H \gamma^{h-1} R(s_h, a_h) \mid s_1 = s, a_h \sim \pi(\cdot | s_h) \forall h \geq 1 \right]$$

Goal: Calculate the expected discounted total reward of a stationary policy π

$V^\pi(s)$ = the expected discounted total reward starting from state s , follow π

Key quantity: $V_i^\pi(s)$ = the expected discounted total reward starting from state s supposed that i more steps can be executed

$$V_i^\pi(s) = \mathbb{E} \left[\sum_{h=1}^i \gamma^{h-1} R(s_h, a_h) \mid \sim \right]$$

$V_0^\pi(s) = 0 \quad \forall s$

For $i = 1, 2, 3 \dots$

$$V_i^\pi(s) = \sum_a \pi(a|s) \left(R(s, a) + \gamma \sum_{s'} P(s'|s, a) V_{i-1}^\pi(s') \right) \quad \forall s$$

$$V^\pi(s) = \lim_{i \rightarrow \infty} V_i^\pi(s) \quad (\text{need to prove that the limit exists})$$

Value Iteration for V^π

$$\lim_{i \rightarrow \infty} \hat{V}_i(s) = V^\pi(s)$$

Arbitrary $\hat{V}_0(s) \quad \forall s$

For $i = 1, 2, 3 \dots$

$$\hat{V}_i(s) = \sum_a \pi(a|s) \left(R(s, a) + \gamma \sum_{s'} P(s'|s, a) \hat{V}_{i-1}(s') \right) \quad \forall s$$

To show that this algorithm converges, we prove the following statement:

For any $\epsilon > 0$, there exists a large enough N such that

$$|\hat{V}_i(s) - \hat{V}_j(s)| \leq \epsilon$$

for any $i, j \geq N$.

Proof of Convergence

$$|\hat{V}_{i+1}(s) - \hat{V}_i(s)| \leq O(\gamma^i) \quad \forall s.$$

$$|\hat{V}_i(s) - \hat{V}_j| \leq \sum_{k=j}^{i-1} |\hat{V}_k(s) - \hat{V}_{k+1}(s)| = \sum_{k=j}^{i-1} O(\gamma^k) \leq O\left(\frac{\gamma^j}{1-\gamma}\right)$$

$$\hat{V}_i(s) = \sum_a \pi(a|s) \left(R(s,a) + \gamma \sum_{s'} P(s'|s,a) \hat{V}_{i-1}(s') \right) \quad \forall s$$

$$\hat{V}_{i+1}(s) = \sum_a \pi(a|s) \left(R(s,a) + \gamma \sum_{s'} P(s'|s,a) \hat{V}_i(s') \right) \quad \forall s.$$

$$\Rightarrow \hat{V}_{i+1}(s) - \hat{V}_i(s) = \gamma \sum_a \pi(a|s) \sum_{s'} P(s'|s,a) \left(\hat{V}_i(s') - \hat{V}_{i-1}(s') \right)$$

$$\Rightarrow \max_s |\hat{V}_{i+1}(s) - \hat{V}_i(s)| \leq \gamma \sum_a \pi(a|s) \sum_{s'} P(s'|s,a) \underbrace{|\hat{V}_i(s') - \hat{V}_{i-1}(s')|}_{\text{max over } s'}$$

$$\approx \underbrace{\gamma \sum_a \pi(a|s)}_{\text{max over } s} \underbrace{\sum_{s'} P(s'|s,a)}_{\text{max over } s'} \max_{s^*} |\hat{V}_i(s^*) - \hat{V}_{i-1}(s^*)|$$


$$\leq \gamma \max_{s^*} |\hat{V}_i(s^*) - \hat{V}_{i-1}(s^*)|$$

Proof of Convergence

For any $\epsilon > 0$, there exists a large enough N such that

$$|\hat{V}_i(s) - \hat{V}_j(s)| \leq \epsilon$$

for any $i, j \geq N$.


$$\hat{V}(s) = \lim_{i \rightarrow \infty} \inf \{ \hat{V}_j(s) : j \geq i \}$$

For any $\epsilon > 0$, there exists a large enough N such that

$$|\hat{V}_i(s) - \hat{V}(s)| \leq \epsilon$$

for any $i \geq N$.

Proof of Uniqueness

No matter what the initial values of $\hat{V}_0(s)$ are, the limit $\lim_{i \rightarrow \infty} \hat{V}_i(s)$ is the same.

(This value is $V^\pi(s)$)

Assume $V_{(s)}^{(1)}$ and $V_{(s)}^{(2)}$ are different convergence point.

$$\begin{array}{c} V^{(1)}(s) = \sum_a \pi(a|s) \left(R(s,a) + \gamma \sum_{s'} p(s'|s,a) V^{(1)}(s') \right) \\ V^{(2)}(s) \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad V^{(2)}(s') \end{array}$$

$$V^{(1)}(s) - V^{(2)}(s) = \gamma \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) (V^{(1)}(s') - V^{(2)}(s'))$$

$$\max_s |V^{(1)}(s) - V^{(2)}(s)| \leq \gamma \max_{s'} |V^{(1)}(s') - V^{(2)}(s')| \rightarrow |V^{(1)}(s) - V^{(2)}(s)| = 0$$

Bellman Equation

$$V^\pi(s) = \sum_a \pi(a|s) \left(\underbrace{R(s, a) + \gamma \sum_{s'} P(s'|s, a) V^\pi(s')}_{Q^\pi(s, a)} \right)$$

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

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

Occupancy Measures

$d_{\rho}^{\pi}(s)$: the expected discounted number of times state s is visited, under policy π and initial state distribution ρ

Key quantity: $d_{\rho,h}^{\pi}(s)$ = the discounted probability of state s being visited at step h , under policy π and initial state distribution ρ

Forward calculation:

$$d_{\rho,1}^{\pi}(s) = \rho(s) \quad \forall s$$

For $h = 2, 3, \dots$

$$d_{\rho,h}^{\pi}(s) = \gamma \sum_{s'} d_{\rho,h-1}^{\pi}(s') \sum_{a'} \pi(a'|s') P(s|s', a') \quad \forall s$$

Reverse Bellman Equation

$$d_{\rho}^{\pi}(s) = \rho(s) + \gamma \sum_{s', a'} \underbrace{d_{\rho}^{\pi}(s') \pi(a'|s')}_{d_{\rho}^{\pi}(s', a')} P(s|s', a')$$

$$d_{\rho}^{\pi}(s) = \rho(s) + \gamma \sum_{s', a'} d_{\rho}^{\pi}(s', a') P(s|s', a')$$
$$d_{\rho}^{\pi}(s, a) = d_{\rho}^{\pi}(s) \pi(a|s)$$

Another (more common) version makes $d_{\rho}^{\pi}(s)$ a distribution over s

→ Just change the $\rho(s)$ in the first equation by $(1 - \gamma)\rho(s)$

Dynamic Programming

Goal: find optimal policy

Key quantity: $V_i^*(s)$ = the optimal discounted total reward starting from state s **supposed that i more steps can be executed**

$$V_0^*(s) = 0 \quad \forall s$$

For $i = 1, 2, 3 \dots$

$$V_i^*(s) = \max_a \left(R(s, a) + \gamma \sum_{s'} P(s'|s, a) V_{i-1}^*(s') \right) \quad \forall s$$

Value Iteration

$$V^*(s) = \lim_{i \rightarrow \infty} V_i^*(s) \quad \pi^*(s) = \operatorname{argmax}_a R(s, a) + \gamma \sum_{s'} P(s'|s, a) V^*(s')$$

Bellman Optimality Equation

$$V^*(s) = \max_a \left(\underbrace{R(s, a) + \sum_{s'} P(s'|s, a) V^*(s')}_{Q^*(s, a)} \right)$$

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

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

$$\pi^*(s) = \operatorname{argmax}_a Q^*(s, a)$$

Recap

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a|s) Q^\pi(s, a)$$

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

Bellman Equation
(Value Iteration for V^π)

$$d_\rho^\pi(s, a) = d_\rho^\pi(s) \pi(a|s)$$

$$d_\rho^\pi(s) = (1 - \gamma) \rho(s) + \gamma \sum_{s', a'} d_\rho^\pi(s', a') P(s|s', a')$$

Reverse Bellman Equation

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

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

Bellman Optimality Equation
(Value Iteration)

If Bellman (Optimality) Equations Only Hold Approximately

$$\text{If } \left| \hat{V}(s) - \max_a \left(R(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a)} [\hat{V}(s')] \right) \right| \leq \epsilon \quad \forall s$$

$$\text{then } |\hat{V}(s) - V^*(s)| \leq \frac{\epsilon}{1 - \gamma} \quad \forall s$$

Policy Iteration

Policy Iteration

Policy Iteration

For $k = 1, 2, \dots$

$$\forall s, \quad \pi^{(k+1)}(s) \leftarrow \operatorname{argmax}_a Q^{\pi^{(k)}}(s, a)$$

Theorem (monotonic improvement). Policy Iteration ensures

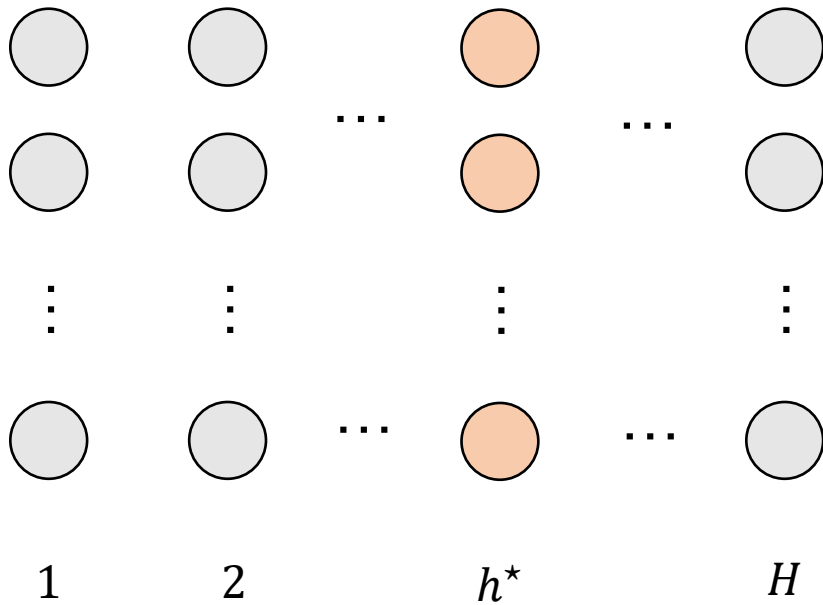
$$\forall s, \quad V^{\pi^{(k+1)}}(s) \geq V^{\pi^{(k)}}(s)$$

Below, we will establish a more general lemma (not only show monotonic improvement, but also quantify *how much* the improvement is).

Single-Step Policy Modification under Fixed Horizon

Assume $\pi'_h(\cdot | s) = \pi_h(\cdot | s)$ for all $h \neq h^*$

$$\mathbb{E}_{s \sim \rho} [V_1^{\pi'}(s)] - \mathbb{E}_{s \sim \rho} [V_1^{\pi}(s)] = ?$$

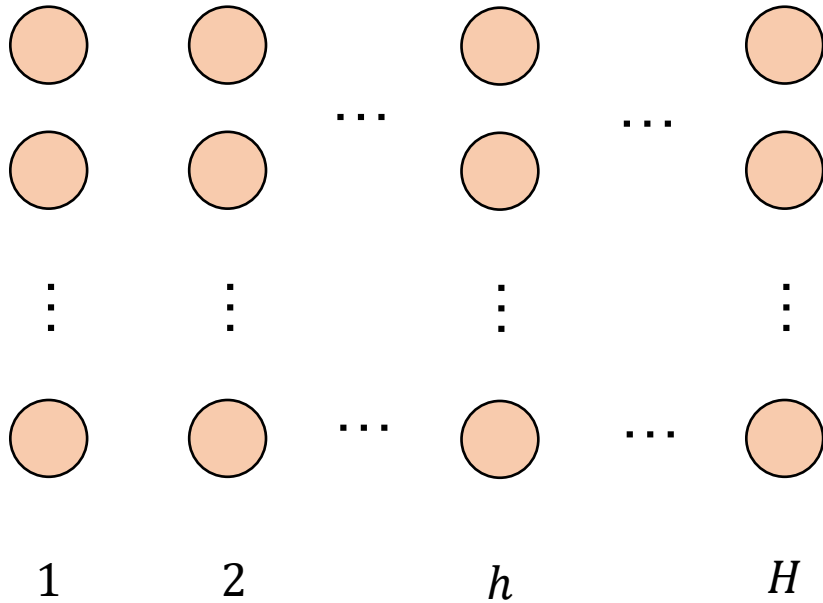


All-Step Policy Modification under Fixed Horizon

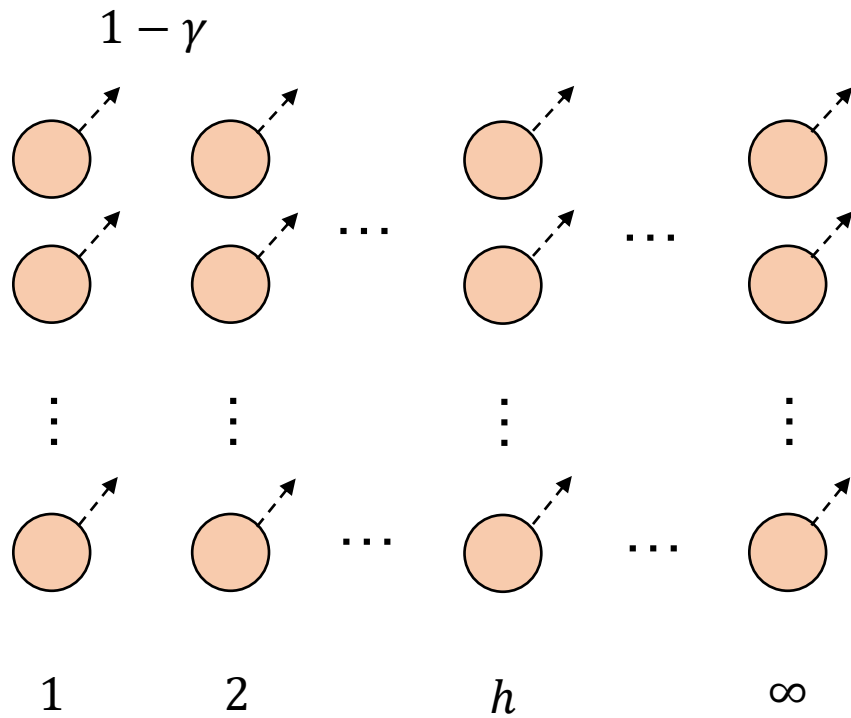
Let $\pi^{(h)}$ be a Markov policy such that it is

$\left\{ \begin{array}{l} \text{same as } \pi' \text{ in steps } 1 \text{ to } h - 1 \\ \text{same as } \pi \text{ in steps } h \text{ to } H \end{array} \right.$

$\pi' = \pi^{(H+1)}$ and $\pi = \pi^{(1)}$



Discounted Total Reward Setting



Performance / Value Difference Lemma

For any two stationary policies π' and π in the discounted total reward setting,

$$\begin{aligned}\mathbb{E}_{s \sim \rho} [V^{\pi'}(s)] - \mathbb{E}_{s \sim \rho} [V^{\pi}(s)] &= \sum_{s,a} d_{\rho}^{\pi'}(s) (\pi'(a|s) - \pi(a|s)) Q^{\pi}(s, a) \\ &= \sum_{s,a} d_{\rho}^{\pi'}(s, a) (Q^{\pi}(s, a) - V^{\pi}(s))\end{aligned}$$

Modified Policy Iteration

Bellman Operator \mathcal{T}^π

$$(\mathcal{T}^\pi V)(s) = \sum_a \pi(a|s) \left(R(s, a) + \gamma \sum_{s'} P(s'|s, a) V(s') \right)$$

Greedy Policy Operator \mathcal{G}

$$(\mathcal{G}V)(s) = \operatorname{argmax}_a \left(R(s, a) + \gamma \sum_{s'} P(s'|s, a) V(s') \right)$$

Value Iteration:

$$\pi_{k+1} = \mathcal{G}V_k$$

$$V_{k+1} = \mathcal{T}^{\pi_{k+1}} V_k$$

Policy Iteration:

$$\pi_{k+1} = \mathcal{G}V_k$$

$$V_{k+1} = (\mathcal{T}^{\pi_{k+1}})^\infty V_k$$

MPI:

$$\pi_{k+1} = \mathcal{G}V_k$$

$$V_{k+1} = (\mathcal{T}^{\pi_{k+1}})^m V_k$$

Policy update

Value update

Difference:

Relative speed
between policy and
value updates

Summary for the Basics of MDPs

- MDPs model decision-making problems where the return depends on sequences of actions.
- “State” summarizes all the information needed to make decisions (in the fixed-horizon setting, the step index is also important).
- While the number of **action sequence is exponential** in the horizon length, the optimal policy can be computed in **poly(#state, #actions, horizon length)** time using dynamic programming techniques (Value Iteration).
- Interaction Protocols: fixed-horizon, goal-oriented, infinite-horizon
- Performance Metrics: total reward, average reward, discounted total reward
- Policies: history-dependent, Markov, stationary
- Bellman equation, Reverse Bellman equation, Bellman optimality equation
- Policy Iteration and Performance Difference Lemma
- Unifying Value Iteration and Policy Iteration