

Foundations of Machine Learning

Reinforcement Learning

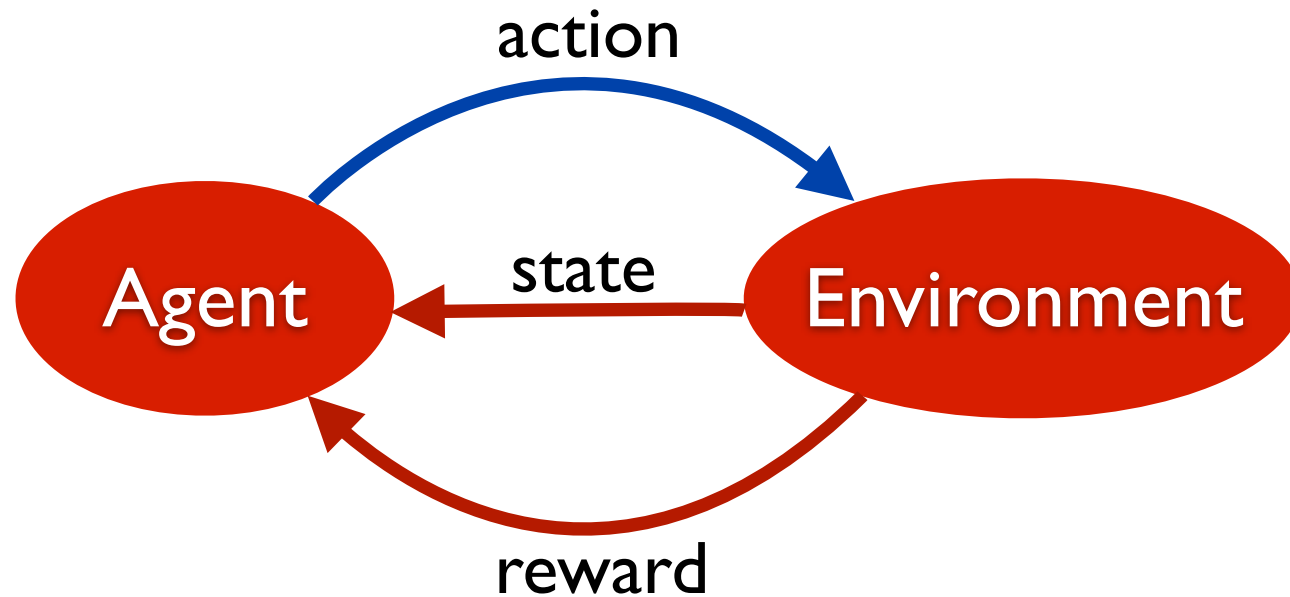
Mehryar Mohri

Courant Institute and Google Research

mohri@cims.nyu.edu

Reinforcement Learning

- Agent exploring environment.
- Interactions with environment:



- **Problem:** find action **policy** that maximizes cumulative reward over the course of interactions.

Key Features

- Contrast with supervised learning:
 - no explicit labeled training data.
 - distribution defined by actions taken.
- Delayed rewards or penalties.
- RL trade-off:
 - **exploration** (of unknown states and actions) to gain more reward information; vs.
 - **exploitation** (of known information) to optimize reward.

Applications

- Robot control e.g., Robocup Soccer Teams (Stone et al., 1999).
- Board games, e.g., TD-Gammon (Tesauro, 1995).
- Elevator scheduling (Crites and Barto, 1996).
- Ads placement.
- Telecommunications.
- Inventory management.
- Dynamic radio channel assignment.

This Lecture

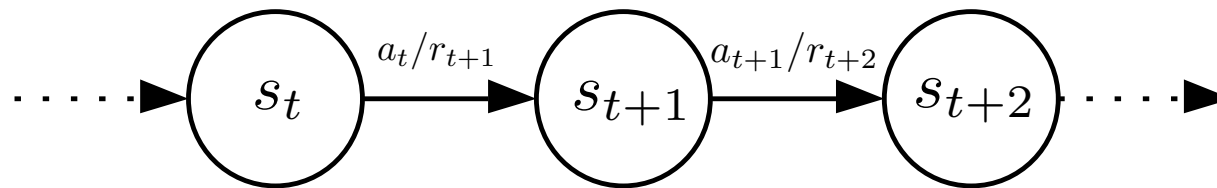
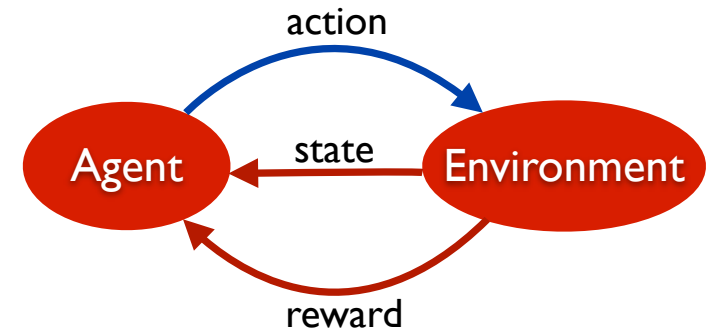
- Markov Decision Processes (MDPs)
- Planning
- Learning
- Multi-armed bandit problem

Markov Decision Process (MDP)

- **Definition:** a Markov Decision Process is defined by:
- a set of **decision epochs** $\{0, \dots, T\}$.
 - a set of **states** S , possibly infinite.
 - a start state or initial state s_0 ;
 - a set of **actions** A , possibly infinite.
 - a **transition probability** $\Pr[s' | s, a]$: distribution over destination states $s' = \delta(s, a)$.
 - a **reward probability** $\Pr[r' | s, a]$: distribution over rewards returned $r' = r(s, a)$.

Model

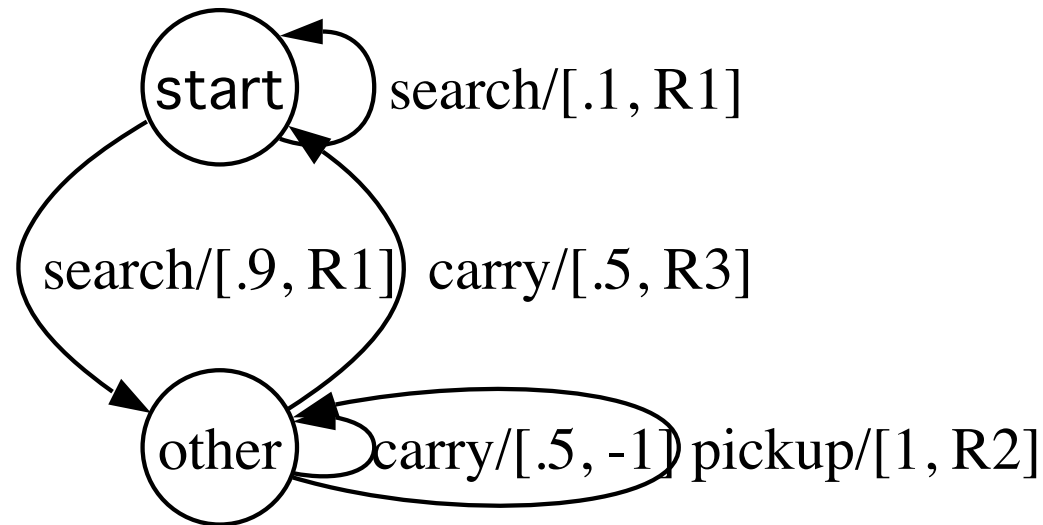
- State observed at time t : $s_t \in S$.
- Action taken at time t : $a_t \in A$.
- State reached $s_{t+1} = \delta(s_t, a_t)$.
- Reward received: $r_{t+1} = r(s_t, a_t)$.



MDPs - Properties

- Finite MDPs: A and S finite sets.
- Finite horizon when $T < \infty$.
- Reward $r(s, a)$: often deterministic function.

Example - Robot Picking up Balls



Policy

- **Definition:** a **policy** is a mapping $\pi: S \rightarrow A$.
- **Objective:** find policy π maximizing expected return.
 - finite horizon return: $\sum_{t=0}^{T-1} r(s_t, \pi(s_t))$.
 - infinite horizon return: $\sum_{t=0}^{+\infty} \gamma^t r(s_t, \pi(s_t))$.
- **Theorem:** there exists an optimal policy from any start state.

Policy Value

■ **Definition:** the **value** of a policy π at state s is

- finite horizon:

$$V_{\pi}(s) = \mathbb{E} \left[\sum_{t=0}^{T-1} r(s_t, \pi(s_t)) \middle| s_0 = s \right].$$

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

$$V_{\pi}(s) = \mathbb{E} \left[\sum_{t=0}^{+\infty} \gamma^t r(s_t, \pi(s_t)) \middle| s_0 = s \right].$$

■ **Problem:** find policy π with maximum value for all states.

Policy Evaluation

■ Analysis of policy value:

$$\begin{aligned} V_{\pi}(s) &= \mathbb{E} \left[\sum_{t=0}^{+\infty} \gamma^t r(s_t, \pi(s_t)) \middle| s_0 = s \right]. \\ &= \mathbb{E}[r(s, \pi(s))] + \gamma \mathbb{E} \left[\sum_{t=0}^{+\infty} \gamma^t r(s_{t+1}, \pi(s_{t+1})) \middle| s_0 = s \right] \\ &= \mathbb{E}[r(s, \pi(s))] + \gamma \mathbb{E}[V_{\pi}(\delta(s, \pi(s)))]. \end{aligned}$$

■ Bellman equations (system of linear equations):

$$V_{\pi}(s) = \mathbb{E}[r(s, \pi(s))] + \gamma \sum_{s'} \text{Pr}[s' | s, \pi(s)] V_{\pi}(s').$$

Bellman Equation - Existence and Uniqueness

■ Notation:

- transition probability matrix $\mathbf{P}_{s,s'} = \Pr[s'|s, \pi(s)]$.
- value column matrix $\mathbf{V} = V_{\pi}(s)$.
- expected reward column matrix: $\mathbf{R} = \mathbb{E}[r(s, \pi(s))]$.

■ **Theorem:** for a finite MDP, Bellman's equation admits a unique solution given by

$$\mathbf{V}_0 = (\mathbf{I} - \gamma \mathbf{P})^{-1} \mathbf{R}.$$

Bellman Equation - Existence and Uniqueness

■ **Proof:** Bellman's equation rewritten as

$$\mathbf{V} = \mathbf{R} + \gamma \mathbf{P} \mathbf{V}.$$

- \mathbf{P} is a stochastic matrix, thus,

$$\|\mathbf{P}\|_{\infty} = \max_s \sum_{s'} |\mathbf{P}_{ss'}| = \max_s \sum_{s'} \Pr[s'|s, \pi(s)] = 1.$$

- This implies that $\|\gamma \mathbf{P}\|_{\infty} = \gamma < 1$. The eigenvalues of \mathbf{P} are all less than one and $(\mathbf{I} - \gamma \mathbf{P})$ is invertible.

■ **Notes:** general shortest distance problem (MM, 2002).

Optimal Policy

■ **Definition:** policy π^* with maximal value for all states $s \in S$.

- value of π^* (**optimal value**):

$$\forall s \in S, V_{\pi^*}(s) = \max_{\pi} V_{\pi}(s).$$

- **optimal state-action value function:** expected return for taking action a at state s and then following optimal policy.

$$\begin{aligned} Q^*(s, a) &= \mathbb{E}[r(s, a)] + \gamma \mathbb{E}[V^*(\delta(s, a))] \\ &= \mathbb{E}[r(s, a)] + \gamma \sum_{s' \in S} \Pr[s' \mid s, a] V^*(s'). \end{aligned}$$

Optimal Values - Bellman Equations

■ **Property:** the following equalities hold:

$$\forall s \in S, V^*(s) = \max_{a \in A} Q^*(s, a).$$

■ **Proof:** by definition, for all s , $V^*(s) \leq \max_{a \in A} Q^*(s, a)$.

- If for some s we had $V^*(s) < \max_{a \in A} Q^*(s, a)$, then maximizing action would define a better policy.

■ Thus,

$$V^*(s) = \max_{a \in A} \left\{ E[r(s, a)] + \gamma \sum_{s' \in S} \Pr[s' | s, a] V^*(s') \right\}.$$

This Lecture

- Markov Decision Processes (MDPs)
- Planning
- Learning
- Multi-armed bandit problem

Known Model

- **Setting:** environment model known.
- **Problem:** find optimal policy.
- **Algorithms:**
 - value iteration.
 - policy iteration.
 - linear programming.

Value Iteration Algorithm

$$\Phi(\mathbf{V})(s) = \max_{a \in A} \left\{ \mathbb{E}[r(s, a)] + \gamma \sum_{s' \in S} \Pr[s'|s, a] V(s') \right\}.$$
$$\Phi(\mathbf{V}) = \max_{\pi} \{ \mathbf{R}_{\pi} + \gamma \mathbf{P}_{\pi} \mathbf{V} \}.$$

VALUEITERATION(\mathbf{V}_0)

- 1 $\mathbf{V} \leftarrow \mathbf{V}_0 \quad \triangleright \mathbf{V}_0$ arbitrary value
- 2 **while** $\|\mathbf{V} - \Phi(\mathbf{V})\| \geq \frac{(1-\gamma)\epsilon}{\gamma}$ **do**
- 3 $\mathbf{V} \leftarrow \Phi(\mathbf{V})$
- 4 **return** $\Phi(\mathbf{V})$

VI Algorithm - Convergence

■ **Theorem:** for any initial value V_0 , the sequence defined by $V_{n+1} = \Phi(V_n)$ converge to V^* .

■ **Proof:** we show that Φ is γ -contracting for $\|\cdot\|_\infty$

→ **existence and uniqueness of fixed point for Φ .**

- for any $s \in S$, let $a^*(s)$ be the maximizing action defining $\Phi(V)(s)$. Then, for $s \in S$ and any U ,

$$\begin{aligned}\Phi(V)(s) - \Phi(U)(s) &\leq \Phi(V)(s) - \left(E[r(s, a^*(s))] + \gamma \sum_{s' \in S} \Pr[s' | s, a^*(s)] U(s') \right) \\ &= \gamma \sum_{s' \in S} \Pr[s' | s, a^*(s)] [V(s') - U(s')] \\ &\leq \gamma \sum_{s' \in S} \Pr[s' | s, a^*(s)] \|V - U\|_\infty = \gamma \|V - U\|_\infty.\end{aligned}$$

Complexity and Optimality

■ **Complexity:** convergence in $O(\log \frac{1}{\epsilon})$. Observe that

$$\|\mathbf{V}_{n+1} - \mathbf{V}_n\|_{\infty} \leq \gamma \|\mathbf{V}_n - \mathbf{V}_{n-1}\|_{\infty} \leq \gamma^n \|\Phi(\mathbf{V}_0) - \mathbf{V}_0\|_{\infty}.$$

Thus, $\gamma^n \|\Phi(\mathbf{V}_0) - \mathbf{V}_0\|_{\infty} \leq \frac{(1 - \gamma)\epsilon}{\gamma} \Rightarrow n = O\left(\log \frac{1}{\epsilon}\right).$

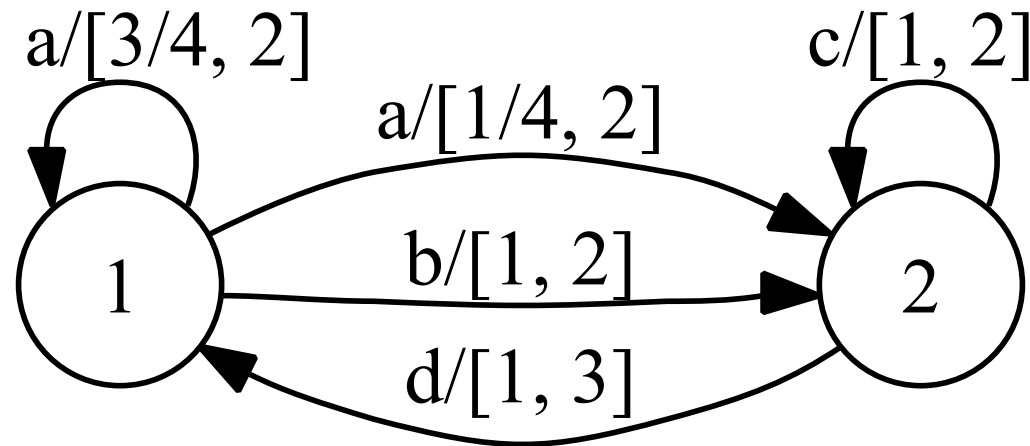
■ **ϵ -Optimality:** let \mathbf{V}_{n+1} be the value returned. Then,

$$\begin{aligned} \|\mathbf{V}^* - \mathbf{V}_{n+1}\|_{\infty} &\leq \|\mathbf{V}^* - \Phi(\mathbf{V}_{n+1})\|_{\infty} + \|\Phi(\mathbf{V}_{n+1}) - \mathbf{V}_{n+1}\|_{\infty} \\ &\leq \gamma \|\mathbf{V}^* - \mathbf{V}_{n+1}\|_{\infty} + \gamma \|\mathbf{V}_{n+1} - \mathbf{V}_n\|_{\infty}. \end{aligned}$$

Thus,

$$\|\mathbf{V}^* - \mathbf{V}_{n+1}\|_{\infty} \leq \frac{\gamma}{1 - \gamma} \|\mathbf{V}_{n+1} - \mathbf{V}_n\|_{\infty} \leq \epsilon.$$

VI Algorithm - Example



$$\mathbf{V}_{n+1}(1) = \max \left\{ 2 + \gamma \left(\frac{3}{4} \mathbf{V}_n(1) + \frac{1}{4} \mathbf{V}_n(2) \right), 2 + \gamma \mathbf{V}_n(2) \right\}$$

$$\mathbf{V}_{n+1}(2) = \max \left\{ 3 + \gamma \mathbf{V}_n(1), 2 + \gamma \mathbf{V}_n(2) \right\}.$$

For $\mathbf{V}_0(1) = -1, \mathbf{V}_0(2) = 1, \gamma = 1/2, \mathbf{V}_1(1) = \mathbf{V}_1(2) = 5/2$.

But, $\mathbf{V}^*(1) = 14/3, \mathbf{V}^*(2) = 16/3$.

Policy Iteration Algorithm

POLICYITERATION(π_0)

```
1  $\pi \leftarrow \pi_0$   $\triangleright \pi_0$  arbitrary policy
2  $\pi' \leftarrow \text{NIL}$ 
3 while ( $\pi \neq \pi'$ ) do
4      $\mathbf{V} \leftarrow \mathbf{V}_\pi$   $\triangleright$  policy evaluation: solve  $(\mathbf{I} - \gamma \mathbf{P}_\pi) \mathbf{V} = \mathbf{R}_\pi$ .
5      $\pi' \leftarrow \pi$ 
6      $\pi \leftarrow \operatorname{argmax}_\pi \{ \mathbf{R}_\pi + \gamma \mathbf{P}_\pi \mathbf{V} \}$   $\triangleright$  greedy policy improvement.
7 return  $\pi$ 
```

PI Algorithm - Convergence

- **Theorem:** let $(V_n)_{n \in \mathbb{N}}$ be the sequence of policy values computed by the algorithm, then,

$$V_n \leq V_{n+1} \leq V^*.$$

- **Proof:** let π_{n+1} be the policy improvement at the n th iteration, then, by definition,

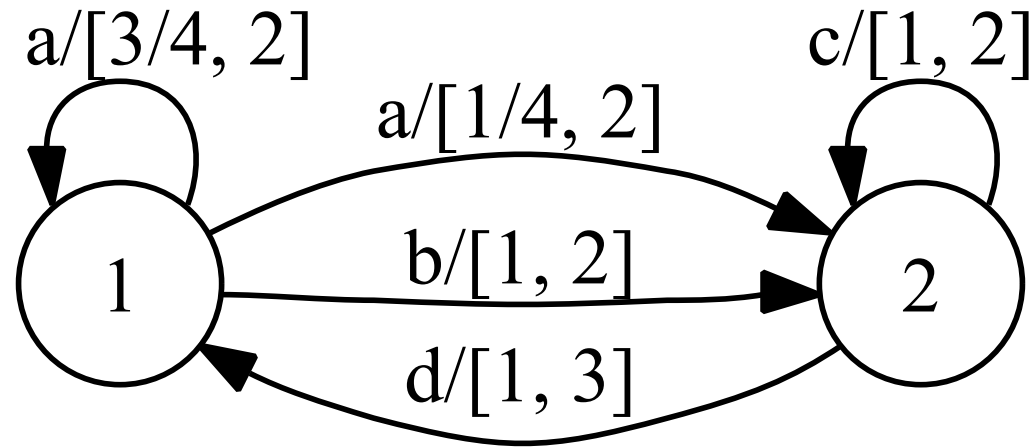
$$R_{\pi_{n+1}} + \gamma P_{\pi_{n+1}} V_n \geq R_{\pi_n} + \gamma P_{\pi_n} V_n = V_n.$$

- therefore, $R_{\pi_{n+1}} \geq (I - \gamma P_{\pi_{n+1}}) V_n.$
- note that $(I - \gamma P_{\pi_{n+1}})^{-1}$ preserves ordering:
$$X \geq 0 \Rightarrow (I - \gamma P_{\pi_{n+1}})^{-1} X = \sum_{k=0}^{\infty} (\gamma P_{\pi_{n+1}})^k X \geq 0.$$
- thus, $V_{n+1} = (I - \gamma P_{\pi_{n+1}})^{-1} R_{\pi_{n+1}} \geq V_n.$

Notes

- Two consecutive policy values can be equal only at last iteration.
- The total number of possible policies is $|A|^{|S|}$, thus, this is the maximal possible number of iterations.
 - best upper bound known $O\left(\frac{|A|^{|S|}}{|S|}\right)$.

PI Algorithm - Example



Initial policy: $\pi_0(1) = b, \pi_0(2) = c$.

Evaluation: $V_{\pi_0}(1) = 1 + \gamma V_{\pi_0}(2)$

$$V_{\pi_0}(2) = 2 + \gamma V_{\pi_0}(2).$$

Thus, $V_{\pi_0}(1) = \frac{1 + \gamma}{1 - \gamma} \quad V_{\pi_0}(2) = \frac{2}{1 - \gamma}.$

VI and PI Algorithms - Comparison

■ **Theorem:** let $(U_n)_{n \in \mathbb{N}}$ be the sequence of policy values generated by the VI algorithm, and $(V_n)_{n \in \mathbb{N}}$ the one generated by the PI algorithm. If $U_0 = V_0$, then,

$$\forall n \in \mathbb{N}, U_n \leq V_n \leq V^*.$$

■ **Proof:** we first show that Φ is monotonic. Let U and V be such that $U \leq V$ and let π be the policy such that $\Phi(U) = R_\pi + \gamma P_\pi U$. Then,

$$\Phi(U) \leq R_\pi + \gamma P_\pi V \leq \max_{\pi'} \{R'_\pi + \gamma P'_\pi V\} = \Phi(V).$$

VI and PI Algorithms - Comparison

- The proof is by induction on n . Assume $U_n \leq V_n$, then, by the monotonicity of Φ ,

$$U_{n+1} = \Phi(U_n) \leq \Phi(V_n) = \max_{\pi} \{R_{\pi} + \gamma P_{\pi} V_n\}.$$

- Let π_{n+1} be the maximizing policy:

$$\pi_{n+1} = \operatorname{argmax}_{\pi} \{R_{\pi} + \gamma P_{\pi} V_n\}.$$

- Then,

$$\Phi(V_n) = R_{\pi_{n+1}} + \gamma P_{\pi_{n+1}} V_n \leq R_{\pi_{n+1}} + \gamma P_{\pi_{n+1}} V_{n+1} = V_{n+1}.$$

Notes

- The PI algorithm converges in a smaller number of iterations than the VI algorithm due to the optimal policy.
- But, each iteration of the PI algorithm requires computing a policy value, i.e., solving a system of linear equations, which is more expensive to compute than an iteration of the VI algorithm.

Primal Linear Program

■ **LP formulation:** choose $\alpha(s) > 0$, with $\sum_s \alpha(s) = 1$.

$$\min_{\mathbf{V}} \sum_{s \in S} \alpha(s) V(s)$$

subject to $\forall s \in S, \forall a \in A, V(s) \geq \mathbb{E}[r(s, a)] + \gamma \sum_{s' \in S} \Pr[s'|s, a] V(s')$.

■ **Parameters:**

- number rows: $|S||A|$.
- number of columns: $|S|$.

Dual Linear Program

■ LP formulation:

$$\max_{\mathbf{x}} \sum_{s \in S, a \in A} \mathbb{E}[r(s, a)] x(s, a)$$

$$\text{subject to } \forall s \in S, \sum_{a \in A} x(s', a) = \alpha(s') + \gamma \sum_{s \in S, a \in A} \Pr[s' | s, a] x(s', a)$$

$$\forall s \in S, \forall a \in A, x(s, a) \geq 0.$$

■ Parameters: more favorable number of rows.

- number rows: $|S|$.
- number of columns: $|S||A|$.

This Lecture

- Markov Decision Processes (MDPs)
- Planning
- Learning
- Multi-armed bandit problem

Problem

- Unknown model:
 - transition and reward probabilities not known.
 - realistic scenario in many practical problems, e.g., robot control.
- Training information: sequence of immediate rewards based on actions taken.
- Learning approaches:
 - model-free: learn policy directly.
 - model-based: learn model, use it to learn policy.

Problem

- How do we estimate reward and transition probabilities?
 - use equations derived for policy value and Q-functions.
 - but, equations given in terms of some expectations.
 - → instance of a stochastic approximation problem.

Stochastic Approximation

■ **Problem:** find solution of $\mathbf{x} = H(\mathbf{x})$ with $\mathbf{x} \in \mathbb{R}^N$ while

- $H(\mathbf{x})$ cannot be computed, e.g., H not accessible;
- i.i.d. sample of noisy observations $H(\mathbf{x}_i) + \mathbf{w}_i$, available, $i \in [1, m]$, with $E[\mathbf{w}] = 0$.

■ **Idea:** algorithm based on iterative technique:

$$\begin{aligned}\mathbf{x}_{t+1} &= (1 - \alpha_t)\mathbf{x}_t + \alpha_t[H(\mathbf{x}_t) + \mathbf{w}_t] \\ &= \mathbf{x}_t + \alpha_t[H(\mathbf{x}_t) + \mathbf{w}_t - \mathbf{x}_t].\end{aligned}$$

- more generally $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha_t D(\mathbf{x}_t, \mathbf{w}_t)$.

Mean Estimation

■ **Theorem:** Let X be a random variable taking values in $[0, 1]$ and let x_0, \dots, x_m be i.i.d. values of X . Define the sequence $(\mu_m)_{m \in \mathbb{N}}$ by

$$\mu_{m+1} = (1 - \alpha_m)\mu_m + \alpha_m x_m \quad \text{with } \mu_0 = x_0.$$

Then, for $\alpha_m \in [0, 1]$, with $\sum_{m \geq 0} \alpha_m = +\infty$ and $\sum_{m \geq 0} \alpha_m^2 < +\infty$,

$$\mu_m \xrightarrow{\text{a.s.}} \mathbb{E}[X].$$

Proof

■ **Proof:** By the independence assumption, for $m \geq 0$,

$$\begin{aligned}\text{Var}[\mu_{m+1}] &= (1 - \alpha_m)^2 \text{Var}[\mu_m] + \alpha_m^2 \text{Var}[x_m] \\ &\leq (1 - \alpha_m) \text{Var}[\mu_m] + \alpha_m^2.\end{aligned}$$

- We have $\alpha_m \rightarrow 0$ since $\sum_{m \geq 0} \alpha_m^2 < +\infty$.
- Let $\epsilon > 0$ and suppose there exists $N \in \mathbb{N}$ such that for all $m \geq N$, $\text{Var}[\mu_m] \geq \epsilon$. Then, for $m \geq N$,

$$\text{Var}[\mu_{m+1}] \leq \text{Var}[\mu_m] - \alpha_m \epsilon + \alpha_m^2,$$

which implies $\text{Var}[\mu_{m+N}] \leq \underbrace{\text{Var}[\mu_N] - \epsilon \sum_{n=N}^{m+N} \alpha_n + \sum_{n=N}^{m+N} \alpha_n^2}_{\rightarrow -\infty \text{ when } m \rightarrow \infty},$

contradicting $\text{Var}[\mu_{m+N}] \geq 0$.

Mean Estimation

- Thus, for all $N \in \mathbb{N}$ there exists $m_0 \geq N$ such that $\text{Var}[\mu_{m_0}] < \epsilon$. Choose N large enough so that $\forall m \geq N, \alpha_m \leq \epsilon$. Then,
$$\text{Var}[\mu_{m_0+1}] \leq (1 - \alpha_{m_0})\epsilon + \epsilon\alpha_{m_0} = \epsilon.$$
- Therefore, $\mu_m \leq \epsilon$ for all $m \geq m_0$ (L_2 convergence).

Notes

- special case: $\alpha_m = \frac{1}{m}$.
 - Strong law of large numbers.
- Connection with stochastic approximation.

TD(0) Algorithm

- **Idea:** recall Bellman's linear equations giving V

$$\begin{aligned} V_{\pi}(s) &= \mathbb{E}[r(s, \pi(s)) + \gamma \sum_{s'} \Pr[s'|s, \pi(s)] V_{\pi}(s')] \\ &= \mathbb{E}_{s'} [r(s, \pi(s)) + \gamma V_{\pi}(s') | s]. \end{aligned}$$

- **Algorithm:** temporal difference (TD).

- sample new state s' .
- update: α depends on number of visits of s .

$$\begin{aligned} V(s) &\leftarrow (1 - \alpha)V(s) + \alpha[r(s, \pi(s)) + \gamma V(s')] \\ &= V(s) + \underbrace{\alpha[r(s, \pi(s)) + \gamma V(s') - V(s)]}_{\text{temporal difference of } V \text{ values}}. \end{aligned}$$

TD(0) Algorithm

TD(0)()

```
1   $\mathbf{V} \leftarrow \mathbf{V}_0$   $\triangleright$  initialization.
2  for  $t \leftarrow 0$  to  $T$  do
3       $s \leftarrow \text{SELECTSTATE}()$ 
4      for each step of epoch  $t$  do
5           $r' \leftarrow \text{REWARD}(s, \pi(s))$ 
6           $s' \leftarrow \text{NEXTSTATE}(\pi, s)$ 
7           $V(s) \leftarrow (1 - \alpha)V(s) + \alpha[r' + \gamma V(s')]$ 
8           $s \leftarrow s'$ 
9  return  $\mathbf{V}$ 
```

Q-Learning Algorithm

- **Idea:** assume deterministic rewards.

$$\begin{aligned} Q^*(s, a) &= \mathbb{E}[r(s, a)] + \gamma \sum_{s' \in S} \Pr[s' \mid s, a] V^*(s') \\ &= \mathbb{E}_{s'}[r(s, a) + \gamma \max_{a \in A} Q^*(s', a)] \end{aligned}$$

- **Algorithm:** $\alpha \in [0, 1]$ depends on number of visits.

- sample new state s' .
- update:

$$Q(s, a) \leftarrow \alpha Q(s, a) + (1 - \alpha)[r(s, a) + \gamma \max_{a' \in A} Q(s', a')].$$

Q-Learning Algorithm

(Watkins, 1989; Watkins and Dayan 1992)

Q-LEARNING(π)

```
1   $Q \leftarrow Q_0$   $\triangleright$  initialization, e.g.,  $Q_0 = 0$ .
2  for  $t \leftarrow 0$  to  $T$  do
3       $s \leftarrow \text{SELECTSTATE}()$ 
4      for each step of epoch  $t$  do
5           $a \leftarrow \text{SELECTACTION}(\pi, s)$   $\triangleright$  policy  $\pi$  derived from  $Q$ , e.g.,  $\epsilon$ -greedy.
6           $r' \leftarrow \text{REWARD}(s, a)$ 
7           $s' \leftarrow \text{NEXTSTATE}(s, a)$ 
8           $Q(s, a) \leftarrow Q(s, a) + \alpha[r' + \gamma \max_{a'} Q(s', a') - Q(s, a)]$ 
9           $s \leftarrow s'$ 
10 return  $Q$ 
```

Notes

- Can be viewed as a stochastic formulation of the value iteration algorithm.
- Convergence for any policy so long as states and actions visited infinitely often.
- How to choose the action at each iteration? Maximize reward? Explore other actions? Q-learning is an off-policy method: no control over the policy.

Policies

■ Epsilon-greedy strategy:

- with probability $1 - \epsilon$ greedy action from s ;
- with probability ϵ random action.

■ Epoch-dependent strategy (**Boltzmann exploration**):

$$p_t(a|s, Q) = \frac{e^{\frac{Q(s,a)}{\tau_t}}}{\sum_{a' \in A} e^{\frac{Q(s,a')}{\tau_t}}},$$

- $\tau_t \rightarrow 0$: greedy selection.
- larger τ_t : random action.

Convergence of Q-Learning

- **Theorem:** consider a finite MDP. Assume that for all $s \in S$ and $a \in A$, $\sum_{t=0}^{\infty} \alpha_t(s, a) = \infty$, $\sum_{t=0}^{\infty} \alpha_t^2(s, a) < \infty$ with $\alpha_t(s, a) \in [0, 1]$. Then, the Q-learning algorithm converges to the optimal value Q^* (with probability one).
- note: the conditions on $\alpha_t(s, a)$ impose that each state-action pair is visited infinitely many times.

SARSA: On-Policy Algorithm

SARSA(π)

```
1   $Q \leftarrow Q_0$   $\triangleright$  initialization, e.g.,  $Q_0 = 0$ .
2  for  $t \leftarrow 0$  to  $T$  do
3       $s \leftarrow \text{SELECTSTATE}()$ 
4       $a \leftarrow \text{SELECTACTION}(\pi(Q), s) \triangleright$  policy  $\pi$  derived from  $Q$ , e.g.,  $\epsilon$ -greedy.
5      for each step of epoch  $t$  do
6           $r' \leftarrow \text{REWARD}(s, a)$ 
7           $s' \leftarrow \text{NEXTSTATE}(s, a)$ 
8           $a' \leftarrow \text{SELECTACTION}(\pi(Q), s') \triangleright$  policy  $\pi$  derived from  $Q$ , e.g.,  $\epsilon$ -greedy.
9           $Q(s, a) \leftarrow Q(s, a) + \alpha_t(s, a)[r' + \gamma Q(s', a') - Q(s, a)]$ 
10          $s \leftarrow s'$ 
11          $a \leftarrow a'$ 
12 return  $Q$ 
```

Notes

- Differences with Q-learning:
 - two states: current and next states.
 - maximum reward for next state not used for next state, instead new action.
- SARSA: name derived from sequence of updates.

TD(λ) Algorithm

■ Idea:

- TD(0) or Q-learning only use immediate reward.
- use multiple steps ahead instead, for n steps:

$$R_t^n = r_{t+1} + \gamma r_{t+2} + \dots + \gamma^{n-1} r_{t+n} + \gamma^n V(s_{t+n})$$
$$V(s) \leftarrow V(s) + \alpha (R_t^n - V(s)).$$

- TD(λ) uses $R_t^\lambda = (1 - \lambda) \sum_{n=0}^{\infty} \lambda^n R_t^n$.

■ Algorithm:

$$V(s) \leftarrow V(s) + \alpha (R_t^\lambda - V(s)).$$

TD(λ) Algorithm

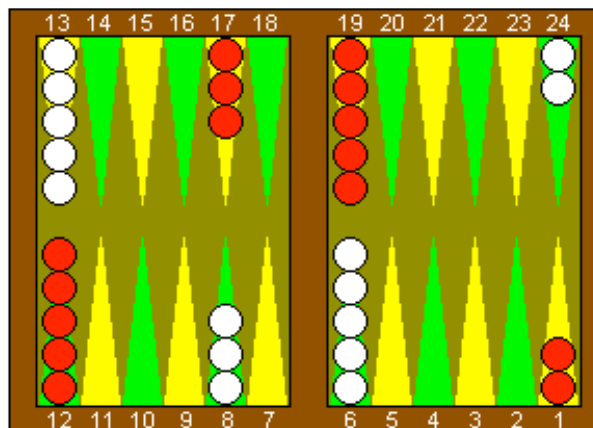
TD(λ)()

```
1   $\mathbf{V} \leftarrow \mathbf{V}_0$   $\triangleright$  initialization.
2   $\mathbf{e} \leftarrow \mathbf{0}$ 
3  for  $t \leftarrow 0$  to  $T$  do
4       $s \leftarrow \text{SELECTSTATE}()$ 
5      for each step of epoch  $t$  do
6           $s' \leftarrow \text{NEXTSTATE}(\pi, s)$ 
7           $\delta \leftarrow r(s, \pi(s)) + \lambda V(s') - V(s)$ 
8           $e(s) \leftarrow \lambda e(s) + 1$ 
9          for  $u \in S$  do
10             if  $u \neq s$  then
11                  $e(u) \leftarrow \gamma \lambda e(u)$ 
12                  $V(u) \leftarrow V(u) + \alpha \delta e(u)$ 
13              $s \leftarrow s'$ 
14  return  $\mathbf{V}$ 
```

TD-Gammon

(Tesauro, 1995)

- Large state space or costly actions: use regression algorithm to estimate Q for unseen values.
- Backgammon:
 - large number of positions: 30 pieces, 24-26 locations,
 - large number of moves.
- TD-Gammon: used neural networks.
 - non-linear form of $TD(\lambda)$, 1.5M games played,
 - almost as good as world-class humans (master level).



This Lecture

- Markov Decision Processes (MDPs)
- Planning
- Learning
- Multi-armed bandit problem

Multi-Armed Bandit Problem

(Robbins, 1952)

- **Problem:** gambler must decide which arm of a N -slot machine to pull to maximize his total reward in a series of trials.
- stochastic setting: N lever reward distributions.
- adversarial setting: reward selected by adversary aware of all the past.



Applications

- Clinical trials.
- Adaptive routing.
- Ads placement on pages.
- Games.

Multi-Armed Bandit Game

■ For $t = 1$ to T do

- adversary determines outcome $y_t \in Y$.
- player selects probability distribution p_t and pulls lever $I_t \in \{1, \dots, N\}$, $I_t \sim p_t$.
- player incurs loss $L(I_t, y_t)$ (adversary is informed of p_t and I_t).

■ **Objective:** minimize regret

$$\text{Regret}(T) = \sum_{t=1}^T L(I_t, y_t) - \min_{i=1, \dots, N} \sum_{t=1}^T L(i, y_t).$$

Notes

- Player is informed only of the loss (or reward) corresponding to his own action.
- Adversary knows past but not action selected.
- Stochastic setting: loss $(L(1, y_t), \dots, L(N, y_t))$ drawn according to some distribution $D = D_1 \otimes \dots \otimes D_N$. Regret definition modified by taking expectations.
- Exploration/Exploitation trade-off: playing the best arm found so far versus seeking to find an arm with a better payoff.

Notes

- Equivalent views:
 - special case of learning with partial information.
 - one-state MDP learning problem.
- Simple strategy: ϵ -greedy: play arm with best empirical reward with probability $1 - \epsilon_t$, random arm with probability ϵ_t .

Exponentially Weighted Average

■ **Algorithm:** Exp3, defined for $\eta, \gamma > 0$ by

$$p_{i,t} = (1 - \gamma) \frac{\exp \left(- \eta \sum_{s=1}^{t-1} \hat{l}_{i,s} \right)}{\sum_{i=1}^N \exp \left(- \eta \sum_{s=1}^{t-1} \hat{l}_{i,s} \right)} + \frac{\gamma}{N},$$

with $\forall i \in [1, N], \hat{l}_{i,t} = \frac{L(I_t, y_t)}{p_{I_t, t}} 1_{I_t=i}$.

■ **Guarantee:** expected regret of

$$O(\sqrt{NT \log N}).$$

Exponentially Weighted Average

- **Proof:** similar to the one for the Exponentially Weighted Average with the additional observation that:

$$\mathbb{E}[\hat{l}_{i,t}] = \sum_{i=1}^N p_{i,t} \frac{L(I_t, y_t)}{p_{I_t,t}} 1_{I_t=i} = L(i, y_t).$$

References

- Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. 2 vols. Belmont, MA: Athena Scientific, 2007.
- Mehryar Mohri. Semiring Frameworks and Algorithms for Shortest-Distance Problems. *Journal of Automata, Languages and Combinatorics*, 7(3):321-350, 2002.
- Martin L. Puterman *Markov decision processes: discrete stochastic dynamic programming*. Wiley-Interscience, New York, 1994.
- Robbins, H. (1952), "Some aspects of the sequential design of experiments", *Bulletin of the American Mathematical Society* 58 (5): 527–535.
- Sutton, Richard S., and Barto, Andrew G. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

References

- Gerald Tesauro. *Temporal Difference Learning and TD-Gammon*. Communications of the ACM 38 (3), 1995.
- Watkins, Christopher J. C. H. *Learning from Delayed Rewards*. Ph.D. thesis, Cambridge University, 1989.
- Christopher J. C. H. Watkins and Peter Dayan. *Q-learning*. Machine Learning, Vol. 8, No. 3-4, 1992.

Appendix

Stochastic Approximation

- **Problem:** find solution of $\mathbf{x} = H(\mathbf{x})$ with $\mathbf{x} \in \mathbb{R}^N$ while
 - $H(\mathbf{x})$ cannot be computed, e.g., H not accessible;
 - i.i.d. sample of noisy observations $H(\mathbf{x}_i) + \mathbf{w}_i$, available, $i \in [1, m]$, with $E[\mathbf{w}] = 0$.

- **Idea:** algorithm based on iterative technique:

$$\begin{aligned}\mathbf{x}_{t+1} &= (1 - \alpha_t)\mathbf{x}_t + \alpha_t[H(\mathbf{x}_t) + \mathbf{w}_t] \\ &= \mathbf{x}_t + \alpha_t[H(\mathbf{x}_t) + \mathbf{w}_t - \mathbf{x}_t].\end{aligned}$$

- more generally $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha_t D(\mathbf{x}_t, \mathbf{w}_t)$.

Supermartingale Convergence

- **Theorem:** let X_t, Y_t, Z_t be non-negative random variables such that $\sum_{t=0}^{\infty} Y_t < \infty$. If the following condition holds: $E[X_{t+1} | \mathcal{F}_t] \leq X_t + Y_t - Z_t$, then,
- X_t converges to a limit (with probability one).
 - $\sum_{t=0}^{\infty} Z_t < \infty$.

Convergence Analysis

- Convergence of $\mathbf{x}_{t+1} = \mathbf{x}_t + \alpha_t D(\mathbf{x}_t, \mathbf{w}_t)$, with history \mathcal{F}_t defined by

$$\mathcal{F}_t = \{(\mathbf{x}_{t'})_{t' \leq t}, (\alpha_{t'})_{t' \leq t}, (\mathbf{w}_{t'})_{t' < t}\}.$$

- **Theorem:** let $\Psi: \mathbf{x} \rightarrow \frac{1}{2} \|\mathbf{x} - \mathbf{x}^*\|_2^2$ for some \mathbf{x}^* and assume that

- $\exists K_1, K_2: \mathbb{E} \left[\|D(\mathbf{x}_t, \mathbf{w}_t)\|_2^2 \mid \mathcal{F}_t \right] \leq K_1 + K_2 \Psi(\mathbf{x}_t);$
- $\exists c: \nabla \Psi(\mathbf{x}_t)^\top \mathbb{E} \left[D(\mathbf{x}_t, \mathbf{w}_t) \mid \mathcal{F}_t \right] \leq -c \Psi(\mathbf{x}_t);$
 $\alpha_t > 0, \sum_{t=0}^{\infty} \alpha_t = \infty, \sum_{t=0}^{\infty} \alpha_t^2 < \infty.$

Then, $\mathbf{x}_t \xrightarrow{\text{a.s.}} \mathbf{x}^*.$

Convergence Analysis

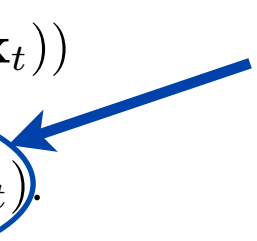
■ **Proof:** since Ψ is a quadratic function,

$$\Psi(\mathbf{x}_{t+1}) = \Psi(\mathbf{x}_t) + \nabla \Psi(\mathbf{x}_t)^\top (\mathbf{x}_{t+1} - \mathbf{x}_t) + \frac{1}{2} (\mathbf{x}_{t+1} - \mathbf{x}_t)^\top \nabla^2 \Psi(\mathbf{x}_t) (\mathbf{x}_{t+1} - \mathbf{x}_t).$$

■ **Thus,**

$$\begin{aligned} \mathbb{E} [\Psi(\mathbf{x}_{t+1}) | \mathcal{F}_t] &= \Psi(\mathbf{x}_t) + \alpha_t \nabla \Psi(\mathbf{x}_t)^\top \mathbb{E} [D(\mathbf{x}_t, \mathbf{w}_t) | \mathcal{F}_t] + \frac{\alpha_t^2}{2} \mathbb{E} [\|D(\mathbf{x}_t, \mathbf{w}_t)\|^2 | \mathcal{F}_t] \\ &\leq \Psi(\mathbf{x}_t) - \alpha_t c \Psi(\mathbf{x}_t) + \frac{\alpha_t^2}{2} (K_1 + K_2 \Psi(\mathbf{x}_t)) \\ &= \Psi(\mathbf{x}_t) + \frac{\alpha_t^2 K_1}{2} - \left(\alpha_t c - \frac{\alpha_t^2 K_2}{2} \right) \Psi(\mathbf{x}_t). \end{aligned}$$

non-neg. for large t



■ By the supermartingale convergence theorem, $\Psi(\mathbf{x}_t)$ converges and $\sum_{t=0}^{\infty} \left(\alpha_t c - \frac{\alpha_t^2 K_2}{2} \right) \Psi(\mathbf{x}_t) < \infty$.

■ Since $\alpha_t > 0$, $\sum_{t=0}^{\infty} \alpha_t = \infty$, $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$, $\Psi(\mathbf{x}_t)$ must converge to 0.