Reinforcement Learning

RL vs. Supervised learning

① Sequential decision making

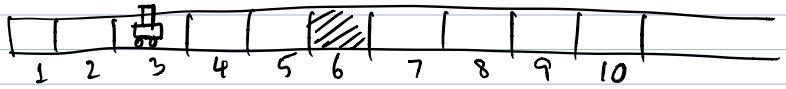
→ return vs. risk tradeoff
- long term future consequences
- greedy is not optimal

② no or little human supervision available

③ learn from designed reward fn

④ can collect more data iteratively

Markov Decision Process



S: set of states (all possible configurations)

A: set of actions ($A = \{L, R\}$)

P_{sa} dynamics / transitions

State transition prob. distrib^n

At state s, when applying action a

the dist.^{n} of next s'

$P_{sa}(s') = Pr[s' \mid s, a]$

P_{sa} is a distrib^n, $P_{sa} \in \mathbb{R}^{|S|}$

$$\sum_{s' \in S} P_{sa}(s') = 1$$

Example: L action succeeds w. prob 0.9

$P_{7,L}(6) = 0.9$

$P_{7,L}(7) = 0.1$

$P_{7,L}(s') = 0 \quad \forall\ s' \neq 6, 7$

R action succeeds w. prob. 0.8

$P_{7,R}(8) = 0.8$

$P_{7,R}(7) = 0.2$

$P_{7,R}(s') = 0 \quad \forall\ s' \neq 7, 8$

Once you reach 6, robot stays there

$P_{6,L}(6) = 1$

$P_{6,L}(s') = 0 \quad \forall\ s' \neq 6$

$P_{6,R}(6) = 1$

$P_{6,R}(s') = 0 \quad \forall\ s' \neq 6$

Sequential decision making process

- s_0 initial state (given)
- algo chooses action $a_0 \in \mathbf{A}$
- environment step: $s_1 \sim P_{s_0 a_0}$
- algo sees s_1, takes action a_1
- environment: $s_2 \sim P_{s_1, a_1}$

$s_0 = 4$
$a_0 = R$
$s_1 \sim P_{4,R}$
$s_1 = 5$
$a_1 = R$

Reward function

$R : \mathbf{S} \to \mathbb{R}$

$R(s)$

$R(6) = 1.0$
$R(s) = -0.1 \quad \forall\ s \neq 6$

$s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \dots$ trajectory / episode

Total payoff:

$R(s_0) + R(s_1) + \dots + R(s_t)$

Discounted total Payoff

Discount factor: $\gamma < 1$

$R(s_0) + \gamma R(s_1) + \gamma^2 R(s_2) \dots + \gamma^t R(s_t) + \dots$

Suppose $R(s) \in [-M, M]$

$\Rightarrow$ discounted total payoff $\leq M + \gamma M + \gamma^2 M + \dots$

$\frac{-M}{1-\gamma} \leq \qquad = \frac{M}{1-\gamma}$

finite horizon: stop process at time T

infinite horizon

Expected payoff

$\mathbb{E}[R(s_0) + \gamma R(s_1) + \dots \gamma^t R(s_t) + \dots]$

Reward can be $R(s, a)$

$MDP(S, A, \{P_{sa}\}_{\substack{s \in S \\ a \in A}}, \gamma, R)$

goal: maximize expected discounted payoff

Markov property:

at any time t, given s_t, all future states s_{t+j} don't depend on past states

$s_0, a_0, s_1, a_1 \dots s_{t-1}, a_{t-1}$

$\Rightarrow$ optimal action at time t does not depend on past states and actions

$\Rightarrow$ Best strategy can be expressed as a policy

$\pi : S \longrightarrow A$ policy

$a_t = \pi(s_t)$

optimal policy

$\pi(s) = R$ if $s \leq 6$

$\pi(s) = L$ if $s \gg 7$

randomized policy $\pi(a \mid s)$

$\exists$ deterministic optimum policy

Value function:

$V^\pi : S \longrightarrow \mathbb{R}$

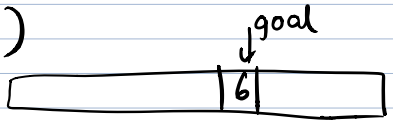
$V^\pi(s) \triangleq$ Expected payoff of executing policy π starting from state s

$s_0, a_0 \ldots$

$a_t = \pi(s_t)$

$V^\pi(s) = \mathbb{E}\left[R(s_0) + \gamma R(s_1) + \ldots + \gamma^t R(s_t) + \ldots \mid s_0 = s\right]$

How do we compute $V^\pi(s)$



$V^\pi(6) = 1 + \gamma + \gamma^2 + \gamma^3 + \ldots$

$= \frac{1}{1-\gamma}$

$V^\pi(6) = 1 + \gamma\left(1 + \gamma + \gamma^2 + \ldots\right)$

$= 1 + \gamma V^\pi(6)$

$V^\pi(6) = \frac{1}{1-\gamma}$

$V^\pi(5) = \mathbb{E}\left[R(s_0) + \gamma R(s_1) + \ldots \mid s_0 = 5\right]$

$$= R(s) + \gamma \underbrace{\mathbb{E}\left[R(s_1) + \gamma R(s_2) + \dots \mid s_0 = s\right]}_{\text{total payoff starting from } s_1 \text{ and executing } \pi}$$

$$= R(s) + \gamma \mathbb{E}_{s_1 \sim P_{s,\pi(s)}}\left[R(s_1) + \gamma R(s_2) + \dots\right]$$

$$= R(s) + \gamma \mathbb{E}_{s_1 \sim P_{s,\pi(s)}}\left[V^{\pi}(s_1)\right]$$

$$V^{\pi}(s) = R(s) + \gamma \sum_{s' \in S} P_{s,\pi(s)}(s') \cdot V^{\pi}(s')$$

Bellman Equation

$V^{\pi}(s)$ as variables

$|S|$ variables, $|S|$ equations

Solving system of linear eq^n $\Rightarrow$ $V^{\pi}(s)$ $\forall s$

given π, $P_{s,a}$ $\xrightarrow[\text{linear system}]{\text{solve}}$ $V^{\pi}(s)$, $\forall s$

Next: $V^*(s) = \max_{\pi} V^{\pi}(s)$

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

Markov property $\Rightarrow \exists\, \pi^*$ that does not depend on s

$\pi^*: S \to A$

Bellman equation for V^*

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

$$= \max_{\pi} \left(R(s) + \gamma \sum_{s'} P_{s,\pi(s)}(s') \cdot V^{\pi}(s')\right)$$

$$= R(s) + \max_{\pi} \left(\gamma \sum_{s'} P_{s,\pi(s)}(s') \cdot V^{\pi}(s')\right)$$

$$= R(s) + \max_{a} \left(\gamma \sum_{s'} P_{sa}(s') \max_{\pi} V^{\pi}(s')\right)$$

$$V^*(s) = R(s) + \gamma \max_a \sum_{s'} P_{sa}(s')\, V^*(s')$$

the best expected payoff after taking action a

$V^* \in \mathbb{R}^{|S|}$ $\begin{bmatrix} V^*(1) \\ \vdots \\ V^*(|S|) \end{bmatrix}$

B operator $\mathbb{R}^{|S|} \longrightarrow \mathbb{R}^{|S|}$

$V^* = B(V^*)$

Value iteration

- Initialize $V \in \mathbb{R}^{|S|}$, $V = 0$ $V(s) = 0 \ \forall s$

Loop

$V = B(V)$

$$V(s) = R(s) + \gamma \max_a \sum_{s'} P_{sa}(s')\, V(s')$$

$$\pi^*(s) = \arg\max_a \sum_{s' \in S} P_{sa}(s') \cdot V^*(s')$$

Policy iteration

- initialize policy π
- loop {
  - let $V = V^\pi$ (obtained by solving system of eqn)

$$\pi(s) = \arg\max_a \sum_{s'} P_{sa}(s')\, V(s')$$

}

V^*, π^* is stationary point of algorithm

$V^* = V^{\pi^*}$

Common: V^*, π^* are stationary points of algos.

Policy iteration:

Con: each iteration requires solving linear system
expensive for large state space

Pro: converges in finite time exactly
Atmost $|A|^{|S|}$ policies

Value iteration always has some approximation

Practice: discrete state space
- value iteration often slightly better
- continuous controls policy iteration
(or some combin°)

because max operation in Value iteration is hard to do exactly