Lecture 11

Reinforcement learning:

A car (robot car)

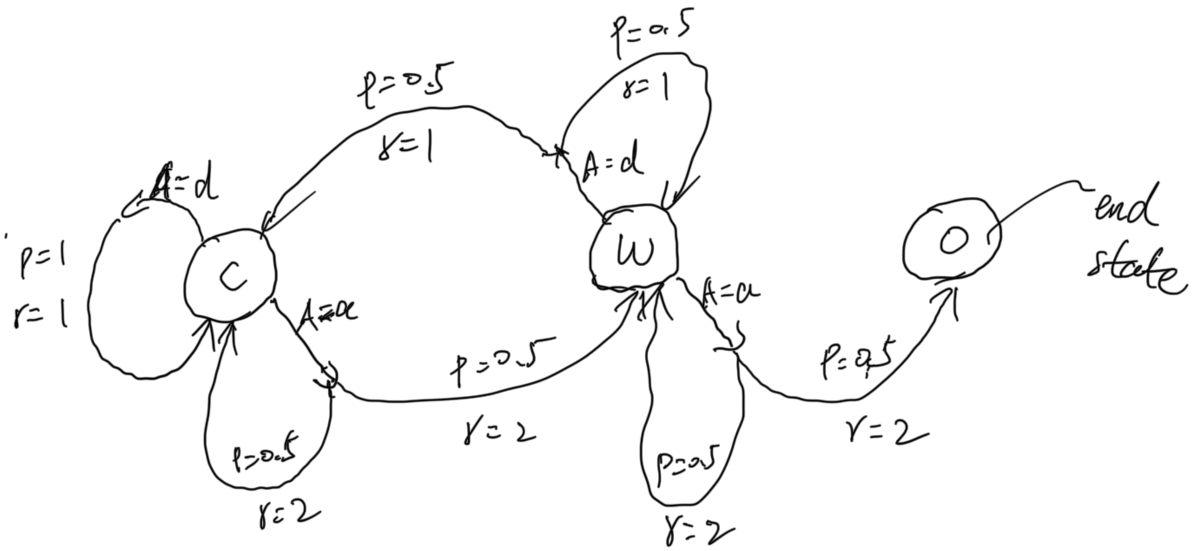
States = {cool, warm, overheated} = {c, w, o}.

Actions = {Acc, De-acc} = {a, d}.

standard time step

reward(A): $r(a) = 2$, $r(d) = 1$, $r(o) = -10$

$r(A=a) = 2$, $r(A=d) = 1$, $r(s=o) = -10$



Markov Decision Process:

- set of state S
- Start state S_0
- Set of actions A
- Transition Prob $P(s' | s, a)$ (or $T(s, a, s')$)
- reward $R(s, a, s')$ { receive after an action

└ receive after reach a state

reward discount γ.

receive a sequence of reward over 3 steps:

case1 : $r_1 = [1, 2, 3]$

case2 : $r_2 = [3, 2, 1]$

without discount : tot. reward $R_1 = 6$

$R_2 = 6$

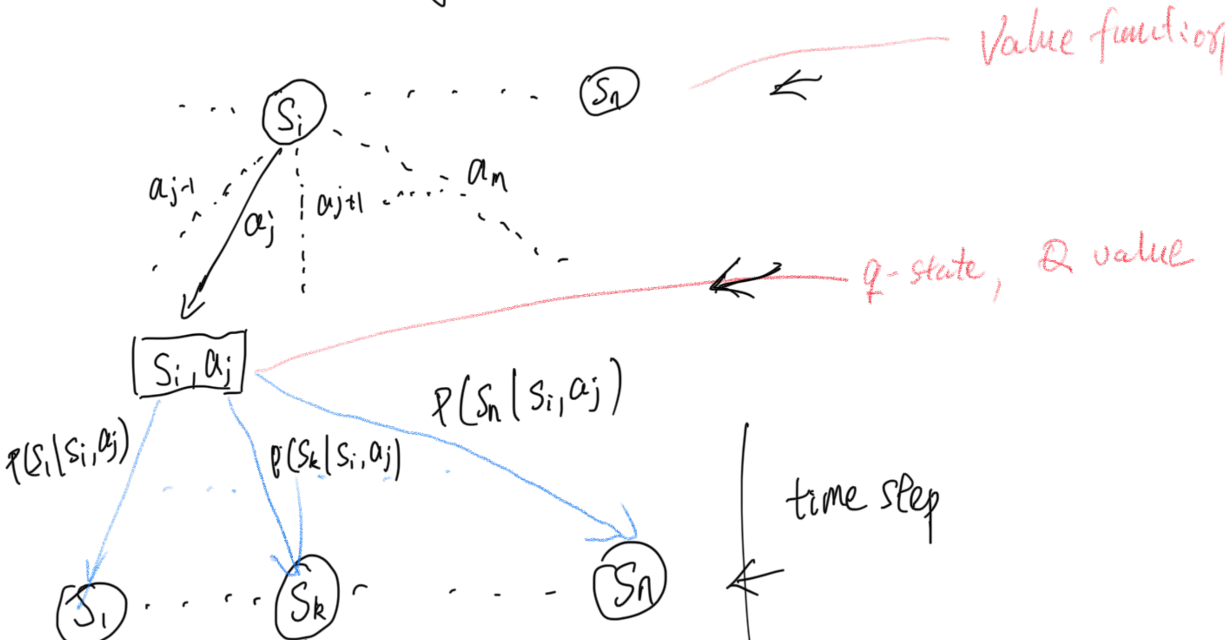
discount : 0.5

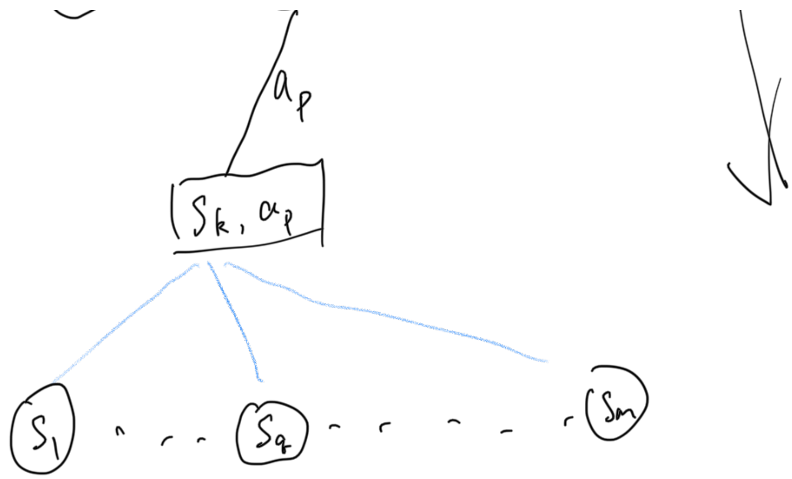
$R_1 = 1 + 2 \times 0.5 + 3 \times 0.5 \times 0.5 = 2.75$

$R_2 = 3 + 2 \times 0.5 + 1 \times 0.5 \times 0.5 = 4.25$

Goal : find or generate a policy that can get the max possible sum of discounted rewards.

Policy is a handbook that tells the robot which action to take at a given state.





$V(s)$ — expected all (sum of) future rewards in state s.

$Q(s,a)$ — expected all (sum of) future rewards, after taking action a, at state s.

$$\begin{cases} V(s) = \max\limits_{a_i} Q(s, a_i) \quad i = 1 \to m \\ Q(s,a) = r_a + \sum\limits_{s'} P(s' \mid s, a)\left(r(s) + \cancel{\gamma}^{\lambda} V(s')\right) \end{cases}$$

$$V(s) = \max_{a_i} Q(s, a_i) \quad i = 1 \to m$$

$\vdots$

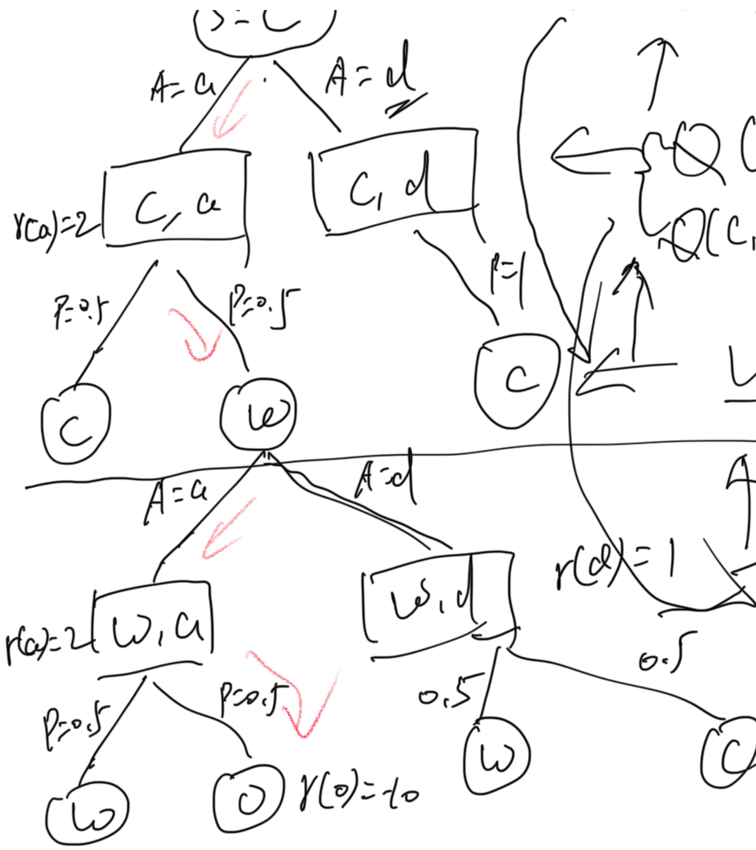
Bellman equations.

$S = \{c, w, o\}$, $A = \{a, d\}$, $r(a) = 2$, $r(d) = 1$, $r(o) = -10$.

$P(c|c,a) = 0.5$, $P(c|c,d) = 1$, $P(o|w,a) = 0.5$, $P(c|w,d) = 0.5$

$P(w|c,a) = 0.5$, $P(w|c,d) = 0$, $P(w|w,a) = 0.5$, $P(w|w,d) = 0.5$

$\leftarrow V(C) = 6$



0.25(15)

$Q(C,a) = 2 + \underline{0.5 \cdot 0.5} \cdot 5 + \underline{0.5 \cdot 0.5} \cdot 10 = 5.$

$Q(C,d) = 1 + 0.5 \cdot 10 = 6$

expecte

$\underline{V^*(c) = 5}, \ \underline{V^*(w) = 10}$

$\gamma = 0.5$

Q

V

Q - learning algorithm, step size $\alpha \in (0, 1]$

initialize $Q(S, a)$, for S, a, λ - discount

Loop for each episode:

choose A at S based on Q - value (ϵ-greedy)

Take action A, get reward, reach a state S'

compute Q.

ϵ-greedy:

pick a ϵ value

$\underset{\text{percentage}}{\underline{(1-\epsilon)}}$ of steps: act greedi

ϵ of steps: random

$$Q(S, A) \leftarrow \underline{Q^*(S, A)} + \alpha [\underbrace{r(a) + \lambda \max_a Q^*(s', a)}_{\text{new } Q} - \underbrace{Q^*(S, A)}_{\text{old } Q}]$$

new Q $\quad Q = r(a) + \sum_{s'} P \lambda \underline{V(s')} + r(s')$

$V(s') = \max Q(s', a)$

new Q $Q = r(a) + \sum_{s'} p \lambda \max_{a} Q^{*}(s', a) + r(s')$

from s, take action a, reach s'

$s \Leftarrow s'$

until s reach end state, or for certain # of steps.

learns $Q(s, a)$

$$\begin{array}{c|c|c|c|c}
 & s_1 & \cdots & s_i & \cdots & s_n \\
\hline
a_1 & Q(s_1, a_1) & \cdots & & \cdots & Q(s_n, a_1) \\
\vdots & \vdots & & & \ddots & \vdots \\
a_m & Q(s_1, a_m) & & & & Q(s_n, a_m)
\end{array}$$