Reinforcement Learning
Part 1
CS 760@UW-Madison
Goals for the lecture

you should understand the following concepts

• the reinforcement learning task
• Markov decision process
• value functions
• value iteration
• Q functions
• Q learning
Reinforcement learning (RL)

Task of an agent embedded in an environment

repeat forever

1) sense world
2) reason
3) choose an action to perform
4) get feedback (usually reward = 0)
5) learn

the environment may be the physical world or an artificial one
Reinforcement learning

- set of states $S$
- set of actions $A$
- at each time $t$, agent observes state $s_t \in S$ then chooses action $a_t \in A$
- then receives reward $r_t$ and changes to state $s_{t+1}$
RL as Markov decision process (MDP)

- Markov assumption
  \[ P(s_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots) = P(s_{t+1} \mid s_t, a_t) \]

- also assume reward is Markovian
  \[ P(r_{t+1} \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots) = P(r_{t+1} \mid s_t, a_t) \]

Goal: learn a policy \( \pi : S \rightarrow A \) for choosing actions that maximizes

\[ E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots] \quad \text{where } 0 \leq \gamma < 1 \]

for every possible starting state \( s_0 \)
Reinforcement learning task

• Suppose we want to learn a control policy \( \pi : S \rightarrow A \) that maximizes \( \sum_{t=0}^{\infty} \gamma^t E[r_t] \) from every state \( s \in S \).

Each arrow represents an action \( a \) and the associated number represents deterministic reward \( r(s, a) \).
Value function for a policy

• given a policy $\pi : S \rightarrow A$ define

$$V^\pi (s) = \sum_{t=0}^{\infty} \gamma^t E[r_t]$$

assuming action sequence chosen according to $\pi$ starting at state $s$

• we want the optimal policy $\pi^*$ where

$$^* = \arg \max \ V (s) \ \text{for all} \ s$$

we’ll denote the value function for this optimal policy as $V^*(s)$
Value function for a policy $\pi$

- Suppose $\pi$ is shown by red arrows, $\gamma = 0.9$

$$V^\pi(s)$$ values are shown in red
Value function for an optimal policy $\pi^*$

- Suppose $\pi^*$ is shown by red arrows, $\gamma = 0.9$

$V^*(s)$ values are shown in red
Using a value function

If we know $V^*(s)$, $r(s_t, a)$, and $P(s_t | s_{t-1}, a_{t-1})$ we can compute $\pi^*(s)$

$$\pi^*(s_t) = \arg \max_{a \in A} \left[ r(s_t, a) + \gamma \sum_{s \in S} P(s_{t+1} = s | s_t, a) V^*(s) \right]$$
Value iteration for learning $V^*(s)$

initialize $V(s)$ arbitrarily
loop until policy good enough
{
    loop for $s \in S$
    {
        loop for $a \in A$
        {
            $Q(s, a) \leftarrow r(s, a) + \gamma \sum_{s' \in S} P(s'|s, a)V(s')$
        }
        $V(s) \leftarrow \max_a Q(s, a)$
    }
}
Value iteration for learning $V^*(s)$

- $V(s)$ converges to $V^*(s)$
- works even if we randomly traverse environment instead of looping through each state and action methodically
  - but we must visit each state infinitely often
- implication: we can do online learning as an agent roams around its environment

- assumes we have a model of the world: i.e. know $P(s_t | s_{t-1}, a_{t-1})$
- What if we don’t?
**Q learning**

define a new function, closely related to $V^*$

\[
V^*(s) \leftarrow E[r(s, \pi^*(s))] + \gamma E_{s'|s,\pi^*(s)}[V^*(s')]
\]

\[
Q(s, a) \leftarrow E[r(s, a)] + \gamma E_{s'|s,a}[V^*(s')]
\]

if agent knows $Q(s, a)$, it can choose optimal action without knowing $P(s' | s, a)$

\[
\pi^*(s) \leftarrow \arg \max_a Q(s,a) \quad V^*(s) \leftarrow \max_a Q(s,a)
\]

and it can learn $Q(s, a)$ without knowing $P(s' | s, a)$
$Q$ values

$r(s, a)$ (immediate reward) values

$Q(s, a)$ values

$V^*(s)$ values
for each $s$, $a$ initialize table entry $\hat{Q}(s,a) \leftarrow 0$

observe current state $s$

do forever

select an action $a$ and execute it

receive immediate reward $r$

observe the new state $s'$

update table entry

$$\hat{Q}(s,a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s',a')$$

$s \leftarrow s'$
Updating $Q$

\[
\hat{Q}(s_1, a_{right}) \leftarrow r + \gamma \max_{a'} \hat{Q}(s_2, a')
\]
\[
\leftarrow 0 + 0.9 \max\{63, 81, 100\}
\]
\[
\leftarrow 90
\]
for each $s, a$ initialize table entry $\hat{Q}(s, a) \leftarrow 0$

observe current state $s$

do forever

select an action $a$ and execute it

receive immediate reward $r$

observe the new state $s'$

update table entry

$$\hat{Q}_n(s, a) \leftarrow (1 - \alpha_n)\hat{Q}_{n-1}(s, a) + \alpha_n[r + \gamma \max_a \hat{Q}_{n-1}(s', a')]$$

$s \leftarrow s'$

where $\alpha_n$ is a parameter dependent on the number of visits to the given $(s, a)$ pair

$$n = \frac{1}{1 + \text{visits}_n(s, a)}$$
Convergence of $Q$ learning

- $Q$ learning will converge to the correct $Q$ function
  - in the deterministic case
  - in the nondeterministic case (using the update rule just presented)

- in practice it is likely to take many, many iterations
THANK YOU

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, and Pedro Domingos.