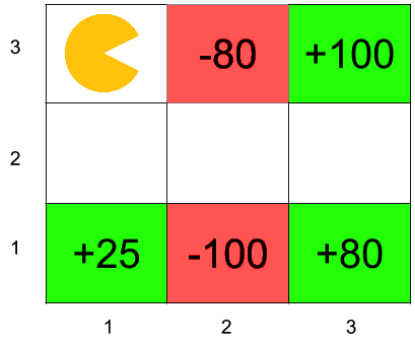
CS188: Exam Practice Session 5 Solutions

Q1. Q-Learning

Consider the grid-world given below and Pacman who is trying to learn the optimal policy. If an action results in landing into one of the shaded states, the corresponding reward is awarded during that transition. All shaded states are terminal states, i.e., the MDP terminates once arrived in a shaded state. The other states have the *North, East, South, West* actions available, which deterministically move Pacman to the corresponding neighboring state (or have Pacman stay in place if the action tries to move out of the grid). Assume the discount factor $\gamma = 0.5$ and the Q-learning rate $\alpha = 0.5$ for all calculations. Pacman starts in state (1, 3).



(a) What is the value of the optimal value function V^* at the following states:

$V^*(3,2) =$ 100 $V^*(2,2) =$ 50 $V^*(1,3) =$ 12.5

The optimal values for the states can be found by computing the expected reward for the agent acting optimally from that state onwards. Note that you get a reward when you transition *into* the shaded states and not *out* of them. So for example the optimal path starting from (2,2) is to go to the +100 square which has a discounted reward of $0 + \gamma * 100 = 50$. For (1,3), going to either of +25 or +100 has the same discounted reward of 12.5.

(b) The agent starts from the top left corner and you are given the following episodes from runs of the agent through this grid-world. Each line in an Episode is a tuple containing (s, a, s', r).

Episode 1	Episode 2	Episode 3
(1,3), S, (1,2), 0	(1,3), S, (1,2), 0	(1,3), S, (1,2), 0
(1,2), E, (2,2), 0	(1,2), E, (2,2), 0	(1,2), E, (2,2), 0
(2,2), S, (2,1), -100	(2,2), E, (3,2), 0	(2,2), E, (3,2), 0
	(3,2), N, (3,3), +100	(3,2), S, (3,1), +80

Using Q-Learning updates, what are the following Q-values after the above three episodes:

$Q((3,2),\text{N}) =$ 50 $Q((1,2),\text{S}) =$ 0 $Q((2,2),E) =$ 12.5

Q-values obtained by Q-learning updates - $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha(R(s,a,s') + \gamma \max_{a'} Q(s',a'))$.

(c) Consider a feature based representation of the Q-value function:

$$Q_f(s,a) = w_1 f_1(s) + w_2 f_2(s) + w_3 f_3(a)$$

$f_1(s)$: The x coordinate of the state $f_2(s)$: The y coordinate of the state

$$f_3(N) = 1,\ f_3(S) = 2,\ f_3(E) = 3,\ f_3(W) = 4$$

(i) Given that all w_i are initially 0, what are their values after the first episode:

$w_1 =$ -100 $w_2 =$ -100 $w_3 =$ -100

Using the approximate Q-learning weight updates: $w_i \leftarrow w_i + \alpha[(R(s,a,s') + \gamma \max_{a'} Q(s',a')) - Q(s,a)]f_i(s,a)$. The only time the reward is non zero in the first episode is when it transitions into the -100 state.

(ii) Assume the weight vector w is equal to $(1,1,1)$. What is the action prescribed by the Q-function in state $(2,2)$?

West

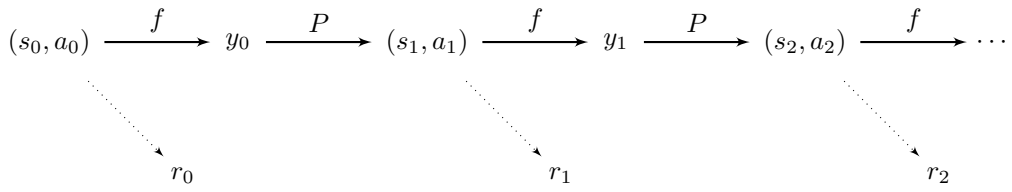
The action prescribed at (2,2) is $\max_a Q((2,2),a)$ where $Q(s,a)$ is computed using the feature representation. In this case, the Q-value for *West* is maximum $(2+2+4=8)$.

Q2. Bellman Equations for the Post-Decision State

Consider an infinite-horizon, discounted MDP (S, A, T, R, γ). Suppose that the transition probabilities and the reward function have the following form:

$$T(s, a, s') = P(s'|f(s, a))$$
$$R(s, a, s') = R(s, a)$$

Here, f is some deterministic function mapping $S \times A \to Y$, where Y is a set of states called *post-decision states*. We will use the letter y to denote an element of Y, i.e., a post-decision state. In words, the state transitions consist of two steps: a deterministic step that depends on the action, and a stochastic step that does not depend on the action. The sequence of states (s_t), actions (a_t), post-decision-states (y_t), and rewards (r_t) is illustrated below.



You have learned about $V^\pi(s)$, which is the expected discounted sum of rewards, starting from state s, when acting according to policy π.

$$V^\pi(s_0) = E\left[R(s_0, a_0) + \gamma R(s_1, a_1) + \gamma^2 R(s_2, a_2) + \dots\right]$$
$$\text{given } a_t = \pi(s_t) \text{ for } t = 0, 1, 2, \dots$$

$V^*(s)$ is the value function of the optimal policy, $V^*(s) = \max_\pi V^\pi(s)$.

This question will explore the concept of computing value functions on the post-decision-states y. [1]

$$W^\pi(y_0) = E\left[R(s_1, a_1) + \gamma R(s_2, a_2) + \gamma^2 R(s_3, a_3) + \dots\right]$$

We define $W^*(y) = \max_\pi W^\pi(y)$.

(a) Write W^* in terms of V^*.
$W^*(y) =$

- ● $\sum_{s'} P(s' \mid y) V^*(s')$
- ○ $\sum_{s'} P(s' \mid y)[V^*(s') + \max_a R(s', a)]$
- ○ $\sum_{s'} P(s' \mid y)[V^*(s') + \gamma \max_a R(s', a)]$
- ○ $\sum_{s'} P(s' \mid y)[\gamma V^*(s') + \max_a R(s', a)]$
- ○ None of the above

Consider the expected rewards under the optimal policy.

$$\begin{aligned} W^*(y_0) &= E\left[R(s_1, a_1) + \gamma R(s_2, a_2) + \gamma^2 R(s_3, a_3) + \dots \mid y_0\right] \\ &= \sum_{s_1} P(s_1 \mid y_0) E\left[R(s_1, a_1) + \gamma R(s_2, a_2) + \gamma^2 R(s_3, a_3) + \dots \mid s_1\right] \\ &= \sum_{s_1} P(s_1 \mid y_0) V^*(s_1) \end{aligned}$$

[1] In some applications, it is easier to learn an approximate W function than V or Q. For example, to use reinforcement learning to play Tetris, a natural approach is to learn the value of the block pile *after* you've placed your block, rather than the value of the pair (current block, block pile). TD-Gammon, a computer program developed in the early 90s, was trained by reinforcement learning to play backgammon as well as the top human experts. TD-Gammon learned an approximate W function.

V^* is time-independent, so we can replace y_0 by y and replace s_1 by s', giving

$$W^*(y) = \sum_{s'} P(s' \mid y) V^*(s')$$

(b) Write V^* in terms of W^*.
$V^*(s) =$

- ○ $\max_a [W^*(f(s,a))]$
- ○ $\max_a [R(s,a) + W^*(f(s,a))]$
- ● $\max_a [R(s,a) + \gamma W^*(f(s,a))]$
- ○ $\max_a [\gamma R(s,a) + W^*(f(s,a))]$
- ○ None of the above

$$\begin{aligned}
V^*(s_0) &= \max_{a_0} Q(s_0, a_0) \\
&= \max_{a_0} E\left[R(s_0,a_0) + \gamma R(s_1,a_1) + \gamma^2 R(s_2,a_2) + \dots \mid s_0, a_0\right] \\
&= \max_{a_0} \left(E\left[R(s_0,a_0) \mid s_0,a_0\right] + E\left[\gamma R(s_1,a_1) + \gamma^2 R(s_2,a_2) + \dots \mid s_0,a_0\right]\right) \\
&= \max_{a_0} \left(R(s_0,a_0) + E\left[\gamma R(s_1,a_1) + \gamma^2 R(s_2,a_2) + \dots \mid f(x_0,a_0)\right]\right) \\
&= \max_{a_0} \left(R(s_0,a_0) + \gamma W^*(f(s_0,a_0))\right)
\end{aligned}$$

Renaming variables, we get

$$V^*(s) = \max_a \left(R(s,a) + \gamma W^*(f(s,a))\right)$$

(c) Recall that the optimal value function V^* satisfies the Bellman equation:

$$V^*(s) = \max_a \sum_{s'} T(s,a,s') \left(R(s,a) + \gamma V^*(s')\right),$$

which can also be used as an update equation to compute V^*.
Provide the equivalent of the Bellman equation for W^*.

$W^*(y) =$ $\sum_{s'} P(s'|y) \max_a \left(R(s',a) + \gamma W^*(f(s',a))\right)$

The answer follows from combining parts (a) and (b)

(d) Fill in the blanks to give a policy iteration algorithm, which is guaranteed return the optimal policy π^*.

- Initialize policy $\pi^{(1)}$ arbitrarily.
- For $i = 1, 2, 3, \dots$
  - Compute $W^{\pi^{(i)}}(y)$ for all $y \in Y$.
  - Compute a new policy $\pi^{(i+1)}$, where $\pi^{(i+1)}(s) = \arg\max_a$ ___(1)___ for all $s \in S$.
  - If ___(2)___ for all $s \in S$, **return** $\pi^{(i)}$.

Fill in your answers for blanks (1) and (2) below.

(1)
- ○ $W^{\pi^{(i)}}(f(s,a))$
- ○ $R(s,a) + W^{\pi^{(i)}}(f(s,a))$
- ● $R(s,a) + \gamma W^{\pi^{(i)}}(f(s,a))$
- ○ $\gamma R(s,a) + W^{\pi^{(i)}}(f(s,a))$
- ○ None of the above

(2) $\pi^{(i)}(s) = \pi^{(i+1)}(s)$

Policy iteration performs the following update:

$$\pi^{(i+1)}(s) = \arg\max_a Q^{\pi^{(i)}}(s,a)$$

Next we express Q^π in terms of W^π (similarly to part b):

$$\begin{aligned} Q^\pi(s_0,a_0) &= E\left[R(s_0,a_0) + \gamma R(s_1,a_1) + \gamma^2 R(s_2,a_2) + \dots \;\middle|\; s_0,a_0\right] \\ &= R(s_0,a_0) + \gamma E\left[R(s_1,a_1) + \gamma R(s_2,a_2) + \dots \;\middle|\; f(s_0,a_0)\right] \\ &= R(s_0,a_0) + \gamma W^\pi(f(s_0,a_0)) \end{aligned}$$

(e) In problems where f is known but $P(s'|y)$ is not necessarily known, one can devise reinforcement learning algorithms based on the W^* and W^π functions. Suppose that an the agent goes through the following sequence of states, actions and post-decision states: $s_t, a_t, y_t, s_{t+1}, a_{t+1}, y_{t+1}$. Let $\alpha \in (0,1)$ be the learning rate parameter. Write an update equation analogous to Q-learning that enables one to estimate W^*

$$W(y_t) \leftarrow (1-\alpha)W(y_t) + \alpha\left(\max_a \left(R(s_{t+1},a) + \gamma W(f(s_{t+1},a))\right)\right)$$

Recall the motivation for Q learning: the Bellman equation satisfied by the Q function is $Q(s,a) = E_{s'}[R(s,a) + \gamma \max_{a'} Q(s',a')]$. Q learning uses an estimate of the right-hand side of this equation obtained from a single sample transition s, a, s'. That is, we move $Q(s,a)$ towards $R(s,a) + \gamma \max_{a'} Q(s',a')$.

We have obtained a Bellman-like equation for W^* in part c.

$$\begin{aligned} W^*(y) &= \sum_{s'} P(s'|y) \max_{a'} \left(R(s',a') + \gamma W^*(f(s',a'))\right) \\ &= E_{s'}\left[\max_a \left(R(s',a) + \gamma W^*(f(s',a))\right)\right] \end{aligned}$$

If we don't know $P(s'|y)$, we can still estimate the expectation above from a single transition (y, s') using the expression inside the expectation: $\max_a \left(R(s',a) + \gamma W^*(f(s',a))\right.$. Doing a partial update with learning rate parameter α, we get

$$W(y_t) \leftarrow (1-\alpha)W(y_t) + \alpha \max_a \left(R(s_{t+1},a) + \gamma W(f(s_{t+1},a))\right.$$