1. (a) $\quad V_*(s) = \max_a q_*(s.a)$

(b) $\quad q_*(s,a) = \max \sum_{} P(s',r|s,a) V_*(s)$

(c) $\quad \pi(a|s) = \max q_*(s.a)$

(d) $\quad \pi(a|s) = \max_a \sum_{s,r} P(s',r|s,a)\left[r + \gamma V_*(s)\right]$

(e) $\quad V_\pi(s) = \sum_a \pi(a|s) \sum_{s'} P(s'|s.a)\left[r(s.a) + V_\pi(s')\right]$

$\quad V_*(s) = \max_a \sum_{s'} P(s'|s.a)\left[r(s.a) + V_*(s')\right]$

$\quad q_\pi(s.a) = \sum_{s'} P(s'|s.a)\left[r(s.a) + \gamma \sum_{a'} \pi(a'|s') q_\pi(s',a')\right]$

$\quad q_*(s,a) = \sum_{s'} P(s'|s.a)\left[r(s.a) + \gamma \max_{a'} q_*(s'.a')\right]$

2. (b) The following algorithm is the entire definition for Policy iteration.

   1. input. $q_*(s.a) = 0$, $s \in S^+$. $a \in A$

   2.# Policy evaluation.
      Loop:
         $\Delta \leftarrow 0$
         Loop for each $s \in S^+$:
            $v = V(s)$
            $V(s) = \sum_{s',r} P(s',r|s. \pi(s))\left[r + \gamma V(s')\right]$
            $\Delta \leftarrow \max(\Delta, |v - V(s)|)$

Until $\Delta < \theta$

3#Policy Improvement

Policy-Stable ← true

For each $s \in S^*$

old action ← $\bar{\pi}(s)$

$\pi(s)$ ← arg max$_a$ $\sum_{s',r} P(s',r|s,a)[r + \gamma V(s)]$

If old-action $\neq \bar{\pi}(s)$, Policy-Stable ← false.

If Policy-Stable

break

return $V \approx V_*$ $\pi = \pi_*$

else: go to 2.

(b)

$$q_{k+1}(s,a) = \sum_{s',r} P(s',r|s,a)[r + \gamma \max_a q_k(s',a')]$$

3.(a) Optimal Policy at x: keep taking action c until get z

Optimal Policy at y: keep taking action b until get x

then keep taking action c until get z

(b) Policy evaluation $\pi = c$

$V_x = [0.9 \times (-1 + V_x) + 0.1 \times (-1 + V_z)] = -1 + 0.9 V_x$

$V_x = -10$

$V_y = [0.9 \times (-2 + V_y) + 0.1 \times (-1 + V_z)] = -2 + 0.9 V_y$

$V_y = -20$

Policy Improvement:

Iter1 - X: $V_{\pi c} = -1 + 0.9\, V_x = -10$

$V_{\pi b} = 0.8 \times (-2 + V_y) + 0.2 \times (-1 + V_x) = -1.6 + 0.8\, V_y - 0.2 + 0.2\, V_x$

$\qquad = -1.8 + 0.8\, V_y + 0.2\, V_x$

$\qquad = -19.8$

$\Rightarrow$ new action = argmax $(V_{\pi c}, V_{\pi b})$

$\qquad = \pi_c$

$\Rightarrow$ new action = old action

$\Rightarrow$ policy stable

Y: $V_{\pi c} = -2 + 0.9\, V_y$

$\qquad = -20$

$V_{\pi b} = 0.8 \times (-1 + V_x) + 0.2 \times (-2 + V_y) =$

$\qquad = -0.8 + 0.8\, V_x - 0.4 + 0.2\, V_y$

$\qquad = -1.2 - 8 - 4$

$\qquad = -13.2$

$\Rightarrow$ new action = argmax $(V_{\pi c}, V_{\pi b})$

$\qquad = \pi_b$

$\Rightarrow$ new action $\neq$ old action

$\Rightarrow$ policy not stable

Iter-2: y: $V_{xc} = -2 + 0.9 V_y$

$\qquad = -20$

$\qquad V_{xb} = 0.8 \times (-1 + V_x) + 0.2 \times (-2 + V_y) =$

$\qquad\qquad = -0.8 + 0.8 V_x - 0.4 + 0.2 V_y$

$\qquad\qquad = -1.2 - 8 - 4$

$\qquad\qquad = -13.2$

$\qquad\qquad \Rightarrow$ newaction = argmax (V_{xc}, V_{xb})

$\qquad\qquad\qquad = a_b$

$\qquad \Rightarrow$ newaction = old action

$\qquad \Rightarrow$ policy stable

(c) Policy evaluation $\pi = b$

$\qquad V_x = -2 + 0.8 V_y - 1 + 0.2 V_x$

$\qquad V_x = -3 + 0.8 V_y + 0.2 V_x \qquad \Rightarrow 0.8 V_x = -3 + 0.8 V_y$

$\qquad V_y = -1 + 0.8 V_x - 2 + 0.2 V_y$

$\qquad V_y = -3 + 0.8 V_x + 0.2 V_y. \qquad \Rightarrow 0.8 V_y = -3 + 0.8 V_x$

$\qquad\qquad\qquad\qquad \Rightarrow$ No solution

add discounting γ

$\qquad \Rightarrow V_x = -2 + 0.8 \gamma V_y - 1 + 0.2 \gamma V_x$

$\qquad (1 - 0.2\gamma) V_x = -3 + 0.8 \gamma V_y$

$$\Rightarrow V_y = -1 + 0.8\gamma V_x - 2 + 0.2\gamma V_y$$

$$(1 - 0.2\gamma)V_y = -3 + 0.8\gamma V_x$$

$$\Rightarrow \text{solution exist} \quad \Rightarrow \text{discounting help}$$

if $\gamma = 0.1$ Policy evolution,

$$0.98\, V_x = -3 + 0.08\, V_y$$

$$0.98\, V_y = -3 + 0.08\, V_x$$

$$\Rightarrow 12.005\, V_y = -3 + 0.98\, V_x$$

$$\Rightarrow 12.005\, V_y + 3 = -3 + 0.08\, V_y$$

$$11.925\, V_y = -6$$

$$V_y = -0.503$$

$$0.98\, V_x = -3 - 0.04$$

$$V_x = -3.1$$

if $\gamma = 0.5$ Policy evolution,

$$0.9\, V_x = -3 + 0.4\, V_y$$

$$0.9\, V_y = -3 + 0.4\, V_x$$

$$2.025\, V_y = -6.75 + 0.9\, V_x$$

$$2.025\, V_y + 6.75 = -3 + 0.4\, V_y$$

Policy improvement

Iter1 $\quad \pi_1(x) = \text{argmax}\left(V_{\pi_c}, V_{\pi_b}\right)$

$$= C$$

$$\pi_1(y) = \text{argmax}\left(V_{\pi_c}, V_{\pi_b}\right)$$

$$= b$$

$$\pi_1(y) = b = \pi_0(y)$$

$$\Rightarrow \text{stable}$$

Iter $\quad \pi_2(x) = \text{argmax}\left(V_{\pi_c}, V_{\pi_b}\right)$

$$= c$$

$$\pi_2(x) = C = \pi_1(x)$$

$$\Rightarrow \text{stable}$$

Policy improvement

Iter1 $\quad \pi_1(x) = \text{argmax}\left(V_{\pi_c}, V_{\pi_b}\right)$

$$= C$$

$$\pi_1(y) = \text{argmax}\left(V_{\pi_c}, V_{\pi_b}\right)$$

$$= b$$

$$\pi_1(y) = b = \pi_0(y)$$

$$\Rightarrow \text{stable}$$

$$1.625 V_y = -9.75$$
$$V_y = -6$$
$$V_p = -9.56$$

Iter $\pi_2(x) = \arg\max(V_{nc}, V_{ab})$
$$= C$$
$$\pi_2(x) = C = \pi_1(x)$$
$$\Rightarrow \text{stable}$$

$$\pi(\gamma = 0.5) = \pi(\gamma = 0.1)$$
$$\Rightarrow \text{Optimal policy do not depend on } \gamma$$

5(b)

The reward have been changed as following:

    One car can be moved from lot1 to lot2 free;

    If cars in parking lot1>10, additional $4 are charged, If cars in both parking lot1 and lot2 >10, additional $8 are charged.

The policy become non-linear because of the modification on reward, the major difference is that in previous policy, there is no action needed when parking number around 10. But since Jack need to pay for extra money if a location have>10 cars, there are actions needed when the number of cars in a parking lot is close to 10 to avoid penalty.