PS9

● Graded

Student

Chetan Hiremath

Total Points

98 / 100 pts

Question 1

9.1 Lottery Tickets — **10** / 10 pts

1.1 **(a) EMV** — **5** / 5 pts

✔ **+ 5 pts** Correct

1.2 **(b) rest** — **5** / 5 pts

✔ **+ 5 pts** Correct

Question 2

Buying a Used Car — **18** / 20 pts

2.1 **(b)** — **7** / 7 pts

✔ **+ 7 pts** Correct

2.2 **(c)** — **7** / 7 pts

✔ **+ 7 pts** Correct

2.3 **(d)** — Resolved — **4** / 6 pts

✔ **+ 6 pts** Correct

✔ **– 2 pts** Decision - Since Both>0, Optimal Decision is "Buy IN Both Cases"

Regrade Request — **Submitted on: Apr 16**

Why I have lost 2 points on this question when my solutions are correct? Will you explain your comment to me and regrade this question?

your calculation is correct. You did not mention the decision.

Reviewed on: Apr 18

Question 3

Sequential Movements **20** / 20 pts

3.1 **(a) [Up, Up]** **10** / 10 pts

✔ **+ 10 pts** Correct

3.2 **(b) [Up, Up, Right]** **10** / 10 pts

✔ **+ 10 pts** Correct

Question 4

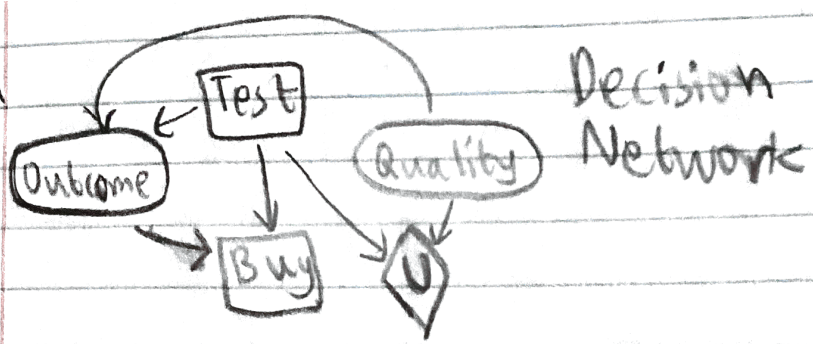
Value Iteration by Computer **50** / 50 pts

✔ **+ 50 pts** Correct

Question assigned to the following page: 2.1

2a.

2a.



Decision Network

b. $E = P(q^+)(\$2000 - \$1500) + P(q^-)(\$2000 - \$1500 - \$700)$

$= (0.70)(\$500) + (0.30)(-\$200)$

$= \$350 - \$60$

$= \$290.$

Question assigned to the following page: 1.1

1a. $EMV = \frac{1}{50}(\$10) + \frac{1}{2000000}(\$1{,}000{,}000)$

$= \$\frac{1}{5} + \$\frac{1}{2}$

$= \$\frac{2}{10} + \$\frac{5}{10}$

$= \$\frac{7}{10}$

$= \$0.70$

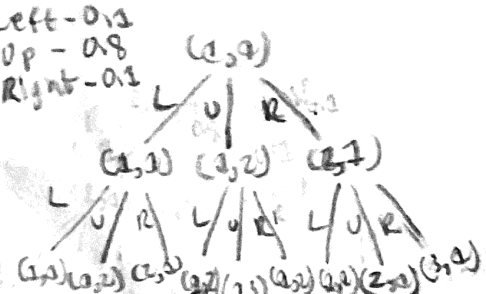
It's not rational to buy a ticket since $\$0.70 < \$1$.

Question assigned to the following page: 3.1

Chetan H.
4/9/24

3a.

L/Left - 0.1
U/Up - 0.8
R/Right - 0.1



[Up, Up]

3	0.8(0.8) = 0.64	0	0	+1
2	0.2(0.8) + 0.8(0.2) + 0.8(0.2) = 0.24		0	−1
1	0.2(0.2) + 0.2(0.2) = 0.02 (start)	0.2(0.2) + 0.2(0.8) = 0.09	0.2(0.2) = 0.01	0
	1	2	3	4

P(1,2) + P(1,2) + P(3,3) + P(2,1) + P(3,1) = 0.64 + 0.24 + 0.02 + 0.09 + 0.01 = 1.

Question assigned to the following page: 3.2

b.

[Up, Up, Right]

	1	2	3	4
3	0.64(0.1)+ 0.24(0.1)= 0.088	0.64(0.8)= 0.512	0	+1
2	0.64(0.1)+ 0.24(0.8)+ 0.02(0.1)= 0.258		0.02(0.1)= 0.001	−1
1	0.24(0.1)+ 0.02(0.1)= 0.026 Start	0.02(0.8)+ 0.09(0.1)= 0.034	0.09(0.8)+ 0.02(0.1)= 0.073	0.02(0.8)= 0.008

$P(1,1)+P(1,2)+P(1,3)+$
$P(2,1)+P(2,3)+P(3,1)+$
$P(3,2)+P(4,1) = 0.026+$
$0.258+0.088+0.034+$
$0.512+0.073+0.001+$
$0.009 = 1$

Question assigned to the following page: 1.2

b. $EU = \frac{1}{50} U(S_{k+20}) + \frac{1}{2000000} U(S_{k+2,000,000})$

$= \frac{1}{50}(100\, U(S_{k+2})) + \frac{1}{2000000} U(S_{k+2,000,000})$

$= \frac{2}{5} U(S_{k+2}) + \frac{1}{2000000} U(S_{k+2,000,000})$

$\frac{2}{5} U(S_{k+2}) + \frac{1}{2000000} U(S_{k+2,000,000}) > U(S_{k+2})$

$\frac{1}{2000000} U(S_{k+2,000,000}) > \frac{4}{5} U(S_{k+2})$

$U(S_{k+2,000,000}) > 1,600,000\, U(S_{k+2})$ It's greater than $U(S_{k+2})$ by this condition.

$U(S_{k+2,000,000}) > 1,600,000\, U(\$1)$

Question assigned to the following page: 2.2

c. $P(\text{Pass}) = P(q^+) P(\text{Pass} | q^+) + P(q^-) P(\text{Pass} | q^-)$

$= 0.70(0.80) + 0.30(0.35)$

$= 0.665$

$P(\text{not Pass}) = 1 - P(\text{Pass})$

$= 1 - 0.665$

$= 0.335$

Question assigned to the following page: 2.2

$$P(q^+ \mid Pass) = \frac{P(Pass \mid q^+)\,P(q^+)}{P(Pass)} = \frac{0.80(0.70)}{0.665} = 0.842$$

$$P(q^- \mid Pass) = \frac{P(Pass \mid q^-)\,P(q^-)}{P(Pass)} = \frac{0.35(0.30)}{0.665} = 0.158$$

$$P(q^+ \mid not\ Pass) = \frac{P(not\ Pass \mid q^+)\,P(q^+)}{P(not\ Pass)} = \frac{0.20(0.70)}{0.335} = 0.418$$

$$P(q^- \mid not\ Pass) = \frac{P(not\ Pass \mid q^-)\,P(q^-)}{P(not\ Pass)} = \frac{0.65(0.30)}{0.335} = 0.582$$

Question assigned to the following page: 2.3

d. $EU(Pass) = P(q^+ | Pass)(\$2000 - \$1500) + P(q^- | Pass)(\$2000 - \$1500 - \$700)$

$= 0.842(\$1500) + 0.158(-\$200)$

$= \$399.4$

$EU(Fail) = P(q^+ | not\ Pass)(\$2000 - \$1500) + P(q^- | not\ Pass)(\$2000 - \$1500 - \$700)$

$= 0.418(\$500) + 0.582(-\$200)$

$= \$92.6$

Question assigned to the following page: 4

Chetan M.
4/8/24

Source - The code is used and borrowed from Mdp.ipynb's Python code for the transition probabilities.

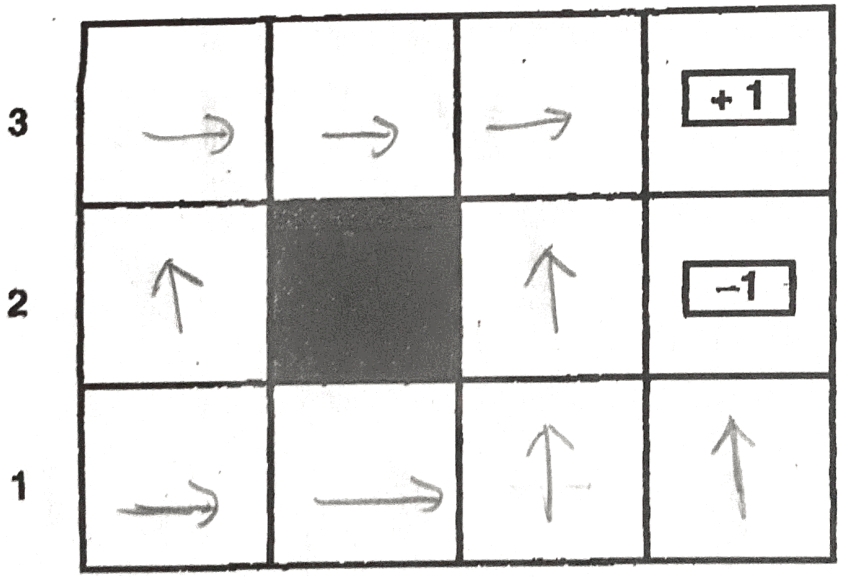
4.

	1	2	3	4
3	-3.054	-1.647	-0397	+1
2	-4.304		-1.575	-1
1	-5.296	-4.158	-2.908	-2.323

$\ell = -1$

Gamma = 1

Optimal Utility Values



Optimal Policy

Question assigned to the following page: 4

4.

Source- This code is used, borrowed, and modified from AIMA Python File: mdp.py and R&N Textbook's Figure 17.6's Value Iteration Algorithm's Pseudocode.

```
import numpy as np

R = -1
def actionRewardFunction(initialPosition, action):
    if initialPosition in termination_states:
        return initialPosition, 0
    finalPosition = np.add(initialPosition, action)
    if R in finalPosition or finalPosition[0] == gridSize[0] or
finalPosition[1] == gridSize[1] or (finalPosition == [1,1]).all():
        finalPosition = initialPosition
    return finalPosition, R
def otherActions(action):
    if action == 0 or action == 2:
        return 1, 3
    else:
        return 0, 2
```

Question assigned to the following page: 4

```
gamma = 1
gridSize = [4,3]
termination_states = [[3,2], [3,1]]
states = [[i,j] for i in range(gridSize[0]) for j in range(gridSize[1])]
states.remove([1,1])
actions = {0: [1,0], 1: [0,1], 2: [-1,0], 3: [0,-1]}
values = np.zeros((4,3))
values[3][2] = 1
values[3][1] = -1
print("Initial Iteration")
print(values)
print()

for i in range(100):
    copyValues = np.copy(values)
    for s in states:
        q_values = {a: 0 for a in actions}
        for a in actions:
            s_, reward = actionRewardFunction(s, actions[a])
            q_values[a] += 0.8*(reward + gamma*values[s_[0], s_[1]])
            for a_ in otherActions(a):
                s_, reward = actionRewardFunction(s, actions[a_])
                q_values[a] += 0.1*(reward + gamma*values[s_[0], s_[1]])
        copyValues[s[0], s[1]] = np.max(list(q_values.values()))
    comparison = values == copyValues
    values = copyValues
    if (comparison).all():
        print("Final Iteration")
        print(values)
        break
```

Question assigned to the following page: 4

Result-

```
Initial Iteration
[[ 0.  0.  0.]
 [ 0.  0.  0.]
 [ 0.  0.  0.]
 [ 0. -1.  1.]]

Final Iteration
[[-5.285664   -4.30351027 -3.05351027]
 [-4.15843322  0.         -1.64726027]
 [-2.90843322 -1.57534247 -0.39726027]
 [-2.32315925 -1.          1.        ]]
```

I have used the modified program to calculate the optimal utility values and policy. Then, I have written the results on the top and the bottom grids of the linked sheet. The result shows that the policy is going from the initial state to the +1 terminal state and avoiding the -1 terminal state. The probabilities are decreasing when it picks the path to reach the +1 terminal states, so this path is correct and optimal.