CS5446 AI Planning and Decision Making

Semester 1, AY2022-23
Assignment 2

Agrawal, Shubhankar
A0248330L

Sagar, Sanchit
A0232478Y

September 26, 2022

1 Homework Assignment

1.1 Homework Problem 1: Decision Theory

a Should Curious eat the mooncake?

The problem assumes that Curious has a pink mooncake with which she needs to make a decision of whether to eat or not.

We define the utities and probabilities of eating a mooncake with/without Durian as follows:

$$U(Eat\ Mooncake_{Durian}) = -100$$
$$U(Eat\ Mooncake_{NoDurian}) = +10$$
$$U(Don't\ eat\ Mooncake) = 0$$

$$P(Mooncake_{Durian}) = 0.1$$
$$P(Mooncake_{NoDurian}) = 0.9$$

The decision tree for this problem is represented in Figure 1

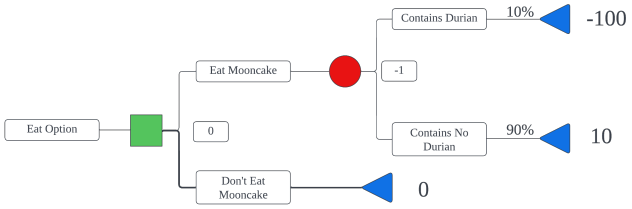

Figure 1: Decision Tree: Simple Choice

We calculate the expected utility of eating and not eating the mooncake to decide.

$$
\begin{aligned}
EU(Eat\ Mooncake) &= P(Mooncake_{Durian}) * U(Eat\ Mooncake_{Durian}) \\
&\quad + P(Mooncake_{NoDurian}) * U(Eat\ Mooncake_{NoDurian}) \\
&= (0.1 * -100) + (0.9 * 10) = -1 \\
EU(Don't\ eat\ Mooncake) &= 0
\end{aligned}
$$

Since $EU(Eat\ Mooncake) < EU(Don't\ eat\ Mooncake)$, Curious should not eat the mooncake.

b Problem Representation

Using the laws of probability and Bayes' theorem, we construct the probability for each of the nodes that would be required in our problem representations.

$$
\begin{aligned}
P(Smelled_{Durian}|Mooncake_{Durian}) &= 0.7 \\
P(Smelled_{NoDurian}|Mooncake_{Durian}) &= 0.3 \\
P(Smelled_{Durian}|Mooncake_{NoDurian}) &= 0.2 \\
P(Smelled_{NoDurian}|Mooncake_{NoDurian}) &= 0.8
\end{aligned}
$$

$$
\begin{aligned}
P(Smelled_{Durian}) &= P(Smelled_{Durian}|Mooncake_{Durian}) * P(Mooncake_{Durian}) \\
&\quad + P(Smelled_{Durian}|Mooncake_{NoDurian}) * P(Mooncake_{NoDurian}) \\
P(Smelled_{Durian}) &= 0.7 * 0.1 + 0.2 * 0.9 = 0.25
\end{aligned}
$$

$$
\begin{aligned}
P(Smelled_{NoDurian}) &= P(Smelled_{NoDurian}|Mooncake_{Durian}) * P(Mooncake_{Durian}) \\
&\quad + P(Smelled_{NoDurian}|Mooncake_{NoDurian}) * P(Mooncake_{NoDurian}) \\
P(Smelled_{NoDurian}) &= 0.3 * 0.1 + 0.8 * 0.9 = 0.75
\end{aligned}
$$

$$
\begin{aligned}
P(Mooncake_{Durian}|Smelled_{Durian}) &= \frac{P(Smelled_{Durian}|Mooncake_{Durian}) * P(Mooncake_{Durian})}{P(Smelled_{Durian})} \\
P(Mooncake_{Durian}|Smelled_{Durian}) &= \frac{0.7 * 0.1}{0.25} = 0.28
\end{aligned}
$$

$$
\begin{aligned}
P(Mooncake_{NoDurian}|Smelled_{Durian}) &= \frac{P(Smelled_{Durian}|Mooncake_{NoDurian}) * P(Mooncake_{NoDurian})}{P(Smelled_{Durian})} \\
P(Mooncake_{Durian}|Smelled_{Durian}) &= \frac{0.2 * 0.9}{0.25} = 0.72
\end{aligned}
$$

$$
\begin{aligned}
P(Mooncake_{Durian}|Smelled_{NoDurian}) &= \frac{P(Smelled_{NoDurian}|Mooncake_{Durian}) * P(Mooncake_{Durian})}{P(Smelled_{NoDurian})} \\
P(Mooncake_{Durian}|Smelled_{NoDurian}) &= \frac{0.3 * 0.1}{0.75} = 0.04
\end{aligned}
$$

$$P(Mooncake_{NoDurian}|Smelled_{NoDurian}) = \frac{P(Smelled_{NoDurian}|Mooncake_{NoDurian}) * P(Mooncake_{NoDurian})}{P(Smelled_{NoDurian})}$$

$$P(Mooncake_{NoDurian}|Smelled_{NoDurian}) = \frac{0.8 * 0.9}{0.75} = 0.96$$

The Decision Network is represented in Figure 2 below. The Decision Tree is represented in Figure 3 below.

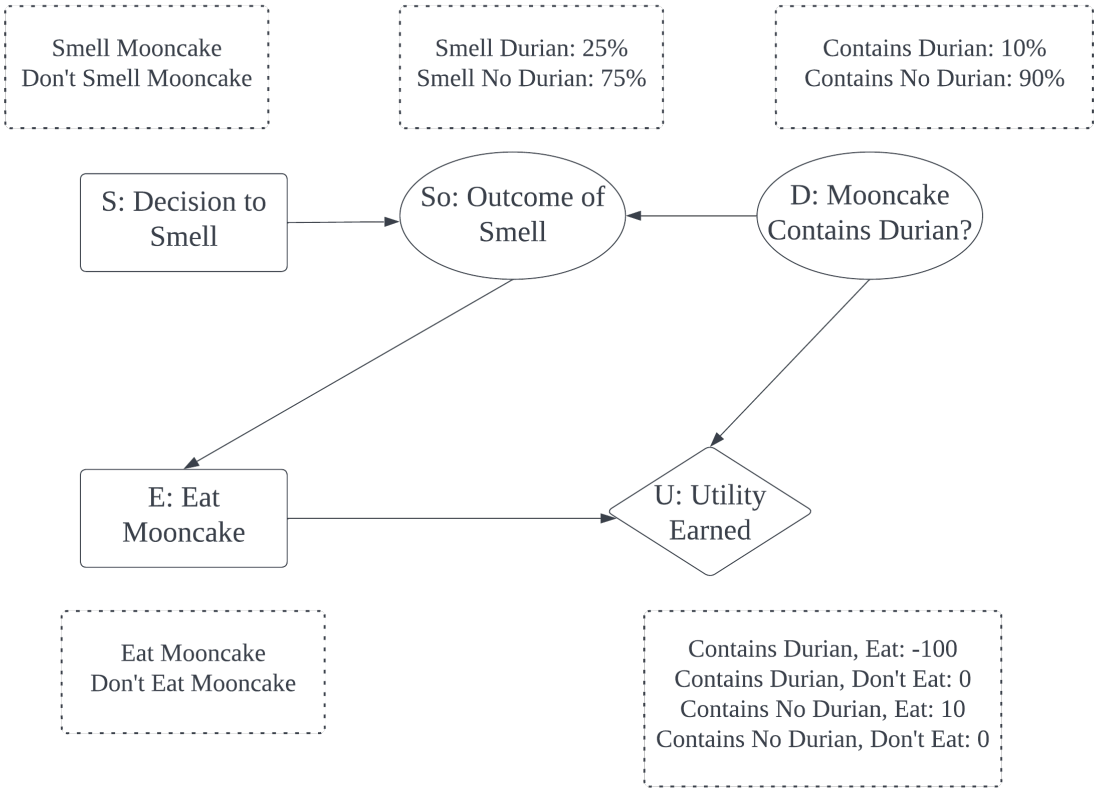

Figure 2: Decision Network

c Probability of Mooncake having Durian given smelled No Durian

Using Bayes Theorem and values from before, we have the following:

$$P(Mooncake_{Durian}|Smelled_{NoDurian}) = \frac{P(Smelled_{NoDurian}|Mooncake_{Durian}) * P(Mooncake_{Durian})}{P(Smelled_{NoDurian})}$$

$$P(Mooncake_{Durian}|Smelled_{NoDurian}) = \frac{0.3 * 0.1}{0.75} = 0.04$$

d Maximum Expected Utility given smelled No Durian

The MEU given smelled No Durian would be the maximum of eating or not eating Durian given that Curious has smelled no Durian.

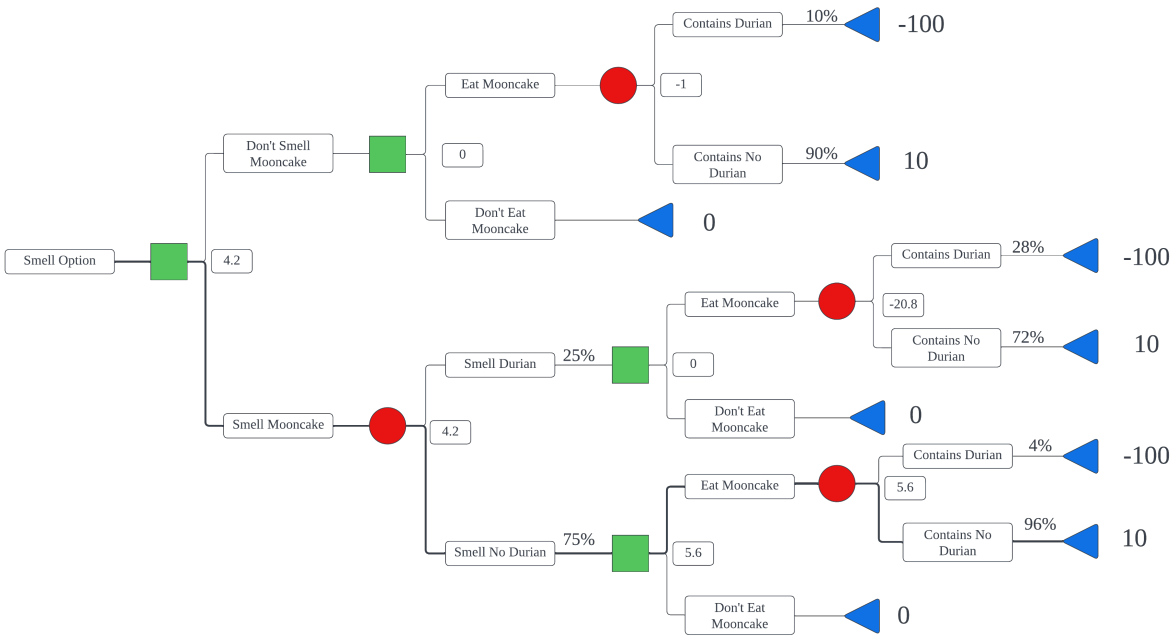

Figure 3: Decision Tree

$$
\begin{aligned}
EU(Eat\ Mooncake|Smelled_{NoDurian}) &= P(Mooncake_{Durian}|Smelled_{NoDurian}) * U(Eat\ Mooncake_{Durian}) \\
&\quad + P(Mooncake_{NoDurian}|Smelled_{NoDurian}) * U(Eat\ Mooncake_{NoDurian}) \\
&= (0.04 * -100) + (0.9 * 10) = -1 \\
EU(Don't\ eat\ Mooncake|Smelled_{NoDurian}) &= 0
\end{aligned}
$$

$$
\begin{aligned}
MEU(Smelled_{NoDurian}) &= max(EU(Eat\ Mooncake|Smelled_{NoDurian}), EU(Don't\ eat\ Mooncake|Smelled_{NoDurian})) \\
MEU(Smelled_{NoDurian}) &= 5.6\ (Eat\ Mooncake)
\end{aligned}
$$

e Value of Perfect Information

When we have perfect information, it means that smelling the mooncake tells us for sure whether the mooncake would contain durian or not. Thus, the outcome of the smell would be with the same probabilities as that of the mooncake containing durian. This is represent in the Figure 4. The value of perfect information with smelling the mooncake is given as

$$MEU(Smell\ Mooncake) - MEU(Don't\ Smell\ Mooncake)$$

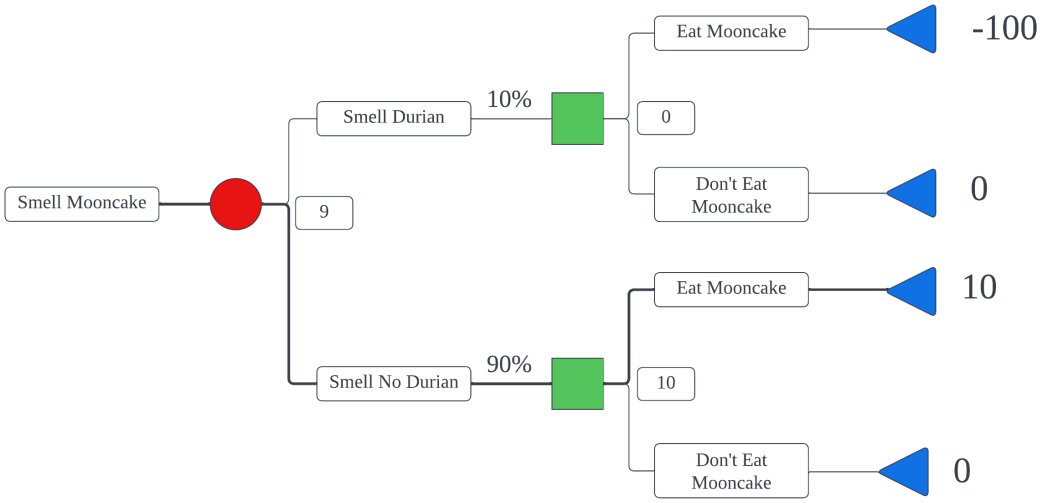

Figure 4: Decision Tree: Perfect Information of Smelling

$$
\begin{aligned}
MEU(Don't\ Smell\ Mooncake) &= max(EU(Eat\ Mooncake|Don't\ Smell\ Mooncake),\\
&\qquad EU(Don't\ Eat\ Mooncake|Don't\ Smell\ Mooncake))\\
MEU(Don't\ Smell\ Mooncake) &= max(-1,0) = 0\\
MEU(Smell\ Mooncake) &= P(Mooncake_{Durian}) * max(EU(Eat\ Mooncake|Mooncake_{Durian}),\\
&\qquad EU(Don't\ Eat\ Mooncake|Mooncake_{Durian})),\\
&\quad + P(Mooncake_{NoDurian}) * max(EU(Eat\ Mooncake|Mooncake_{NoDurian}),\\
&\qquad EU(Don't\ Eat\ Mooncake|Mooncake_{NoDurian}))\\
&= 0.1max(-100,0) + 0.9max(10,0) = 9
\end{aligned}
$$

$$
\begin{aligned}
VPI_{SmellMooncake} &= MEU(Smell\ Mooncake) - MEU(Don't\ Smell\ Mooncake)\\
VPI_{SmellMooncake} &= 9
\end{aligned}
$$

1.2 Homework Problem 2: Markov Decision Process

With the given 3 states s_1, s_2 and s_3, and actions a and b, our probability functions are defined as follows:

$$P(s_2|s_1, a) = 0.7$$
$$P(s_1|s_1, a) = 0.3$$
$$P(s_1|s_2, a) = 0.7$$
$$P(s_2|s_2, a) = 0.3$$
$$P(s_3|s_1, b) = 0.2$$
$$P(s_1|s_1, b) = 0.8$$
$$P(s_3|s_2, b) = 0.2$$
$$P(s_2|s_2, b) = 0.8$$

Additionally, the rewards for the 3 states are as follows:

$$R(s_1) = -1$$
$$R(s_2) = -2$$
$$R(s_3) = \quad 0$$

a State Transition Diagram

The state transition diagram is represented in Figure 5 below.

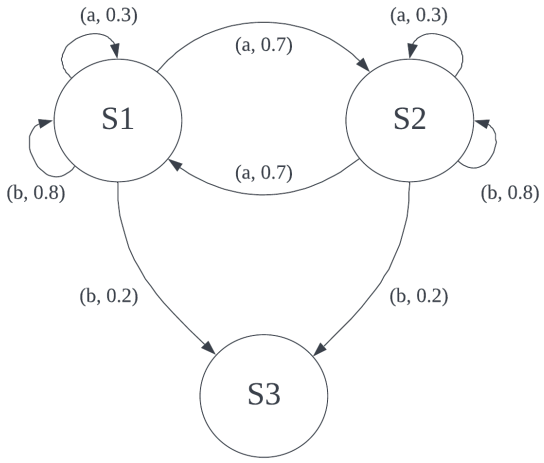

Figure 5: State Transition Diagram

b Qualitative Optimal Policy

- Considering the rewards in each state, we can see that $R(s_3) > R(s_1) > R(s2)$
- Overall, the agent should try to get to s_3 as quick as possible since it has the highest reward.
- Since the probability for getting to s_3 is low with action b, we should try to minimize the cost (maximize the rewards) in steps taken to get to s_3.

- Thus in state 1, the optimal policy should be the one which tries to get to s_1 as soon as possible. In this case, this would mean the optimal policy in s_1 would always be to pick action b, since that tries to move to s_3
- In state s_2, it might be a better option to try to move to s_1 first with action a (since it has a lower probability of staying in s_2).

c Policy Iteration with policy b

Defining out initial values:

$$\pi_0 = S_1 : b, S_2 : b$$
$$U(s_3) = 0(Since\, it\, is\, goal\, state)$$

We use the following formulae:

$$PolicyEvaluation : U_1(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U_i(s')$$
$$PolicyImprovement : \pi^*(s) = \arg\max_{a \in A(s)} R(s) + \gamma \sum_{s'} P(s'|s, a) U_i(s')$$

Iteration 1:

$$PolicyEvaluation :$$
$$U_1(s_1) = -1 + 0.2U_1(s_3) + 0.8U_1(s_1)$$
$$U_1(s_2) = -2 + 0.2U_1(s_3) + 0.8U_1(s_2)$$
$$U_1(s_3) = 0$$
$$Therefore :$$
$$U(s_1) = -5$$
$$U(s_2) = -10$$
$$PolicyImprovement :$$
$$\sum_{s}^{'} P(s'|s_1, a) U_1(s') = 0.7 * -10 + 0.3 * -5 = -8.5$$
$$\sum_{s}^{'} P(s'|s_1, b) U_1(s') = 0.2 * 0 + 0.8 * -5 = -4$$
$$\pi_1(s_1) = b \quad (Since\ -4 > -8.5)$$
$$\sum_{s}^{'} P(s'|s_2, a) U_1(s') = 0.7 * -5 + 0.3 * -10 = -6.5$$
$$\sum_{s}^{'} P(s'|s_2, b) U_1(s') = 0.2 * 0 + 0.8 * -10 = -8$$
$$\pi_1(s_2) = a \quad (Since\ -6.5 > -8)$$

Our policy is changed, thus we continue with the next iteration.

Iteration 2:

$$
\begin{aligned}
&PolicyEvaluation: \\
&\qquad U_2(s_1) = -1 + 0.2U_2(s_3) + 0.8U_2(s_1) \\
&\qquad U_2(s_2) = -2 + 0.7U_2(s_1) + 0.3U_2(s_2) \\
&\qquad U_2(s_3) = 0 \\
&\qquad Therefore: \\
&\qquad U(s_1) = -5 \\
&\qquad U(s_2) = -7.86 \\
&PolicyImprovement: \\
&\sum_s^{'} P(s'|s_1, a)U_2(s') = 0.7 * -7.86 + 0.3 * -5 = -7 \\
&\sum_s^{'} P(s'|s_1, b)U_2(s') = 0.2 * 0 + 0.8 * -5 = -4 \\
&\qquad \pi_2(s_1) = b \quad (Since\ -4 > -7) \\
&\sum_s^{'} P(s'|s_2, a)U_2(s') = 0.7 * -5 + 0.3 * -7.86 = -5.86 \\
&\sum_s^{'} P(s'|s_2, b)U_2(s') = 0.2 * 0 + 0.8 * -7.86 = -6.3 \\
&\qquad \pi_2(s_2) = a \quad (Since\ -5.86 > -6.3)
\end{aligned}
$$

Since policy has not changed further, we terminate policy iteration here. Finally policy:

$\pi^* = S_1 : b, S_2 : a$

We can see that this agrees with our qualitative findings for the optimal policy as well.

d Policy Iteration with policy a

Defining out initial values:

$$
\begin{aligned}
\pi_0 &= S_1 : a, S_2 : a \\
U(s_3) &= 0(Since\ it\ is\ goal\ state)
\end{aligned}
$$

Iteration 1:

$$
\begin{aligned}
&PolicyEvaluation: \\
&\qquad U_1(s_1) = -1 + 0.3U_1(s_1) + 0.7U_1(s_2) \\
&\qquad U_1(s_2) = -2 + 0.7U_1(s_1) + 0.3U_1(s_2) \\
&\qquad U_1(s_3) = 0 \\
&Building\ \ equations: \\
&\qquad 1 = -0.7U_1(s_1) + 0.7U_1(s_2) \\
&\qquad 2 = 0.7U_1(s_1) - 0.7U_1(s_2)
\end{aligned}
$$

It appears that the equations are inconsistent and thus there would be no solution.

However, adding a discount factor might lead to having a solution by limiting the reward penalty.

- Adding a discount factor, as well as its choice can lead to a change in the policy that is selected by our agent.
- A small γ value indicates lesser importance to utilities in the distant future (more steps) for the calculation.
- This could potentially lead to the optimal policy b be selected from state s_2 since it would always try to reach s_3 as soon as possible ignoring the high chances of repeatedly staying in s_2 thus accumulating a bad reward.