Aasish Tammana

CSE 571 – Artificial Intelligence

Homework 5

**Exercise 1.1**

1. The bellman equations for utilities is given by

V(s) = R(s) + γ maxa∑s’ T(s,a,s’) V(s')

Here, V is the utility value

s represents states

a represents actions

T is the Transition function which is the probability of the action a leading to the state s

R is Reward Function

γ is the discounting factor

As per the value iteration algorithm, To solve MDP’s, if there are n states, each state will have one bellman equation which are given by

Vi+1(s) = R(s) + γ maxa∑s’ T(s,a,s’) Vi(s')

In case of R(s,a), these equations can be written as

V(s) = maxa [R(s,a) + γ ∑s’ T(s,a,s’) V(s') ]

Vi+1(s) = maxa [R(s,a) + γ ∑s’ T(s,a,s’) Vi(s')]

By the recursive definition, we can say that,

V\*(s) = maxaQ\*(s,a)

Q\*(s,a)= ∑s’ T(s,a,s’)[ R(s,a,s’) + γ V\*(s’)]

Therefore, V\*(s) =maxa∑s’ T(s,a,s’)[ R(s,a,s’) + γ V\*(s’)]

This equation characterises the optimal values and

V\*(s) is expected utility at state s when optimal

Q\*(s,a) is expected utility at state s with action a when optimal

The equations for each state can be represented as,

Vk+1(s) =maxa∑s’ T(s,a,s’)[ R(s,a,s’) + γ Vk(s’)]

1. From the bellman’s equation for optimal policy, we know that

V(s) =maxa∑s’ T(s,a,s’)[ R(s,a,s’) + γ V(s’)]

Therefore, V(s) =maxa [∑s’ T(s,a,s’)R(s,a,s’) + γ∑s’ T(s,a,s’) V(s’)]

However, we also know that ∑s’ T(s,a,s’)R(s,a,s’)=R(s,a)

Therefore, this transforms to V(s)=maxa [R(s,a)+ γ∑s’ T(s,a,s’) V(s’)]

Alternatively, we can assume a tertiary state given by z where we can say that taking an action a1 lead to z instead of s’

Now we can give the new MDP with a probability of T',

We know that this probability T’ will be the same as T for any given action i.e T’=T

Consequently, we can call another action a2 which leads the agent from any state to s’ with 100 percent guarantee. In this scenario, the T value will be equal to 1.

T’(x, a2, s’) = 1 where x represents any state

R’(z, a2) = γ’R (s, a1, s’)

After substituting T’, R’ in V(s)=maxa[R(s,a,s’) + γ∑s’ T(s,a,s’) V(s’)]

V\*(s) = 𝑚𝑎𝑥𝑎 [R’ (z,a2) + γ’[∑s’ T’(s,a1,s′)V\*(s)] which is the same as derived above.

For this as well, we can use the similar approach used in part b.

We know that V(s) = R(s) + γ maxa∑s’ T(s,a,s’) V(s')

We can assume a tertiary state given by z where we can say that taking an action a1 lead to z instead of s’

Now we can give the new MDP with a probability of T',

We know that this probability T’ will be the same as T for any given action i.e T’=T

Consequently, we can call another action a2 which leads the agent from any state to s’ with 100 percent guarantee. In this scenario, the T value will be equal to 1.

T’(x, a2, s’) = 1 where x represents any state

R’(z) = γ’R (s, a1, s’)

We have V(s)=R’(s) + γ’maxa1[∑zT’(s,a1,z){ R’(z) + γ’maxa2 T’(z,a2,s’)V’(s)}]

We know that R’(z) is 0 because it is a tertiary state.

By replacing above conditions, the equation transforms to

V(s)= R’(s)+γ’maxa1[∑zT’(s,a1,z){ γ’maxa2V’(s)}]

Therefore, V(s)= R’(s)+γnewmaxa[∑sT’(s,a1,z) V’(s)]

Which is of the form of R(s)

**Exercise 1.2**

Discount factor is mentioned as 0.99

|  |  |  |
| --- | --- | --- |
| **r** | **-1** | **+10** |
| **-1** | **-1** | **-1** |
| **-1** | **-1** | **-1** |

1. Given r=100

|  |  |  |
| --- | --- | --- |
| **r** | **Left** | **+10** |
| **Up** | **Left** | **Down** |
| **Up** | **Left** | **Left** |

Since the reward is a very high positive value, the agent will move towards the reward.

1. Given r=-3

|  |  |  |
| --- | --- | --- |
| **r** | **Right** | **+10** |
| **Right** | **Right** | **Up** |
| **Right** | **Right** | **Up** |

Since the reward is negative and smaller than living cost, the agent will actively try to avoid the square with the reward and move towards the goal step by step. [Note: There is always a possibility that the agent might end up being in the reward square because of the 80% probability]

1. Given r=0

|  |  |  |
| --- | --- | --- |
| **r** | **Right** | **+10** |
| **Up** | **Up** | **Up** |
| **Up** | **Up** | **Up** |

Since the living reward is negative, the agent tries to immediately move upwards and then progress towards the goal. (If possible/necessary agent will go through the path where reward is 0 because it is better than the living cost)

1. Given r=3

|  |  |  |
| --- | --- | --- |
| **r** | **Left** | **+10** |
| **Up** | **Left** | **Down** |
| **Up** | **Left** | **Left** |

Here, since the reward is positive, the agent will move towards the reward and then proceed to move towards the goal later because it is better than the living cost.

**Exercise 1.3**

As mentioned, the agent has only two possible actions in the start state which are up and down.

V(s) = maxa ∑s’ T(s,a,s’) [R(s,a) + γ V(s')]

Vup = 50γ +

Vup = 50γ – [γ2+ γ3 +γ4 +….+ γ101]

Vup = 50γ – γ2 [1+γ+γ2+….+ γ99]

The expression is an Geometric progression

i.e Summation of a, ar, ar2, ar3, ... is given by a(rn - 1) / (r - 1)

Here n Is 100 and a is 1

Vup = 50γ – γ2

Similarly,

Vdown = -50γ +

Vdown = -50γ + [γ2+ γ3 +γ4 +….+ γ101]

Vdown = -50γ + γ2 [1+γ+γ2+….+ γ99]

Vdown = -50γ + γ2

50γ – γ2  = 0

Since γ is not 0, we have

Solving the equation with calculator, we obtain

This implies if the γ value is larger than this, the agent should prefer going down (for gains in the longer run) and when the γ value is smaller than this, the agent should prefer going up(for immediate gain).

Alternatively, for calculation purpose let us ignore all the higher order terms.

Therefore, using Vi+1(s) = maxa ∑s’ T(s,a,s’) [R(s,a) + γ Vi(s')]

For going up we can say

V0=max(50+ γ (0)) =50

V1=max(-1+ γ (50)) =50 γ – 1

For going down we can say

V0=max(-50+ γ (0)) =-50

V1=max(1+ γ (-50)) =-50γ+1

Ignoring the higher order terms,

50-50γ =1– 50γ-50

100 γ=99

Therefore γ=0.99 (approximate value)

**Exercise 1.4**

Given 0(cool) = Slow and 0(warm) = Slow

We do not have policy for the overheating state and can ignore it.

Equation 1

V(cool) = 1 + 0. 5V(cool)

Therefore V(cool) - 0.5 V(cool)=1

0.5V(cool)=1

V1(cool)=2

Equation 2

V(warm) =0. 5[1 + 0. 5V(cool)] + 0.5[1 + 0. 5V(warm)]

V(warm)=0.5[1+0.5\*2]+0.5+0.25V(warm)

V(warm)=1.5+0.25V(warm)

0.75 V(warm)=1.5

V1 (warm)=2

Equation 3 - V(overheated) = 0

1(cool)=

maximum{

Slow:1[1+0.5 V1 (cool)],

Fast: 0.5[2+0.5 V1 (cool)]+ 0.5[2+0.5 V1 (warm)]

}

1(cool)=

maximum{

Slow:1[1+0.5\*2],

Fast: 0.5[2+0.5\*2]+ 0.5[2+0.5 \*2]

}

1(cool)=maximum{Slow:2,Fast:3}

Therefore 1(cool)=**Fast**

1(warm)=

maximum{

Slow: 0.5[1+0.5 V1 (cool)]+ 0.5[1+0.5 V1 (warm)],

Fast: 1[-10+0.5 V1 (overheated)]]

}

1(warm)=

maximum{

Slow: 0.5[1+0.5\*2]+ 0.5[1+0.5\*2],

Fast: 1[-10+0.5\*0]

}

1(warm)=maximum{Slow:2,Fast:-10}

Therefore 1(warm)=**Slow**

Equation 3

V(cool) = 1 + 0. 5V1(cool)

V2(cool)=2

Equation 4

V(warm) =0. 5[1 + 0. 5V1 (cool)] + 0.5[1 + 0. 5V1(warm)]

V(warm)=0.5[1+0.5\*2]+0.5+0.25\*2

V(warm)=1.5+0.5

V2 (warm)=2

2(cool)=

maximum{

Slow:1[1+0.5 V2 (cool)],

Fast: 0.5[2+0.5 V2 (cool)]+ 0.5[2+0.5 V2 (warm)]

}

2(cool)=

maximum{

Slow:1[1+0.5\*2],

Fast: 0.5[2+0.5\*2]+ 0.5[2+0.5 \*2]

}

2(cool)=maximum{Slow:2,Fast:3}

Therefore 2(cool)=**Fast**

2(warm)=

maximum{

Slow: 0.5[1+0.5 V2 (cool)]+ 0.5[1+0.5 V2 (warm)],

Fast: 1[-10+0.5 V2 (overheated)]]

}

2(warm)=

maximum{

Slow: 0.5[1+0.5\*2]+ 0.5[1+0.5\*2],

Fast: 1[-10+0.5\*0]

}

2(warm)=maximum{Slow:2,Fast:-10}

Therefore 2(warm)=**Slow**

Therefore, we can say,

|  |  |  |
| --- | --- | --- |
|  | cool | warm |
|  | Slow | Slow |
|  | Fast | Slow |
|  | Fast | Slow |

As seen above, the policy iteration for 2 is the same as 1

This implies the policy has converged.

**Exercise 1.5**

a)

We can see that cool, slow occurs in 3 instances, it is followed by cool in all three instances which means

**T(cool, slow, cool)=1**

Cool, fast occurs in 6 instances and is followed by cool in 3 instances and warm in 3 instances

**T(cool, fast, cool)=3/6=0.5**

**T(cool, fast, warm)=3/6=0.5**

Warm, fast occurs in 2 instances, it is followed by overheated in both instances which means

**T(warm, fast, overheated)=1**

Warm, slow occurs in 1 instances, it is followed by cool which means

**T(warm, slow, cool)=1**

This can be summarised as

T(cool, slow, cool)=1

T(cool, fast, cool)=0.5

T(cool, fast, warm)=0.5

T(warm, fast, overheated)=1

T(warm, slow, cool)=1

The values of R are given as

R(cool, slow, cool)=+1

R(cool, fast, cool)=+2

R(cool, fast, warm)=+2

R(warm, fast, overheated)=-10

R(warm, slow, cool)=+1

b)

Occurrences of the actions are as follows

(cool,slow) – 3 times

(cool,fast) – 6 times

(warm,slow) – 1 time

(warm, fast) – 2 times

The Q functions are calculated by adding the total scores till end of episode for each function for all the occurrences , divided by the total number of occurrences. This is given by

Q(cool,slow)=(-2-3-5)/3 = -10/3

Q(cool,fast)=(-4-6-8-2-6-8)/6=-17/3

Q(warm,slow)=-4/1=-4

Q(warm, fast)=(-10-10)/2=-10

c)

V(cool)= 0

V(warm)= 0

V(overheated) = 0

Episode 1

V(cool)= 0.5\*0+0.5[1+0]=0.5

V(cool)=0.5\*0.5+0.5[1+0.5]=1

V(cool)= 0.5\*1+0.5[2+0.5]=1.75

V(cool)= 0.5\*1.75+0.5[2+0.75]=2.25

V(cool)= 0.5\*2.25+0.5[2+0.5]=2.375

V(warm)= 0.5\*0+0.5[-10+0]=-5

Episode 2

V(cool)=0.5\*2.375+0.5[2-7.375]=-1.5

V(warm)=0.5\*(-5)+0.5[1+3.5]=-0.25

V(cool)=0.5\*(-1.5)+0.5[1+1.25]=0.375

V(cool)=0.5\*(0.375)+0.5[2+0.625]=1.5

V(cool)=0.5\*1.5+0.5[2+1.125]=2.3125

V(warm)=0.5\*(-0.25)+0.5[-10+0.8125]=-4.71

Therefore we can say

V(cool)= 2.313

V(warm) = -4.71

V(overheated)=0

d)

Q(cool,slow)=0

Q(cool,fast)=0

Q(warm,slow)=0

Q(warm, fast)=0

Episode 1

Q(cool,slow)= 0.5\*0+0.5[1+max(0,0)]=0.5

Q(cool,slow)= 0.5\*0.5+0.5[1+max(0.5,0)]=1

Q(cool,fast)= 0.5\*0+0.5[2+ max(1,0)]=1.5

Q(cool,fast)= 0.5\*1.5+0.5[2+ max(1.5,1)]=2.5

Q(cool,fast)=0.5\*2.5+0.5[2+max(0,0)]=2.25

Q(warm, fast)= 0.5\*0+0.5[-10+max(0,0)]=-5

Episode 2

Q(cool,fast)= 0.5\*2.25+0.5[2+max(0,0)]=2.25

Q(warm,slow)= 0.5\*0+0.5[1+max(0,-5)]=0.5

Q(cool,slow)=0.5\*1+0.5[1+max(1,2.25)]=2.125

Q(cool,fast)=0.5\*2.25+0.5[2+max(2.25,2.125)]=3.25

Q(cool,fast)=0.5\*3.25+0.5[1+max(-5,0.5)]=2.375

Q(warm, fast)= 0.5\*-5+0.5[-10+max(0,0)]=-7.5

Therefore we can say

Q(cool,slow)=2.125

Q(cool,fast)=2.375

Q(warm,slow)=0.5

Q(warm, fast)=-7.5