Roxanne Farhad – HW4 - AI

1a)

a_4 = argmax(-1 + $\sqrt{(In(4)}/1)$, 0 + $\sqrt{(In(4)}/1)$, 1 + $\sqrt{(In(4)}/1)$)
a_4 = C

Q_4(C) = 1 + 0.5(2 – 1) = 1.5

a_5 = argmax(-1 + $\sqrt{(In(5)}/1)$, 0 + $\sqrt{(In(5)}/1)$, 1.5 + $\sqrt{(In(5)}/2)$)
a_5 = C

Q_5(C) = 1.5 + 0.5(2 – 1.5) = 1.75

a_6 = argmax(-1 + $\sqrt{(In(6)}/1)$, 0 + $\sqrt{(In(6)}/1)$, -1 + $\sqrt{(In(6)}/3)$)
a_6 = C

Q_6(C) = 1.75 + 0.5(2 – 1.75) = 1.875

1b)

a_4 = argmax(-1 + 5$\sqrt{(In(4)}/1)$, 0 + 5$\sqrt{(In(4)}/1)$, 1 + 5$\sqrt{(In(4)}/1)$)
a_4 = C

Q_4(C) = 1 + 0.5(2 – 1) = 1.5

a_5 = argmax(-1 + 5$\sqrt{(In(5)}/1)$, 0 + 5$\sqrt{(In(5)}/1)$, 1.5 + 5$\sqrt{(In(5)}/2)$)
a_5 = B

Q_5(B) = 0 + 0.5(0 – 0) = 0

a_6 = argmax(-1 + 5$\sqrt{(In(6)}/1)$, 0 + 5$\sqrt{(In(6)}/2)$, 1 + 5$\sqrt{(In(6)}/2)$)
a_6 = C

Q_6(C) = 1.5 + 0.5(2 – 1.5) = 1.75

When the value of C = 5, rather than 1, it means that the upper bound confidence function places a less weight of the current value of the state, and places more weight on the number of times that the action has been called. This means that the larger the value of C, the agent places greater priority to exploring.

1c)

The next action that is not the value of C is chosen when this equation is satisfied:

$$0 + \sqrt{\frac{In(t)}{10}} > 2 + \sqrt{\frac{\ln(t)}{10 + t}}$$

Therefore, the value of t would be:

$$t > 2.35385 \times 10^{17}$$

1d)

This means that the smaller the value of C, there is more exploitation (choosing the best known action) than exploration. On a real system, exploring never does stop because there will be an eventual different action like in part c.

2a)

P(v):

v	**Pr(v)**
+v	0.5
-v	0.5

P(P):

p	**Pr(p)**
+p	0.4
-p	0.6

Pr(v, p)

v	**p**	**Pr(v, p)**
+v	+p	0.2
+v	-p	0.3
-v	+p	0.2
-v	-p	0.3

If v, p are independent variables then Pr(v, p) = P(v) * p(p) which holds for all values.
But, Pr(v | p) = pr(v) must also hold
Pr(+v | +p) = 0.2/0.4 = 0.5
Pr(-v | +p) = 0.2/0.4 = 0.5
Pr(+v | -p) = 0.3/0.6 = 0.5
Pr(-v | -p) = 0.3/0.6 = 0.5
Therefore, Pr(v, p) are independent variables.

2b)

Pr(a | v) = Pr(a, v)/Pr(v)

P(a, v):

a	**v**	**Pr(a, v)**
+a	+v	0.3
-a	+v	0.2
+a	-v	0.15
+a	-v	0.35

Pr(a | v):

v	**a**	**Pr(a | v)**
+v	+a	0.6
+v	-a	0.4
-v	+a	0.3
-v	-a	0.7

A, P are conditionally independent given V if:
Pr(A, P | V) = Pr(A | V)Pr(P | V) and if Pr(A | V, P) = Pr(A | V)

p	**v**	**Pr(P | V)**
+p	+v	0.4
+p	-v	0.4
-p	+v	0.6
-p	-v	0.6

p	v	a	Pr(a, p | v)
+p	+v	+a	0.24
+p	-v	+a	0.12
-p	+v	-a	0.24
-p	-v	+a	0.18
+p	+v	-a	0.16
-p	-v	-a	0.42
+p	-v	-a	0.28
-p	+v	+a	0.36

0.24 = 0.6 * 0.4 ; 0.12 = 0.3 * 0.4 ; 0.24 = 0.4 * 0.6 ; 0.18 = 0.3 * 0.6 ; 0.16 = 0.4 * 0.4 ;
0.42 = 0.7 * 0.6 ; 0.28 = 0.7 * 0.4 ; 0.36 = 0.6 * 0.6

Therefore, A, P are conditionally independent given V

2c)

$$\Pr(A,R) = \sum_{p,v} \Pr(R,P,V,A) = \Pr(R) * \Pr(P \mid R) * \Pr(V \mid R,P) * \Pr(A \mid R,P,V) \quad \begin{matrix} A\ independent\ of\ R \mid (P,V) \\ ; \\ P\ indepedent\ of\ V \mid R \end{matrix}$$

$$\Pr(A,R) = \Pr(R) * P(P \mid R) * \Pr(V \mid R) * \Pr(V \mid R) * \Pr(A \mid V)$$

V	R	P(V | R)
+v	+r	0.8
-v	+r	0.2
+v	-r	0.3
-v	-r	0.7

P	R	P(VP | R)
+p	+r	0.1
-p	+r	0.9
+p	-r	0.6
-p	-r	0.4

P(+a, +r) = (0.4 * 0.1 * 0.8 * 0.6) + (0.4 * 0.9 * 0.8 * 0.6) + (0.4 * 0.1 * 0.2 * 0.6) + (0.4 * 0.9 * 0.2 * 0.6) = 0.24/0.4 = 0.6

P(+a, -r) = (0.3 * 0.6 * 0.6 * 0.3) + (0.3 * 0.4 * 0.6 * 0.3) + (0.7 * 0.6 * 0.6 * 0.3) + (0.7 * 0.4 * 0.6 * 0.7) = 0.18/0.6 = 0.3

P(-a, +r) = (0.4 * 0.1 * 0.8 * 0.4) + (0.4 * 0.9 * 0.8 * 0.4) + (0.4 * 0.1 * 0.2 * 0.4) + (0.4 * 0.9 * 0.2 * 0.4) = 0.16/0.4 = 0.4

P(-a, -r) = (0.3 * 0.6 * 0.6 * 0.7) + (0.3 * 0.4 * 0.6 * 0.7) + (0.7 * 0.6 * 0.6 * 0.7) + (0.7 * 0.4 * 0.6 * 0.7) = 0.42/0.6 = 0.7

Pr(a | r):

A	**R**	**Pr(A | R)**
+a	+r	0.6
+a	-r	0.3
-a	+r	0.4
-a	-r	0.7

2d)

Pr(+r | +a) = (Pr(+a | +r)*Pr(+r)) / p(+a)) = (0.6 * 0.4)/((0.6*0.4) + (0.3 * 0.6) = 0.24/0.42 = 0.57… = 0.57 (2.d.p) which represents the likelihood that it's raining given that an ad pops up.

Q3.4

Floor Plan 1

Value Iteration (Synchronous): 25
Value Iteration (Asynchronous): 15

Policy Iteration (Synchronous): 9
Policy Iteration (Asynchronous): 9

Floor Plan 2

Value Iteration (Synchronous): 47
Value Iteration (Asynchronous): 32

Policy Iteration (Synchronous): 9
Policy Iteration (Asynchronous): 9

3)

For policy iteration, the number of iterations for convergence goes from 9 to 7 for the first floor plan, but for floor plan 2 the number of iterations for convergences goes from 9 to 10 (when asynch = False)

4)

When the number of episodes is close to 1000 it learns a policy that is close to the same values for value-iteration and policy iteration.

5)

A high epsilon value and lernign rate, with a low discount factor helps the robot learn at a quicker rate.