Please feel free to write down your answers and discuss together

COMP90054: AI Planning for Autonomy
Revision exercises

Question 1 (20 Marks)

(a). The following search algorithms can be implemented in a similar manner, differing only in the *abstract data structure* used to implement the *open list*.

- Breadth First Search
- Depth First Search
- Uniform Cost Search

(i). State which *abstract data structure* you would use for *Breadth First Search*. Briefly explain, in one or two sentences, why you would choose the particular data structure.

(ii). State which *abstract data structure* you would use for *Depth First Search*. Briefly explain, in one or two sentences, why you would choose the particular data structure.

(iii). State which *abstract data structure* you would use for *Uniform Cost Search*. Briefly explain, in one or two sentences, why you would choose the particular data structure.

(b). Recall that A* requires a consistent heuristic to guarantee finding an optimal plan without re-opening nodes. Draw or define a graph and make up an admissible *but inconsistent* heuristic function where A* returns a *suboptimal* solution.

Write down the **steps** involved in the A* algorithm by devising a simple example to illustrate. Note: avoid making large examples, a graph with 4 nodes should be sufficient.

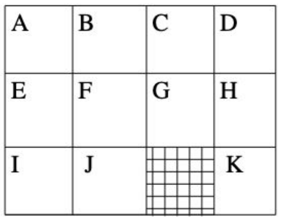
(a)
(i)BFS: QUEUE, FIFO (first in, first out)
(ii)DFS: STACK, LIFO (last in, first out)
(iii)PRIORITY QUEUE: Ordered by g (minimum cumulative cost, maximum priority)

(b) initial, goal, heuristics, costs
s1
s4
s1 12
s2 9

s3 5
s4 0
s1 s2 2
s1 s3 5
s2 s3 2
s3 s4 10
Note: suboptimal if not reopening, otherwise optimal

Question 2 (20 Marks)

(a). This question concerns a classical planning problem where a robot can move horizontally and vertically to adjacent cells as depicted in the figure below. Note that the robot *cannot* move diagonally between cells and the *hashed* cell is inaccessible to the robot.



In answering the sub-questions below, you are allowed to use variables as arguments for the actions (action schemes), specifying the values of the variables. Note: it is **not** compulsory to use PDDL syntax, as long as you can convey the main ideas.

Hint: consider that the position of the robot could be modelled as either

- row/column *tuples*, e.g. $\langle 0, 0 \rangle$ could refer to the lower left cell, or
- single cells, e.g. position $\langle I \rangle$ could refer to the lower left cell.

(i). Describe briefly in STRIPS how to model the problem where a robot can move horizontally and vertically among adjacent cells, such that the h_{add} heuristic estimates the same values as the *Manhattan* heuristic.

(ii). Describe briefly in STRIPS how to model the problem where a robot can move horizontally and vertically among adjacent cells, such that the h_{add} heuristic estimates the same values as the *Optimal* heuristic.

(iii). Using your last STRIPS encoding where $h_{add} = h^*$, an initial state $s_0 = robot\ at\ location\ I$ and a goal state $s_g = robot\ at\ location\ K$, compute $h_{add}(s_g)$ and h_{ff} from the best supporters induced by h_{add}.

(b). We have 2 rooms A and B, 2 objects o_1 and o_2. A robot can load and unload objects if the robot is at the same location, and move from one room to the other. We want to get o_1 and o_2 into room B, given that both are initially at A.

(i). Model the problem in STRIPS in such a way that the optimal plan would be the following:

$$pick(o_1, A), move(A, B), drop(o_1, B), move(B, A), pick(o_2, A), move(A, B), drop(o_2, B)$$

(ii). What's the $h_{max}(s_0)$ value of your STRIPS model, where s_0 stands for the initial state?

Q2

Head tutor explain:

1. In STRIPS model, you have to specify everything. For example, every fluents in the F, and every operators in the O.

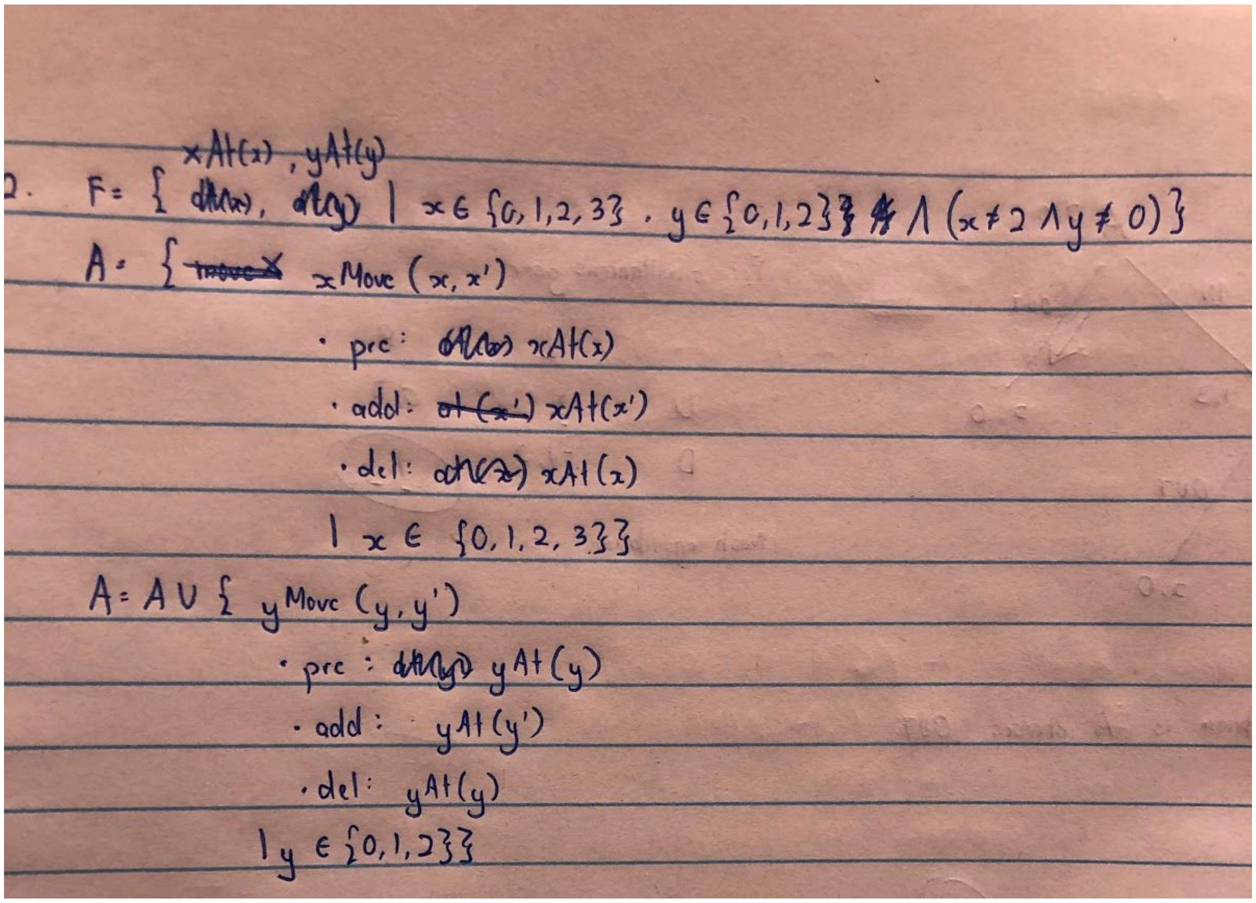
So, there are several ways to model the constraints of the inaccessible cell:

a. If you model the cell with characters, such "at(A),at(B)", then just don't assign character to the inaccessible cell. Then there is no way your O make the agent in inaccessible cell as it is not even in F

b. If you model the cell with a pair of x,y, such as "at(0,0)", then just don't include "at(2,0)" in F. Then, following the same idea, at(2,0) is not in F, so it is should not be in the O as well. which means operators such as "move(1,0,2,0)" should not be in O.

c. If you model the cell with separate coordinate indicators, such as x_at(0), y_at(0), x_at(1), then, there is no way you can remove "x_at(2)" and "y_at(0)" from F. What you need to make sure is "x_at(2)" and "y_at(0)" cannot be both true in the F at the same time. Then, the only way you can do that is to make sure they are not both true in the I (initial state), and there is no operator have both "x_at(2)" and "y_at(0)" in the "add list" (which means that do not include operators such as "move(1,0,2,0)").

(a)
(i)
TODO 改actions就行了?



xAt(x) , yAt(y)

2. $F = \{$ ~~dAt(x), dAt(y)~~ $\mid x \in \{0,1,2,3\} , y \in \{0,1,2\}\}$ ~~∄~~ $\wedge\ (x \neq 2 \wedge y \neq 0)\}$

$A = \{$ ~~move x~~ xMove (x, x')

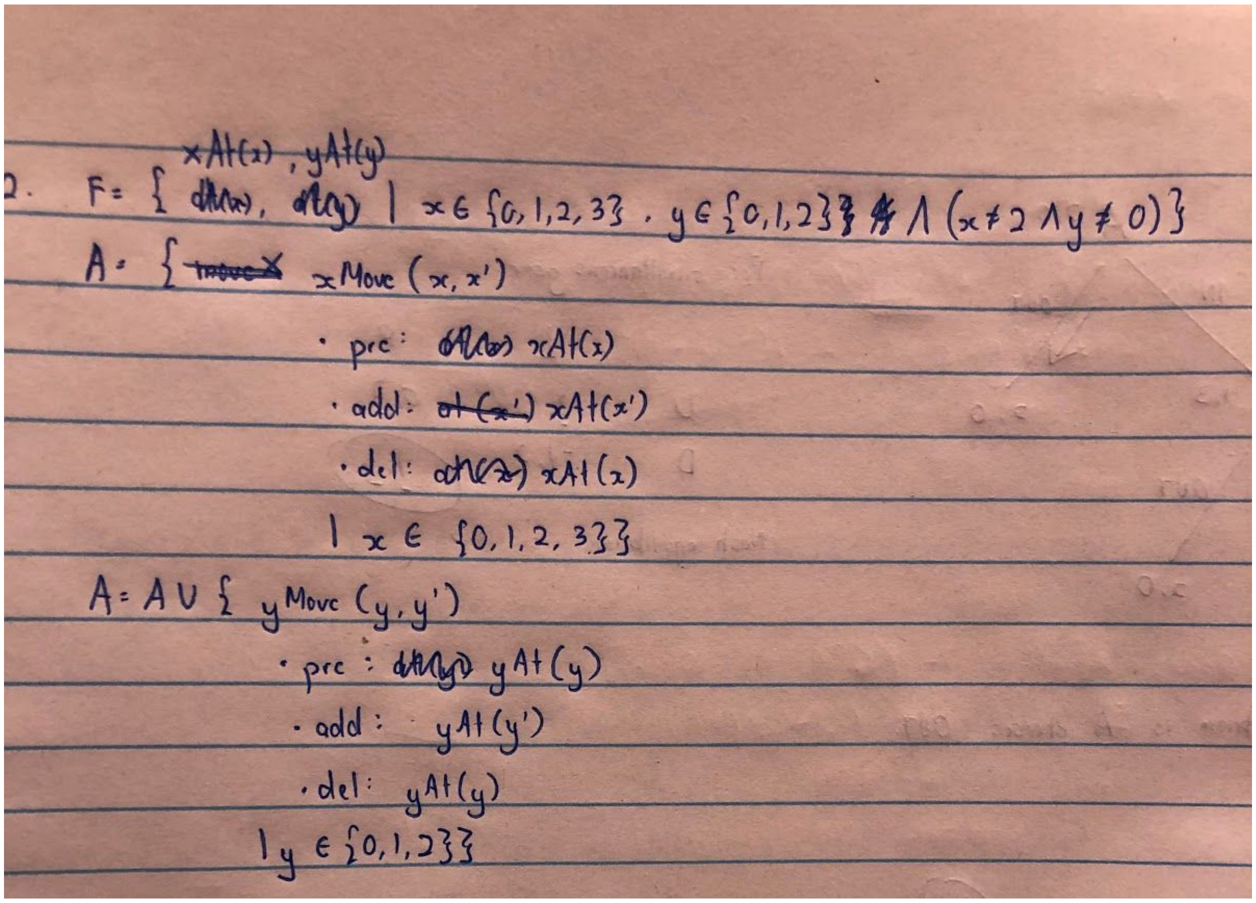
- pre: ~~dAt(x)~~ xAt(x)
- add: ~~at (x')~~ xAt(x')
- del: ~~dAt(x)~~ xAt(x)

$\mid x \in \{0, 1, 2, 3\}\}$

$A = A \cup \{$ yMove (y, y')

- pre: ~~dAt(y)~~ yAt(y)
- add: yAt(y')
- del: yAt(y)

$\mid y \in \{0,1,2\}\}$



2. F= { ~~at(x), at(y)~~ xAt(x), yAt(y) | $x \in \{0,1,2,3\}$, $y \in \{0,1,2\}\}$ ~~∧~~ $\wedge (x \neq 2 \wedge y \neq 0)\}$

A= { ~~move x~~ xMove(x, x')

- pre: ~~at(x)~~ xAt(x)
- add: ~~at(x')~~ xAt(x')
- del: ~~at(x)~~ xAt(x)

| $x \in \{0,1,2,3\}\}$

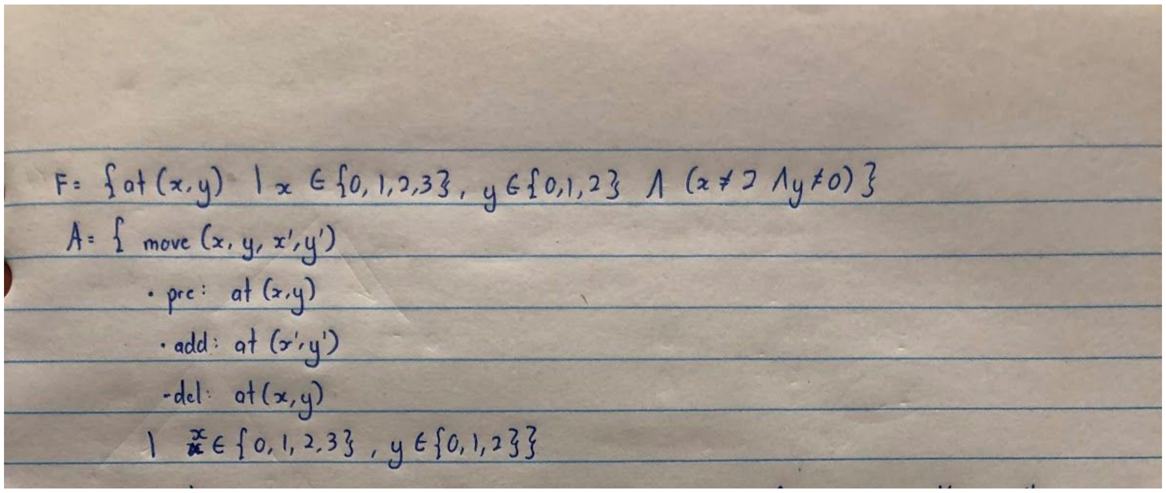
A= A ∪ { yMove(y, y')

- pre: ~~at(y)~~ yAt(y)
- add: yAt(y')
- del: yAt(y)

| $y \in \{0,1,2\}\}$

I suppose it is at a right direction, but I don’t think inaccessible block can be modeled in this way. Basically, if you expand F, it should be like: F={x(0),x(1),x(2),x(3),y(0),y(1),y(2)} so I don’t think you can exclude (2,0).

(ii)



F= {at(x,y) | $x \in \{0,1,2,3\}$, $y \in \{0,1,2\}$ $\wedge (x \neq 2 \wedge y \neq 0)\}$

A= { move(x, y, x', y')

- pre: at(x,y)
- add: at(x',y')
- del: at(x,y)

| $x \in \{0,1,2,3\}$, $y \in \{0,1,2\}\}$

It is strange to model inaccessible block in this way? I think it should be something like: +1
F={at(x,y) | x∈{0,1,2,3},y∈{0,1,2,}} - {at(2,0)}

What’s the difference between (i) and (ii)? The answers for (i) and (ii) look similar +1+1

(iii)

at(A)	B	C	D	E	F	G	H	I	J	K
∞	∞	∞	∞	∞	∞	∞	∞	0	∞	∞
∞	∞	∞	∞	1	∞	∞	∞	0	1	∞
2	∞	∞	∞	1	2	∞	∞	0	1	∞
2	3	∞	∞	1	2	3	∞	0	1	∞
2	3	4	∞	1	2	3	4	0	1	∞
2	3	4	5	1	2	3	4	0	1	5

$h^{add}(s_0)=h^{add}(s_0,G)=5$

如果题目真的问的是$h^{add}(S(g))$, 那么$h^{add}(s(g))=0$ **5？**

为啥？$s_g=K, h^{add}(K)=5$

$h^{add}(I)=5$? ppt上是这么表示的： $h^{max}(I) = 5.5 << 20 = h^*(I).$

Best supporter function:

	at(A)	B	C	D	E	F	G	H	I	J	K
move	E,A	A,B	B,C	C,D	I,E	E,F	F,G	G,H	-	I,J	H,K

$h^{ff}=5$

(b)

(i)

F={at(A),at(B),o1(A),o1(B),o1(Robot),o2(A),o2(B),o2(Robot)}

O={pick(object,location),move(location1,location2),drop(object,location)}

Where:

pick(object,location)

Pre: at(location),object(location)

Add: object(Robot)

Note: I don’t know if the syntax “object(location)” is correct. I mean o1(location) or o2(location). Please feel free to suggest any better expressions. Q: at(A) means robot at A? yes

drop(object, location)
Pre: at(location), object(R)
Add: object(location)

move(location1, location2)
Pre: at(location1)
Add: at(location2)

c=1

I={at(A),o1(A),o2(A)}
G={o1(B),o2(B)}

(ii)

at(A)	at(B)	o1(A)	o1(B)	o1(Robot)	o2(A)	o2(B)	o2(Robot)
0	∞	0	∞	∞	0	∞	∞
0	1	0	∞	1	0	∞	1
0	1	0	2	1	0	2	1

Hmax = max(2, 2) = 2

请问题目问的是不是s0的hmax？是的吧，我理解的s0的hmax就是s0到G的？hmax(s0,G)
$h^{max}(s_0)=h^{max}(s_0,G)=max(2,2)=2$

b).i). 不知道对不对。。麻烦大家提意见

F: { O1 at A, O1 at B,
O2 at A, O2 at B,
Robot (A), Robot (B)
Holding (x) | x ∈ (O1, O2), ArmFree }

O: A = A ∪ { move (AB):
· prec: Robot (A)
· Add: Robot (B)
· Del: Robot (A) }

A = A ∪ { pick (x, y)
· prec: Robot (y), x at y, ArmFree
· Add: Holding (x)
· Del: x at y, ArmFree
| x ∈ (O1, O2)
y ∈ (A, B) }

A = A ∪ { drop (x, y)
· prec: Robot (y), Holding (x)
· Add: x at y
· Del: Holding (x)
| x ∈ (O1, O2)
y ∈ (A, B) }

I: { Robot (A);
O1 at A;
O2 at A;
ArmFree }

G: { O1 at B, O2 at B }

ii).

O1(A)	O1 at B	O2 (A)	O2 (B)	Robot (A)	Robot (B)	Holding (O1)	Hold (O2)	Arm Free
0	∞	0	∞	0	∞	∞	∞	0
0	∞	0	∞	0	1	1	1	0
0	3	0	3	0	1	1	1	0

$G = \{ o1 \text{ at } B, o2 \text{ at } B \}$

$h^{max}(s_0, G) = max(3, 3) = 3$

不好意思不小心把旁边的问题点没了。能得到o1 at B的action是drop(o1, B),它的prec是Robot at B, Holding o1, 再加上这一步的cost是 (1+1+1 = 3)。应该是因为上面的case没有定义holding 这个动作我觉得

hmax(o1(B))=c(drop(o1,B))+hmax(at(B),o1(R))=1+1=2

Where hmax(at(B),o1(R)) = max(hmax(at(B)), hmax(o1(R))) = max(1,1)=1

Question 3 (10 Marks)

Imagine a kitchen robot whose task is to ask the user what kinds of cereals he/she wants for breakfast, wait for the user answer, and then hand out the appropriate cereal box once it knows the desired cereal. We want to design a simple dialogue system to handle the interaction with the user.

A simple way to model it is via a discount-reward MDP with only two states: $UnknownCereal$, where the robot does not know which cereal to give, and $KnownCereal$, where the robot knows the cereal to hand out.

There are only two possible actions in the model:

- Action $AskCerealType$ corresponds to the robot asking the user for the cereal box he/she wishes to have. The action is only available in state $UnknownCereal$, and has a reward of -1 (a cost) in that state.

  The probability of reaching state $KnownCereal$ is 0.8 (if the user answers the robot's question), and otherwise (probability 0.2) the robot remains in $UnknownCereal$ (if the user ignores the question or provides an unclear answer).

- Action $GiveCereal$ corresponds to the robot physically giving the cereal to the user. The action is available in the state $KnownCereal$, and has a reward of 5.

  When the robot executes this action in state $KnownCereal$, the MDP reaches an absorbing goal state and finishes. As such, there is no reward available from this state.

  When the robot executes this action in the $UnknownCereal$ state, the MDP reaches the absorbing goal state with probability 0.3 (it gets lucky and hands the right cereal) and receives a reward of 5, or the person rejects the cereal and the robot goes back to the $UnknownCereal$ state with probability 0.7 and receives a reward of -2.

Answer the following questions about this MDP:

(a). **[4 marks]** Assuming a discount factor $\gamma = 0.9$, calculate the the value function V for each of the states $UnknownCereal$ and $KnownCereal$ using value iteration, for the 2nd and 3rd iterations. Show your working.

Iteration		1	2	3
V(KnownCereal)	=	0.0		
V(UnknownCereal)	=	0.0		

(b). **[3 marks]** Given the value function that you calculate above, what policy would maximise the robot's expected reward? Show your working.

(c). **[3 marks]** In your own words, explain the difference between value iteration and policy iteration.

Q3(a):

$\gamma = 0.9$

$$\begin{tikzcd}
\text{known} \arrow[dr, "\text{give}, p=1, r=5"'] & & \text{unknown} \arrow[ll, "\text{succ}\ p=0.8,\ \text{ask}"', "r=-1"] \arrow[loop above, "\text{ask}, p=0.2, r=-1, \text{remain}"] \arrow[dl, "\text{succ}\ \text{give}, p=0.3, r=5"] \arrow[loop right, "\text{rejected}\ \text{give}, p=0.7, r=-2"] \\
& \text{goal} &
\end{tikzcd}$$

End

$V(goal)=0$

Iteration	1	2	3
V(known)	0.0	5	5
V(unknown)	0.0	0.1	2.62
V(goal)	0	0	0

$V_2(\text{known}) = \max[Q_2(\text{known}, \text{give})] = Q_2(\text{known}, \text{give})$

$Q_2(\text{known}, \text{give}) = P_{(k, \text{give})}[r + \gamma V_1(\text{goal})]$

$= 1 \times [5 + 0.9 \times 0] = 5$

$V_2(\text{unknown}) = \max[Q_2(\text{unknown}, \text{ask}), Q_2(\text{unknown}, \text{give})]$

Iter2

$Q_2(\text{unknown}, \text{ask}) = P_{(UK, \text{ask-succ})}[r_{(\text{success})} + \gamma V_1(\text{known})] + P_{(UK, \text{ask-remain})}[r_{(\text{remain})} + \gamma V_1(\text{unknown})]$

$= 0.8[(-1) + 0.9 \times 0] + 0.2[(-1) + 0.9 \times 0]$

$= -0.8 + 0.2(-1)$

$= -1$

$Q_2(\text{unknown}, \text{give}) = P_{(UK, \text{give-succ})}[r_{(\text{success})} + \gamma V_1(\text{goal})] + P_{(UK, \text{give-rejected})}[r_{(\text{rejected})} + \gamma V_1(\text{unknown})]$

$= 0.3[5 + 0.9 \times 0] + 0.7[(-2) + 0.9 \times 0]$

$= 1.5 + (-1.4)$

$= 0.1$

$V_2(\text{unknown}) = \max[-1, 0.1] = 0.1$

Apply the same logic for iteration 3.

$V_3(\text{known}) = Q_3(\text{known}, \text{give}) = P_{(k, \text{give})}[r + \gamma V_2(\text{goal})] = 1.0 \times [5 + 0.9 \times 0] = 5.0$

✓

Iter3

$Q_3(\text{unknown}, \text{give}) = P_{(UK, \text{give-succ})}[r_{(\text{succ})} + \gamma V_2(\text{goal})] + P_{(UK, \text{give-rejected})}[r_{(\text{rejected})} + \gamma V_2(\text{unknown})]$

$= 0.3[5 + 0.9 \times 0.] + 0.7[-2 + 0.9 \times 0.1)$

$= 1.5 + (-1.4) + (0.7) \times 0.9 \times 0.1 = 0.1 + \frac{63}{1000} = 0.1 + 0.063 = 0.163$

$Q_3(\text{unknown}, \text{ask}) = 0.8[-1 + 0.9 \times 5] + 0.2[-1 + 0.9 \times (0.1)]$

✓

$= 0.8 \times 3.5 + 0.2 \times (-1 + 0.09)$

$= 2.8 - 0.2 \times 0.9 = 2.8 - 0.18 = 2.62$

$V_3(\text{known}) = \max[0.163, 2.62] = \boxed{2.62}$

3

Policy Iteration finishes with an optimal policy π ∗ after a finite number of iterations, because the number of policies is finite, bounded by O(|A||S|), unlike value iteration, which can theoretically require infinite iterations.

Question 4 **(13 marks)**

Consider the same breakfast-making robot from the previous question. The robot designers have found that the probabilities used for outcomes are incorrect and different for each household. As such, they decide to instead use reinforcement learning to learn the policy after deployment.

A few weeks after deployment, one such robot has the following Q-table:

State	$AskCerealType$	$GiveCereal$
$UnknownCereal$	7.2	1.9
$KnownCereal$	—	3.4

Assuming a discount reward factor of 0.9 and a learning rate of 0.5, answer the following questions:

(a). **[6 marks]** In the state $UnknownCereal$, the robot executes $GiveCereal$, which is rejected. It decides to execute $GiveCereal$ again.

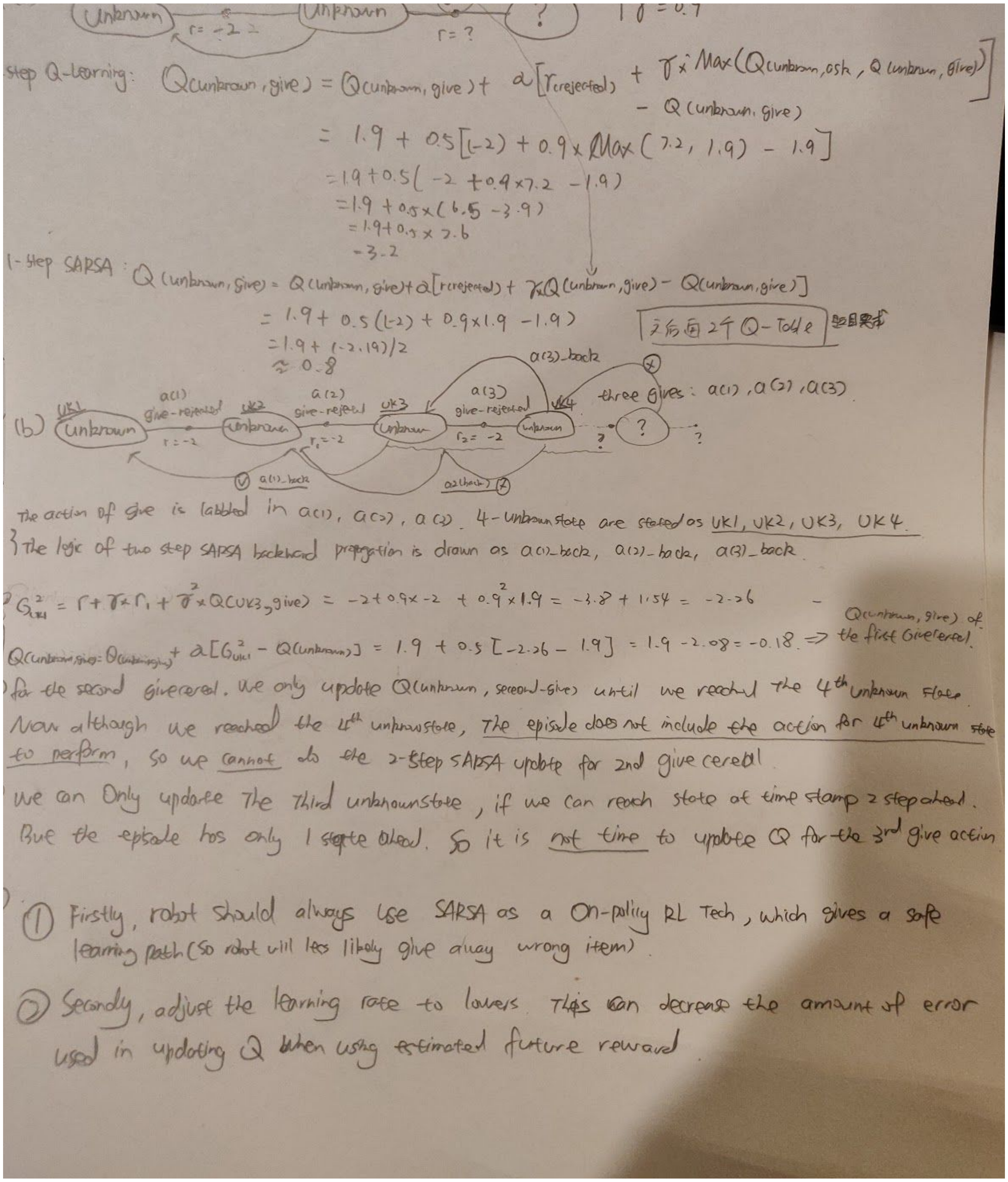
Update the Q-table both the 1-step Q-learning and 1-step SARSA updates for the first $GiveCereal$ action. That is, calculate two new Q-tables.

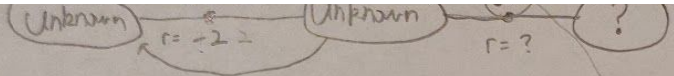
(b). **[4 marks]** *Challenge Question:* In the state $UnknownCereal$, the robot executes $GiveCereal$ three times in a row. Each time the cereal is rejected.

Calculate the 2-step SARSA update for this execution of three actions.

(c). **[3 marks]** The robot designers are finding that the owners of the robots are demanding a refund because the robots get their cereal preference wrong too often. Give one technique that the robot designers could do to improve the situation.

https://miro.medium.com/max/3132/1*-v5wbqLYCvzrOQE2Zmv05g.png





$\gamma = 0.9$

step Q-learning: $Q(\text{unknown}, \text{give}) = Q(\text{unknown}, \text{give}) + \alpha\left[r(\text{rejected}) + \gamma \times \text{Max}(Q(\text{unknown}, \text{ask}), Q(\text{unknown}, \text{give})) - Q(\text{unknown}, \text{give})\right]$

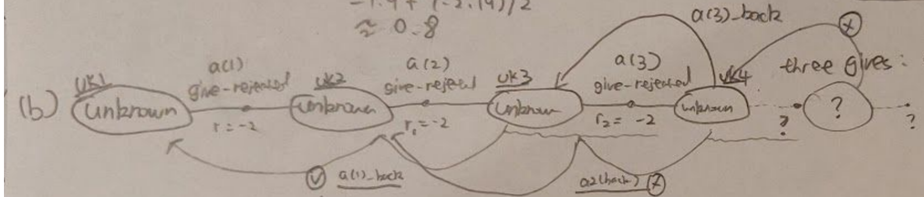
$$= 1.9 + 0.5[(-2) + 0.9 \times \text{Max}(7.2, 1.9) - 1.9]$$
$$= 1.9 + 0.5(-2 + 0.9 \times 7.2 - 1.9)$$
$$= 1.9 + 0.5 \times (6.5 - 3.9)$$
$$= 1.9 + 0.5 \times 2.6$$
$$= 3.2$$

1-step SARSA: $Q(\text{unknown}, \text{give}) = Q(\text{unknown}, \text{give}) + \alpha[r(\text{rejected}) + \gamma \times Q(\text{unknown}, \text{give}) - Q(\text{unknown}, \text{give})]$

$$= 1.9 + 0.5((-2) + 0.9 \times 1.9 - 1.9)$$
$$= 1.9 + (-2.19)/2$$
$$\approx 0.8$$

$\boxed{\text{之后画2个Q-Table}}$ 题目要求

(b)



three gives: a(1), a(2), a(3).

The action of give is labbled in a(1), a(2), a(3). 4-unknown state are stated as UK1, UK2, UK3, UK4.

} The logic of two step SARSA backward propagation is drawn as a(1)_back, a(2)_back, a(3)_back.

$G^2_{UK1} = r + \gamma \times r_1 + \gamma^2 \times Q(UK3, \text{give}) = -2 + 0.9 \times -2 + 0.9^2 \times 1.9 = -3.8 + 1.54 = -2.26$

$Q(\text{unknown}, \text{give}) = Q(\text{unknown}, \text{give}) + \alpha[G^2_{UK1} - Q(\text{unknown})] = 1.9 + 0.5[-2.26 - 1.9] = 1.9 - 2.08 = -0.18 \Rightarrow$ Q(unknown, give) of the first Giverefed.

for the second giverered. We only update Q(unknown, second-give) until we reached the 4th unknown state. Now although we reached the 4th unknown state, The episode does not include the action for 4th unknown state to perform, so we cannot do the 2-step SARSA update for 2nd give cereall

we can Only update The Third unknownstate, if we can reach state at time stamp 2 step ahead. But the episode has only 1 state ahead. So it is not time to update Q for the 3rd give action

① Firstly, robot should always use SARSA as a On-policy RL Tech, which gives a safe learning path (So robot will less likely give away wrong item)

② Secondly, adjust the learning rate to lowers. This can decrease the amount of error used in updating Q when using estimated future reward.

(S0=UK)>(A0=Give,R1=-2)>(S1=UK)>(A1=Give,R2=-2)>(S2=UK)>(A2=Give,...)>...>(A3 =unknown)

$G=\gamma^0 R1+\gamma^1 R2$

$Q(S0,A0) \mathrel{+}= a(R1+0.9R2+0.9^2 Q(S2,A2)-Q(S0,A0))$

Why don't we do two more steps without using SARSA to estimate future rewards?

$Q(S1,A1) \mathrel{+}= a(R2 + 0.9R3 - Q(S1,A1))$

$Q(S2,A2) \mathrel{+}= a(R3 - Q(S2,A2))$

The question asks you to use 2-step SARSA …

And in 2-step SARSA, Q(S1,A1) should be calculated like:

$Q(S1,A1) \mathrel{+}= a(R2 + 0.9R3 + 0.9^2 Q(S3,A3) - Q(S1,A1))$

$Q(S2,A2) \mathrel{+}= a(R3 + 0.9R4 + 0.9^2 Q(S4,A4) - Q(S2,A2)$

We have to use the rewards from the following 1st and 2nd step, you could not omit it and assume it to be 0, which is restricted by the algorithm. Please refers to the algorithm at the post handout 13.

Question 5 (5 Marks)

Imagine a reinforcement learning algorithm that monitors heart beat information from a fitness device, such as a FitBit, to determine whether a person develops a heart problem, such as an irregular heartbeat or a faster heartbeat.

List one potential ethical dilemma that could occur in such a situation. Justify why you believe this could be a serious problem.

Q5(ethic) is non-examinable

Question 6 (12 Marks)

(a). **[6 marks]** Consider the following abstract two-player game in normal form. Find all pure and mixed-strategy equilibria for this game.

Player 2

		L	M	R
Player 1	U	2,4	3,0	1,1
	D	3,2	10,3	0,4

HINT: Consider the notion of *dominated strategies*, in which some strategies are strictly dominated by others, so can be discarded.

(b). **[6 marks]** *Challenge Question:* Two players, A and B play the following game. First A must choose IN or OUT. If A chooses OUT the game ends, and the payoffs: are A gets 2, and B gets 0. If A chooses IN then B observes this and must then choose IN or OUT. If B chooses OUT the game ends, and the payoffs are: B gets 2, and A gets 0. If A chooses IN and B chooses in then they play the following simultaneous move game:

Player B

		L	R
Player A	U	3,1	0,2
	D	-1,2	1,3

Draw the extended-form tree for this game and calculate the equilibria of the extended-form game.

a). No pure Nash Equilibrium;

M is dominated strategy, so discarded

$$\begin{array}{cc|c|c|} & & & B \\ & & y & 1-y \\ & & L & R \\ \hline A & x \quad U & 2,4 & 1,1 \\ & 1-x \quad D & 3,2 & 0,4 \end{array}$$

$E(U) = 2y + (1-y) = y+1$

$E(D) = 3y + 0 = 3y$

$y+1 = 3y$

$1 = 2y$

$y = 1/2 \qquad 1-y = 1/2$.

$E(L) = 4x + 2(1-x) = 4x + 2 - 2x = 2x+2$

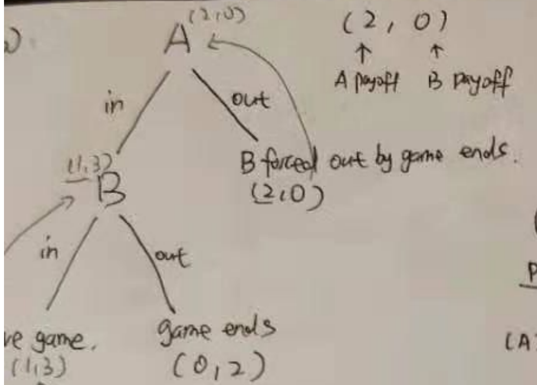
$E(R) = x + 4(1-x) = x + 4 - 4x = 4-3x$

$2x+2 = 4-3x$

$5x = 2$

$x = 2/5 \qquad 1-x = 3/5$

Mixed Nash Eqb: ((player A 2/5 U, 3/5 D), (player 2 1/2 L, 1/2 R))



$\boxed{\text{step}}$ Start from lowest subgame → move game

① As it is "simultaneous move game", so order/time of moves is not required ⇒ move game is Normal form - game.

②

		P2	
		L	R
P1	U	3,1	0,2
	D	-1,2	1,3

— Pure Nash equi

(A) for P1 { P2 - L ⇒ P1 → U ; P2 - R ⇒ P1 → D } No Dominome Strategy for P1

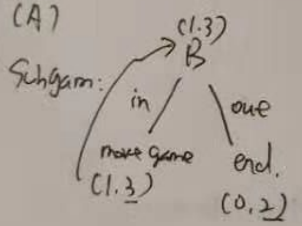
for P2 { P1 - U ⇒ P2 → R ; P1 - D ⇒ P2 → R } P2 has Dominant Strategy R. Eliminate P2-L (it is dominated)

(B) So AS player2 will always (P2-R) to max player1 payoff, player1 plays D. ⇒ (1.3)

(C) pure strategy equilibria: { (1.3) ✓ ; (0,2) ✗ .

[So there is a pure strategy equilibria (P1-D, P2 R). which is also a subgame equilibria.

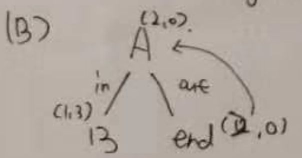
2] Backward induction: (A) Subgam:



(1.3) > (0,2)
so B chases in.

So there is a subgame equilibria & pure strategy equilibria: (A-in , B-in).

(B)



As shown in graph. A chases out.

As A did out, B has No choice but game forced end.
So we can say this is not a equilibria
(or we can still say (A-out , B-forced by game end)
is a whole-game equilibria

Question 7 (5 Marks)

Challenge Question. *The following question is under-specified, but is intended to improve your understanding of subject content and general problem-solving ability, rather than act as an entirely accurate reflection of an exam question.*

Consider a person who is mugged by someone on the street with a gun. The person has an unloaded gun in their own pocket. They could get the gun out to try to scare off the mugger, however, they risk the mugger shooting them instead. If they do not get out the unloaded gun, their mugger has time to search only their left or right pocket, but not both and the mugger will not shoot them. The person has $100 in their left pocket and nothing in their right pocket, and the mugger knows this. Assume that if the mugger does not get the $100, they get no payoff.

Should the person draw their unloaded gun or not? Justify your answer.

To be honest, I'll not risk getting shoot for $100.
Same, I would estimate reward for getting shot of -infinity.

Question 8 (5 marks)

Below is a tree from MCTS in which a collection of roll-outs have been performed, and six nodes expanded are shown (others are omitted as they are not relevant to the question). The notation $V(s) = v$ indicates that v is the value of state s, $N = n$ indicates that the node has been visited n times, and $r = R$ indicates that the reward R is given when this state is visited.

(a). Using backpropagation, calculate the values for the states s and v.

(b). Given these values, what would action should be chosen at state s to maximise expected reward?

(a)

V(v)=1.0(0 + γ12)=γ12, N=5

V(s)=max(

    Q(S,a)=0.6(-1 + γ17)+0.2(-1 + γ8)+0.2(-1 + γ(γ12))

    Q(S,b)=1.0(0+γ190)

)=γ190, N=5+10+5+80=100

(b)

argmax_a(Q(S,a))=b