

# CS3243 Midterm

---

Eric Han

March 10, 2022

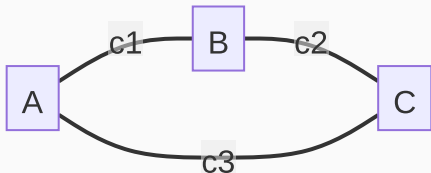
# Announcements

1. Assignment 4 scores are now on Gradebook, please check.
2. Please check turnitin for feedback.

## Good thinking question to further understand AC-3 algorithm

### From student 1

If you did change the domains of the variables you need to recheck some constraints.

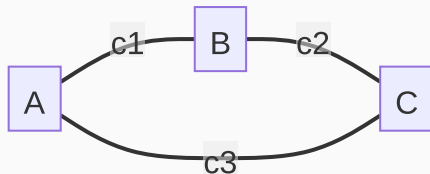


**Figure 1:** Fully connected situation, may need to check again.

### From student 2

Can we consider bidirectional filtering instead of unidirectional filtering?

## Previously from T05, Q3



**Figure 2:** T05,Q3 constraint graph

### Question 3a

$$A = \{1, 2\}, B = \{2, 4\}, C = \{1, 2, 3\}$$

### Question 3b

$$(A, B, C) = (1, 2, 1)$$

## Question 1 [Normal]

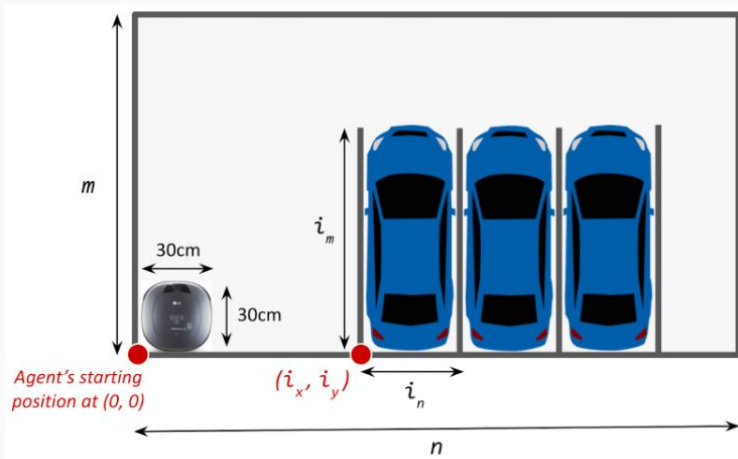


Figure 3: Floor cleaning illustration

**Agent can move:** Up, Down, Left and Right

**Agent size:**  $30 \times 30$

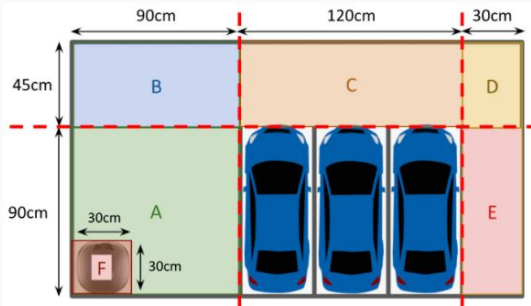
**Agent start:** bottom left

**Parking is orthogonal**

**Environment:**

- Observable
- Single Agent
- Deterministic
- Static
- Continuous
- Sequential

**Recap:** How to formulate?

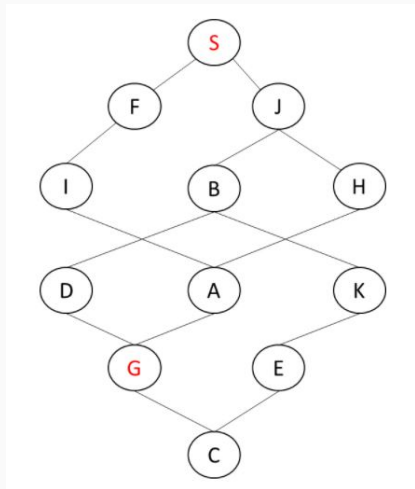


**Figure 4:** Floor cleaning solution

## Answer

- **State representation:** Discretize into cells using GCD -  $A, B, C, D, E, F$ . Then each cell has coordinate and state - clean, not clean and occupied.
- **Initial state:** Fill cells with not clean or occupied from map.
- **Actions:** Up, Down, Left and Right upto cell size, Clean.
- **Transition Model:** Update position, Update State.
- **Step cost:** 1
- **Goal test:** If all cells are clean or occupied.

## Question 2 [Normal/Hard]



**Figure 5:** Q2 Graph

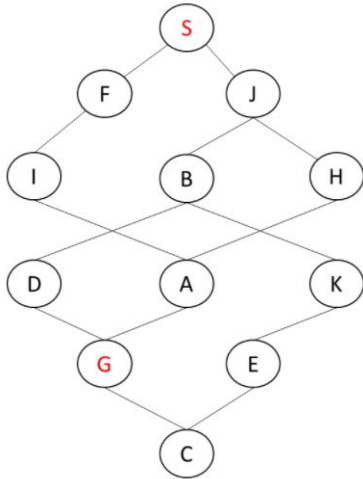
### Question 2a

Trace the following algorithms:

- BFS with limited-graph-based implementation and early goal test.
- BFS with limited-graph-based implementation and late goal test.
- DFS with limited-graph-based implementation

### Question 2b

- Trace the A\* Search with limited-graph-based implementation
- Using A\* search with limited-graph-based implementation



### Answer 2a

- i. S-F-J-I-B-H-A-D-K-A-G
- ii. S-F-J-I-B-H-A-D-K-A-G
- iii. S-J-H-A-I-F-G **[Trap: Graph is undirected!]**

### Answer 2b

- i. S-J-B-F-I-A-D-G
- ii. S-F-I-A-G or S-J-B-D-G

### Recap

Review all the relevant lecture notes.



### Question 3 [Hard]

Let  $C^*$  be the cost of an optimal path. Assuming that every action costs at least some small positive constant  $\epsilon$ . Explain why the time and space complexity of the UCS algorithm is given as:  $O(b^{1+\lceil C^*/\epsilon \rceil})$ .

#### Recap

- What is UCS?
- What is Geometric Series?
- How to compute limits?

### Question 3 [Hard]

Let  $C^*$  be the cost of an optimal path. Assuming that every action costs at least some small positive constant  $\epsilon$ . Explain why the time and space complexity of the UCS algorithm is given as:  $O(b^{1+\lfloor C^*/\epsilon \rfloor})$ .

#### Recap

- What is UCS?
- What is Geometric Series?
- How to compute limits?

#### Answer

Assume at each state has actions  $b$ , at  $k\epsilon$

$k = \lfloor C^*/\epsilon \rfloor + 1$  is generated for goal test.

Using geometric series sum:  $a(\frac{1-r^{n+1}}{1-r})$

Then  $a = 1, r = b, n = \lfloor C^*/\epsilon \rfloor + 1$ :

$$\lim_{b \rightarrow \infty} \frac{1 - b^{\lfloor C^*/\epsilon \rfloor + 2}}{1 - b} \implies O(b^{1+\lfloor C^*/\epsilon \rfloor})$$

1. Initial State: 1 node
2. At level  $k = 1$ :  $b$  nodes  $\dots$
3. At level  $k = \lfloor C^*/\epsilon \rfloor$ :  $b^{\lfloor C^*/\epsilon \rfloor}$  nodes

## Question 4 [Normal]

Suppose you have two admissible heuristics,  $h_1$  and  $h_2$ . You are also given an inadmissible heuristic  $h_3$ . You decide to create the following new heuristic functions.

### Recap

- What is admissible?
- What are the 3 possibilities?
- How to prove admissible?
- How to prove inadmissible?

#### Question 4 - h4

$$h_4(n) = \min(\max(2h_1(n), h_3(n)), h_2(n))$$

#### Question 4 - h4

$$h_4(n) = \min(\max(2h_1(n), h_3(n)), h_2(n))$$

#### Answer

- $\max(2h_1(n), h_3(n))$  inadmissible
- $\min(\text{inadmissible}, \text{admissible})$  is admissible

#### Question 4 - h5

$$h_5(n) = \max(h_1(n), \min(h_2(n), h_3(n)))$$

#### Question 4 - h4

$$h_4(n) = \min(\max(2h_1(n), h_3(n)), h_2(n))$$

#### Answer

- $\max(2h_1(n), h_3(n))$  inadmissible
- $\min(\text{inadmissible}, \text{admissible})$  is admissible

#### Question 4 - h5

$$h_5(n) = \max(h_1(n), \min(h_2(n), h_3(n)))$$

#### Answer

- $\min(h_2(n), h_3(n))$  is admissible
- $\max(\text{admissible}, \text{admissible})$  is admissible

#### Question 4 - h6

$$h_6(n) = \min(h_3(n), \max(h_1(n), 1.1h_2(n)))$$

#### Question 4 - h6

$$h_6(n) = \min(h_3(n), \max(h_1(n), 1.1h_2(n)))$$

#### Answer

Indeterminant, see 2 cases:

1.  $h_1 = h_2 = 0, h_3 = \infty \implies h_6$  admissible
2.  $h_1 = h_2 = h^*, h_3 = \infty \implies h_6$  inadmissible

#### Question 4 - h7

$$h_7(n) = \max((h_1(n) + h_2(n))/2, h_3(n))$$



## Question 4 - h6

$$h_6(n) = \min(h_3(n), \max(h_1(n), 1.1h_2(n)))$$

### Answer

Indeterminant, see 2 cases:

1.  $h_1 = h_2 = 0, h_3 = \infty \implies h_6$  admissible
2.  $h_1 = h_2 = h^*, h_3 = \infty \implies h_6$  inadmissible

## Question 4 - h7

$$h_7(n) = \max((h_1(n) + h_2(n))/2, h_3(n))$$

### Answer

- $(h_1(n) + h_2(n))/2$  is admissible
- $\max(\text{admissible}, \text{inadmissible})$  is inadmissible

## Question 5 [Hard]

Given a set of admissible heuristics:  $h_1, h_2, h_3, \dots, h_n$ , define a new heuristic  $g$  where  $g$  is a function of the set of the heuristics that will result in  $A^*$  expanding a minimal number of nodes while still guaranteeing admissibility.

## Question 5 [Hard]

Given a set of admissible heuristics:  $h_1, h_2, h_3, \dots, h_n$ , define a new heuristic  $g$  where  $g$  is a function of the set of the heuristics that will result in  $A^*$  expanding a minimal number of nodes while still guaranteeing admissibility.

### Answer

- The heuristic that results in  $A^*$  expanding minimal number of nodes is  $h^*$ .
- We are given that  $\forall_s : h_i(s) \leq h^*(s)$

Idea is to get as close as possible to  $h^*$ ; How can we get close given  $h_i$ ?

- Take the max, consequently  $g(s) = \max_{i \in [1, n]}(h_i(s)) \leq h^*(s)$

*Draw a picture to visualize this.*

## Question 6 [Easy/Very Hard]

Wordle Game, we pirate the game to UWordle (unlimited guesses), modelled as follows:

- **State Space:** Each state is represented by the set of candidate words  $C_i$ .
- **Initial State:** Entire dictionary  $D$ .
- **Goal State:** The size of candidate words  $|C| = 1$ .
- **Action:** Pick a word in  $D$ .
- **Transition Model:** From the observation (sequence of colored tiles) after the action, we exclude all words in  $C_i$  that violate the new observed constraints to create  $C_{i+1}$ . Cost of every guess is 1.

P.S. Set by yours truly.

### Recap

Review Tutorial 1.

## Question 6.2a

Determine the environment characteristics of the above problem:

## Question 6.2a

Determine the environment characteristics of the above problem:

## Answer

Environment Characteristic	UWordle
Fully / Partially Observable	Partially Observable
Single / Multi-Agent	Single agent
Deterministic / Stochastic	Deterministic
<b>Episodic / Sequential</b>	Sequential
Static / Dynamic	Static
Discrete / Continuous	Discrete
<b>Known / Unknown</b>	Known

Very important to observe *Partially Observable* and *Known*. Known - whether the agent knows about the rules of the game.

### Question 6.2b

Heuristic  $h_1$  is such that for any  $s_i$ ,  $h_1(s_i) = k$ , where  $k$  is a constant value. State and prove that, only for your chosen value of  $k$ ,  $h_1$  is admissible.

**Question 6.2b**

Heuristic  $h_1$  is such that for any  $s_i$ ,  $h_1(s_i) = k$ , where  $k$  is a constant value. State and prove that, only for your chosen value of  $k$ ,  $h_1$  is admissible.

**Answer**

$k = 0$ , proof:

- $h_1(s_i) = k$
- $h^*(g) = 0, h^*(s_i) \geq 0$  [ $\because$  goal state has the smallest value]
- $h_1(s_i) \leq h^*(s_i)$  [ $\because$  admissibility]
- $k \leq h^*(s_i)$

The only valid  $k$  is 0.



## Question 6.2c

1. Propose another distinctly different but useful<sup>1</sup> heuristic  $h_2$  that dominates  $h_1$ :
  - 1.1 Explicitly state  $h_2$ ,
  - 1.2 explain how  $h_2$  can be computed, and
  - 1.3 prove dominance.
2. Discuss how  $h_2$  is useful to a local search algorithm.
3. Admissibility.
  - 3.1 If your proposed heuristic is admissible, show a proof.
  - 3.2 If your proposed heuristic is not admissible, proof and justify why another admissible heuristic cannot be found.

## Recap

- Environment is Partially Observable, unknowns:  $C_i$ ,  $w_{\text{target}}$ , next observations
- What is a dominating relationship?

---

<sup>1</sup>Similar to Tutorial 3, Q1

### Answer 6.2c.1 [Using Bayesian Statistics, Unknown Agent]

We can compute a function  $f(\ell, C_i)$  which maps the letter to the number of times the letter  $\ell$  is observed in each  $C_i$ . e.g. For  $C_i = \{\text{hello, world}\}$ , then  $f$  is given as follows:

$\ell$	h	e	l	o	w	r	d
$f(\ell, C_i)$	1	1	3	2	1	1	1

Then  $h_2(s_i)$  is defined over the occurrence of the letters in word  $w$  over the  $f(\ell, C_{i-1})$ :

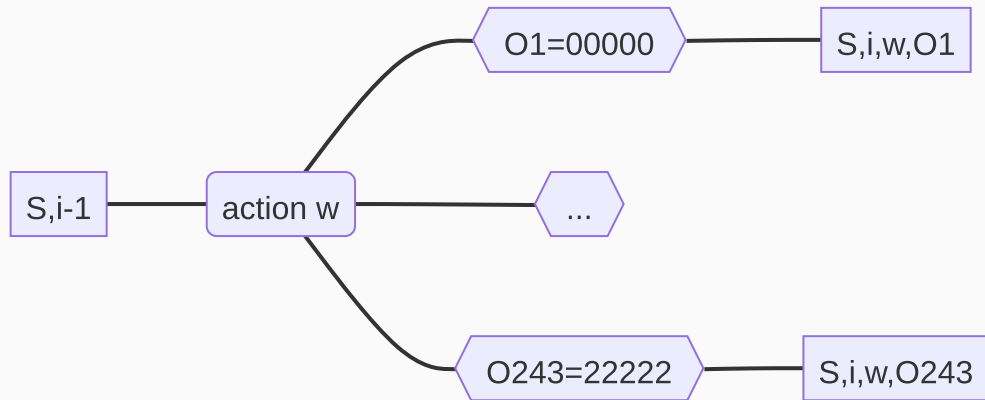
$$h_2(s_i) = \prod_{\ell \in w} \frac{f(\ell, C_{i-1})}{5 \times |C_{i-1}|}$$

The idea here is to calculate the probability of drawing the letters in word  $w$  from letters in  $C_{i-1}$  with replacement.

### Answer 6.2c.1 [Using Transition, Known Agent]

State  $h_2$ , explain how  $h_2$  can be computed and prove dominance.

**Idea:** We want to reduce  $C_i$  as much as possible;  $(0, 1, 2) = (\text{Green}, \text{Yellow}, \text{Grey})$ .  
Since there are 5 letters each with 3 possible observations -  $3^5 = 243$  observations.



For every word  $w$ ,

- We have each  $C_i^{(j)}$  from every observation  $j \in [1, 3^5]$ .
- Now, we can calculate statistics using each  $C_i^{(j)}$ .

One way is to take the expectation over all observations:

$$h_2(s_i) = \mathbb{E} \left[ \frac{|C_i|}{|D|} \right] = \mathbb{E} \left[ \forall_j \frac{|C_i^{(j)}|}{|D|} \right]$$

Or using information entropy; idea is to choose the state that gives us the most info:

$$h_2(s_i) = \sum_{j \in [1, 3^5]} \left[ \frac{|C_i^{(j)}|}{|C_{i-1}|} \log_2 \frac{|C_i^{(j)}|}{|C_{i-1}|} \right]$$

To generalize, consider  $l > 1$  lookahead; note that we are currently looking ahead 1 step.

$h_2$  dominates  $h_1$ :

- $h_1 = 0$  for all states.
- We note that the smallest  $h_2$  is when each observation  $C_i^{(j)}$  is very imbalanced.
- But in that case, it is non-zero  $> 0$ .

$$\min[h_2(s_i)] > 0 = h_1$$

## Answer 6.2c.2

Discuss how  $h_2$  is useful to a local search algorithm.

- We take a 1 step lookahead to see the effect of taking the action  $w$
- We compute over the possible observations to calculate the information gain
- We choose the state with the highest expected information gain

### Answer 6.2c.3

Proof inadmissibility and that there cannot be any other admissible heuristics.

Let  $s_i$  to be the goal state, then  $h^*(s_i) = 0$ , but  $h_2(s_i)$  is non zero. Assume an admissible heuristic  $h'$ :

- At the current state, the target word can be the action.
- If the agent takes the action, the goal state will be reached.
- The  $h^*(g) = 0$ , and consequently  $h'(g) = 0$  for admissible heuristic.
- For  $h'(g) = 0$ ,  $h'$  must have known  $g$  is the goal state or  $h' = 0$ .
- But  $h'$  does not have access to the exact next state but only an estimate.
- So,  $h'$  cannot be admissible unless  $h' = 0$ . There are no other admissible heuristics.

Inadmissible.  $h_2(g) \neq 0$

**Note:** Usually we proof inadmissibility by giving example; But this is an exception.