CS2109s - Tutorial 2

Eric Han (TG12-TG15)

Feb 8, 2024

Annoucements

Important admin

1. Attendance Marking will be done at the end of the lesson via the QR code.
   - Random checks may be performed.
   - Bonus / [@] declaration on the attendance form, no need to put on chat anymore.
2. Survey Results
3. If you are going to submit late for PS, it would help if you can inform me early!

Problem Set 1

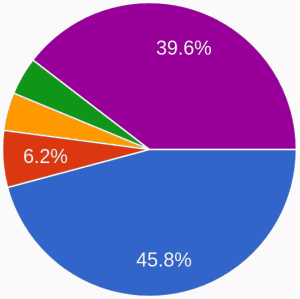
Great job to everyone for submitting all on time! Here are some general comments:

- When giving state representation,
  - Do not model implementation details; ie. list - history of the actions etc.. Focus only on representing the problem's state
  - You should show me the representation explicitly mentioning the datastructure, ie. tuple of (x,y,z). Also for initial state and goal state, explicitly mention the exact state would be better ie. (0,0,0)
- There are many answers which are missing actions where we asked for it; Make sure you read and answer carefully!
- Some students used list to check for visited, make sure you use an efficient datastructure ie. set or hashmap.

Survey Results

Why are you taking CS2109s?

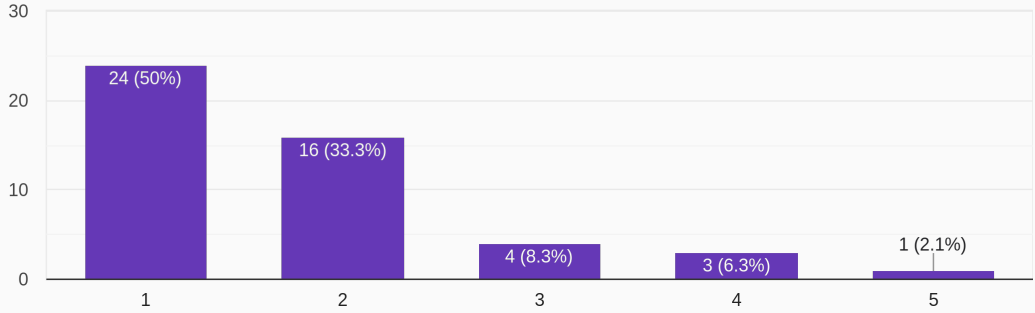
48 responses



- I like AI, want to learn more
- I want to work in AI when I graduate!
- Dunno leh, my friends taking
- Needed to take some modules, so...
- Core module, do bian...

How much do you know about AI/ML?

48 responses



What do you want to achieve in tutorials?

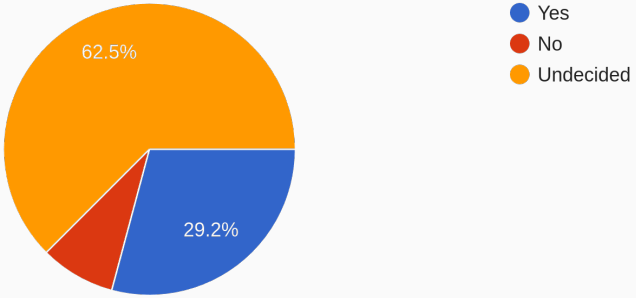
- [32] Learn about AI / Understand Lecture content better
- [3] Get more EXP
- [1] Achieve Full attendance
- [7] Practically apply AI/ML
- [2] Have fun learning
- [1] Make friends

Any suggestions to make our classes effective? [Selected]

- I really like the buddy system: i think it should be done in other tutorials as well
- Go through the thought process of how to derive the solution to the problem
- Maybe cover more from the lecture / start w short recap of last lecture maybe?
- Maybe some additional resources for the content to practice
- I might be one of the students who may not understand new concepts well, so maybe more time can be spent on tougher concepts
- the tougher tutorial questions can js go through, i think asking for answers from the class might not be... efficient
- Perhaps gamify it, or can bring in some real life examples, and statistics about the working world (like anticipated demand, current supply, future salary, basically to help us better understand if we should specialise in AI)!

Would you be interested to participate in social gatherings (across Eric's CS2109s classes) ?

48 responses



Question 1 [G]

In case you havn't: https://www.google.com/logos/2010/pacman10-i.html

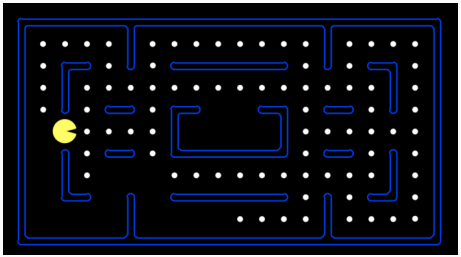


Figure 1: Pac-Man Simplified

- **State representation**: Position of Pac-Man and the positions of the uneaten pellets
- **Initial State**: Filled grid entirely with pellets
- **Goal State**: No pellets left in the grid
- **Action**: Moving up/down/left/right
- **Transition Model**: Updating the position of Pac-Man and eating pellet (if applicable)
- **Cost function**: 1 for each action taken

Give a non-trivial (whats a trivial heuristic?) admissible heuristic for this problem.

Recap

- What is the intuition behind A* Search?
- What is an admissible heuristic?
- What is a consistent heuristic?

Recap

- What is the intuition behind A* Search?
- What is an admissible heuristic?
- What is a consistent heuristic?

Summary

- A* Search: $f(N) = g(N) + h(N)$
- Admissible heuristic: $h(N) \leq h^*(N)$
- Consistent heuristic: $h(N) \leq c(N, N') + h(N')$

Where,

- $f(.)$ - Evaluation Function.
- $g(.)$ - Cost Function from start to current node.
- $h(.)$ - Estimated cost from current node to goal.
- $c(N, N')$ - Action cost from N to N'.

Answer

- Trival
  - h_1 : Number of pellets left at any point in time.
- Non-Trival
  - h_2 : The Maximum among all Manhattan distances from each remaining pellet to the current position of Pac-Man.
    - Admissible, $\max_{p \in P}$ path over all pellets ensure that $\leq h^*$. ie. furthest
  - h_3 : The average over all Euclidean distances from each remaining pellet to the current position of Pac-Man.
    - Admissible, $h_3 \leq h_2 \leq h^*$

Or.. Relax the problem - Pac-Man can pass through walls:

- With less restrictions, we can take short cuts (ie. adding edges between states)
- Optimal solution to the original is a solution to the relaxed, but may not be optimal
- We can find the optimal for relaxed problem on the path by taking short cuts
- Cost of optimal to the relaxed is always lower than the original's
- making it an admissible heuristic for the original problem.

Question 2

Given a graph $G = (V, E)$ where,

- each node v_n having coordinates (x_n, y_n),
- each edge (v_i, v_j) having weight equals to the distance between v_i and v_j, and
- a unique goal node v_g with coordinates (x_g, y_g),

$$h_{SLD}(n) = \sqrt{(x_n - x_g)^2 + (y_n - y_g)^2}$$
$$h_1(n) = max\{|x_n - x_g|, |y_n - y_g|\}$$
$$h_2(n) = |x_n - x_g| + |y_n - y_g|$$

a. Is $h_1(n)$ an admissible heuristic? Proof.
b. Is $h_2(n)$ an admissible heuristic? Proof.
c. Which heuristic function would you choose for A* search? And why?

Recap

- What is the intuition behind A* Search?
- How to decide admissibility?
  - If admissible... Show
  - If not admissible... Show
- How is a heuristic *useful*, how to decide which to use?
- What is the h^* optimal heuristic?
- What is a potential downfall of choosing an optimal heuristic?
- Admissible heuristic: $h(N) \leq h^*(N)$

Answer

a. Admissible, proof:
   - h_{SLD} is admissible - $h^*(n) \geq h_{SLD}(n) = \sqrt{(x_n - x_g)^2 + (y_n - y_g)^2}$
     - $\sqrt{(x_n - x_g)^2 + (y_n - y_g)^2} \geq \sqrt{(x_n - x_g)^2} = |x_n - x_g|$
     - $\sqrt{(x_n - x_g)^2 + (y_n - y_g)^2} \geq \sqrt{(y_n - y_g)^2} = |y_n - y_g|$
   - $h^*(n) \geq h_{SLD}(n) = \sqrt{(x_n - x_g)^2 + (y_n - y_g)^2} \geq h_1(n) = max\{|x_n - x_g|, |y_n - y_g|\}$

b. Not admissible, counter example:
   - Consider a graph where v_s has coordinates $(0, 0)$ and v_g has coordinates $(1, 1)$
   - $h^*(v_s) = \sqrt{2} < h_2(v_s) = 2$.

c. $h_{SLD}(n)$:
   - $h_2(n)$ is not admissible, so we may not get an optimal solution if we use it
   - $h_{SLD}(n) \geq h_1(n)$ so $h_{SLD}(n)$ dominates $h_1(n)$
   - will incur less search cost (on average) if we use $h_{SLD}(n)$.
   - ***Intuition***: *Choose the heuristic that is closest, but under* h^*

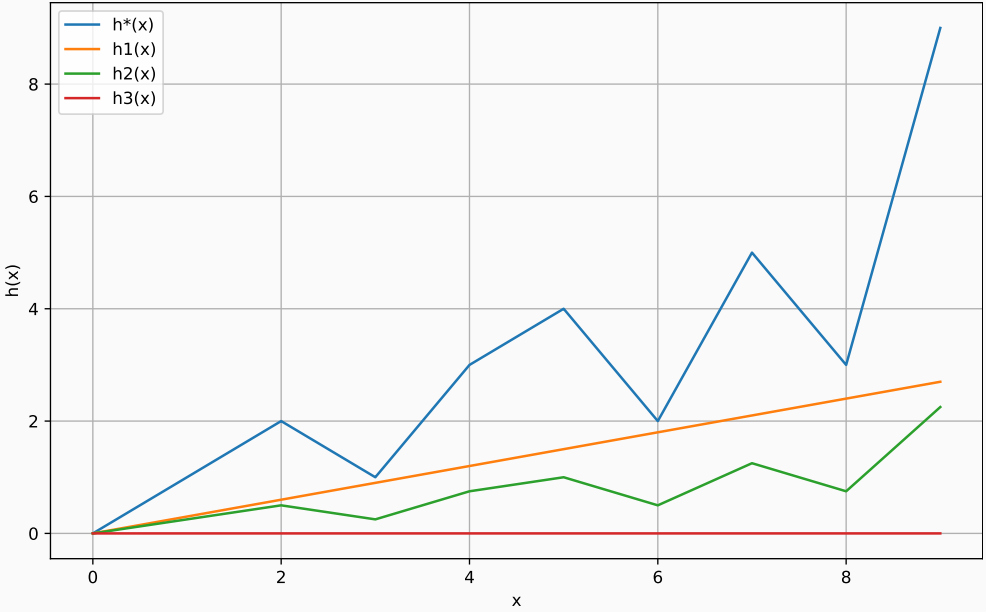


Figure 2: Dominance vs Admissible heuristic

Question 3 [G]

a. Given that a heuristic h is such that $h(G) = 0$, where G is any goal state, prove that if h is consistent, then it must be admissible.
b. Give an example of an admissible heurstic function that is not consistent.

Recap

- Admissible heuristic: $h(N) \leq h^*(N)$
- Consistent heuristic: $h(N) \leq c(N, N') + h(N')$

Answer 3a

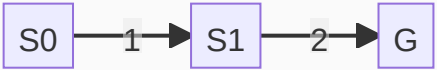
Intuition: Show on the number of action required to reach the goal from n to goal G.

Let the no. of actions k to be required to reach from n_k to G on optimal path $P_{n_k \to G}$. Node n_k is k steps away from G, ie. $k = 3 \implies P_{n_3 \to G} : n_3 \to n_2 \to n_1 \to G$.

- **Base**: 1 action; i.e. node n_1 is one step away from G.
  - Since consistent, $h(n_1) \leq c(n_1, G) + h(G)$ and $h(G) = 0$
  - $\implies h(n_1) \leq c(n_1, G) = h^*(n_1) \implies$ admissible.
- **Inductive**: Assume for $k-1$ actions, path $P_{n_{k-1} \to G}, h(n_{k-1}) \leq h^*(n_{k-1})$.
  - Since consistent, $h(n_k) \leq c(n_k, n_{k-1}) + h(n_{k-1})$
  - $\implies h(n_k) \leq c(n_k, n_{k-1}) + h^*(n_{k-1}) = h^*(n_k) \implies$ admissible.

Answer 3b

Intuition: Construct an admissible heuristic and make it not consistent.

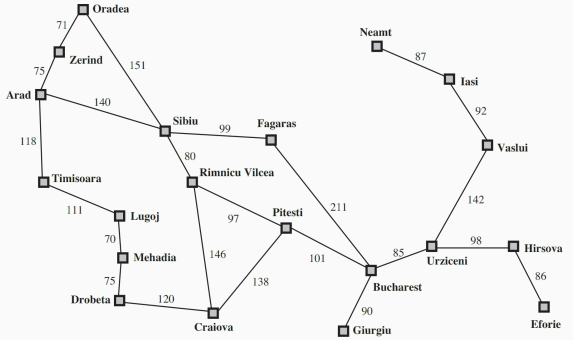


s	$S0$	$S1$	G
$h(s)$	3	1	0
$h^*(s)$	3	2	0
$h(s) \leq h^*(s)$	T	T	T

Heuristic h is

- Admissible - $\forall s : h(s) \leq h^*(s)$
- Not consistent - $3 = h(s_0) > c(s_0, s_1) + h(s_1) = 2$

Question 4 [G]



Values of h_{SLD} - straight-line distances to Bucharest

Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Figure 3: Graph of Romania.

Considering, $h(n) = |h_{SLD}(\text{Craiov}) - h_{SLD}(n)|$

a. Trace A* search (`TREE-SEARCH`) by expanding the search trees, showing (g, h, f)
b. Prove that $h(n)$ is an admissible heuristic.

Answer 4a

The tuple in each node denotes (g, h, f) are shown along the trace:

```
Fagaras (0, 16, 16)
|-- Bucharest (211, 160, 371)
+-- Sibiu (99, 93, 192)
    |-- Arad (239, 206, 445)
    |-- Fagaras (198, 16, 214)
    |   |-- Bucharest (409, 160, 569)
    |   +-- Sibiu (297, 93, 390)
    |-- Oradea (250, 220, 470)
    +-- Rimnicu_Vilcea (179, 33, 212)
        |-- Craiova (325, 0, 325)
        |-- Pitesti (276, 60, 336)
        +-- Sibiu (259, 93, 352)
```

Answer 4b

Let D be the straight-line distance between n and Craiova:

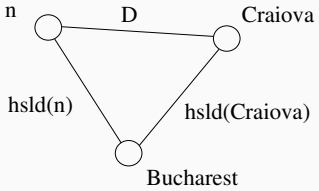


Figure 4: Triangle Inequality Example

- By the Triangle Inequality, D must be at least as much as the difference of the 2 other sides, ie. $D \geq |h_{SLD}(Craiova) - h_{SLD}(n)| = h(n)$.
- But we also know that $h^*(n) \geq D \geq h(n)$, so $h^*(n) \geq h(n)$, and
- $h(n)$ is admissible.

Note: The definition of triangle inequality is, sum of the lengths of any two sides must be greater than or equal to the the length of the third side.

Question 5

Using the example graph,

a. Show that A* using graph search returns a non-optimal solution path from start S to G when using an admissible but inconsistent $h(n)$.
b. Then, show that tree search will return the optimal solution with the same heuristic.

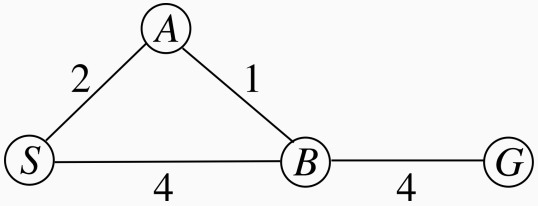


Figure 5: Example graph; We assume that $h(G) = 0$.

Table 2: Inconsistent heuristic; $h(S) > h(B) + 4$

.	S	B	A	G
$h(.)$	7	0	3	0
h^*	7	4	5	0

Answer 5a

Intuition: We may miss out on some shorter paths that we mistakenly think are long after reaching the node as the heuristic function violates the triangle inequality.

The tuple in each node denotes (g, h, f) are shown along the trace:

```
S (0, 7, 7)
|--  A (2, 3, 5)
|    |-- [X] B (3, 0, 3)
|    +-- [X] S (4, 7, 11)
+--  B (4, 0, 4)
     |--  A (5, 3, 8)
     |    |-- [X] B (6, 0, 6)
     |    +-- [X] S (7, 7, 14)
     |--  G (8, 0, 8)
     +-- [X] S (8, 7, 15)
```

Frontier

```
S(7-)
B(4-S) A(5-S)
A(5-S) A(8-SB) G(8-SB)
A(8-SB) G(8-SB)
G(8-SB)
```

Answer 5b

```
S (0, 7, 7)
|--  A (2, 3, 5)
|   |--  B (3, 0, 3)
|   |   |--  A (4, 3, 7)
|   |   |   |--  B (5, 0, 5)
|   |   |   |   |--  A (6, 3, 9)
|   |   |   |   |--  G (9, 0, 9)
|   |   |   |   +--  S (9, 7, 16)
|   |   |   +--  S (6, 7, 13)
|   |   |--  G (7, 0, 7)
|   |   +--  S (7, 7, 14)
|   +--  S (4, 7, 11)
+--  B (4, 0, 4)
    |--  A (5, 3, 8)
    |--  G (8, 0, 8)
```

Frontier

```
S(7-)
B(4-S) A(5-S)
A(5-S) A(8-SB) G(8-SB) S(15-SB)
B(3-SA) A(8-SB) G(8-SB) S(11-SA) S(15-SB)
A(7-SAB) G(7-SAB) A(8-SB) G(8-SB) S(11-SA) S(14-SAB) S(15-SB)
B(5-SABA) G(7-SAB) A(8-SB) G(8-SB) S(11-SA) S(13-SABA) S(14-SAB) S(15-SB)
G(7-SAB) A(8-SB) G(8-SB) A(9-SABAB) G(9-SABAB) S(11-SA) S(13-SABA) S(14-SA
    S(15-SB) S(16-SABAB)
```

Question 6 [@]

Assume no negative cycles.

a. Would A* work with negative edge weights? If yes, prove it; otherwise, provide a counterexample.
b. Why would A* work with negative weights not not Dijkstra's?

Answer 6

a. Recall that in the proof of A*, negative weights do not affect the optimality of A* as long as there are no negative cycles. Therefore, the same proof applies.
b. A* considers f-score while Dijkstra's considers distance ie. g-score. The h-score gives A* additional information.

Bonus Question

To help you further your understanding, not compulsory; Work for Snack/EXP!

Tasks

1. Fork the repository https://github.com/eric-vader/CS2109s-2324s2-bonus
2. We will be first solving Question 5 using code, `astar(graph, inital_node, goal_test, heuristics, is_tree, is_update)` that returns the **best path** found:
   2.1 Able to solve 5a via `is_tree=False,is_update=False`
   2.2 Able to solve 5b via `is_tree=True,is_update=False`
3. Some code have been implemented for you; You can reuse the PQ or use another.
4. You should print the frontier and explored at the beginning of the loop.

To claim your snack & EXP, show me your forked repository and your code's output.

Buddy Attendance Taking

1. [@] and Bonus declaration is to be done here; You should show bonus to Eric.
2. Attempted tutorial should come with proof (sketches, workings etc. . .)
3. Random checks will be conducted - `python ../checks.py T13`

Figure 6: Buddy Attendance: https://forms.gle/jsGfFyfo9PTgWxib6