CS 240: Lab 2
Missionaries and Cannibals

TA: Sai Sriram Sambaraju

Instructions

- This lab will be **graded**.
- Please read the problem statement and submission guidelines carefully.
- For any doubts or questions, please contact either the TA assigned to your lab group or the TA involved in making the lab.
- The deadline for this lab is **Thursday, 23 January, 5 PM** but solutions till 5:30 PM will be accepted. Submissions after 5:30 will incur a penalty of 50% till Sunday, 26 January, 11:59 PM.
- You are not allowed to use any kind of LLM for code generation. You can refer to pseudo codes.
- The submissions will be checked for plagiarism, and any form of cheating will be penalized.

Missionaries and Cannibals

The **Missionaries and Cannibals** problem is a classic river-crossing puzzle. The goal is to move missionaries and cannibals across a river using a boat. However, the boat can carry at most **3 people** at a time, and there are some constraints to consider:

- If there are more cannibals than missionaries on either side of the river or on the boat, the cannibals will eat the missionaries.
- The boat cannot travel by itself; it must always have at least one person on board to operate it.

The problem is to find a sequence of moves that transfers all the missionaries and cannibals from the left bank to the right bank of the river without violating the constraints.

State Representation

Each state in the problem is represented as a list of 3 integers :

$$[\texttt{m\_left}, \texttt{c\_left}, \texttt{boat\_position}]$$

Where:

- `m_left`: Number of missionaries on the left bank
- `c_left`: Number of cannibals on the left bank
- `boat_position`: 1 if boat is on left bank, 0 if on right bank

Since, the total number of missionaries and cannibals is fixed we can find the number of missionaries and cannibals on the right bank.

Tasks to be Completed

Complete the following tasks in the provided Python file:

Task 1. [10 marks] Implement the `check_valid` function to verify if a given state is valid.

Task 2. [10 marks] Implement the `get_neighbours` function to generate all valid neighboring states.

Task 3. [5 marks] Implement the `gstar` function to calculate the cost between two states.

Task 4. [15 marks] Implement heuristic functions `h1`, `h2`, `h3`, `h4` and `h5` to estimate the cost to reach the goal from the current state.

Task 5. [28 marks] Implement the `astar_h1` function to perform A* search with heuristic h1 and find the optimal path and check if the heuristic h1 satisfies the monotone restriction property while exploring.

Task 6. [8 marks] Implement the `astar_h2` function to perform A* search with heuristic h2 and find the optimal path and check if the heuristic h2 satisfies the monotone restriction property while exploring.

Task 7. [8 marks] Implement the `astar_h3` function to perform A* search with heuristic h3 and find the optimal path and check if the heuristic h3 satisfies the monotone restriction property while exploring.

Task 8. [8 marks] Implement the `astar_h4` function to perform A* search with heuristic h4 and find the optimal path and check if the heuristic h4 satisfies the monotone restriction property while exploring.

Task 9. [8 marks] Implement the `astar_h5` function to perform A* search with heuristic h5 and find the optimal path and check if the heuristic h5 satisfies the monotone restriction property while exploring.

Submission

- Submissions should be made on Moodle. Submit the Python file renamed as `rollnumber1_rollnumber2.py` (the "b" in roll number should be in small case).
- The soft deadline for submission is 5:30 pm. Submission after that will incur a penalty of 50% till Sunday, 26 January, 11:59PM.
- Only one person per team should submit their solution.

Evaluation

This lab will be evaluated in two phases- Auto-grading and Viva. For auto-grading, each of the tasks will be evaluated independently, and the respective marks are as shown in the **Tasks to be Completed** section.