Luis Caro
802- 14- 1136
ICOM5015-030
03/26/ 21

Homework Assignment 2

3.7

1. **Suppose the state space consists of all positions (x, y) in the plane. How many states are there? How many paths are there to the goal?**

    - We know by definition that a state space is all points or nodes that can be found or represented in a graph. Links connect these nodes; a collection of these links form a path. Therefore, since we are not working inside a domain or range, we can assume that there are infinite states and paths to the goal.

2. **Explain briefly why the shortest path from one polygon vertex to any other in the scene must consist of straight-line segments joining some of the vertices of the polygons. Define a good state space now. How large is this state space?**

    - We know that the shortest path between two points is a straight line therefore, when working with polygons we have to try to find the order of segments that reach the goal that are closest to a straight line, this is achieved by going from the starting point to the nearest vertices and so on and so forth until reaching the goal. The straightest line is the one that goes to the vertex and not around, as well as the segments that join in corners. The state space consists of the coordinates of the reachable vertices (Must be reachable, no clipping through objects is allowed) as well as the coordinates of the goal.

3. **Define the necessary functions to implement the search problem, including an ACTIONS function that takes a vertex as input and returns a set of vectors, each of which maps the current vertex to one of the vertices that can be reached in a straight line. (Do not forget the neighbors on the same polygon.) Use the straight-line distance for the heuristic function.**

    - This is implemented in the code. For the ACTIONS function I use the visible _vertices in the ***vv.py script***.

4. **Apply one or more of the algorithms in this** chapter **to solve a range of problems in the domain, and comment on their performance.**

    - I chose to only implement Dijkstra's shortest path algorithm because it is an improvement of the breadth-first search. Performance-wise we can say that its worst case performance is O(| V |*log(| V |) + | E |). An implementation of the algorithm can be found in the ***shortest_path.py*** **script**.

2. The **missionaries and cannibal's** problem is usually stated as follows. Three mission- aries and three cannibals are on one side of a river, along with a boat that can hold one or two people. Find a way to get everyone to the other side without ever leaving a group of mis- sionaries in one place outnumbered by the cannibals in that place. This problem is famous in AI because it was the subject of the first paper that approached problem formulation from an analytical viewpoint (Amarel, 1968).

1. **Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution. Draw a diagram of the complete state space.**

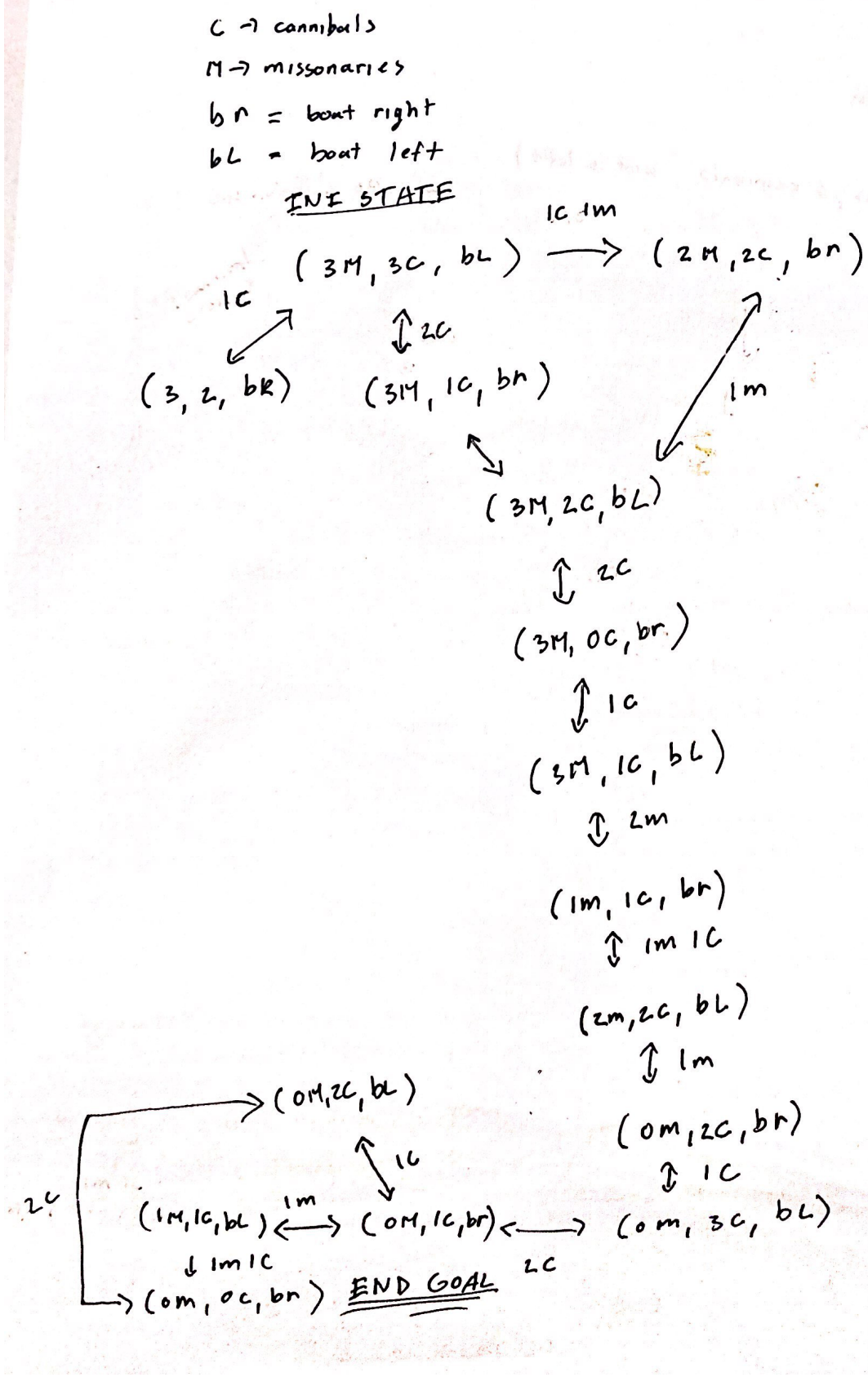
    - **Problem:**
        - There are three cannibals and three missionaries on the left bank of a river.
        - They wish to cross to the other side of the riverbank (right) using a boat.
        - The number of cannibals cannot surpass the number of missionaries on the same bank. This is to prevent the missionaries from dying
    - **State Diagram: (One side the other can be inferred)**
        - **BELOW**

C → cannibals

M → missonaries

bR = boat right

bL = boat left



2. **Implement and solve the problem optimally using an appropriate search algorithm. Is it a good idea to check for repeated states?**

    - The implementation is found in the file. It isn't necessary to search for repeated states since all nodes with the exception of the first and last node have only one other choice.

3. **Why do you think people have a hard time solving this puzzle, given that the state space is so simple?**

    - The answer isn't so evident due to the large branching it has, even though the state may seem simple.