Introduction

The missionaries and cannibals is an artificial intelligence problem that deals with three missionaries and three cannibals must cross a river using a boat which can carry at most two people. The missionaries cannot be outnumbered by the cannibals otherwise they get eaten.

The missionaries and cannibals problem comprises of 3 missionaries and 3 cannibals on one side of the river bank. Our task here is to take the 3 cannibals and 3 missionaries on the other side of the bank without letting the missionaries being outnumbered which eventually leads to them being eaten. They must cross the river using a boat which can carry at most 2 people.

Conditions

- The boat can carry at most 2 of them.
- The boat can also travel itself without a missionary or a cannibal on it.
- On both sides of the river bank the missionaries should not be outnumbered by the cannibals.

Objectives

The objectives of this assignment are:

- To find an appropriate solution for the problem using BFS.
- To visualize the program using proper graphic tools.

Design and implementation

Software used:

- Editor used - Pycharm
- Language used – Python
- Operating system used – Windows 10
- Graphics library used – Tkinter

Methodology

Suppose that we denote the position of the missionaries, cannibals and boat by (M, C, B) respectively.

If the value of B is 0 then the boat is on the left side of the river. Similarly if the value of B is 1 then the boat is on the right side of the river.

We consider that at the beginning all the missionaries and cannibals are on the left bank of the river. Our initial state is (3, 3, 0) which signifies that all 3 missionaries and 3 cannibals are on the same side of the river and the boat is on the left side of the river.

Our goal state is (0, 0, 1) which signifies that there are no missionaries and cannibals left on the left bank and the boat is on the right side of the river.

Initial State:

(3, 3, 0)

Final State:

(0, 0, 1)

Constraint:

The number of cannibal should not exceed the no of missionaries on either side of the river.

Classes Used: make_graph

This class stores the node and its children which can be used to make the graph.

State

- It has the states which are possible (Missionary, Cannibal, Boat Position) It contains the Goal State (0, 0, 1)
- Checks if the new state is valid or not.

child_nodes

- Finds the children for a given parent.
- The possible states are:
    - If the boat is in left bank
        - Move 1 cannibal to right
        - Move 2 cannibal to right
        - Move 1 missionary to right
        - Move 2 missionary to right
        - Move 1 missionary and 1 cannibal to right ○ If the boat is in the right bank
        - Move 1 cannibal to left
        - Move 2 cannibal to left
        - Move 1 missionary to left
        - Move 2 missionary to left
        - Move 1 missionary and 1 cannibal to left

Search Used: Breadth First Search

1. In breadth first search we maintain a queue. It uses first in first out.
2. Place node into the list.
3. Dequeue a element from a list and find its decendents.
4. If the decendents are explored, if not add it to queue.
5. If queue is empty, goal cann't be reached.
6. If queue is not empty dequeue next element and continue till we reach goal.

Output

```
Missionaries and Cannibals solution:
(M,C,B)
(3,3,0)
(3,1,1)
(3,2,0)
(3,0,1)
(3,1,0)
(1,1,1)
(2,2,0)
(0,2,1)
(0,3,0)
(0,1,1)
(1,1,0)
(0,0,1)
```

Fig. Output Steps In CONSOL

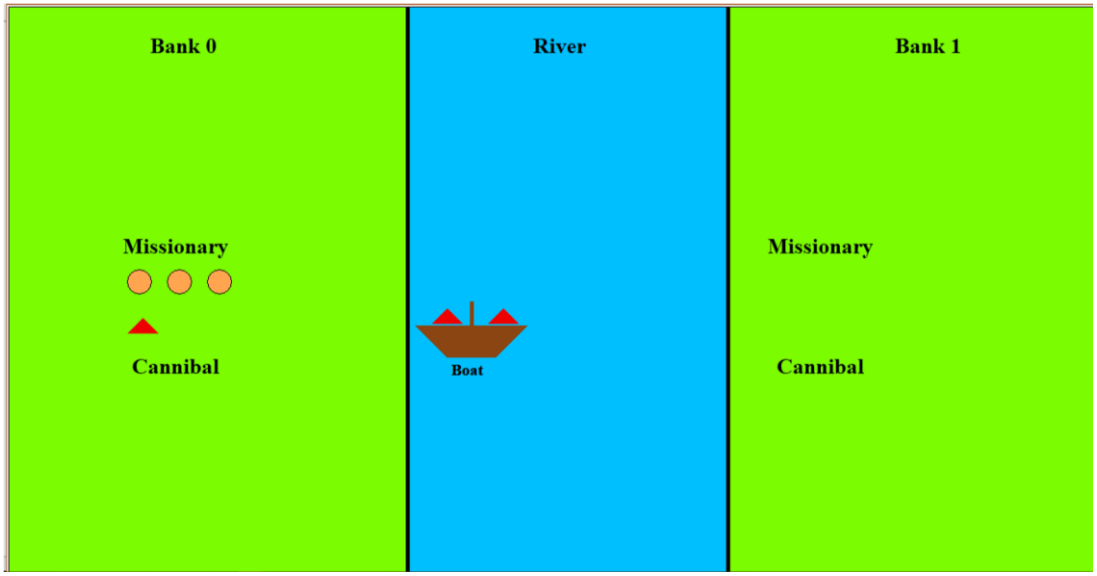

Fig. Graphics Simulation 1

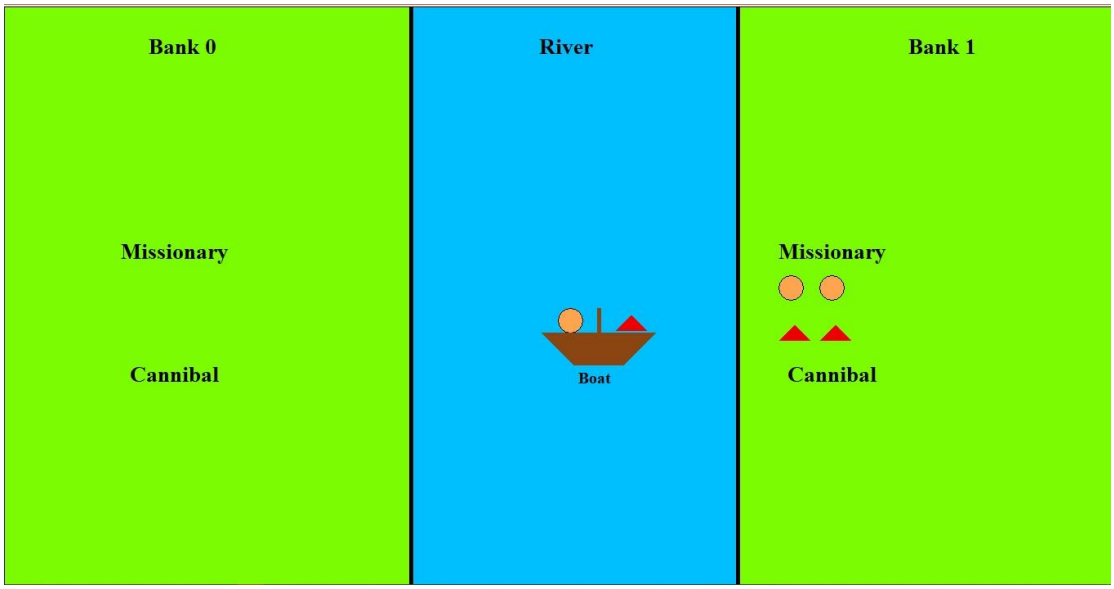

Fig. Graphics Simulation 2

Conclusion and Recommendation

Hence, the given project was completed with much efficiency and effectiveness and the state space of the various states in question were properly oriented. Overall, this project was a very fruitful one.

Limitation

The portrayal of the project has been done using a simple graphics library so the GUI is simple.