



A PROJECT REPORT

On

MISSIONARIES-CANNIBALS PROBLEM

Submitted to CMREC (UGC Autonomous)

In Partial Fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

ARTIFICIAL INTELLIGENCE

Submitted By

R. DUTTA YESHWANTH

228R1A12B4

Under the guidance of

Mrs. G. SWETHA

Assistant Professor

Internal Guide

Mrs. G. SWETHA

Assistant professor,

Department of IT

Head of the Department

Dr. MADHAVI PINGILI

Assoc Professor & HOD,

Department of IT

TABLE OF CONTENTS

S. No.	Section	Page Number
1	ABSTRACT	1
2	INTRODUCTION	2
3	IMPLEMENTATION	3-4
4	OUTPUT	5
5	CONCLUSION	6

ABSTRACT

The Missionaries and Cannibals problem is a foundational AI puzzle that illustrates state-space search and constraint satisfaction, core concepts in artificial intelligence. The problem involves transporting three missionaries and three cannibals across a river using a boat that holds a maximum of two people, with the constraint that missionaries can never be outnumbered by cannibals on either riverbank. This challenge requires designing a safe sequence of moves that adheres to these conditions. Implementing the solution in Python involves representing each state as a tuple and using the Breadth-First Search (BFS) algorithm to systematically explore all possible states, ensuring the shortest path to the solution. Python's efficient data structures, such as queues and sets, simplify state management, making BFS traversal effective. This problem exemplifies constraint-based decision-making and optimization, key skills in AI, and serves as a practical introduction to search algorithms and problem-solving in artificial intelligence. To solve it, search algorithms like Breadth-First Search (BFS) are commonly used, as BFS is guaranteed to find the shortest solution. This makes BFS especially effective for this problem, as it explores states systematically and avoids unnecessary backtracking, ensuring the solution is efficient.

Keywords: Transition State, Feasible State, Goal State, Heuristic Search, Search Tree, Boat Capacity, Conflict-Free State, Artificial Intelligence (AI), Logical puzzles.

INTRODUCTION

The Missionaries and Cannibals problem is a classic AI puzzle that challenges problem-solving and search algorithm skills. In this problem, three missionaries and three cannibals are on one side of a river and must cross to the other side using a boat. The boat can hold up to two people at a time, and at least one person must row it across. The challenge is that, at any time on either side of the river, missionaries cannot be outnumbered by cannibals, or the missionaries would be at risk, making the configuration invalid. The goal is to find a safe sequence of moves that transfers all individuals to the other side of the river without violating this constraint. This problem is significant in artificial intelligence because it can be framed as a state-space search problem. Each state represents the number of missionaries, cannibals, and the boat's position on each side of the river. The allowed actions are specific moves that respect the safety conditions, and the final state (or goal) is reached when all missionaries and cannibals are safely on the opposite side of the river. To solve it, search algorithms like Breadth-First Search (BFS) are commonly used, as BFS is guaranteed to find the shortest solution. This makes BFS especially effective for this problem, as it explores states systematically and avoids unnecessary backtracking, ensuring the solution is efficient. Implementing this problem in Python involves using tuples or lists to represent each state and applying BFS to traverse possible states from the initial configuration to the goal.

IMPLEMENTATION

SOURCE CODE:

```
from collections import deque

initial_state = (3, 3, 1) # (Missionaries on left, Cannibals on left, Boat on left)
goal_state = (0, 0, 0)    # (Missionaries on left, Cannibals on left, Boat on left)

def is_valid_state(m, c):
    # More missionaries than cannibals on any side is invalid, or negative numbers
    return (m >= 0 and c >= 0 and (m == 0 or m >= c))

def get_next_states(m, c, boat):
    possible_moves = [
        (1, 0), # One missionary
        (0, 1), # One cannibal
        (1, 1), # One missionary and one cannibal
        (2, 0), # Two missionaries
        (0, 2) # Two cannibals
    ]
    next_states = []

    for move in possible_moves:
        if boat == 1: # Boat on the left side
            new_m, new_c, new_boat = m - move[0], c - move[1], 0
        else: # Boat on the right side
            new_m, new_c, new_boat = m + move[0], c + move[1], 1

        if is_valid_state(new_m, new_c) and is_valid_state(3 - new_m, 3 - new_c):
            next_states.append((new_m, new_c, new_boat))
    return next_states

def solve_missionaries_cannibals():
    queue = deque([(initial_state, [])]) # Queue stores (state, path)
    visited = set([initial_state])

    while queue:
        current_state, path = queue.popleft()

        if current_state == goal_state:
            return path + [current_state]

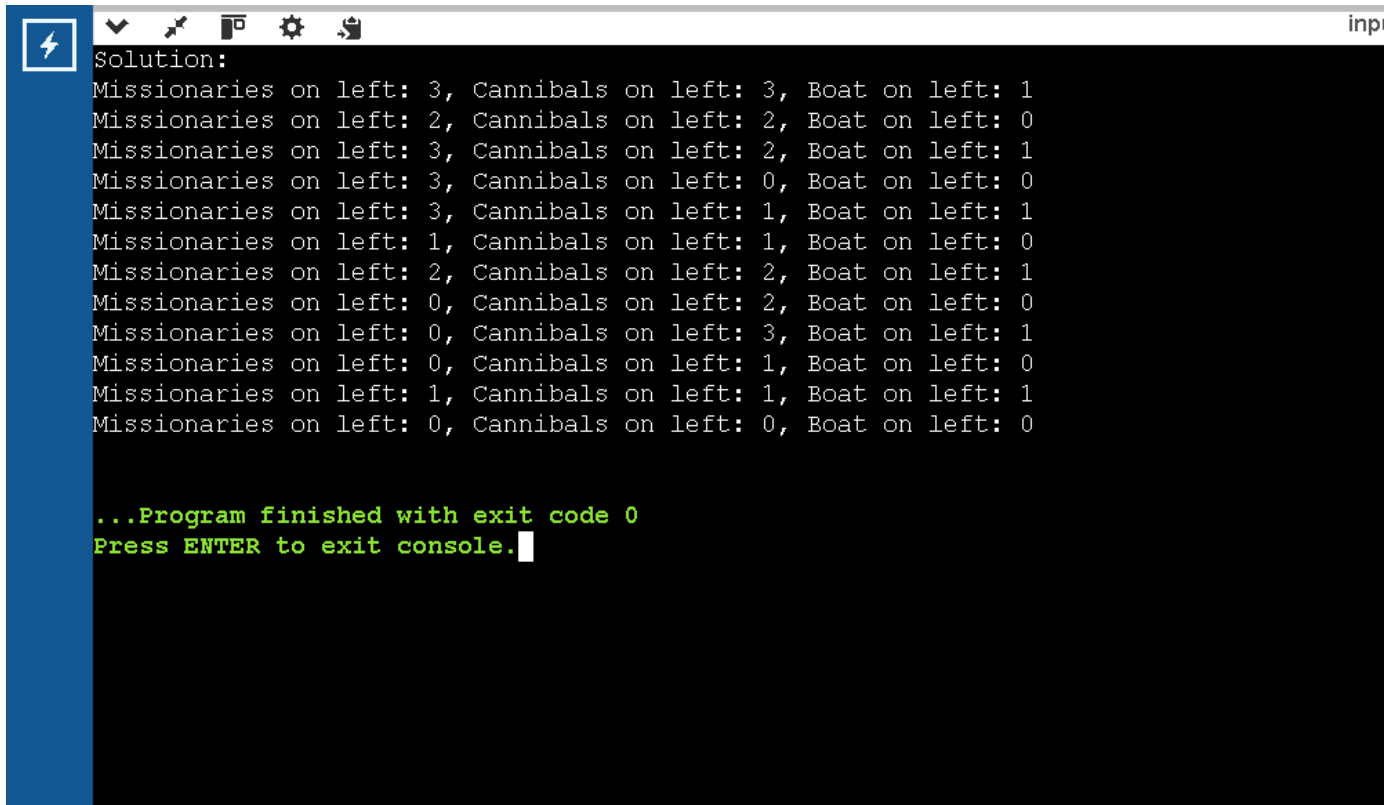
        for next_state in get_next_states(*current_state):
            if next_state not in visited:
```

```
        visited.add(next_state)
        queue.append((next_state, path + [current_state]))

    return None # No solution found

solution = solve_missionaries_cannibals()
if solution:
    print("Solution:")
    for step in solution:
        print(f"Missionaries on left: {step[0]}, Cannibals on left: {step[1]}, Boat on left: {step[2]}")
else:
    print("No solution found.")
```

OUTPUT:



```
Solution:
Missionaries on left: 3, Cannibals on left: 3, Boat on left: 1
Missionaries on left: 2, Cannibals on left: 2, Boat on left: 0
Missionaries on left: 3, Cannibals on left: 2, Boat on left: 1
Missionaries on left: 3, Cannibals on left: 0, Boat on left: 0
Missionaries on left: 3, Cannibals on left: 1, Boat on left: 1
Missionaries on left: 1, Cannibals on left: 1, Boat on left: 0
Missionaries on left: 2, Cannibals on left: 2, Boat on left: 1
Missionaries on left: 0, Cannibals on left: 2, Boat on left: 0
Missionaries on left: 0, Cannibals on left: 3, Boat on left: 1
Missionaries on left: 0, Cannibals on left: 1, Boat on left: 0
Missionaries on left: 1, Cannibals on left: 1, Boat on left: 1
Missionaries on left: 0, Cannibals on left: 0, Boat on left: 0

...Program finished with exit code 0
Press ENTER to exit console.
```

CONCLUSION

In conclusion, the Missionaries and Cannibals problem demonstrates fundamental concepts in artificial intelligence, including state-space representation, constraint satisfaction, and search algorithms. Solving this problem with Python and the Breadth-First Search (BFS) algorithm provides an effective approach to systematically explore all possible states, ensuring the solution is both valid and optimal. This implementation highlights how BFS helps navigate complex decision trees by focusing on shortest-path solutions, making it particularly valuable for constraint-laden problems like this one. Ultimately, the Missionaries and Cannibals problem serves as a foundational exercise in AI problem-solving, offering insights into algorithmic thinking, efficient state management, and constraint handling—skills crucial for tackling more complex AI challenges.