Water Jug Problem Solution Report

M.Junaid 109(4B)

Introduction

This report explains a solution to the Water Jug problem, a classic puzzle in artificial intelligence. The problem involves two jugs with different capacities, and the goal is to measure a specific amount of water using only these jugs and the operations of filling, emptying, and transferring water between them.

Algorithm Explanation

Depth-First Search Approach

The solution uses a Depth-First Search (DFS) algorithm to explore the state space.

In this function wJugDFS():

- Each state is represented as a pair (j1, j2) where j1 is the amount of water in the first jug and j2 is the amount in the second jug
- The initial state is (0, 0), meaning both jugs are empty
- The algorithm uses a stack to keep track of states to explore
- It also maintains a set of visited states to avoid cycles
- For each state, it generates all possible next states based on the allowed operations
- The search continues until either the target amount is found in either jug or all possible states have been explored

```python
def wJugDFS(c1, c2, g):
    s= [(0, 0)]
    v= set()
    v.add((0, 0))
    a = []

    while s:
        j1, j2= s.pop()
        a.append((j1, j2))

        if j1== g or j2== g:
            print("Solution Found")
            for act in a:
                print(act)
            return

        r= [(c1, j2),
            (j1, c2),
            (0, j2),
            (j1, 0),
            (j1 - min(j1, c2 - j2), j2 + min(j1, c2 - j2)),
            (j1 + min(j2, c1 - j1), j2 - min(j2, c1 - j1)),
            (min(j1 + j2, c1), 0),
            (0, min(j1 + j2, c2))]

        for state in r:
            if state not in v:
                v.add(state)
                s.append(state)

    print("No Solution found")
    return
```

State Generation

For each state (j1, j2), the algorithm generates the following possible next states:

1. (c1, j2): Fill the first jug completely

2. (j1, c2): Fill the second jug completely

3. (0, j2): Empty the first jug

4. (j1, 0): Empty the second jug

5. (j1 - min(j1, c2 - j2), j2 + min(j1, c2 - j2)): Pour from first jug to second

6. (j1 + min(j2, c1 - j1), j2 - min(j2, c1 - j1)): Pour from second jug to first jug

7. (min(j1 + j2, c1), 0): Pour all water from second jug to first jug

8. (0, min(j1 + j2, c2)): Pour all water from first jug to second jug,

Main Function

The wJugDFS() function implements the DFS algorithm. It takes three parameters:

1. c1: Capacity of the first jug

2. c2: Capacity of the second jug

3. g: Target amount of water to measure

```
jug1Cap= int(input("Enter the capacity of jug 1: "))
jug2Cap= int(input("Enter the capacity of jug 2: "))
target= int(input("Enter the target amount: "))

wJugDFS(jug1Cap, jug2Cap, target)
```

Results

```
PS D:\4th Semester\PAI Lab\Assignment> & C:/Users/M.Junaid/AppData/Lo
g/waterJug.py"
Enter the capacity of jug 1: 6
Enter the capacity of jug 2: 4
Enter the target amount: 2
Solution Found
(0, 0)
(0, 4)
(4, 0)
(4, 4)
(6, 2)
PS D:\4th Semester\PAI Lab\Assignment>
```