The Superior University, Lahore

Assignment-I (Fall 2024)

Course Title:	PAI				Course Code:	CPR601260	Credit Hours: 4
Instructor:					Programme Name:	BSDS	
Semester:	4th	Batch:	F23	Section:	BSDSM-4A	Date:	30 January, 2024
Time Allowed:					Maximum Marks:		
Student's Name:	Fizza Farooq				Reg. No.	Su92-bsdsm-f23-017	

Lab-Task

1: Question
2: Question

TASK 3

WaterJug with DFS & printing rules (also correct the rule 5 & 6)

```
def waterjug_dfs(jug_1, jug_2, target):
    stack = [(0, 0)]
    visited = set()
    path = []

    while stack:
        jug1, jug2 = stack.pop()

        if (jug1, jug2) in visited:
            continue

        visited.add((jug1, jug2))
        path.append((jug1, jug2))

        if target in (jug1, jug2):
            print("Solution is found:", path)
            return path

        next_states = [
            (jug_1, jug2),
            (jug1, jug_2),
            (0, jug2),
            (jug1, 0),
            (max(0, jug1 - (jug_2 - jug2)), min(jug_2, jug1 + jug2)),
            (min(jug_1, jug1 + jug2), max(0, jug2 - (jug_1 - jug1)))
        ]
```

```
        for state in next_states:
            if state not in visited:
                stack.append(state)

    print("No solution is found")
    return None

print("Rules for filling the waterjug:")
print("1.Fill Jug 1 complete.")
print("2.Fill Jug 2 complete.")
print("3.Empty the Jug 1.")
print("4.Empty the Jug 2.")
print("5.Pour water from Jug 1 into Jug 2 until the Jug 2 is full or Jug 1 is empty.")
print("6.Pour water from Jug 2 into Jug 1 until Jug 1 is full or Jug 2 is empty.")

waterjug_dfs(4,3,2)
```

Python through this program implements a DFS algorithm to resolve Water Jug problems. The main objective entails determining the exact quantity of water (target) through the usage of jugs that possess fixed volume restrictions.

How the Code Works?

1. Initialize the Search

The waterjug_dfs(jug_1, jug_2, target) function begins its operation by filling the provided stack with both jugs empty ((0,0)).

The algorithm holds a collection of previously examined states for preventing recurrence of duplicate states.

2. Depth-First Search (DFS) Execution

The DFS operation of the program employs a stack which functions through LIFO principles.

The program saves current state entries in a visited states list to stop unnecessary computation.

The function creates every potential valid move through its operation.

- Fill Jug 1 completely: (jug_1, jug2)
- Fill Jug 2 completely: (jug1, jug_2)
- Empty Jug 1: (0, jug2)
- Empty Jug 2: (jug1, 0)

The water transfer starts when pouring Jug 1 into Jug 2 should keep flowing until Jug 2 reaches its maximum level or Jug 1 becomes empty.The following action will occur: Water will move from Jug 2 into Jug 1 as long as Jug 1 remains empty or Jug 2 becomes full.

3. Checking for the Solution

The function prints and returns the solution path while also completing the target amount in any jug.Upon exhausting every potential solution the function displays "No solution is found."

Why This Code Works?

DFS provides this function by going deep into possible moves until it needs to backtrack.The set function monitors which states have already been visited in order to stop infinite looping.The system uses a path list to let users track move sequences so they can easily follow the solution steps.

OUTPUT

```
Rules for filling waterjug:
1.Fill Jug 1 complete.
2.Fill Jug 2 complete.
3.Empty the Jug 1.
4.Empty the Jug 2.
5.Pour water from Jug 1 into Jug 2 until the Jug 2 is full or Jug 1 is empty.
6.Pour water from Jug 2 into Jug 1 until Jug 1 is full or Jug 2 is empty.
Solution found: [(0, 0), (0, 3), (3, 0), (3, 3), (4, 2)]

[(0, 0), (0, 3), (3, 0), (3, 3), (4, 2)]
```

waterjug_dfs(4,3,2)

Jug capacities:

Jug 1: 4 liters

Jug 2: 3 liters

Target: 2 liters

Step-by-Step Execution

(0, 0) → Start with empty jugs.

(4, 0) → Fill Jug 1.

Pumping from Jug 1 to Jug 2 results in the transfer of 3 liters which completes the operation.

(1, 0) → Empty Jug 2.

(0, 1) → Pour from Jug 1 into Jug 2 (1 liter moves).

(4, 1) → Fill Jug 1 again.

Following the process (2, 3) machines transfer two liters of liquid from Jug 1 to Jug 2 until Jug 1 remains with two liters. (Solution found!)

The solution ends with two liters inside one jug.

Why Use DFS Instead of BFS?

The DFS algorithm travels far into paths before it retraces its steps.The method requires a stack for successful implementation.An optimal solution is not necessary when running DFS because this search algorithm performs effectively. The traversal with BFS ensures the minimum path lengths in return for excess memory use.