```python
from collections import deque

def is_goal(state, goal):
    return state[0] == goal or state[1] == goal

def get_next_states(state, capacity):
    x, y = state
    max_x, max_y = capacity
    states = []
    states.append((max_x, y))
    states.append((x, max_y))
    states.append((0, y))
    states.append((x, 0))
    transfer = min(x, max_y - y)
    states.append((x - transfer, y + transfer))
    transfer = min(y, max_x - x)
    states.append((x + transfer, y - transfer))
    return states

def water_jug_bfs(capacity, start, goal):
    queue = deque()
    visited = set()
    queue.append((start, [start]))
    visited.add(start)
    while queue:
        current_state, path = queue.popleft()
        if is_goal(current_state, goal):
            return path
        for next_state in get_next_states(current_state, capacity):
            if next_state not in visited:
                visited.add(next_state)
                queue.append((next_state, path + [next_state]))
    return None

```

```
capacity = (4, 3)
start = (0, 0)
goal = 2
solution = water_jug_bfs(capacity, start, goal)

if solution:
    print("Steps to reach the goal:")
    for step, state in enumerate(solution):
        print(f"Step {step}: Jug X = {state[0]}, Jug Y = {state[1]}")
else:
    print("No solution found.")

```

```
>>> %Run -c $EDITOR_CONTENT

  Steps to reach the goal:
  Step 0: Jug X = 0, Jug Y = 0
  Step 1: Jug X = 0, Jug Y = 3
  Step 2: Jug X = 3, Jug Y = 0
  Step 3: Jug X = 3, Jug Y = 3
  Step 4: Jug X = 4, Jug Y = 2

>>>
```