**Mahavir Education Trust's**

**Shah & Anchor Kutchhi Engineering College,**

**Chembur, Mumbai 400 088**

**UG Program in Information Technology**

**Name:** GARIMA MAHENDRA MAHTO

**Class-Batch:** BE\_5A

**Roll No:** 07

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **EXPERIMENT\_NO.04** | | | | | |
| **Date of Performance:** | 26/08/2021 | | | | |
| **Date of Submission:** | 09/09/2021 | | | | |
| **Program formation/ Execution/ ethical practices (07)** | **Documentation (02)** | **Timely Submission (03)** | **Viva Answer**  **(03)** | **Experiment Marks**  **(15)** | **Teacher Signature**  **with date** |
|  |  |  |  |  |  |

**EXPERIMENT\_NO.04**

**AIM:** Document problem definitionand formulate it with statespace method: (Any two from suggested list)

●Water jug problem,

* Tower of Hanoi,
* 8 Tile Problem
* Vacuum Cleaner problem
* Missionaries and Cannibal
* Romania Map Problem
* Wumpus world problem

**LAB OUTCOME:** 7. ITL703.2 Analyse and formalize the problem as a state space, graph, design heuristics and select amongst different search or game-based techniques to solve them.

**THEORY:**

**State Space Method:**

State space search is a process used in the field of computer science, including artificial intelligence (AI), in which successive configurations or states of an instance are considered, with the intention of finding a goal state with a desired property.

Problems are often modelled as a state space, a set of states that a problem can be in. The set of states forms a graph where two states are connected if there is an operation that can be performed to transform the first state into the second.

State space search often differs from traditional computer science search methods because the state space is implicit: the typical state space graph is much too large to generate and store in memory. Instead, nodes are generated as they are explored, and typically discarded thereafter. A solution to a combinatorial search instance may consist of the goal state itself, or of a path from some initial state to the goal state.

**Water Jug Problem:**

**Problem:** You are given two jugs, a 4-gallon one and a 3-gallon one. Neither has any measuring mark on it. There is a pump that can be used to fill the jugs with water. How can you get exactly 2 gallons of water into the 4-gallon jug?

**Solution:**

The state space for this problem can be described as the set of ordered pairs of integers (x, y)

Were,

X represents the quantity of water in the 4-gallon jug X= 0,1,2,3,4

Y represents the quantity of water in 3-gallon jug Y=0,1,2,3

Start State: (0,0)

Goal State: (2,0)

Generate production rules for the water jug problem

Production Rules:

|  |  |  |
| --- | --- | --- |
| Rule | State | Process |
| 1 | (X, Y | X<4) | (4, Y)  {Fill 4-gallon jug} |
| 2 | (X, Y |Y<3) | (X,3)  {Fill 3-gallon jug} |
| 3 | (X, Y |X>0) | (0, Y)  {Empty 4-gallon jug} |
| 4 | (X, Y | Y>0) | (X,0)  {Empty 3-gallon jug} |
| 5 | (X, Y | X+Y>=4 ^ Y>0) | (4, Y-(4-X))  {Pour water from 3-gallon jug into 4-gallon jug until 4-gallon jug is full} |
| 6 | (X, Y | X+Y>=3 ^X>0) | (X-(3-Y),3)  {Pour water from 4-gallon jug into 3-gallon jug until 3-gallon jug is full} |
| 7 | (X, Y | X+Y<=4 ^Y>0) | (X+Y,0)  {Pour all water from 3-gallon jug into 4-gallon jug} |
| 8 | (X, Y | X+Y <=3^ X>0 | (0, X+Y)  {Pour all water from 4-gallon jug into 3-gallon jug} |
| 9 | (0,2) | (2,0)  {Pour 2-gallon water from 3-gallon jug into 4-gallon jug} |

Initialization:

Start State: (0,0)

Apply Rule 2:

(X, Y | Y<3) ->

(X,3)

{Fill 3-gallon jug}

Now the state is (X,3)

Iteration 1:

Current State: (X,3)

Apply Rule 7:

(X, Y | X+Y<=4 ^Y>0)

(X+Y,0)

{Pour all water from 3-gallon jug into 4-gallon jug}

Now the state is (3,0)

Iteration 2:

Current State: (3,0)

Apply Rule 2:

(X, Y | Y<3) ->

(3,3)

{Fill 3-gallon jug}

Now the state is (3,3)

Iteration 3:

Current State:(3,3)

Apply Rule 5:

(X, Y | X+Y>=4 ^ Y>0)

(4, Y-(4-X))

{Pour water from 3-gallon jug into 4-gallon jug until 4-gallon jug is full}

Now the state is (4,2)

Iteration 4:

Current State: (4,2)

Apply Rule 3:

(X, Y | X>0)

(0, Y)

{Empty 4-gallon jug}

Now state is (0,2)

Iteration 5:

Current State: (0,2)

Apply Rule 9:

(0,2)

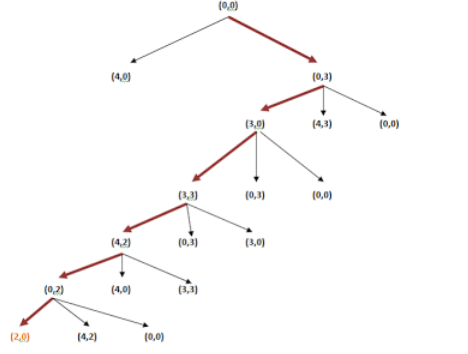
(2,0)

{Pour 2-gallon water from 3-gallon jug into 4-gallon jug}

Now the state is (2,0)

Goal Achieved.

State Space Tree:



**Tower of Hanoi problem:**

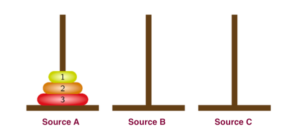
This is a fun puzzle game where the objective is to move an entire stack of disks from the source position to another position. Three simple rules are followed:

1) Only one disk can be moved at a time.

2) Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack. In other words, a disk can only be moved if it is the uppermost disk on a stack.

3)No larger disk may be placed on top of a smaller disk.

Now, let’s try to imagine a scenario. Suppose we have a stack of three disks. Our job is to move this stack from source A to destination C.



We can use B as a helper to finish this job. We are now ready to move on. Let’s go through each of the steps:

1) Move the first disk from A to C

2) Move the first disk from A to B

3)Move the first disk from C to B

4) Move the first disk from A to C

5) Move the first disk from B to A

6) Move the first disk from B to C

7) Move the first disk from A to C



You can see the animated image above for a better understanding

|  |  |  |
| --- | --- | --- |
| Problem characteristic | Satisfied | Satisfied |
| Is the problem decomposable? | No | One game has Single solution |
| Can solution steps be ignored or undone? | Yes | We can undo the previous move |
| Is the problem universe predictable? | Yes | We can predict about the solution. |
| Is a good solution absolute or relative? | absolute | **Absolute solution**: once you get one solution you do need to bother about other possible solution. **Relative Solution**: once you get one solution you have to find another possible solution to check which solution is best |
| Is the solution a state or a path? | Path | a path to a state = you have perfect rules for problem, no need to worry about logic of words. |
| What is the role of knowledge? | Need Knowledge | lot of knowledge helps to constrain the search for a solution. |
| Do the tasks require interaction with a person/human? | No | Conversational In which there is intermediate communication between a person and the computer, either to provide additional assistance to the computer or to provide additional information to the user, or both. In tower of Hanoi additional assistance is not required. |

**CONCLUSION**: Numerous source codes have been devised for solving Water Jug problems using recursion, searching and sorting algorithms. The solution written using Breadth-First Search is considered to be one of the most optimum solutions

With 3 disks, the puzzle can be solved in 7 moves. The minimal number of moves required to solve a Tower of Hanoi puzzle is 2n − 1, where n is the number of disks.