Search Problem Example + Complexity of IDDFS

Dr. Soumajit Pramanik

Two Jug Problem

There is a large bucket B full of water and Two (02) jugs, J1 of volume 3 litre and J2 of volume 5 litre.

You are allowed to fill up any empty jug from the bucket, pour all water back to the bucket from a jug or pour from one jug to another.

The goal is to have jug J1 with exactly one (01) litre of water.

Source: https://cse.iitkgp.ac.in/~pallab/course/2022/autumn%202022/Artificial%20Intelligence%20autumn%202022/index.html

Two Jug Problem

State Definition: <J1, J2>

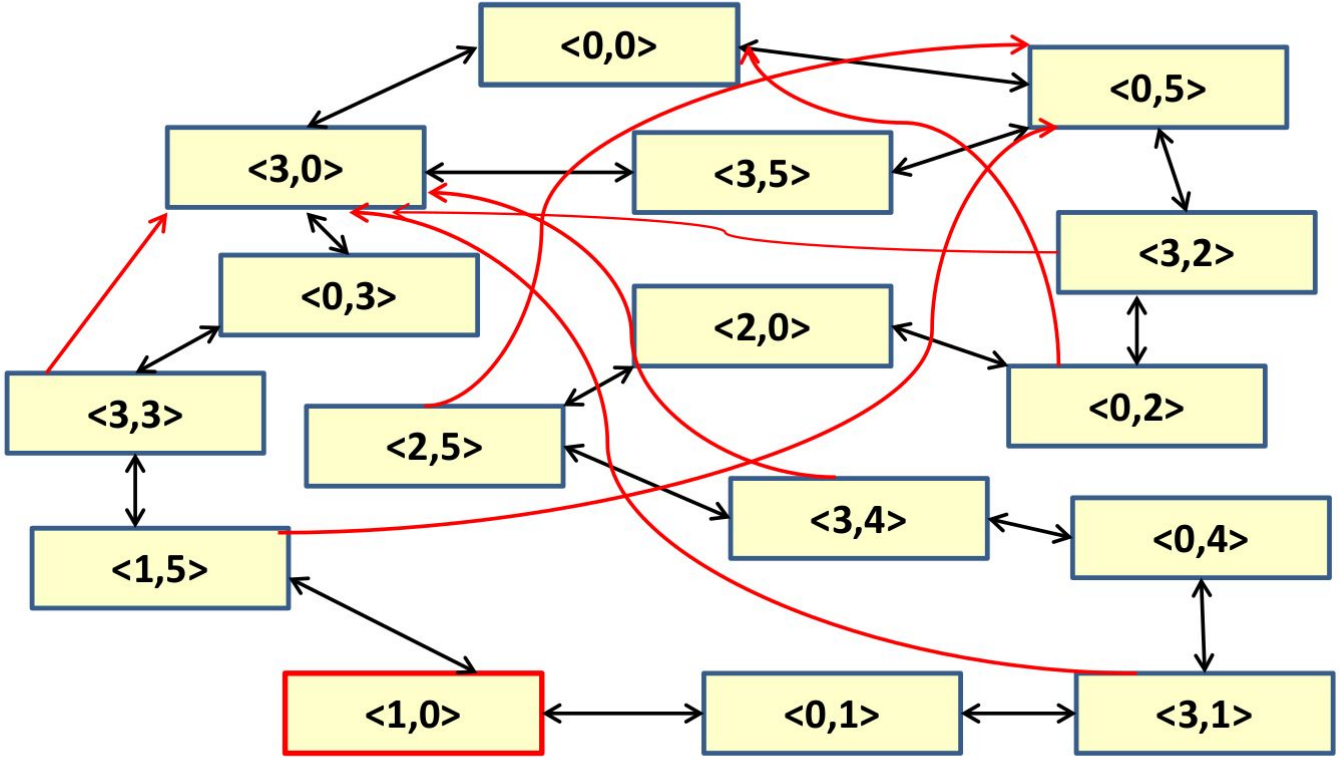
Start: <0,0>, Goal: <1,0>

Rules:

1. Fill (J1): <J1, J2> to <3,J2>
2. Fill (J2): <J1, J2> to <J1, 5>
3. Empty (J1), Empty (J2): Similarly defined
4. Pour (J1, J2): <J1, J2> to <X,Y>, where
    a. X = 0 and Y = J1 + J2 if J1+J2 ≤ 5,
    b. Y = 5 and X = (J1+ J2) - 5, if J1+J2 > 5
5. Pour (J2, J1): Similarly defined

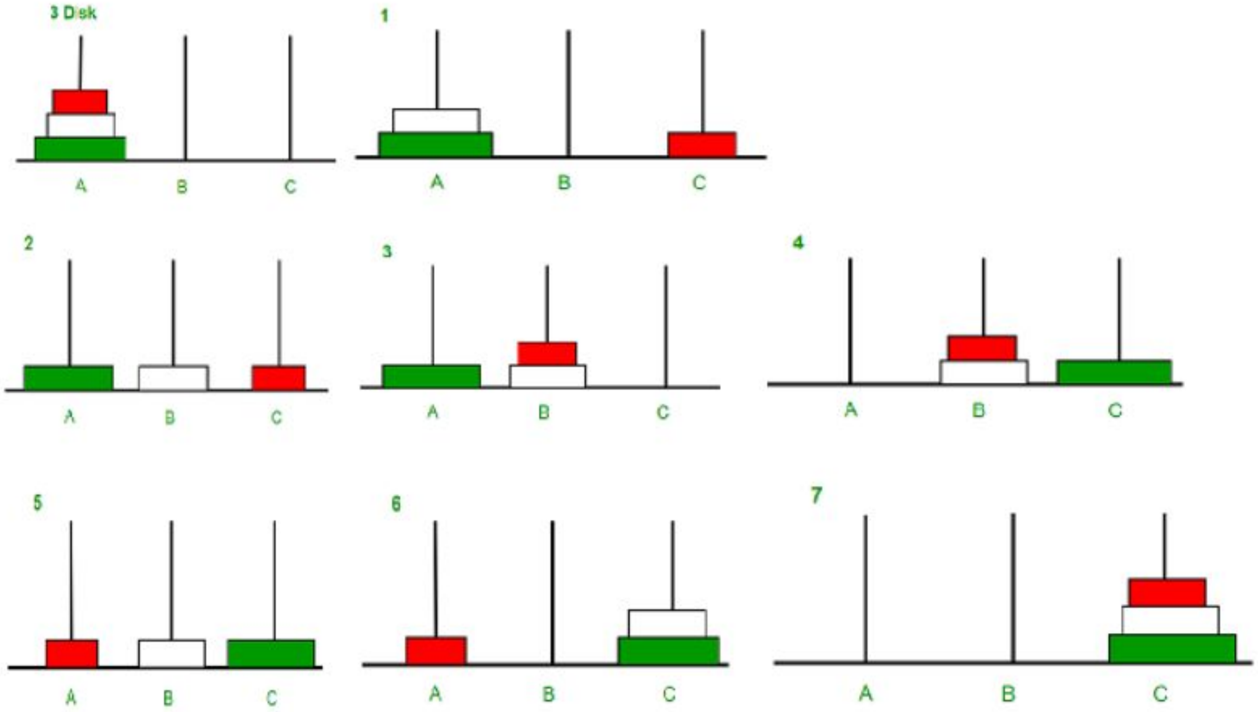
Source: https://cse.iitkgp.ac.in/~pallab/course/2022/autumn%202022/Artificial%20Intelligence%20autumn%202022/index.html

Two Jug Problem - Partial State Space



Source: https://cse.iitkgp.ac.in/~pallab/course/2022/autumn%202022/Artificial%20Intelligence%20autumn%202022/index.html

Tower of Hanoi



3 Discs

3 poles

Constraint:

Cannot put a

larger disc on

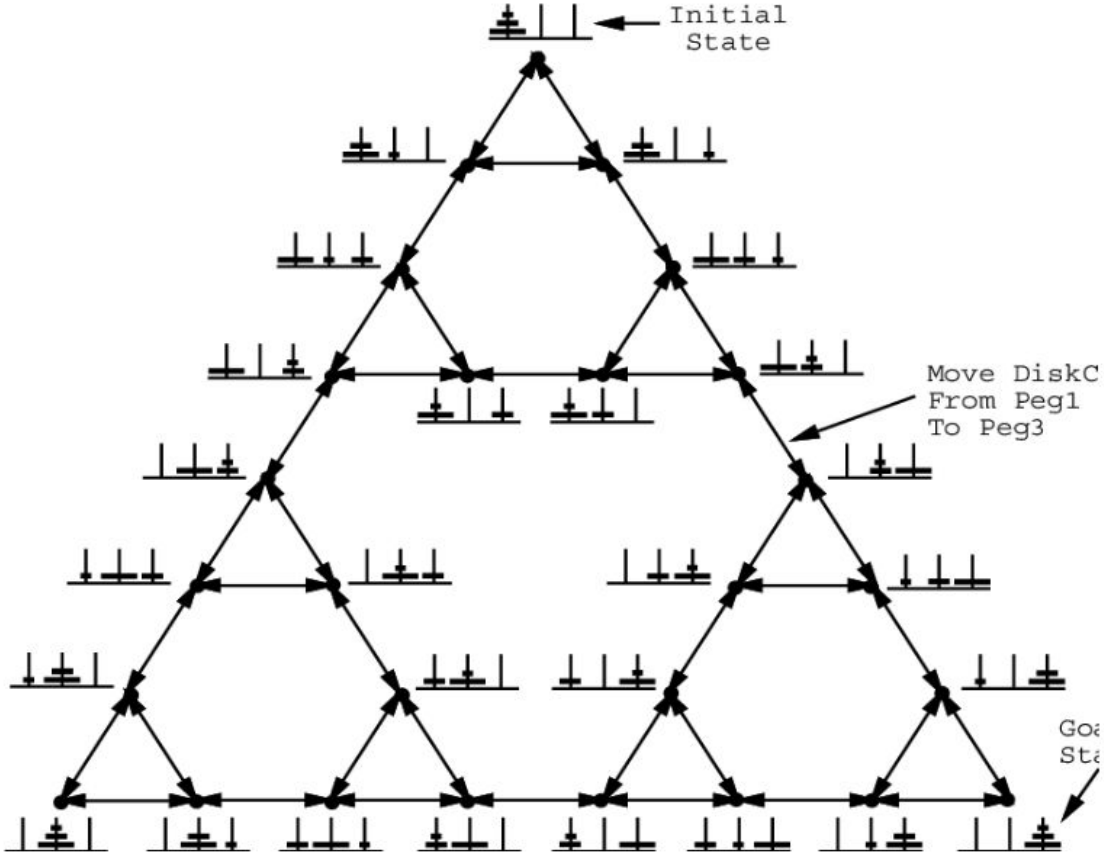
a smaller one.

Rules:

Move top disc from one pole to top position of another.

Source: https://cse.iitkgp.ac.in/~pallab/course/2022/autumn%202022/Artificial%20Intelligence%20autumn%202022/index.html

Tower of Hanoi



Source: https://cse.iitkgp.ac.in/~pallab/course/2022/autumn%202022/Artificial%20Intelligence%20autumn%202022/index.html

IDDFS

Time Complexity

In an iterative deepening search, the nodes at depth d are expanded once, those at depth $d-1$ are expanded twice, and so on up to the root of the search tree, which is expanded $d+1$ times.[1]:5 So the total number of expansions in an iterative deepening search is

$$b^d + 2b^{d-1} + 3b^{d-2} + \cdots + (d-1)b^2 + db + (d+1) = \sum_{i=0}^{d} (d+1-i)b^i$$

where b^d is the number of expansions at depth d, $2b^{d-1}$ is the number of expansions at depth $d-1$, and so on. Factoring out b^d gives

$$b^d(1 + 2b^{-1} + 3b^{-2} + \cdots + (d-1)b^{2-d} + db^{1-d} + (d+1)b^{-d})$$

Now let $x = \frac{1}{b} = b^{-1}$. Then we have

$$b^d(1 + 2x + 3x^2 + \cdots + (d-1)x^{d-2} + dx^{d-1} + (d+1)x^d)$$

This is less than the infinite series

$$b^d(1 + 2x + 3x^2 + 4x^3 + \cdots) = b^d\left(\sum_{n=1}^{\infty} nx^{n-1}\right)$$

which converges to

$$b^d(1-x)^{-2} = b^d\frac{1}{(1-x)^2}, \text{ for } abs(x) < 1$$

That is, we have

$$b^d(1 + 2x + 3x^2 + \cdots + (d-1)x^{d-2} + dx^{d-1} + (d+1)x^d) \leq b^d(1-x)^{-2}, \text{ for } abs(x) < 1$$

Since $(1-x)^{-2}$ or $\left(1-\frac{1}{b}\right)^{-2}$ is a constant independent of d (the depth), if $b > 1$ (i.e., if the branching factor is greater than 1), the running time of the depth-first iterative deepening search is $O(b^d)$.

Source: wikipedia