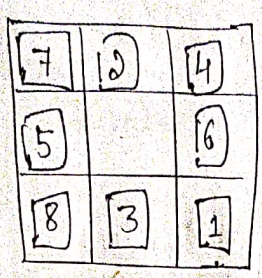
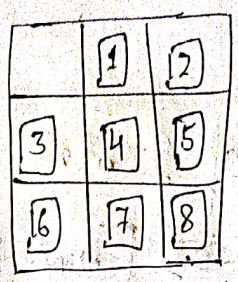
Search

Ex: 8-puzzle Problem



Start State



Goal State

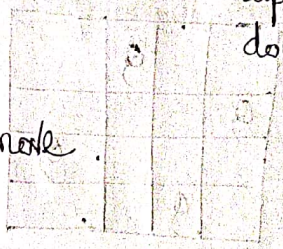
States : Locations tiles

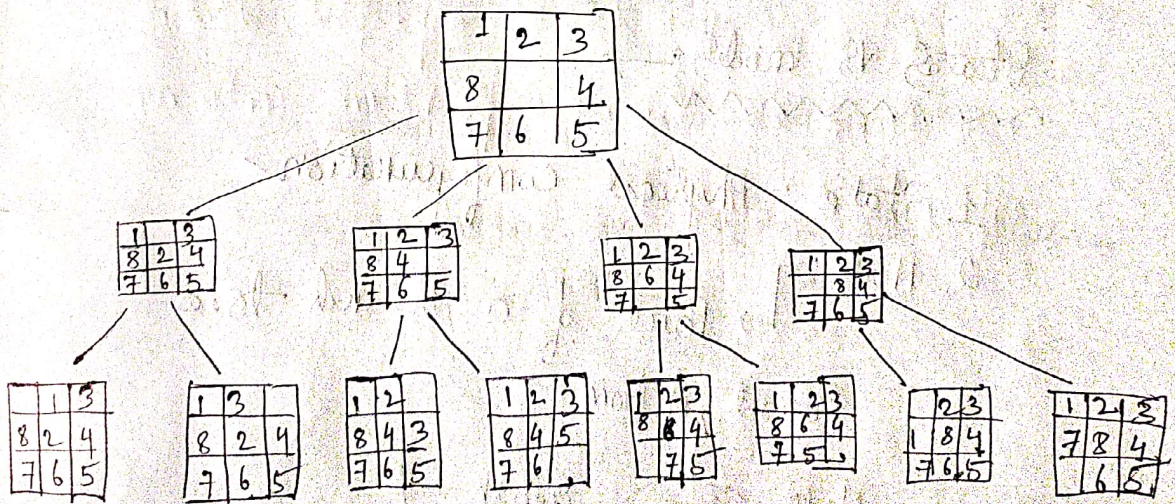
Actions : Movement of the blank (left, Right, up, down)

Goal state → Goal test :

Path Cost :

↳ 1 per move





Search - Tree Example

8 - Puzzle - Problem Space

Example:

N-Queens Problem:

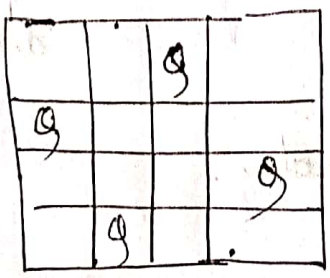
Input:

Set of States

Operators [and costs]

Start State

Goal State (test)



States vs nodes: →

1. State: Physical configuration
2. Node: ↳ Part of a search tree

Parent, Action:

Node

depth = 7

State.

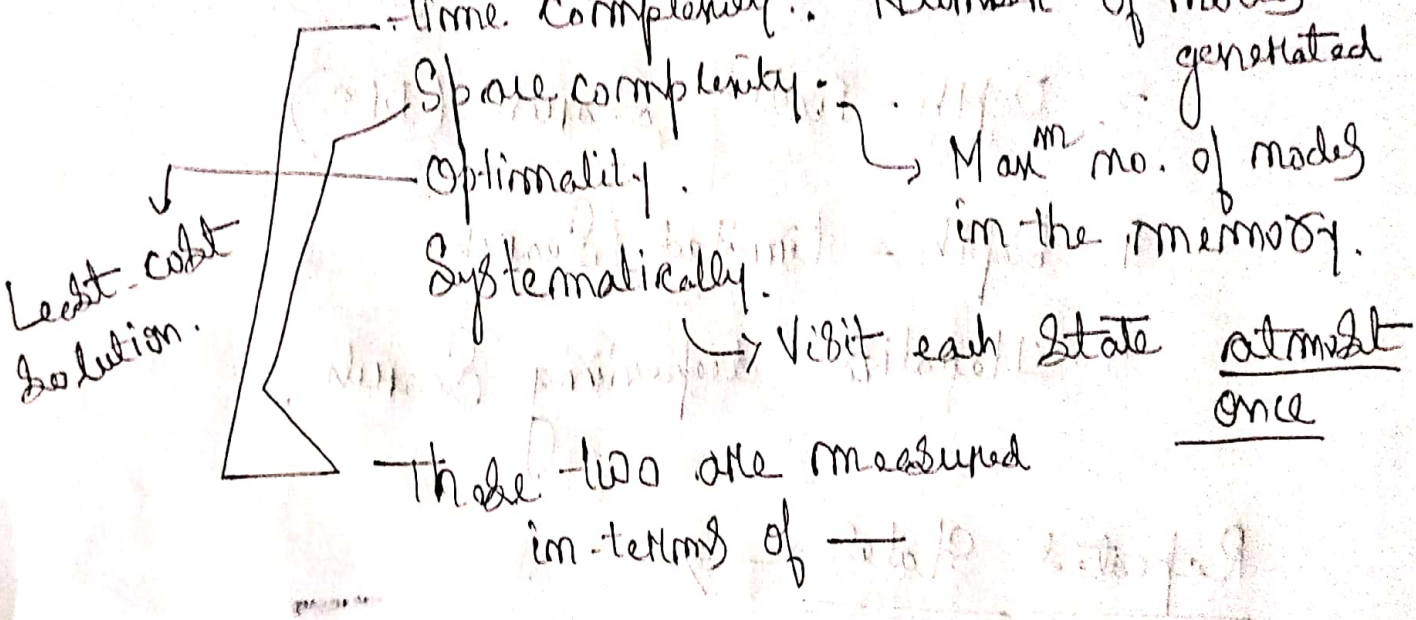
5	4	
6	1	8
7	3	2

g = 6 (Path cost)

Search Strategies

1. Search Strategies are based on <u>order of node expansion</u>

2. Evaluation of the Strategies :→



→ b : Max^m branching Factor of the search tree.

→ d : Depth of the least cost solution.

→ m : Maximum depth of the state space (∞)

↓

Infinity.

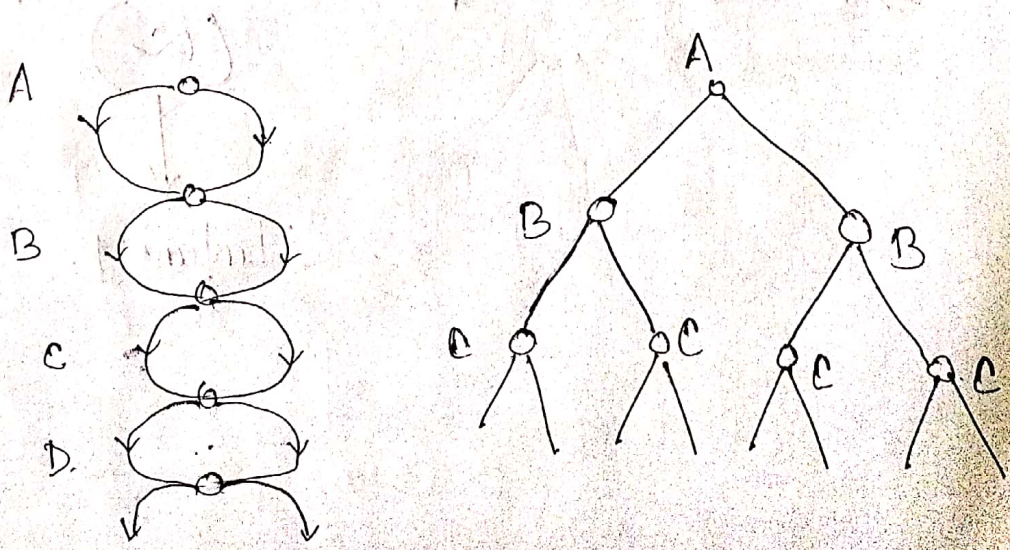
Types of uninformed Search Strategy

1. Uninformed Search Strategies use only the information available in the problem definition.

2. Breadth first Search (BFS)
3. Depth - first Search (DFS)
4. Depth – Limited Search
5. Iterative deepening Search.

Repeated States :→

Failure to detect repeated states can turn a linear-problem into an exponential one.!

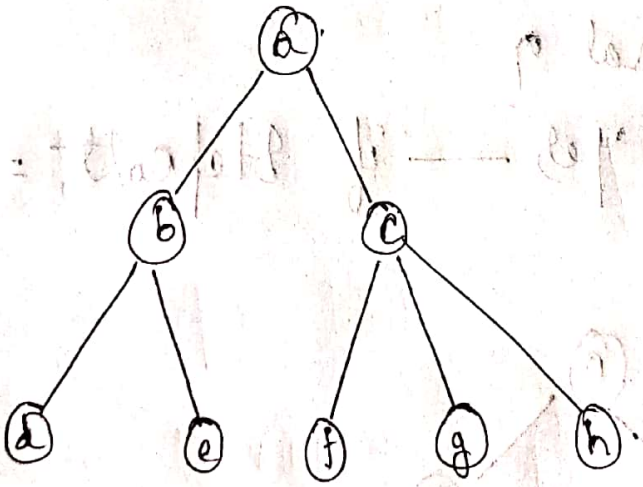


Depth first Search (DFS)

1. Data-Structure used : Stack

2. Evaluation:

- Complete ? (No)
- Time Complexity
  $O(b^m)$
- Space Complexity
  $O(bm)$



Breadth First Search (BFS)

(Shortest First)

1. Data Structure used : queue

2. Evaluation : →

- Complete ? yes (b is finite)
- Time Complexity ?

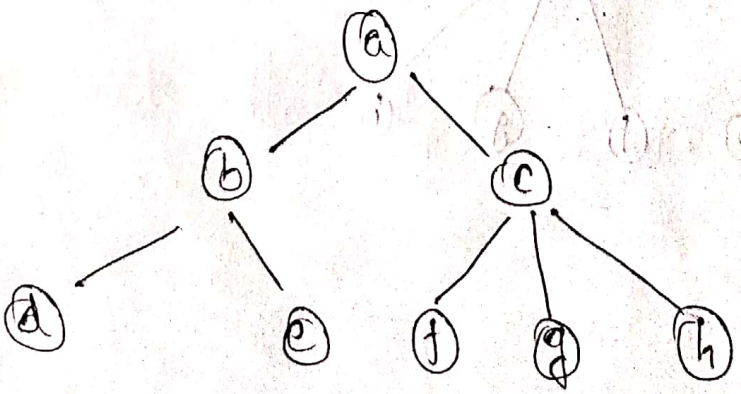
$$O(b^d)$$

- Space Complexity

$$O(b^d)$$

- Optimal ?

yes — if Stepcost = 1



Uniform Cost Search: Cheapest First

1. Data-Structure: Queue
2. Evaluation —

- Complete? yes (b is finite)
- Time Complexity?

$$O\left(b^{(C^*/\epsilon)}\right)$$

- Space Complexity

$$O\left(b^{(C^*/\epsilon)}\right)$$

- Optimal?

yes