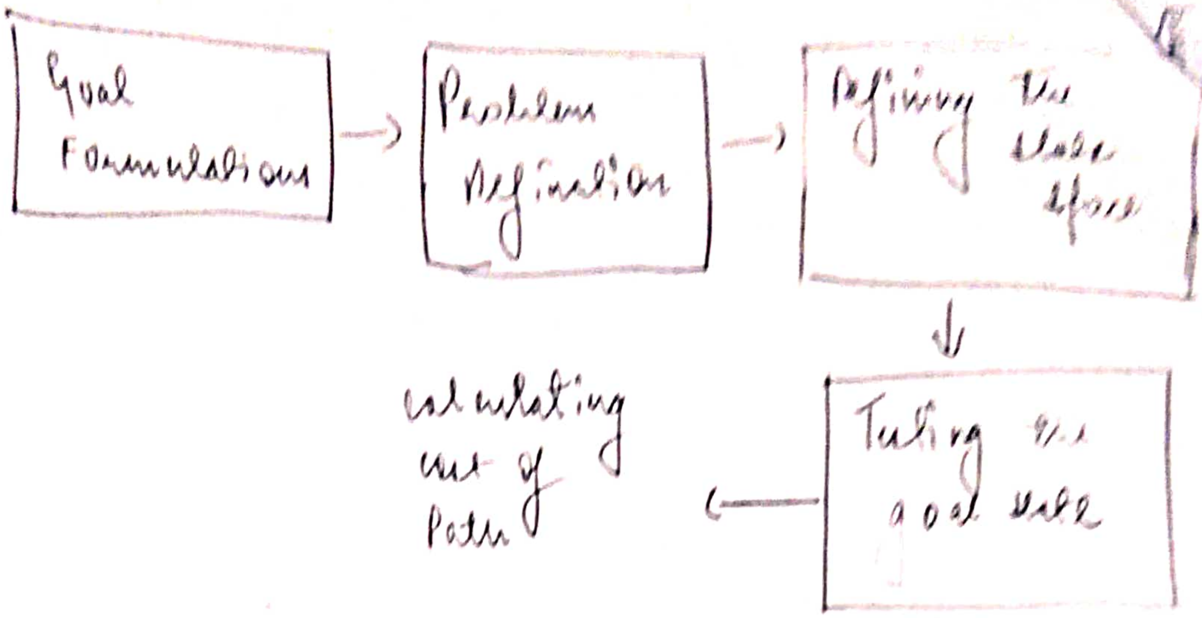
POORNIMA
COLLEGE OF ENGINEERIN

DETAILED LECTURE NOTES

The AI Problems

AI Problem solving process

- AI aims at making such systems which are capable of solving various problems on its own.
- It is process where AI agent tries to reach a goal with an intention to find the solution to a certain problem.
- When a new problem is solved by the AI Process, the AI agent maps the problem states into action.
- If the Problem is too large and cannot be mapped directly by the Agent, then the Agent splits the problem domain into Subproblems.
- It now solves the sub problem seperately and then it clicks the results so obtained in order to get to the solution for the whole problem.
- Problem solving agents are goal Based agents which work with an aim to reach the goal with an apt solution for the problem.



1) Goal formulation : In this step as soon as problem arises, the agent sets a goal or target. This requires the agent to promptly analyze and define the problem. This is the crucial step as if the goal for the problem is wrongly formulated then all the steps taken in order to reach the goal would be of no use.

2) Problem Definition : This is the 1 main step of Problem solving. It is here that whenever a problem arises, then the agent decides on what actions must be taken so as to reach till the formulated goal. This is done in following steps:

• Defining the state space A state space can be defined as a collection of all Valid states in which an agent can be in when finding a solution to the Problem.

→ Defining initial state In order for an agent to start solving the problem, it needs to start from a state. The First state from where the agent starts working is referred to as the initial state.

→ Gather Knowledge: Now the agent gathers information and uses the information required by it to solve the problem. This knowledge will be gathered with past Experiences as well as current learnings.

→ Planning the Transitions: Some Problems are small and so these can be solved easily. But most of the times problems will be such where proper planning and Execution is needed. Hence this requires proper data structures and control strategies well in advance.

Here the Problem is divided into sub problems. The Results of the Various actions taken in solving the previous sub problem are forwarded to the next sub problem and the combined effect of the sub problem leads to the final solution.

This requires proper planning and execution of transitions.

3) Testing the Goal state In this step, the Results yielded from the Agent are compared with that of the goal state. If the goal has been reached, then the agents stop any further Actions and the problem reaches the last state. But if the goal is not reached then the Agent continues to find actions to reach till the goal

4) Calculate the Cost of Path Taken - Whenever an agent takes a path in order to solve a problem it allots a Numeric Value (or cost) to that path These Values are then Evaluated using a cost function. The Evaluated result is hence used in the agents performance measure. The solution which is reached with the minimum or lowest cost of path is termed as an Optimal solution

POORNIMA
COLLEGE OF ENGINEERIN

DETAILED LECTURE NOTES

- Much of the early work in the field of AI focused on formal tasks, such as game playing and Theorem proving.

- Game playing and Theorem proving share the property that people who do them well are considered to be displaying intelligence.

- Initially computers could perform well at those tasks simply by being fast at exploring a large no. of solution paths and then selecting the Best one.

- Humans learn mundane (Ordinary) tasks since their birth. They learn by Perception, speaking, using language, and training. They learn formal tasks and Expert tasks later.

→ Mundane tasks
→ Formal tasks
→ Expert tasks

} - PPT

Problem solving by searching

Searching → Finding

- Finding a solution

-> In a website finding the answer of question.

-> In a city finding the correct path

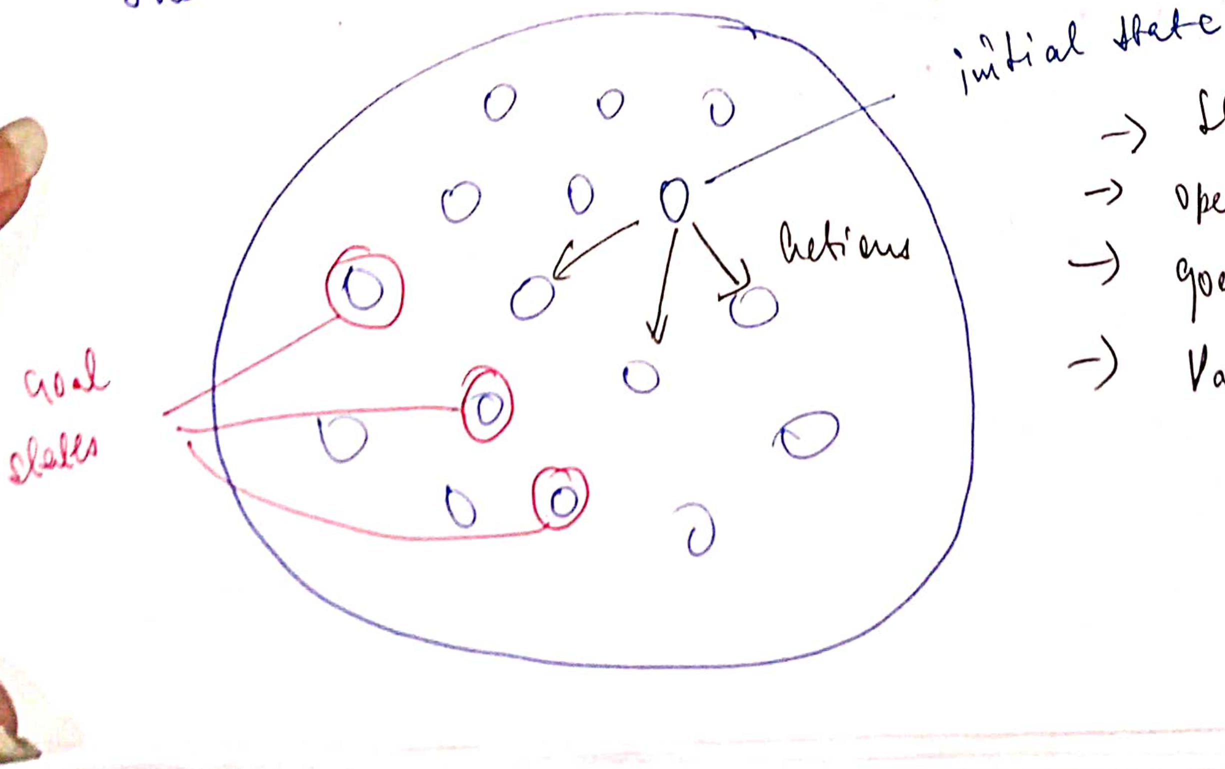
- In chess finding the best step to move

In terms of AI searching is the process of considering various possible sequences of operators applied to the initial state, and finding out a sequence which culminates in a goal state

How search works?

• A search problem is represented using a directed graph.

- The states are represented as Nodes.
- The allowed actions are represented as arcs.



-> States

-> Operators / Actions

-> Goal test

-> Path test

POORNIMA
COLLEGE OF ENGINEERIN

DETAILED LECTURE NOTES

Search Algorithm Terminologies :

- Search : Searching is a step by step procedure to solve a search problem in a given search space.
  a) Search Space: It represents a set of possible solutions, which a system may have.
  b) Start State: It is a state where agent begins the search.
  c) Goal test: It is the function which observes the current state and returns whether the goal state is achieved or not.
- Actions: It gives the description of all available actions to the agents.
- Transition model: A description of what each action do.
- Path cost: Function which assigns a numeric cost to each path
- Solution: It is an action sequence which leads from the start Node to Goal Node.
- Optimal Solution: If a solution has the lowest c among all the solutions.

Search Algorithms

Uninformed / Blind Search

Informed Search

Uninformed / Blind Search

The Uninformed search does not contain any knowledge such as closeness, the location of the goal.

- It just includes the information about how to traverse the tree and how to identify leaf and goal nodes
- It applies in a way in which search tree is searched without any information about the search space like initial state operators and test for the goal, so it also called blind search.
- It examines each node of the tree until it achieves the goal node

Informed search

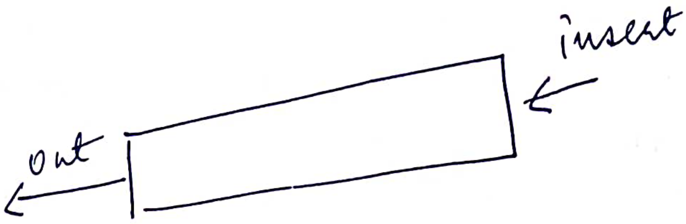
- Uses domain knowledge
- Some Problem Information is available which can guide on search
- Informed search strategies can find a solution more efficiently than an Uninformed search strategy.
- Informed search is also known as Heuristic search.

→ A Heuristic is a way which might not always be guaranteed for Best solutions but guaranteed to find a good solution in Reasonable time.

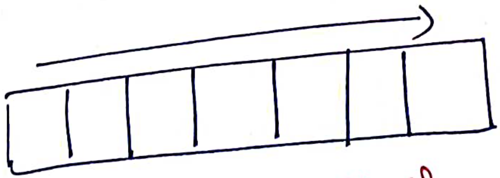
- It can solve much complex Problems which could not be solved in another way.

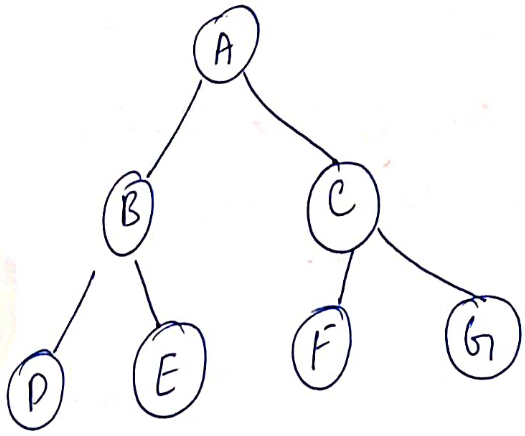
→ Breadth First Search (BFS)

- Uninformed search technique
- FIFO (Queue)
- Shallowest Node
- Complete
- Optimal
- Time Complexity

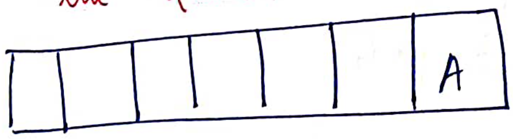


Step 1: Take an empty Queue

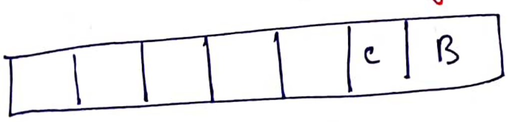




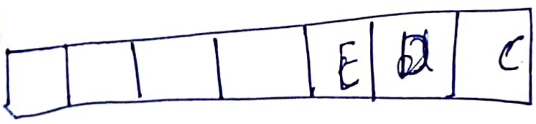
Step 2: Select a starting node (visiting a node) and insert it into the Queue.



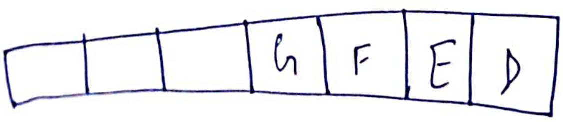
Step 3: Provided that the Queue is not empty, extract the node from the Queue and insert its child Nodes (Exploring a Node) into the Queue.



A (Print A & insert its child node into the Queue.



Print B & inserts its child Nodes into the queue



Print C & insert its child nodes into the Queue

POORNIMA
COLLEGE OF ENGINEERIN

DETAILED LECTURE NOTES

				G	F	E

Print D & insert its child nodes in to the Queue.

					G	F

Print E & insert its child nodes in to the Queue

						G

Print F & insert its child nodes in to the Queue

Algorithm of Breadth First Search

i) Enter ~~starting~~ Starting Nodes on Queue

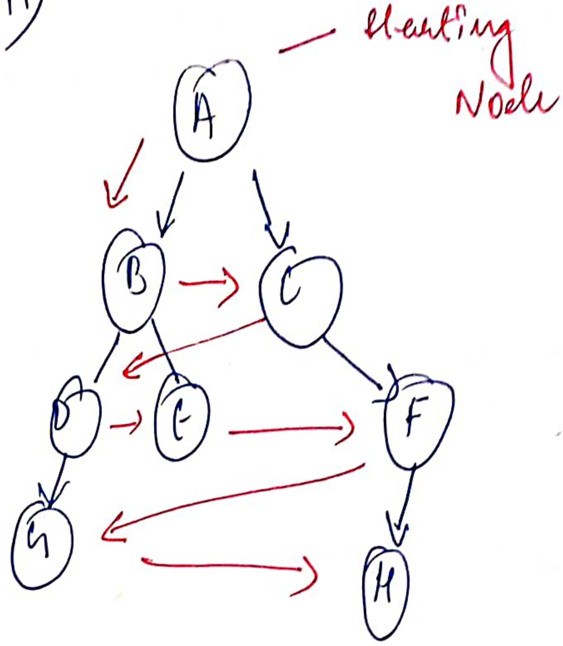
ii) If Queue is empty, then return fail and stop.

iii) If first element on queue is GOAL NODE, then return Success and Stop.

imp Else

iv) Remove and expand first element from Queue and place children at end of Queue.

v) Goto step ii)



A
↓
B
↓
C
↓
D
↓
E → F → G → H.

Initial Queue {A} = Goal Node. X - Proceed

↓

{B, C} — Remove from Queue and add its successor to Queue

Remove

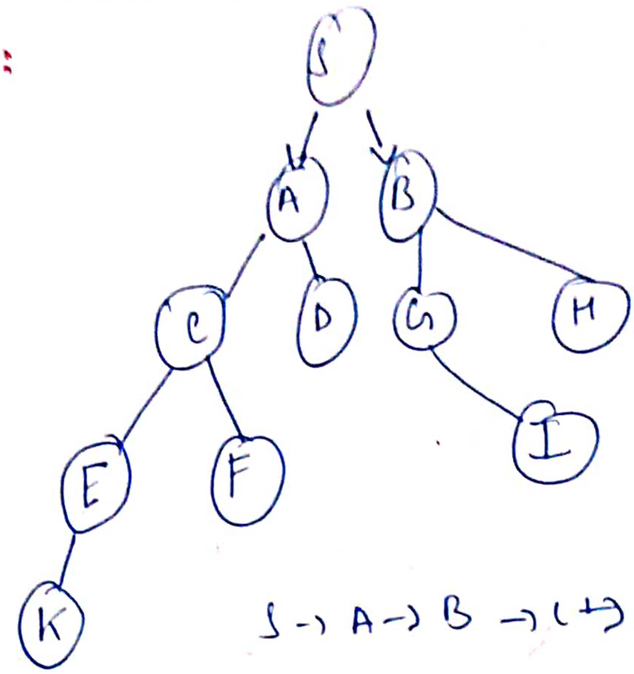
↓

{C, D, E}

↓

{D, E, F} → {E, F, G} → {E, F, G} → {G, H}

Example 2:



{S}
↓
{A, B}
{B, C, D}
{C, D, G, H}
{D, G, H, E, F}
{H, E, F, I}
{F, I, K}

S → A → B → C → D → G → H → E → F → I → K.

Advantages:

- It will provide a solution if any solution Exists.
- If there are more than 1 solutions for a given problem, then BFS will provide the minimal solution which requires the least No. of steps.

Disadvantages:

- It Requires lots of Memory since each level of the tree must be saved into memory to expand the Next level.
- BFS needs lots of time if the solution is far away from the Root Node.

Depth First Search (DFS) Algorithm

- Depth First Search (DFS) Algorithm starts with the initial Node of the Graph G, and then goes to deeper and deeper until we find the goal node or the node which has no children
- The Algorithm then Backtracks from the dead end towards the most recent node that is yet to be completely unexplored.
- The Data Structure which is being used in DFS is Stack.
- In DFS, the edges that leads to an Unvisited Node are called Discovery edges while the edges that leads to an already visited Node are called Block edges
- Recursive Algorithm.
- Starts from Root Node and follows each path to its greatest depth node before moving it to the Next path.
- Implemented using Stack.
- LIFO

Algorithm of DFS:

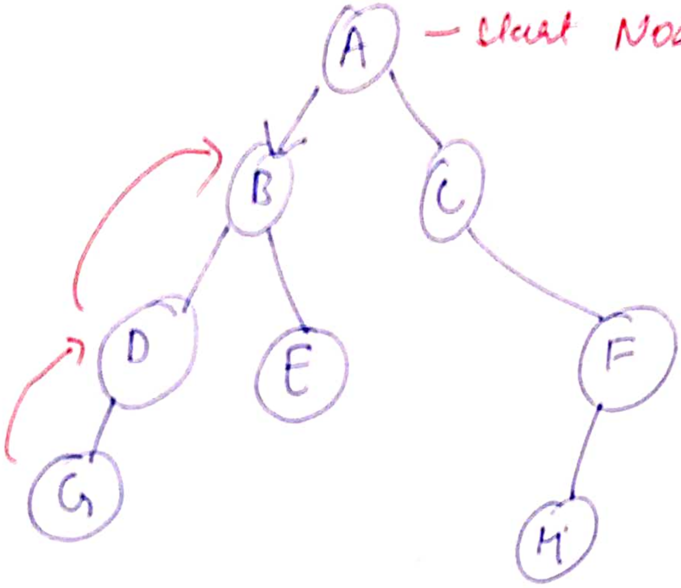
i) Insert Root Node in stack

ii) Do while stack is not empty

↳ a) Remove Node (Pop)

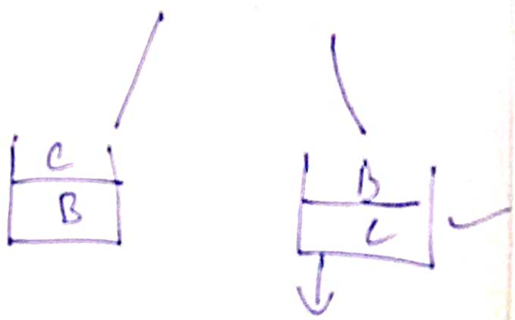
→ i) if Node = goal Node → stop

ii) Push all children of node in stack.

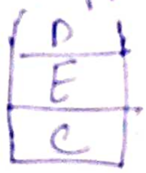


– start Node.

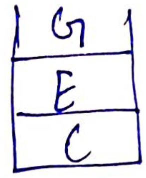
is A a goal node



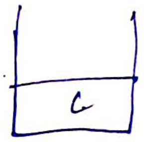
Here B is popped



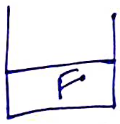
Here D will be popped.



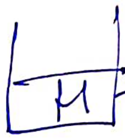
Pop G and E



Pop C



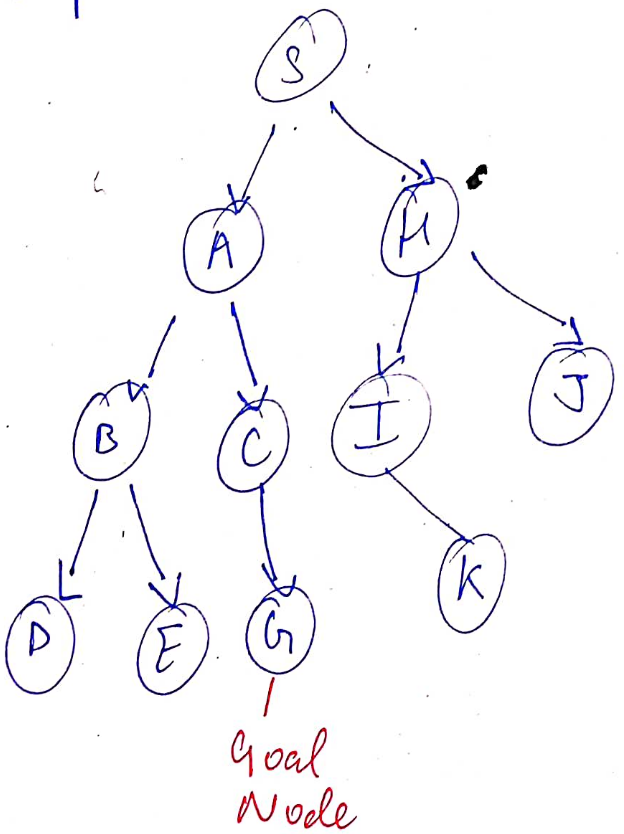
Pop F

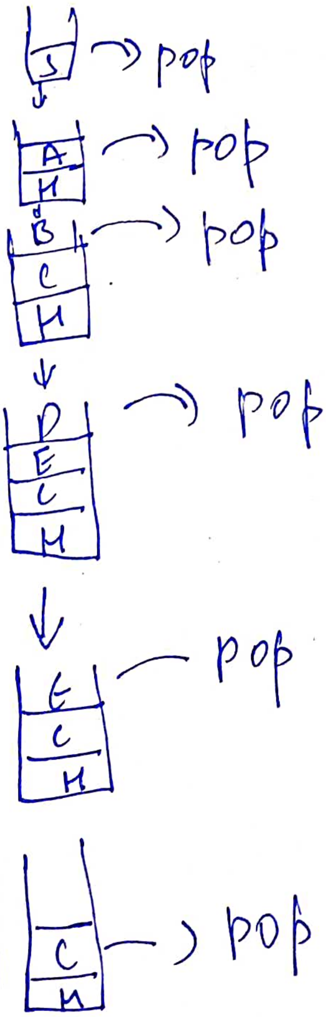


→ Pop.

A → B → D → G → E → C → F → H

Example - 2





G
H

→ G — Goal Node

↓

Time complexity $= O(b^d)$ → depth

b ↓ branching factor

Space complexity

↳ $O(bd)$.

S → A → B → D →

↳ C → G.

Advantages:

1) less memory
2) less time to reach Goal Node if traversal in Right path.

Disadvantages

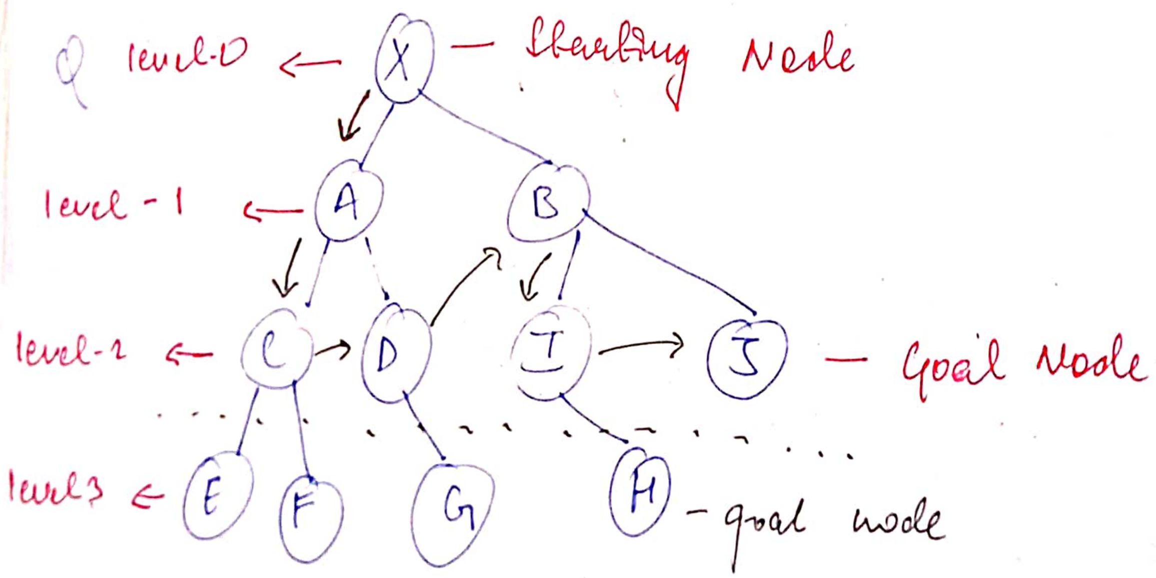
1) No guarantee of finding a solution
2) It can go infinite loop.

Depth - Limited Search

- Variant of depth first search
- A Depth - limited search algorithm is similar to depth - first search with a predetermined limit.
- Depth limited search can solve the problem of the infinite path in Depth - First search

Termination conditions

1) Failure value : there is no solution
2) Cut - off failure : Terminates on reaching predetermined depth } - no solution



→ here [depth, d = 2] X → A → C → D → B → I → J

• If H is your goal Node, then the [illegible] Depth, [illegible] will result in No solution

Advantages:

1) Memory Efficient.

Disadvantages

1) can be terminated without finding any solution. This is the situation of incompleteness.

2) Not Optimal

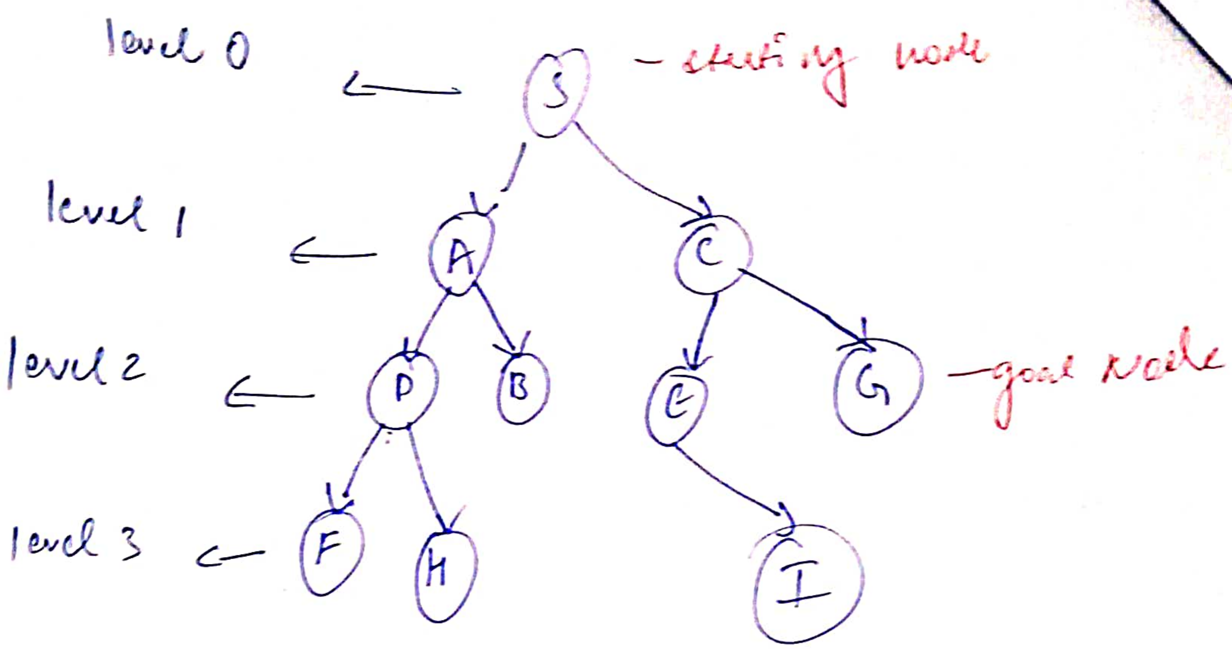
Time complexity: $O(b^d)$

Space complexity: $O(b \times d)$

Iterative Deepening Depth - First Search.

- It is a combination of DFS and BFS Algorithms.
- This search Algorithm finds out the Best depth limit and does it by gradually increasing the limit until a goal is found.
- This Algorithm performs Depth - first search up to a certain "Depth- limit", and it keeps increasing the depth limit after each iteration until the goal node is found.
- It combines the Benefits of Breadth - first search's fast search and depth - first search's memory efficiency.
- The iterative search Algorithm is useful uninformed search when search space is large, and depth of goal node is unknown.

- Initially depth limit is 0.
- on every iteration increase by 1
- Implemented using stack



1st Iteration : here d = 0 [S]

2nd Iteration : here d = 0+1 = 1 [S → A → C] {DFS}

3rd Iteration : here d = 1+1 = 2 [S → A → D → B → C → E → G] goal node

4th Iteration : here d = 2+1 = 3

Advantages.

- Incorporates Benefits of Both BFS and DFS. (Fast searching and less memory requirement)

Disadvantages

- Repeat the work / process.

Complete : Yes (By limiting the depth)

Time Complexity : $O(b^d)$

Space Complexity : $O(b \times d)$

Optimal : Yes (if step cost is uniform.

Iterative deepening Algorithm

1. Assign value of depth limited = 1.
2. Apply depth first search to a depth of depth - limited.
3. Increase depth - limited by 1.
4. If a solution found then return success.
5. Go to step 2.

Bidirectional search Algorithm

- Bidirectional search Algorithm runs 2 simultaneous searches, one from initial state called as forward search and other from end node called as Backward search, to find the goal node.

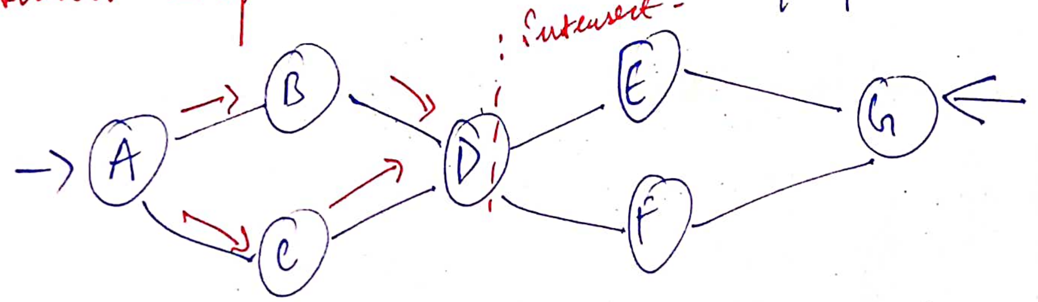
→ Forward search : looking - in front of the end from start

→ Backward search: looking from end to the start - backwards.

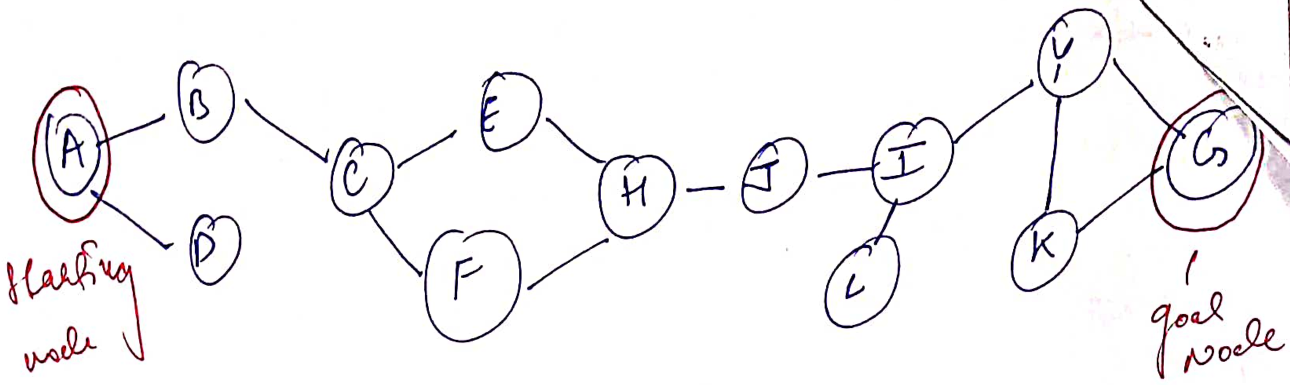
→ 2 different searches runs simultaneously.

- Single search graph is replaced with 2 small graphs.
- Any search technique can be used (BFS, DFS)

→ Search stop condition : when graphs intersect



A → B → C → D   G → E → F → D.



Forward	Backward
A	G
A B D	G K Y
A B D C	G K Y I
A B D C E F	G K Y I L J
A B D C E F H	G K Y I L J H

Terminating point.

A → B → D → C → E → F → H → J → L → I → Y → K → G

When to use Bidirectional Approach ?

We can consider Bidirectional Approach when

1) Both initial and goal states are unique and completely defined.

2) The Branching factor is exactly the same in both directions.

Advantages of Bidirectional search :-

1) Speed at which we get the desired results.
2) Drastically reduces the time taken by the search by having 2 simultaneous searches.
3) It also saves resources for users as it requires less memory capacity to store all the searches.

Disadvantages of Bidirectional search:

1) Implementation of Bidirectional search tree is difficult.
2) Goal state should be known in advance