DS & AI ENGINEERING

Artificial Intelligence

Un-Informed search

Lecture No.- 06

By- Aditya sir

Recap of Previous Lecture

- Topic
- Topic
- Topic
- Topic

UCS Questions

Topics to be Covered

- Topic
- Topic — UCS Questions
- Topic — Beam Search
- Bidirectional Search.

About Aditya Jain sir

1. Appeared for GATE during BTech and secured AIR 60 in GATE in very first attempt - City topper
2. Represented college as the first Google DSC Ambassador.
3. The only student from the batch to secure an internship at Amazon. (9+ CGPA)
4. Had offer from IIT Bombay and IISc Bangalore to join the Masters program
5. Joined IIT Bombay for my 2 year Masters program, specialization in Data Science
6. Published multiple research papers in well known conferences along with the team
7. Received the prestigious excellence in Research award from IIT Bombay for my Masters thesis in ML
8. Completed my Masters with an overall GPA of 9.36/10
9. Joined Dream11 as a Data Scientist
10. Have mentored 15,000+ students & working professions in field of Data Science and Analytics
11. Have been mentoring & teaching GATE aspirants to secure a great rank in limited time
12. Have got around 27.5K followers on Linkedin where I share my insights and guide students and professionals.

Telegram

Telegram Link for Aditya Jain sir:

https://t.me/AdityaSir_PW

Topic : Uninformed Search

DFS:

- It may stuck into any ∞ length branch and may not be able to reach result.
- 2 Solution

(1) DLS
(2) IDDFS

Topic : Uninformed Search

Depth Limited Search (DLS):

- Depth Limited Search is a modified version of DFS that imposes a limit on the depth of the search. This means that the algorithm will only explore nodes up to a certain depth, effectively preventing it from going down excessively deep paths that are unlikely to lead to the goal. By setting a maximum depth limit, DLS aims to improve efficiency and ensure more manageable search times.
- It may give no result
- If the goal state is below the mentioned depth.

Topic : Uninformed Search

Depth Limited Search (DLS):

- will not stuck into branch with ∞ nodes.
- d : depth of tree
- b : branching factor

Advantage:

- Less search time if the goal state is with in given depth.
- Time and space complexity
- TC ⇒ $O(b^L)$
- L : depth limit given for DLS
- SC ⇒ $O(b \times L)$

Topic : Uninformed Search

Depth Limited Search (DLS):

Disadvantage:

- Will not give result if the goal node is below specified depth.

$d > l$

Topic : Uninformed Search

Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

- There are two common ways to traverse a graph, BFS and DFS. Considering a Tree (or Graph) of huge height and width, both BFS (optimal But space) and DFS space less, not optimal sol are not very efficient due to following reasons.
- DFS first traverses nodes going through one adjacent of root, then next adjacent. The problem with this approach is, if there is a node close to root, but not in first few subtrees explored by DFS, then DFS reaches that node very late. Also, DFS may not find shortest path to a node (in terms of number of edges).
- BFS goes level by level, but requires more space.

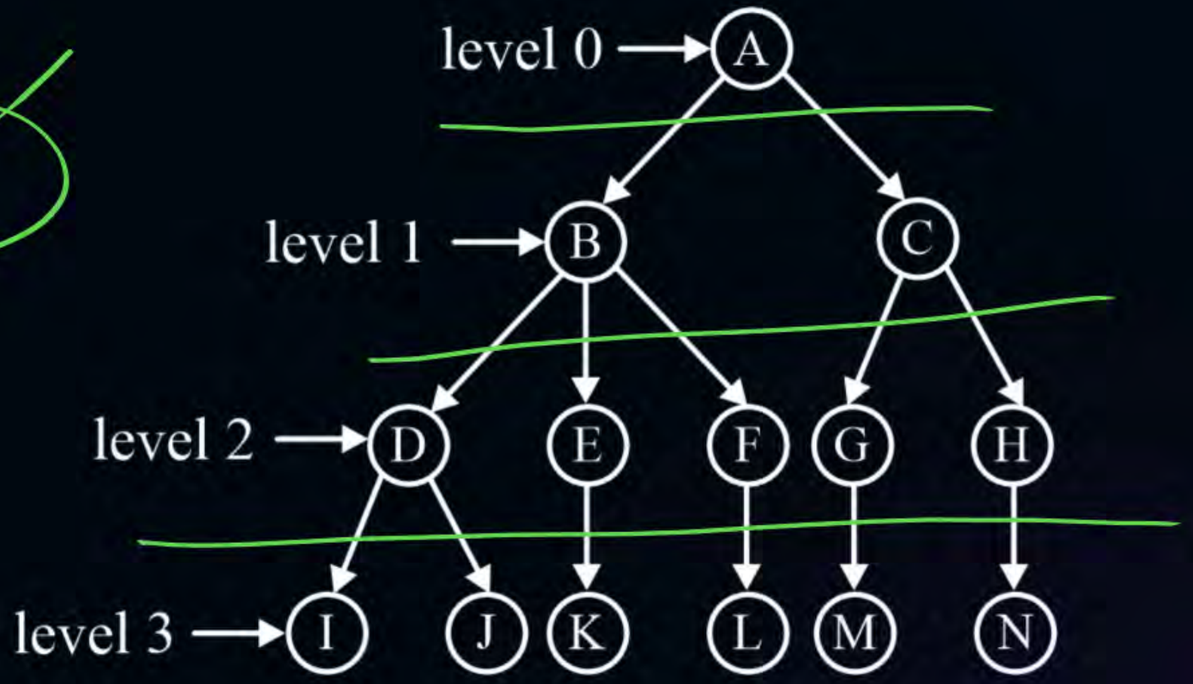
Topic : Uninformed Search

Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

IDDFS:

Better than DLS , DFS, BFS

- No depth is pride fine
- we start with depth limit = 0
- we start with level 0.
- goal node reached no.
- inc the depth limit by 1
- apply DFS.



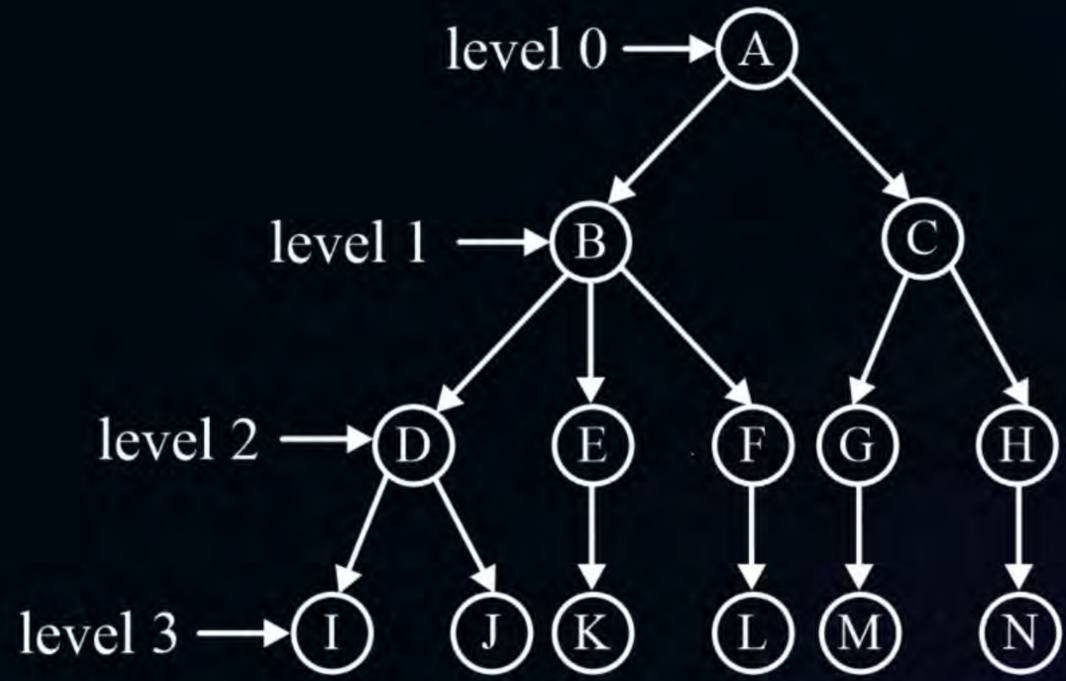
Topic : Uninformed Search

Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

Goal node (alpha)

I, H

- BFS
- ~~IDS~~ DFS
- IDDFS



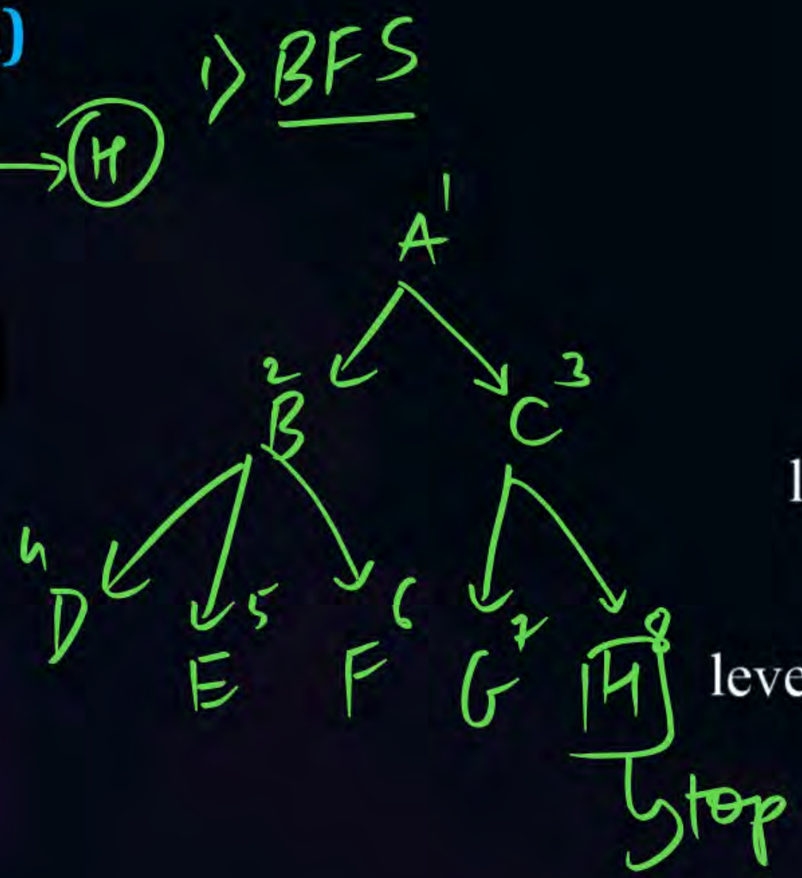
Topic : Uninformed Search

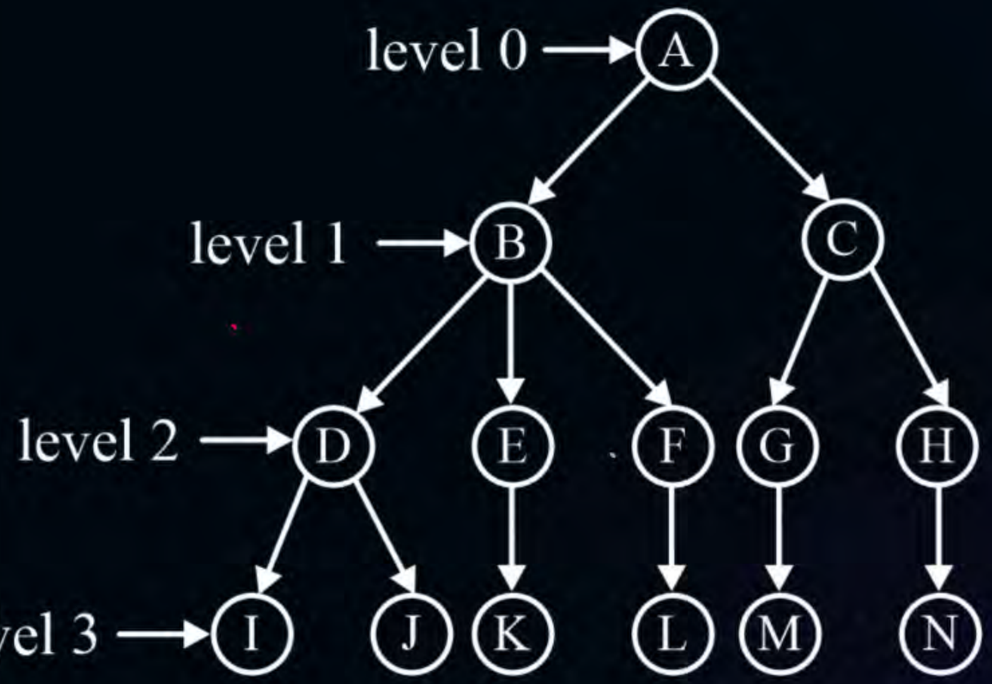
Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

Goal node (alpha)

I, H

- BFS
- ~~IDS~~ DFS
- IDDFS





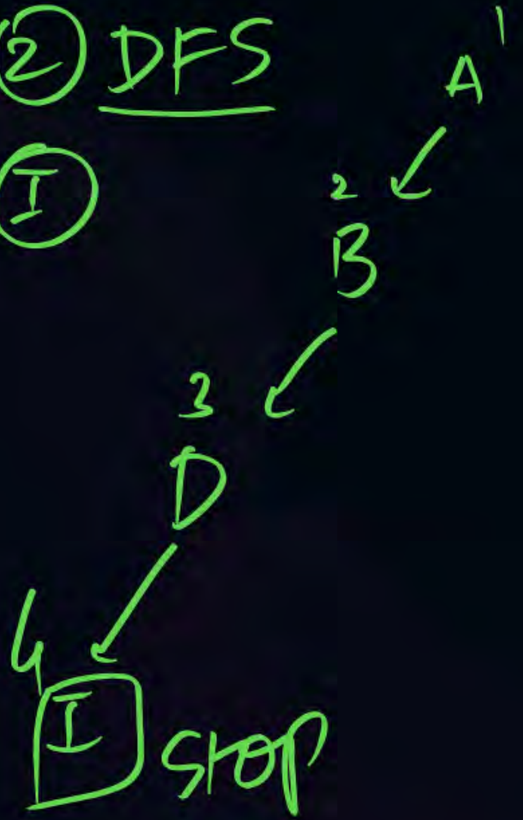
Topic : Uninformed Search

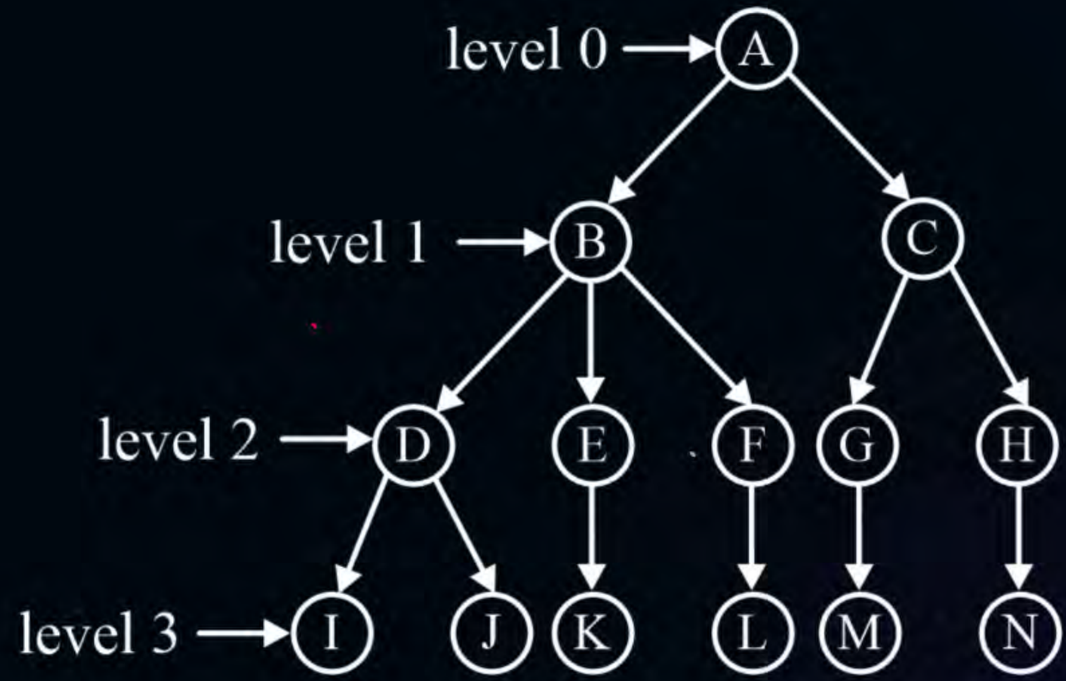
Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

Goal node (alpha)

I, H

- BFS
- ~~IDS~~ DFS → I
- IDDFS





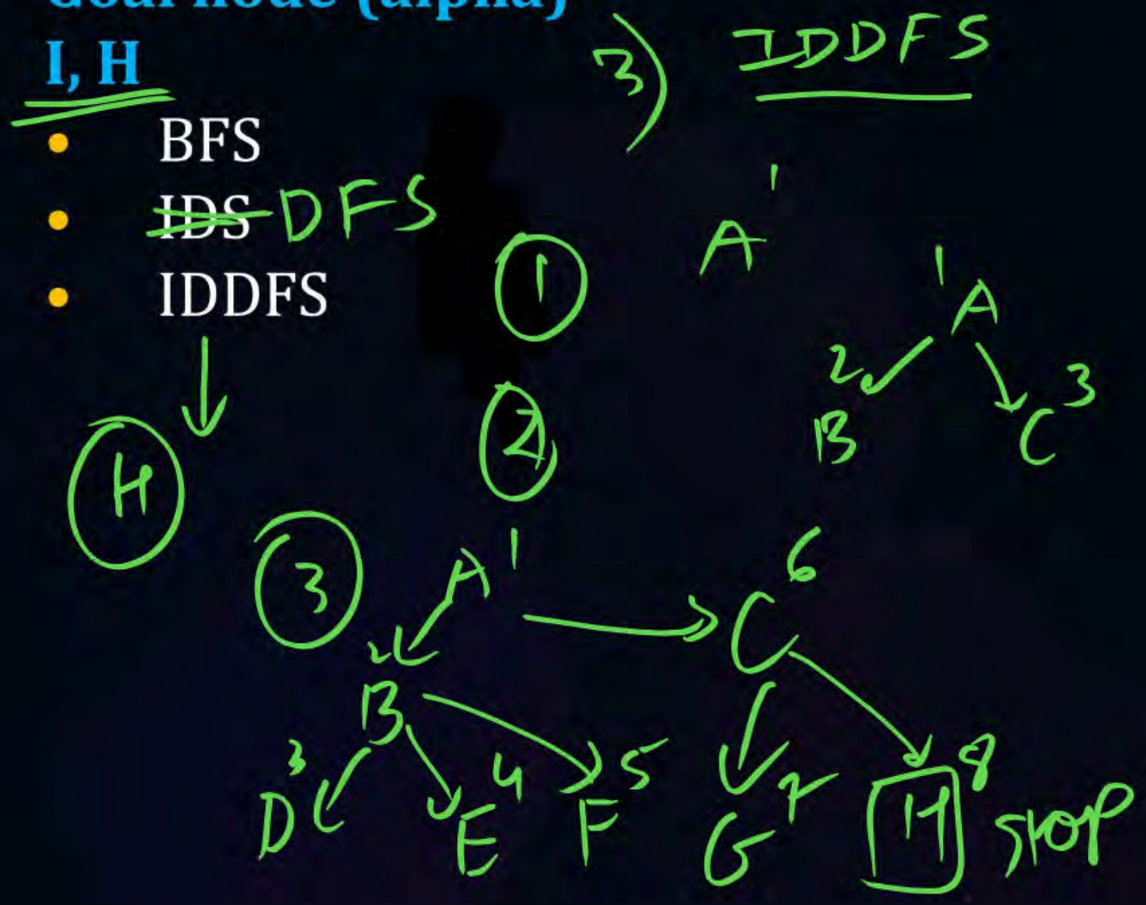
Topic : Uninformed Search

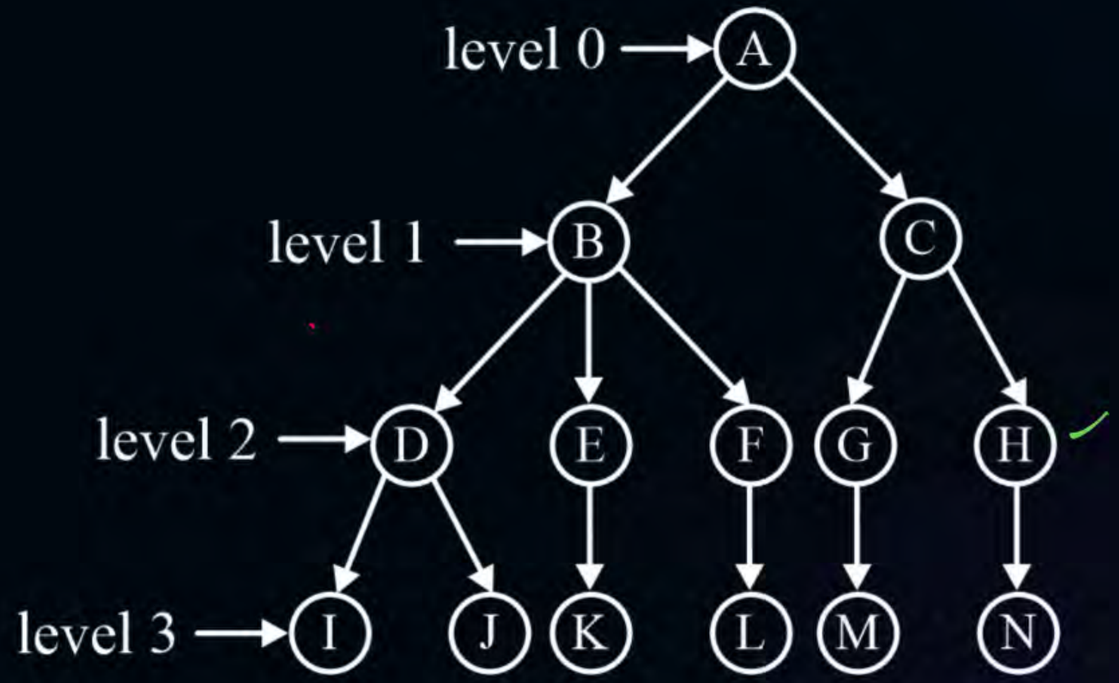
Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

Goal node (alpha)

I, H

- BFS
- ~~IDS~~ DFS
- IDDFS





Topic : Uninformed Search

Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

1. Completeness
2. Optimal result
3. Less space complexity

BFS

DFS

IDDFS:

- [redacted] DFS → less space complexity
- Result BFS → advantage optimum result.

Topic : Uninformed Search

Iterative Deepening Search (IDS) or Iterative Deepening Depth First Search (IDDFS):

- Iterative Deepening Depth-First Search (IDDFS) combines the depth-first search's space efficiency with the breadth-first search's completeness. It repeatedly performs depth-limited searches with increasing depth limits until the goal is found.

Topic : Uninformed Search

Uniform Cost search:

Uniform Cost Search (UCS) is a search algorithm used to find the least-cost path from a start node to a goal node in a graph. UCS is a variant of Dijkstra's algorithm and is used in situations where the path costs vary, making it more suitable for weighted graphs.

Topic : Uninformed Search

Uniform Cost search:

Key Concepts:

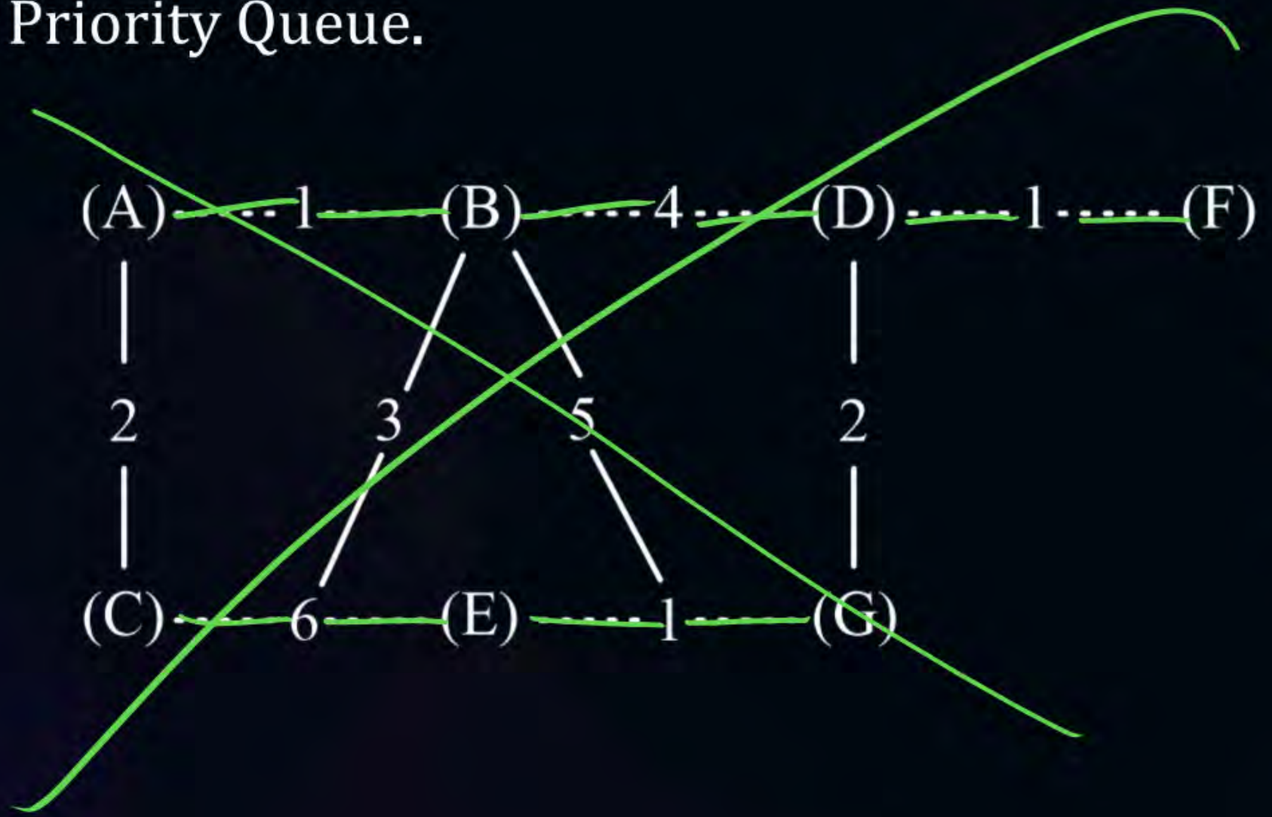
- Priority Queue UCS uses a priority queue to explore the least-cost path first. Nodes are expanded in order of their cumulative cost from the start node.
- Cost: Each edge in the graph has a cost associated with it. UCS aims to find the path with the minimum cumulative cost.
- Expansion: The algorithm expands the node with the smallest cumulative cost, ensuring that the cheapest path is always chosen.

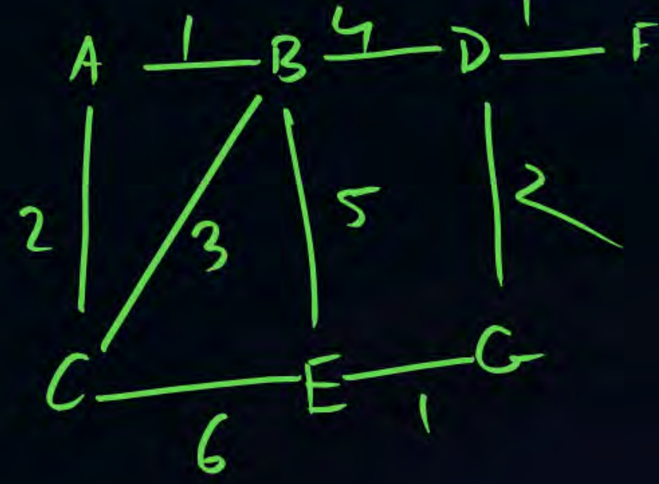
Topic : Uninformed Search

A → G

Uniform Cost search:

Here we use a Priority Queue.



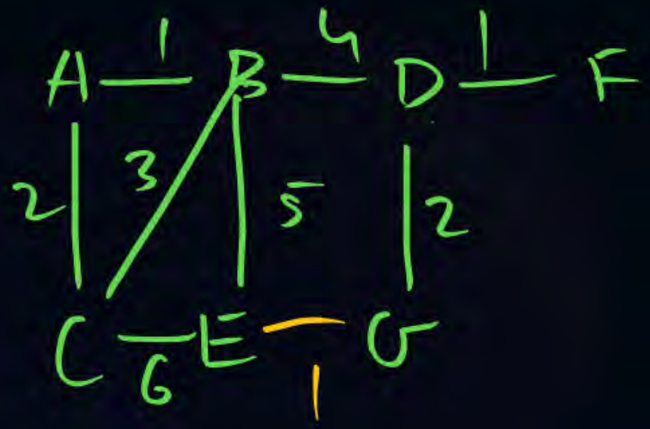


Topic : Uninformed Search

A → G : 7

Uniform Cost search:

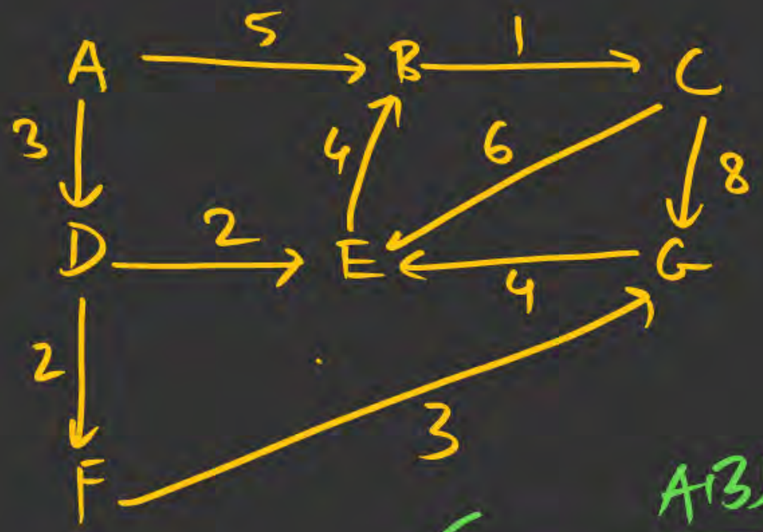
Your task is to find the least cost path from city A to city G using Uniform Cost Search (UCS).

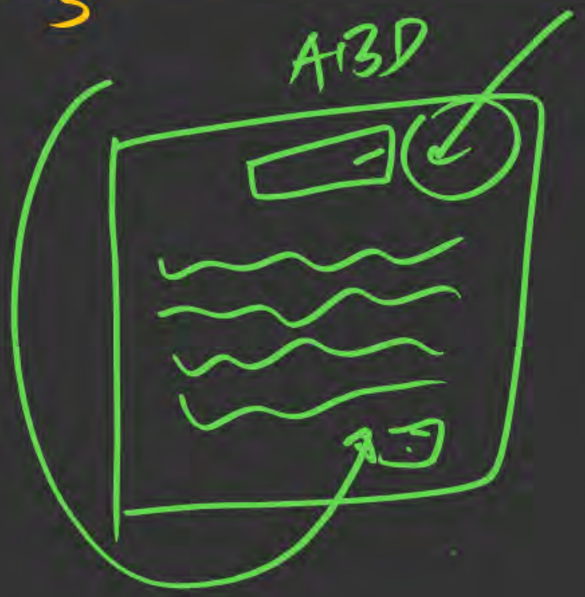


open	closed						
A	A	B	C	D	E	F	G
B	(1)	1	1	1	1	1	1
C	2	(2)	2	2	2	2	2
D	X	5	(5)	5	5	5	5
E	X	6	6	(6)	6	6	6
G	X	X	X	7	7	(7)	(7) ✓
F	X	X	X	6	(6)	6	6

UCS

A → G





(5)

VCS

Cost: 8

Open	Closed					
A	A	D	B	E	F	C
B	5	(5)	5	5	5	5
D	(3)	3	3	3	3	3
E	X	5	(5)	5	5	5
F	X	5	5	(5)	5	5
C	X	X	6	6	(6)	6
G	X	X	X	X	8	(8) → goal

D → F → G

A

ADFG

TC & SC of UCS

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

C^* = approx optimal path cost of the Soln.

ε = Smallest path weight in entire graph.

UCS 

If all paths are of equal cost (ε_e)

$C^* = (\text{depth of the graph} * \varepsilon_e) \longrightarrow$ Worst Case

$= d \times \varepsilon_e$

WC TC & SC of UCS $= O(b^{d * \varepsilon_e / \varepsilon_e}) = O(b^d)$

↓

BFS

Beam Search:

UCS → (all childs of visited node)

→ Select the one with min cost.

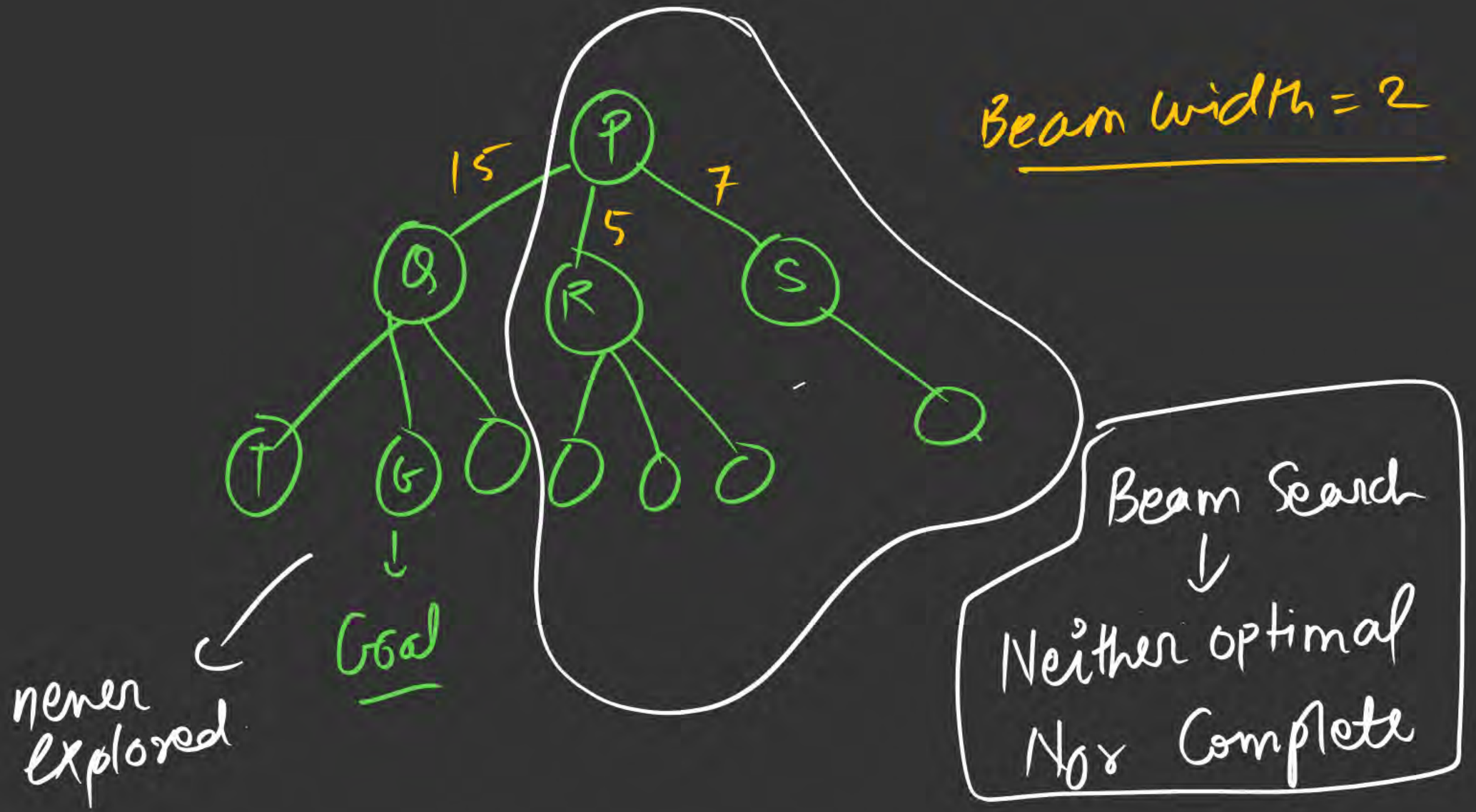
↓

Large SC due to many childs (larger b)

BW = 2

Beam Search → [Beam width]

limit child and prefer the ones with min cost.


Beam width = 2
P
15
5
7
Q
R
S
T
G
Goal
never explored
Beam Search
↓
Neither optimal
Nor Complete

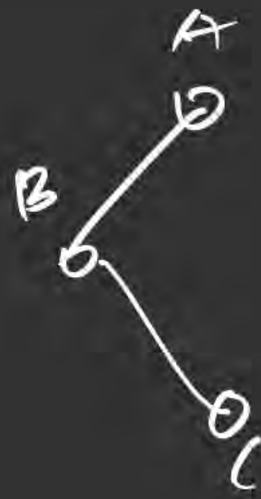
* Bidirectional Search :

BFS

$\rightarrow O(b^d)$ due to large number of visited nodes

Bidirectional Search

$2^8 \rightarrow 2^4$



TC & SC of Bidirectional Search:

↳ Graph depth = d

Both BFS from S & G will meet at $d/2$

$$TC \& SC = O(b^{d/2} + b^{d/2}) = O(b^{d/2})$$

Bidirectional Search:

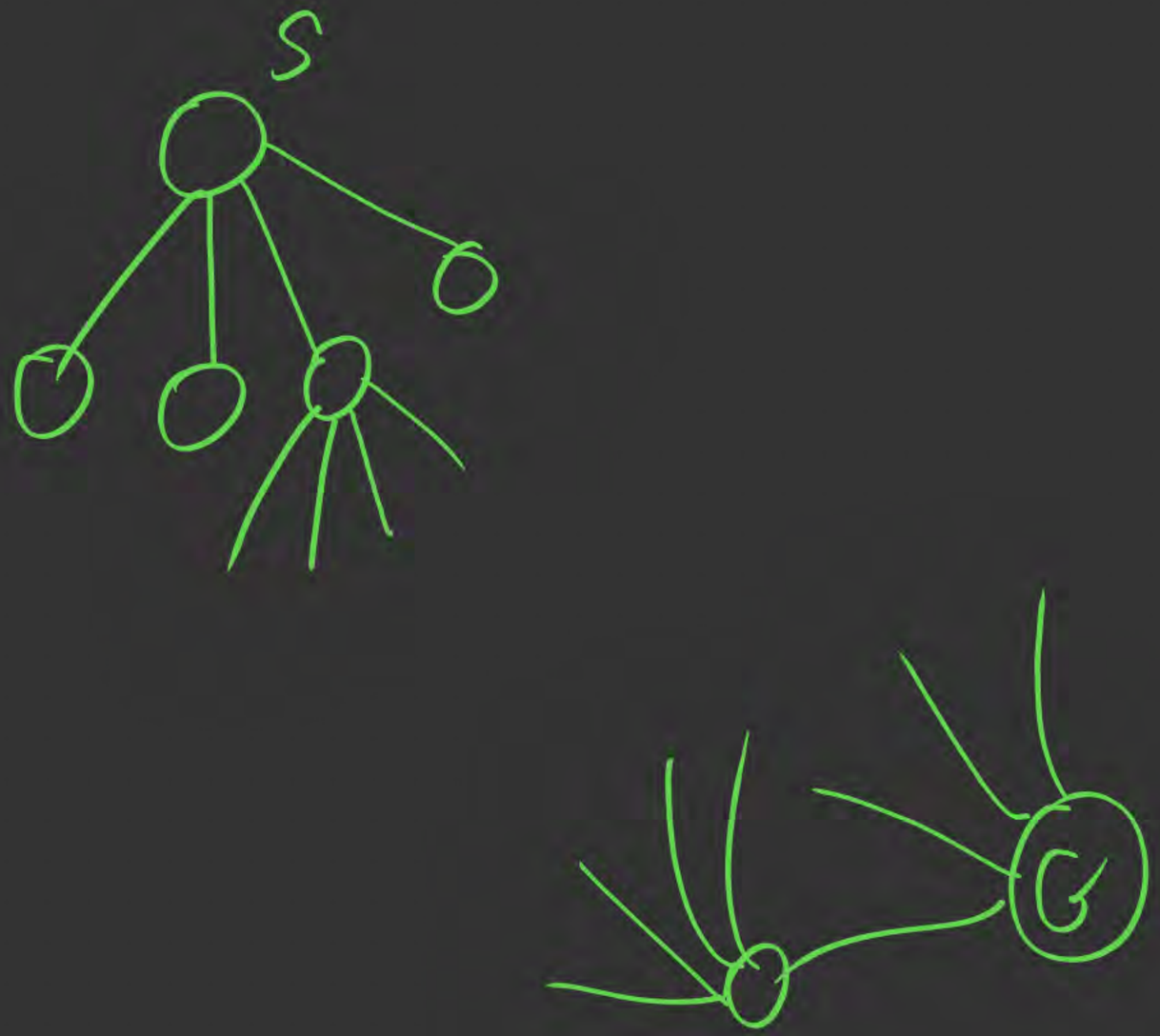
Adv:

① TC & SC reduced $\Rightarrow O(b^{d/2})$

Disadvantages:

① Presence of multiple Goals

② Should be able to have operation to move to parent node


S
G

Types :-

Uninformed
- (1) Non-wt graph → BFS
- (2) wt graph → UCS

(3) Informed → A^*

THANK - YOU