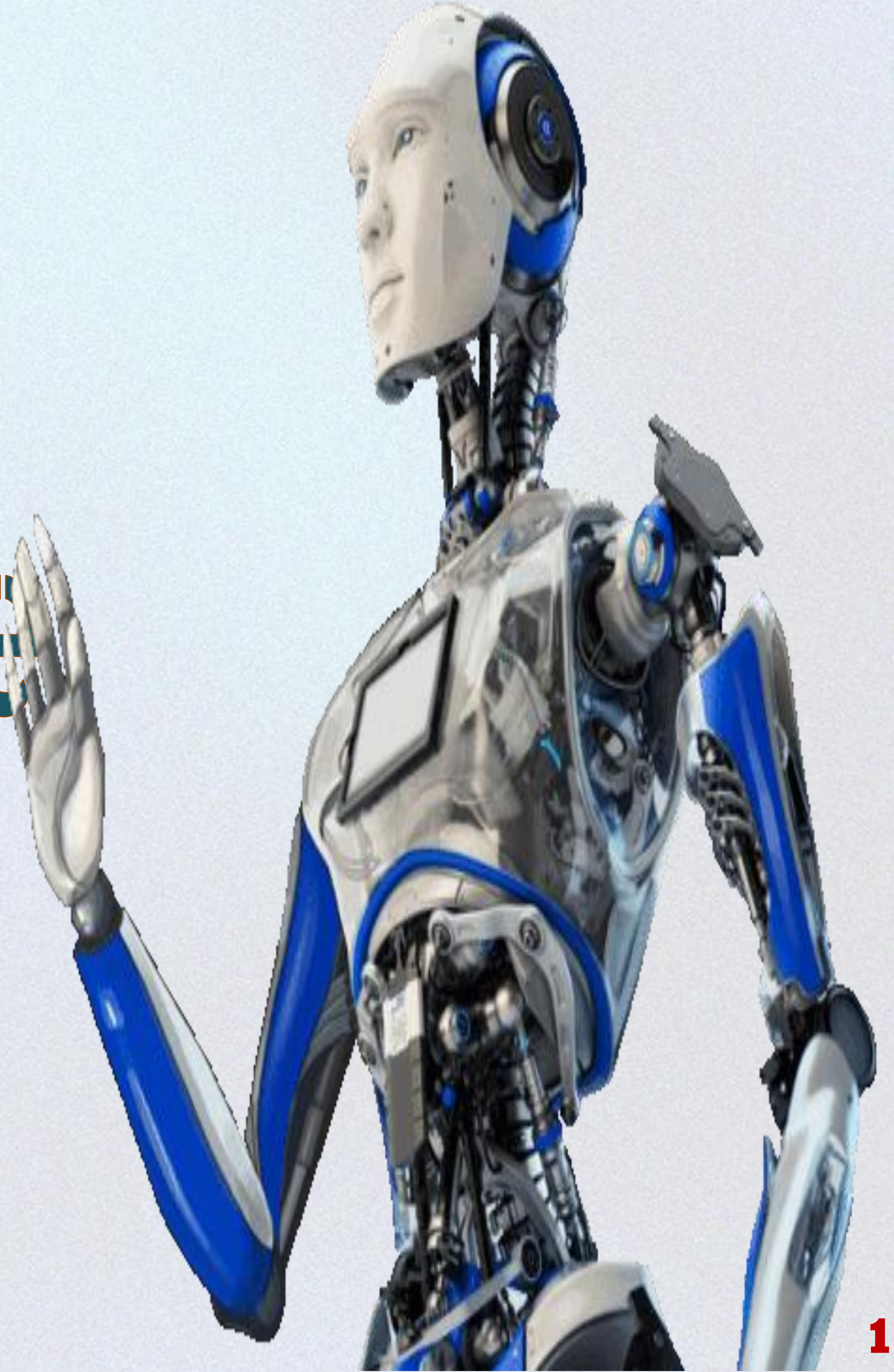
AIE111 :

Artificial Intelligence



Dr. Noha El-Sayad

Lecture 3:

Solving Problems by Searching Algorithm

Searching Algorithm in AI: Types

Traversal vs Search

- Traversal: Visit each node once
- Search: Find a path between two nodes

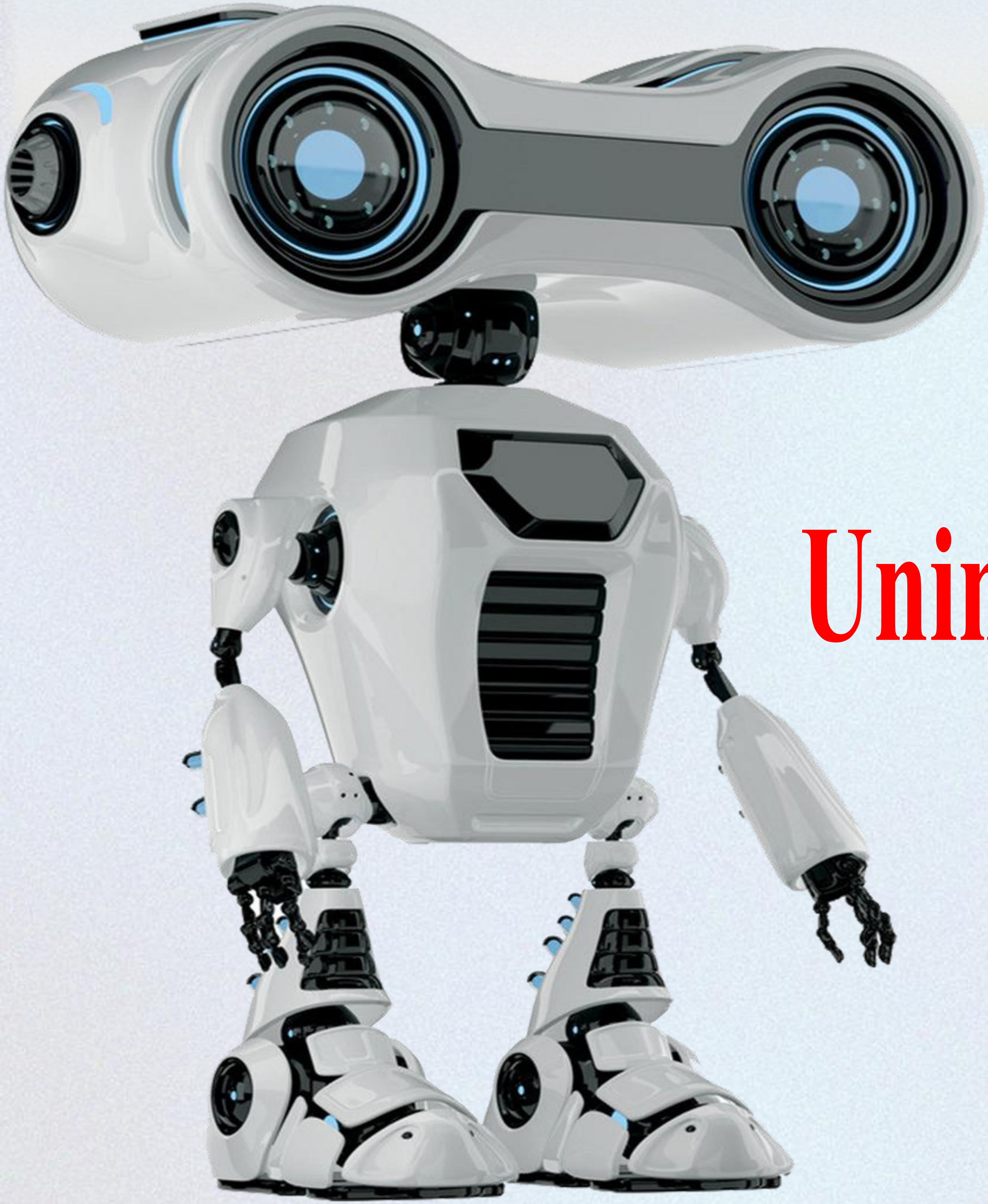
Uninformed Search	Informed Search
– breadth-first	– best-first search
– uniform-cost search	– search with heuristics
– depth-first	– memory-bounded search
– depth-limited search	– iterative improvement search
– iterative deepening	
– bi-directional search	

Blind Algorithm

- ✓ Number of **steps**, **path cost** unknown
- ✓ Agent knows when it reaches a goal

heuristic search

- ✓ Agent has background information about the problem

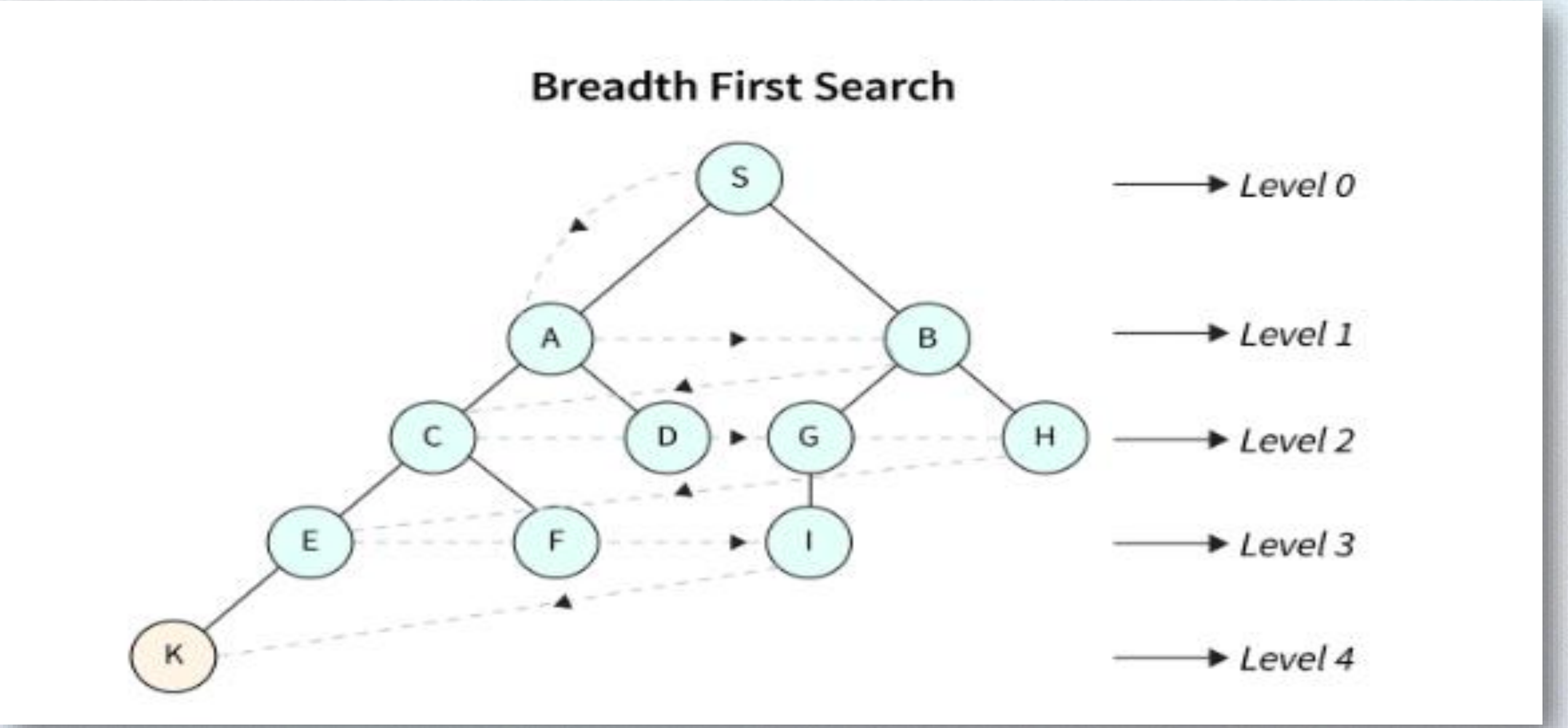


Uninformed Search Strategies

1. Breadth-first search (BFS)

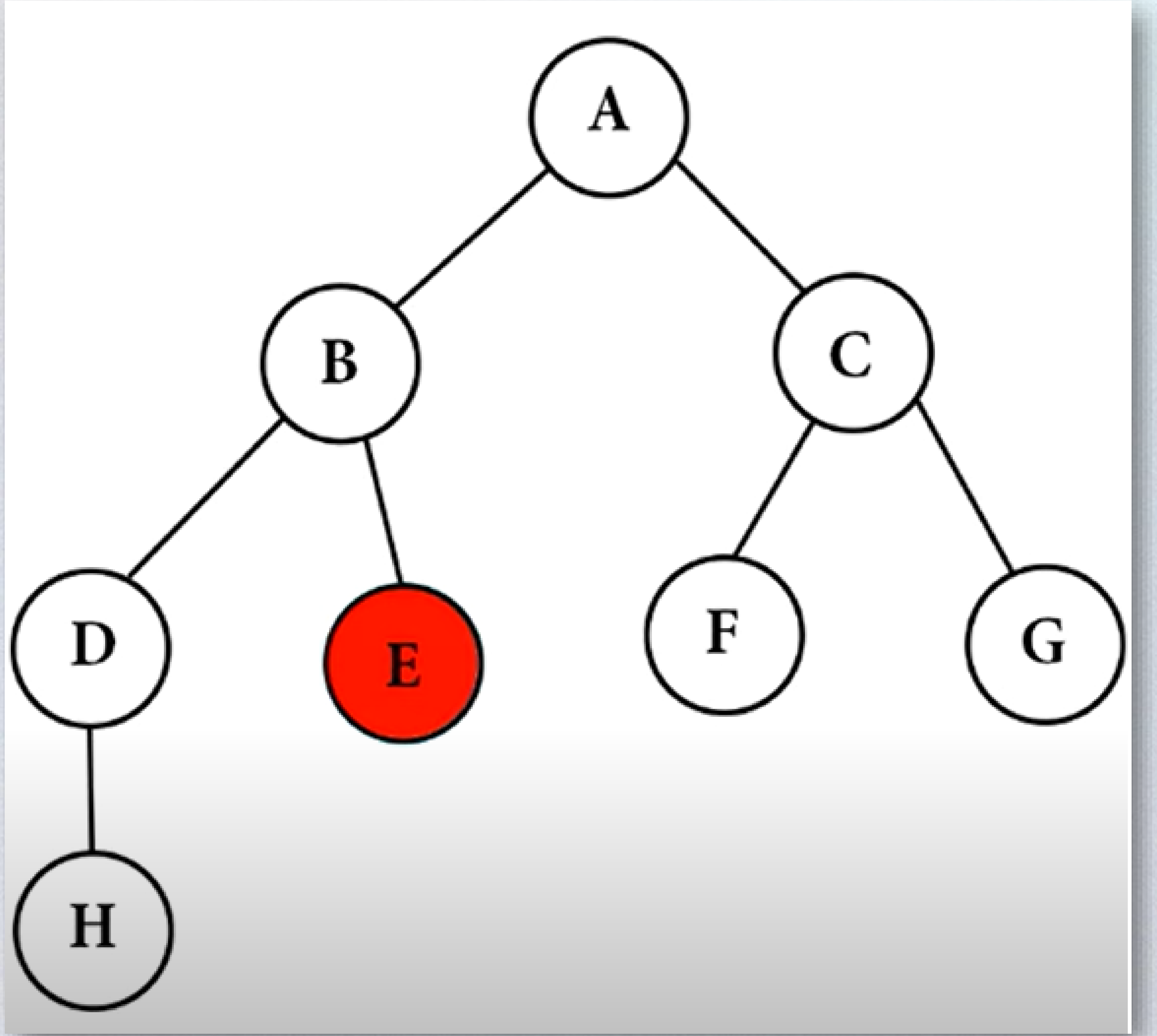
Breadth-First Search (BFS): A graph or tree search method that starts at the root and explores all nodes at the current depth before moving to the next level.

It uses a **FIFO queue** and is a **complete algorithm**, guaranteeing a solution if one exists.



Example:

Breadth-First Search (BFS), Goal = E



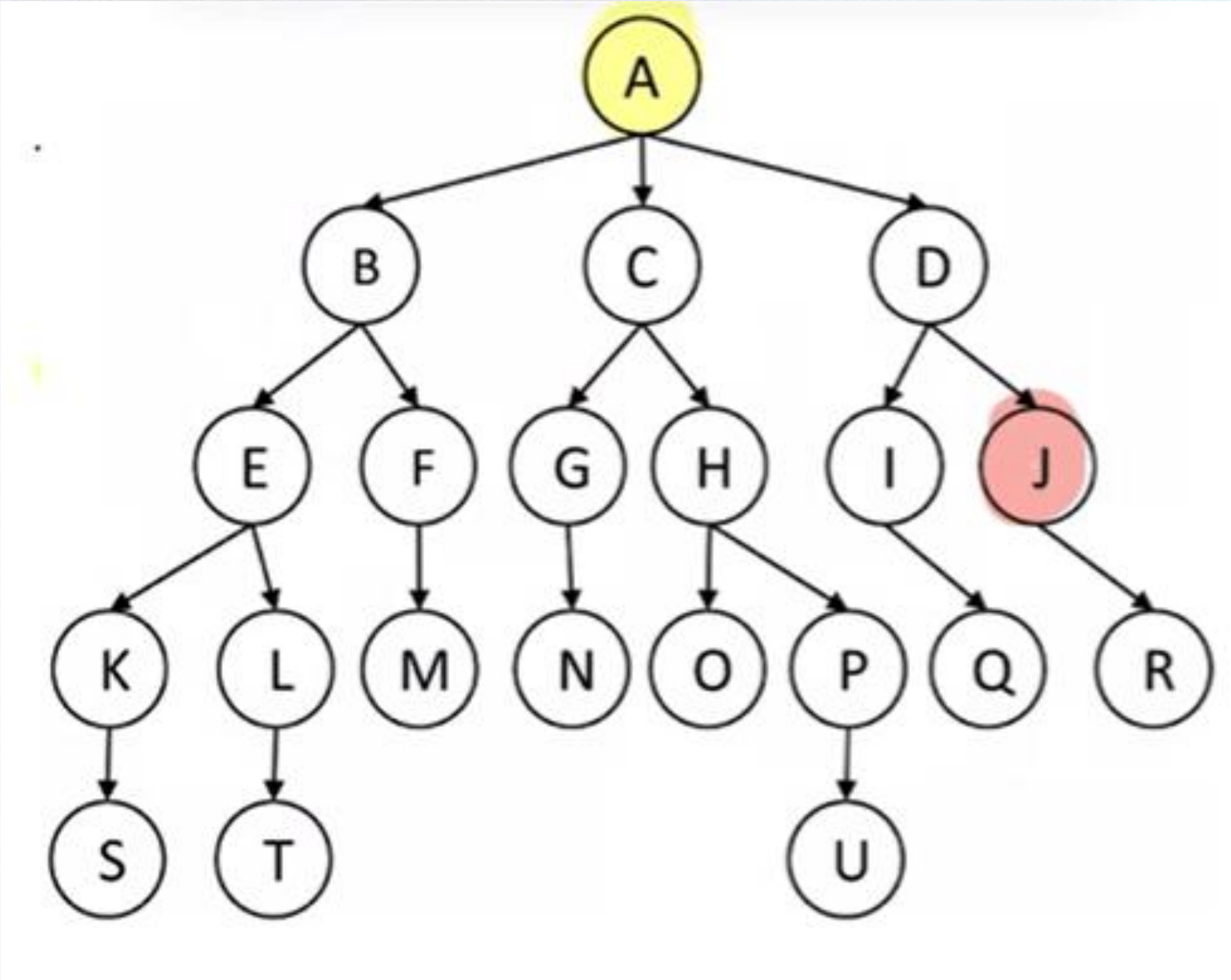
Solution

Visited	Queue
—	A
A	B C
B	C D E
C	D E F G
D	E F G H
E	Goal

Complete : no
Optimal: no
Time Complexity: $O(b^m)$
Space Complexity: $O(bm)$
Implementation: Stack {LIFO}

Example:

Breadth-First Search (BFS), Goal = j



Solution

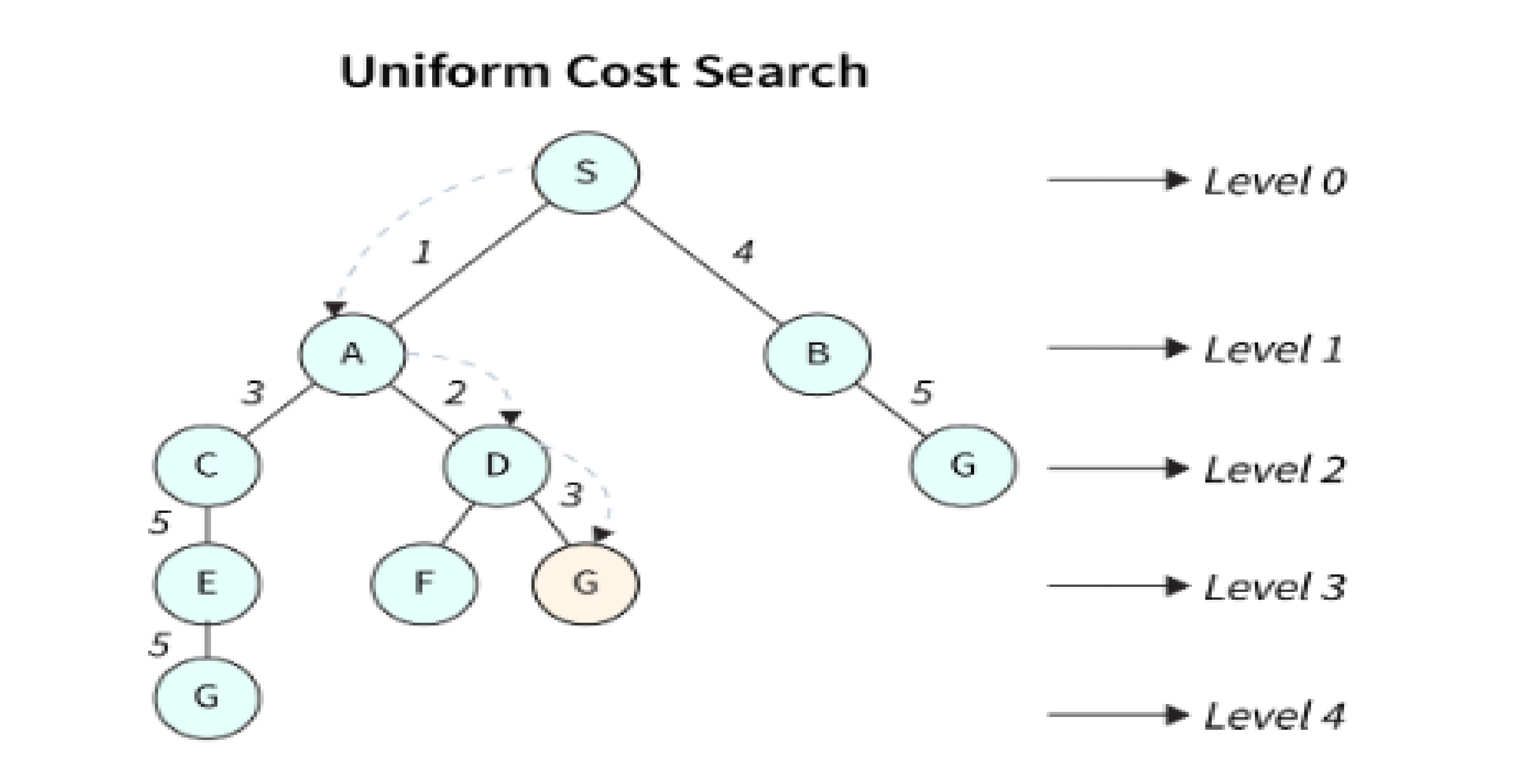
Visited	Queue
-	A
A	B C D
B	C D E F
C	D E F G H
D	E F G H I J
E	F G H I J K L
F	G H I J K L M
G	H I J K L M N
H	I J K L M N O P
I	J K L M N O P Q
J	Goal

2. Uniform-Cost-First (UCS)

- ✓ A search algorithm that **prioritizes** the lowest-cost path when multiple paths exist.
- ✓ Visits the next node which has the least total cost from the root, until agoal state is reached.

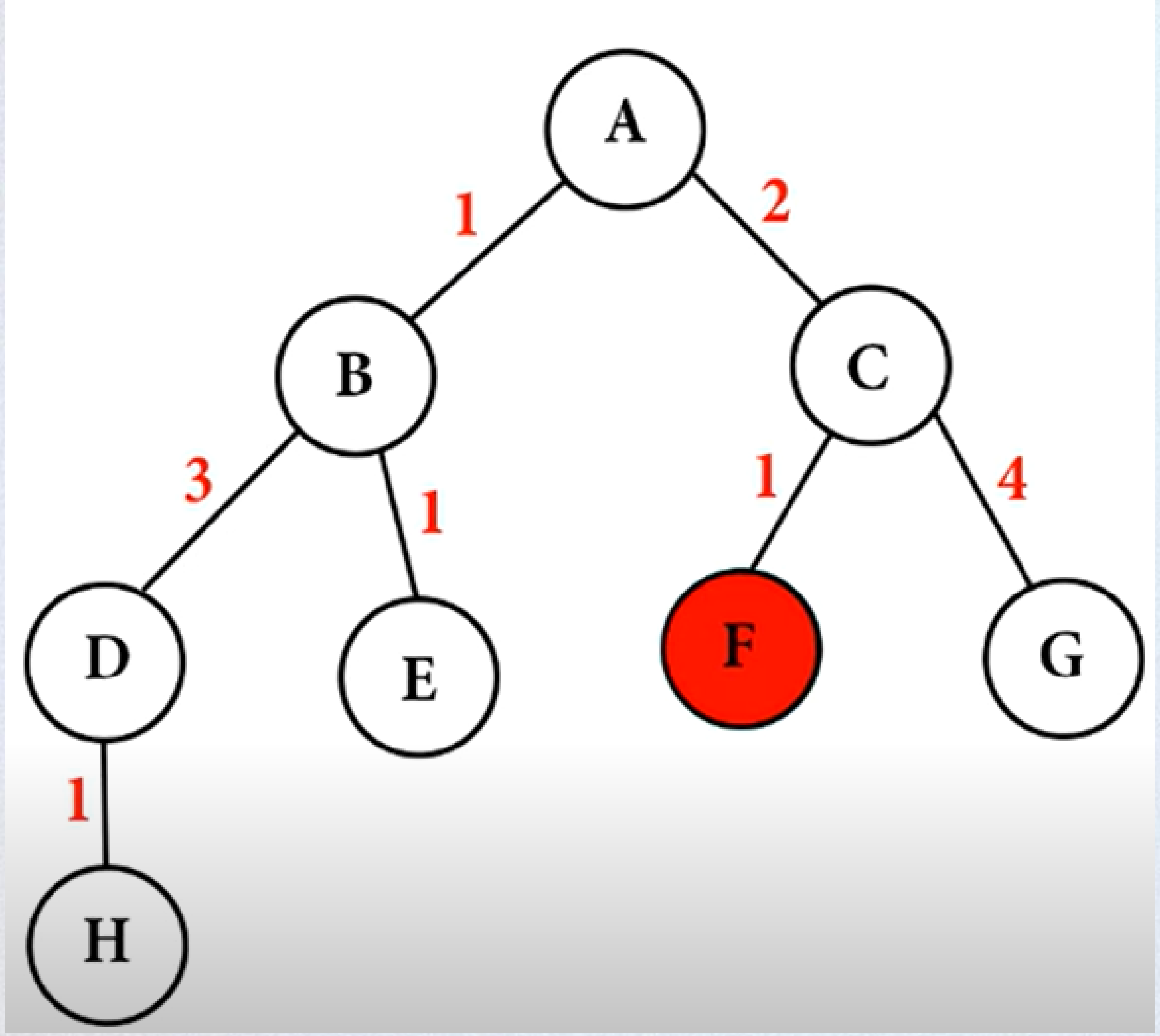
  It ensures **optimality** if no negative costs are present

  and **completeness** if states are finite with no zero-weight loops.
- ✓ **UCS behaves like BFS when all transitions have equal costs.**
- ✓ g(n) = path cost(n) = sum of individual edge costs to reach the current node.

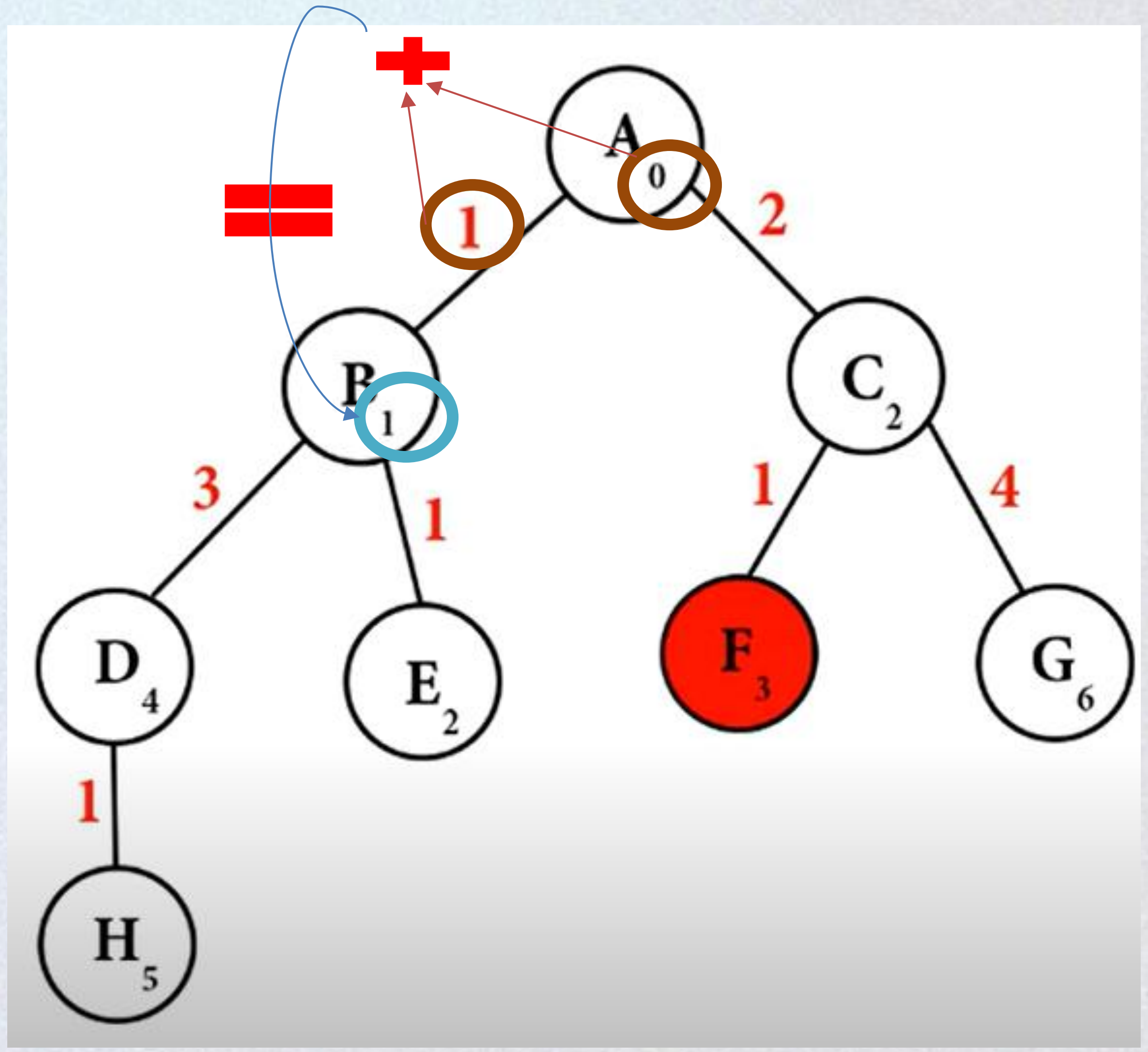


Example:

Uniform-Cost-First (UCS), Goal = F



Solution



1St, calculate coat for each node , start with Root then level by level, where root coast = 0

Example:

Uniform-Cost-First (UCS), Goal = F

2nd, Start the queue with the root node based on its cost. Then, mark the root as visited and enqueue its children, arranging each child's cost in ascending order.

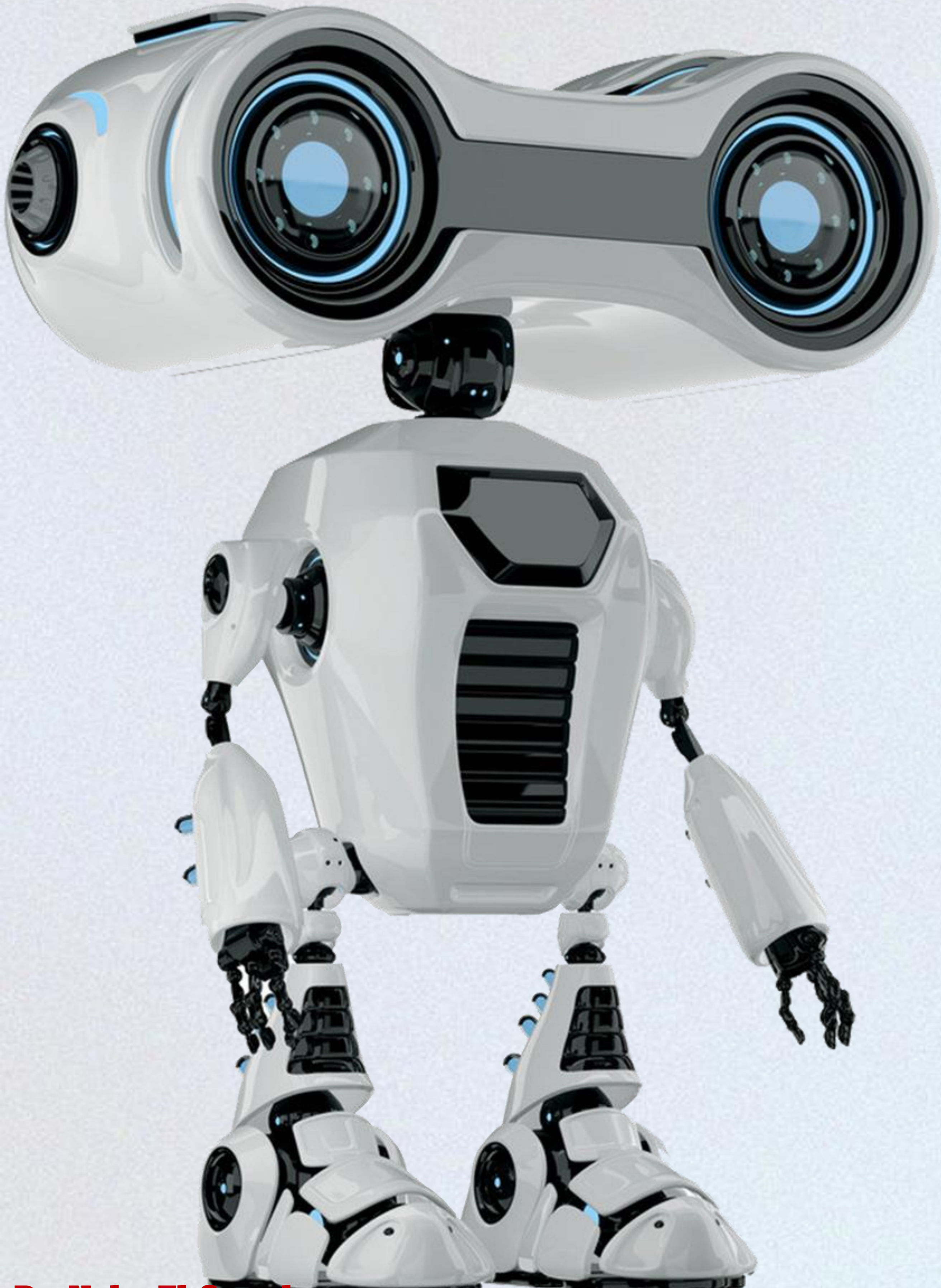
Visited	Queue	
—	A_0	
A_0	B_1 C_2	
B_1	C_2 D_4 E_2	C_2 E_2 D_4
C_2	E_2 D_4 F_3 G_6	E_2 F_3 D_4 G_6
E_2	F_3 D_4 G_6	
F_3	Goal	

Start with root node A
Cost = 0

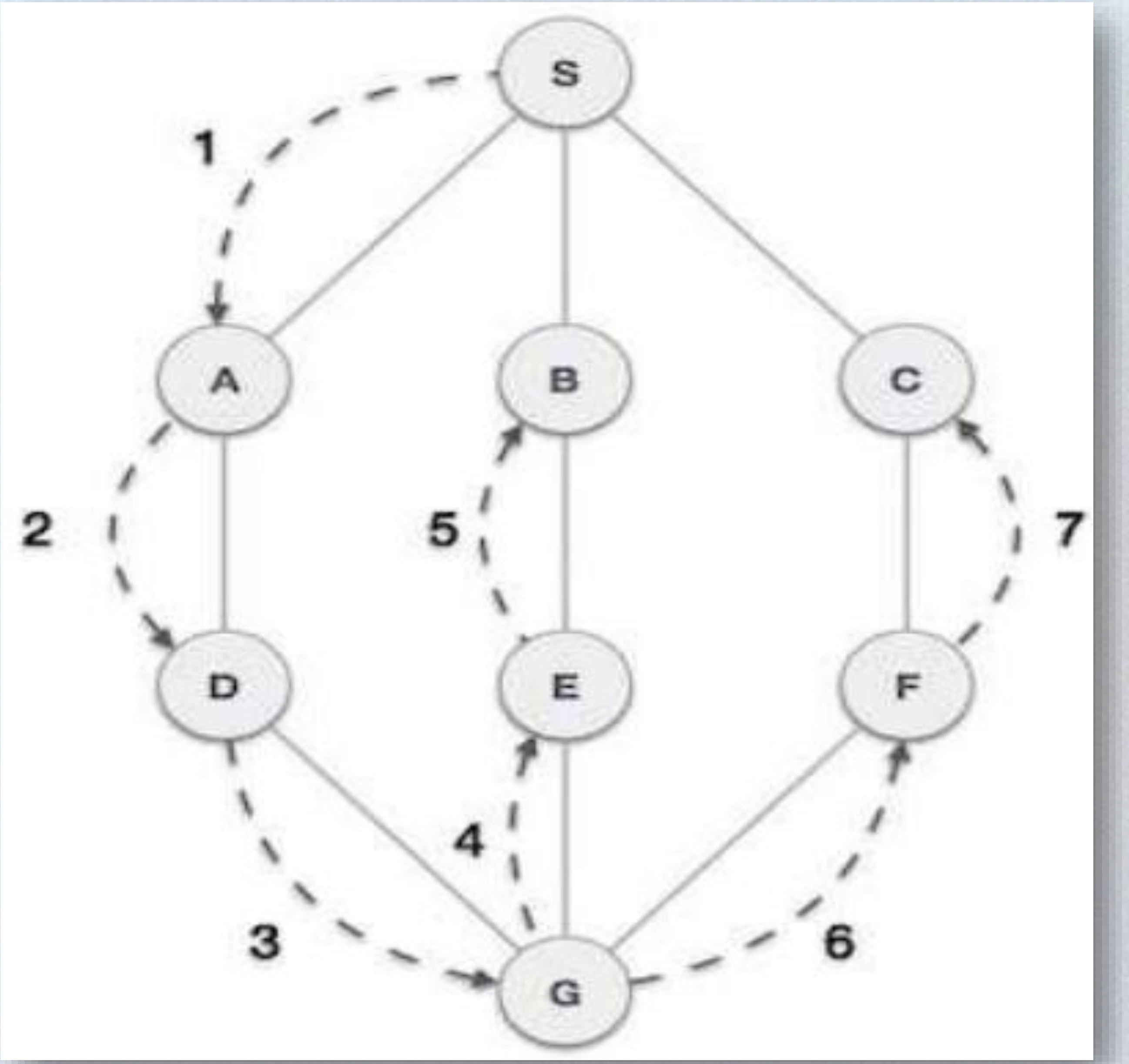
Add its children to the queue and arrange their costs in ascending order

Place C before E, even though they have the same cost, because C entered the queue before E

3. Depth-first search (DFS)

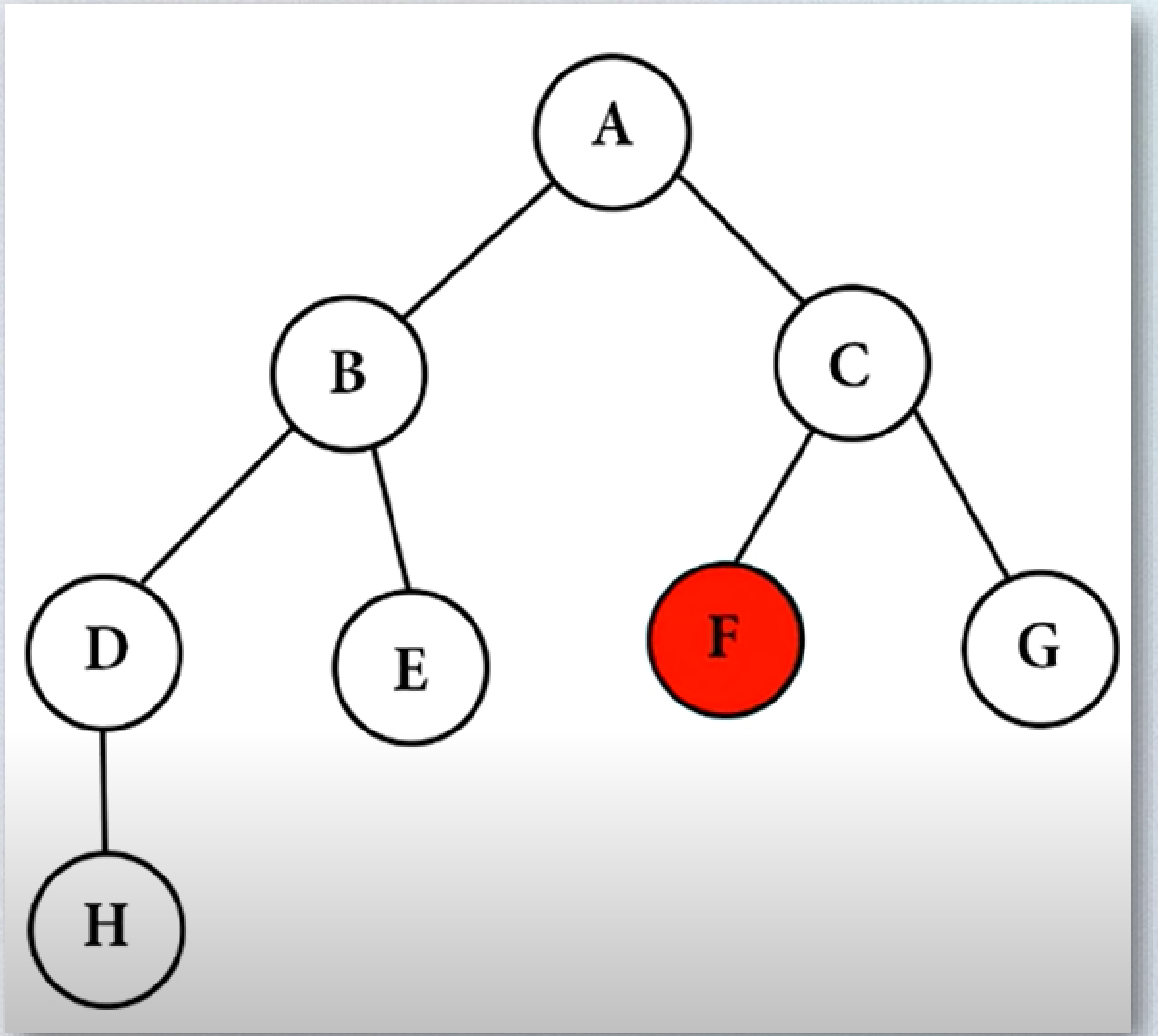


- ✓ A graph or tree traversal algorithm that **starts at the root** node and explores branch nodes deeply before backtracking.
- ✓ For example, in the figure shown: DFS traverses S, A, D, G, E, B before **backtracking** to E to G and then **visiting** F then C.
- ✓ Backtracking is implemented using a **(LIFO)** stack for implementation.



Example:

Depth-first search (DFS), Goal = F



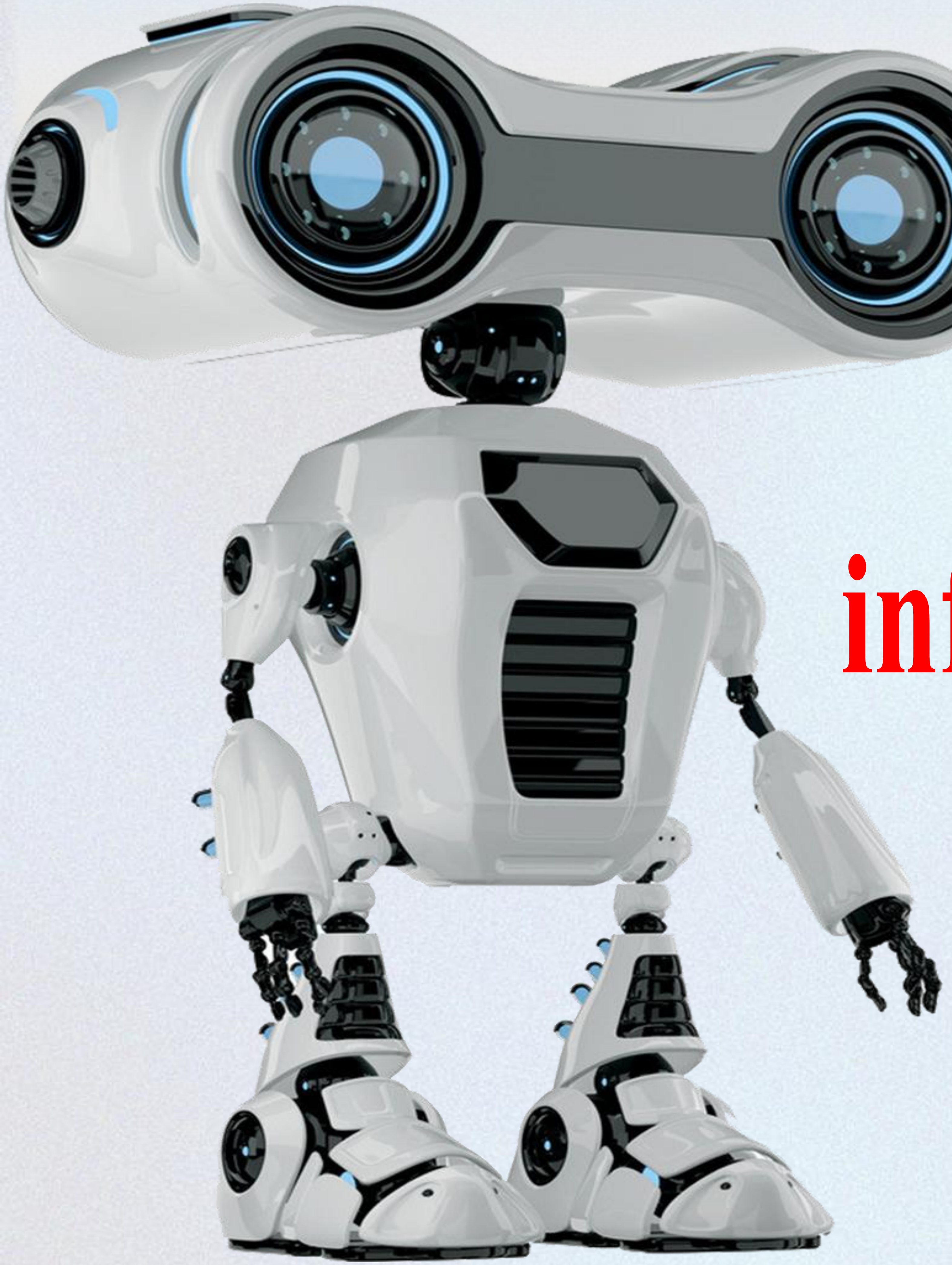
Solution

Visited	Queue
—	A
A	B C
B	D E C
D	H E C
H	E C
E	C
C	F G
F	Goal

Root

Children of A

Place Children of B at first



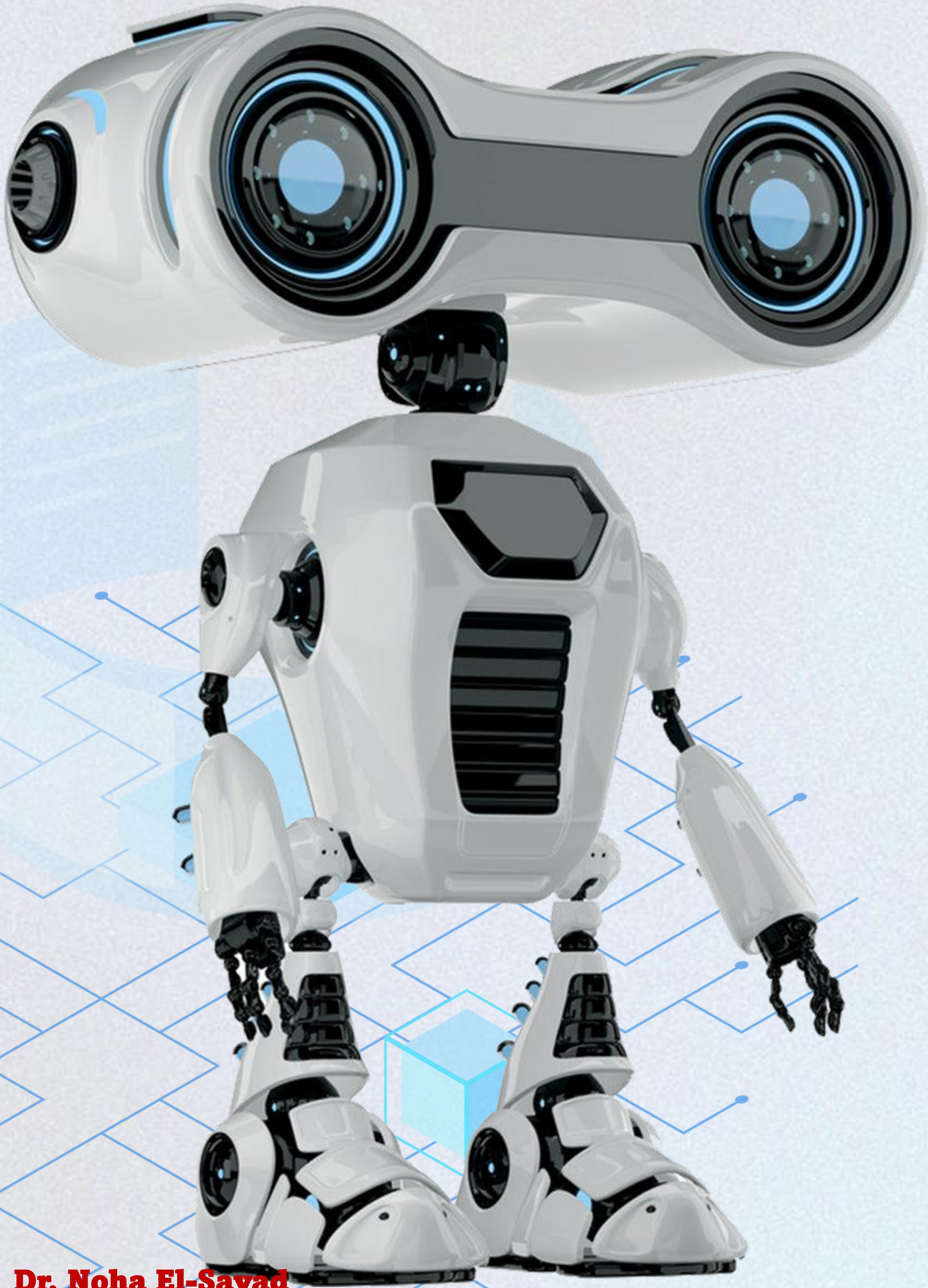
informed Search Strategies

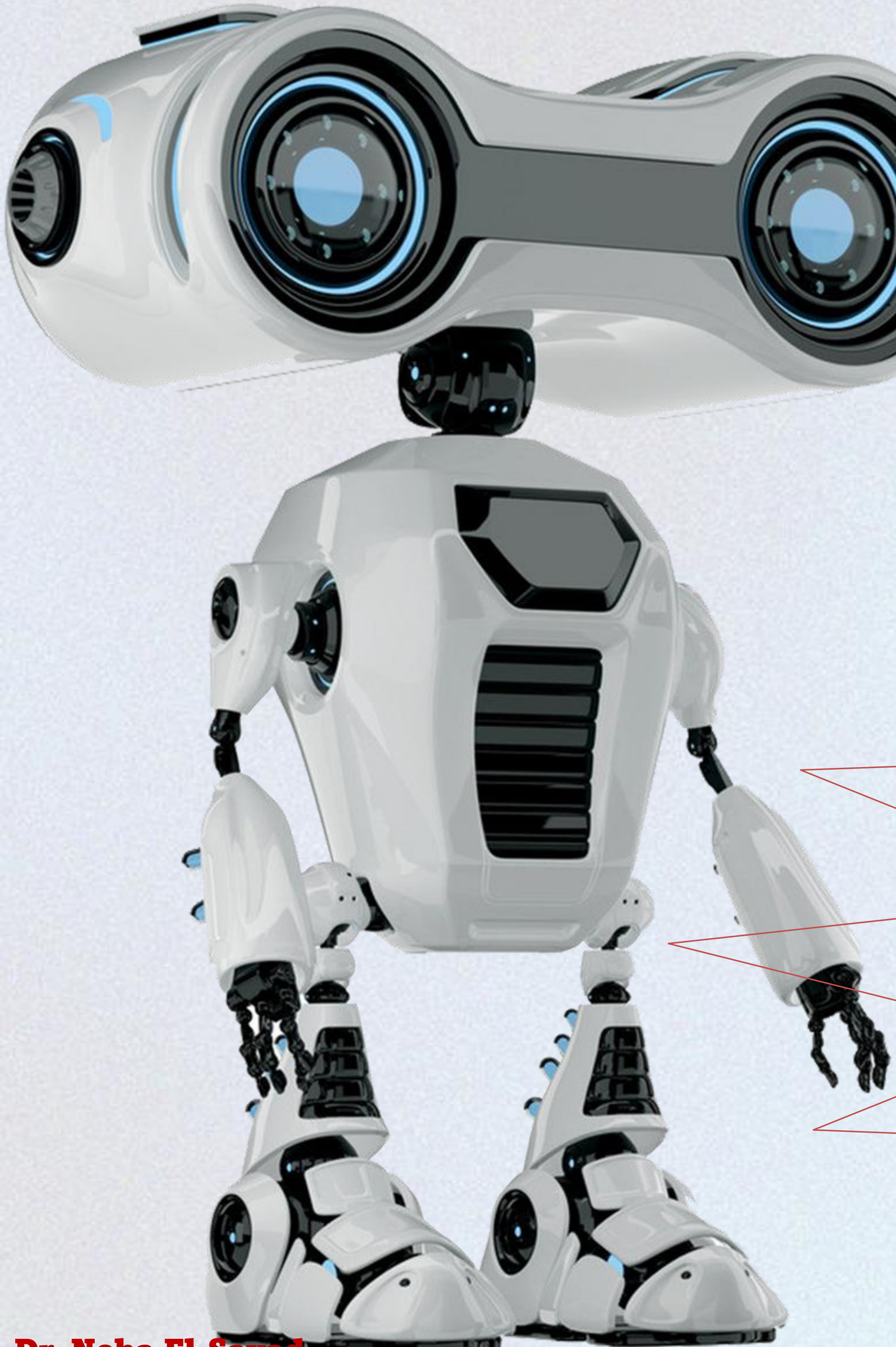
Summary of Informed Search in AI

- **Informed Search (Heuristic Search):**

- Uses domain knowledge to guide the search, making it more efficient than uninformed search.

- heuristic function provides an estimate of the best path to the goal, though it doesn’t always guarantee the optimal solution.





1. Greedy best-first search

- ✓ A heuristic function h(n) = estimated cost of the cheapest path from
- ✓ the state at node n to a goal state.
- ✓ At each step, best-first search sorts the queue according to a heuristic function.
- ✓ evaluation function $f(n) = h(n)$.

✓ Expands the closest node to the goal based on a heuristic function h(x) (estimated distance to goal).

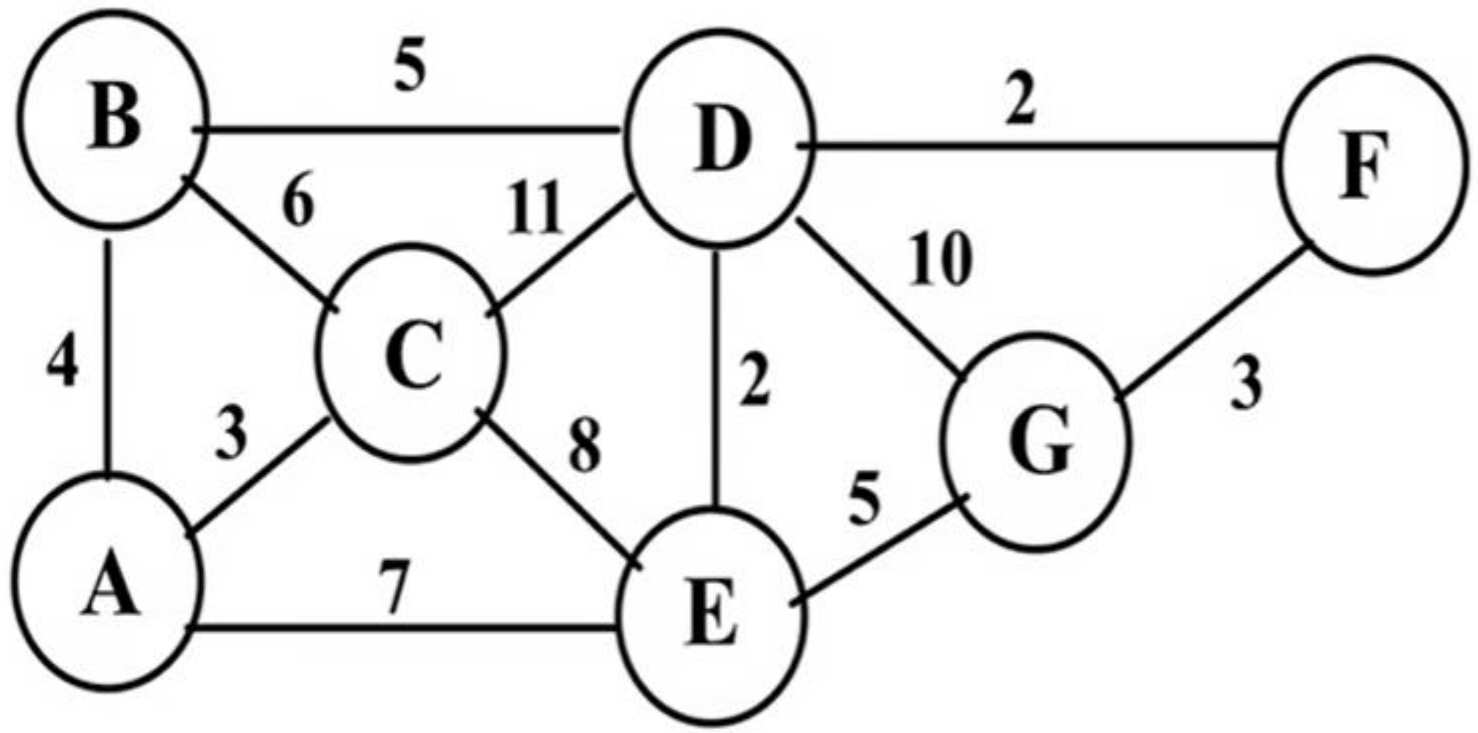
✓ Uses a priority queue for implementation.

✗ Not optimal; can get stuck in loops or dead ends.

Example:

Greedy best-first search , Start = A, Goal = F

Find shortest path from **A** node to **F** node. Note that the heurestic value h(n) equal to (node level)x2.



Solution

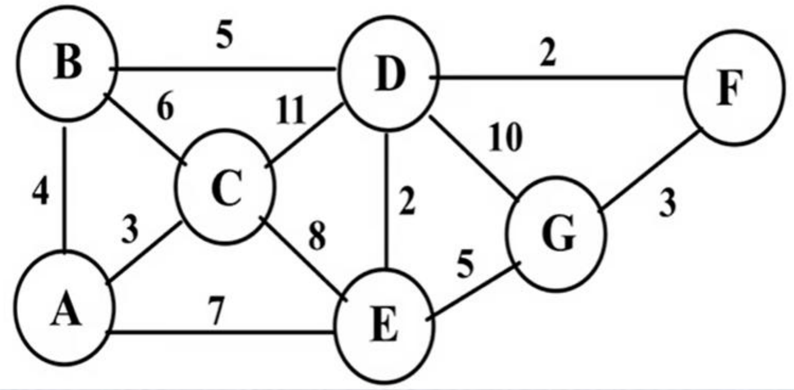
1st Step

node(n)	h(n)
n(A) = 7+3+4=14	h(A) = 28
n(B) = 15	h(B) = 30
n(C) = 28	h(c) = 56
n(D) = 30	h(D) = 60
n(E) = 22	h(E) = 44
n(F) = 5	h(F) = 10
n(G) = 18	h(E) = 36

Example:

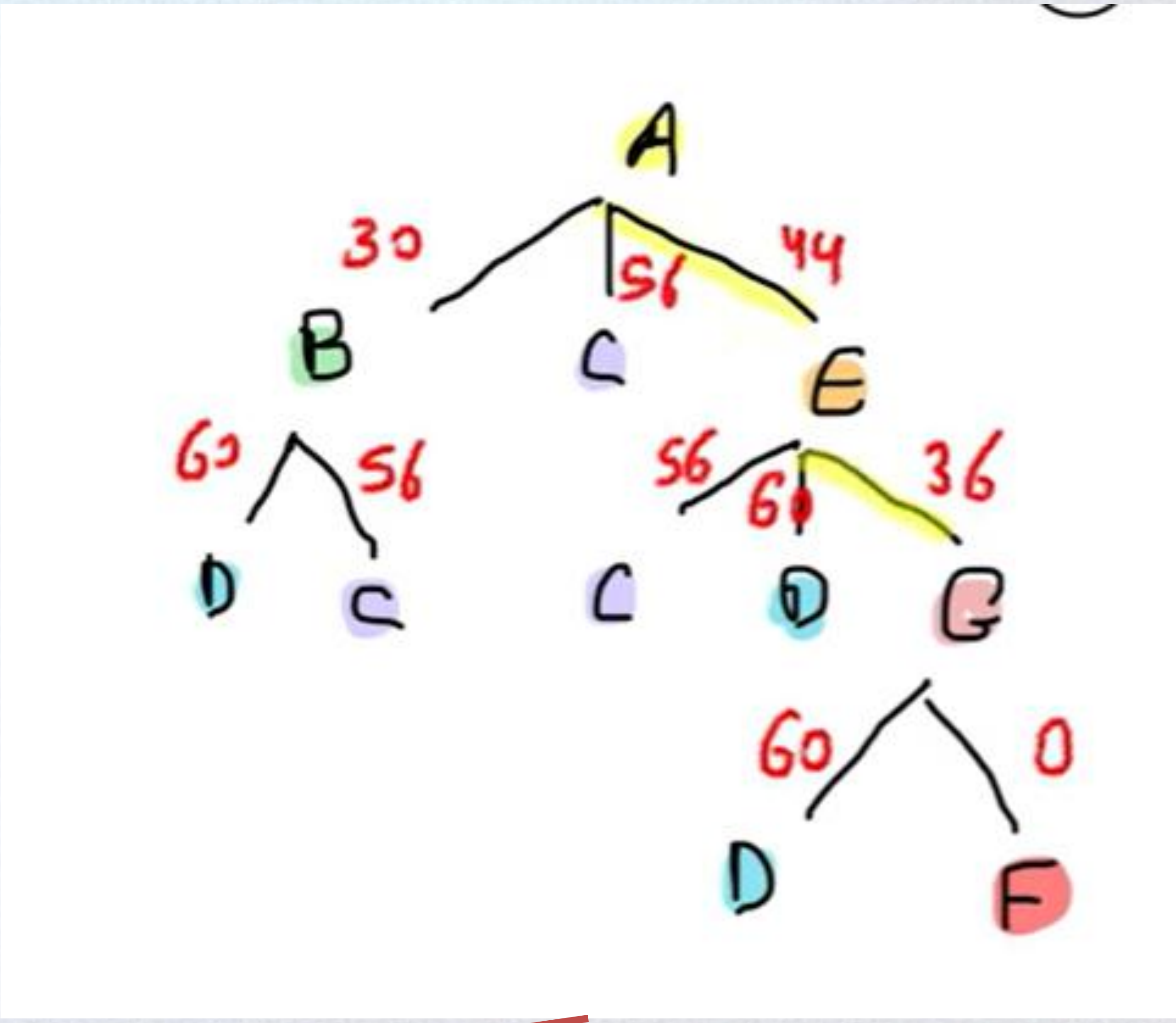
Greedy best-first search , Start = A, Goal = F

Find shortest path from **A** node to **F** node. Note that the heurestic value h(n) equal to (node level)x2.



node(n)	h(n)
n(A) = 7+3+4=14	h(A) = 28
n(B) = 15	h(B) = 30
n(C) = 28	h(c) = 56
n(D) = 30	h(D) = 60
n(E) = 22	h(E) = 44
n(F) = 5	h(F) = 10
n(G) = 18	h(E) = 36

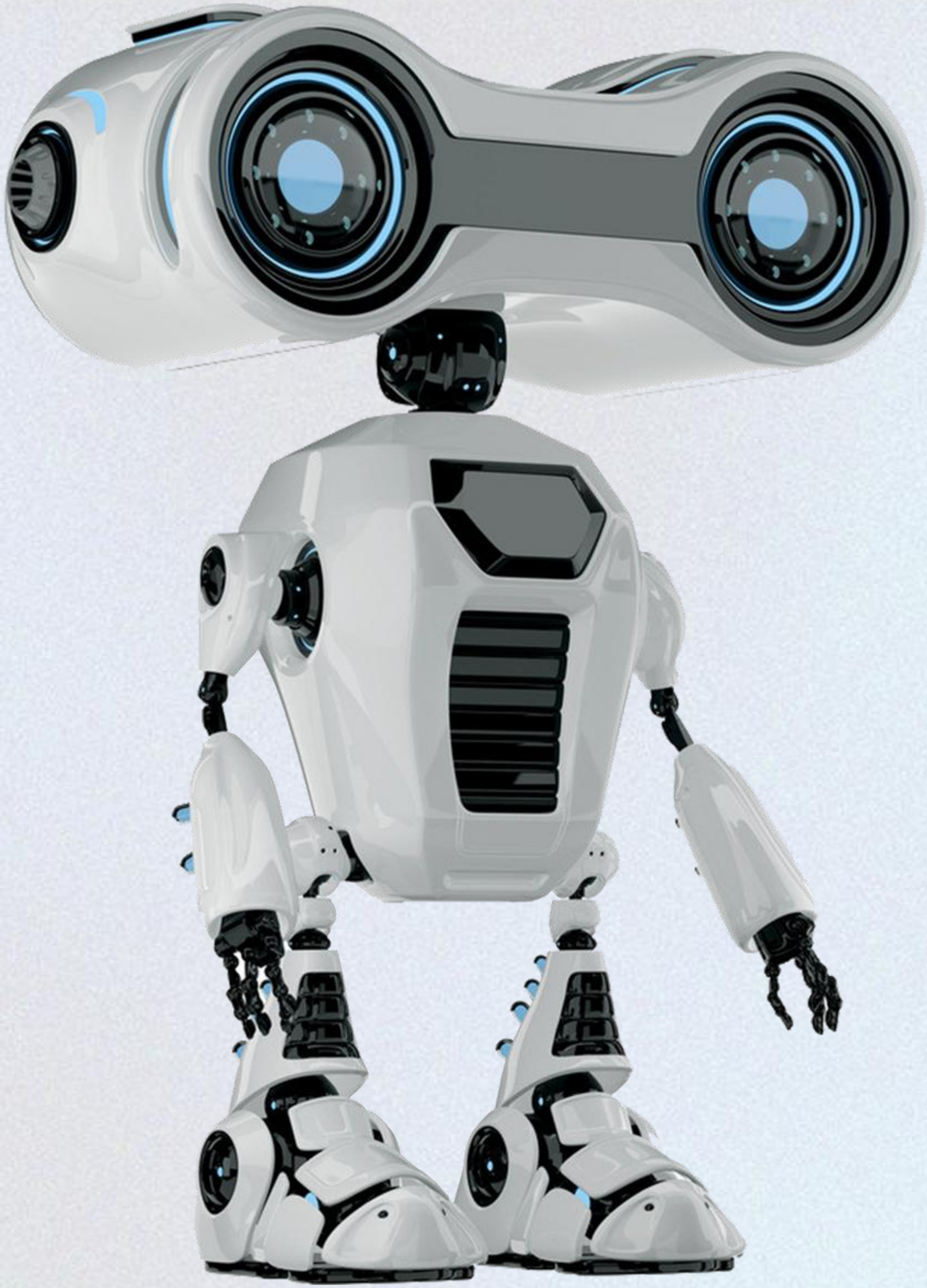
2nd Step



3rd Step

$A \xrightarrow{7} E \xrightarrow{5} G \xrightarrow{3} F$ (15)

2. A* search



- ✓ best-first search that uses the evaluation function

$$f(n) = g(n) + h(n)$$

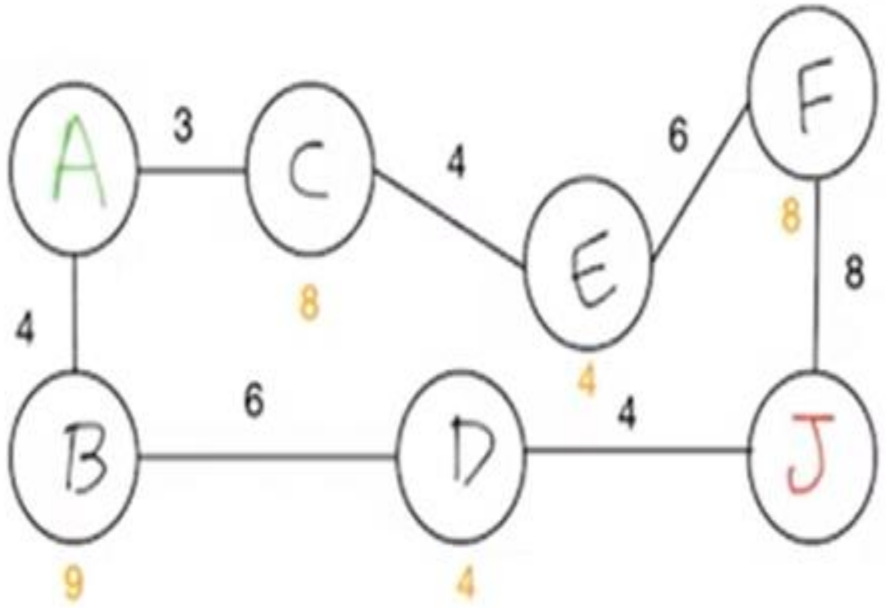
  - where $g(n)$ is the path cost from
  - the initial state to node n and $h(n)$ is the estimated cost of the shortest path from n to a goal state,
- ✓ so, we have
  - UCS keeps solution cost low
  - Best-first helps find solution quickly
  - A* combines these approaches

Example:

A* search , Start = A, Goal = J

1st Step

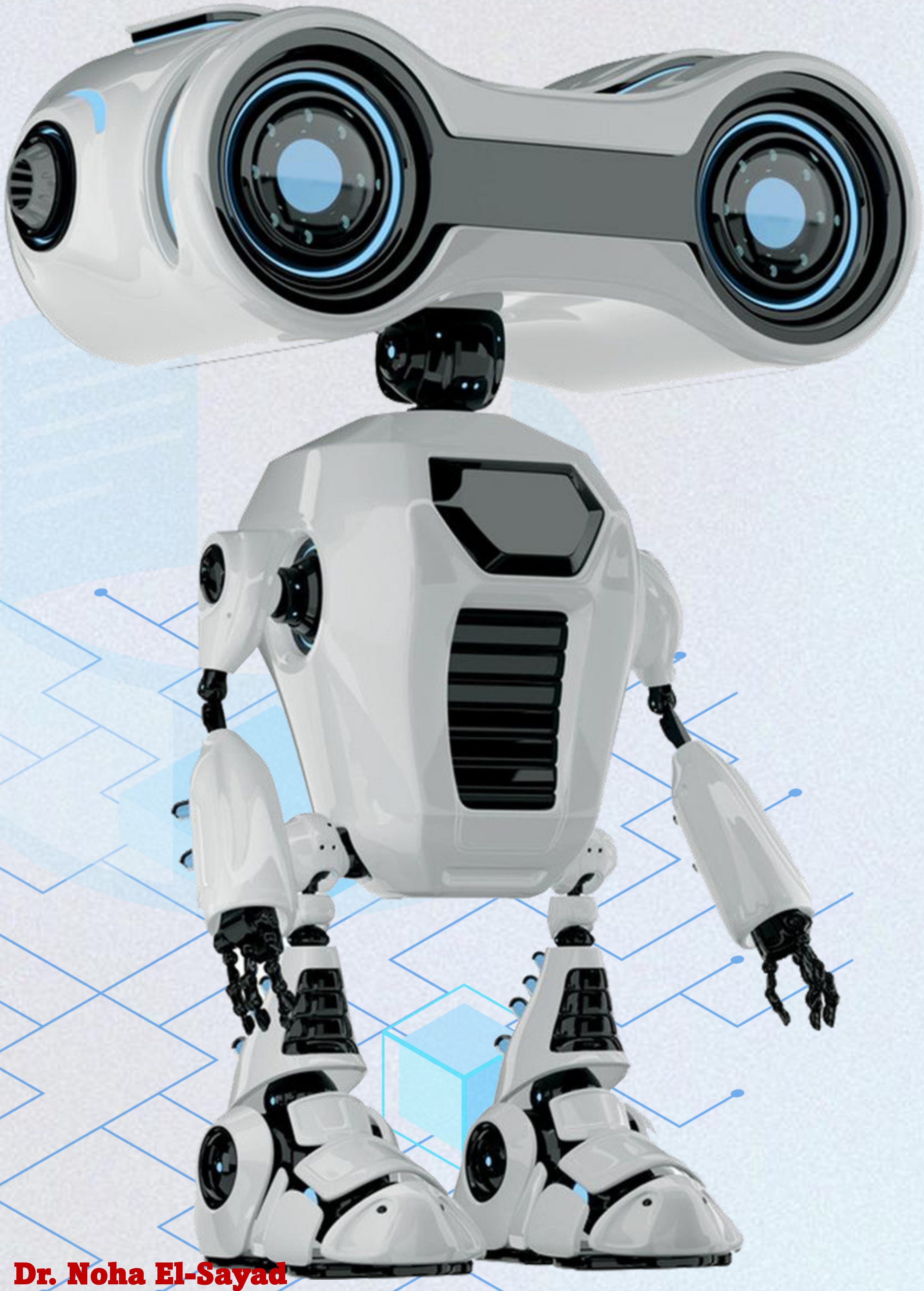
Consider the following graph,



Find the most cost-effective way to reach from start state (A) to end state (F) using A* algorithm.

Solution

THANK YOU!



Bidirectional Search:

Bidirectional Search: A search algorithm that runs two searches simultaneously—one forward from the start state and one backward from the goal state—until they meet at a common node. This approach reduces the search space, covering only half the path compared to traditional methods.