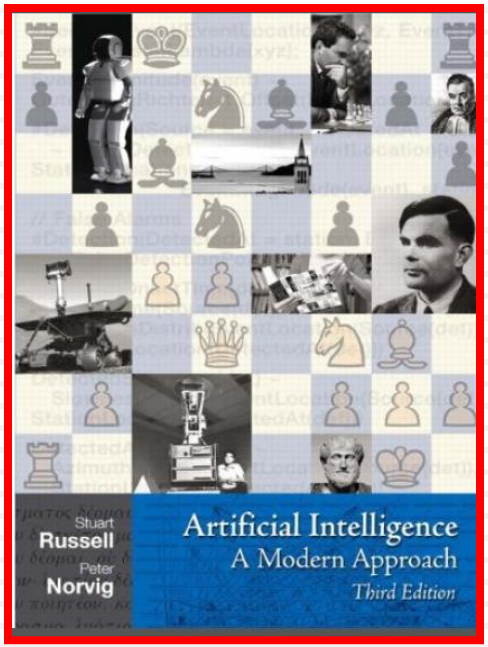
Princess Nora University
Faculty of Computer & Information Systems

ARTIFICIAL INTELLIGENCE

(CS 370D)

Computer Science
Department

جامعة الأميرة نورة بنت عبد الرحمن
Princess Nora Bint Abdul Rahman University



College of Computer and Information Sciences

(CHAPTER-3-PART2)

PROBLEM SOLVING AND SEARCH

Dr. Abeer Mahmoud
(Course coordinator)

Searching algorithm

Uninformed Search Algorithms(Blind Search)

3.1 Breadth first Search
3.2 Depth First Search
3.3 Depth limited Search
3.4 Iterative Deeping Search
3. 5 Bidirectional Search

Informed Search (Heuristic Search)

Best First Search
Greedy Search
Perfect Information Search
A* Search
Iterative Deepening A* Search
A* with PathMax

Dr. Abeer Mahmoud
(course coordinator)

Uninformed Search Algorithms(Blind Search)

1. Breadth first Search
2. Uniform Cost Search (UCS)
3. Depth First Search
4. Depth limited Search
5. Iterative Deeping Search
6. Bidirectional Search

Dr. Abeer Mahmoud
(course coordinator)

Blind Searches - Characteristics

- Simply searches the State Space

- Can only distinguish between a **goal state** and **a non-goal state**

- Blind Searches have no **preference** as to which state (node) that is expanded next.

- The different types of blind searches are characterised by the **order** in which they **expand the nodes**.

Dr. Abeer Mahmoud
(course coordinator)

1-Breadth first Search

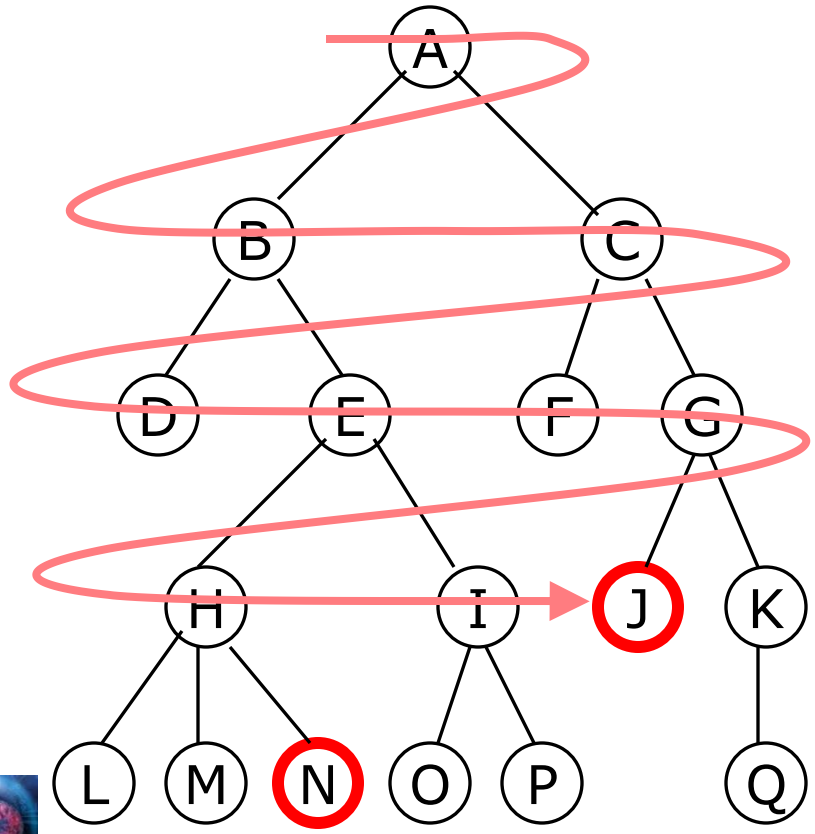
- Use a queuing function that adds nodes to the end of the queue (**FIFO**)
- Expand Root Node First
- Expand all nodes at level **1** before expanding level **2**

OR

- Expand all nodes at level **d** before expanding nodes at level **d+1**
- Assuming all nodes that are visited **first** will be expanded **first**

Dr. Abeer Mahmoud
(course coordinator)

1-Breadth-first searching



- A breadth-first search (BFS) explores nodes nearest the root before exploring nodes further away
- For example, after searching A, then B, then C, the search proceeds with D, E, F, G
- Node are explored in the order A B C D E F G H I J K L M N O P Q
- J will be found before N

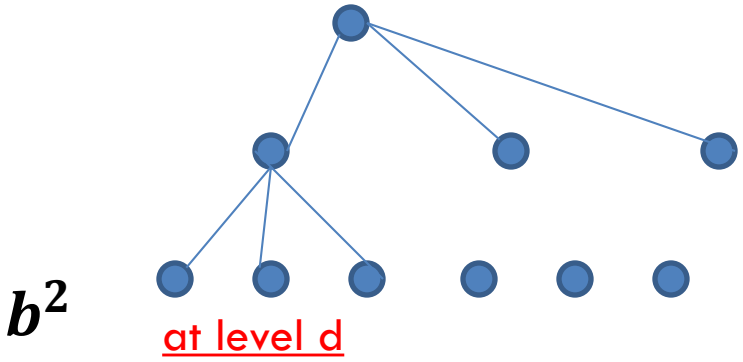
(course coordinator)

Evaluating Breadth First Search

- **Observations**
  - Very systematic
  - **If** there is a solution breadth first search is guaranteed to find it
  - **If** there are several solutions then breadth first search will always find the shallowest (having little depth) goal state first and
  - **if** the cost of a solution is a non-decreasing function of the depth then it will always find the cheapest solution

Dr. Abeer Mahmoud
(course coordinator)

Evaluating Breadth First Search



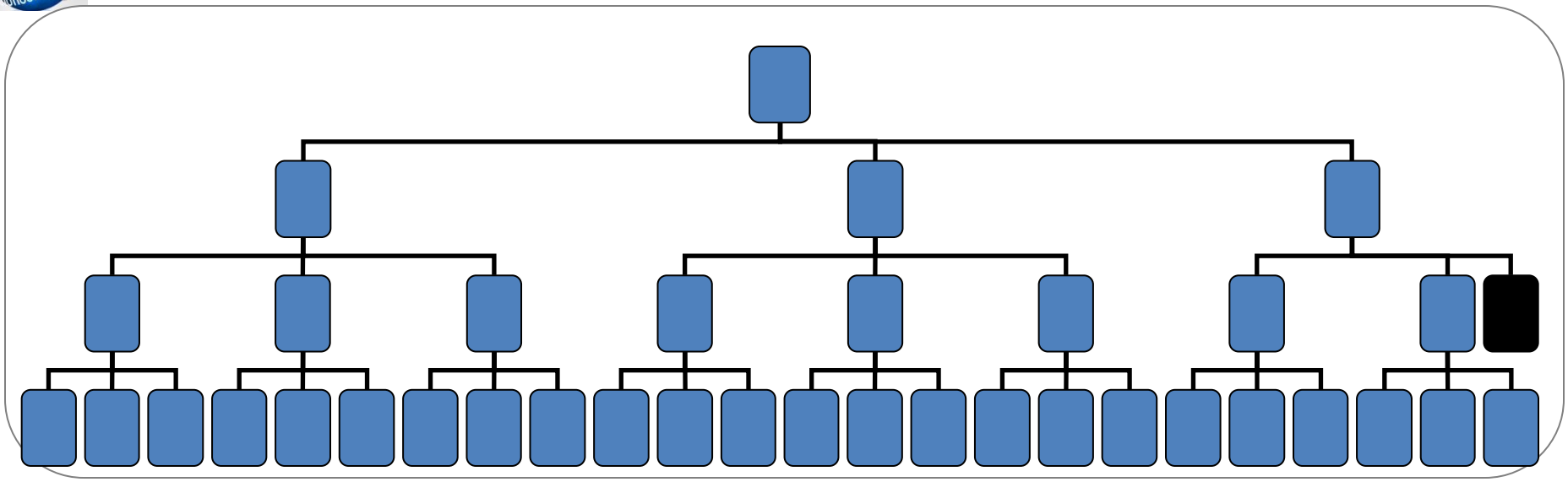
- Space Complexity : $1 + b + b^2 + b^3 + \ldots + b^d$ i.e $O(b^d)$
- Time Complexity : $1 + b + b^2 + b^3 + \ldots + b^d$ i.e. $O(b^d)$

- Where **b** is the branching factor and **d** is the depth of the search tree

- Note : The space/time complexity could be less as the solution could be found anywhere on the **d^{th}** level.

Dr. Abeer Mahmoud
(course coordinator)

Evaluating Breadth First Search



- Every node that is generated must remain in memory so space complexity is therefore as time complexity
- Memory requirements are a bigger problem for breadth first search than is the execution

Dr. Abeer Mahmoud
(course coordinator)

2-Uniform Cost Search (UCS)

- Expand the **cheapest** node.
- **Main idea**: Expand the cheapest node. Where the cost is the path cost g(n).
- **Implementation**: Enqueue nodes in order of cost g(n) in the queue **frontier**(insert in order of increasing path cost).

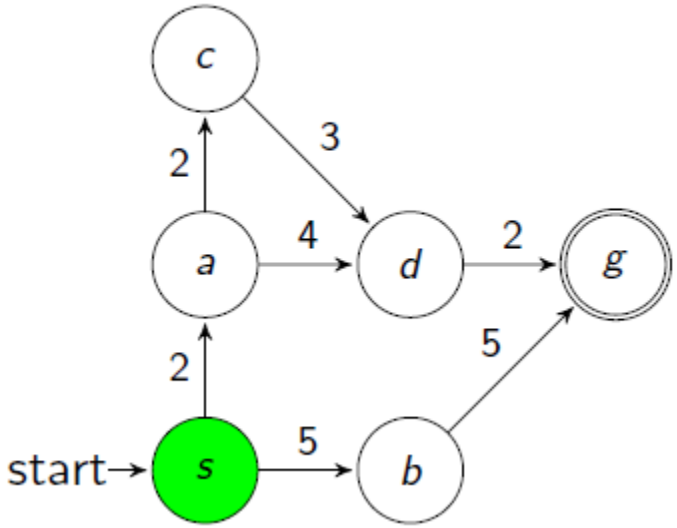
  - **If a node n already exists in Frontier and a new path to n is found with a smaller cost, remove the node n from Frontier and insert the new node n with the new cost into Frontier**

Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

Q:

path	cost
$\langle s \rangle$	0

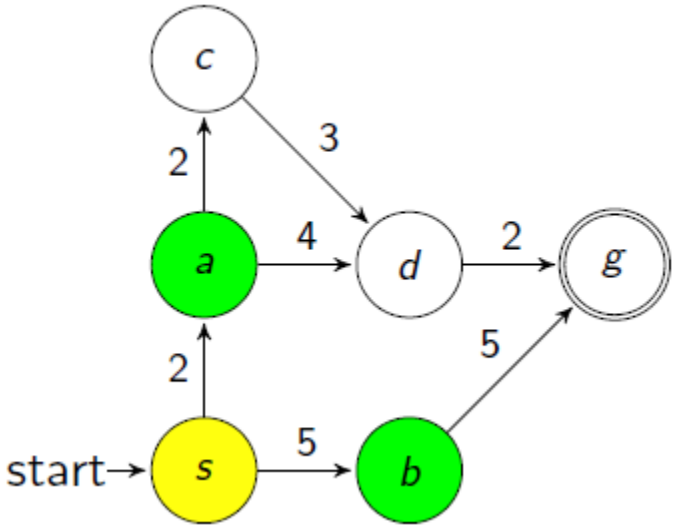


Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

Q:

path	cost
$\langle a, s \rangle$	2
$\langle b, s \rangle$	5

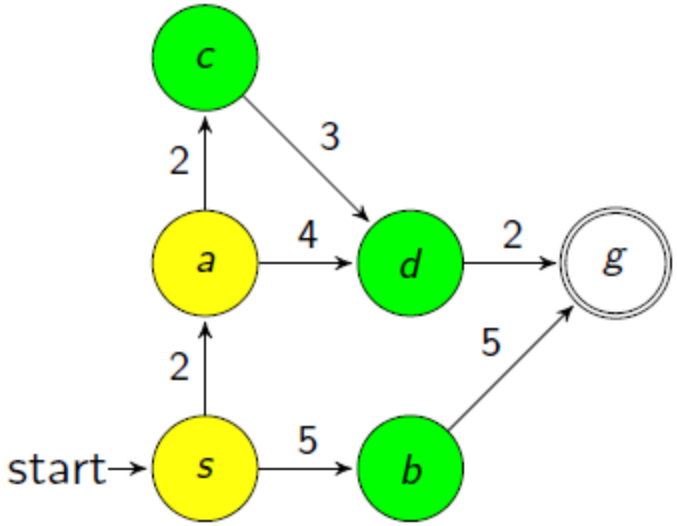


Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

Q:

state	cost
$\langle c, a, s \rangle$	4
$\langle b, s \rangle$	5
$\langle d, a, s \rangle$	6

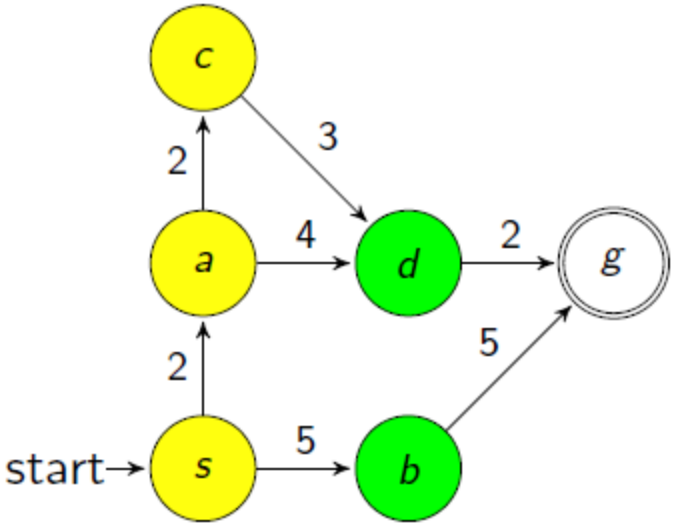


Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

Q:

state	cost
$\langle b, s \rangle$	5
$\langle d, a, s \rangle$	6
$\langle d, c, a, s \rangle$	7

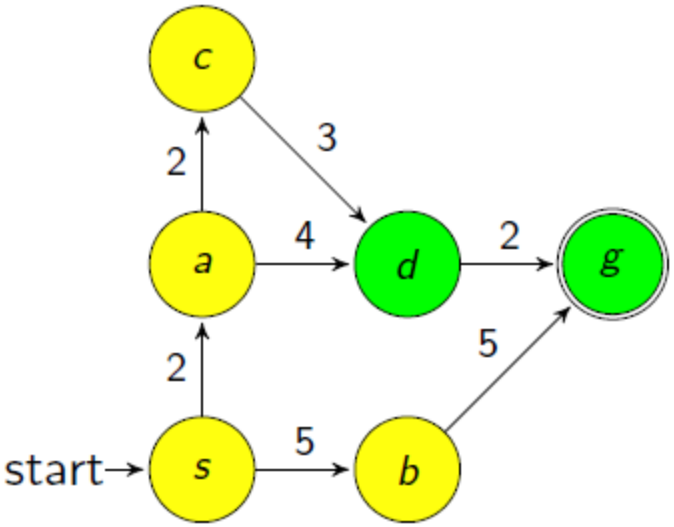


Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

Q:

state	cost
$\langle d, a, s \rangle$	6
$\langle d, c, a, s \rangle$	7
$\langle g, b, s \rangle$	10

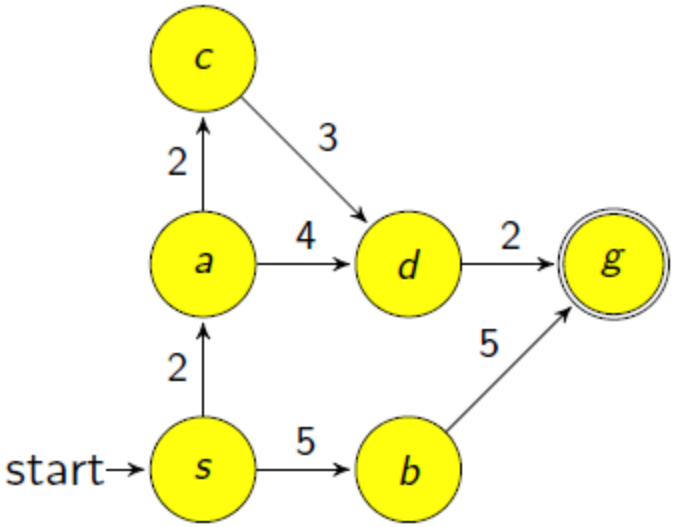


Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

Q:

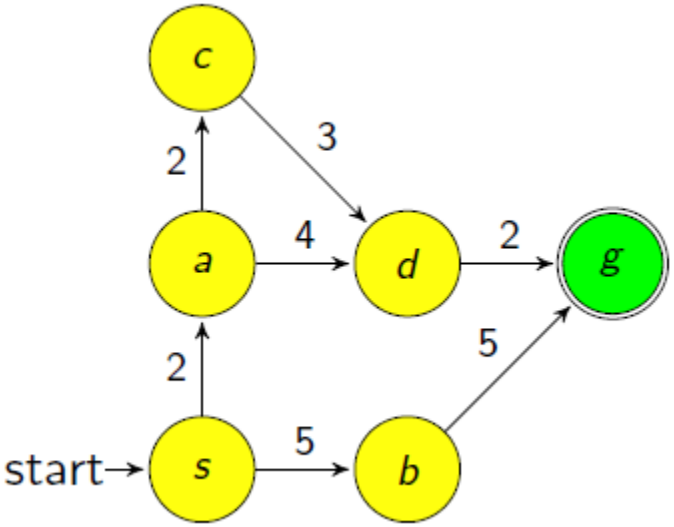
state	cost
$\langle d, c, a, s \rangle$	7
$\langle g, d, a, s \rangle$	8
$\langle g, b, s \rangle$	10



Dr. Abeer Mahmoud
(course coordinator)

Q:

state	cost
$\langle g, d, a, s \rangle$	8
$\langle g, d, c, a, s \rangle$	9
$\langle g, b, s \rangle$	10



Dr. Abeer Mahmoud
(course coordinator)

Uniform Cost Search (UCS)

- UCS is an extension of BFS to the weighted-graph case
- (UCS = BFS if all edges have the same cost).

Complete? Yes

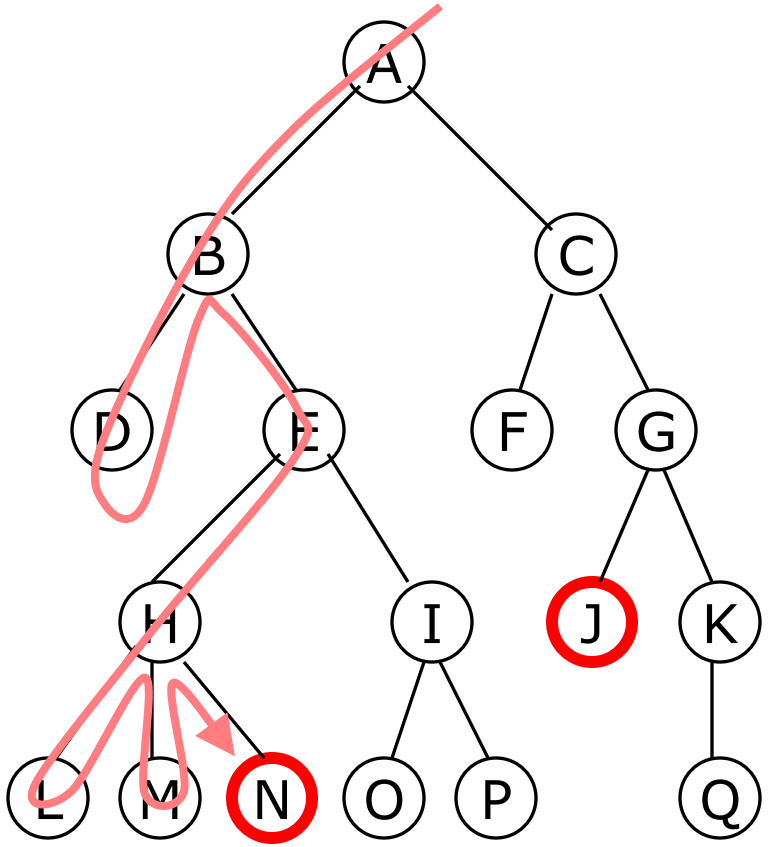
Time? $O(b^d)$

Space? : $O(b^d)$, note that every node in the fringe keep in the queue.

Optimal? Yes

Dr. Abeer Mahmoud
(course coordinator)

3-Depth-first searching

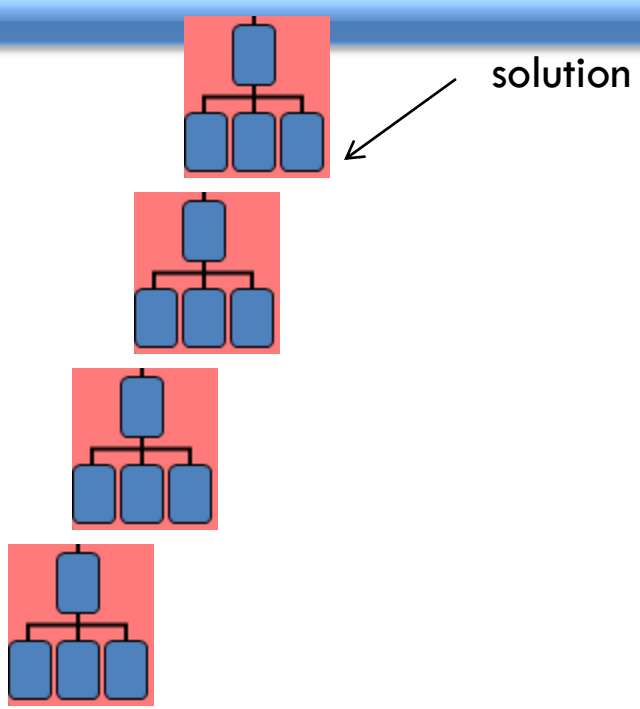


- A depth-first search (DFS) explores a path all the way to a leaf before backtracking and exploring another path
- For example, after searching A, then B, then D, the search **backtracks** and tries another path from B
- Node are explored in the order A B D B E H L H M H N H E I O I P I E B A C F C G J G K Q
- N will be found before J

- Always expands the **deepest** node in the current fringe
- The search proceeds immediately to deepest level of search tree , where nodes have no successors
- As those nodes are expanded they are dropped from fringe
- Can be implemented by **LIFO** queue (stack)
- For memory , this algorithm need only to store one path

Dr. Abeer Mahmoud
(course coordinator)

4-Depth-limited search



- The depth first search disadvantage is that the algorithm go deep and deep while solution may be near root
- It is depth-first search with an imposed **limit** on the depth of exploration, to guarantee that the algorithm ends.

Depth-limited search

= depth-first search with depth limit *l*,

i.e., nodes at depth *l* have no successors

- Recursive implementation:

```
function DEPTH-LIMITED-SEARCH( problem, limit) returns soln/fail/cutoff
    RECURSIVE-DLS(MAKE-NODE(INITIAL-STATE[problem]), problem, limit)

function RECURSIVE-DLS(node, problem, limit) returns soln/fail/cutoff
    cutoff-occurred? ← false
    if GOAL-TEST[problem](STATE[node]) then return SOLUTION(node)
    else if DEPTH[node] = limit then return cutoff
    else for each successor in EXPAND(node, problem) do
        result ← RECURSIVE-DLS(successor, problem, limit)
        if result = cutoff then cutoff-occurred? ← true
        else if result ≠ failure then return result
    if cutoff-occurred? then return cutoff else return failure
```

5-Iterative Deeping Search

- The algorithm consists of iterative, depth-first searches, with a maximum depth that increases at each iteration. Maximum depth at the beginning is 1.
- Only the actual path is kept in memory; nodes are regenerated at each iteration.
- DFS problems related to infinite branches are avoided.
- To guarantee that the algorithm ends if there is no solution, a general maximum depth of exploration can be defined.

Dr. Abeer Mahmoud
(course coordinator)

5-Iterative Deeping Search

- Depth-limited search, with
  - depth = 0
  - then again with depth = 1
  - then again with depth = 2
  - ... until you find a solution

Dr. Abeer Mahmoud
(course coordinator)

5-Iterative Deeping Search

Depth = 0

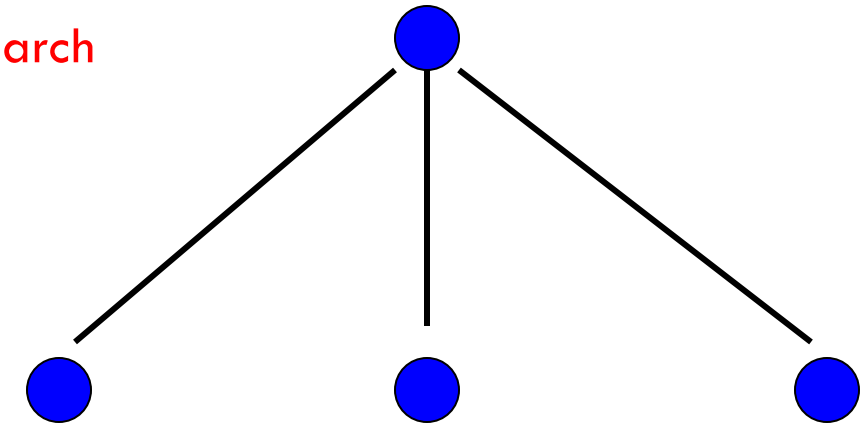
Depth Limit = 0

Dr. Abeer Mahmoud
(course coordinator)

5-Iterative Deeping Search

Depth = 0

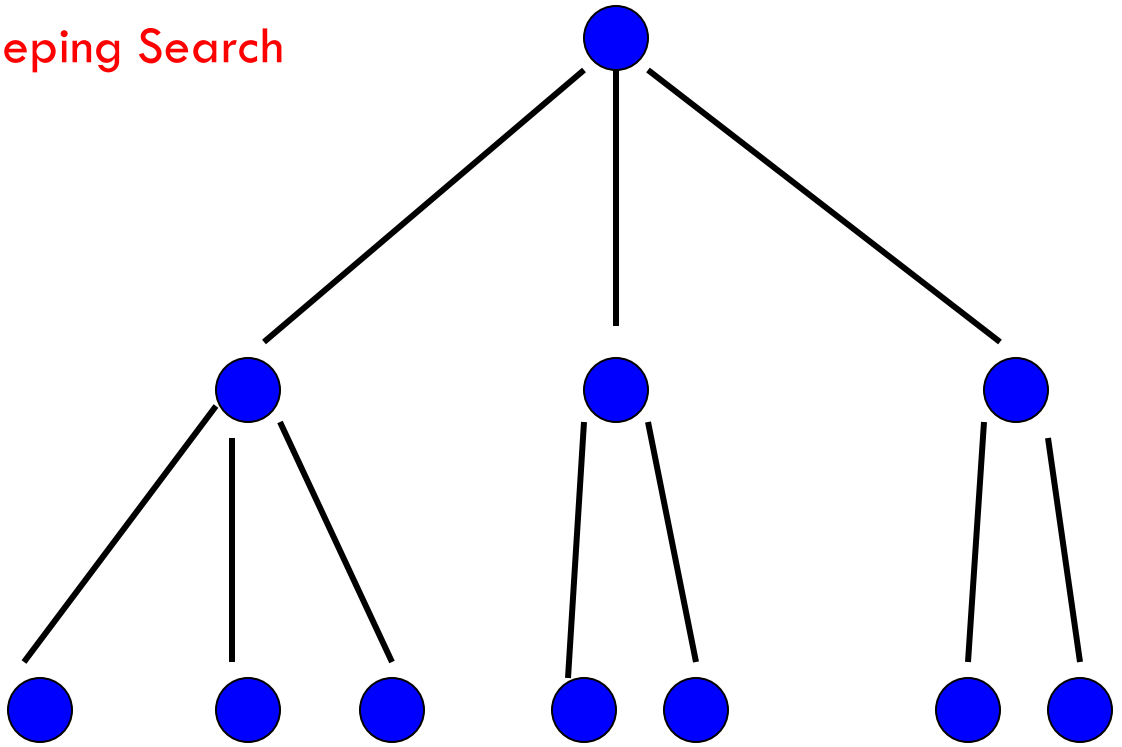
Depth Limit = 1



Dr. Abeer Mahmoud
(course coordinator)

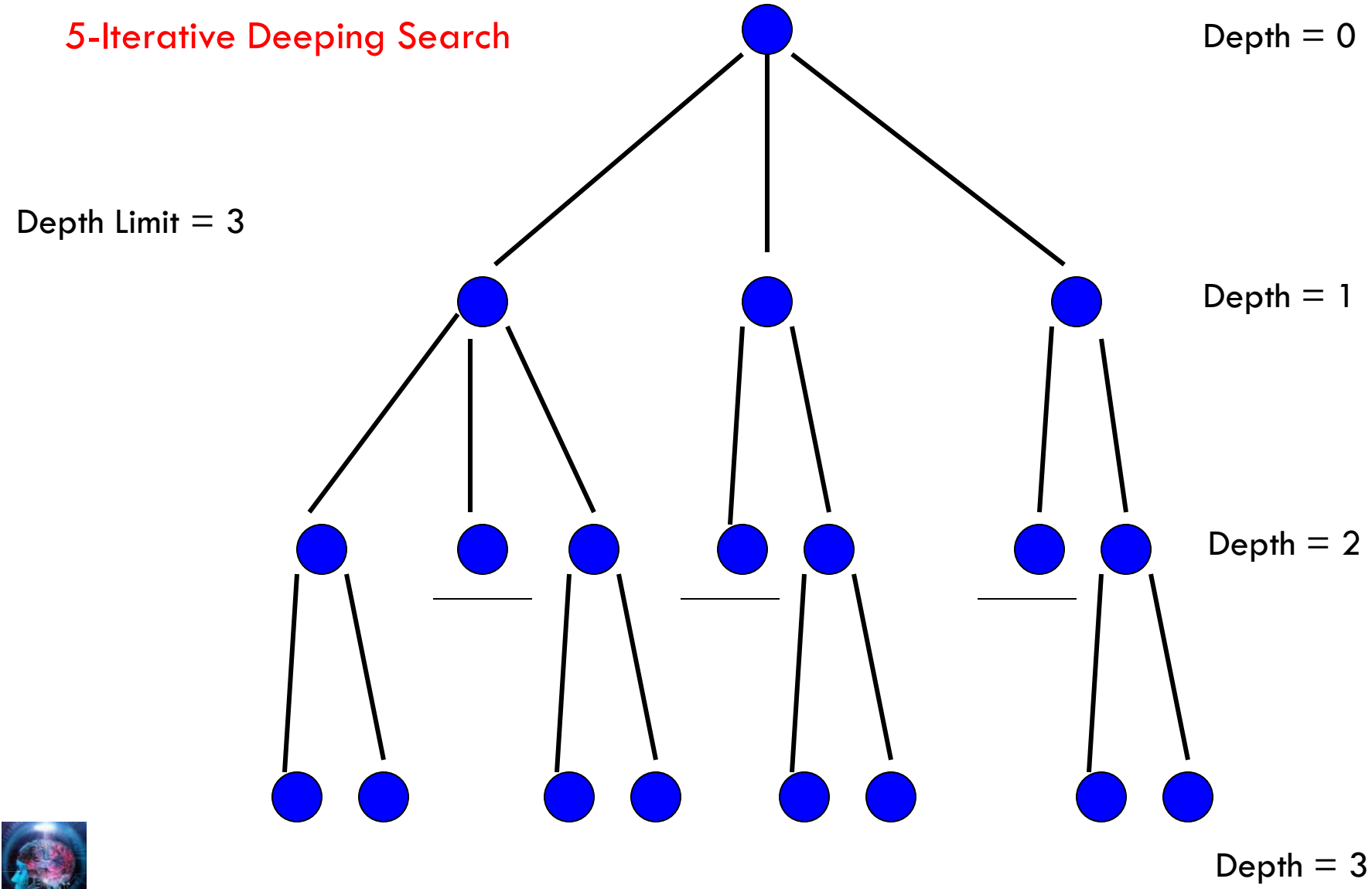
5-Iterative Deeping Search

Depth Limit = 2



Dr. Abeer Mahmoud
(course coordinator)

5-Iterative Deeping Search



Dr. Abeer Mahmoud
(course coordinator)

6-Bidirectional Search

- Start searching forward from initial state and backwards from goal, and try to meet in the middle

Dr. Abeer Mahmoud
(course coordinator)

Thank you

End of Chapter 3-Part2

Dr. Abeer Mahmoud
(course coordinator)