Problem Solving by Searching

Faculty of DS & AI
Autumn semester, 2025

Trong-Nghia Nguyen

nghiant@neu.edu.vn

Business AI Lab

2025-08

Content

- Problem-Solving Agents
- Example Problems
- Solving the Problem by Searching

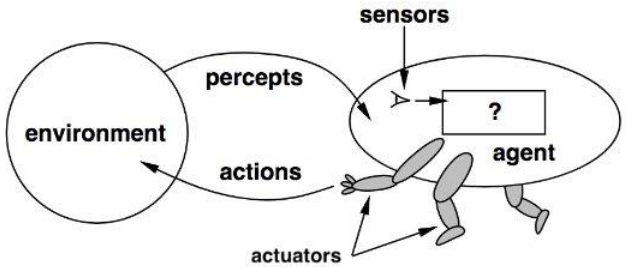
Content

- Problem-Solving Agents
- Example Problems
- Solving the Problem by Searching

Problem-Solving Agents

Four Steps of Problem-Solving Process

- Goal formulation
- Problem formulation
- Search for the Solution
- Execute the Solution



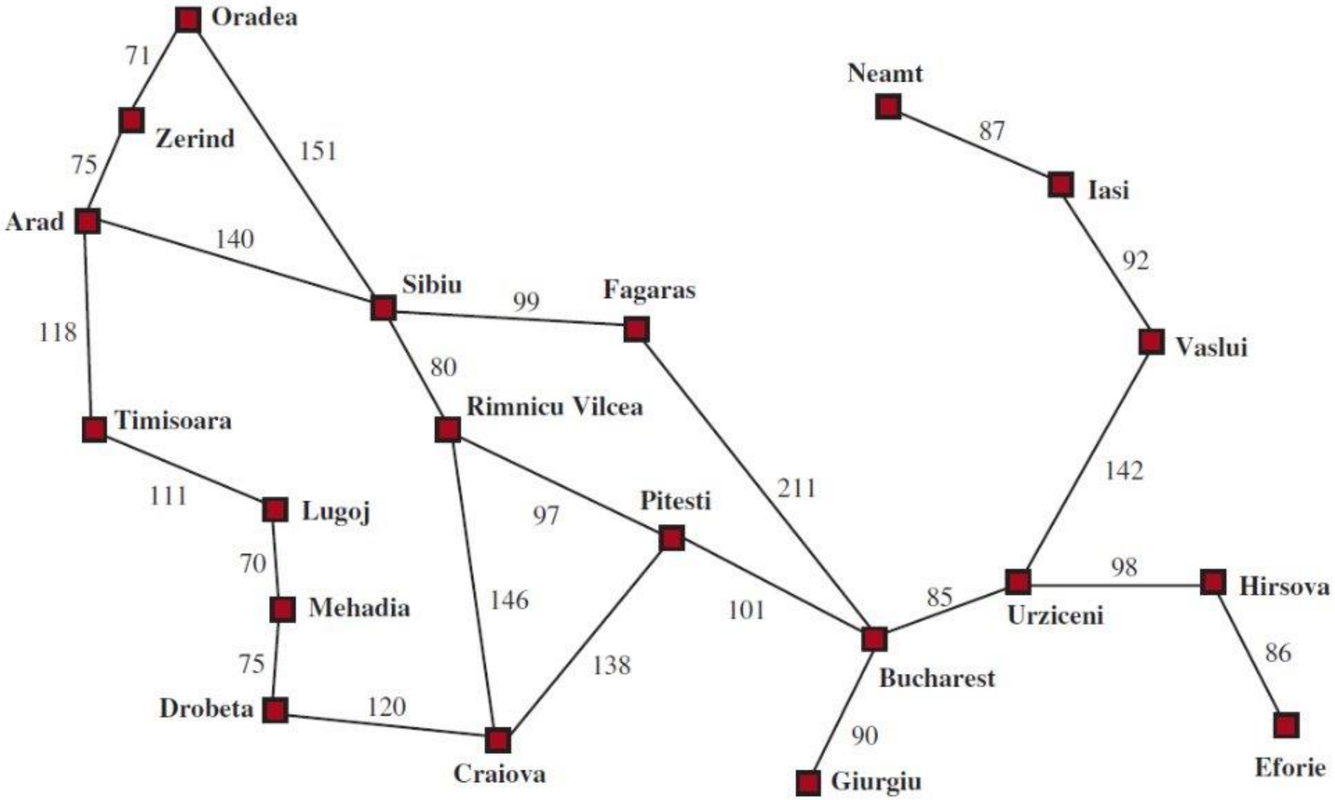


Figure 3.1 A simplified road map of part of Romania, with road distances in miles.

Problem-Solving Agents

Problem Formulation - Search Problem

- set of **states,** or state space
  - > **initial** state, **goal** state, …
- set of **actions**
- **transition model** for **state space graph**
- action **cost** function, f(s, a, s')

Solution

- a **path** from the initial state to goal state
- optimal solution: optimal path

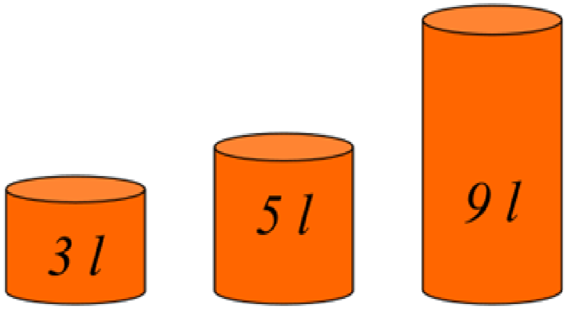
Content

- Problem-Solving Agents
- Example Problems
- Solving the Problem by Searching

Example Problems

Water measurement problem

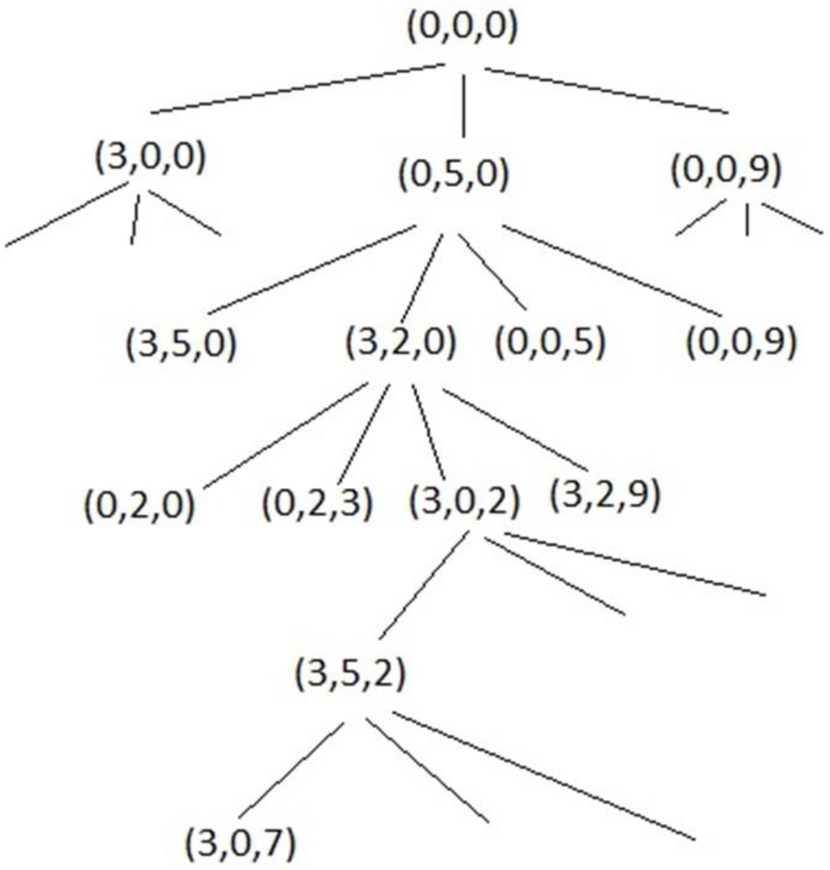
- Using three jugs of 3 liters, 5 liters, and 9 liters, how can we measure exactly 7 liters of water?
- **State:** Water amounts in the 3 jugs respectively: a, b, c (a ≤ 3, b ≤ 5, c ≤ 9), where the triple (a, b, c) represents the problem state
- **Initial State:** (0, 0, 0) // all three jugs are empty
- **Goal State:** (-, -, 7) // the third jug contains 7 liters
- **State Transitions:** From state (a,b,c) to state (x,y,z) through operations such as emptying a jug, or pouring from one jug to another until the source jug is empty or the destination jug is full
- **Cost:** Each state transition has cost 1



Example Problems

Water measurement problem

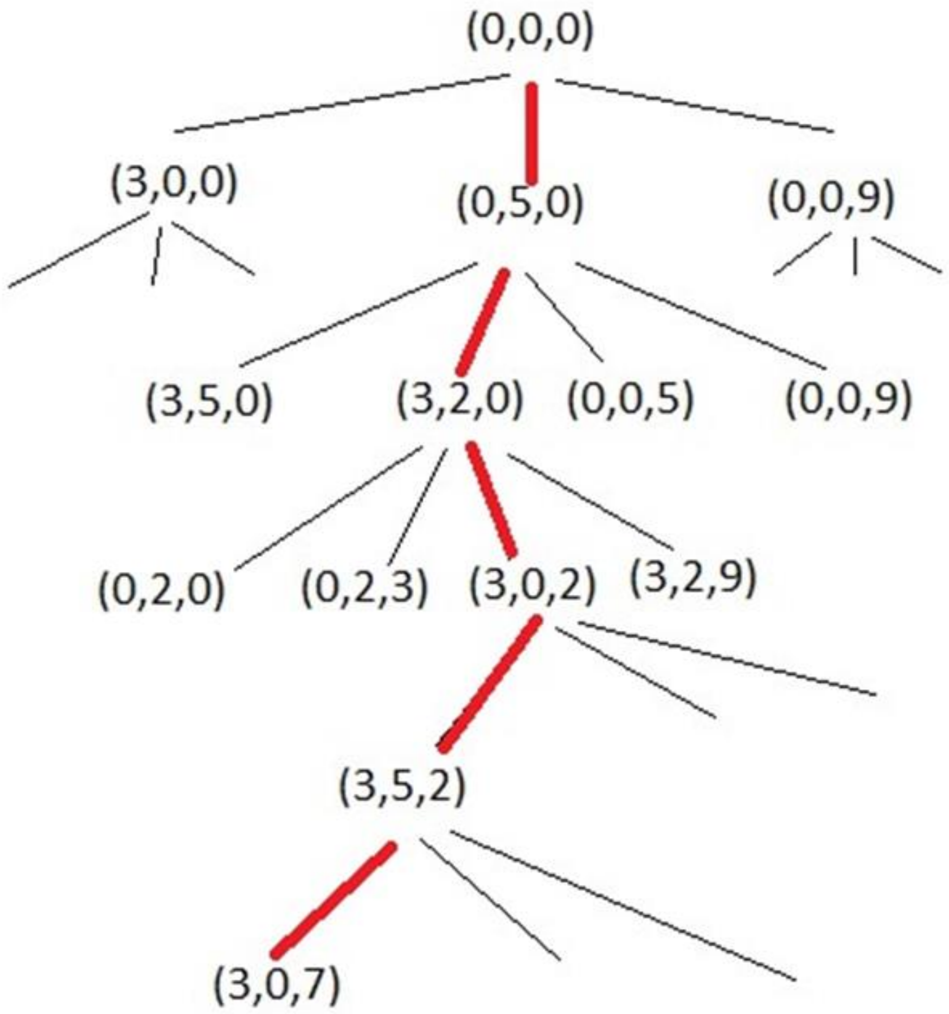
- State space



Example Problems

Water measurement problem

- Solution with cost = 5



Example Problems

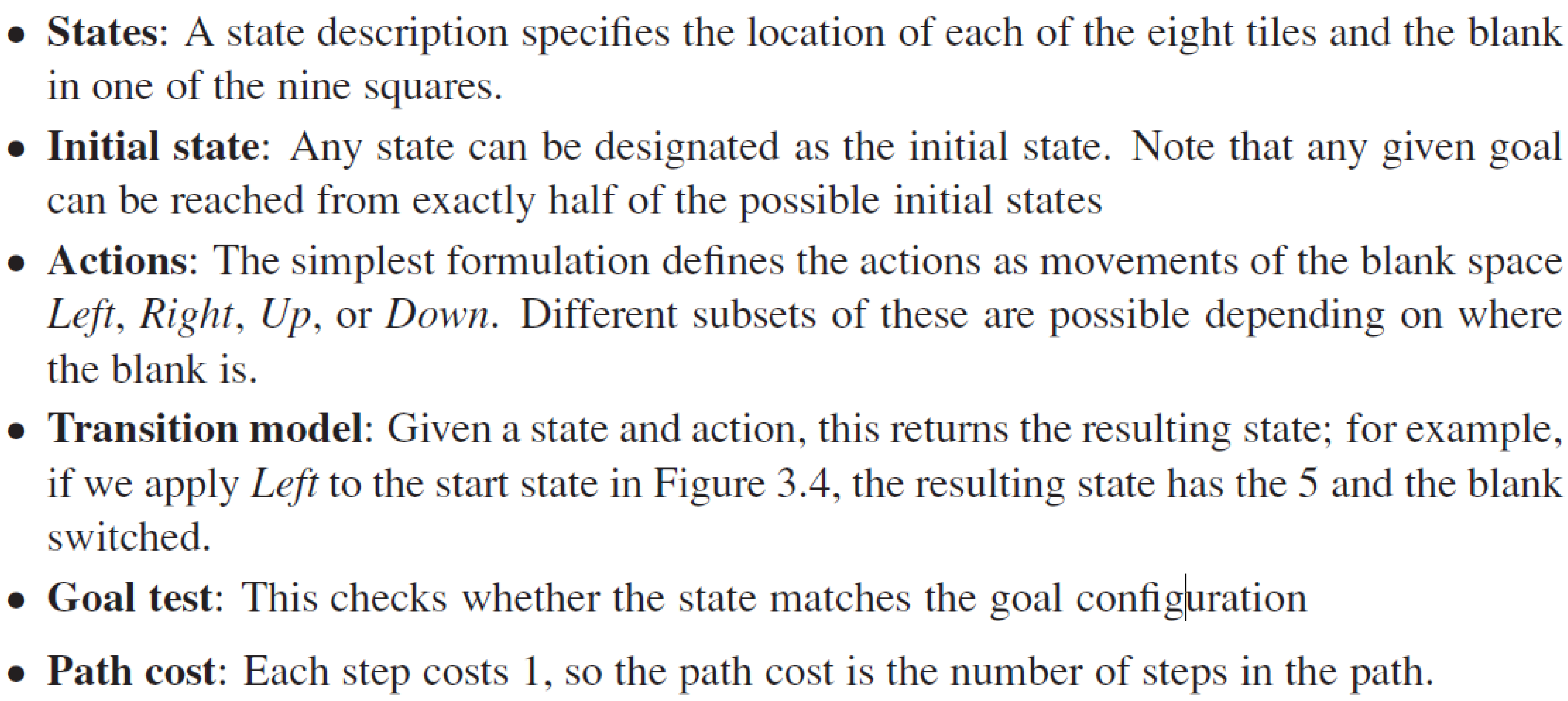
8-Puzzle Problem

2	8	3
1	6	4
7		5

Initial State

1	2	3
8		4
7	6	5

Goal State



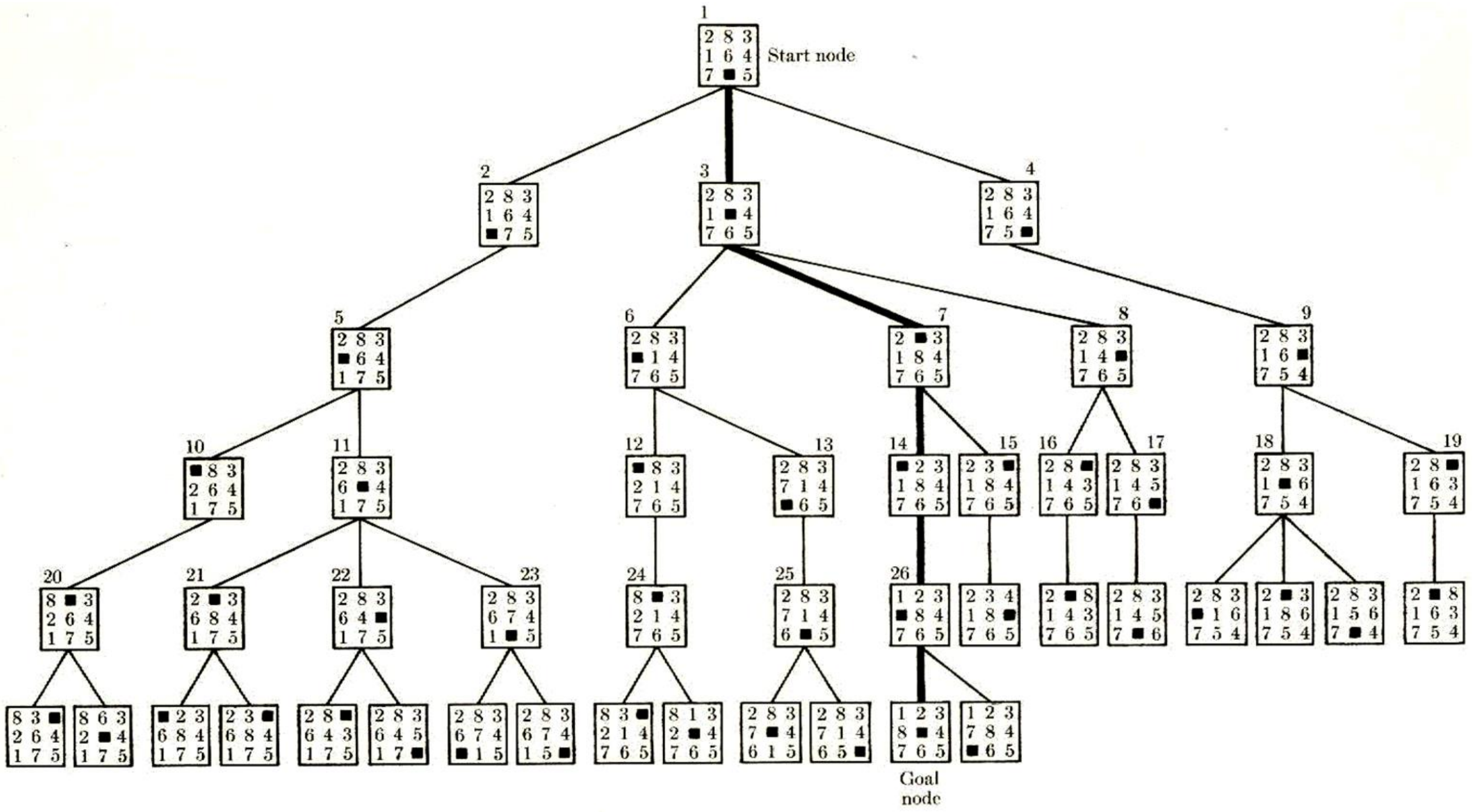
- **States**: A state description specifies the location of each of the eight tiles and the blank in one of the nine squares.
- **Initial state**: Any state can be designated as the initial state. Note that any given goal can be reached from exactly half of the possible initial states
- **Actions**: The simplest formulation defines the actions as movements of the blank space *Left*, *Right*, *Up*, or *Down*. Different subsets of these are possible depending on where the blank is.
- **Transition model**: Given a state and action, this returns the resulting state; for example, if we apply *Left* to the start state in Figure 3.4, the resulting state has the 5 and the blank switched.
- **Goal test**: This checks whether the state matches the goal configuration
- **Path cost**: Each step costs 1, so the path cost is the number of steps in the path.

Example Problems

8-Puzzle Problem

State space of the problem

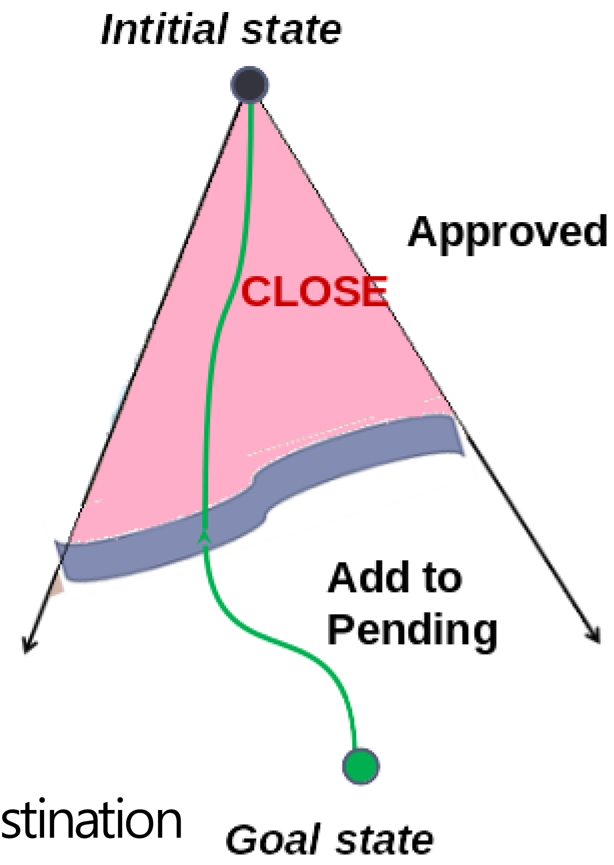
- Each node is a state
- Each edge corresponds to a state transition



Content

- Problem-Solving Agents
- Example Problems
- Solving the Problem by Searching

Search Algorithms



Method

- Is to find a path from the Initial state to a goal state in the state space

  - Solution: set of transformations associated with the found path.

- The search strategies differ by the order of development of the nodes tc choose a next state of to be traversed)

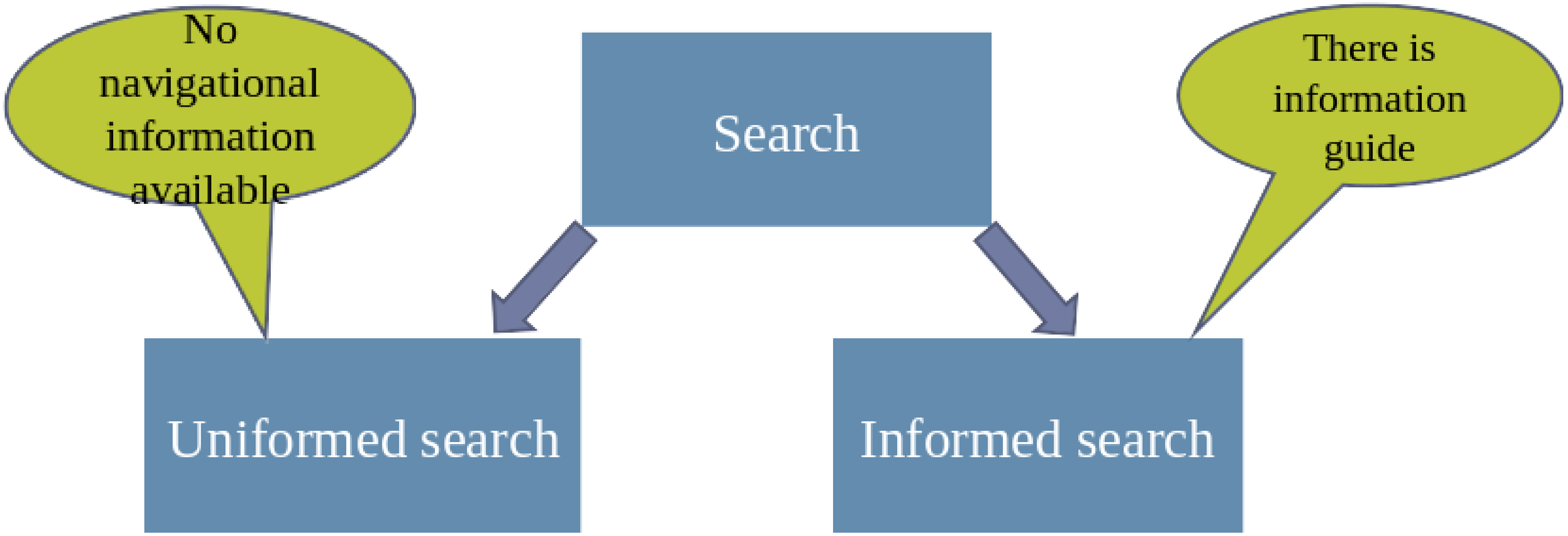
- Information used to find a path:

  - Past: estimate the path cost from the root to the currently traversed node

  - Future: estimate the path cost from the currently traversed node to the destination

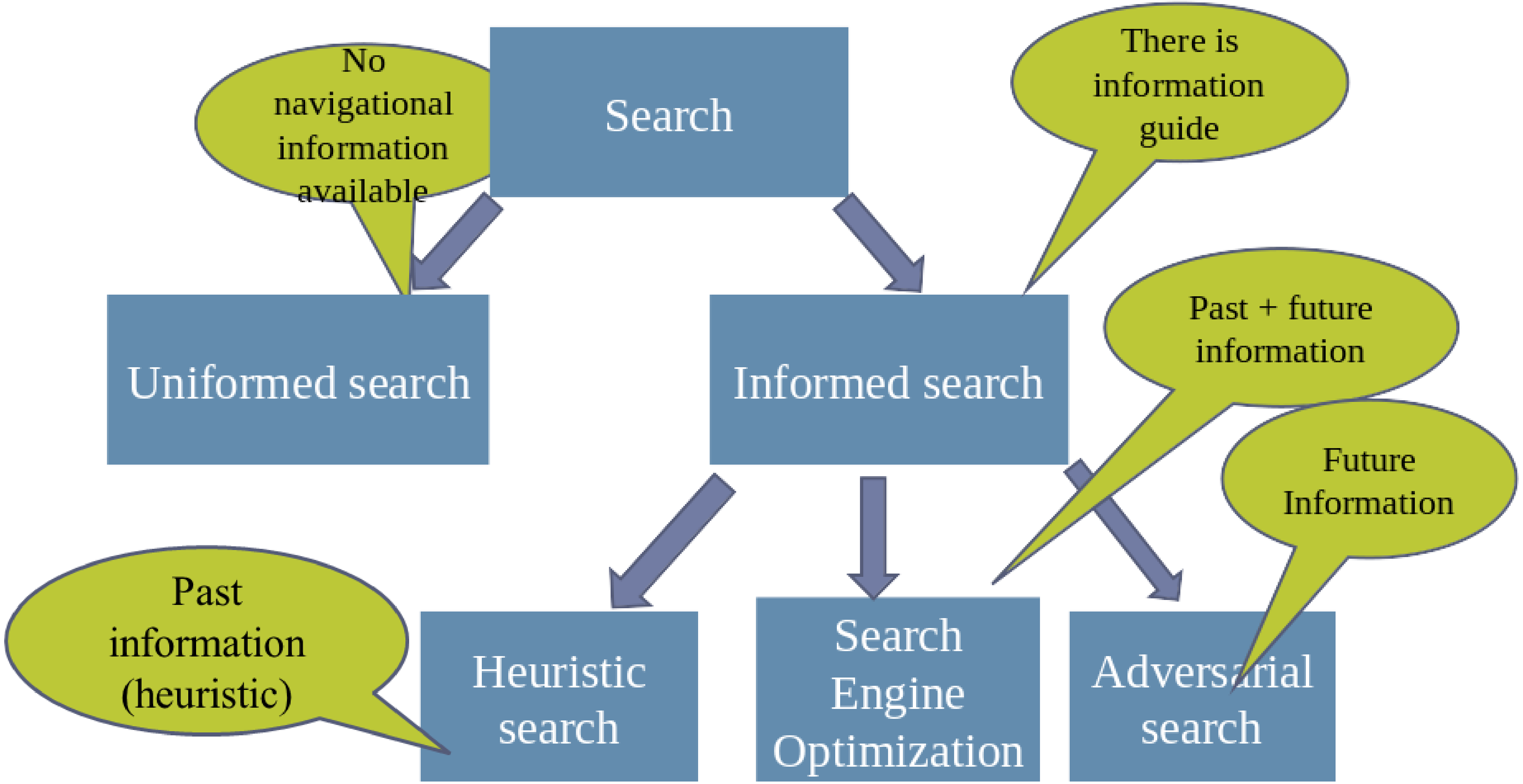
Search Algorithms

Method


No navigational information available
Search
There is information guide
Uniformed search
Informed search

Search Algorithms

Method

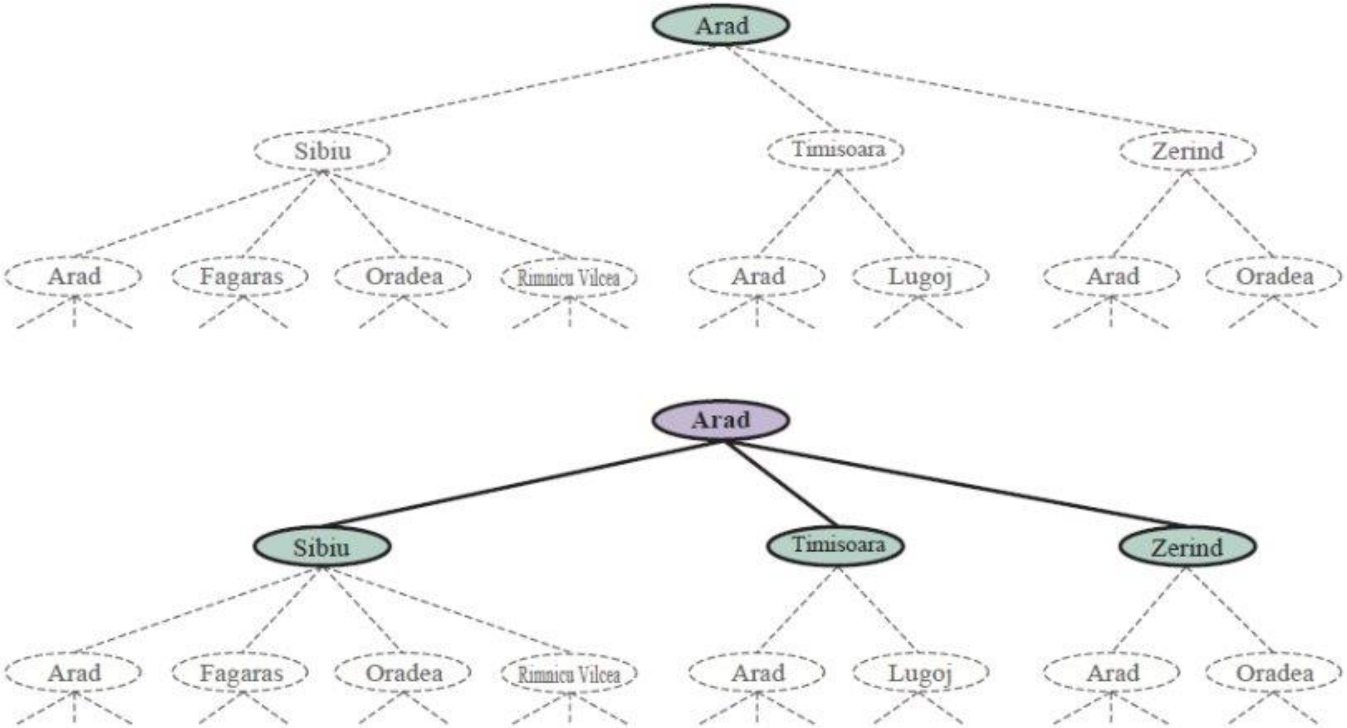


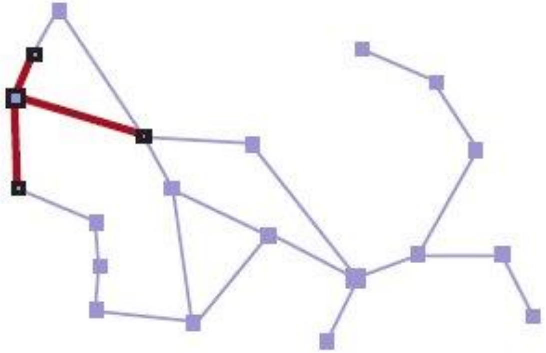
Search
No navigational information available
There is information guide
Uniformed search
Informed search
Past + future information
Future Information
Past information (heuristic)
Heuristic search
Search Engine Optimization
Adversarial search

Search Algorithms

Select & Expand Strategy

- select among green nodes
- and then expand (lavender nodes)





Search Algorithms

Select & Expand Strategy

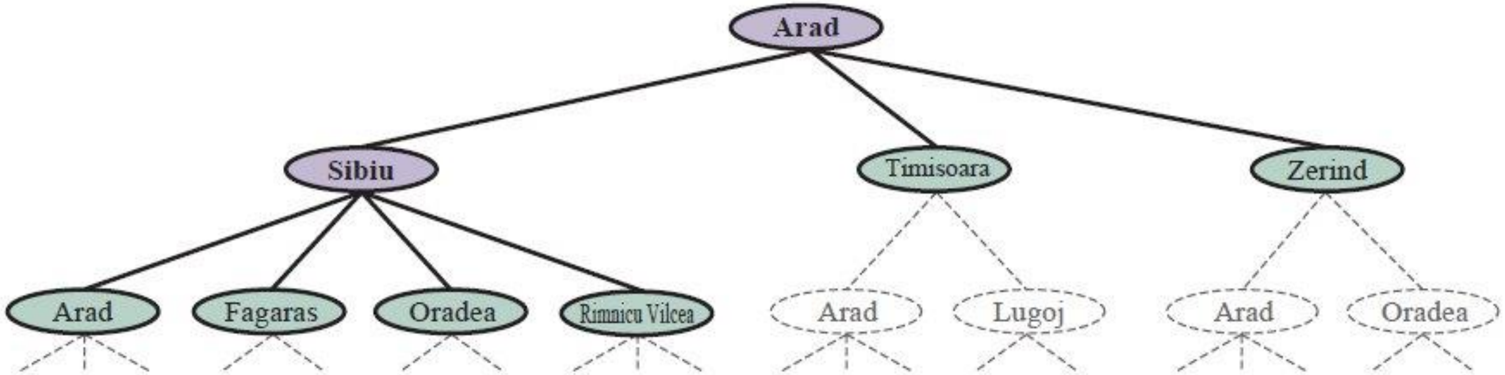
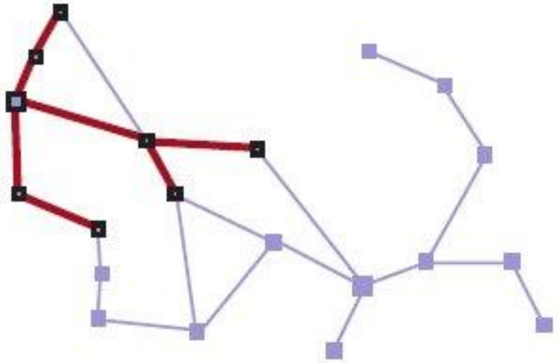
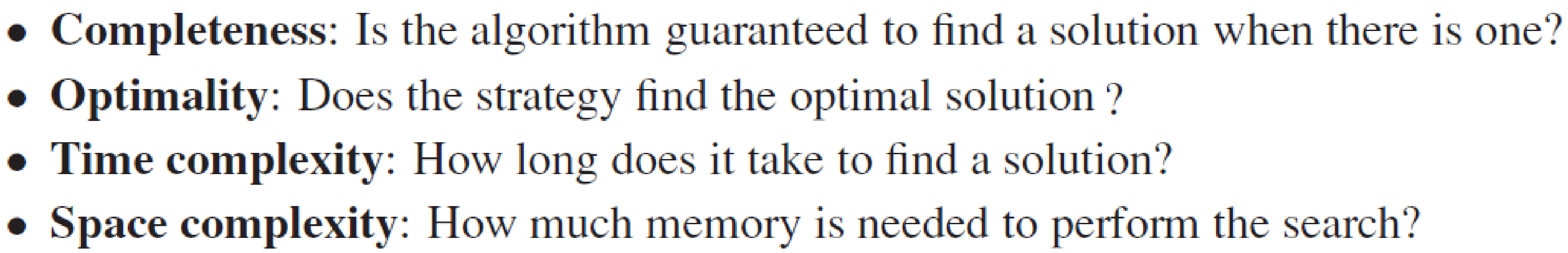


Figure 3.4 Three partial search trees for finding a route from Arad to Bucharest. Nodes that have been *expanded* are lavender with bold letters; nodes on the frontier that have been *generated* but not yet expanded are in green; the set of states corresponding to these two types of nodes are said to have been *reached*. Nodes that could be generated next are shown in faint dashed lines.



Search Algorithms

Performance Measurement



- **Completeness**: Is the algorithm guaranteed to find a solution when there is one?
- **Optimality**: Does the strategy find the optimal solution ?
- **Time complexity**: How long does it take to find a solution?
- **Space complexity**: How much memory is needed to perform the search?

Uninformed Search Strategies

Breadth-first search (BFS)

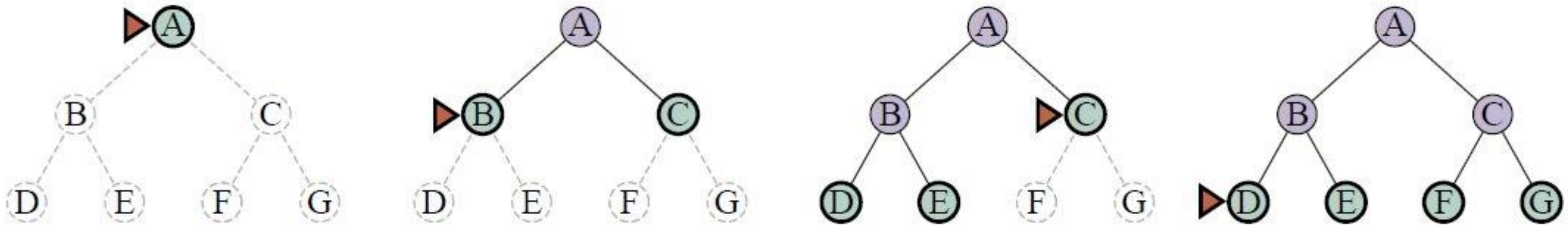


Figure 3.8 Breadth-first search on a simple binary tree. At each stage, the node to be expanded next is indicated by the triangular marker.

A variation:

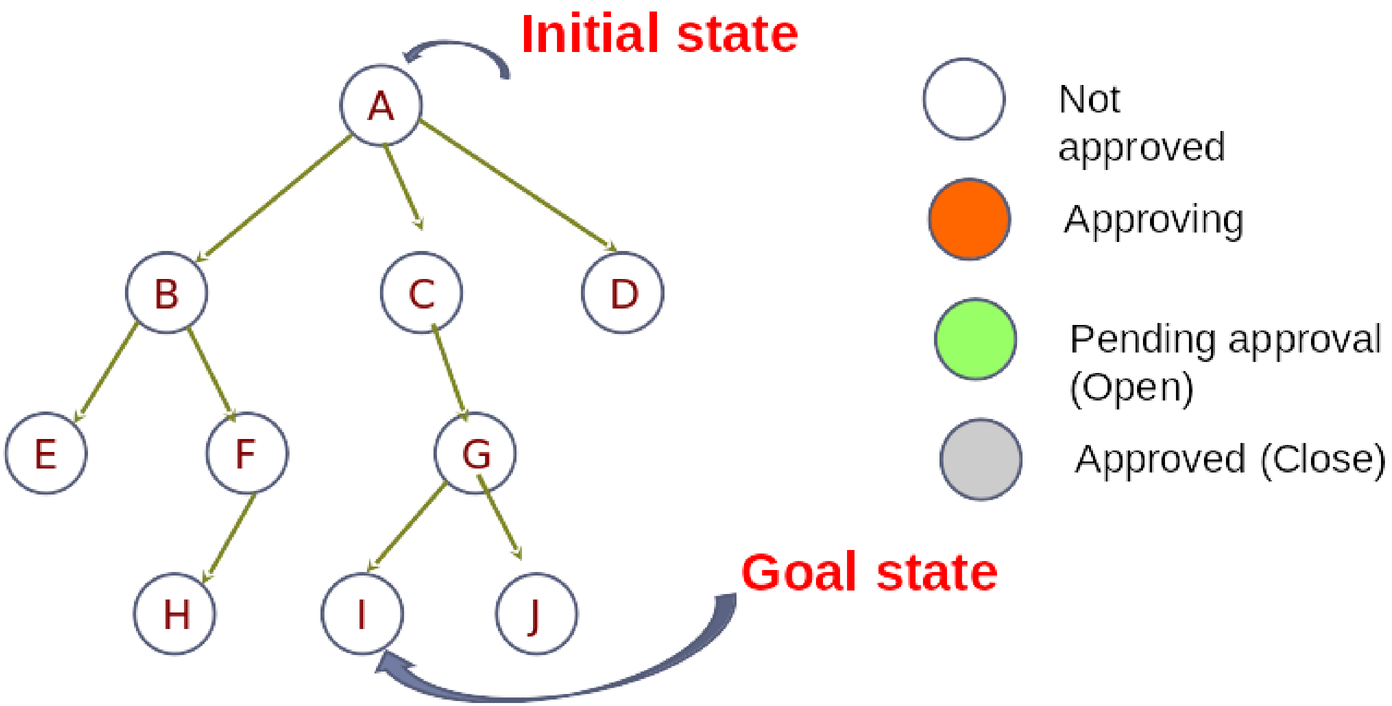
- uniform-cost search

expands the node n with the *lowest path cost* $g(n)$

Uninformed Search Strategies

Breadth-first search

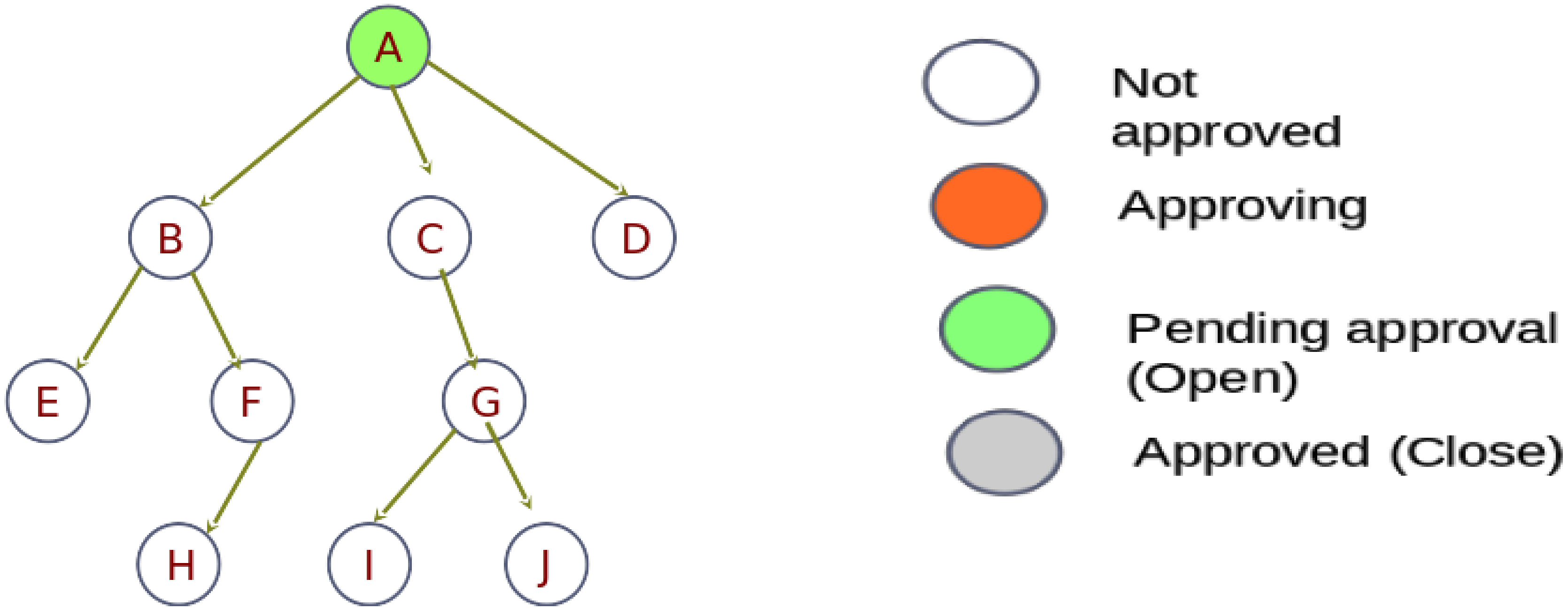
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



○ Not approved

🟧 Approving

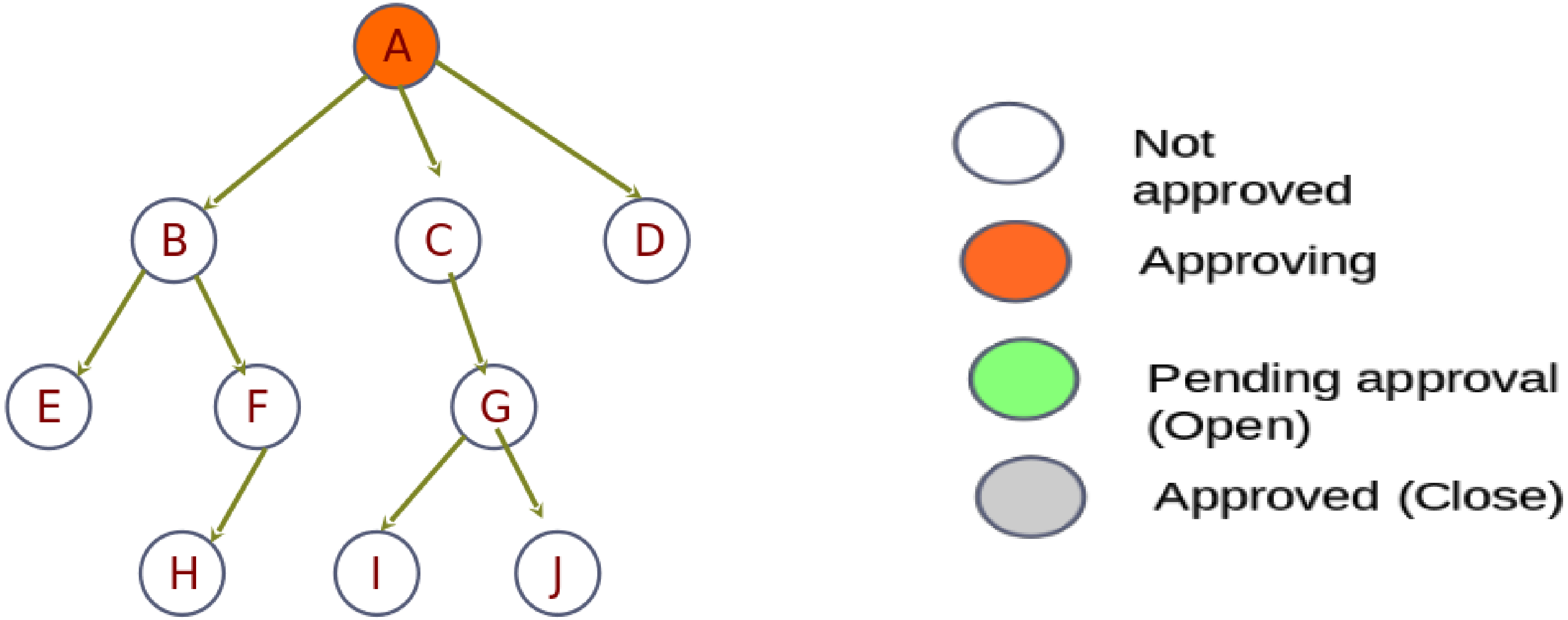
🟩 Pending approval (Open)

● Approved (Close)

Uninformed Search Strategies

Breadth-first search

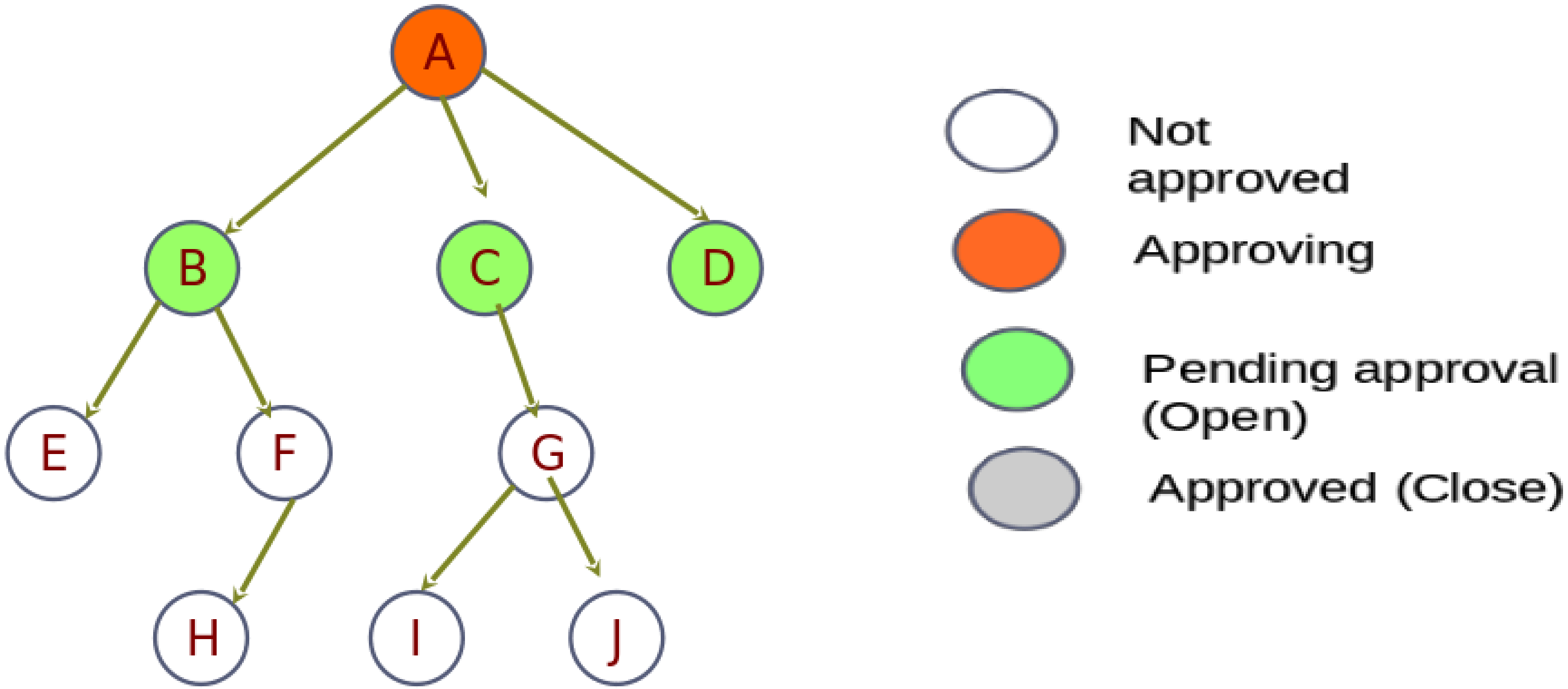
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

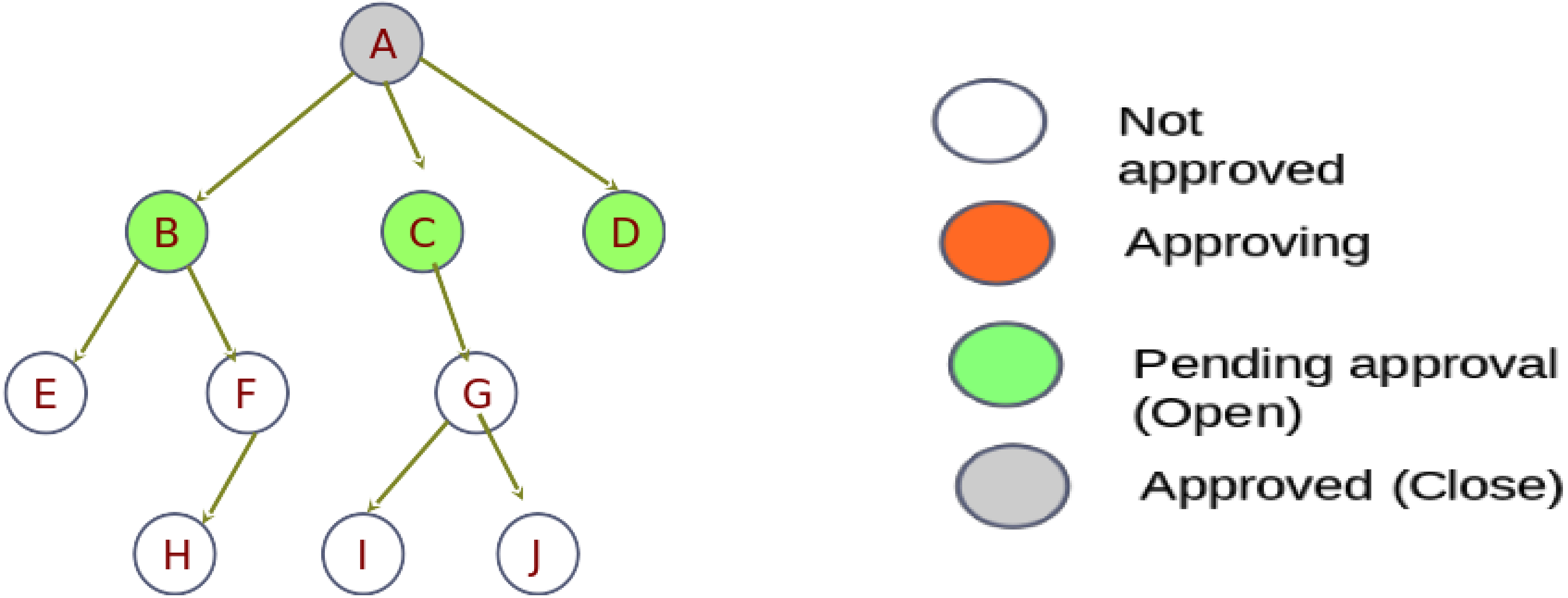
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

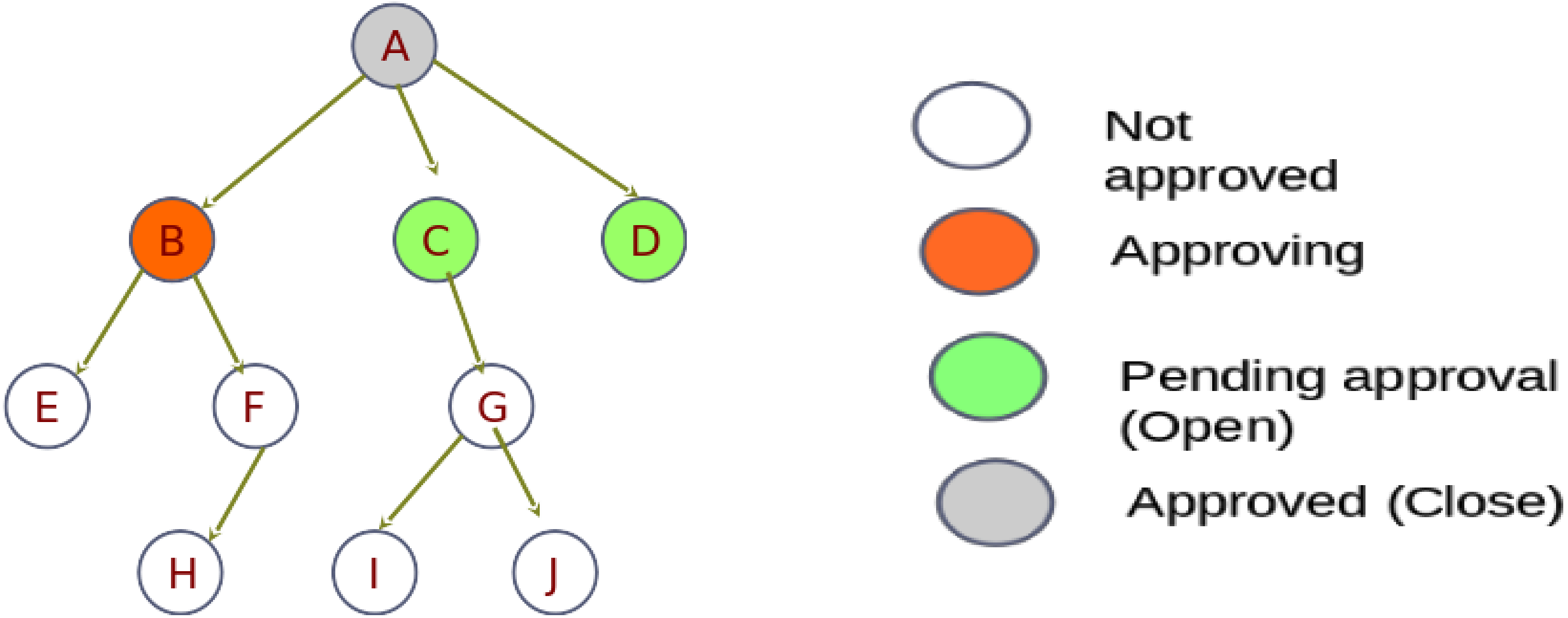
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

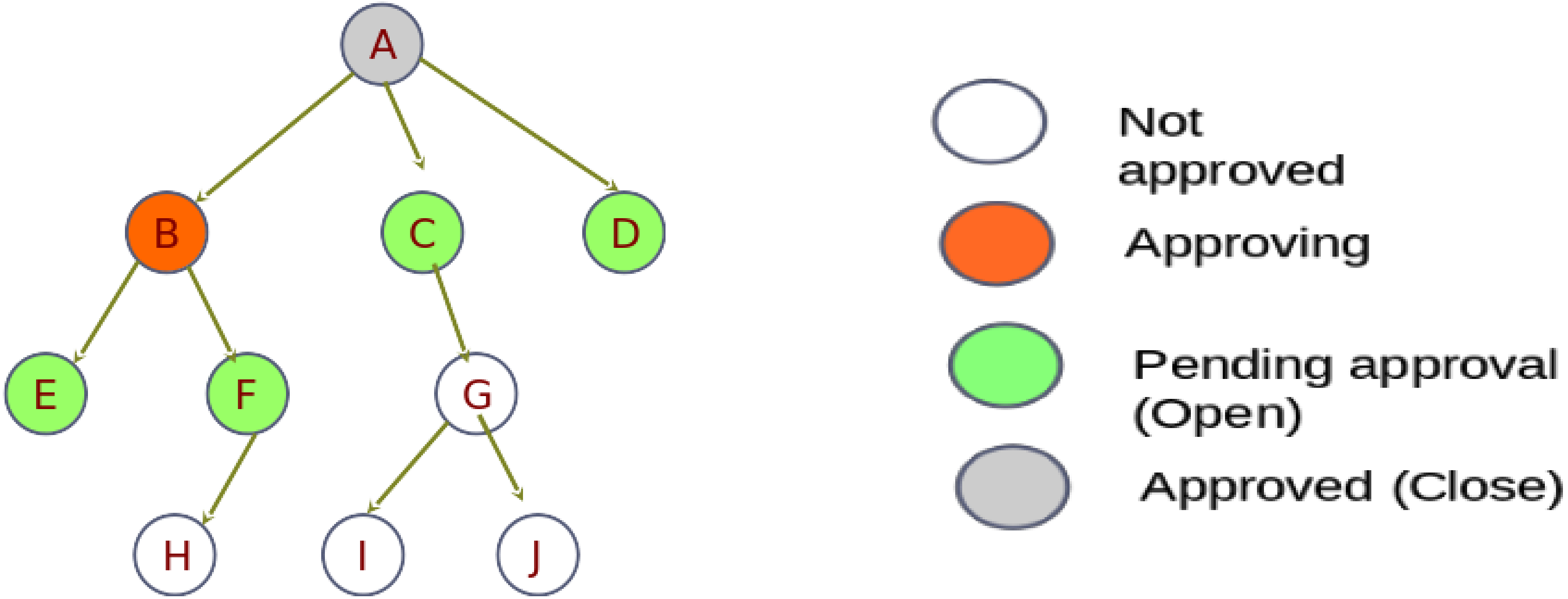
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

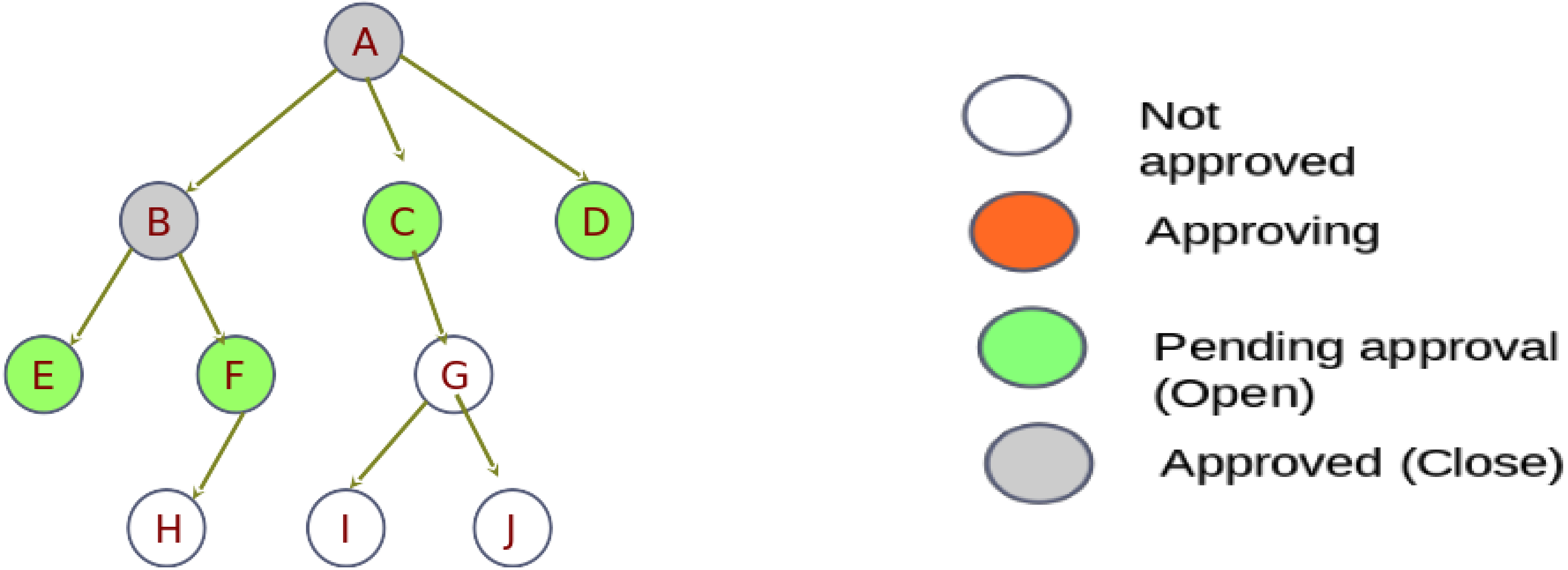
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

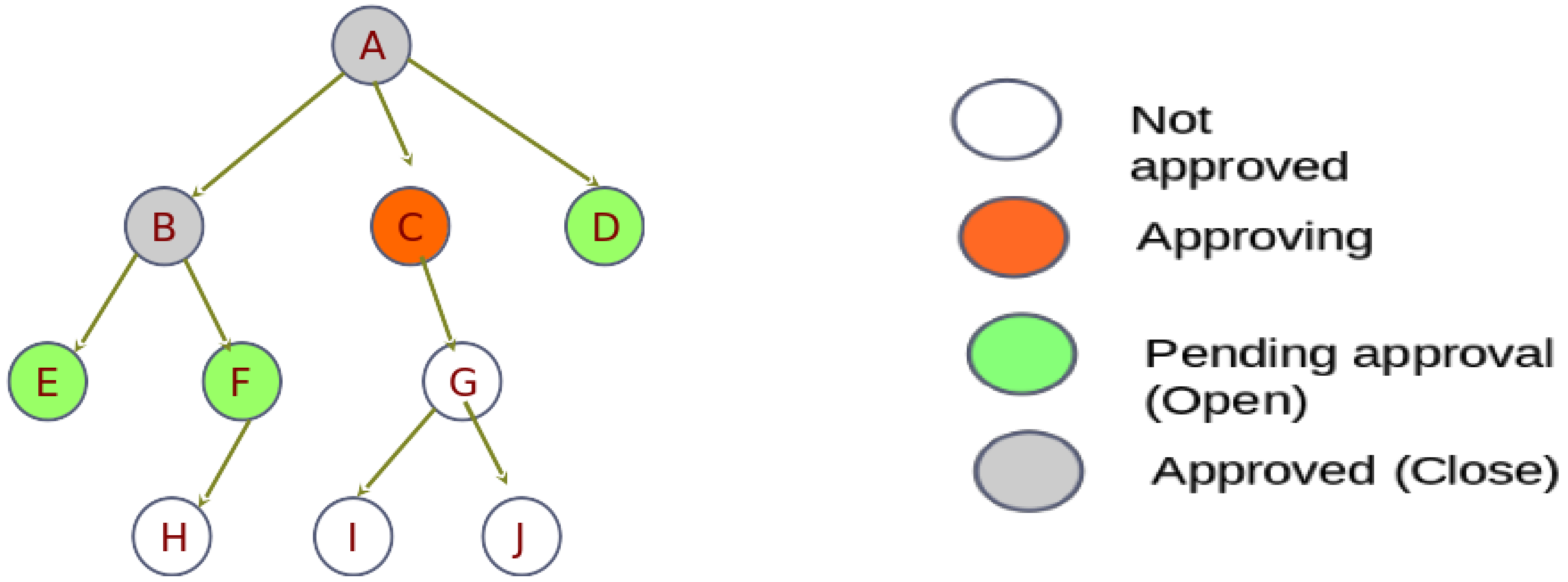
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

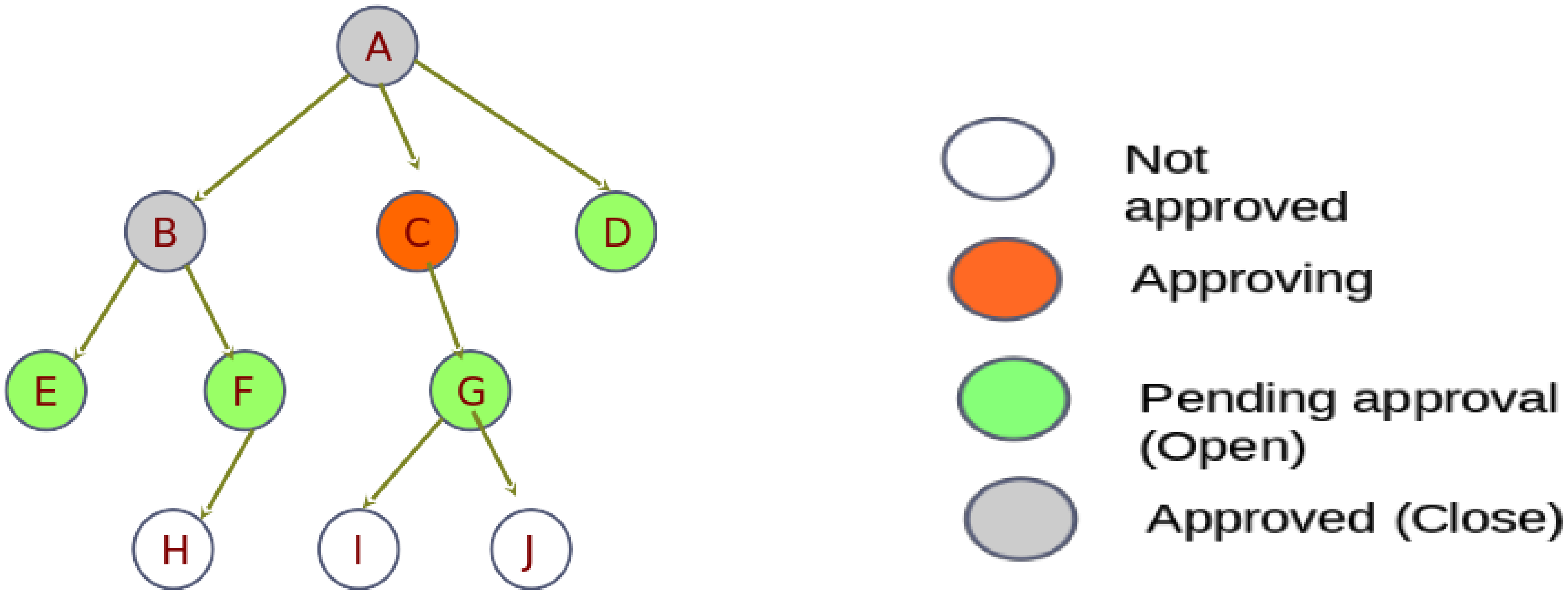
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

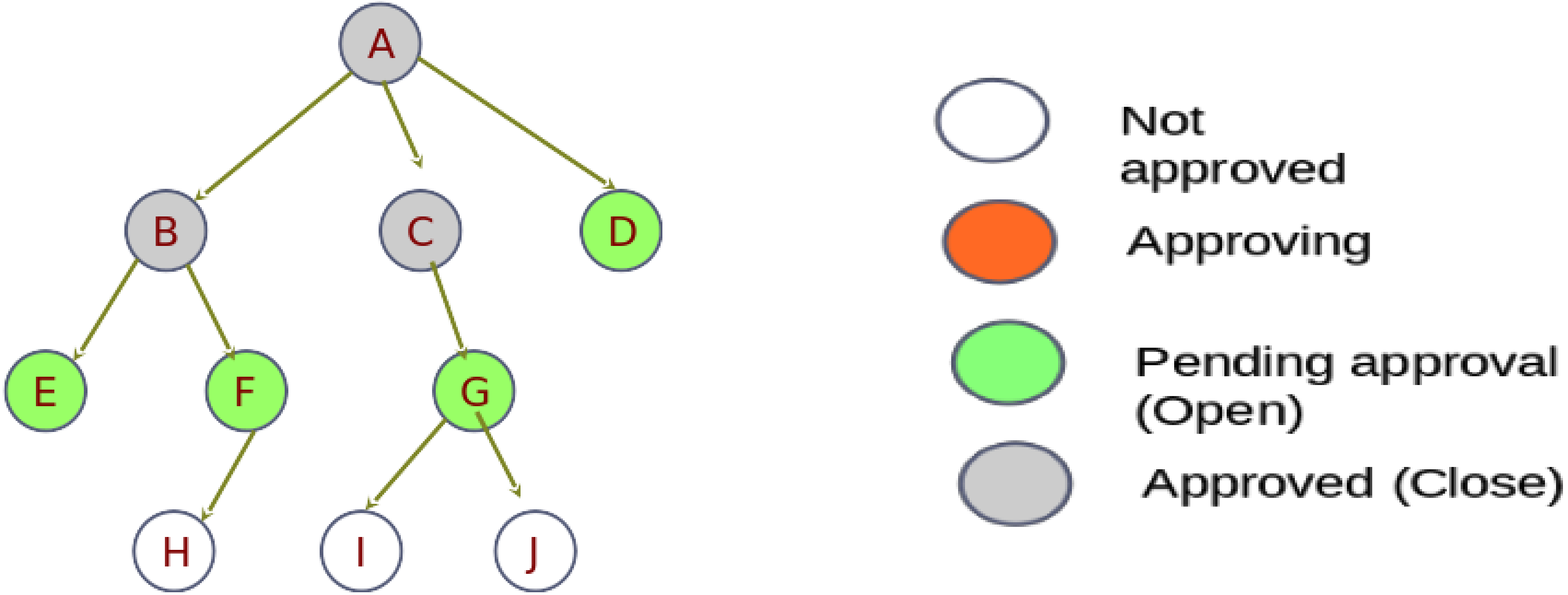
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

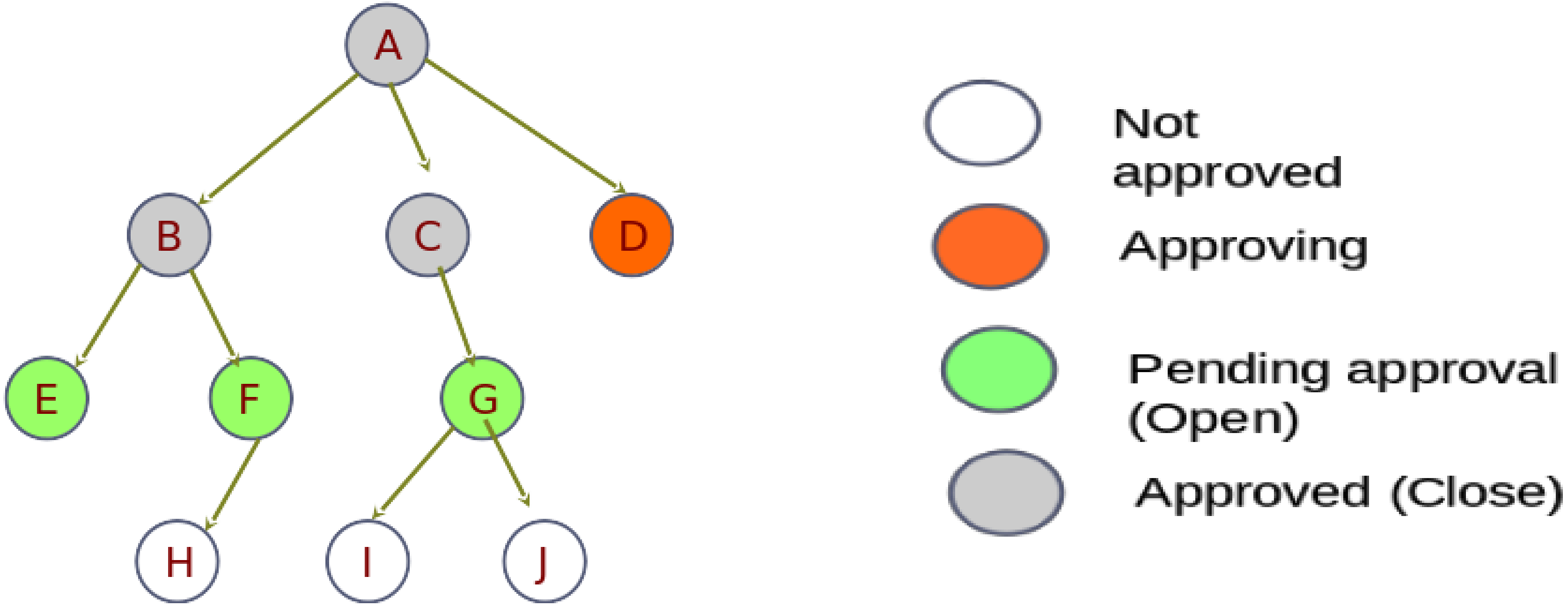
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

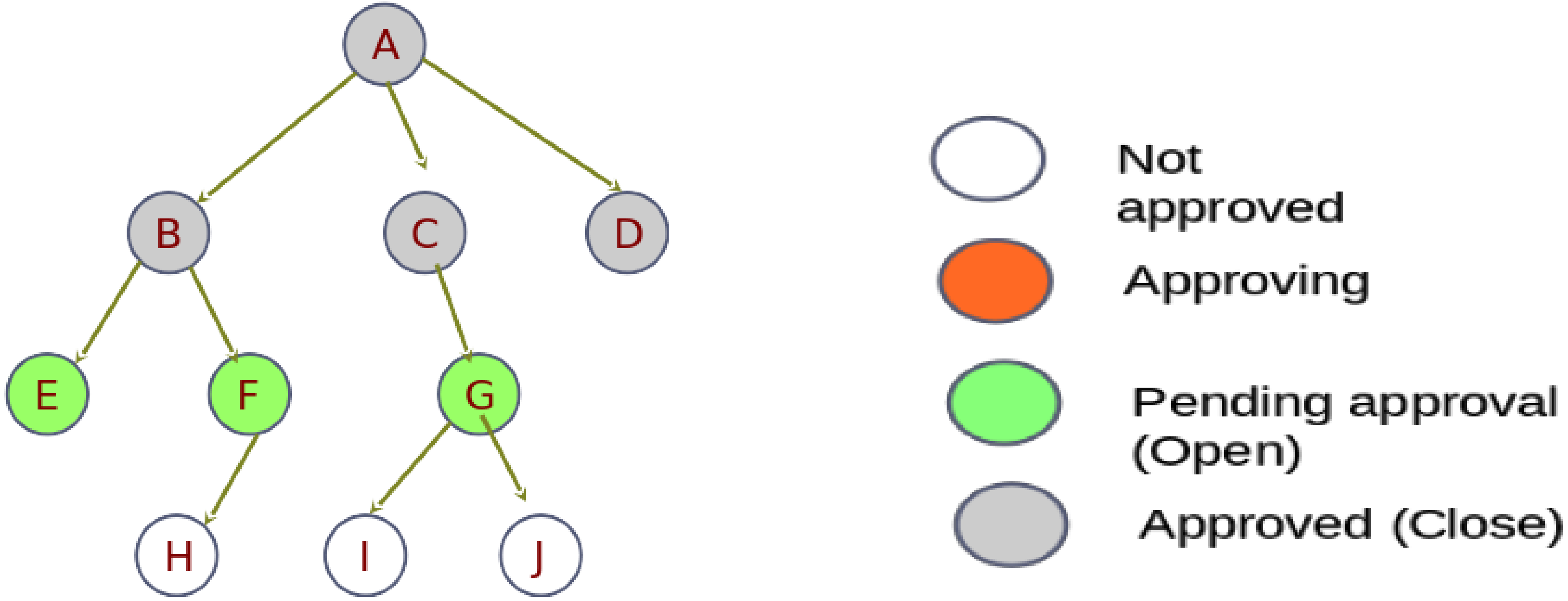
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

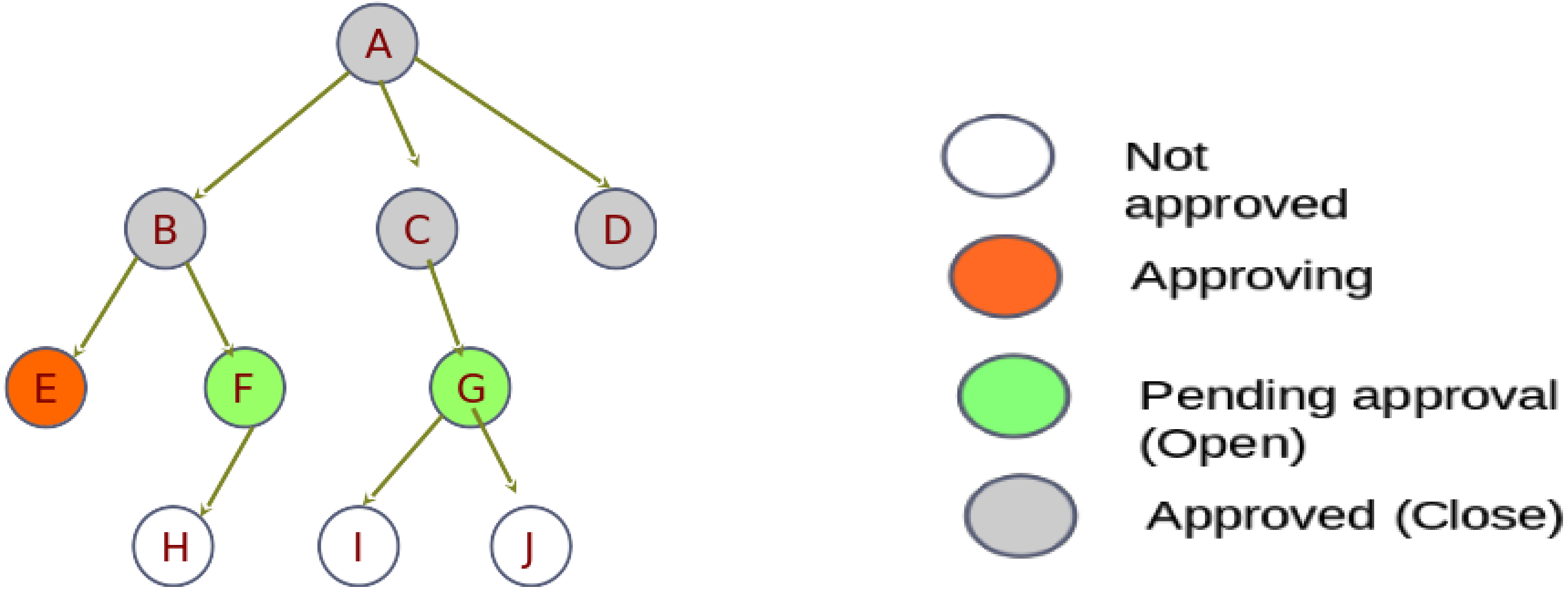
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

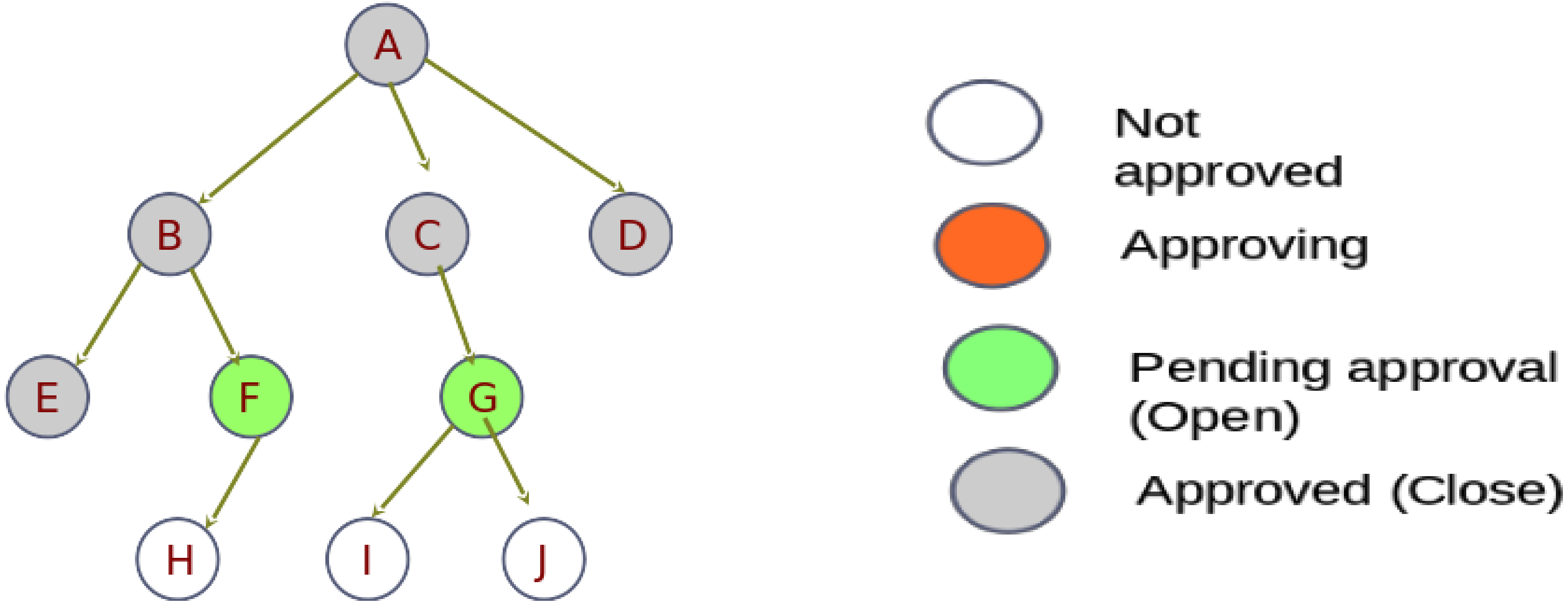
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

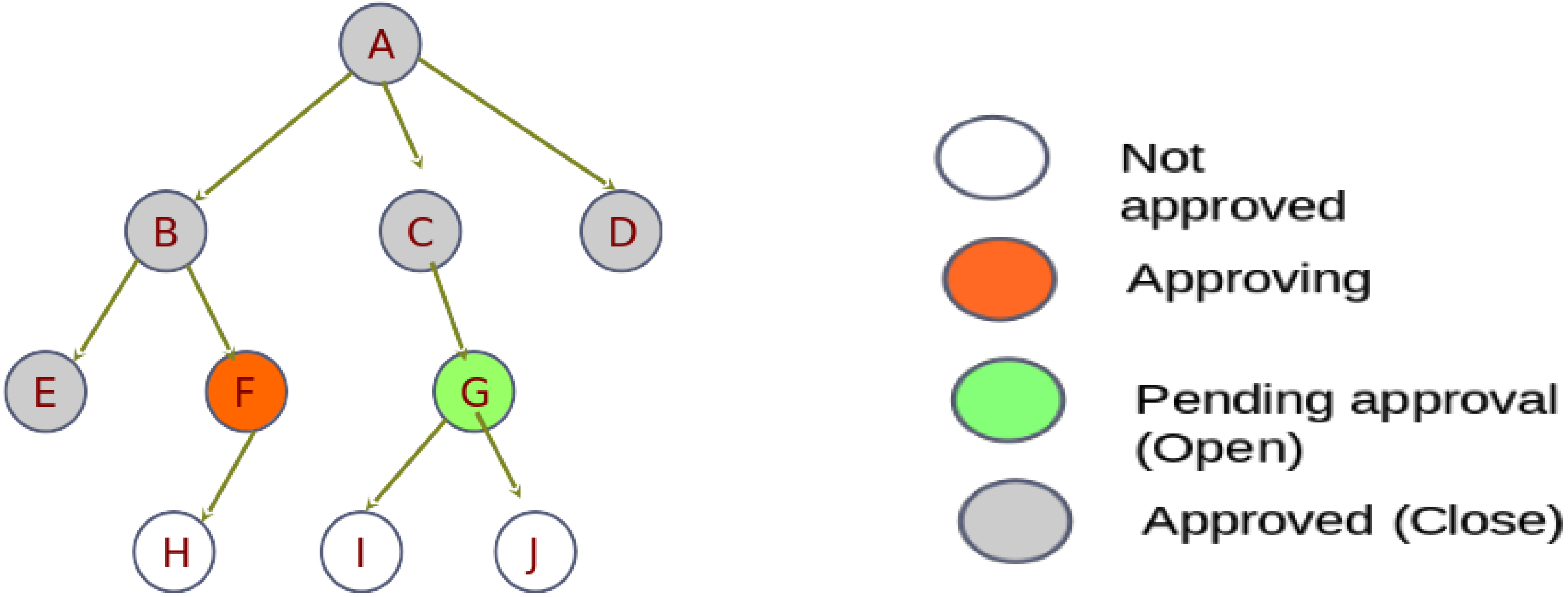
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

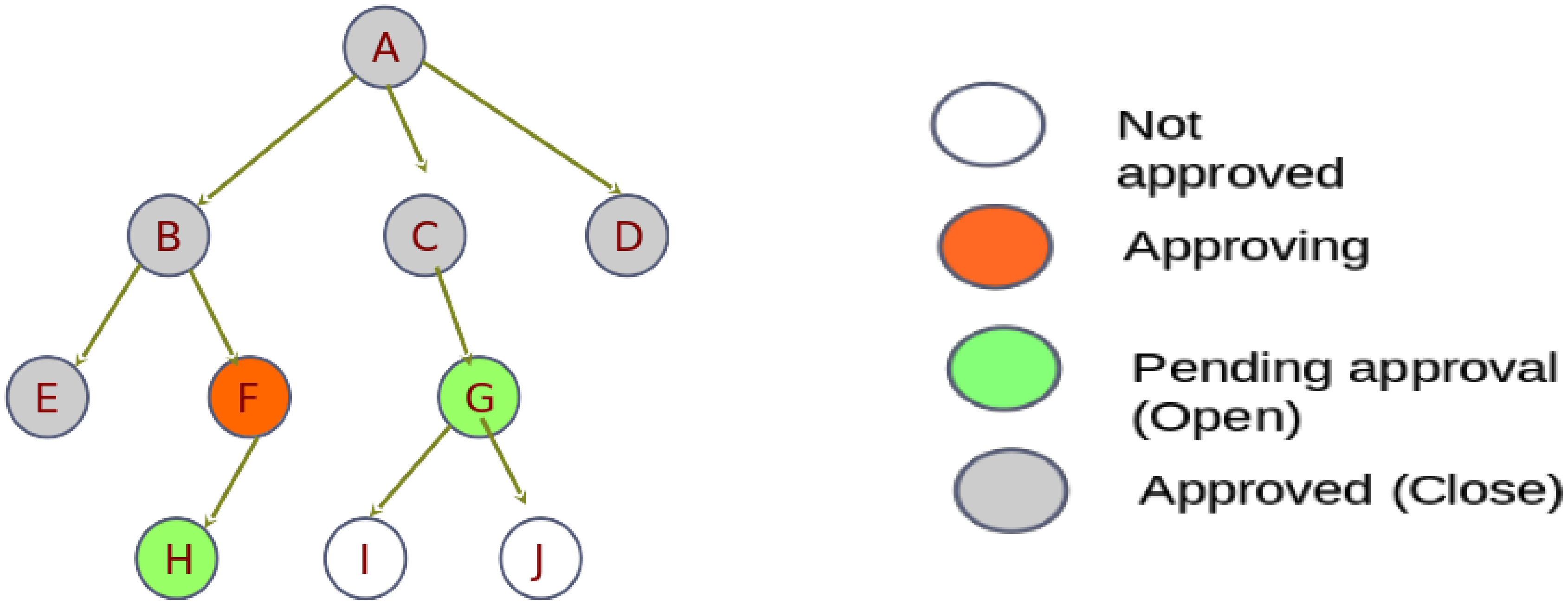
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

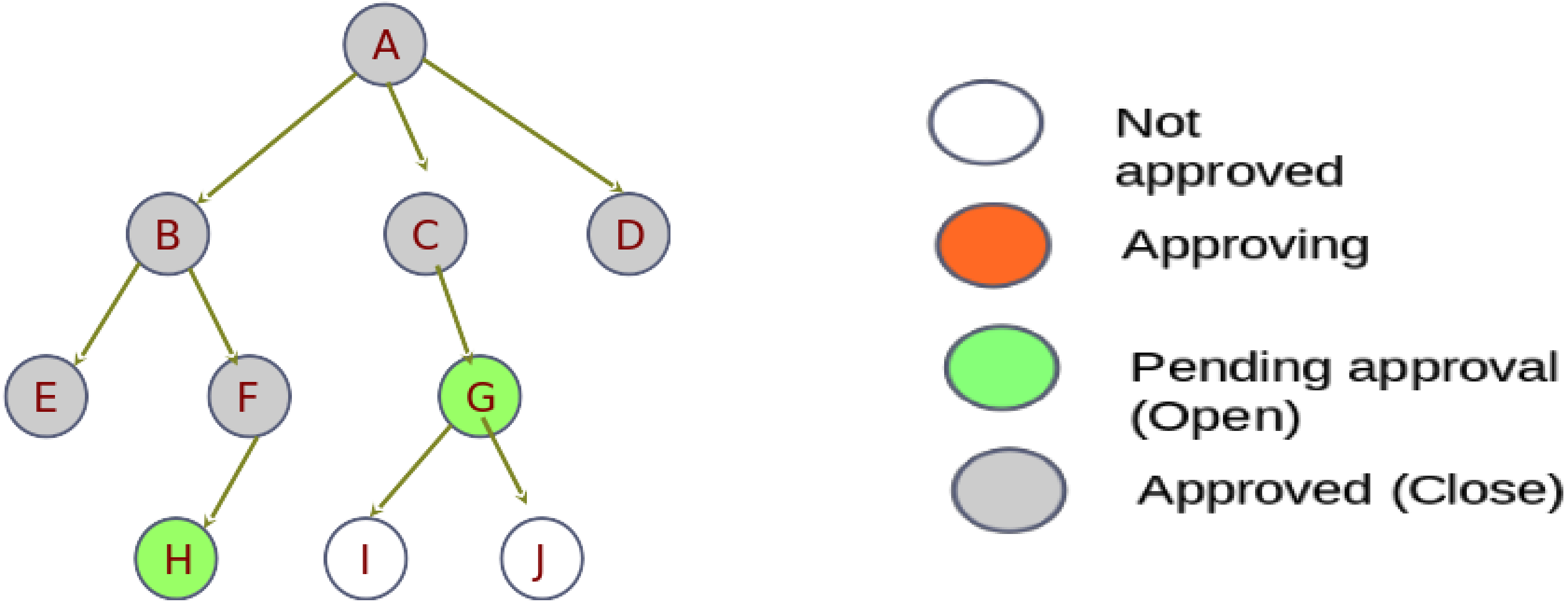
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

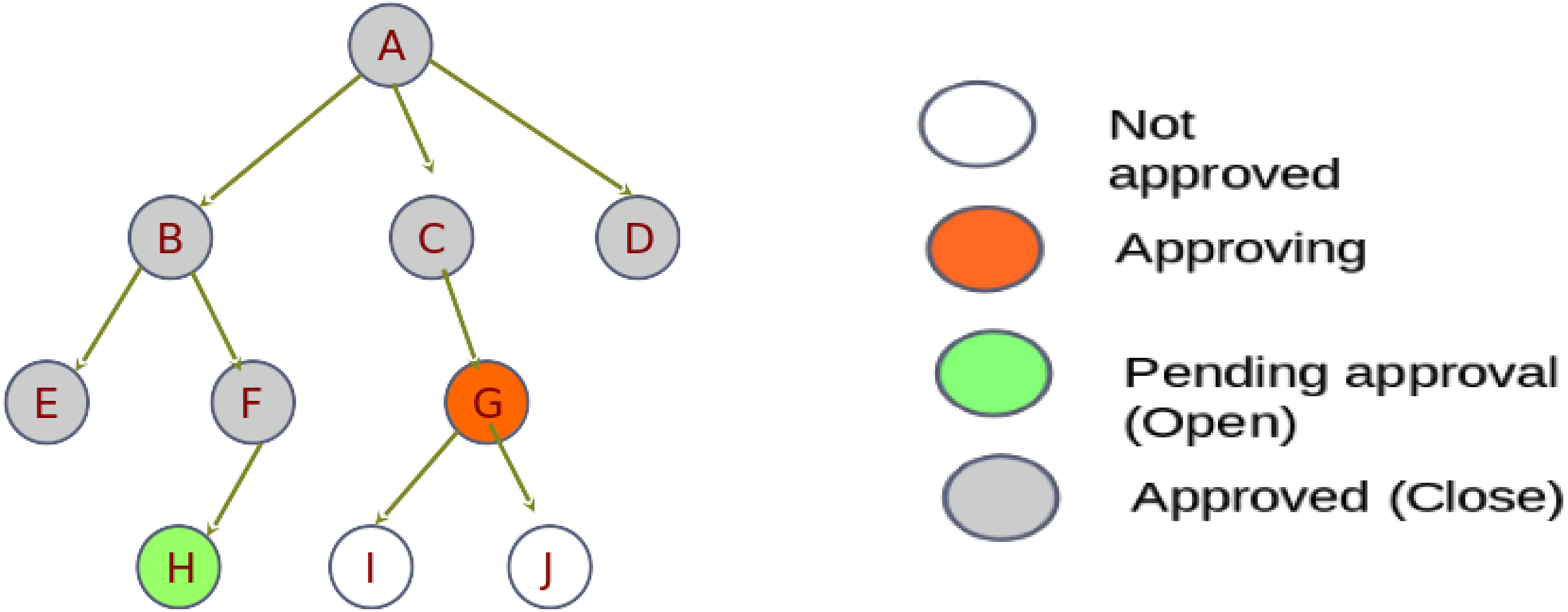
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

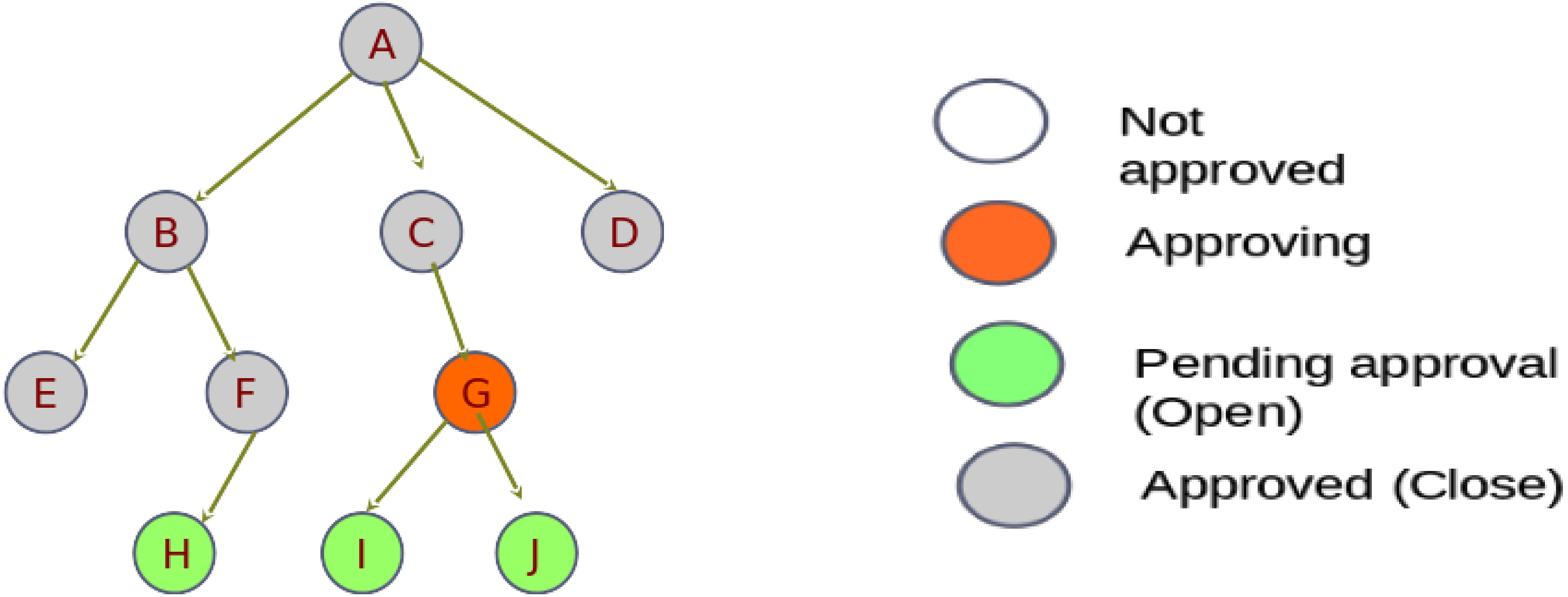
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

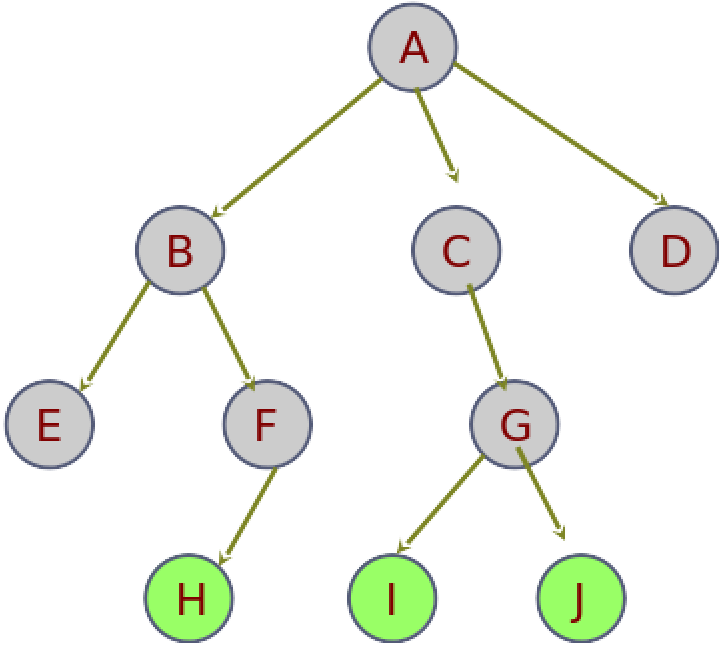
- "**OPEN**" Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



○ Not approved

🟧 Approving

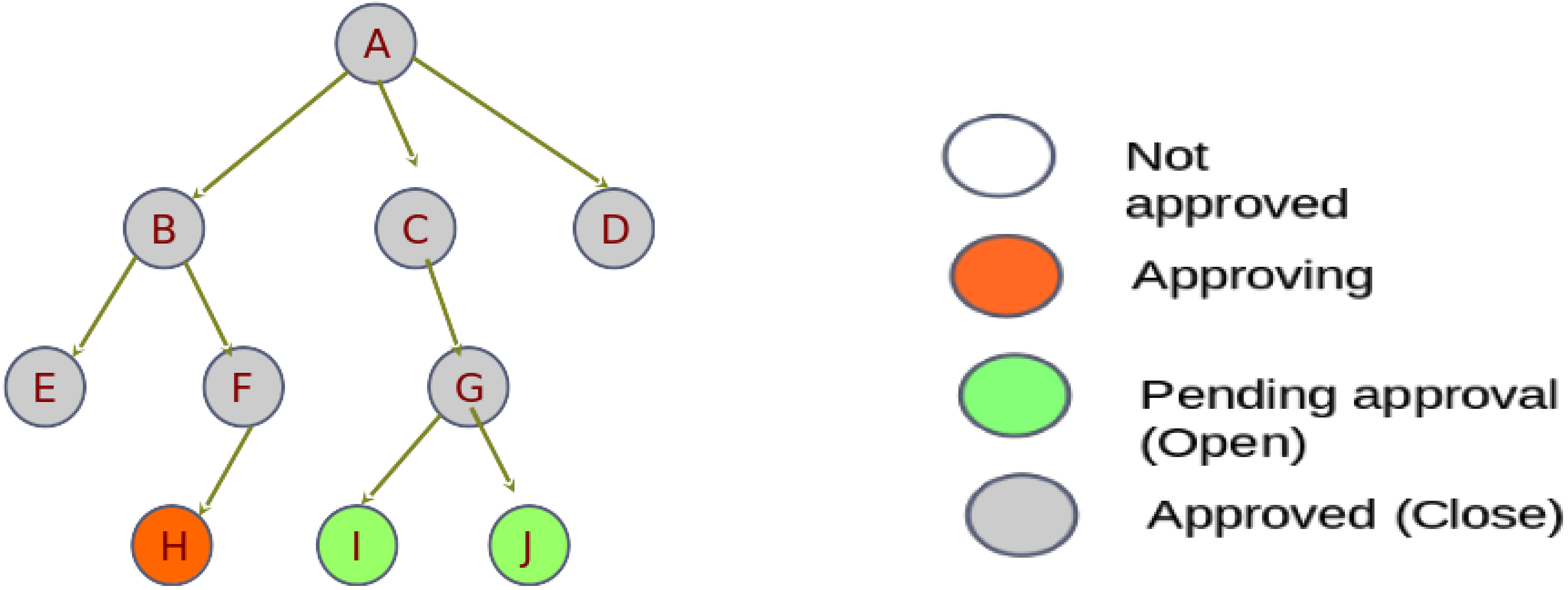
🟩 Pending approval (Open)

● Approved (Close)

Uninformed Search Strategies

Breadth-first search

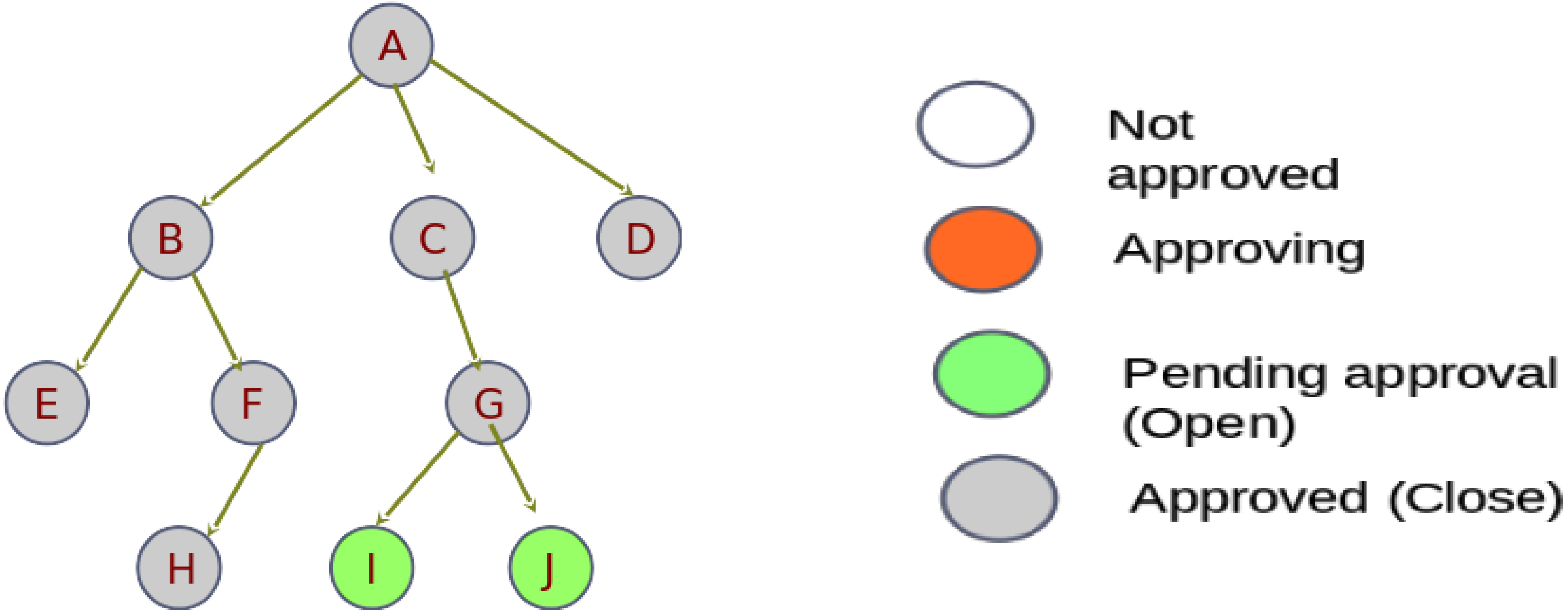
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

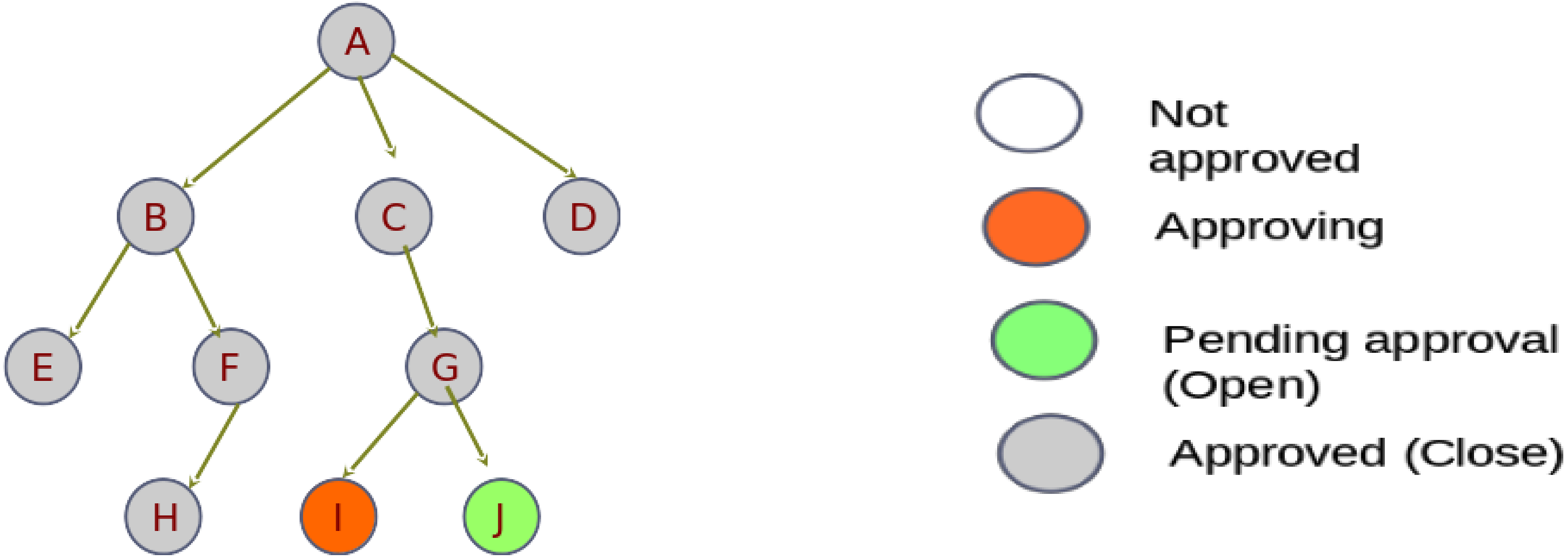
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

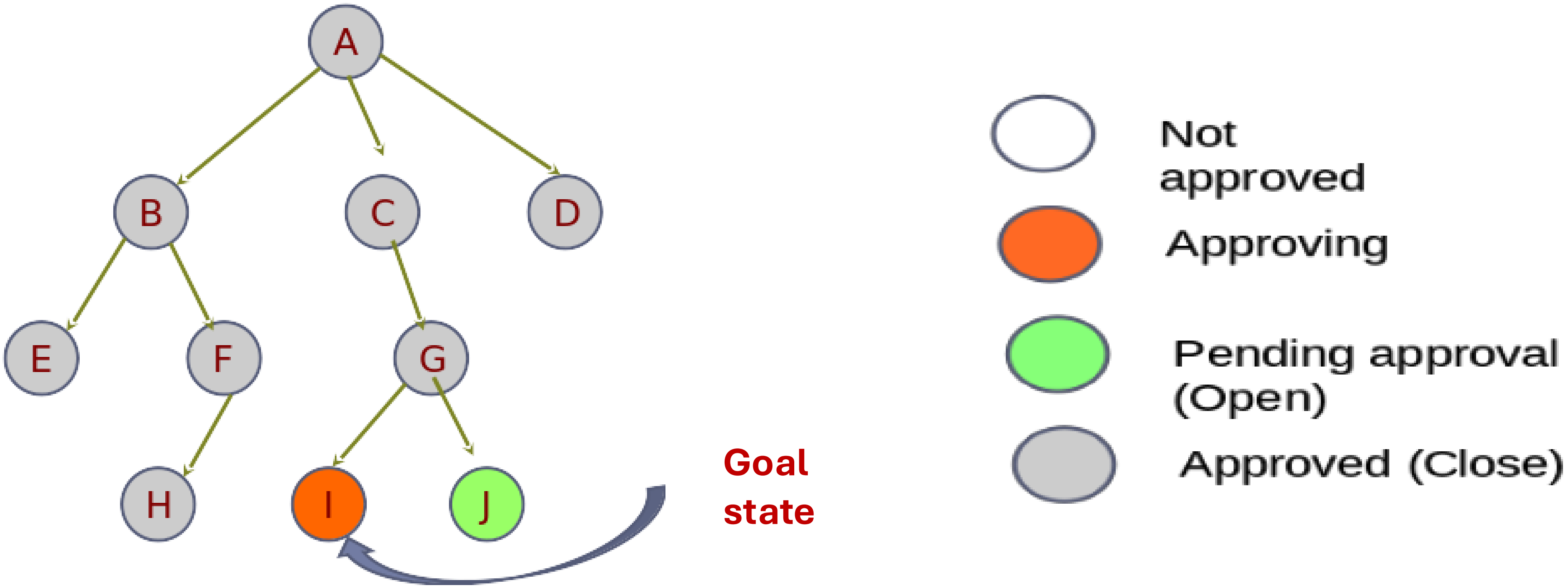
- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

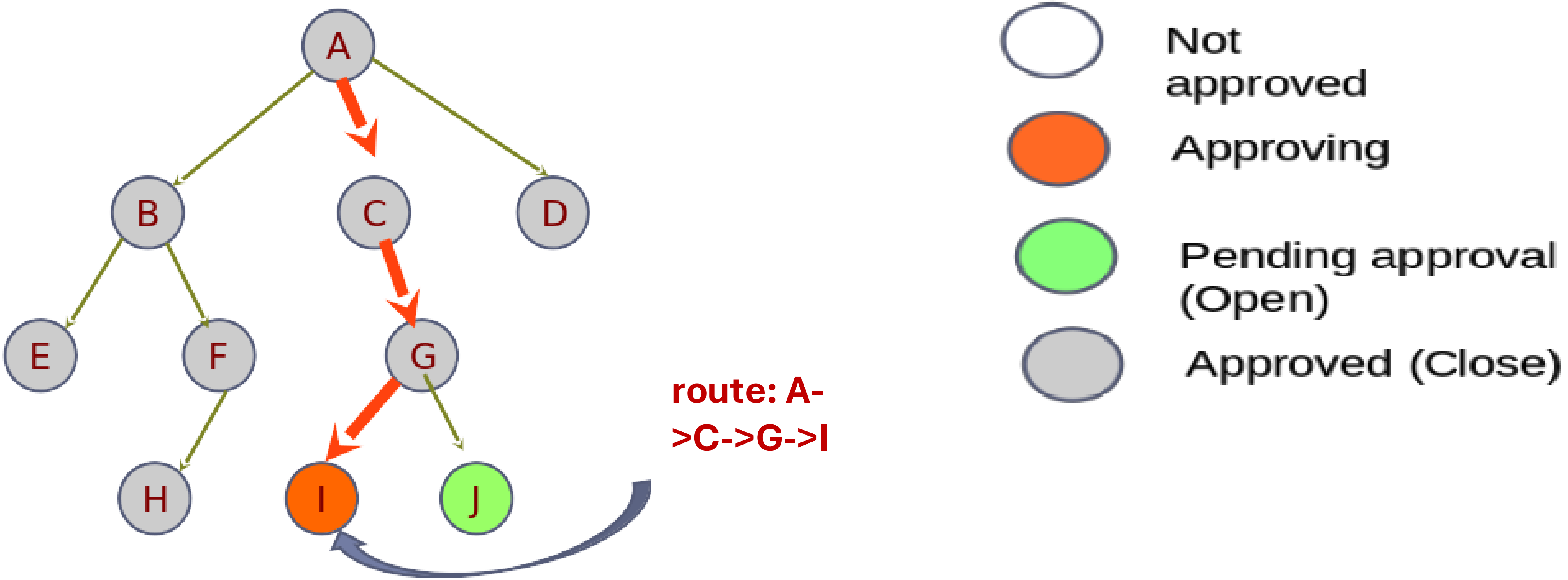
- **"OPEN"** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



Uninformed Search Strategies

Breadth-first search

- **“OPEN”** Pending State Set: Queue
- The state that is **generated first** is **approve first**
- **Approve** all nodes at the **same depth** before **approve** nodes at the **next depth**



route: A->C->G->I

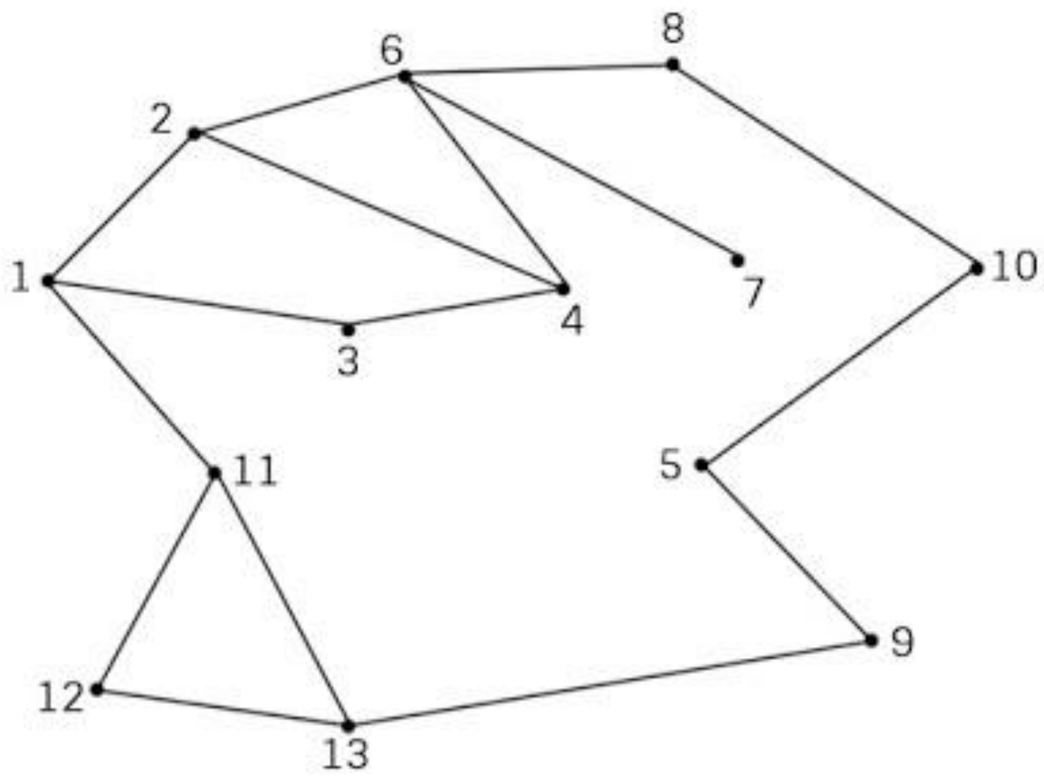
Uninformed Search Strategies

Breadth-first search

Exercise: Find the path and sequence of vertices

- Initial state: 1
- Goal state: 12

Step	u	edge(u)	OPEN
0			1
1	1	2,3,11	11,3,2
2	11	1,12,13	3,2,12,13
3	3	1,4	2,12,13,4
4	2	1,4,6	12,13,4,6
5	12		



Uninformed Search Strategies

Depth-first search (DFS)

- The set of states waiting to be explored "OPEN": Stack.
- The **deepest state** (the one most recently added to Open) is **expanded first.**
- Expand all **descendant nodes** before **moving up** to explore other nodes at the **same depth** if a solution has not been found.

  - Example: route from A-> M: A-C-F-M

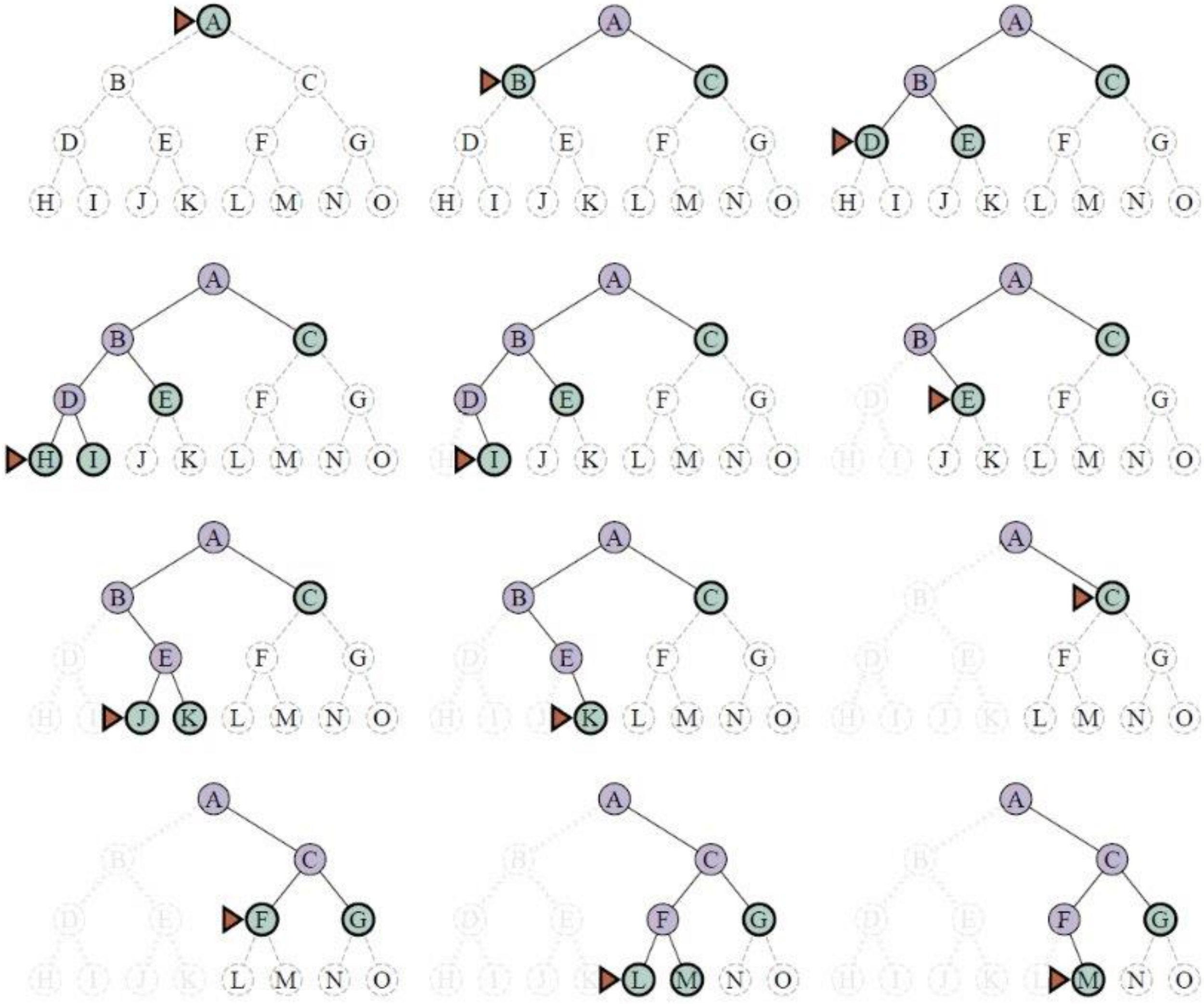


Figure 3.11 A dozen steps (left to right, top to bottom) in the progress of a depth-first search

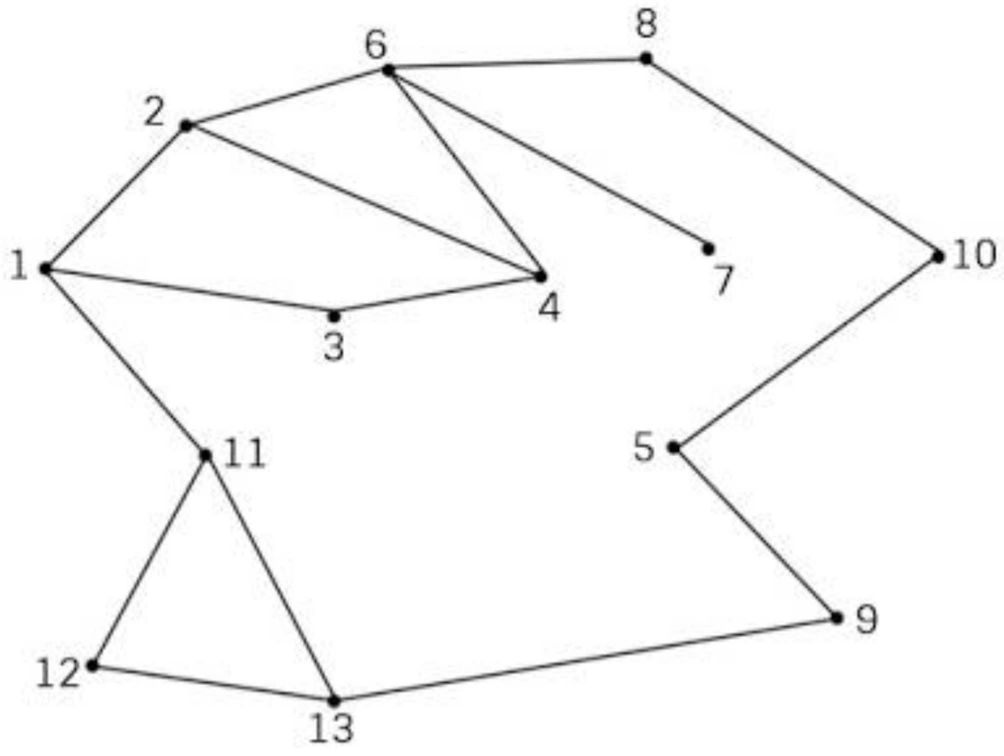
Uninformed Search Strategies

Depth-first search

Exercise: Find the path and sequence of vertices

- Initial state: 1
- Goal state: 12

Step	u	edge(u)	OPEN
0			1
1	1	11,3,2	11,3,2
2	11	12,13,1	12,13,3,2
3	12		



Uninformed Search Strategies

	BFS	DFS
Open	FIFO (Queue)	LIFO (Stack)
Efficiency	When the goal state is near the root of the search tree	When the goal state is deep in the search tree and there is a correct path choice
Complexi ty	Takes more memory Has the same time complexity in theory but is slower in practice	
Result	Sure to find results if any	No solution found in iterative spaces or spaces of infinite depth

Uninformed Search Strategies

Depth-first search

Limitation: The search space can have infinite depth or iterative states.

Depth-Limited search: Search (depth-first) only within a certain depth limit. Nodes at deeper depths are not considered.

Iterative deepening search: Is a limited depth first search with increasing depth limits from 0 to a max level.

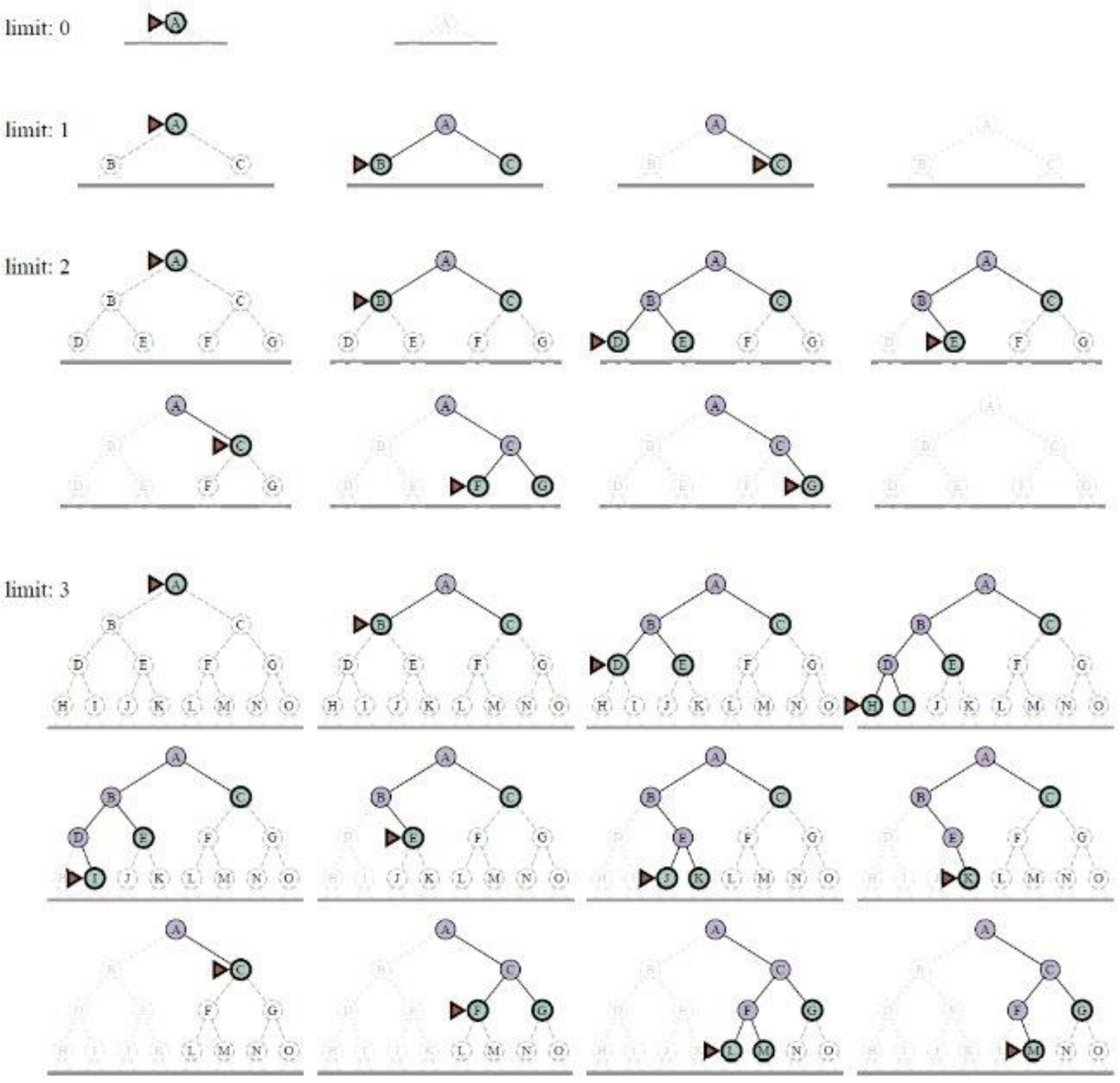
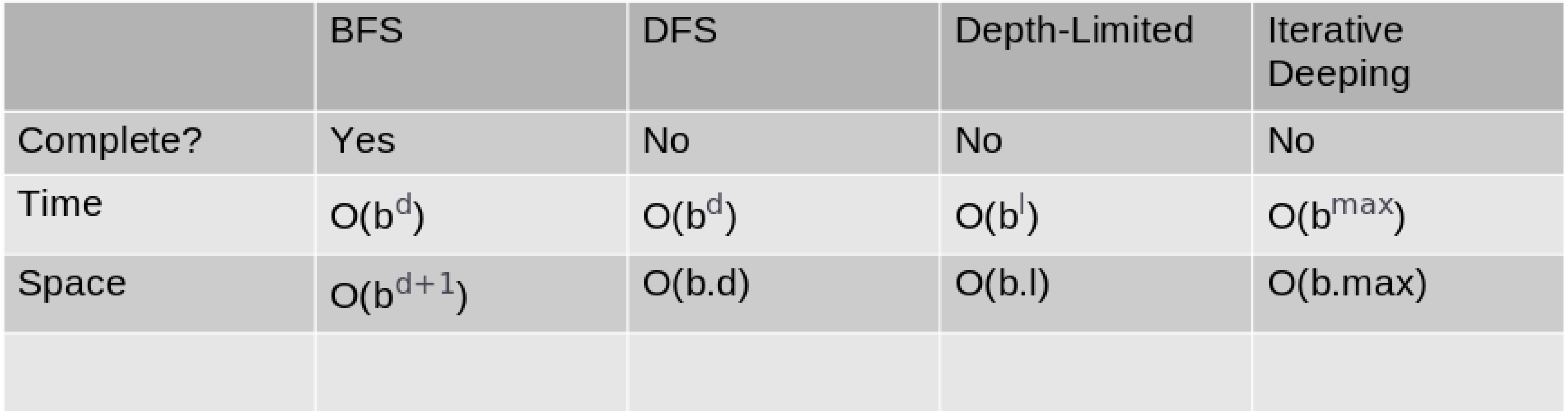


Figure 3.13 Four iterations of iterative deepening search for goal M on a binary tree

Uninformed Search Strategies

Performance Analysis fort Uninformed Searches



	BFS	DFS	Depth-Limited	Iterative Deeping
Complete?	Yes	No	No	No
Time	O(b^d)	O(b^d)	O(b^l)	O(b^{max})
Space	O(b^{d+1})	O(b.d)	O(b.l)	O(b.max)

- Completeness: is the ability to always find a solution if it exists.
- Depth: not found if the state space is iterated or has infinite depth
- Limited depth: not found if the solution is deeper than the limit
- Iterative: not found if the solution is deeper than the maximum search depth

Thank you!

You're now ready to explore the exciting world of AI!