Artificial Intelligence (Course at FMI, SU)

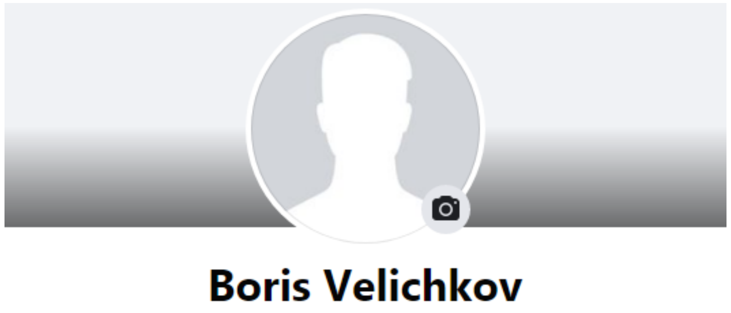
Boris Velichkov

Who Am I

- Professional Experience
  - Since July 2012
- Teaching Assistant
  - Since October 2013
- Education
  - “Software Engineering” (BS at FMI, SU)
  - “Information Extraction and Knowledge Discovery” (MS at FMI, SU)
  - “Software Technologies (Knowledge Discovery)” (PhD at FMI, SU)

Contacts

- Email
  - bobby.velichkov@gmail.com
- Facebook



- Telephone
  - 0884 139 550

AI Courses at FMI, SU

- Computer Science
  - Artificial Intelligence (Prof. Ivan Koychev)
- Software Engineering
  - Intelligent Systems (Prof. Ivan Koychev)
- Elective Courses (BS & MS)
  - Artificial Intelligence (Prof. Ivan Koychev)
- Informatics
  - Artificial Intelligence (Prof. Maria Nisheva)

Evaluation

- Ongoing Assessment (50%)
  - Test 1 ≈ ⅓ (at least 3.00)
  - Test 2 ≈ ⅓ (at least 3.00)
  - Homeworks ≈ ⅓ (at least 3.00 from at least 5 tasks)
- Exam (50%)
  - Project Presentation ≈ ⅔ (optional)
  - Interview (final assessment) ≈ ⅓

** The evaluation is more like a point system - for excellent you do not need to collect the maximum number of points.*

Exercises' Topics

- Problem Solving and Search
  - Uninformed *(Blind)* Search
  - Informed *(Heuristic)* Search
  - Constraint Satisfaction Problems
  - Genetic Algorithms
  - Games
- Machine Learning
  - *k*-Nearest Neighbors
  - Naïve Bayes Classifier
  - Decision Tree
  - *k*Means
  - Neural Networks

$$10\ Topics = \frac{from\ 8\ to\ 10}{\text{Homeworks}}$$

$$\begin{matrix} Programming \\ Languages \end{matrix} = \begin{cases} C/C++ \\ Java \\ C\# \\ Any\ Other \end{cases}$$

Exercises by Week

1. Introduction
2. **Uninformed *(Blind)* Search**
3. **Informed *(Heuristic)* Search**
4. **Constraint Satisfaction Problems**
5. **Genetic Algorithms**
6. **Games**
7. Introduction to Machine Learning
8. ***k*-Nearest Neighbors**
9. **Naïve Bayes Classifier**
10. **Decision Tree**
11. ***k*Means**
12. **Neural Networks**
13. *Additional Topics, Questions and Homeworks' Presentations*
14. *Additional Topics, Questions and Homeworks' Presentations*
15. *Additional Topics, Questions and Homeworks' Presentations*

Resources

- The Main Book
  - Stuart Russell and Peter Norvig. Artificial Intelligence: A Modern Approach. Prentice Hall.
  - http://aima.eecs.berkeley.edu/
- Lectures, Exercises and all other resources at Moodle

Problem Solving and Search

- Basic Concepts
  - State Space
  - Representation of the State Space
  - Search Strategies Evaluation
  - Global Search vs Local Search
  - Uninformed Search vs Informed Search
  - Graph and Tree Traversal

State Space

- **State**: a representation (formulation) of the task in the process of its solution.
  - *Initial State* (presented with ***S***)
  - *Intermediate State* (presented with any capital letter except ***S*** and ***G***)
  - *Goal State* (presented with ***G***, if there are more: $\boldsymbol{G_1}$, $\boldsymbol{G_2}$, etc.)
- **Successor Function *(Operator)***: obtaining one state from another
- **Path Cost**: additive; e.g., sum of distances, number of actions executed, etc.
- **<u>State Space</u>**: the totality of all possible states that can be obtained from a given initial state.

** A **solution** is a sequence of actions leading from the initial state to a goal state*

Representation of the State Space

- **Graph** or **Tree** where each *state* is represented with **Node** and the *successor function (operator)* is represented with **Edge**.
- When the *state space* can be represented as a *tree*, it is called **Search Tree**.
  - The *initial state* is the *root* of the tree.
  - The terminal states and the *goal state* are represented with *leaves*.

Search Strategies Evaluation

Strategies are evaluated along the following dimensions:

- **Completeness**: does it always find a solution if one exists?
- **Time Complexity**: number of nodes generated/expanded
- **Space Complexity**: maximum number of nodes in memory
- **Optimality**: does it always find a least-cost solution?

** We look at the **worst-case** complexity (Big O notation).*

Time and space complexity are measured in terms of:

- ***b*** – maximum branching factor of the search tree
- ***d*** – depth of the least-cost solution
- ***m*** – maximum depth of the state space (may be ∞)

Global Search vs Local Search

- **Global Search**: They look all over the state space. If necessary, all states will be traversed.
- **Local Search**: They only look at the local area, so they can only look at states in this area. If the solution is outside of it, they will not be able to find it.

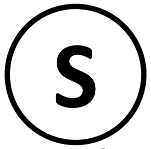
** Generally Local Search is not used for finding a path from state A to state B.*

Uninformed *(Blind)* Search vs Informed *(Heuristic)* Search

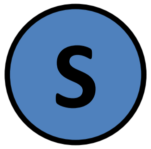
- **Uninformed Search**: Uninformed strategies use only the information available in the problem definition.
  - *Examples: DFS, BFS, UCS, DLS, IDS.*
- **Informed Search**: Informed strategies have information on the *goal state* which helps in more efficient searching. This information is obtained by a function *(heuristic)* that estimates how close a state is to the *goal state*.
  - *Examples: Greedy Best-First Search, A*, Beam Search, Hill Climbing.*

Graph and Tree Traversal

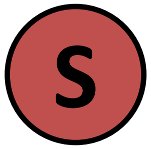
- **Node**: The *graphs* and *trees* consist of *nodes* and *edges*. When a *node* is neither *“visited”* nor *“expanded”*, we do not mark it in any way (sits uncolored).
  - *Example:*



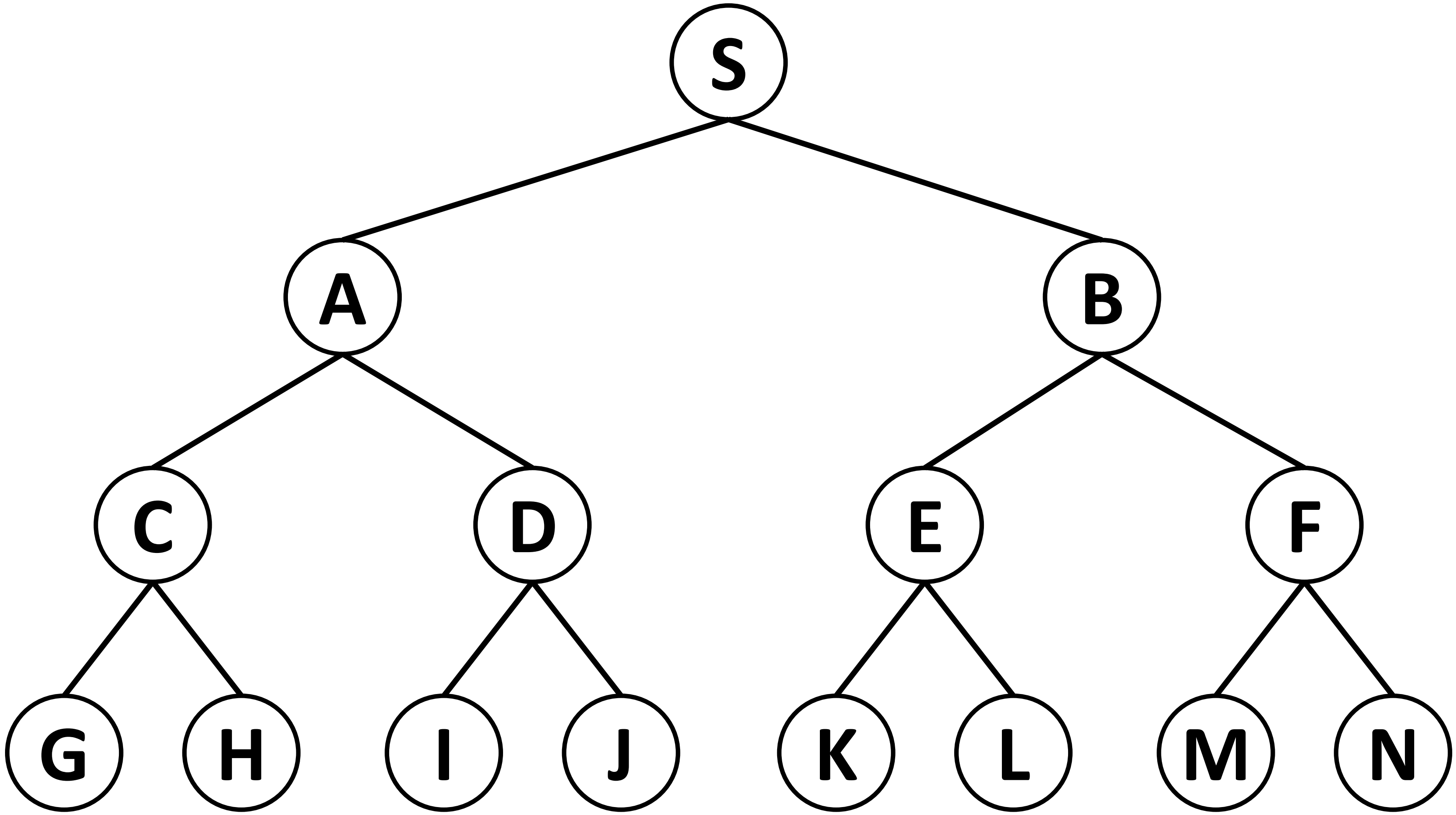
- **Visited Node**: When we visit a *node* or in other words add it to the data structure we use we call it *“visited node”*. We mark it in blue.
  - *Example:*



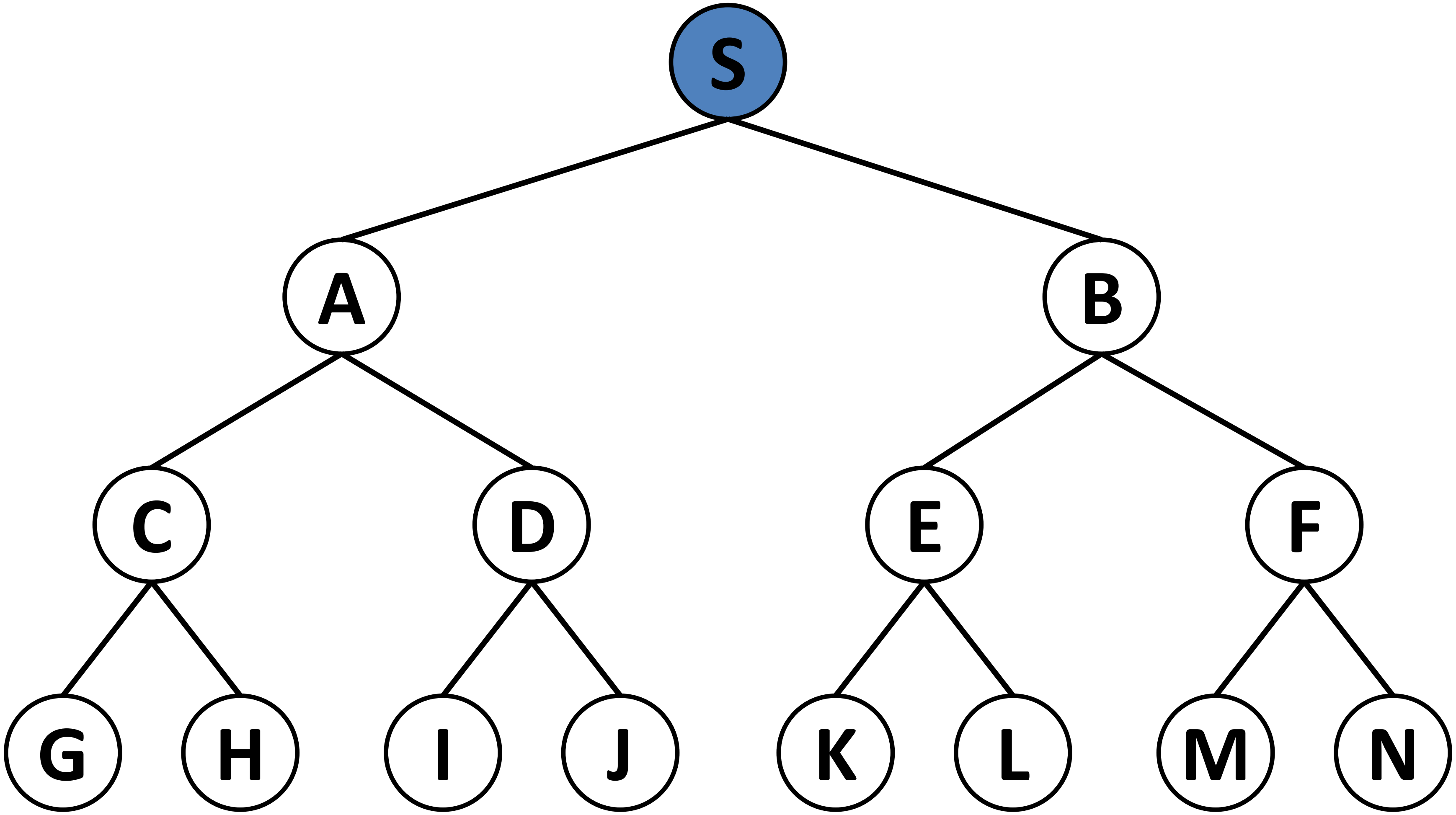
- **Expanded Node**: When we expand a *node* or in other words remove it from the data structure we use and add its children in this data structure (of course if the node has children) we call it *“expanded node”*. We mark it in red.
  - *Example:*



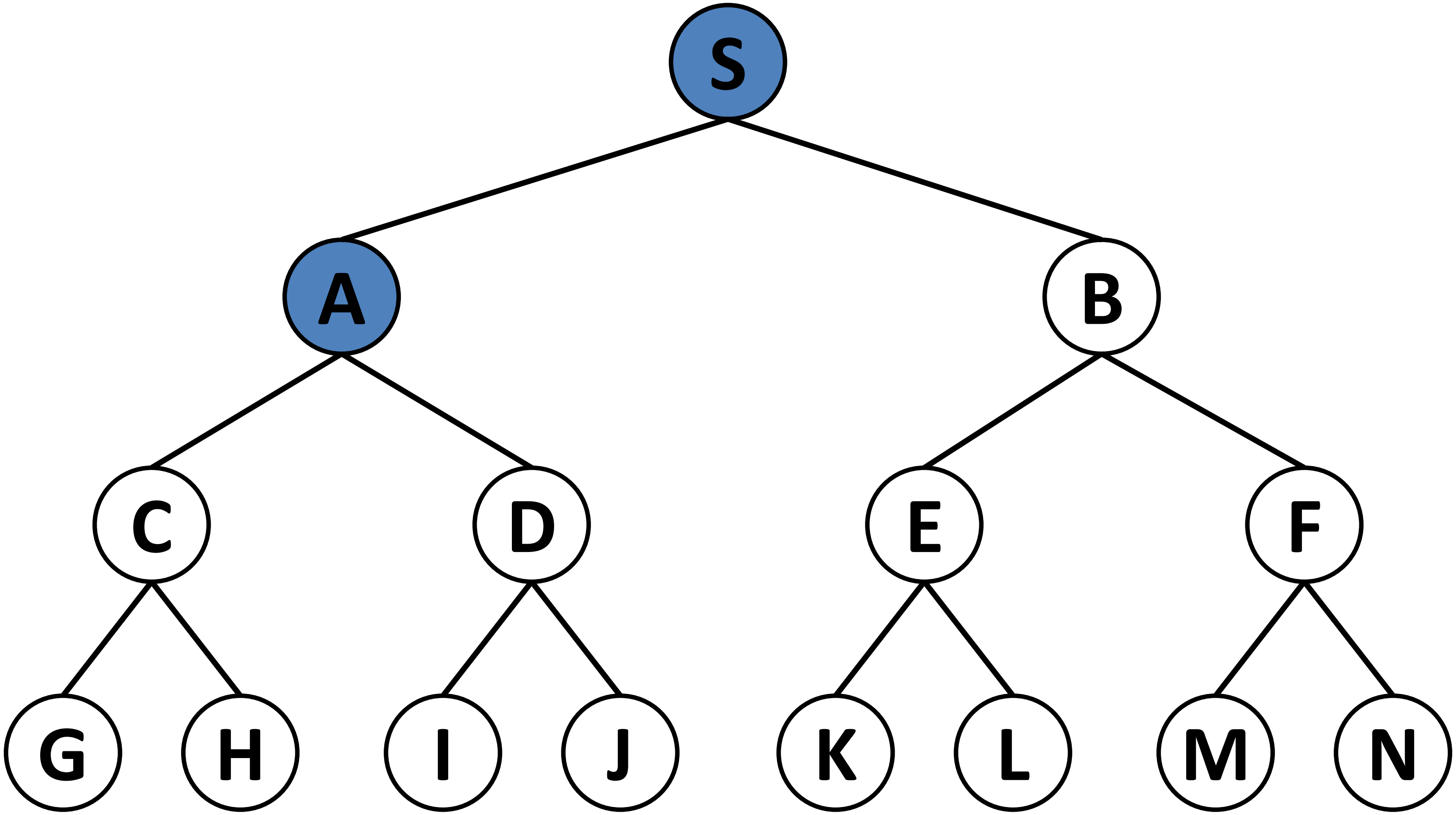
Graph and Tree Traversal: DFS



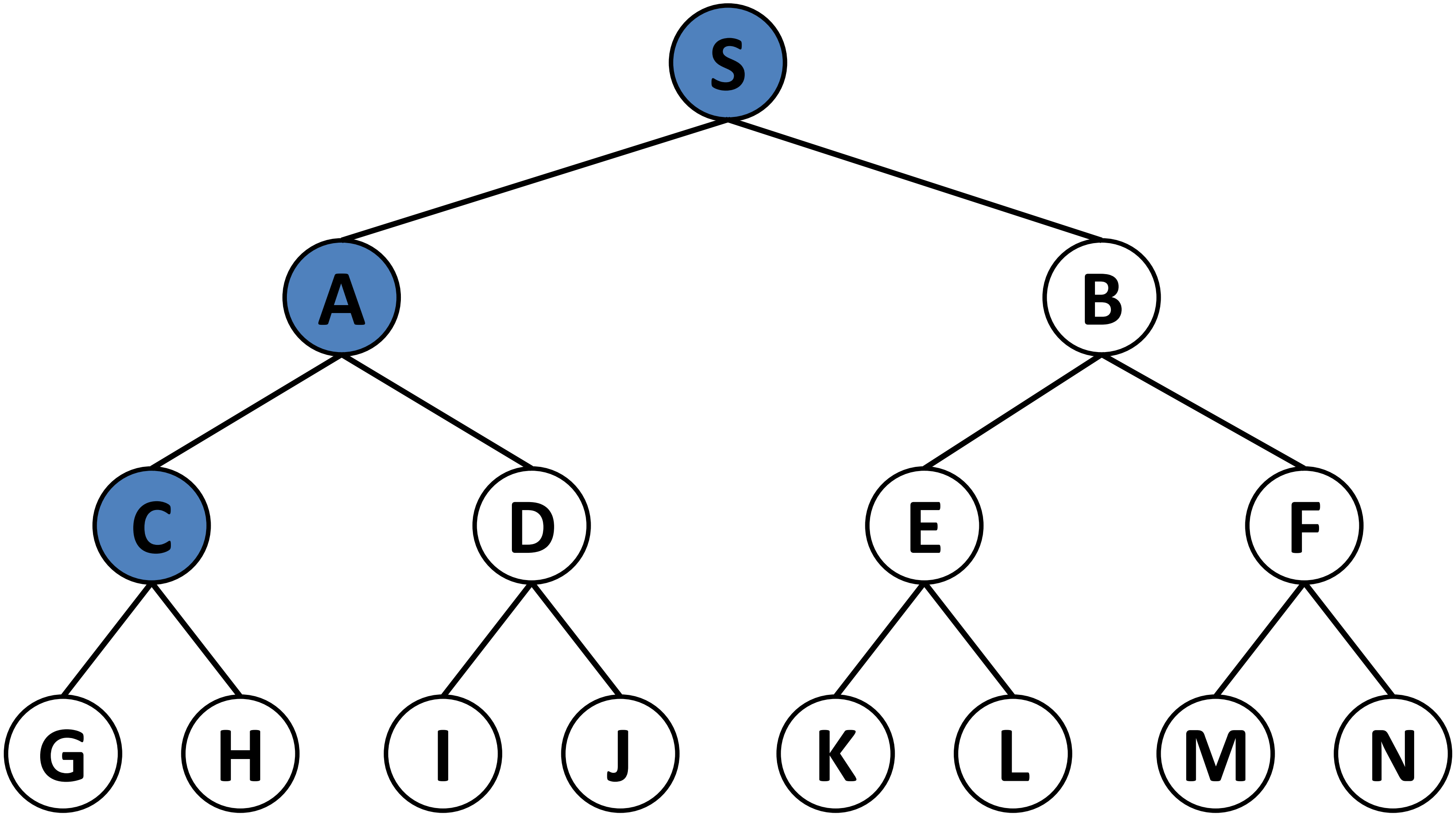
Graph and Tree Traversal: DFS



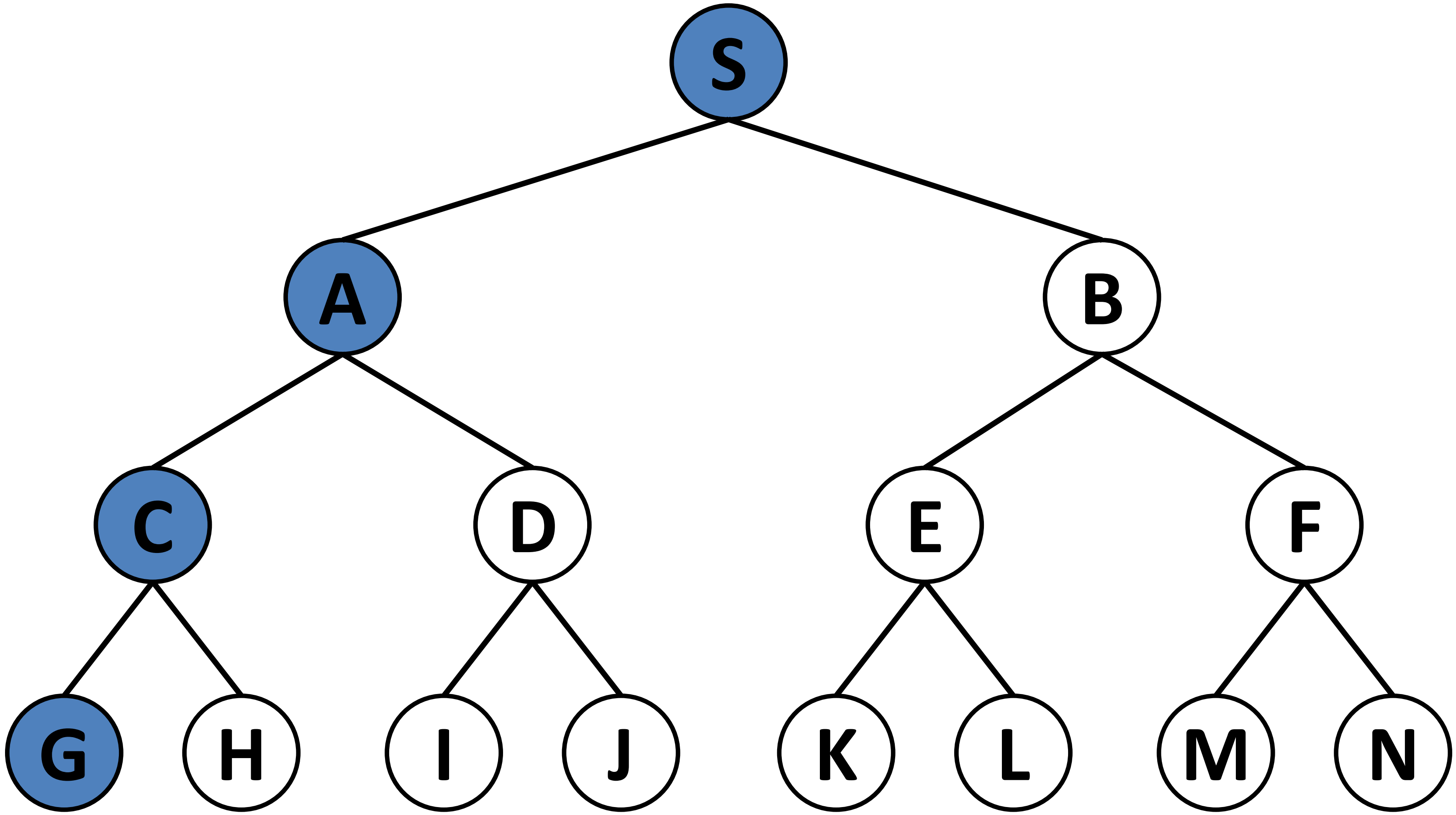
Graph and Tree Traversal: DFS



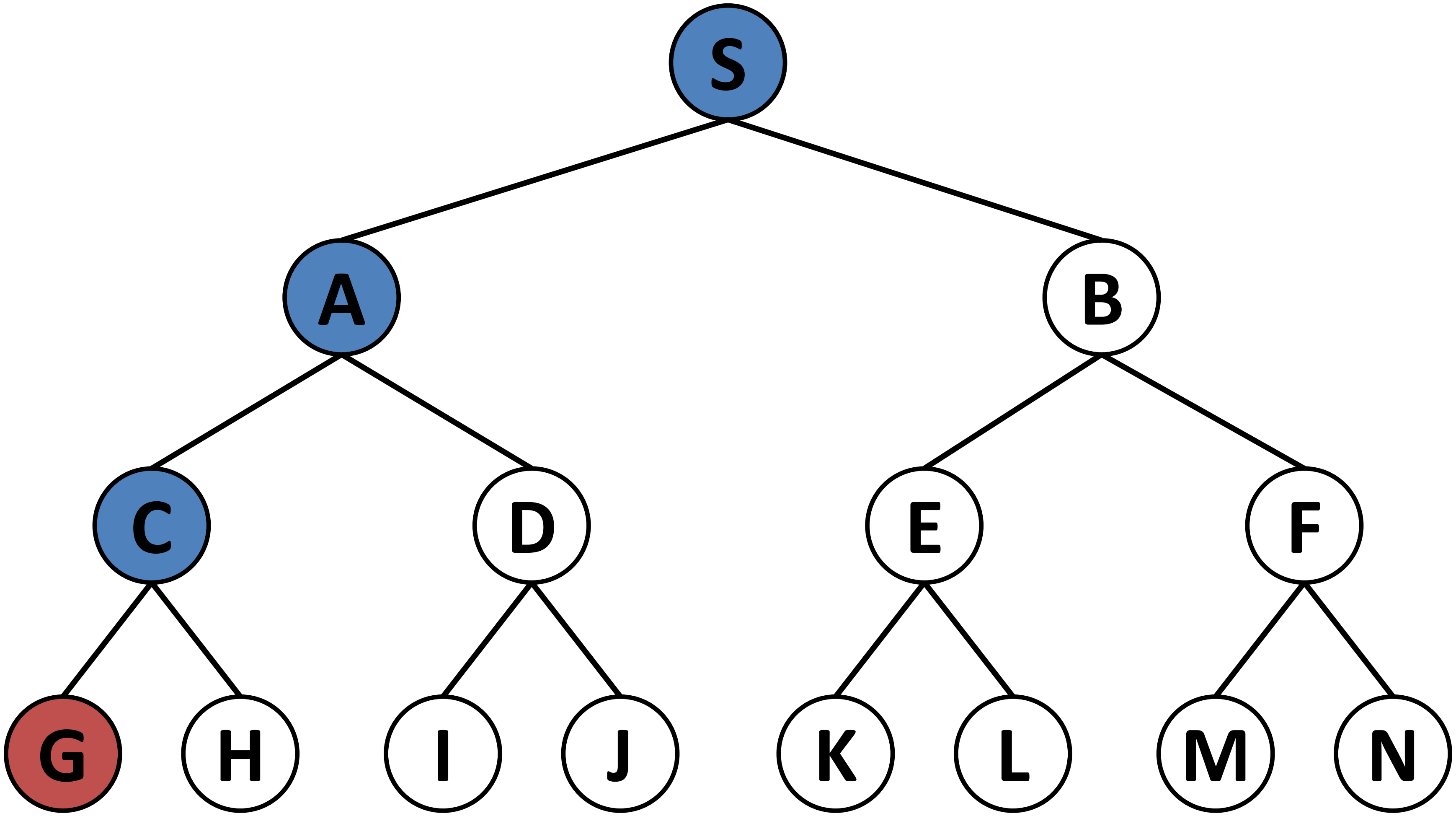
Graph and Tree Traversal: DFS



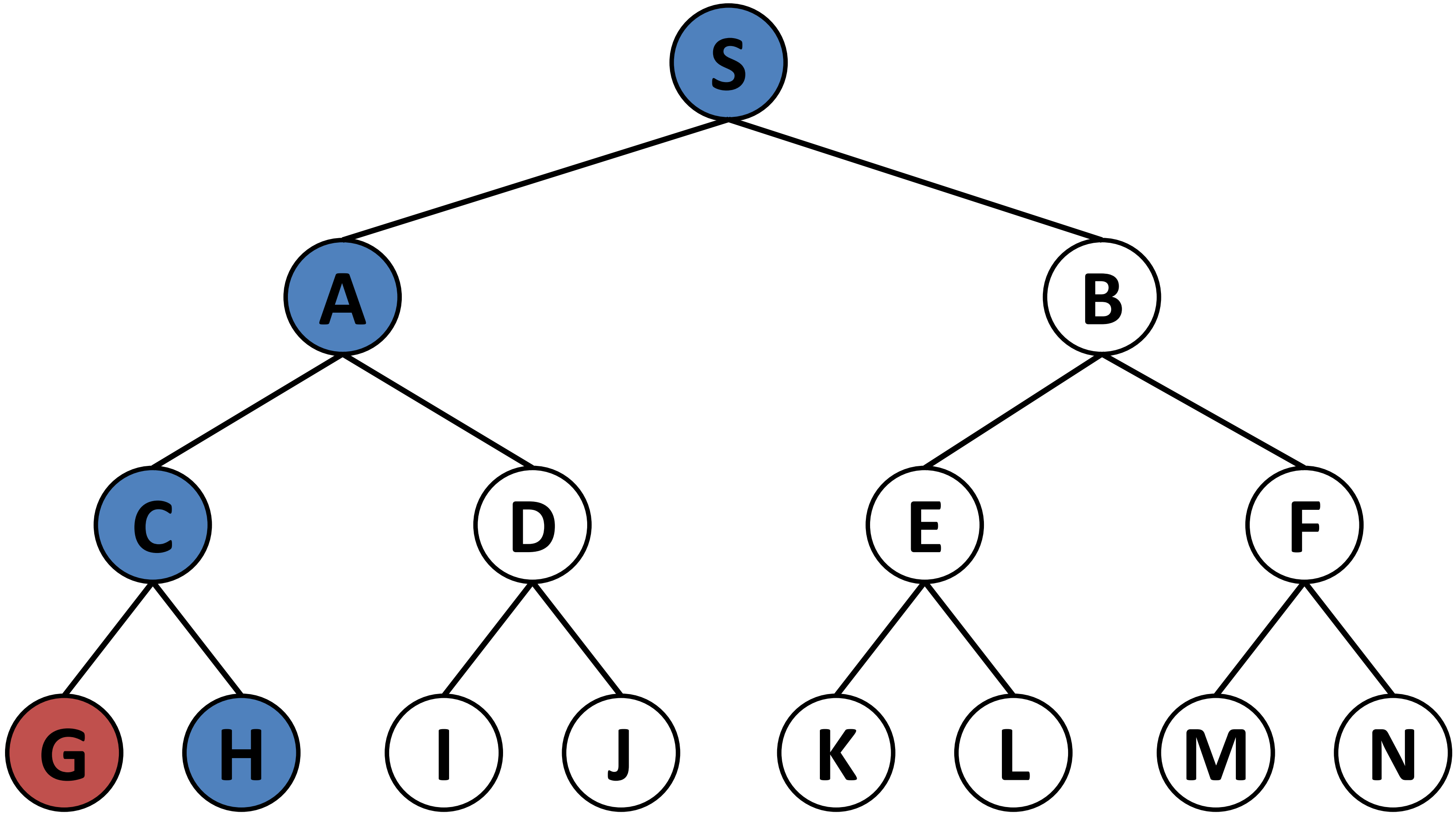
Graph and Tree Traversal: DFS



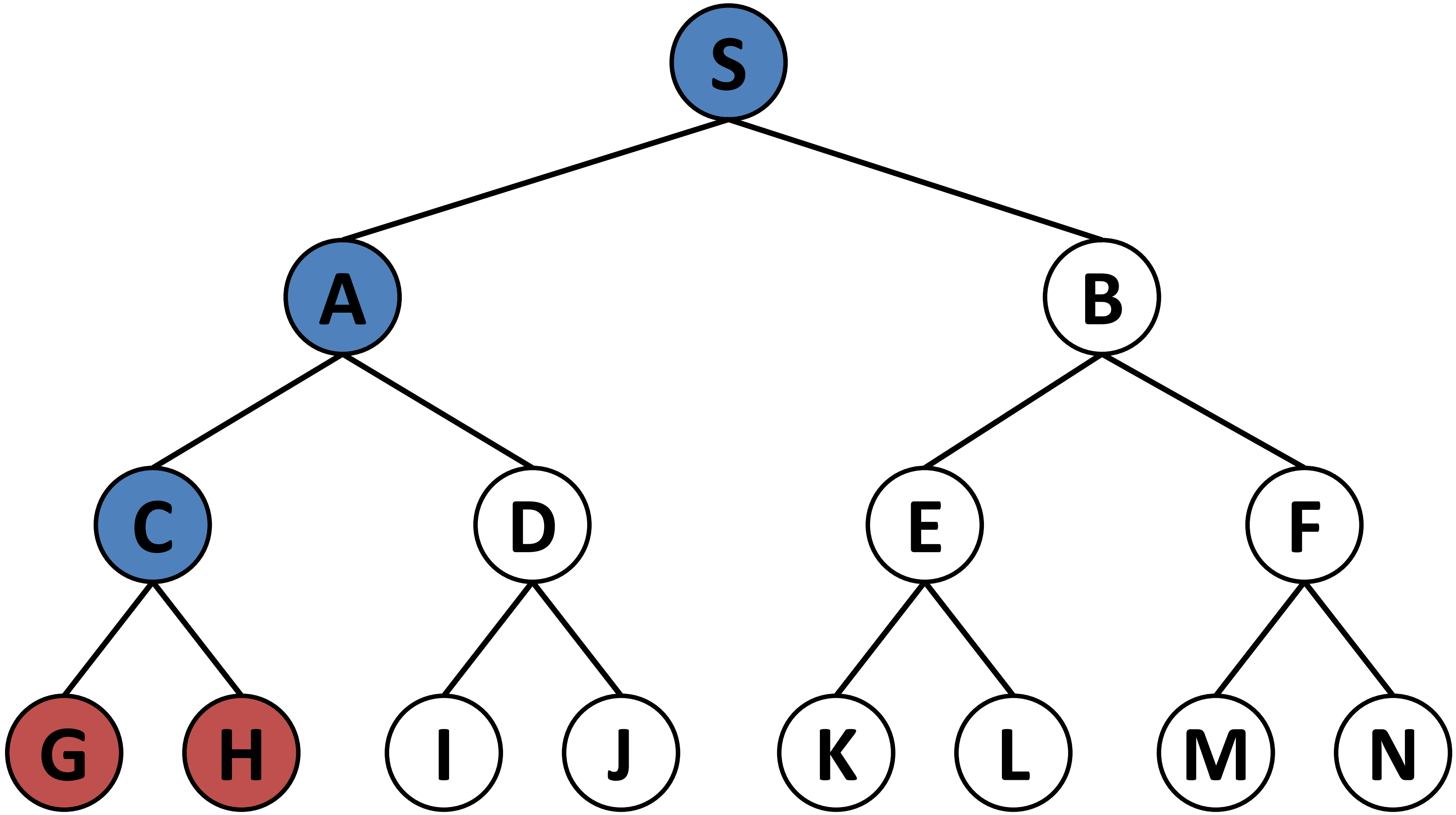
Graph and Tree Traversal: DFS



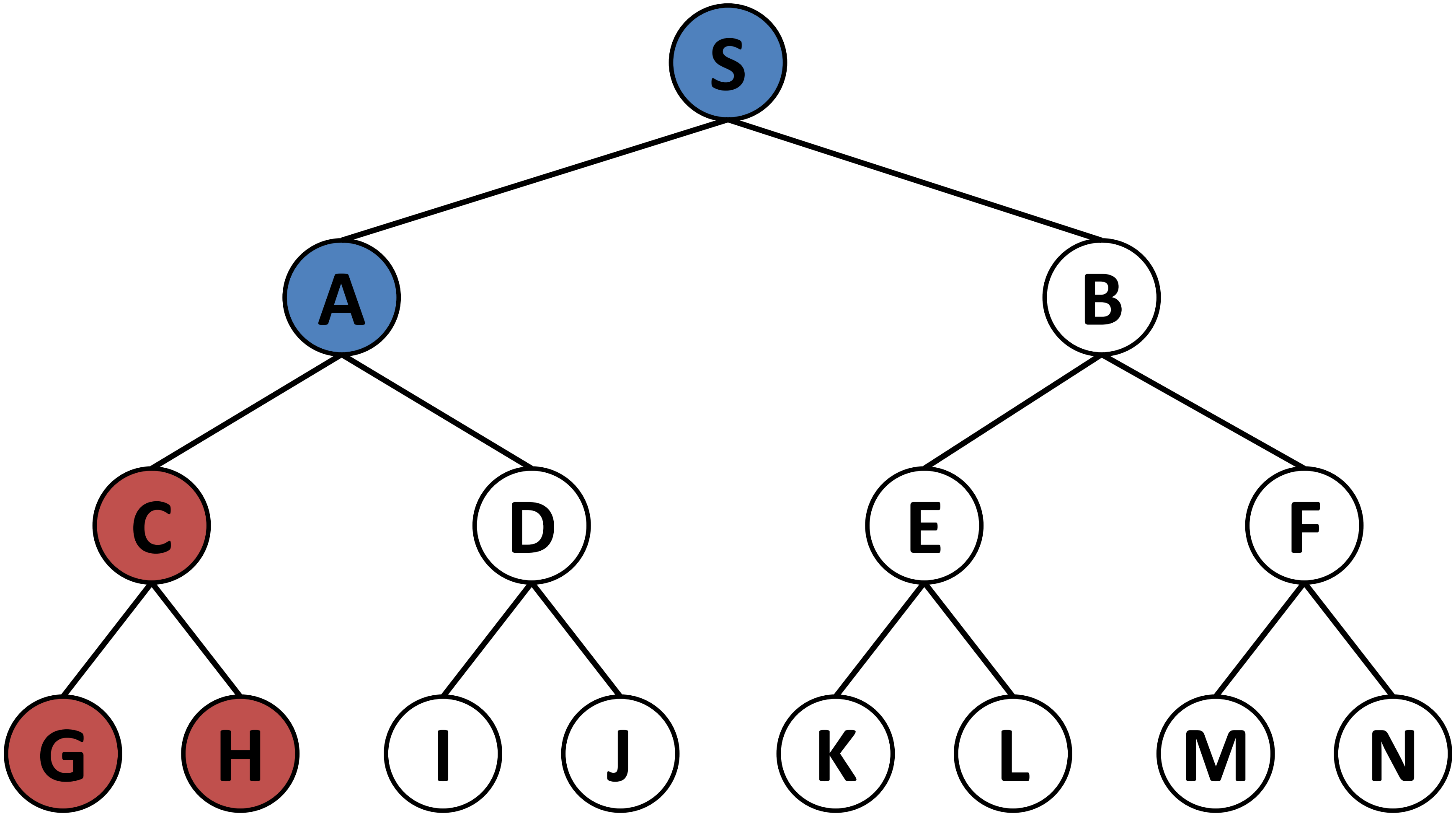
Graph and Tree Traversal: DFS



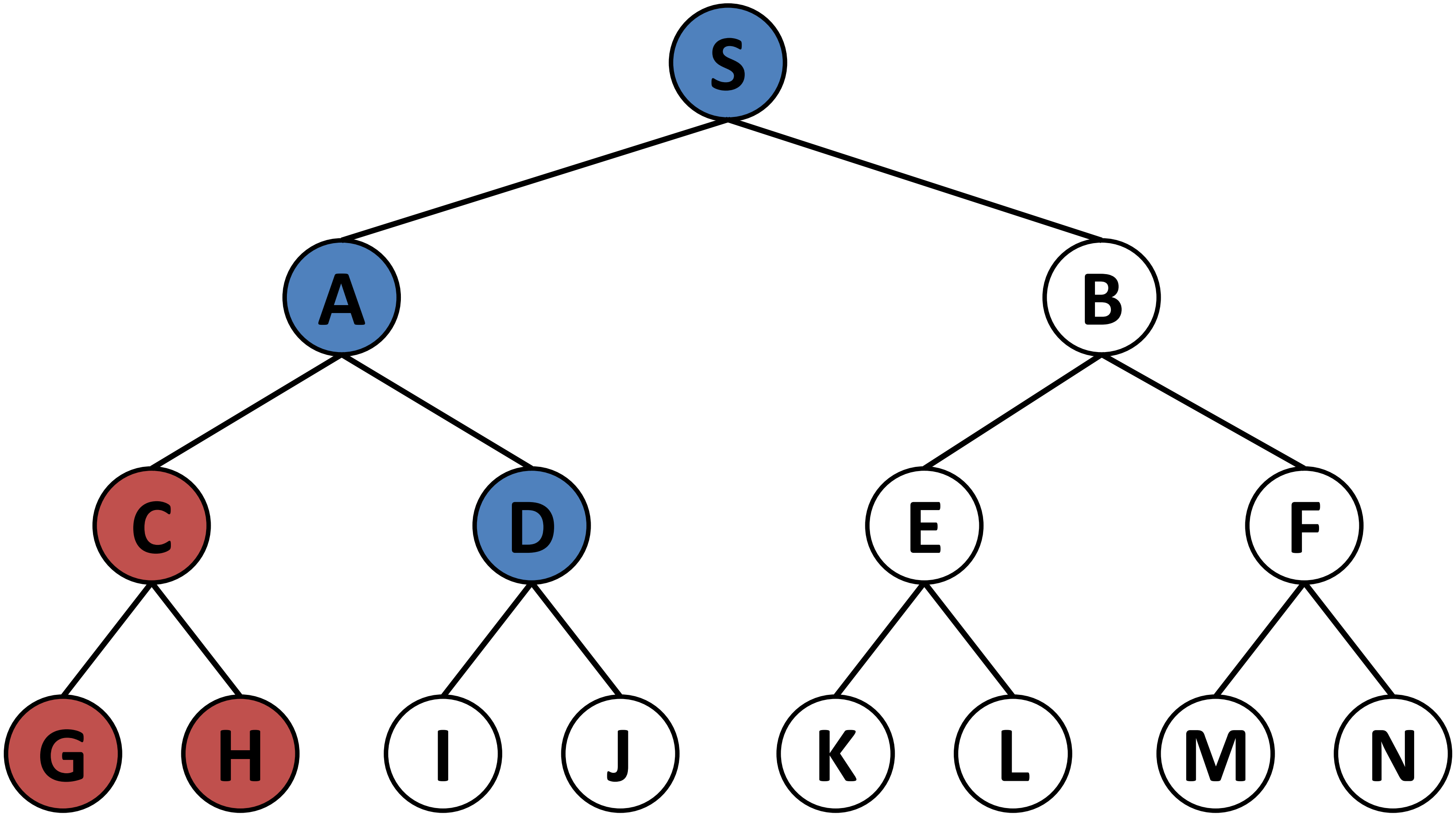
Graph and Tree Traversal: DFS



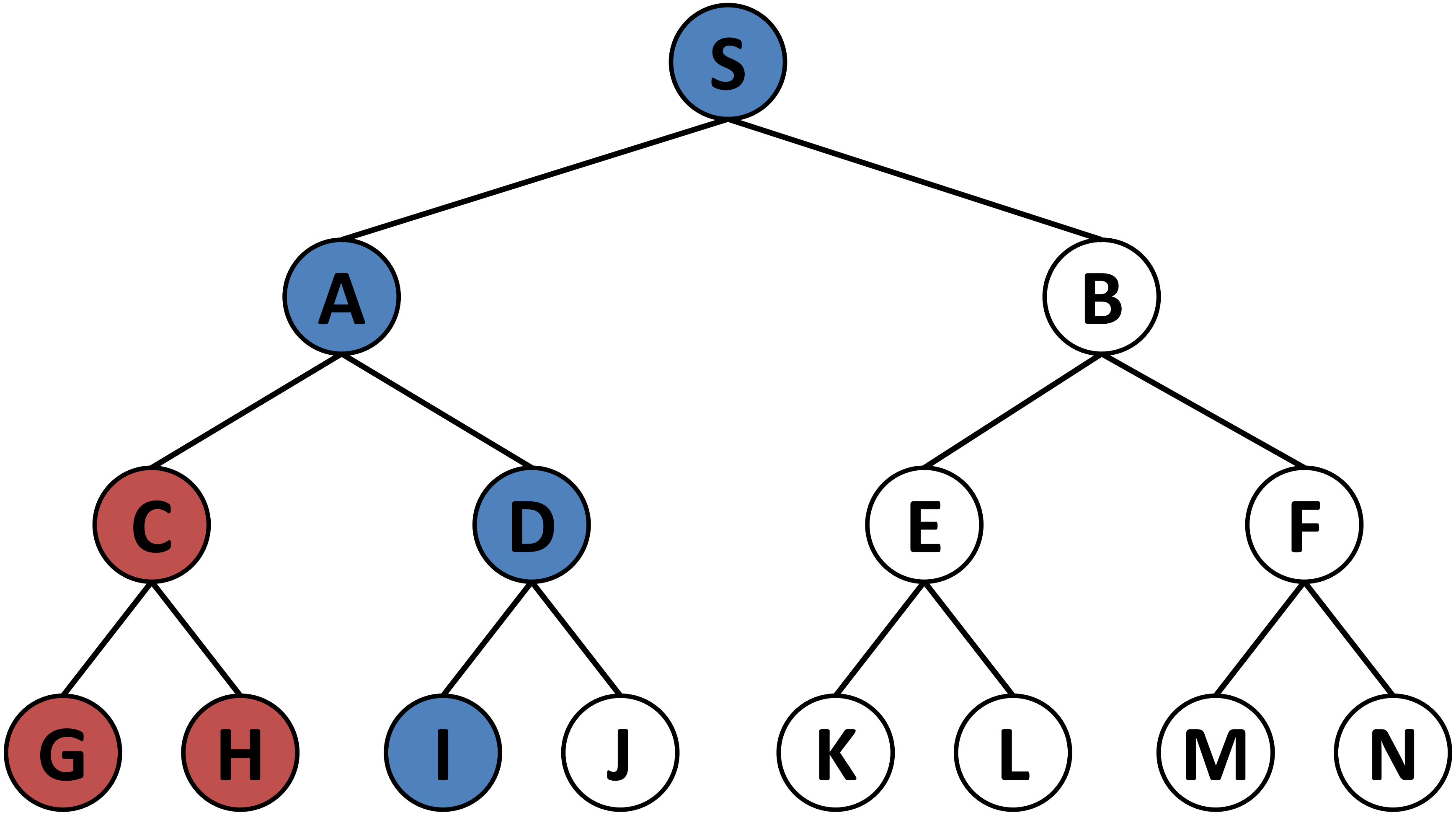
Graph and Tree Traversal: DFS



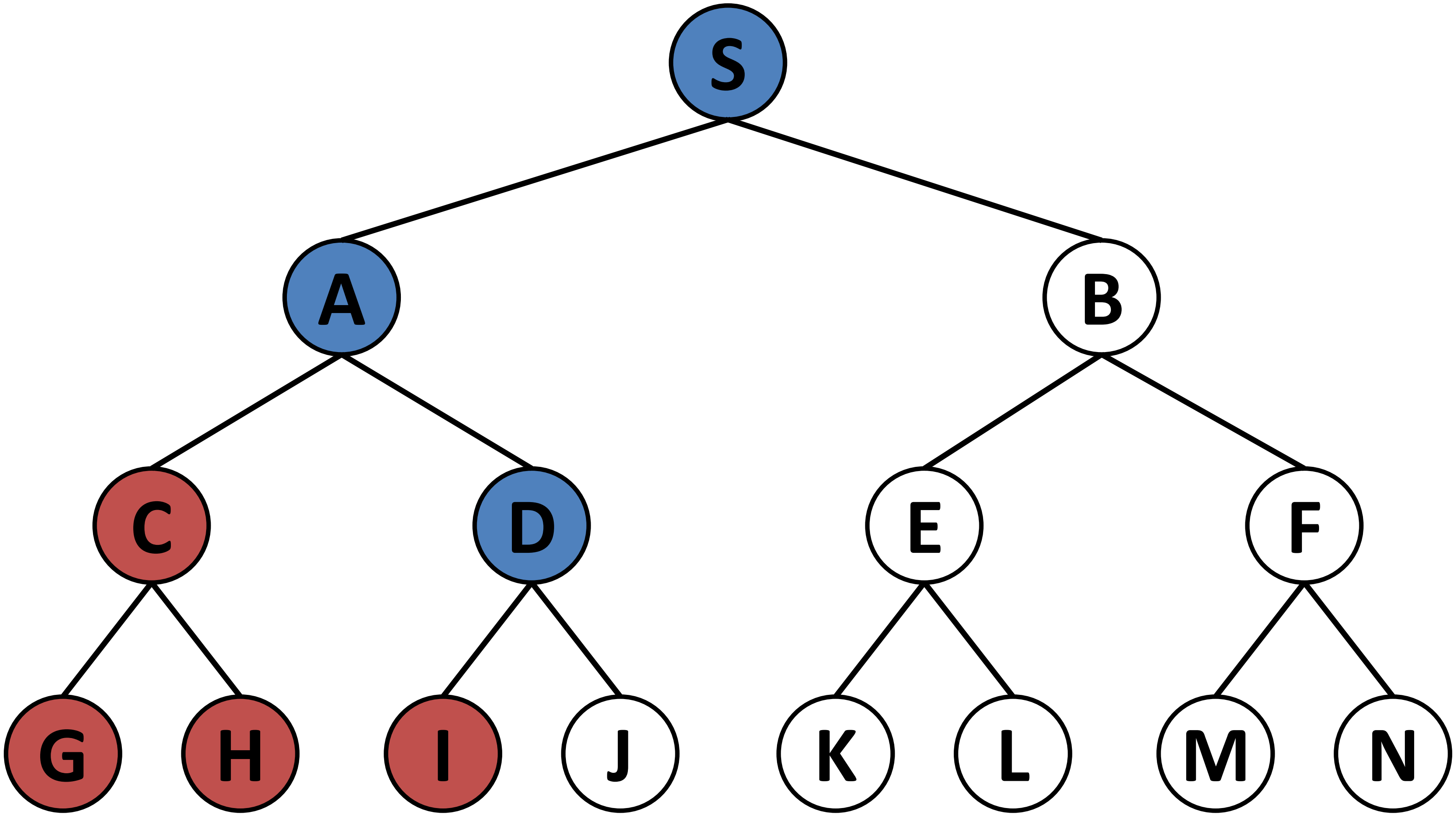
Graph and Tree Traversal: DFS



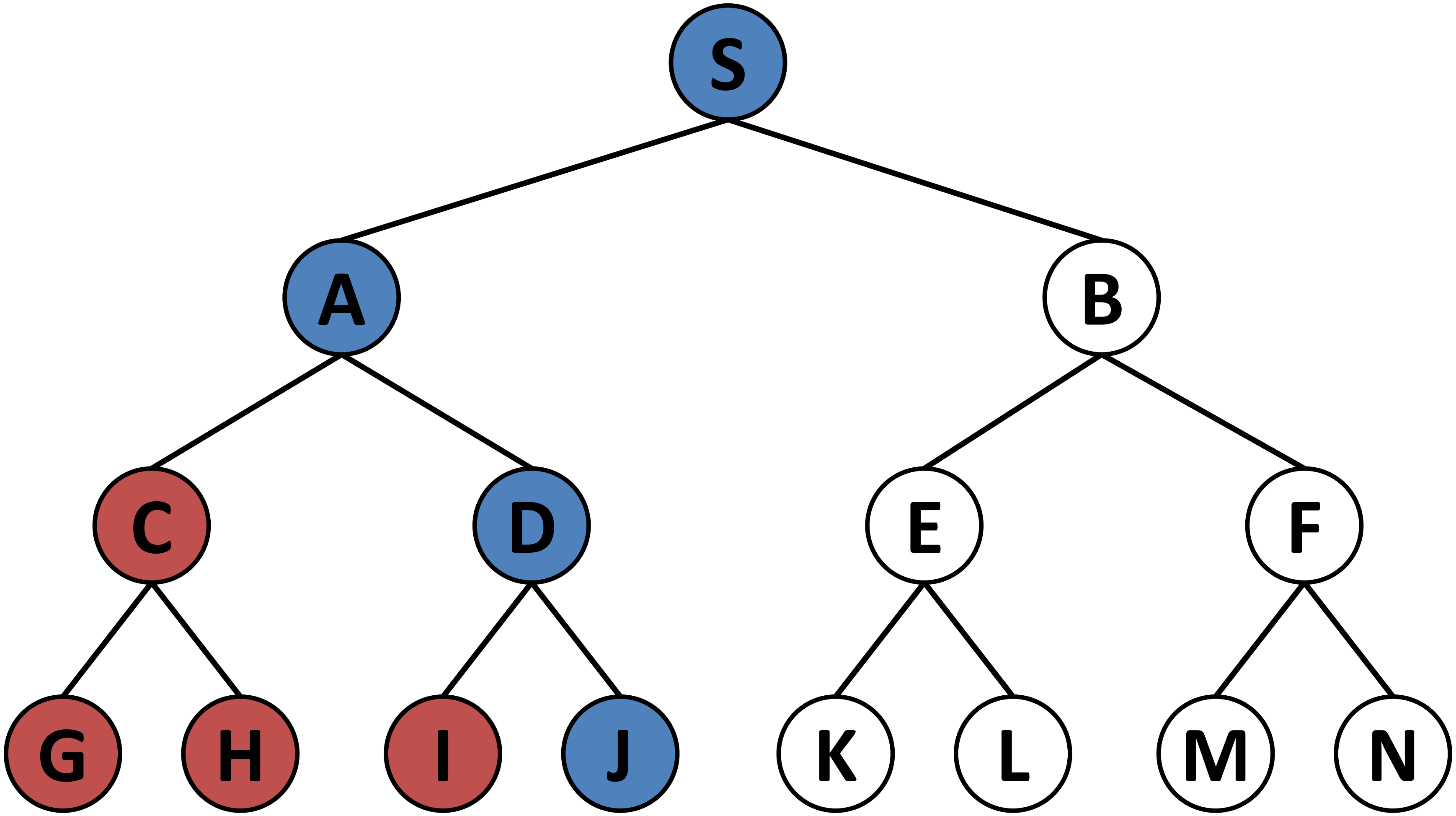
Graph and Tree Traversal: DFS



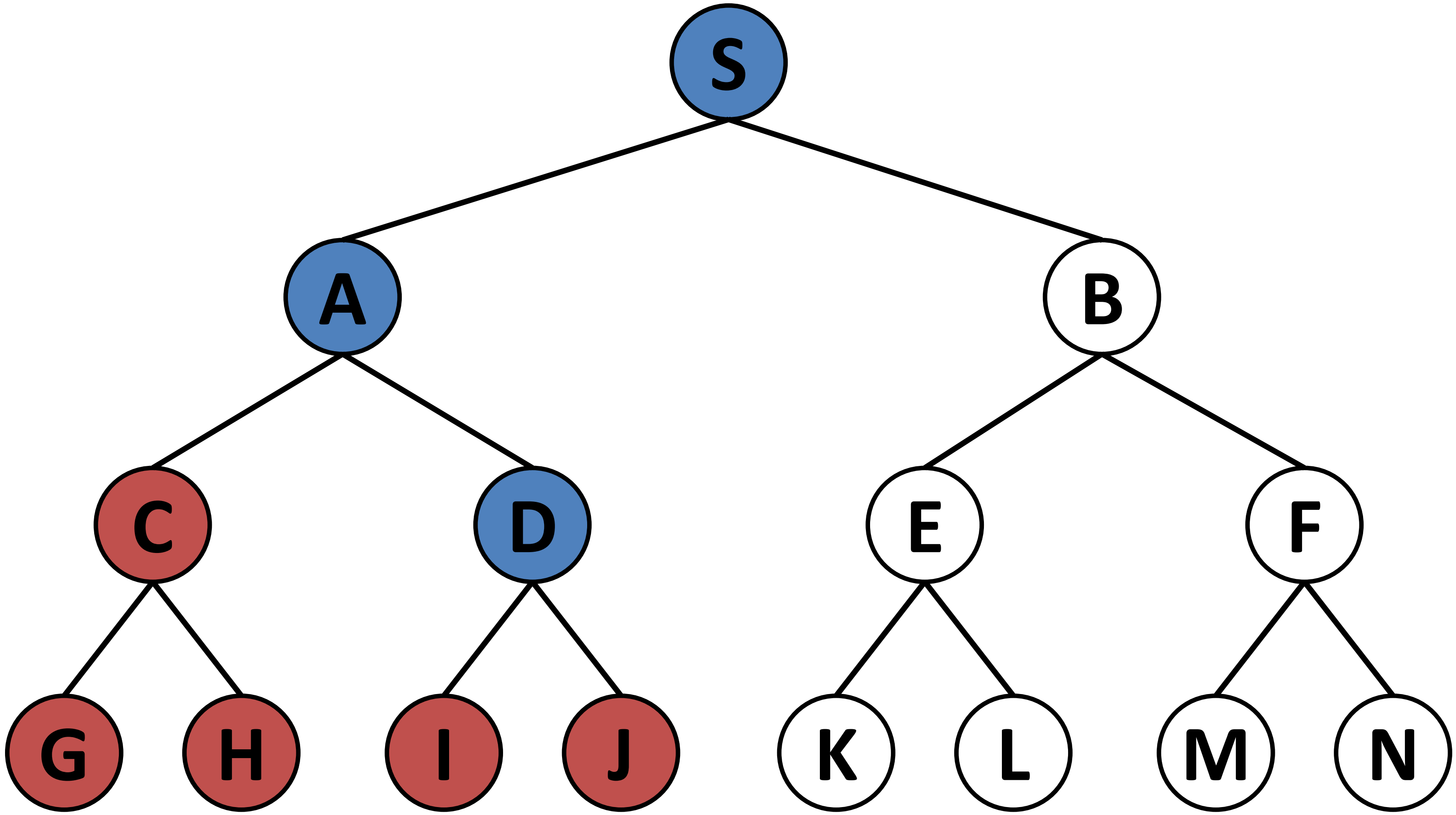
Graph and Tree Traversal: DFS



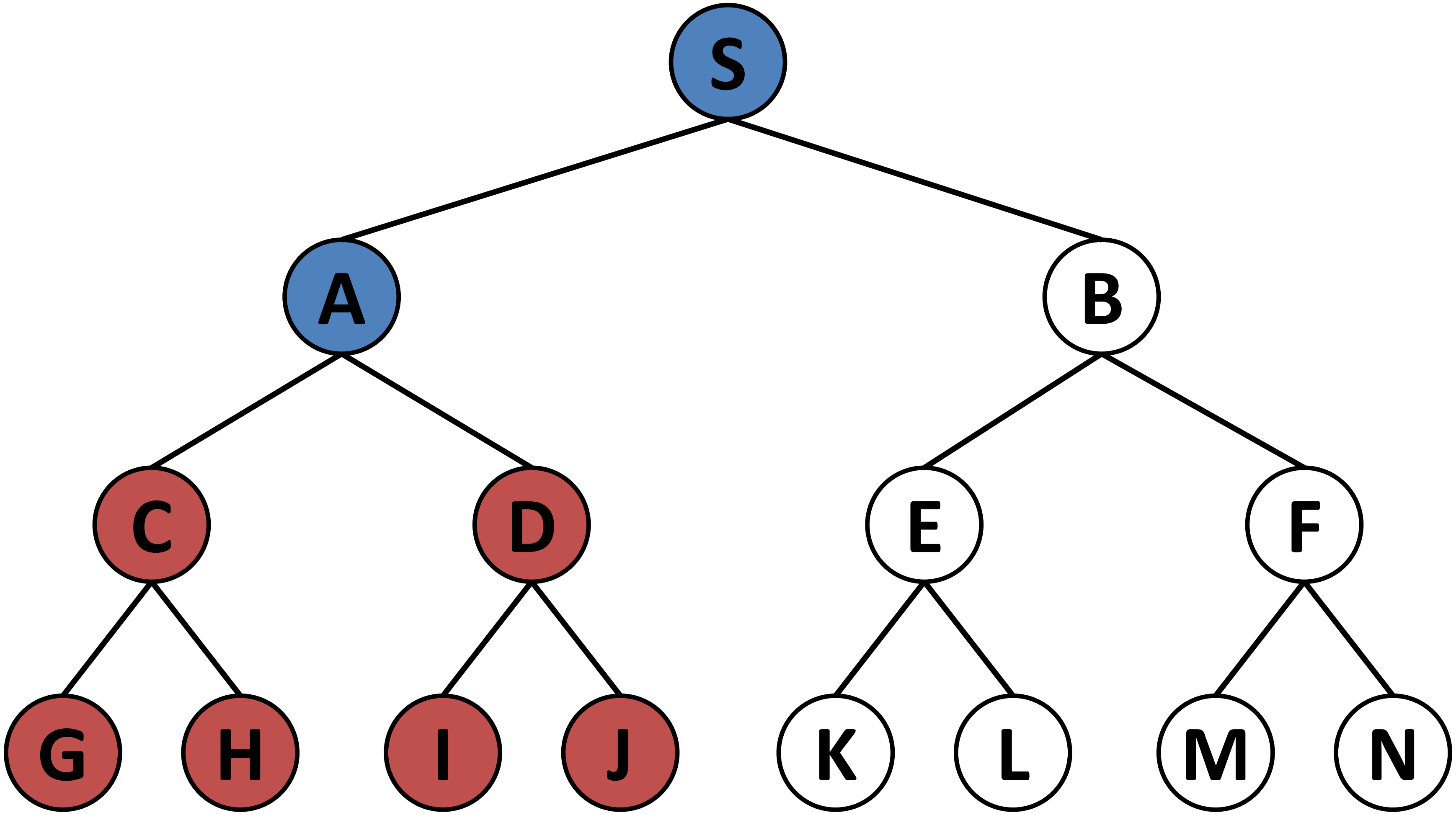
Graph and Tree Traversal: DFS



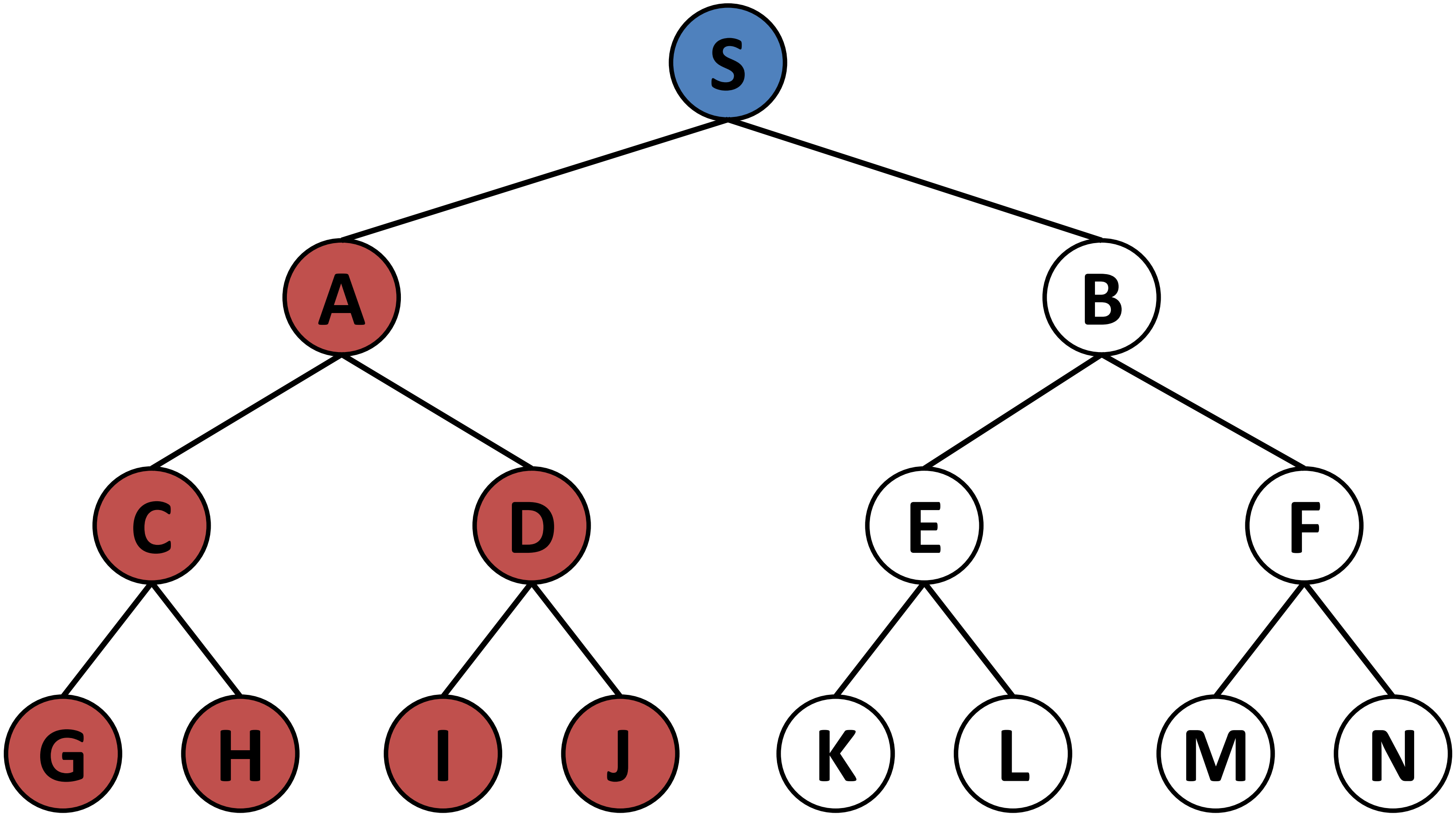
Graph and Tree Traversal: DFS



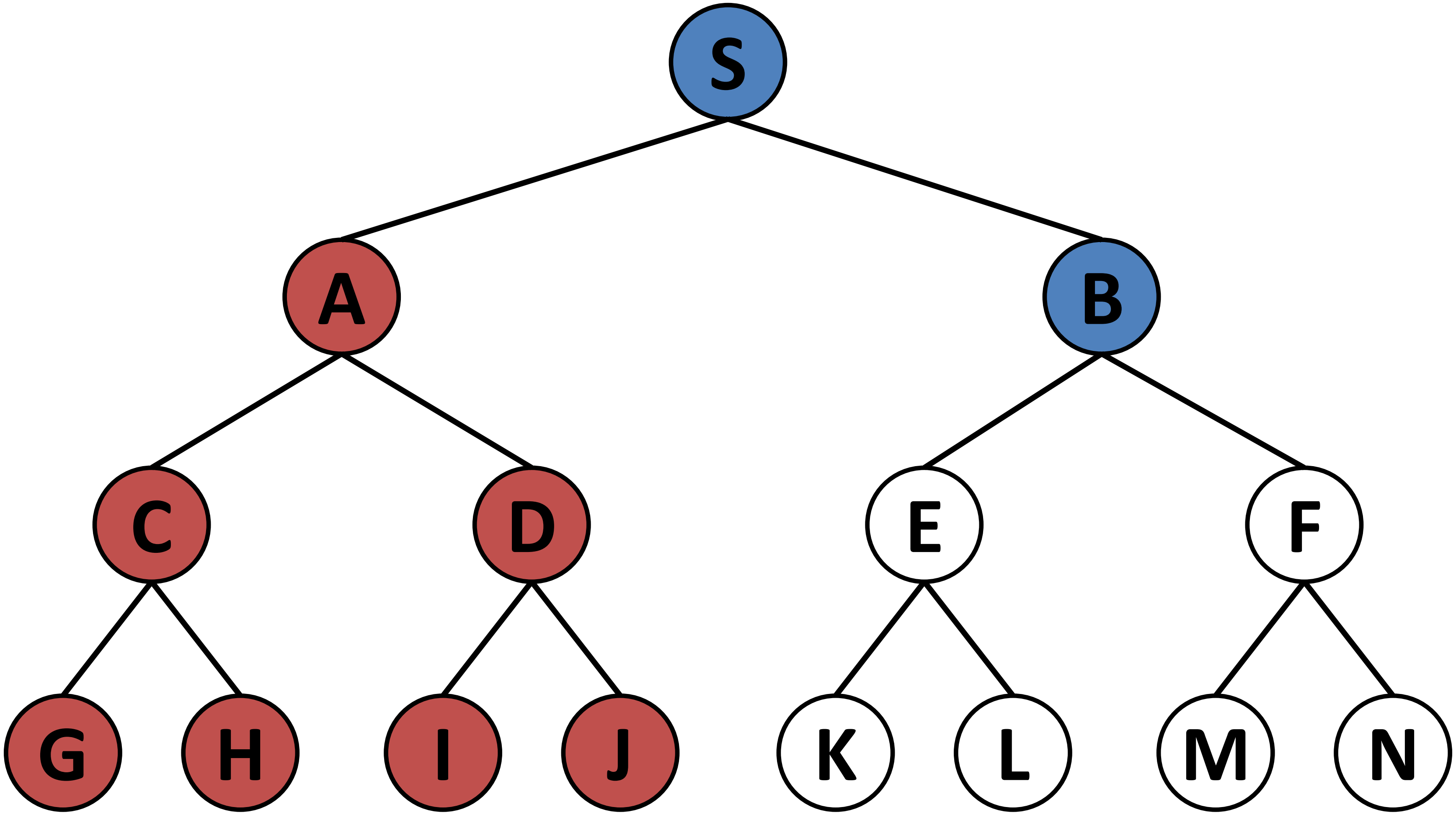
Graph and Tree Traversal: DFS



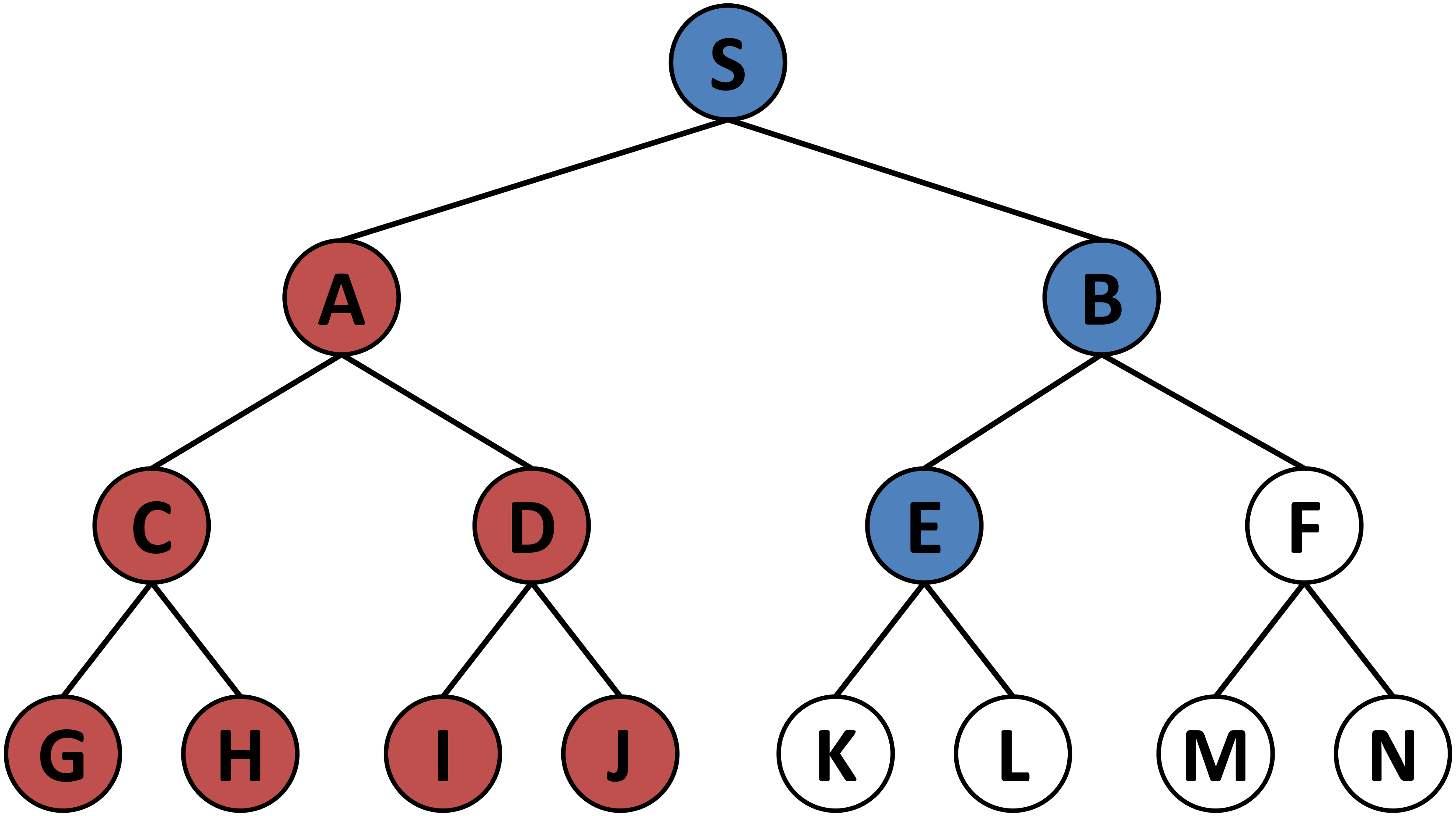
Graph and Tree Traversal: DFS



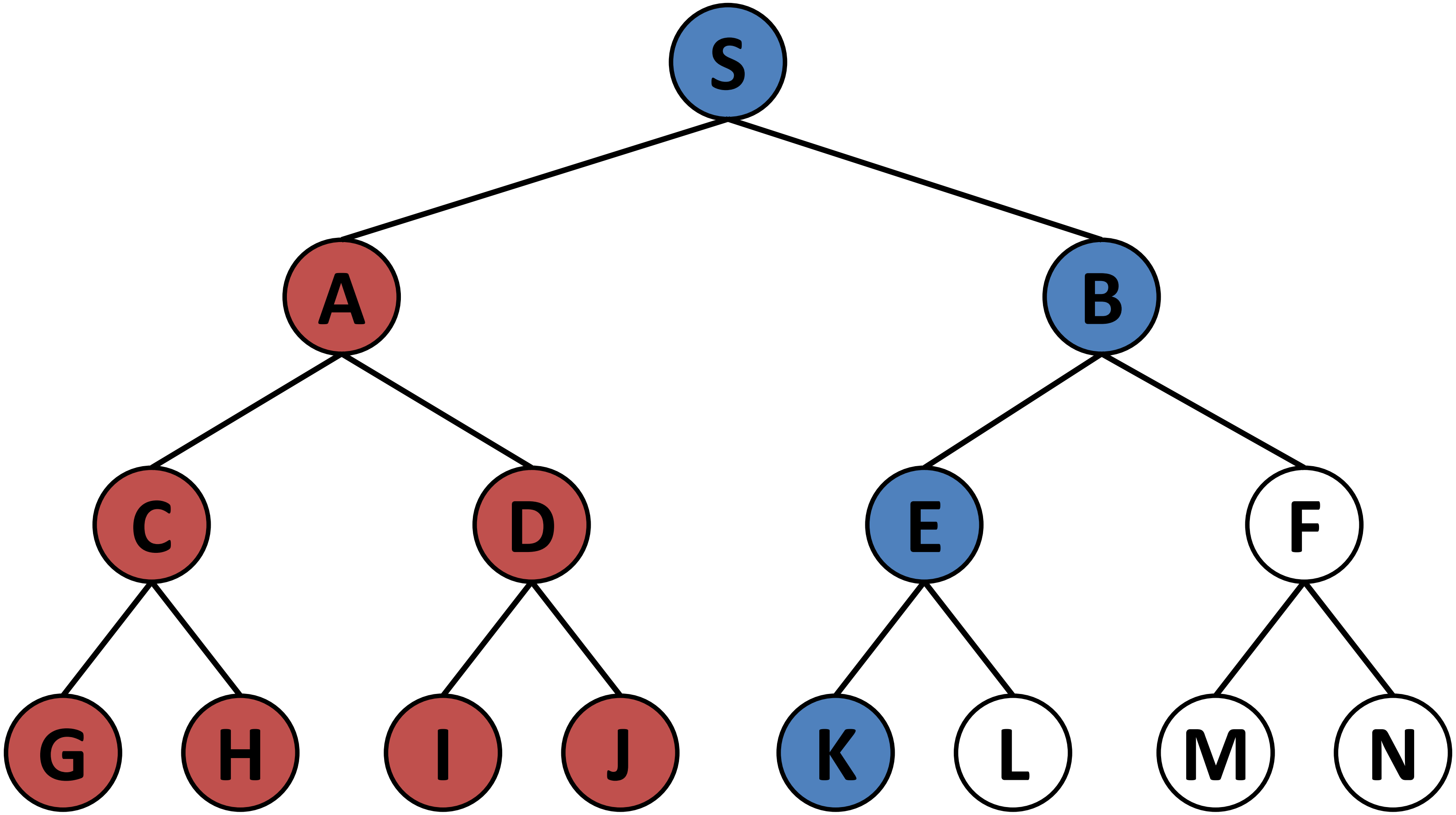
Graph and Tree Traversal: DFS



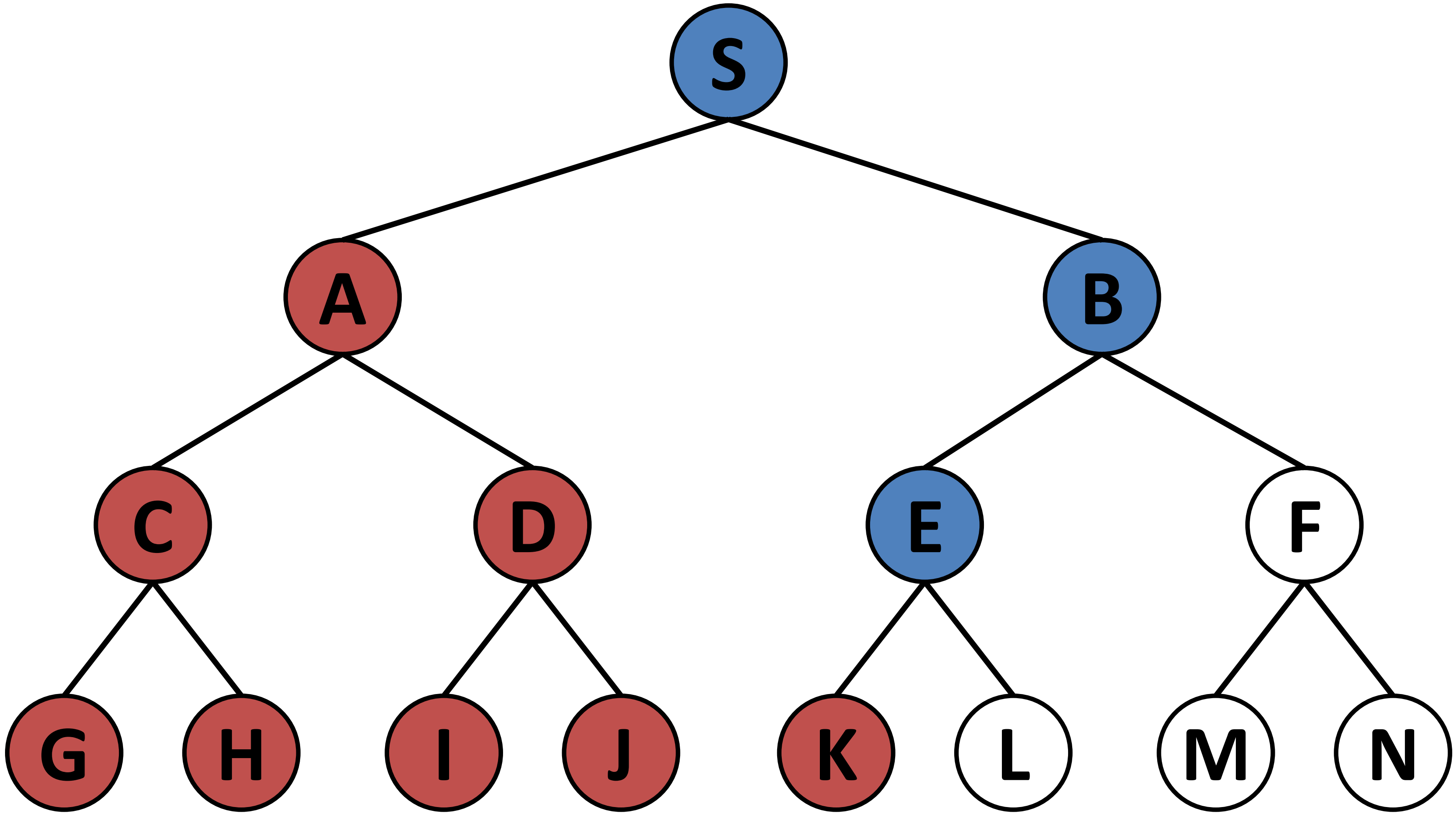
Graph and Tree Traversal: DFS



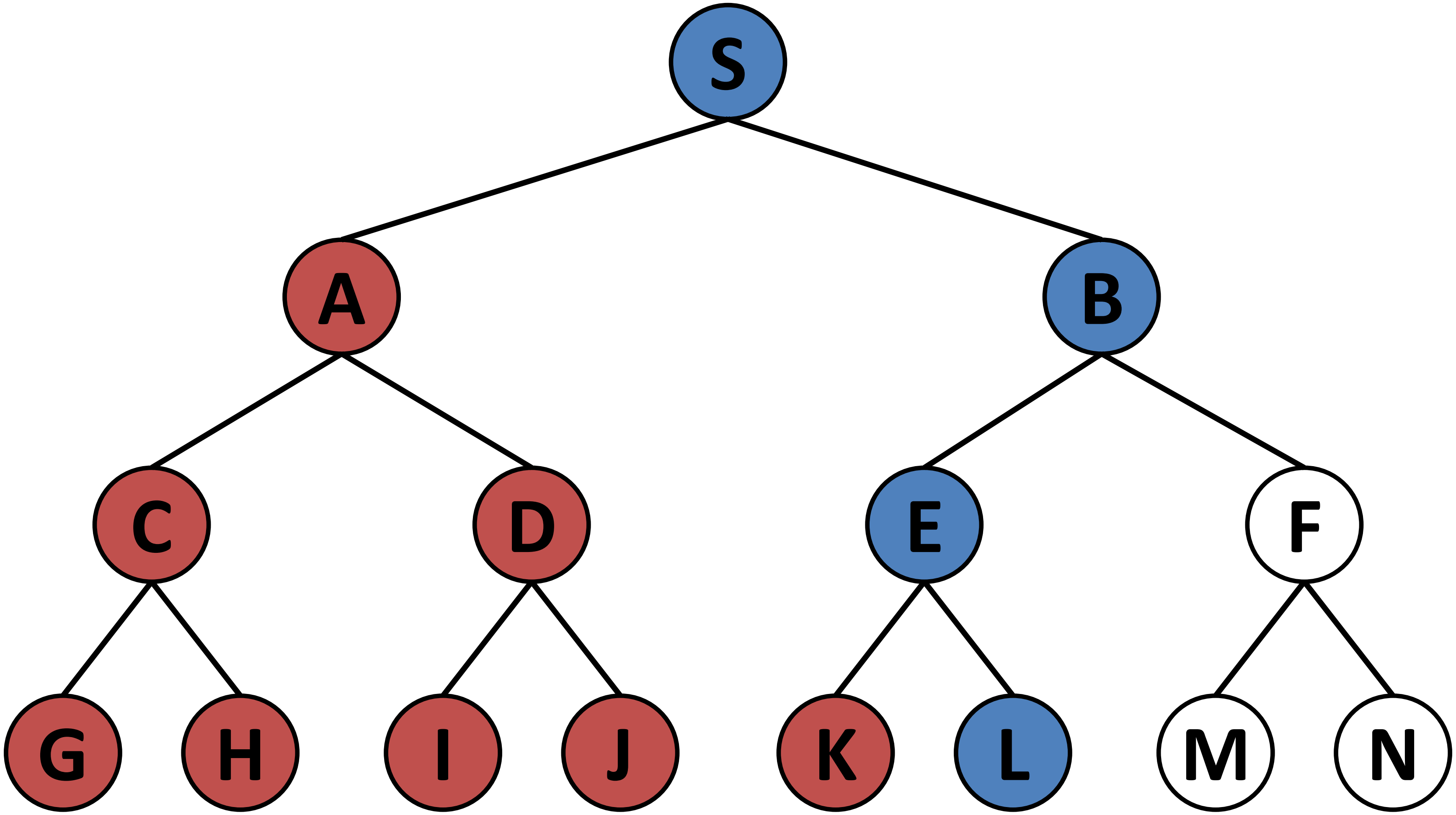
Graph and Tree Traversal: DFS



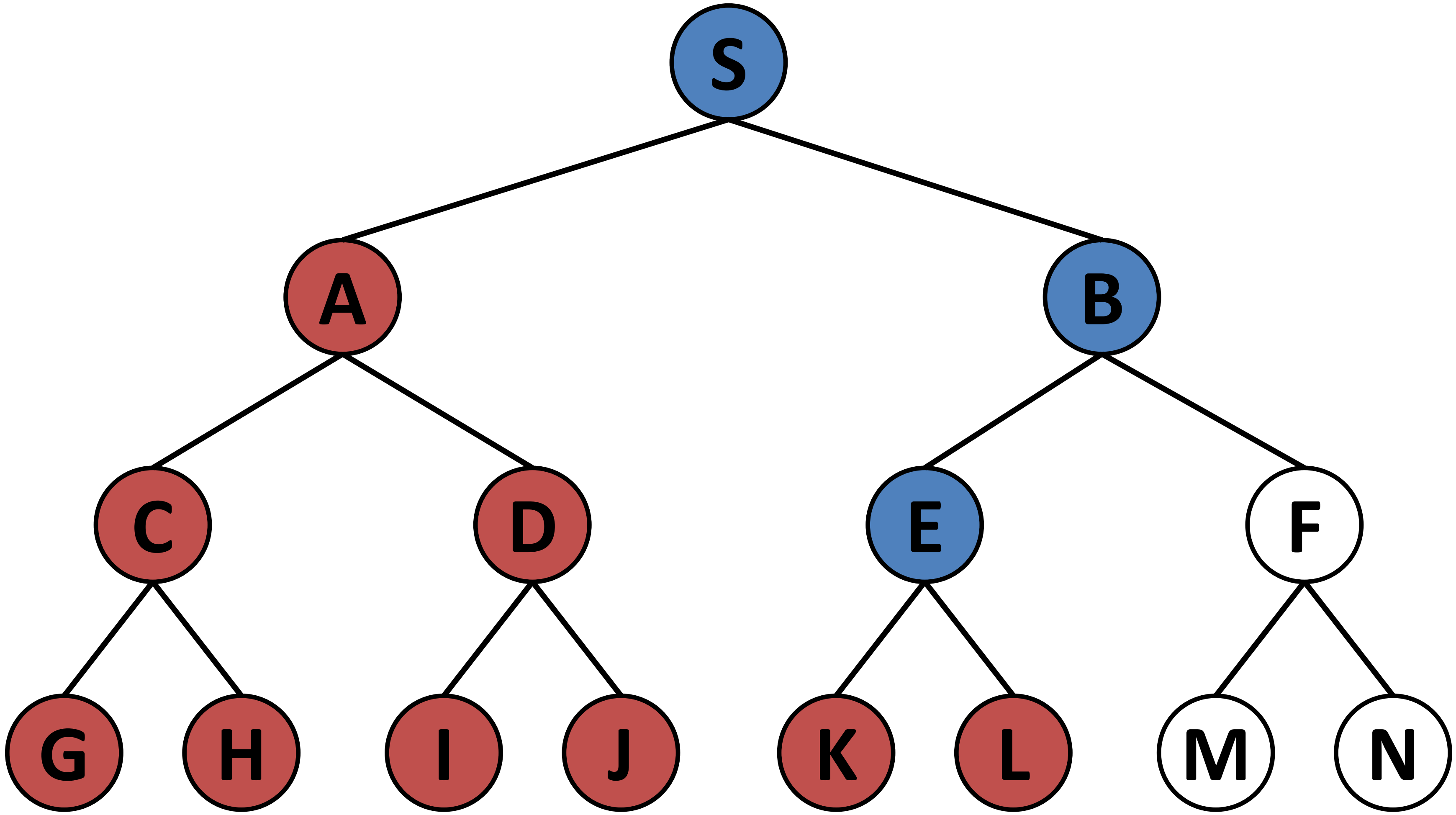
Graph and Tree Traversal: DFS



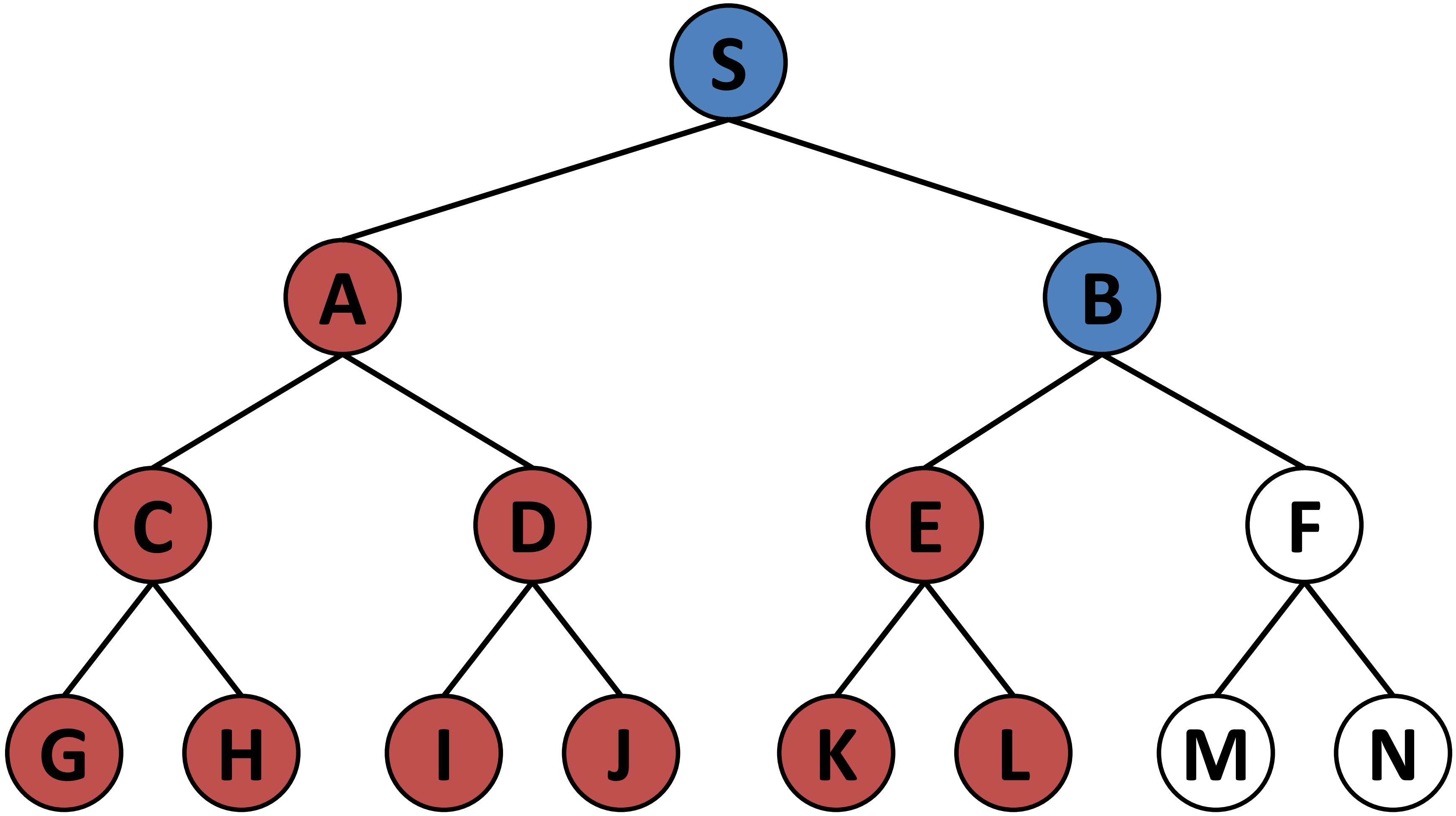
Graph and Tree Traversal: DFS



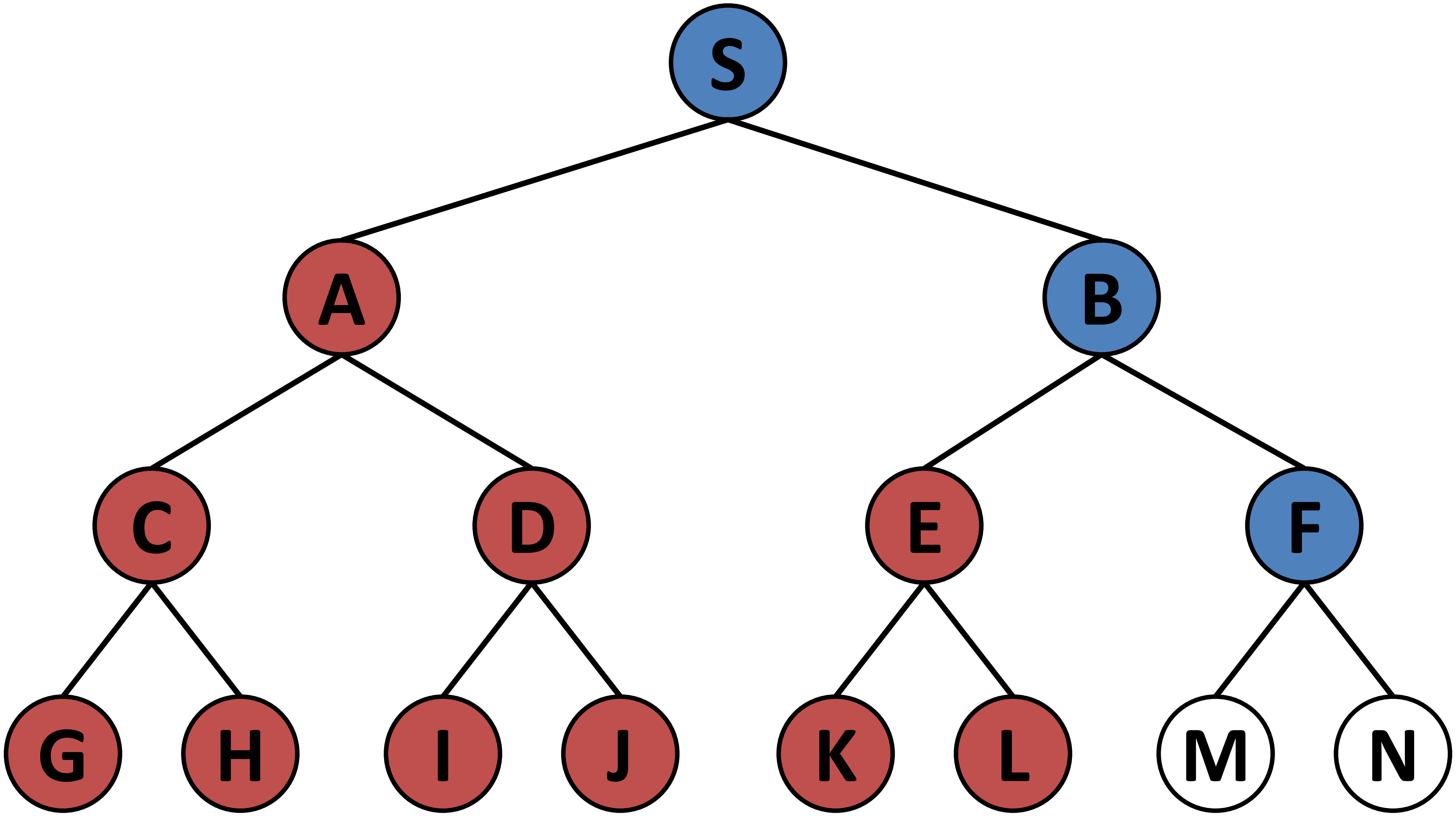
Graph and Tree Traversal: DFS



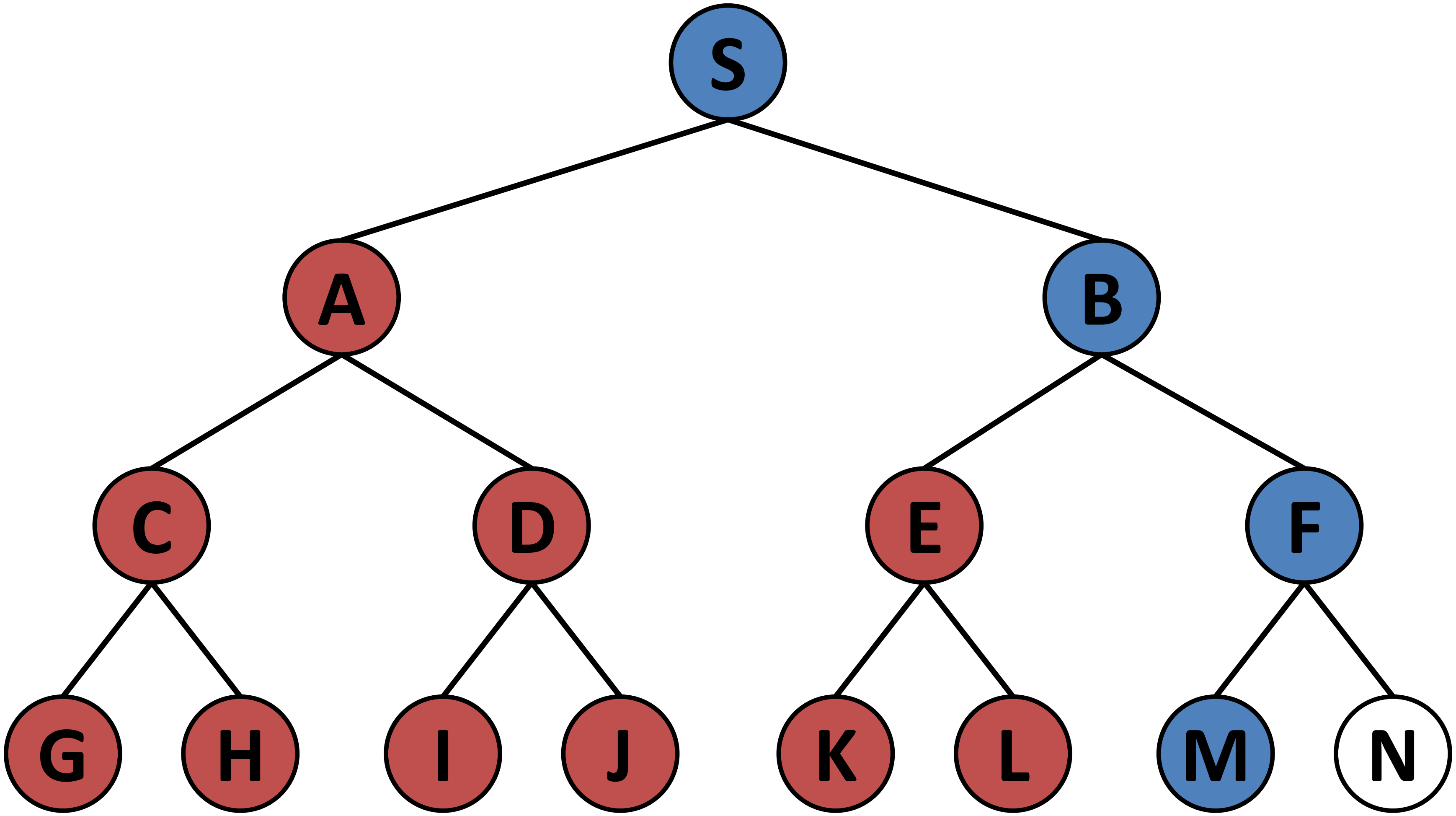
Graph and Tree Traversal: DFS



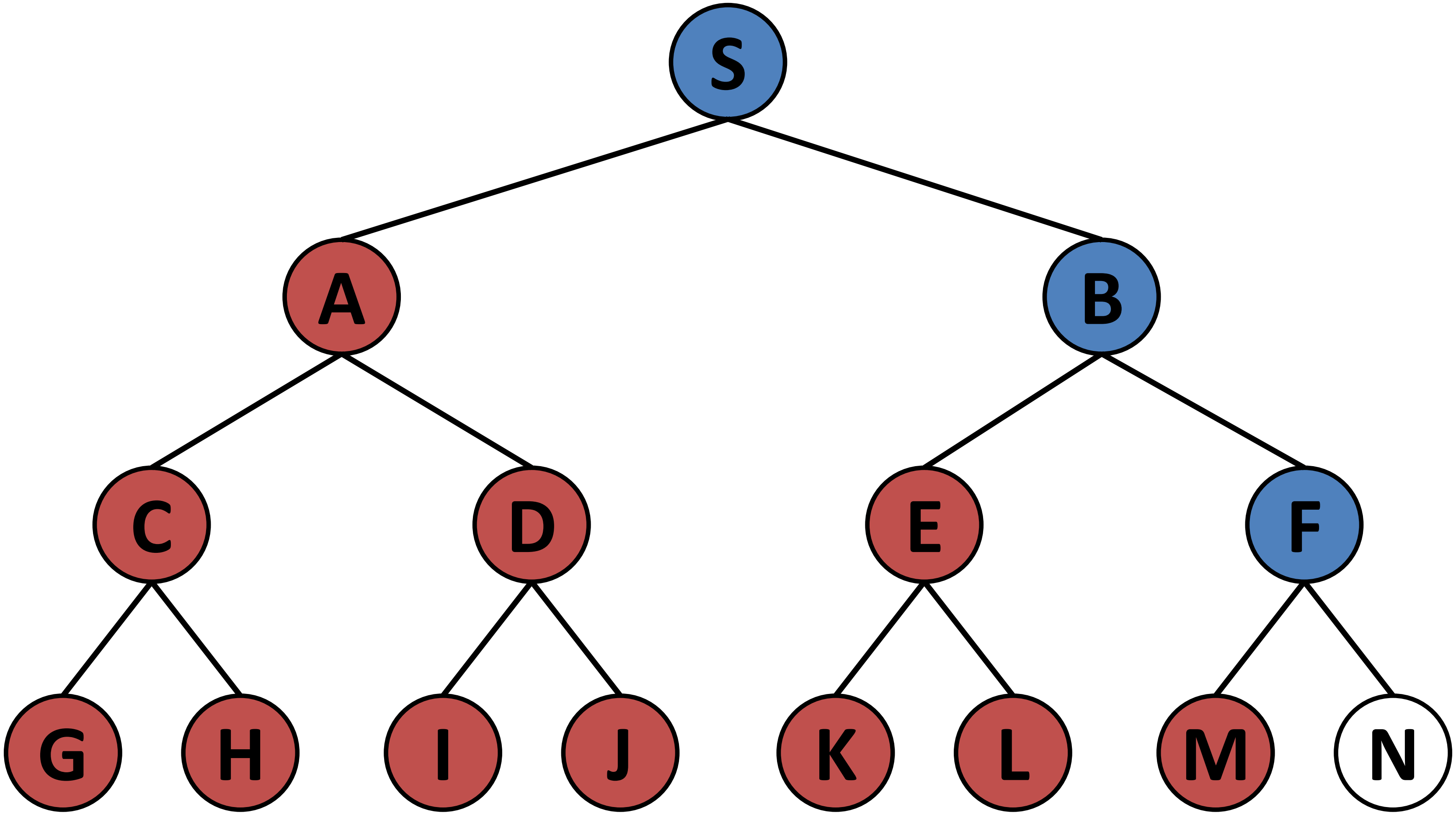
Graph and Tree Traversal: DFS



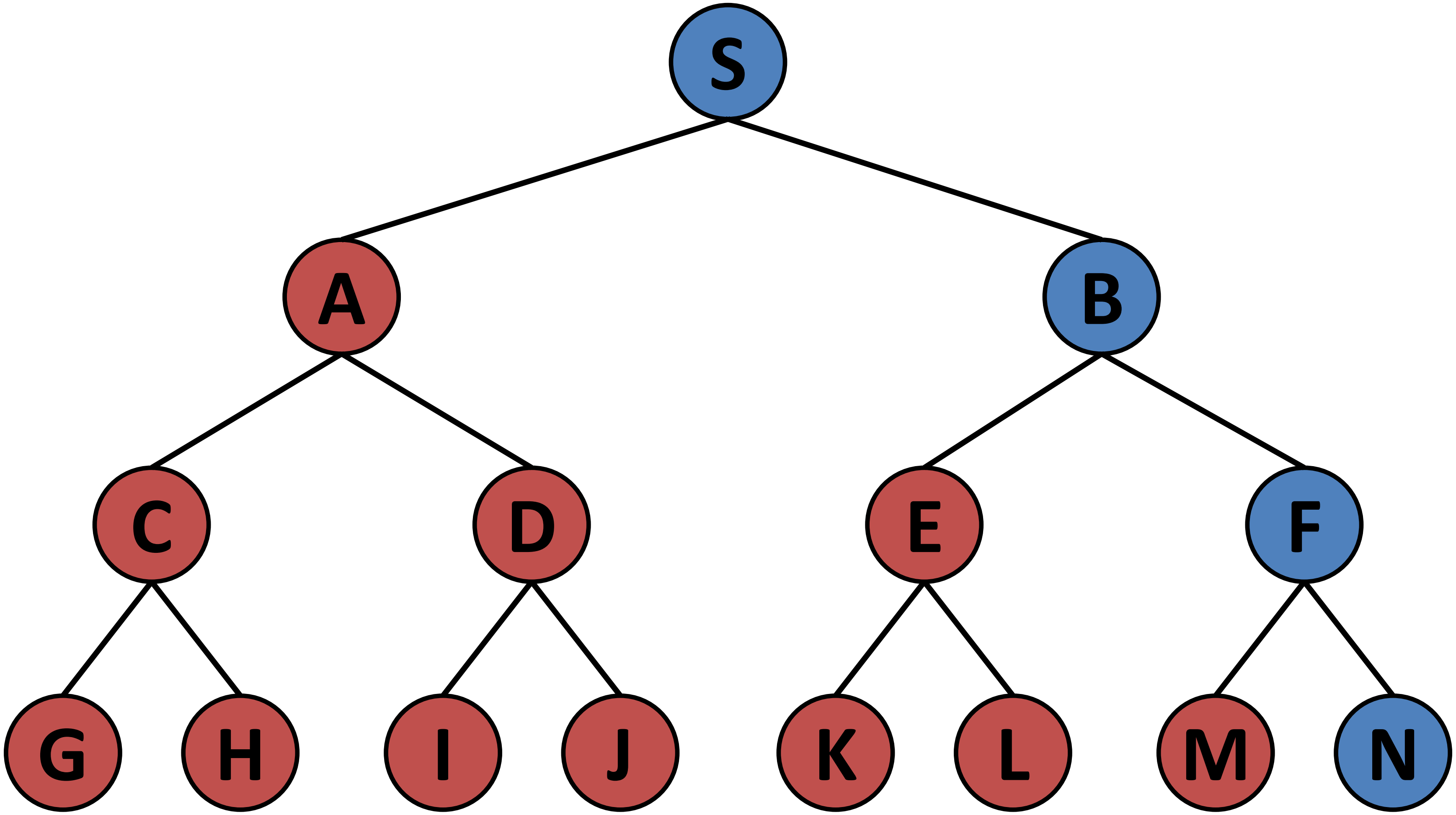
Graph and Tree Traversal: DFS



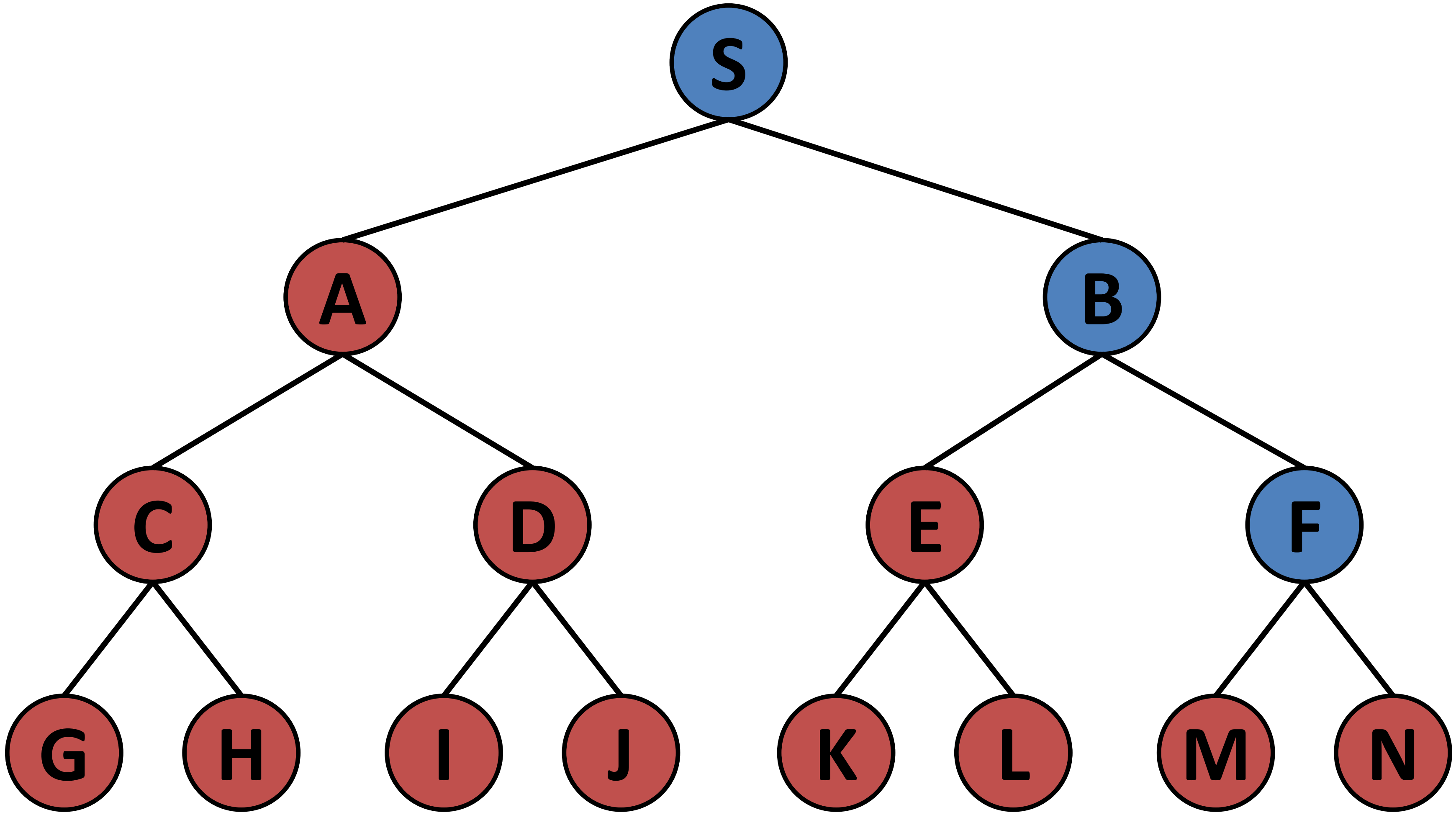
Graph and Tree Traversal: DFS



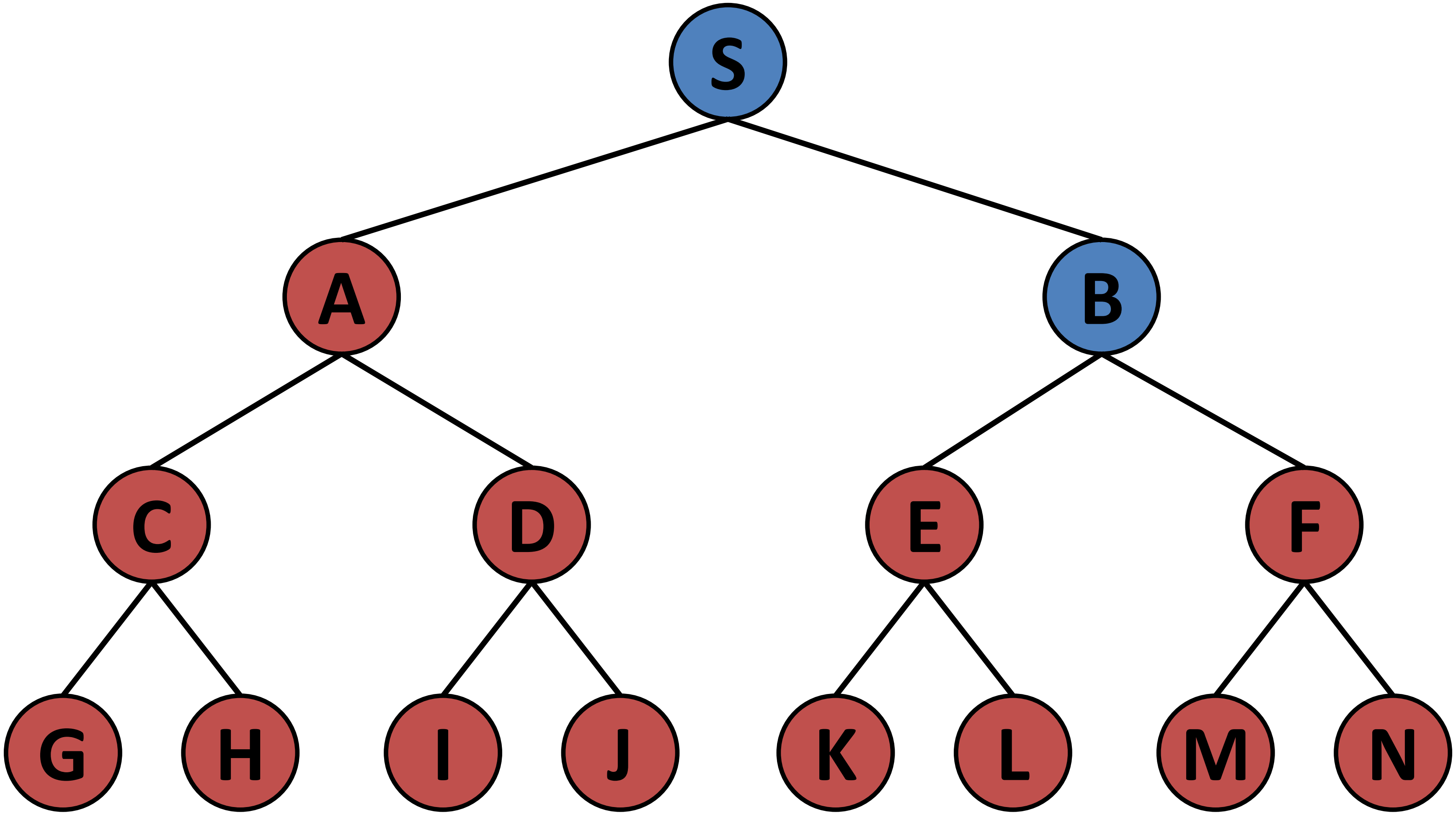
Graph and Tree Traversal: DFS



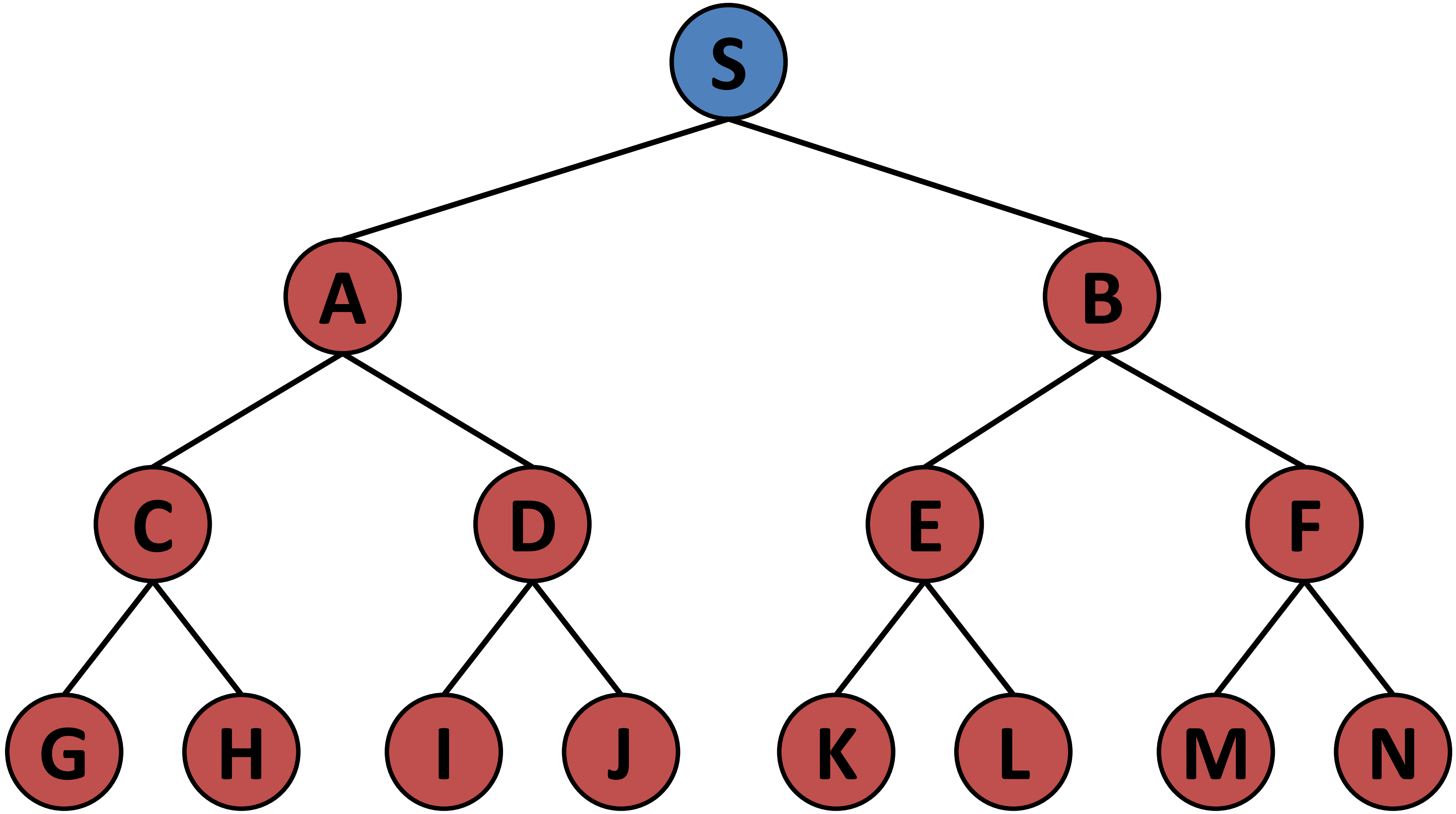
Graph and Tree Traversal: DFS



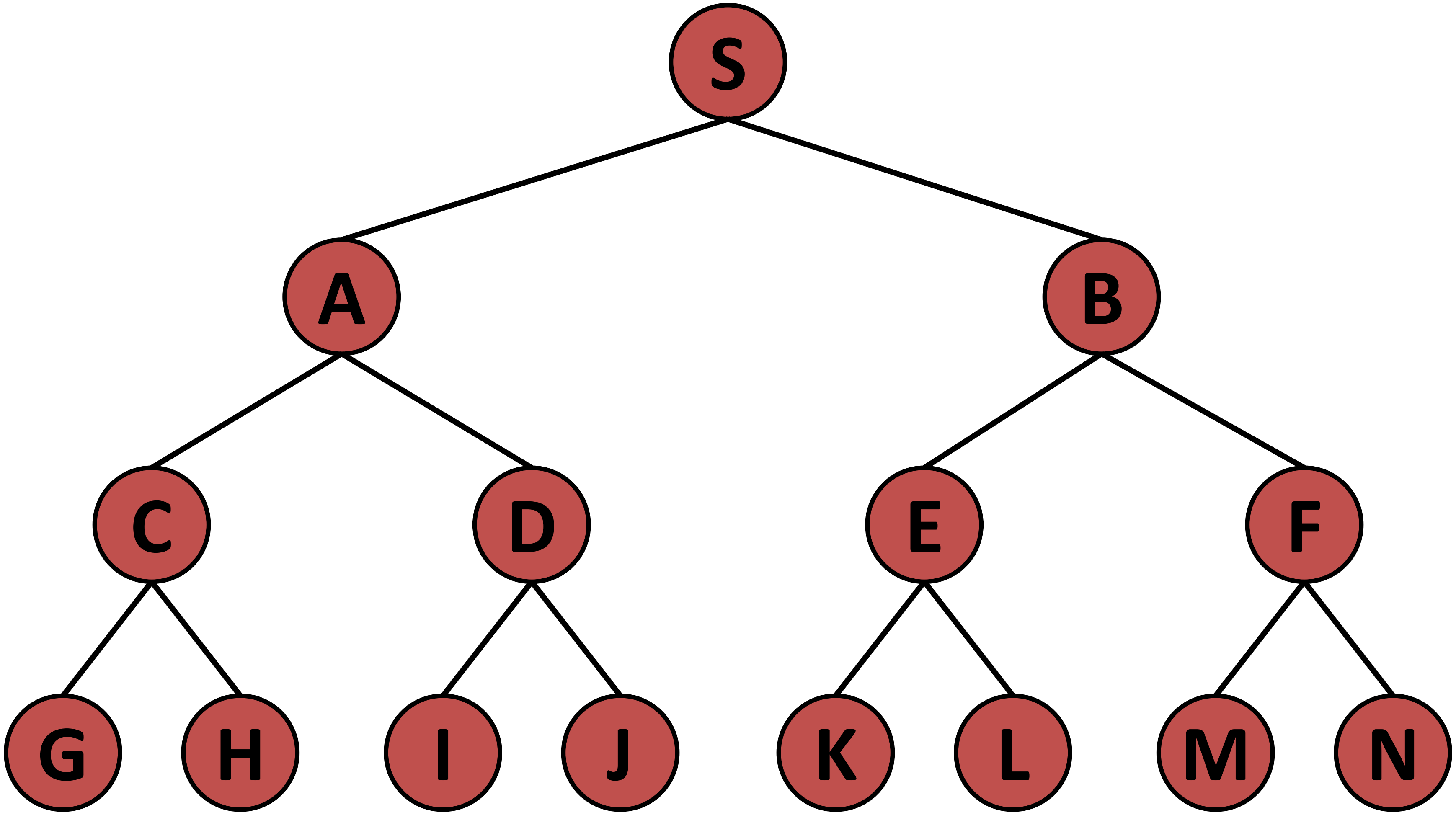
Graph and Tree Traversal: DFS



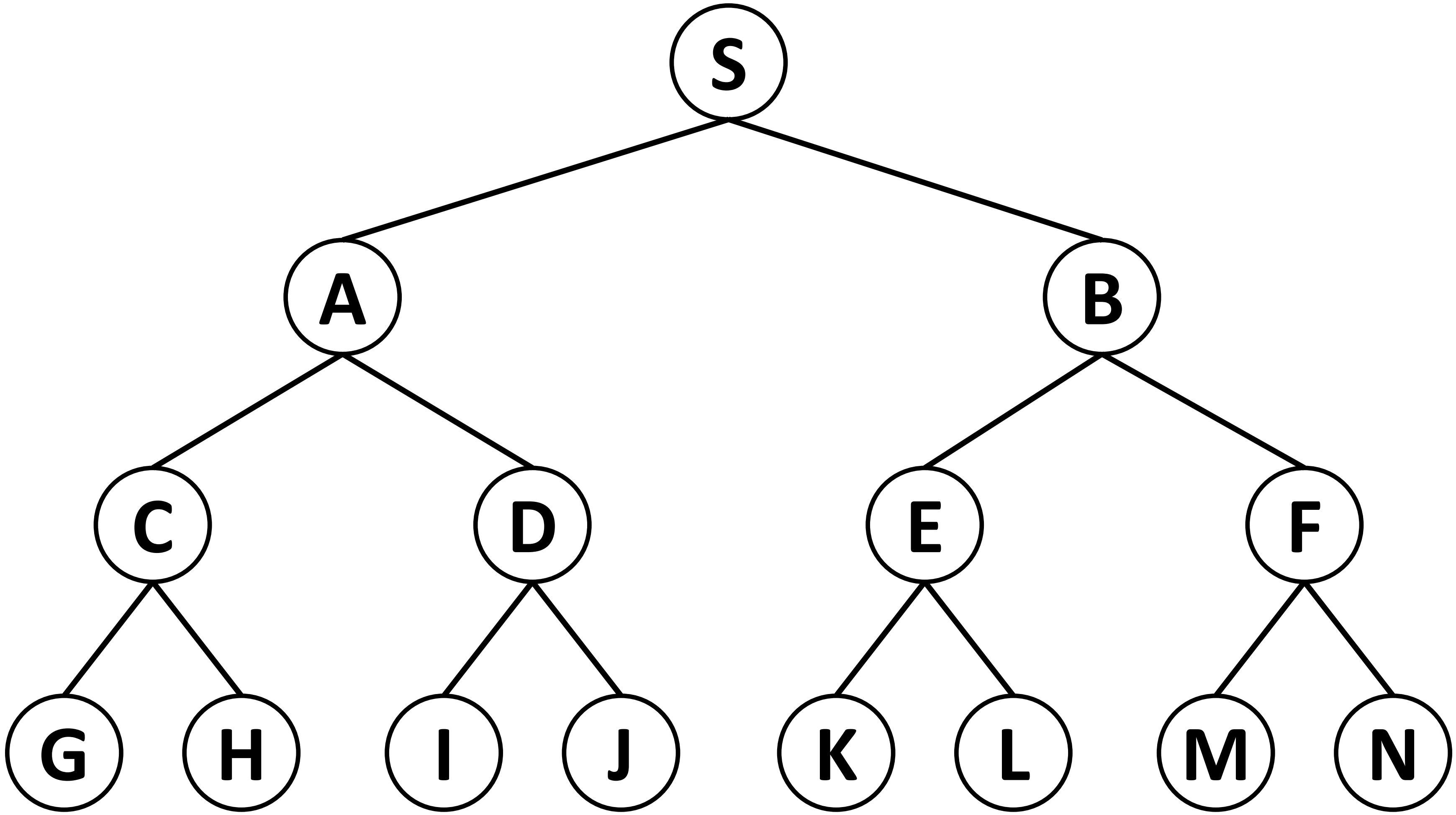
Graph and Tree Traversal: DFS



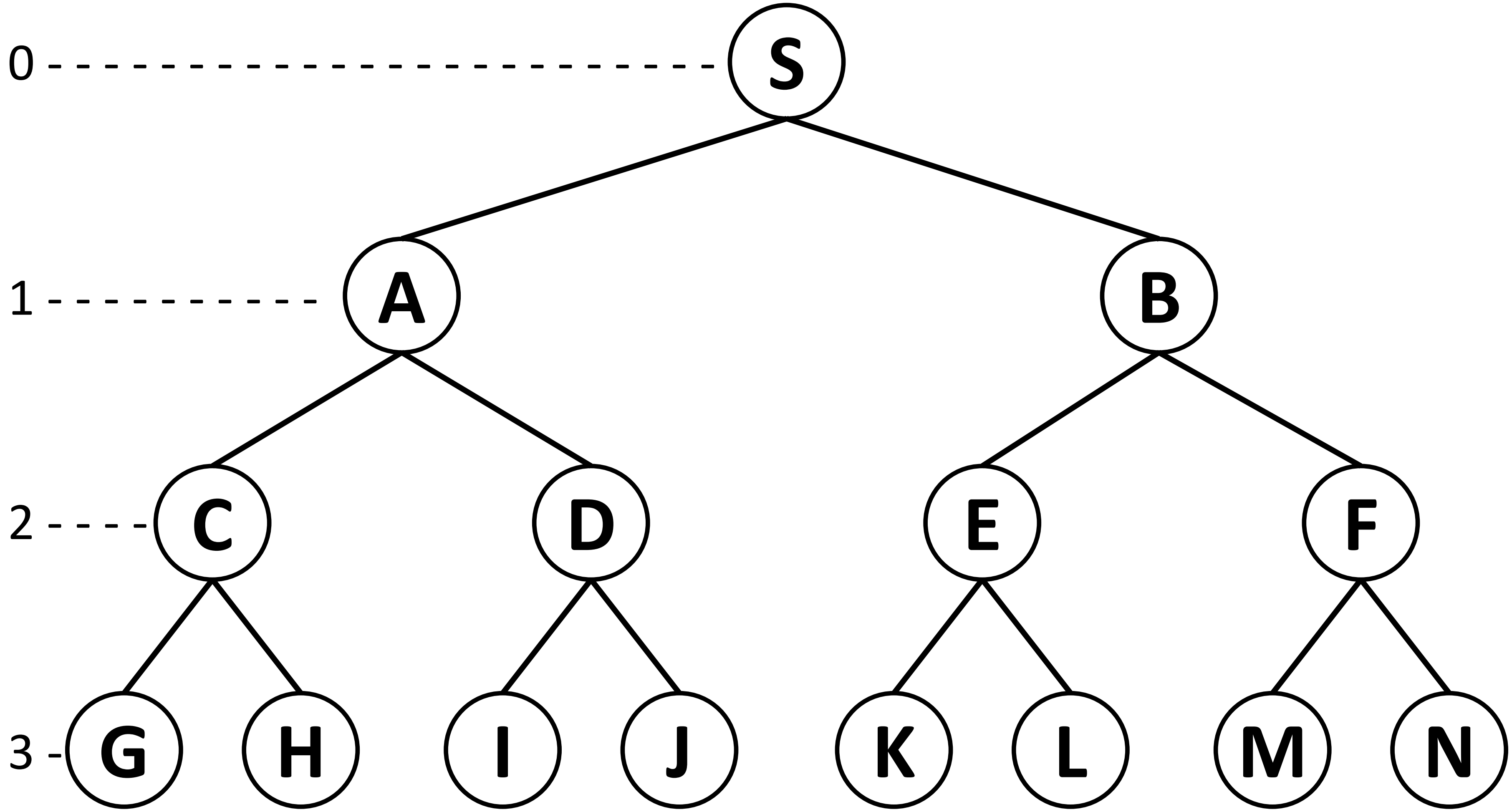
Graph and Tree Traversal: DFS



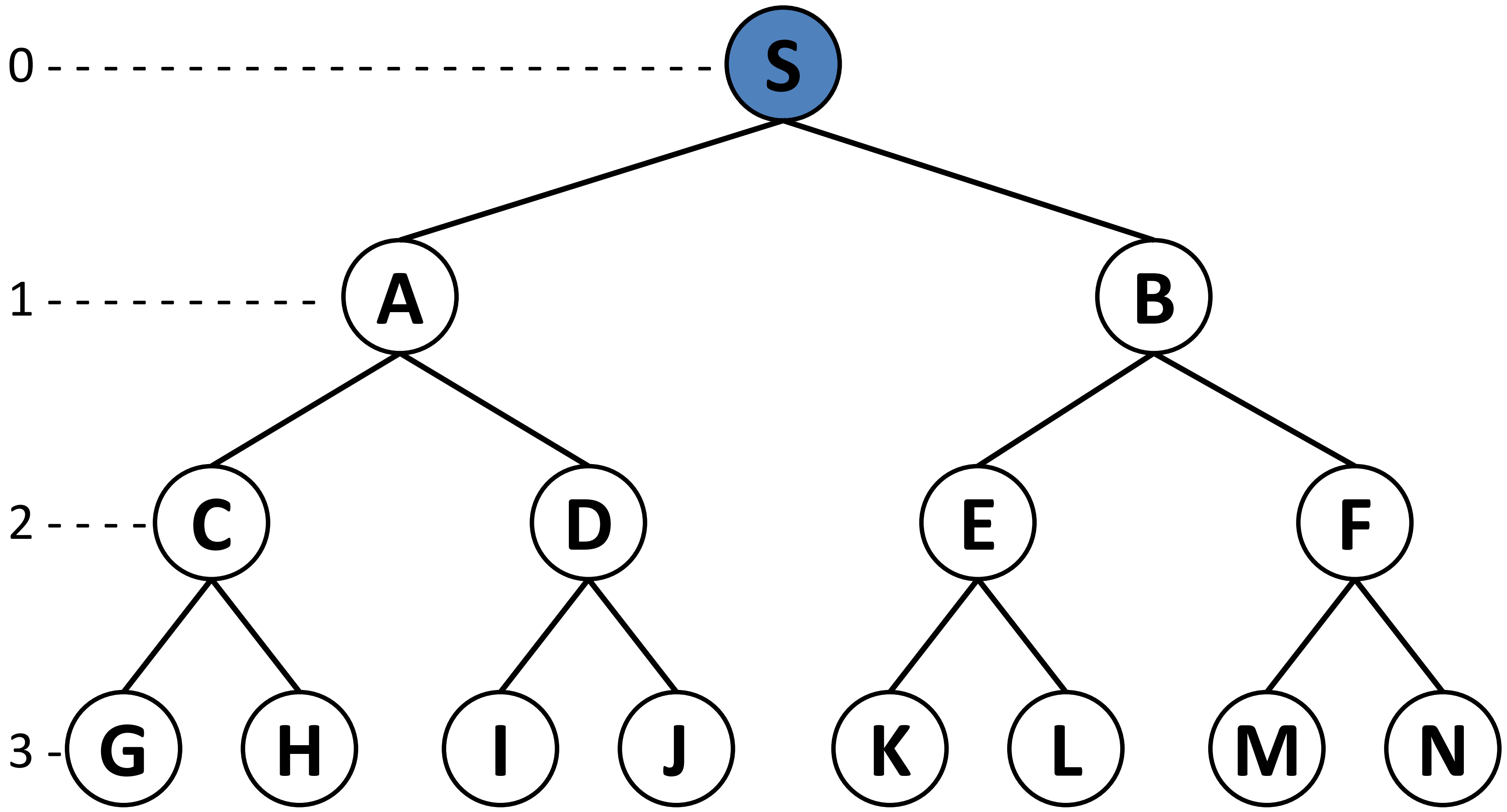
Graph and Tree Traversal: BFS



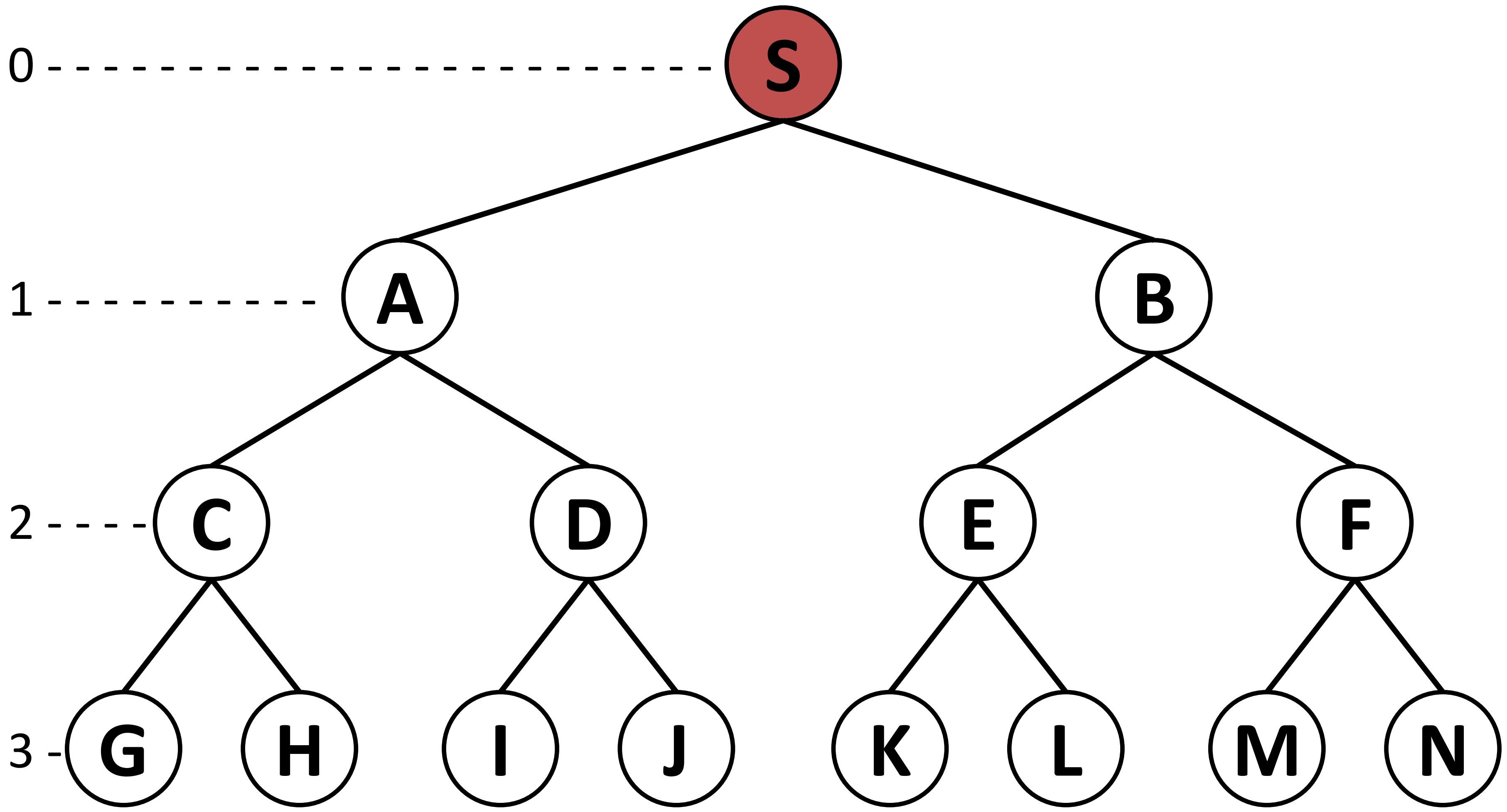
Graph and Tree Traversal: BFS



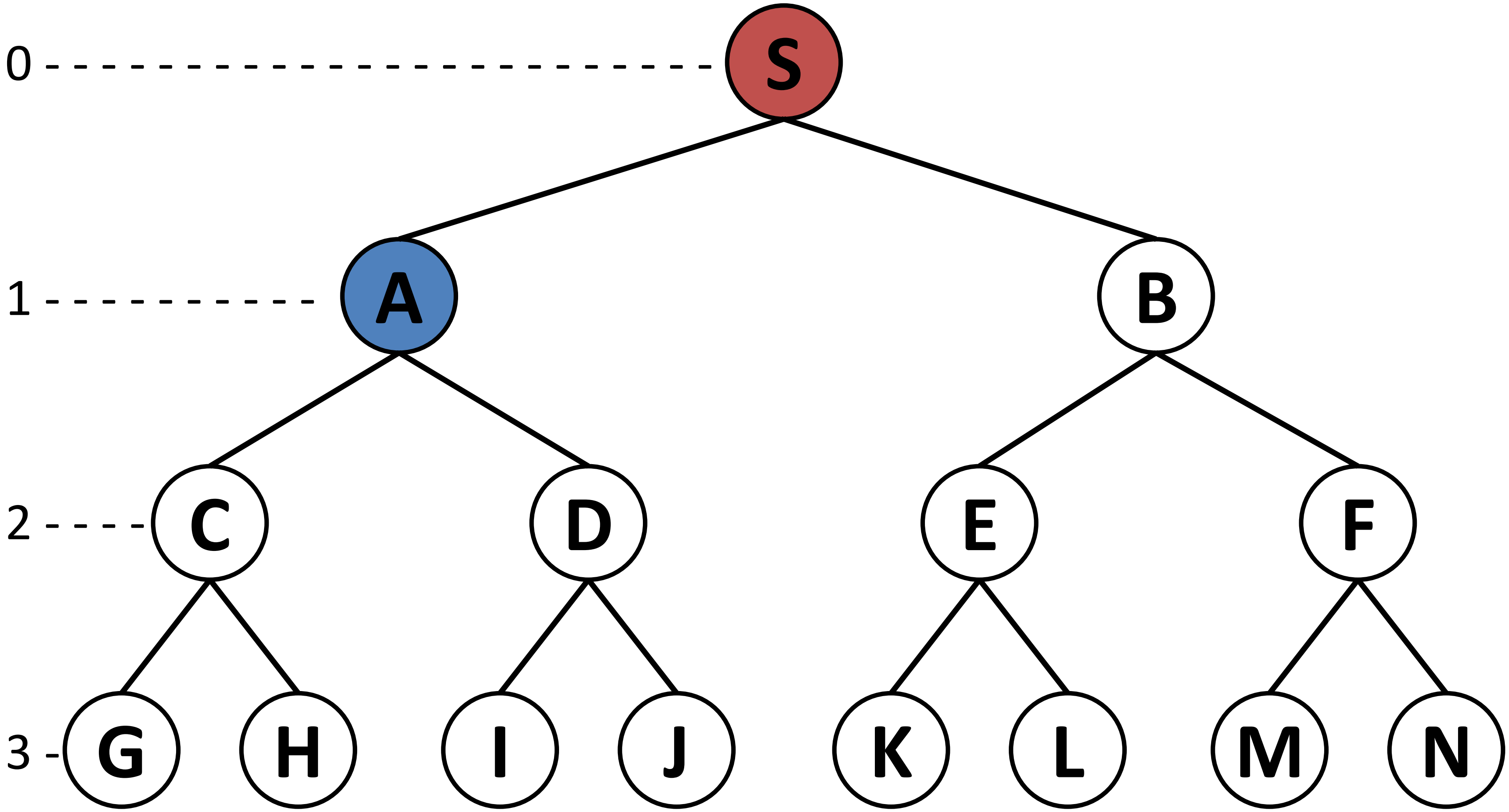
Graph and Tree Traversal: BFS



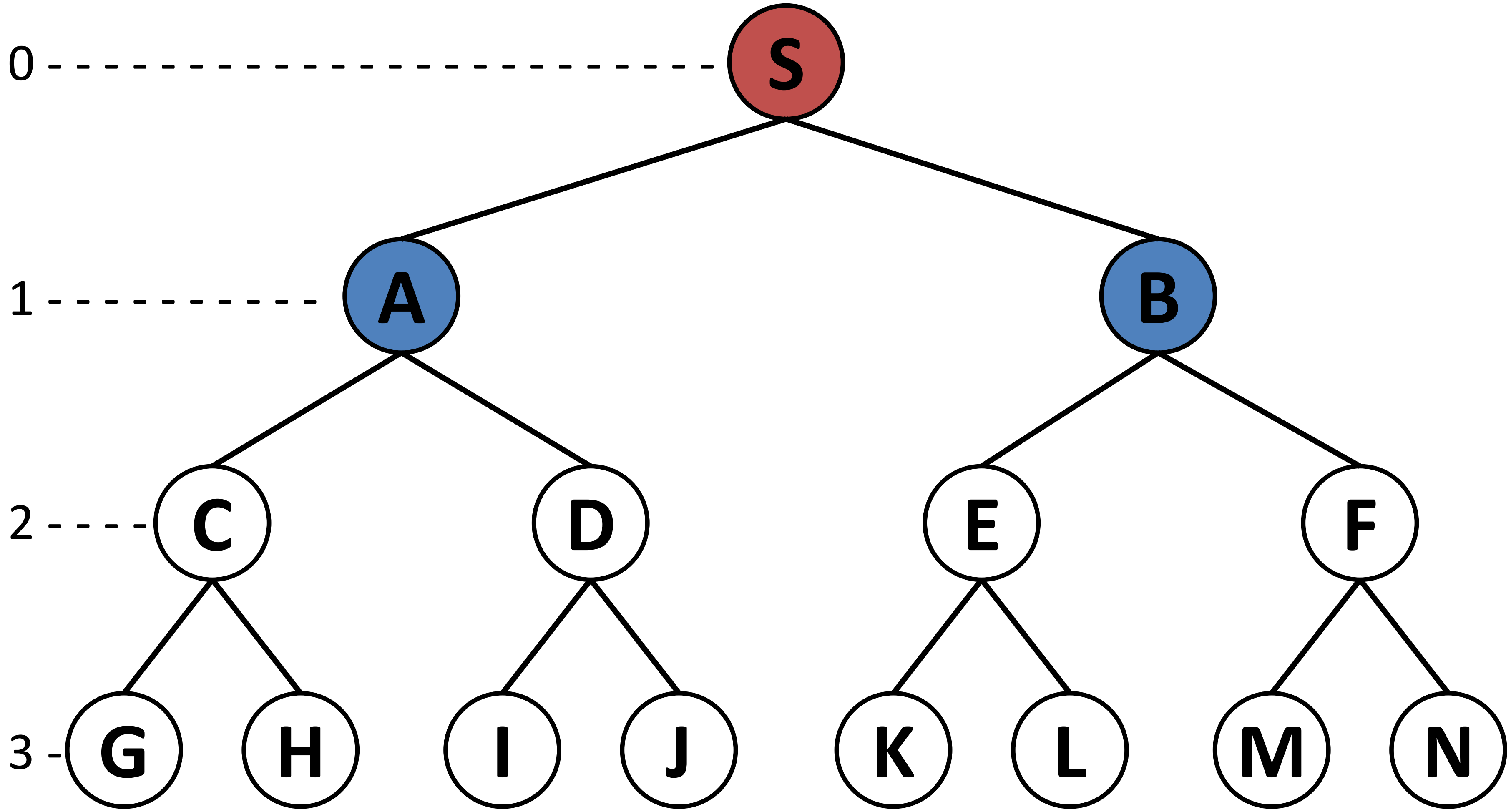
Graph and Tree Traversal: BFS



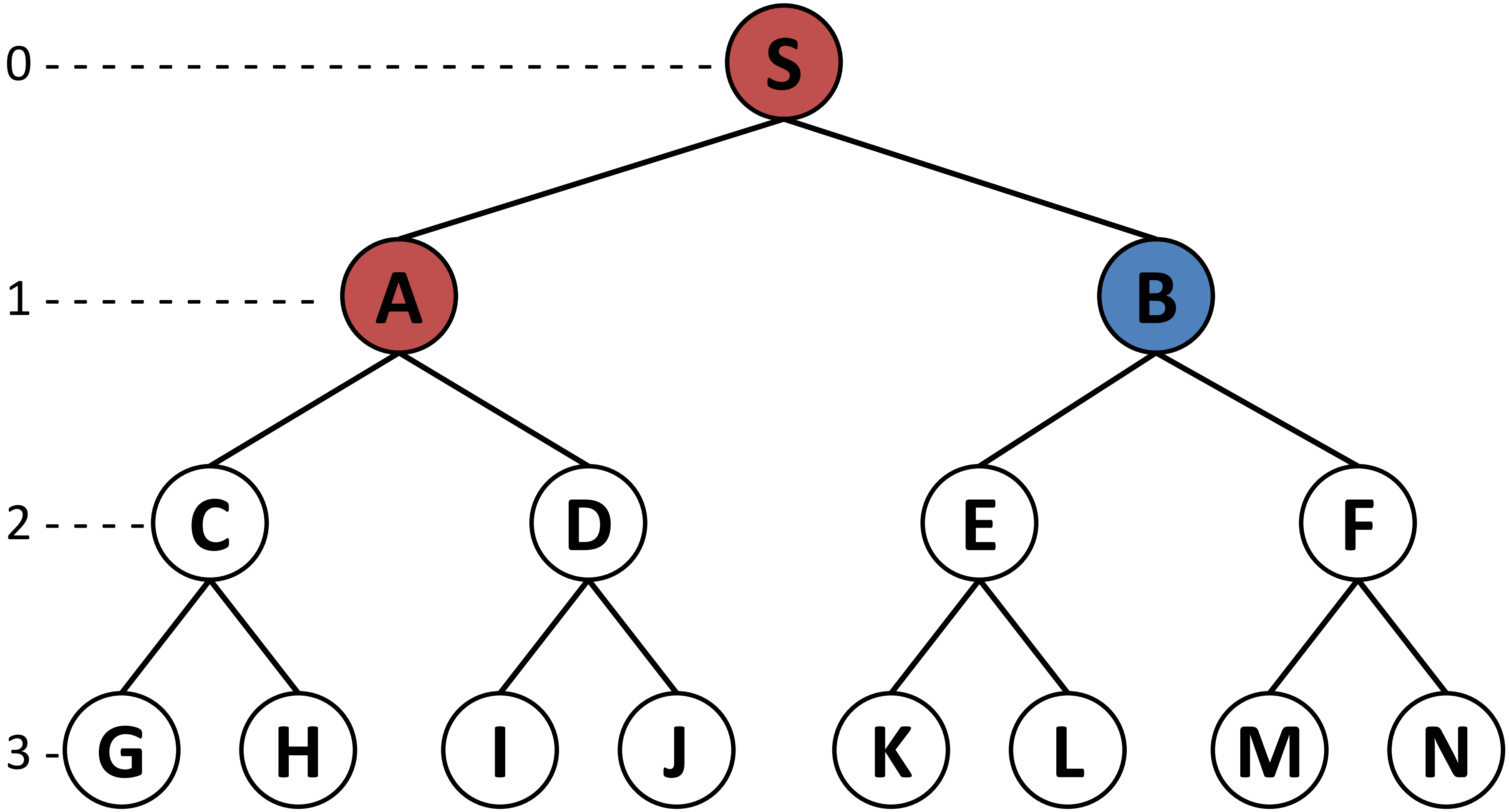
Graph and Tree Traversal: BFS



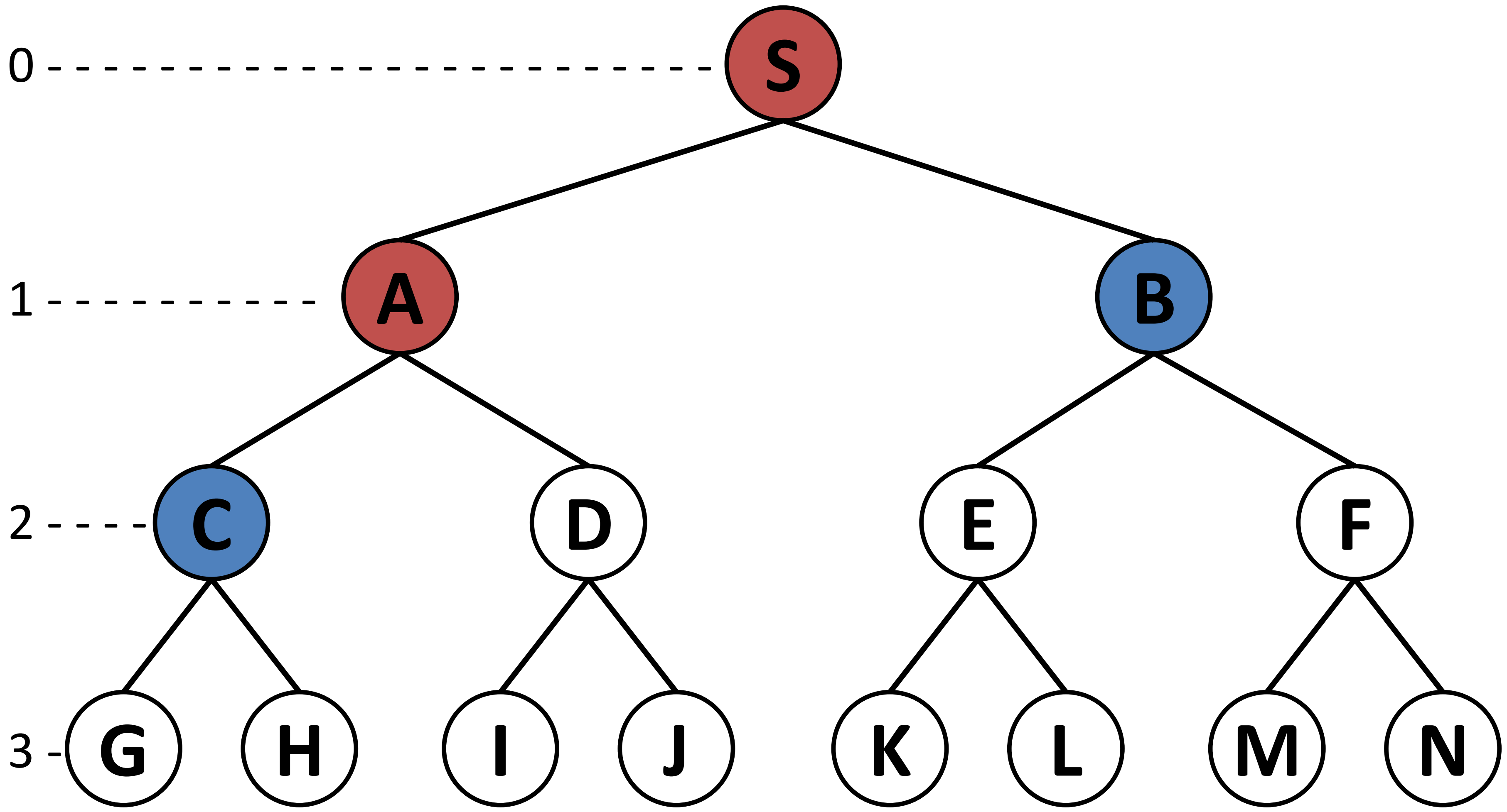
Graph and Tree Traversal: BFS



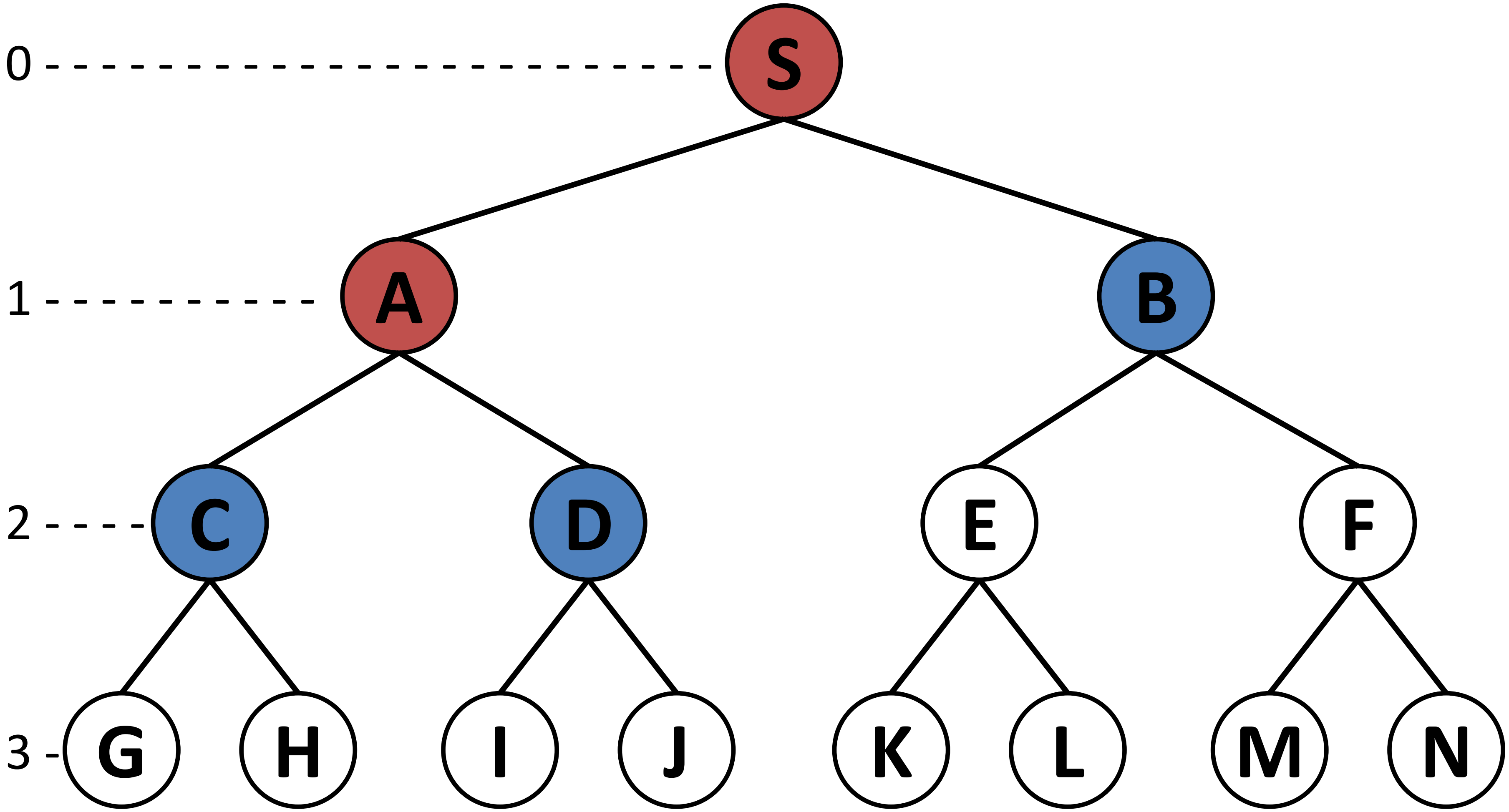
Graph and Tree Traversal: BFS



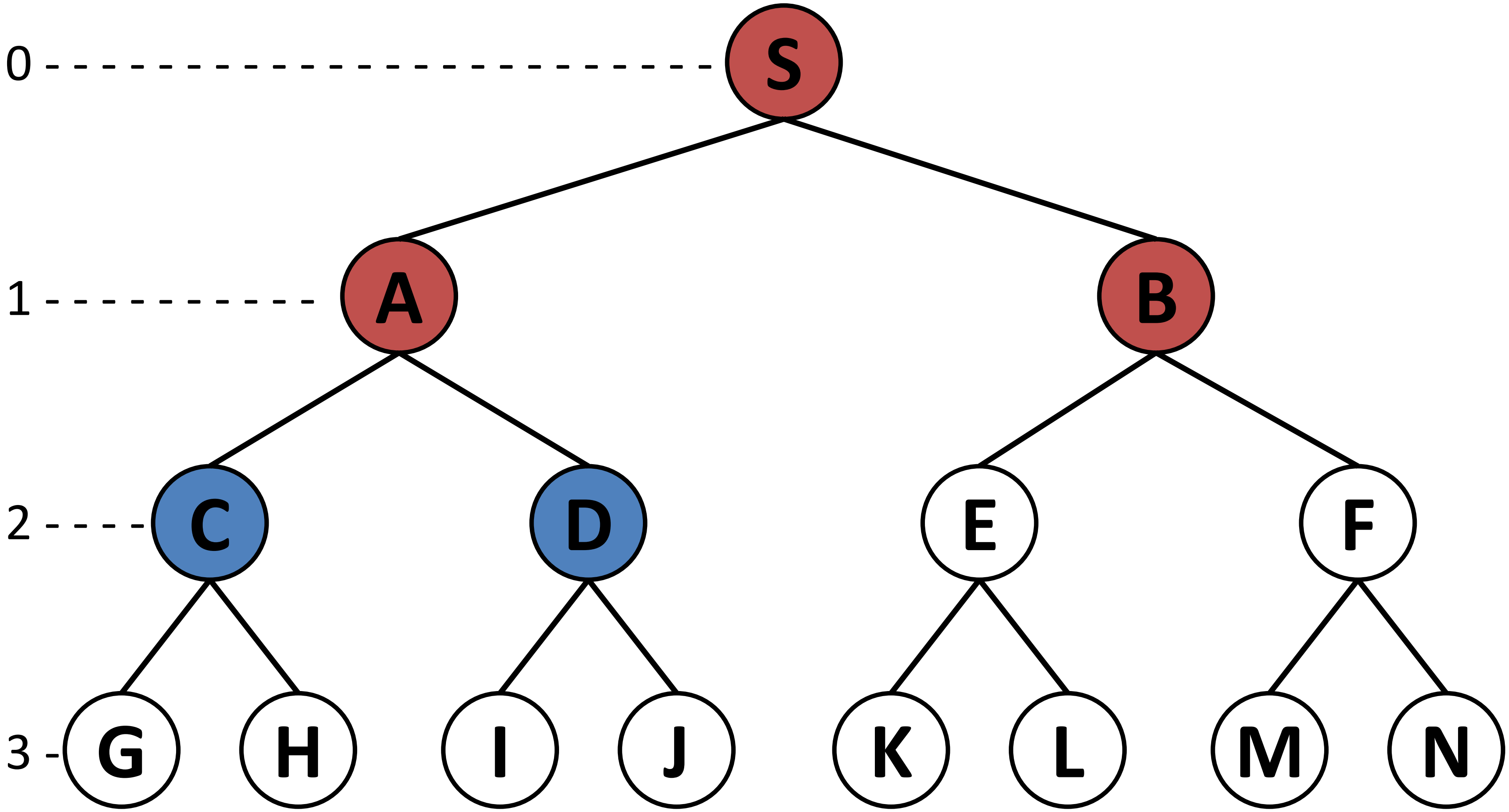
Graph and Tree Traversal: BFS



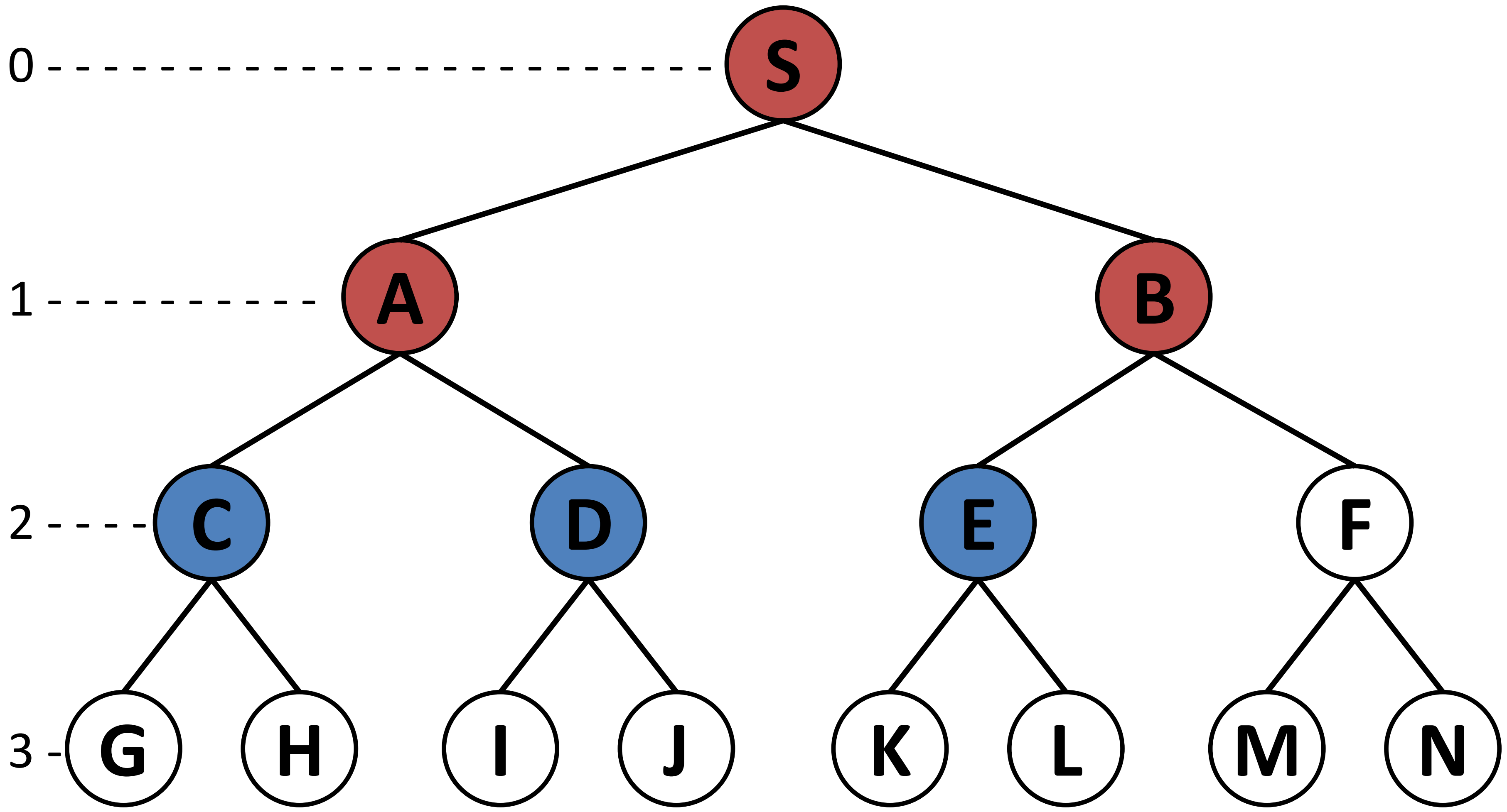
Graph and Tree Traversal: BFS



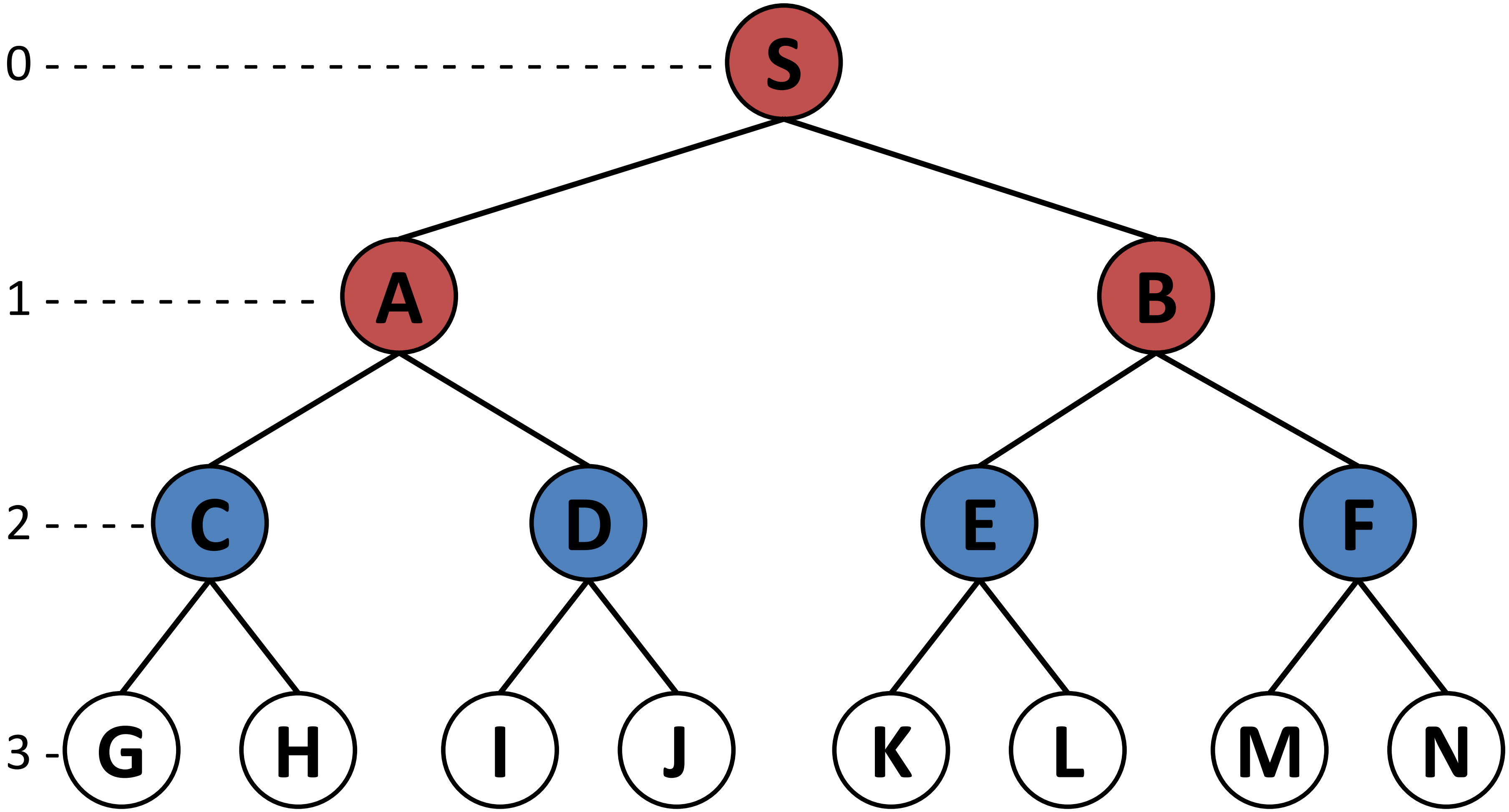
Graph and Tree Traversal: BFS



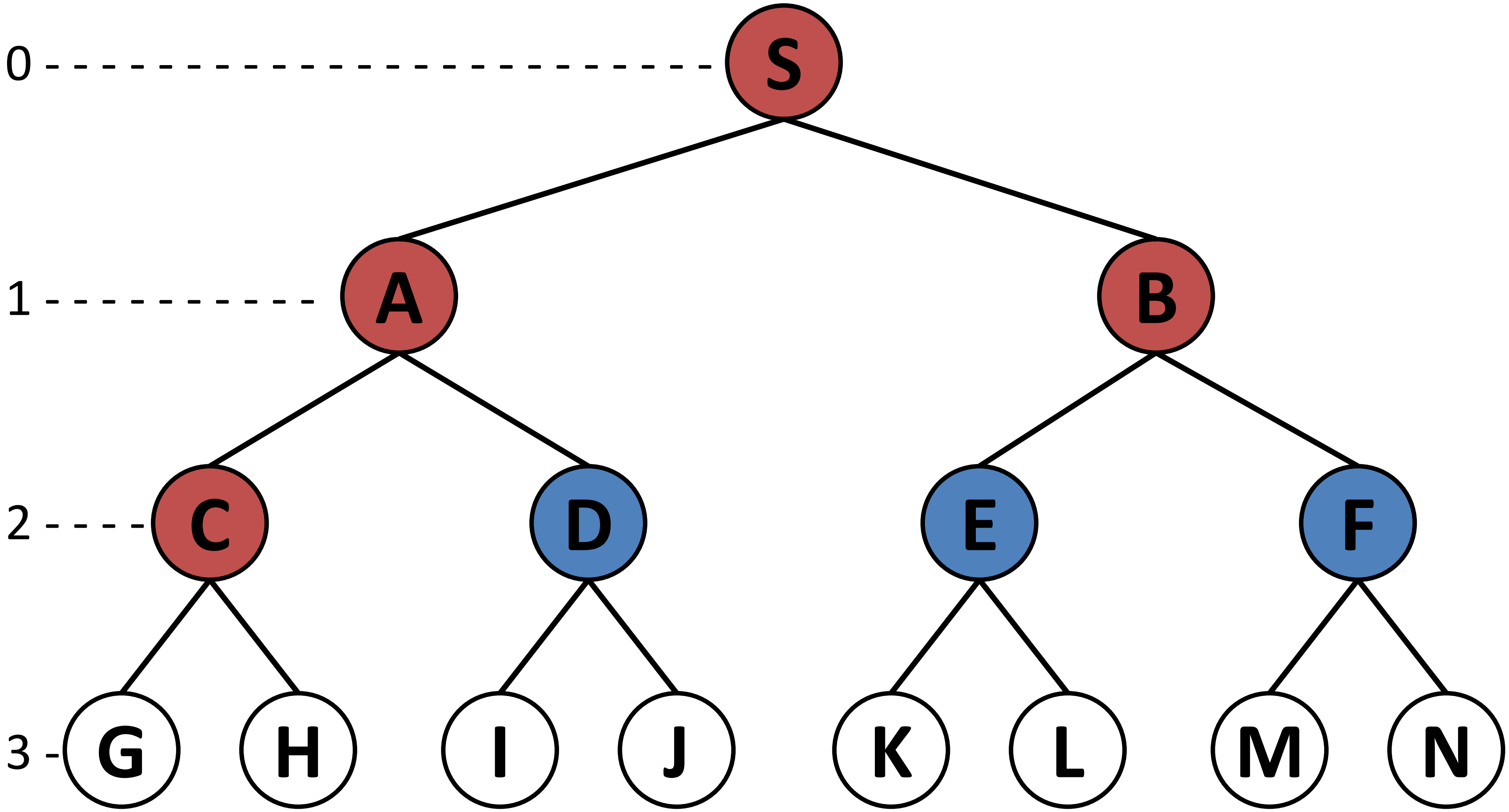
Graph and Tree Traversal: BFS



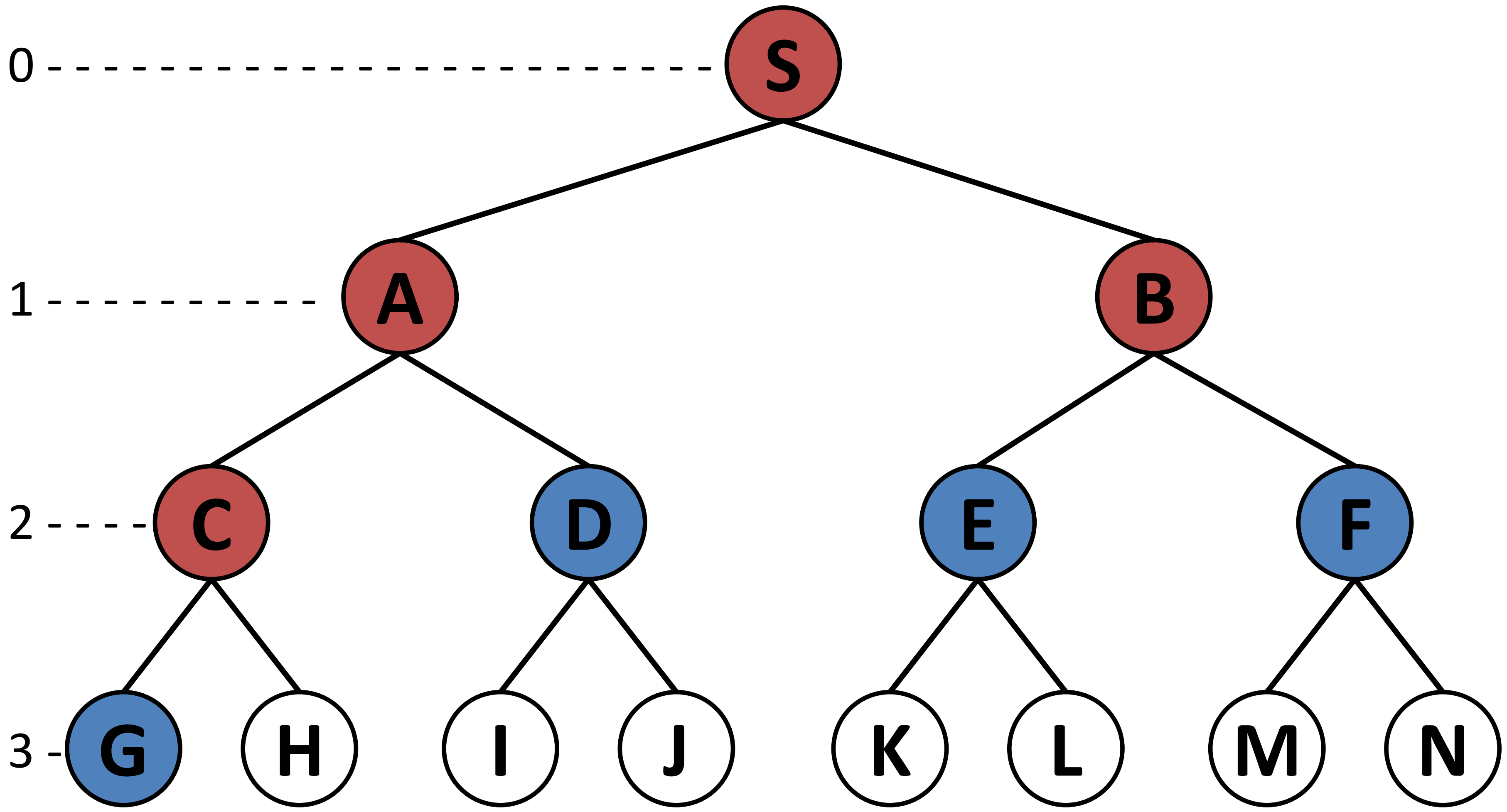
Graph and Tree Traversal: BFS



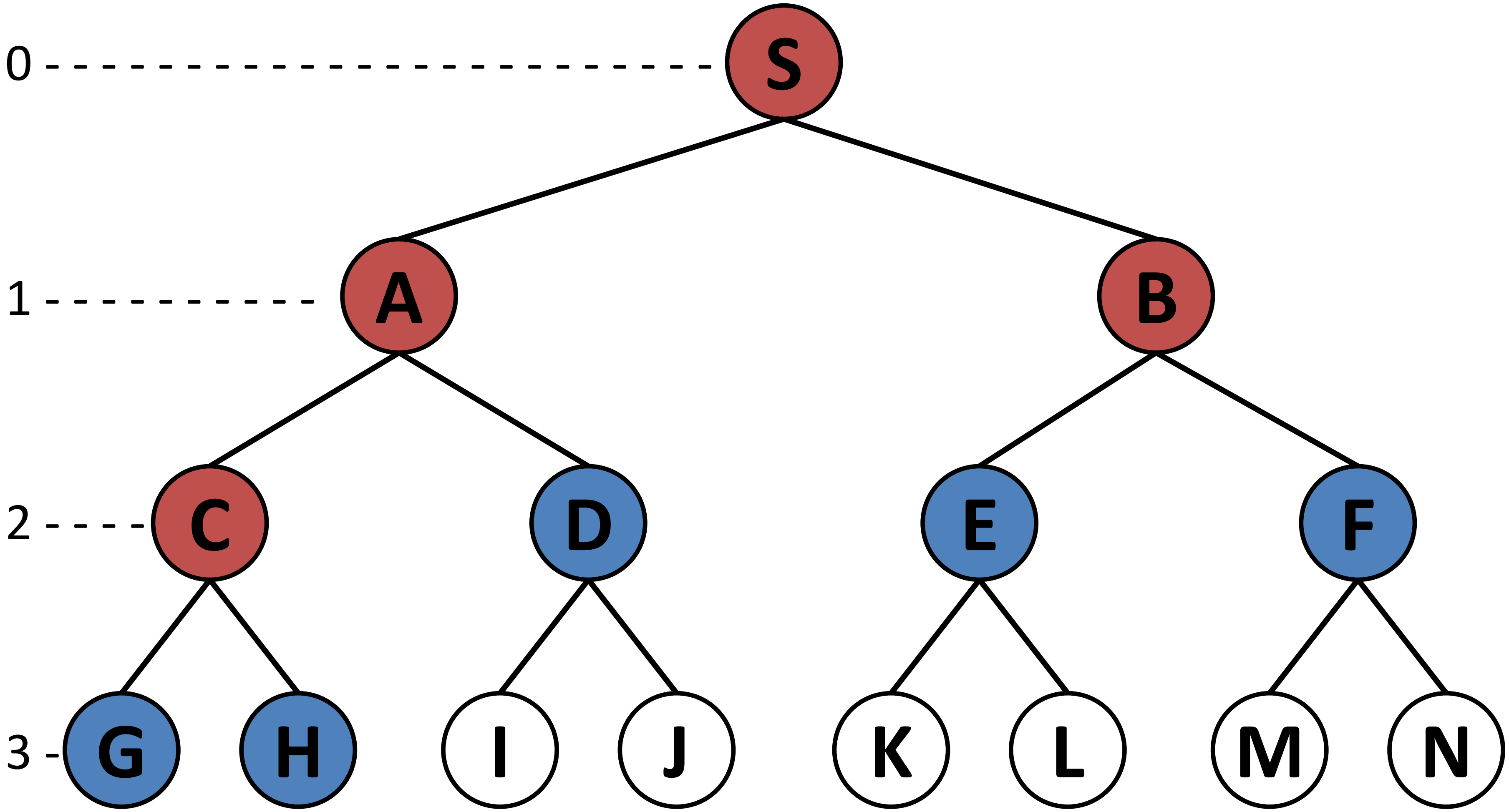
Graph and Tree Traversal: BFS



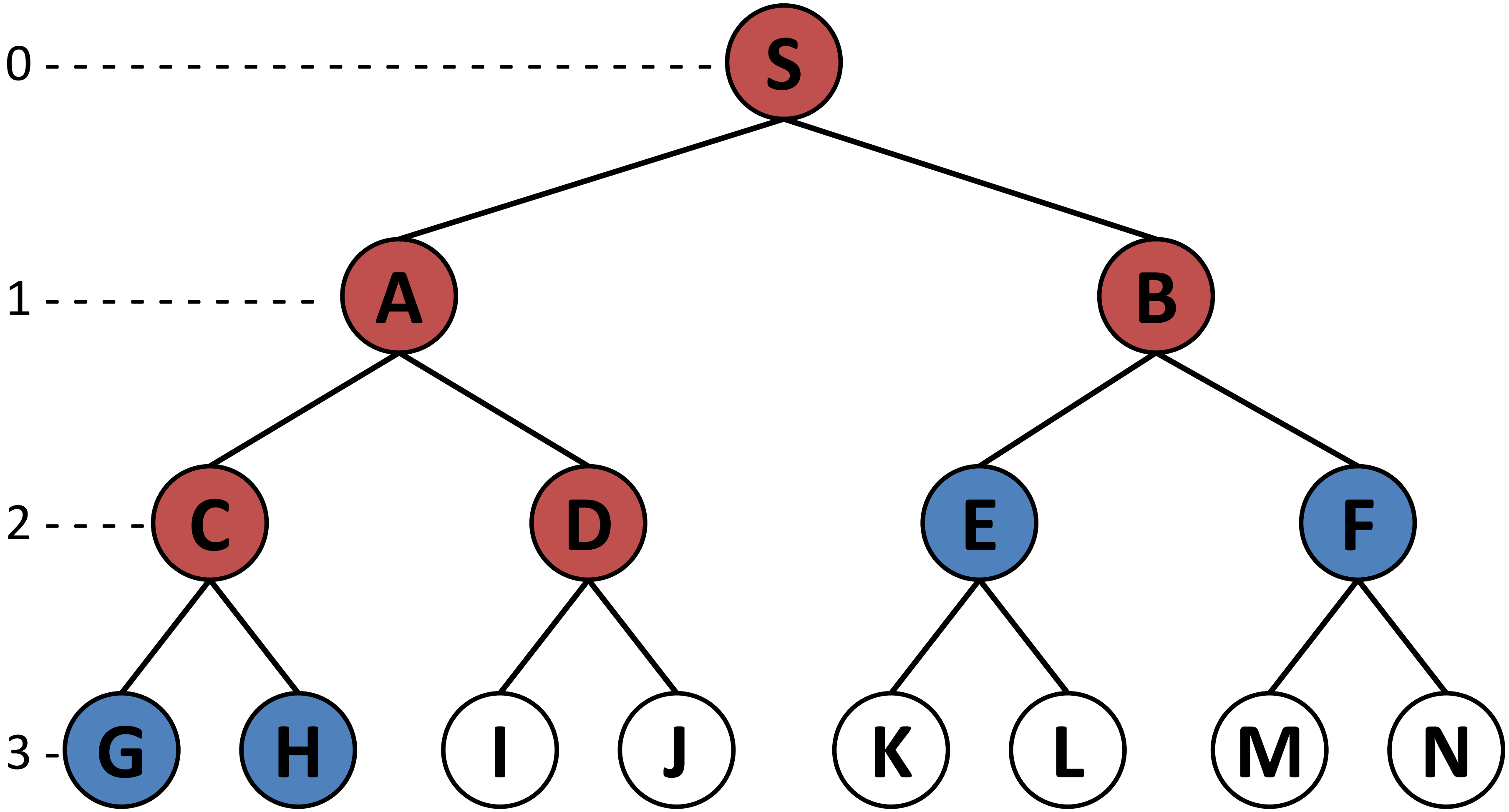
Graph and Tree Traversal: BFS



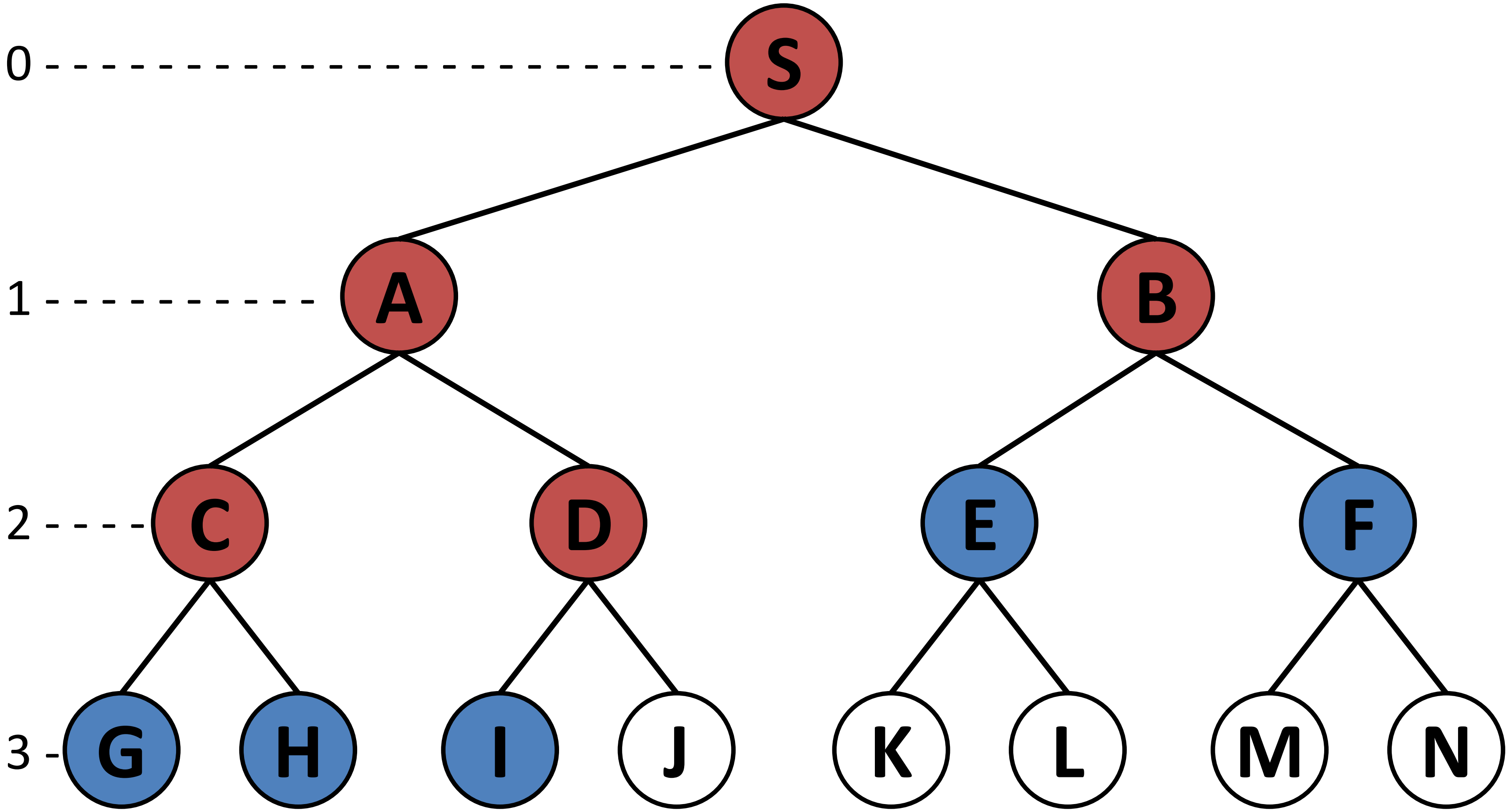
Graph and Tree Traversal: BFS



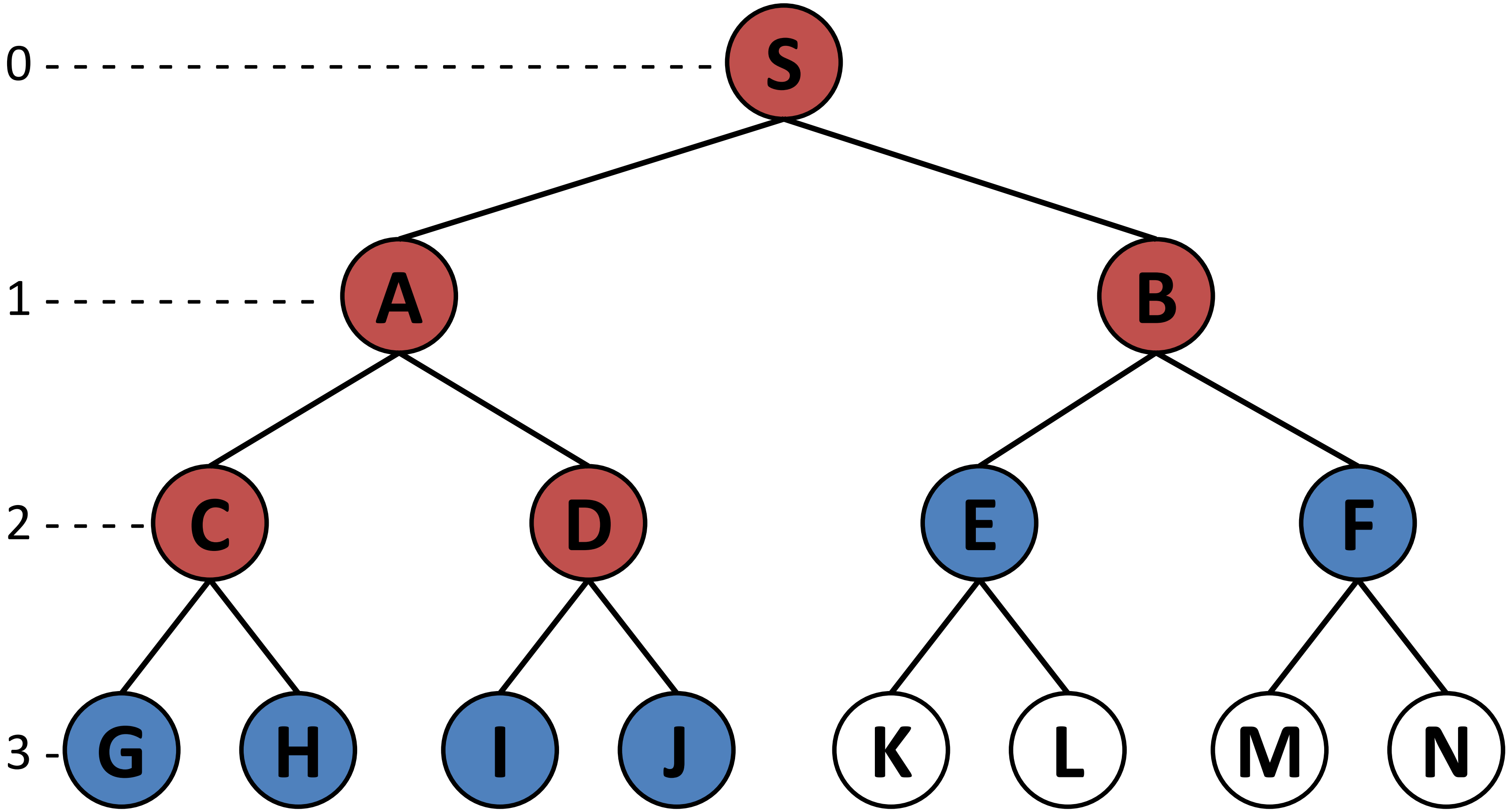
Graph and Tree Traversal: BFS



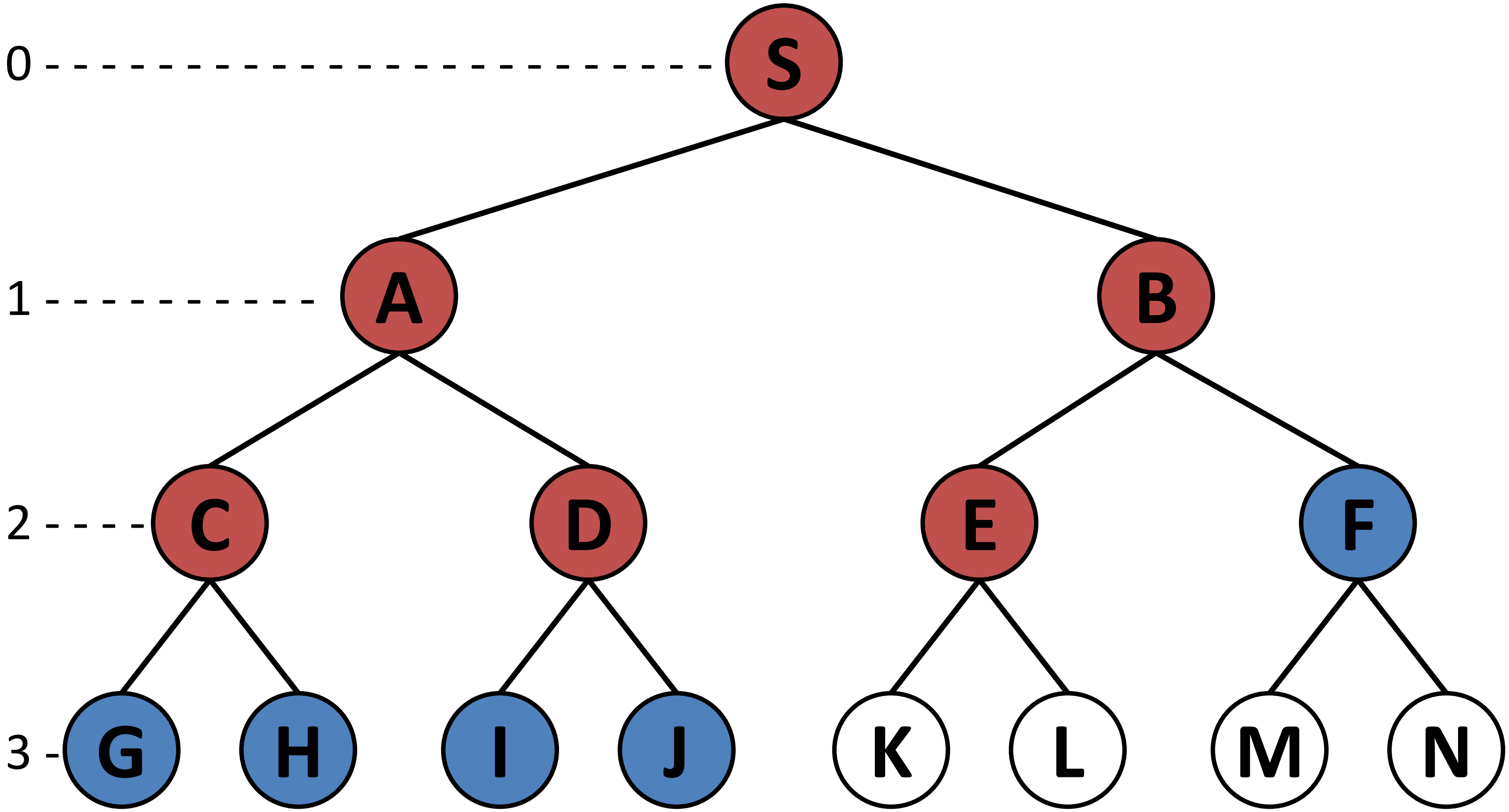
Graph and Tree Traversal: BFS



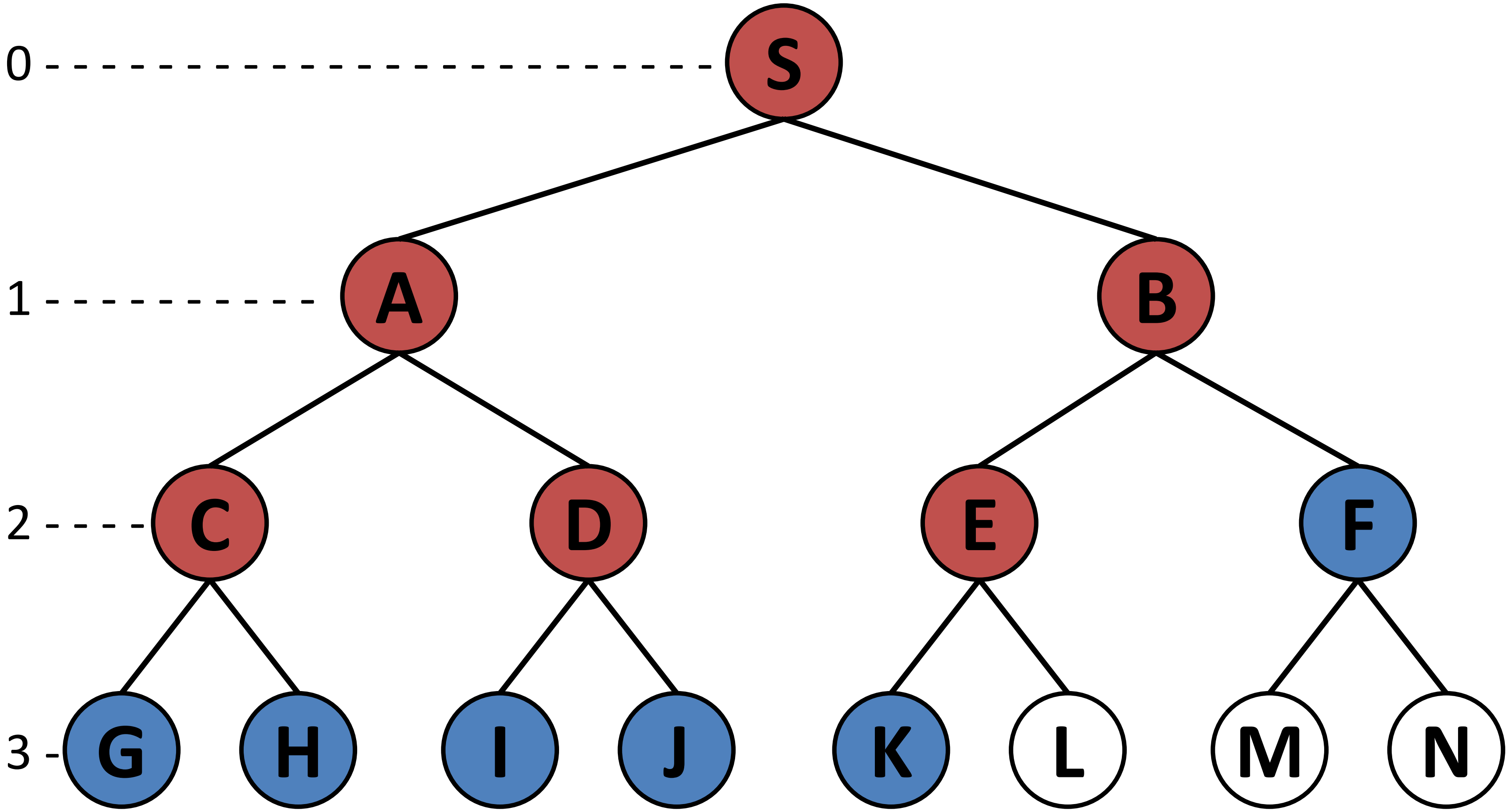
Graph and Tree Traversal: BFS



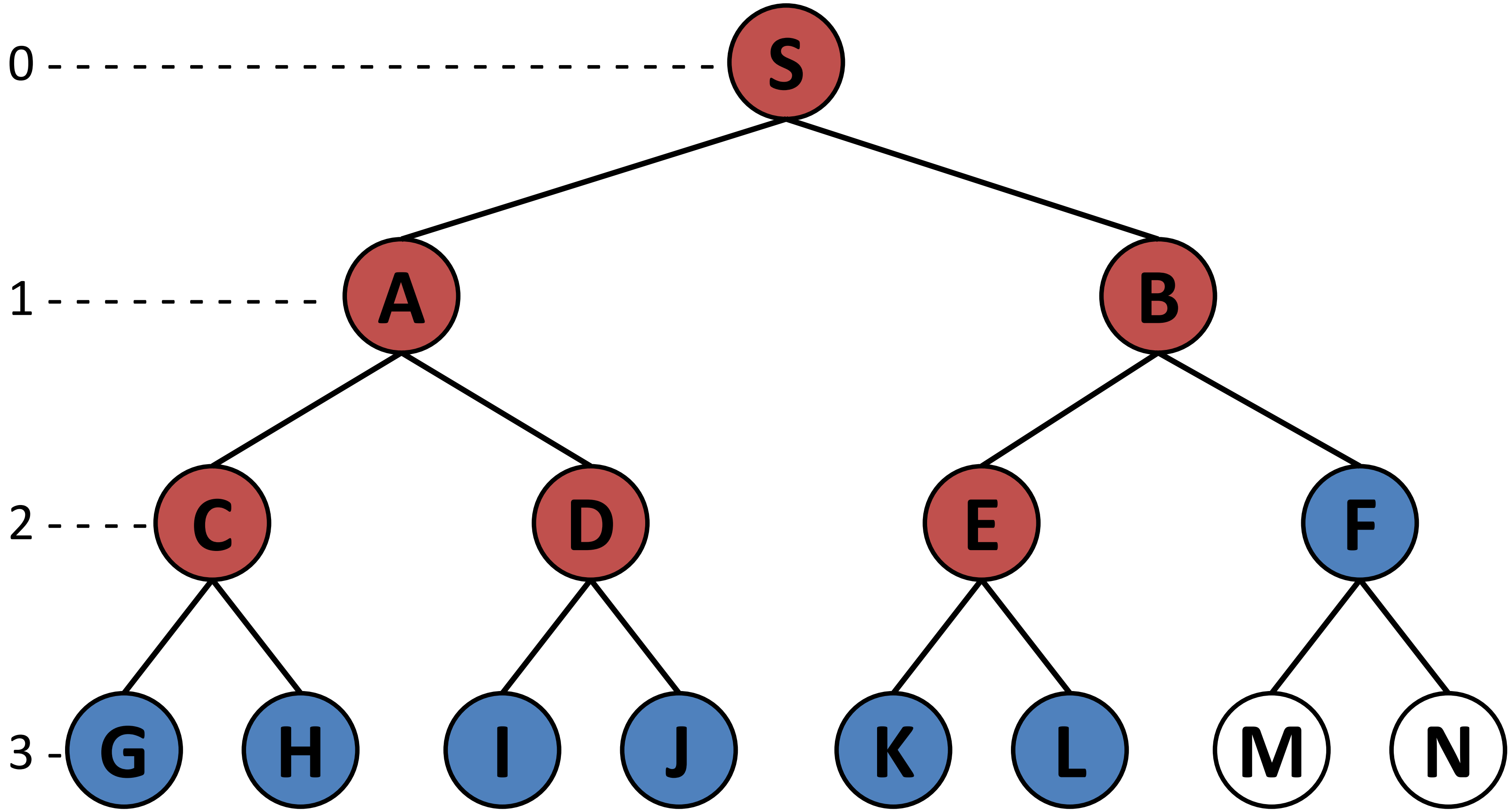
Graph and Tree Traversal: BFS



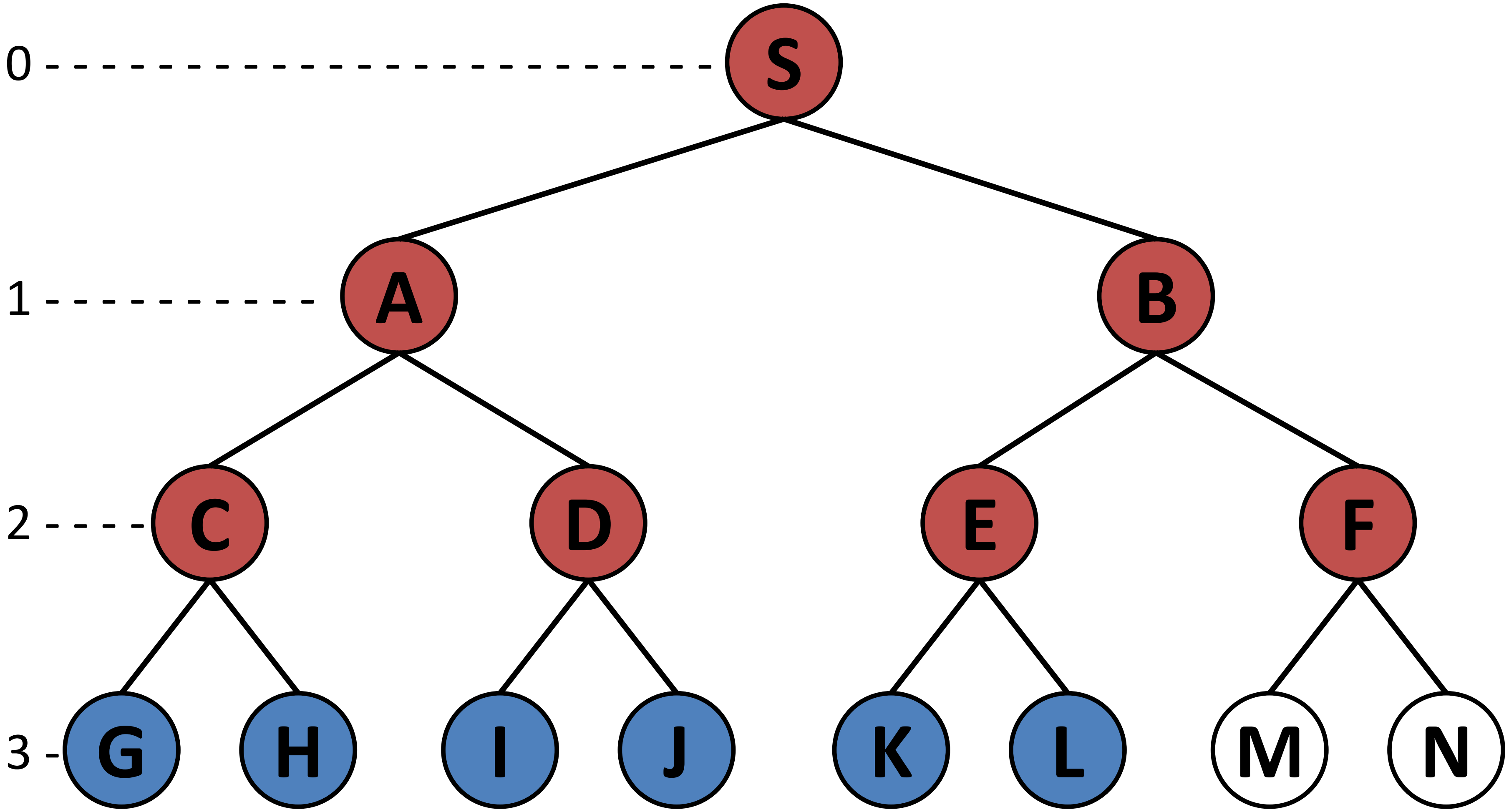
Graph and Tree Traversal: BFS



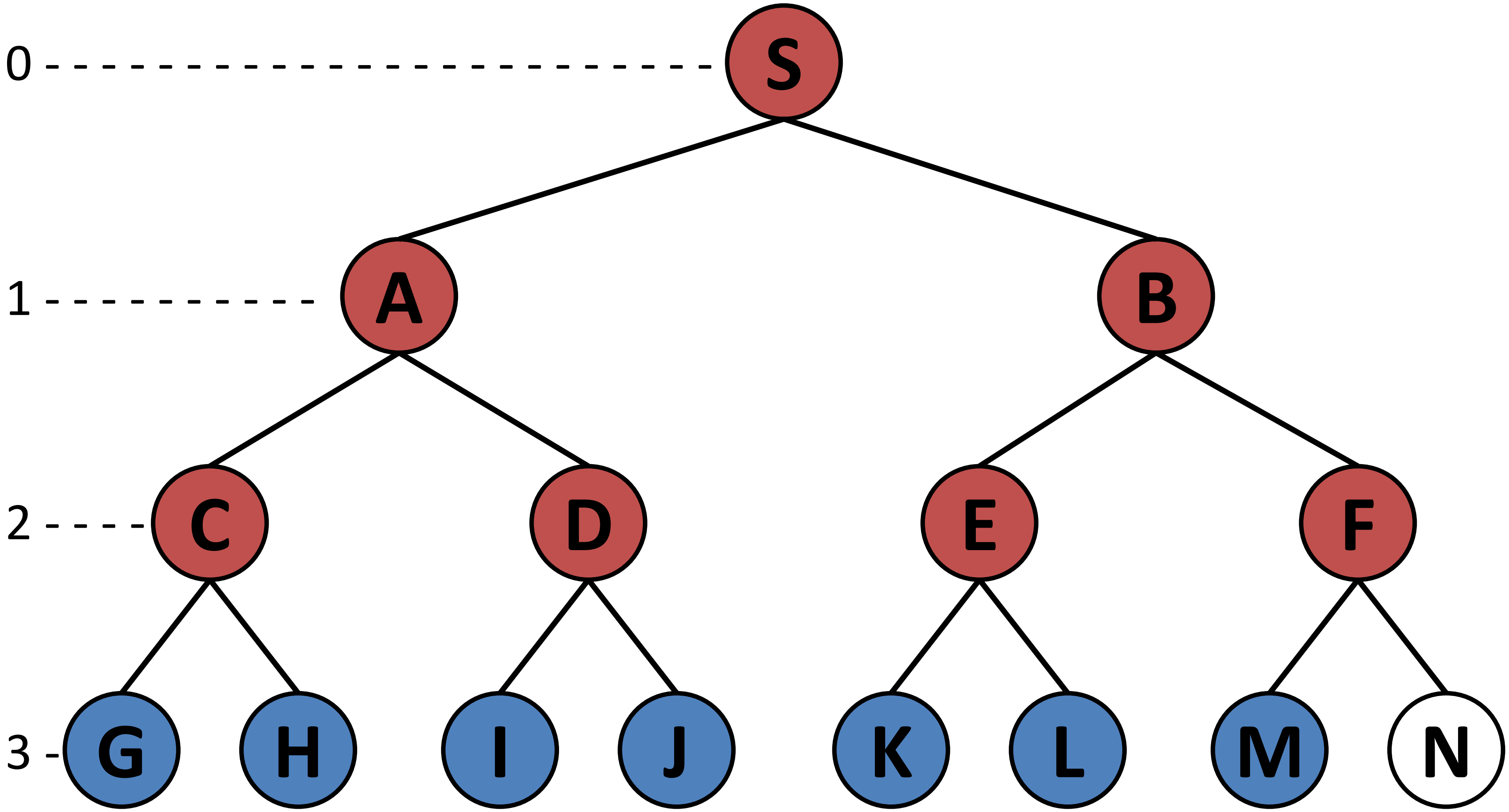
Graph and Tree Traversal: BFS



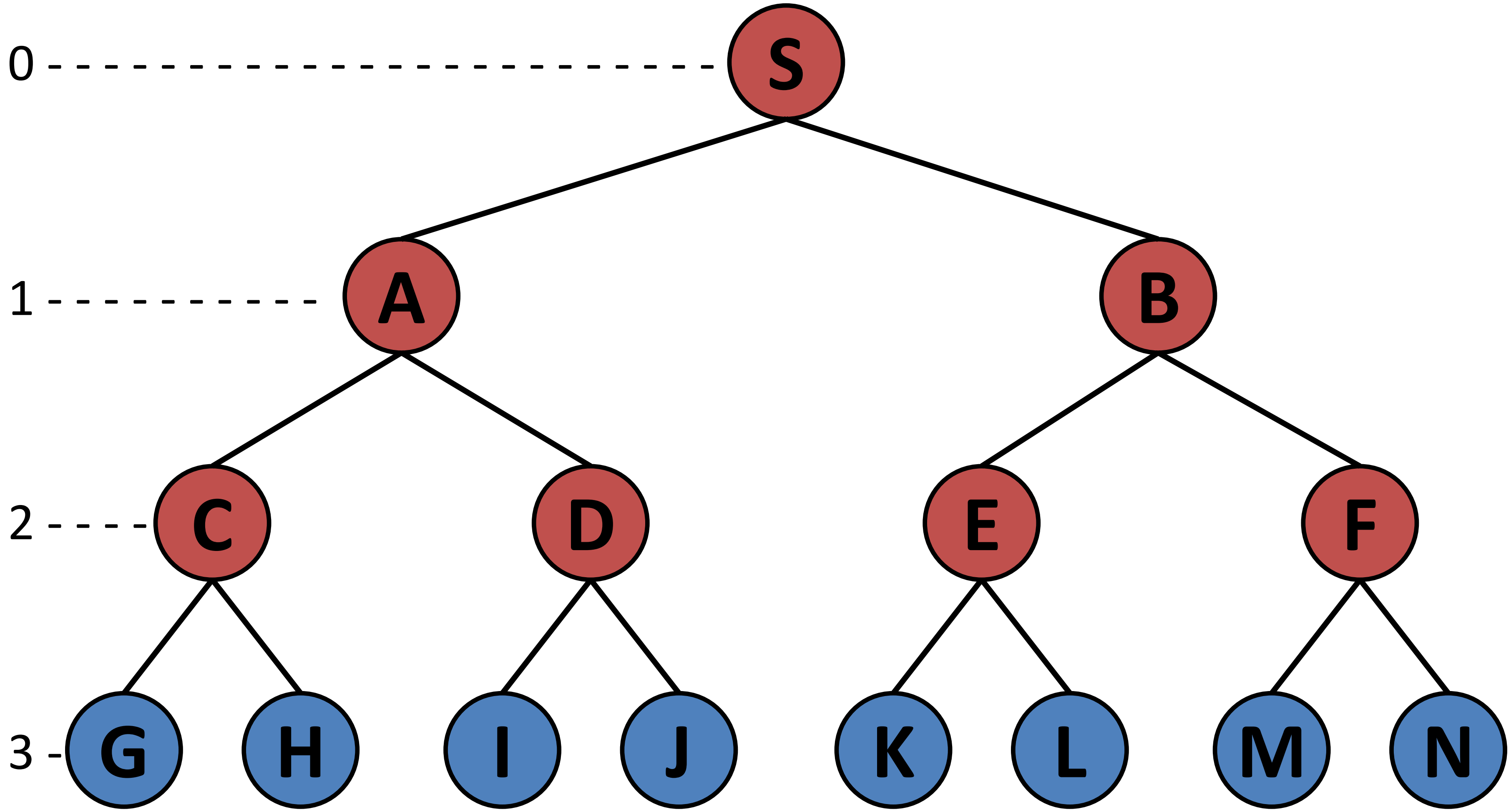
Graph and Tree Traversal: BFS



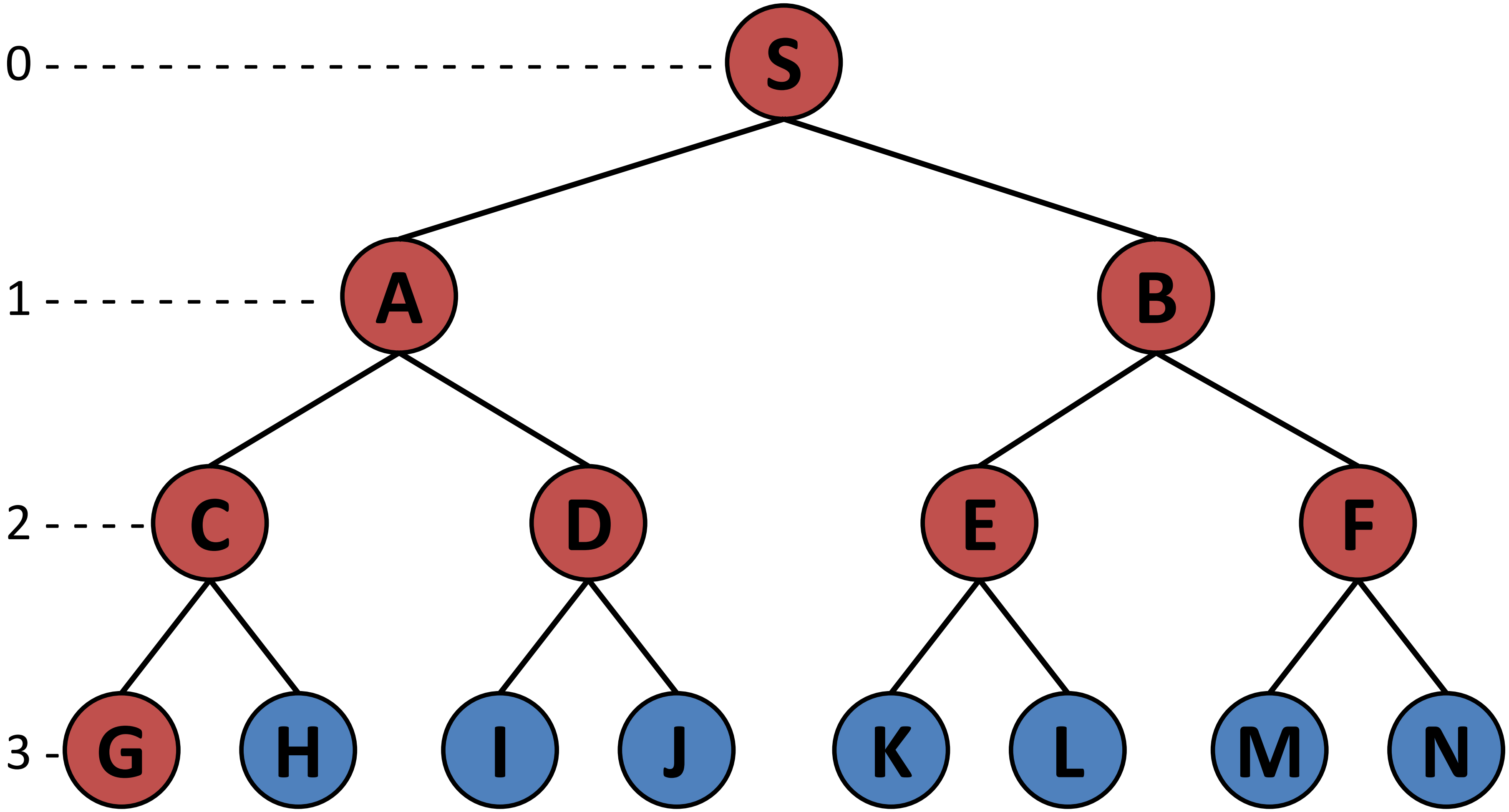
Graph and Tree Traversal: BFS



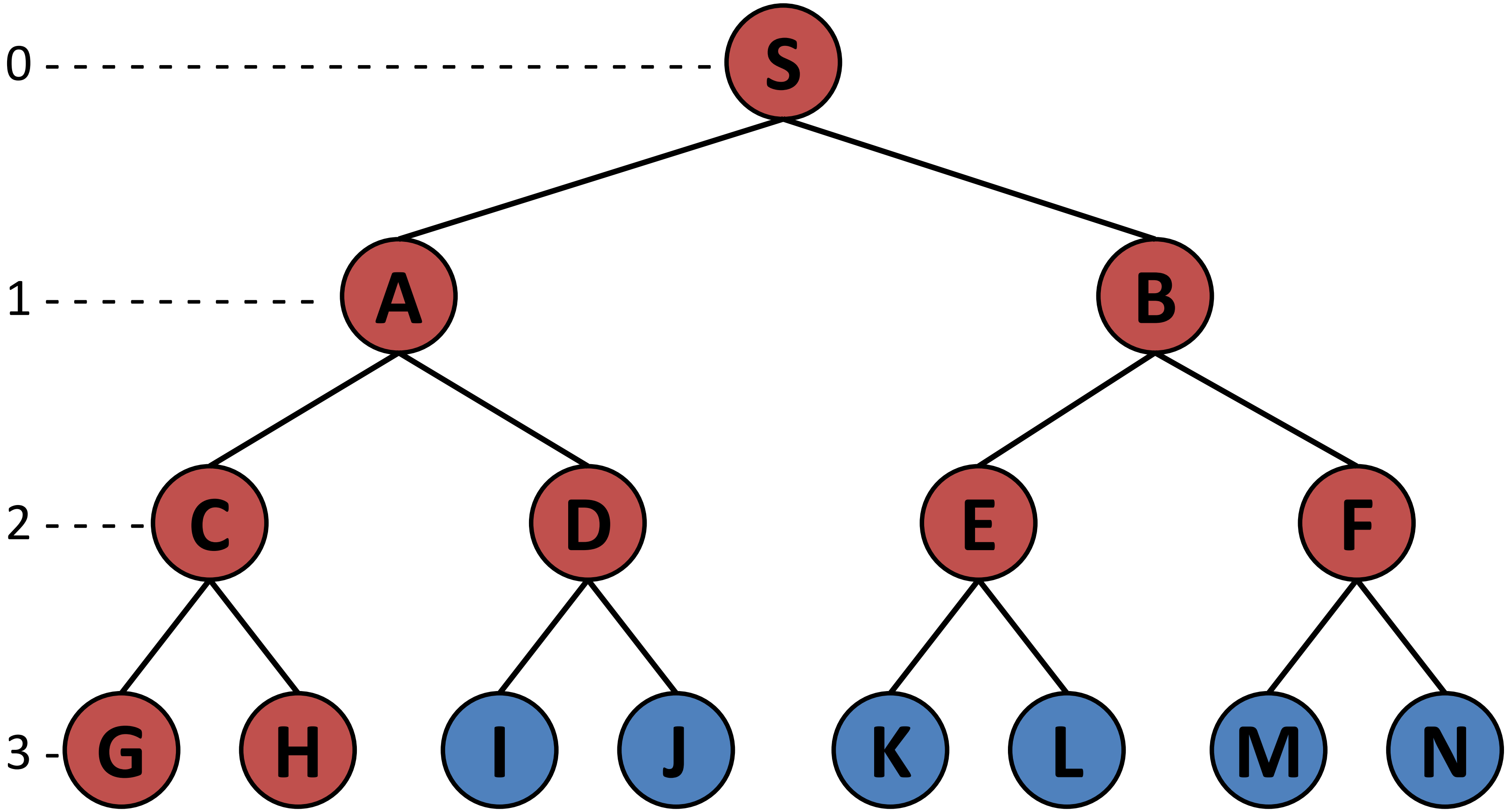
Graph and Tree Traversal: BFS



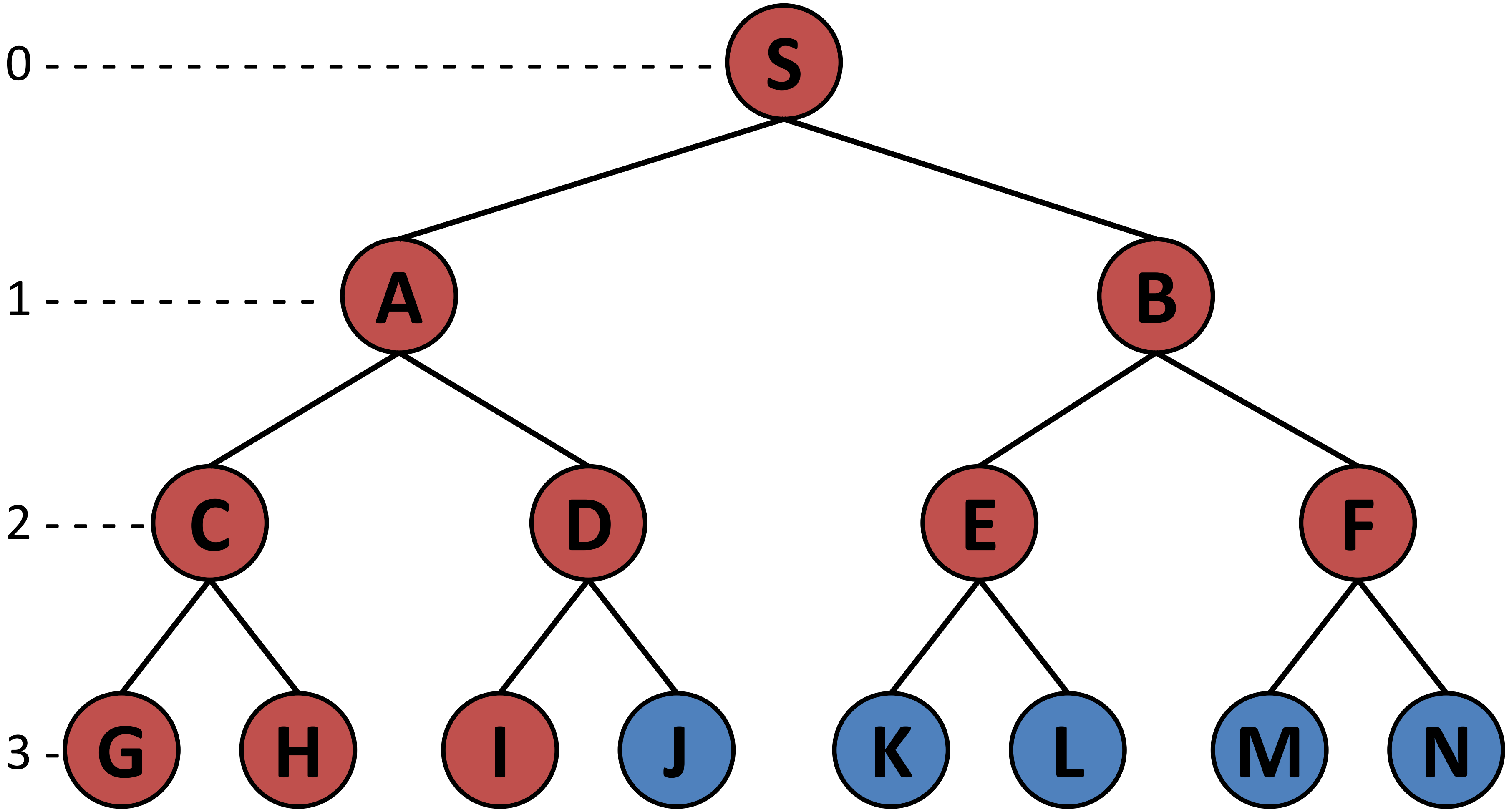
Graph and Tree Traversal: BFS



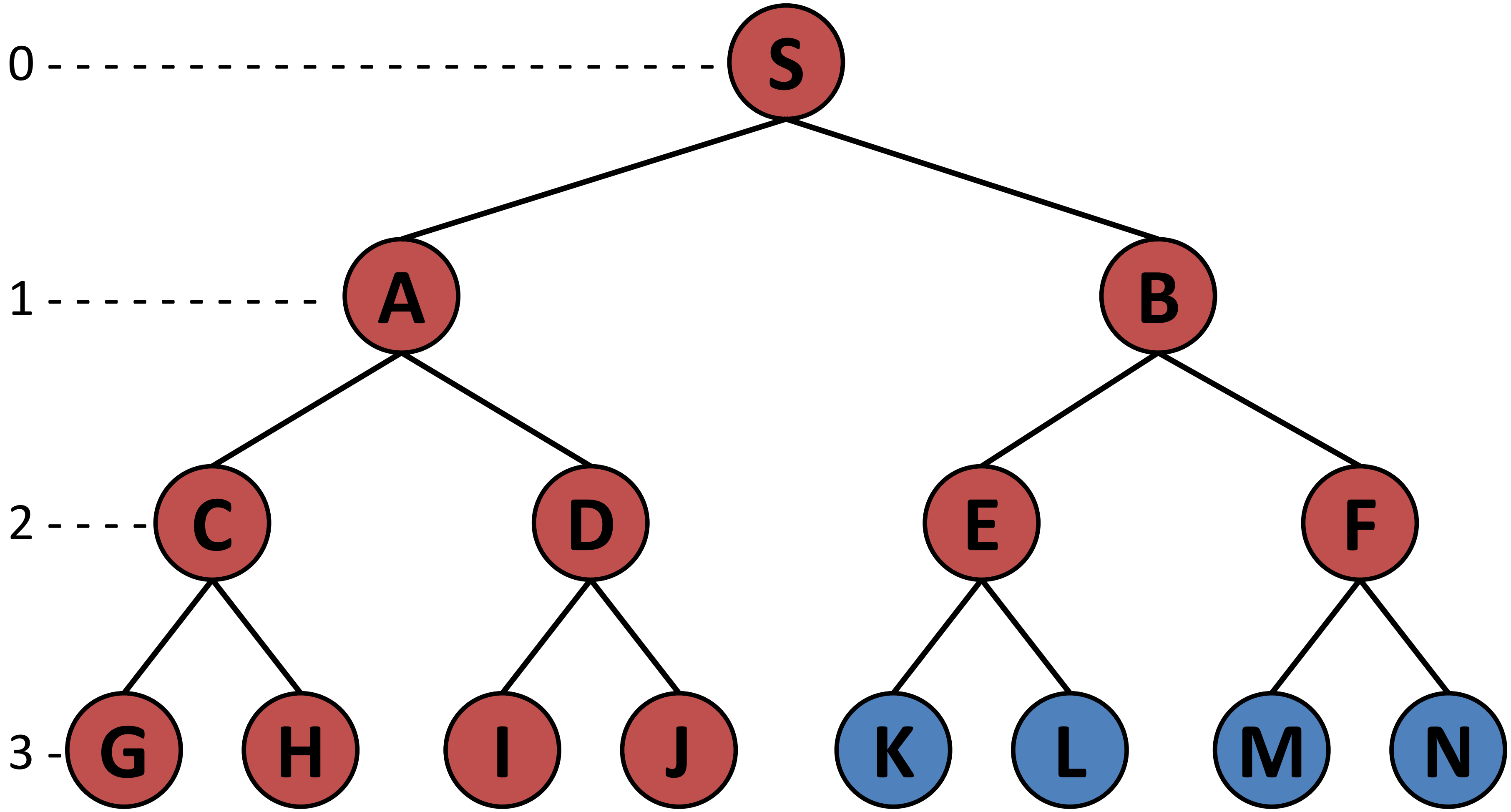
Graph and Tree Traversal: BFS



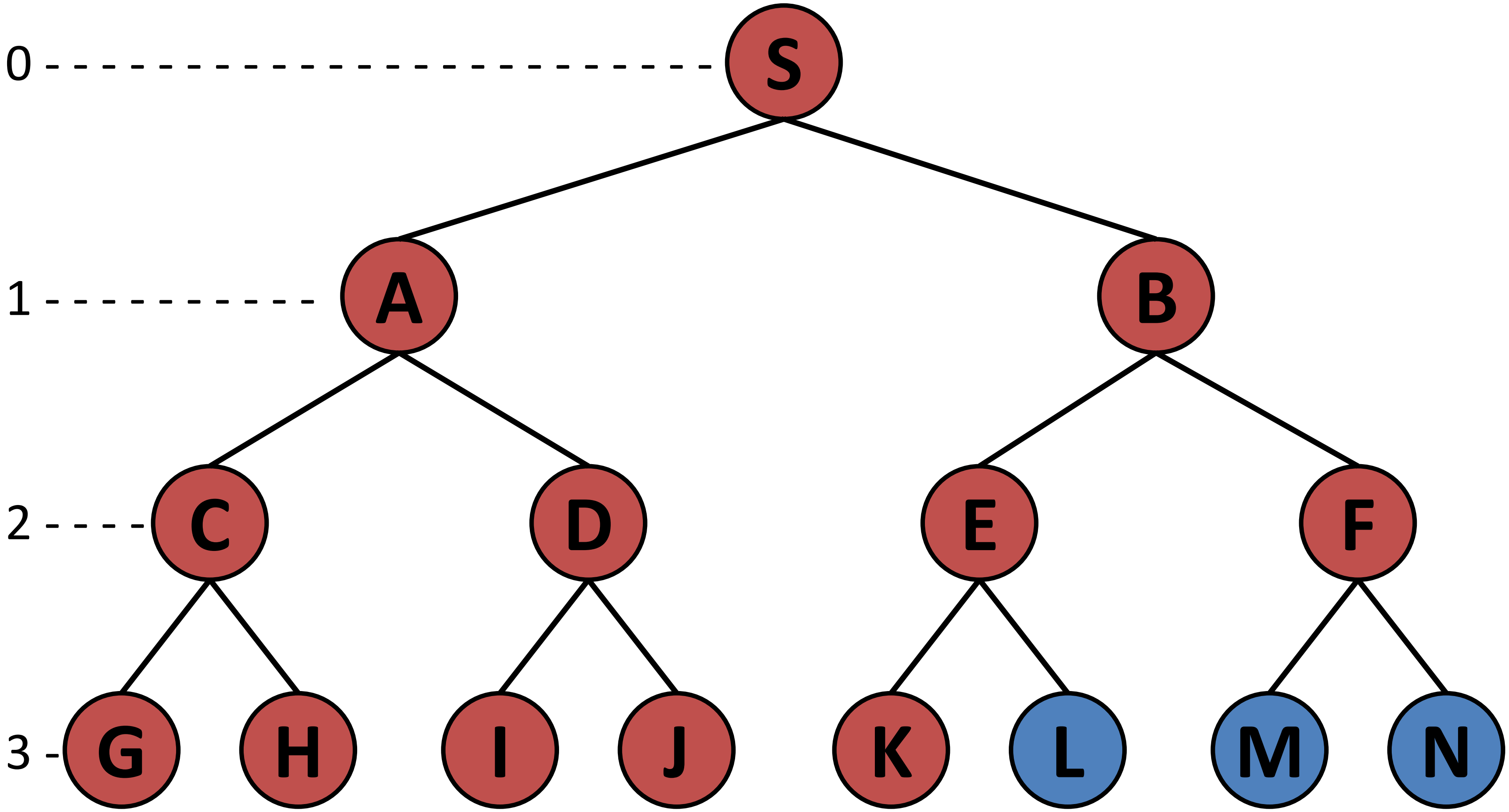
Graph and Tree Traversal: BFS



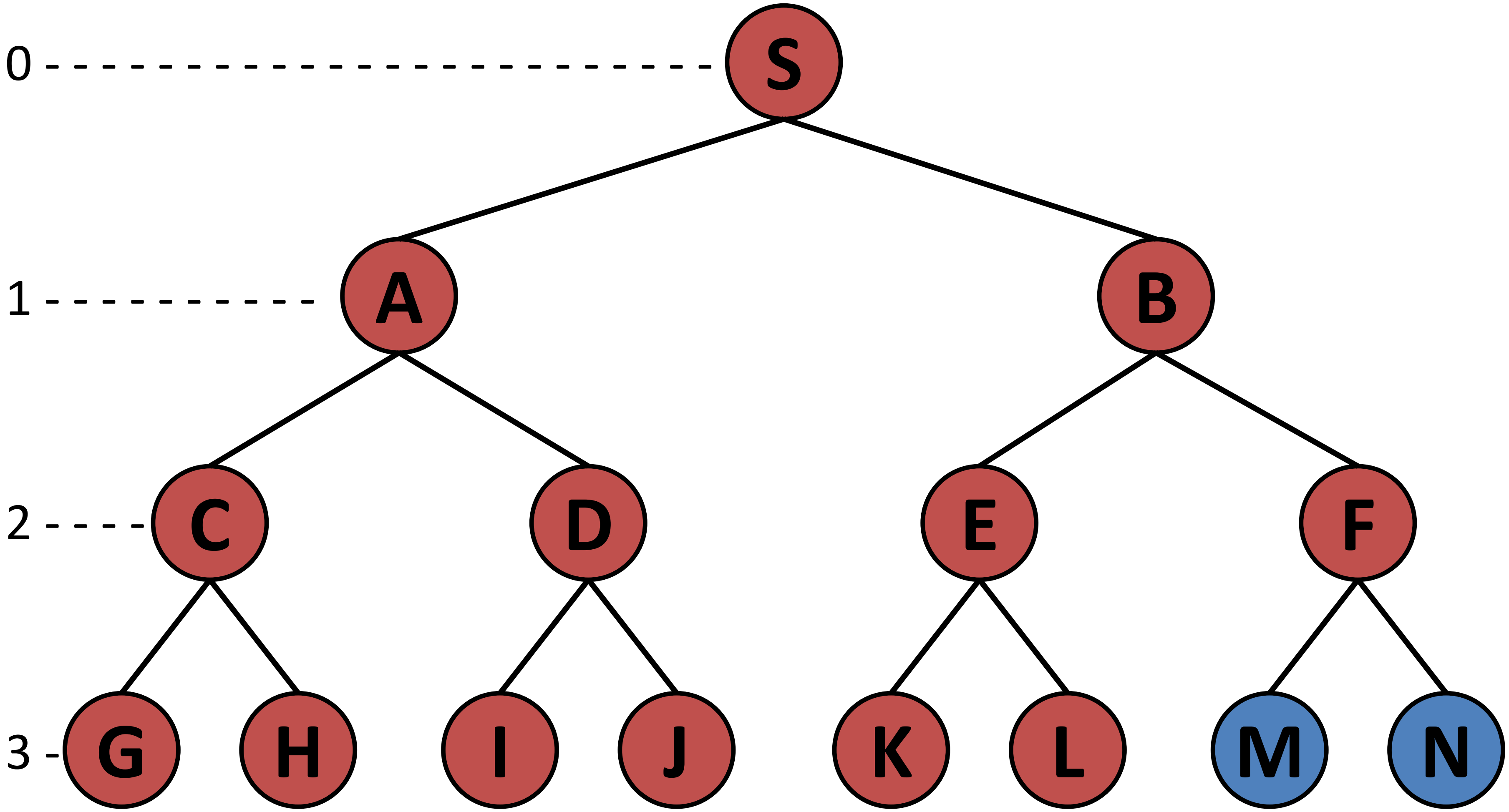
Graph and Tree Traversal: BFS



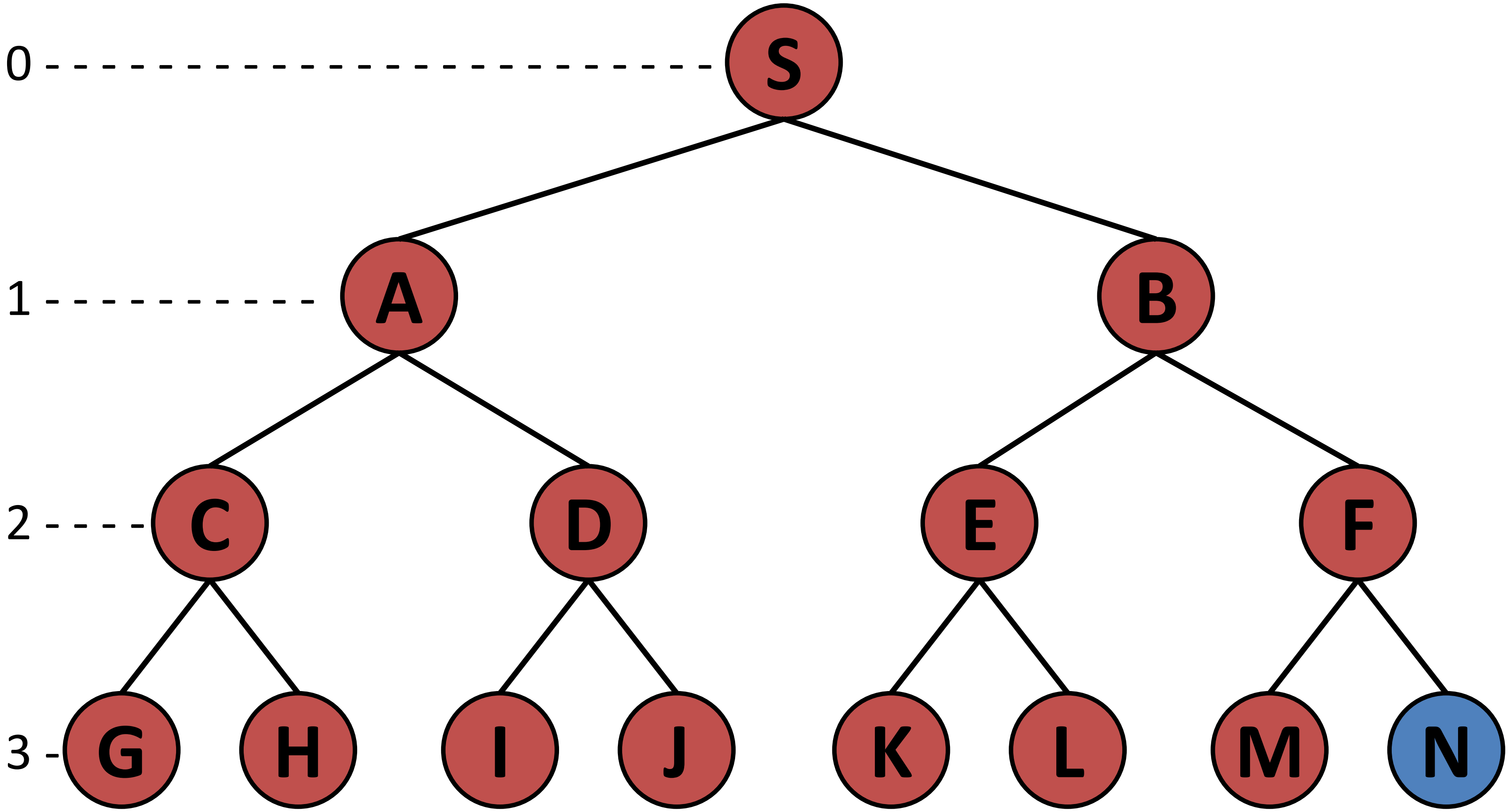
Graph and Tree Traversal: BFS



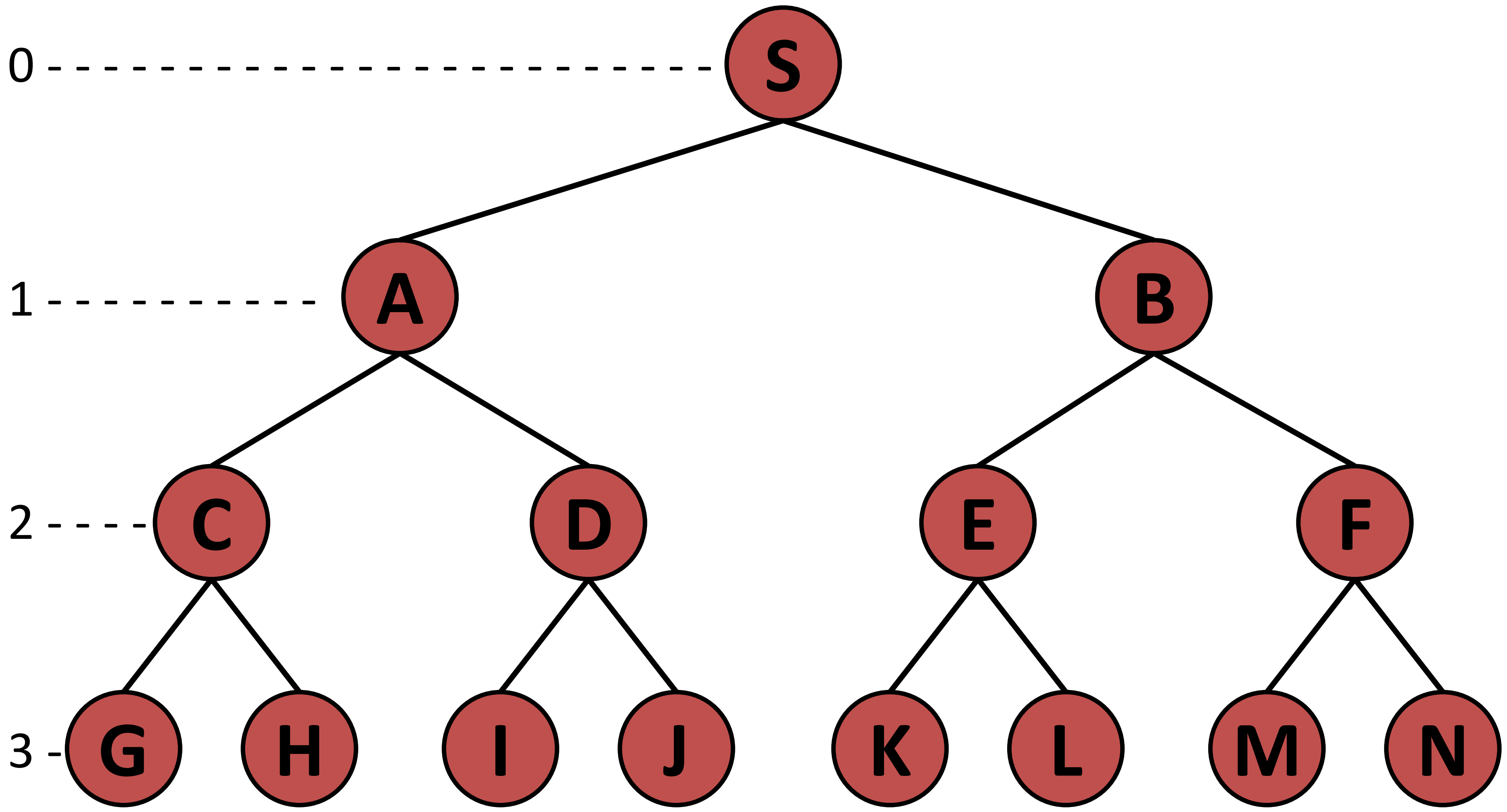
Graph and Tree Traversal: BFS



Graph and Tree Traversal: BFS



Graph and Tree Traversal: BFS


0
1
2
3
S
A
B
C
D
E
F
G
H
I
J
K
L
M
N