Find your way here

CSC 250: Foundations of Computer Science I

Fall 2023 - Lecture 14

Amitabha Dey

Department of Computer Science
University of North Carolina at Greensboro

Tree Traversal

In computer science, tree traversal (also known as tree search and walking the tree) is a form of graph traversal and refers to the process of visiting (e.g. retrieving, updating, or deleting) each node in a tree data structure, exactly once. Such traversals are classified by the order in which the nodes are visited.

Unlike linked lists, one-dimensional arrays and other linear data structures, which are canonically traversed in linear order, trees may be traversed in multiple ways. They may be traversed in depth-first or breadth-first order. There are three common ways to traverse them in depth-first order:

- in-order,
- pre-order and
- post-order.

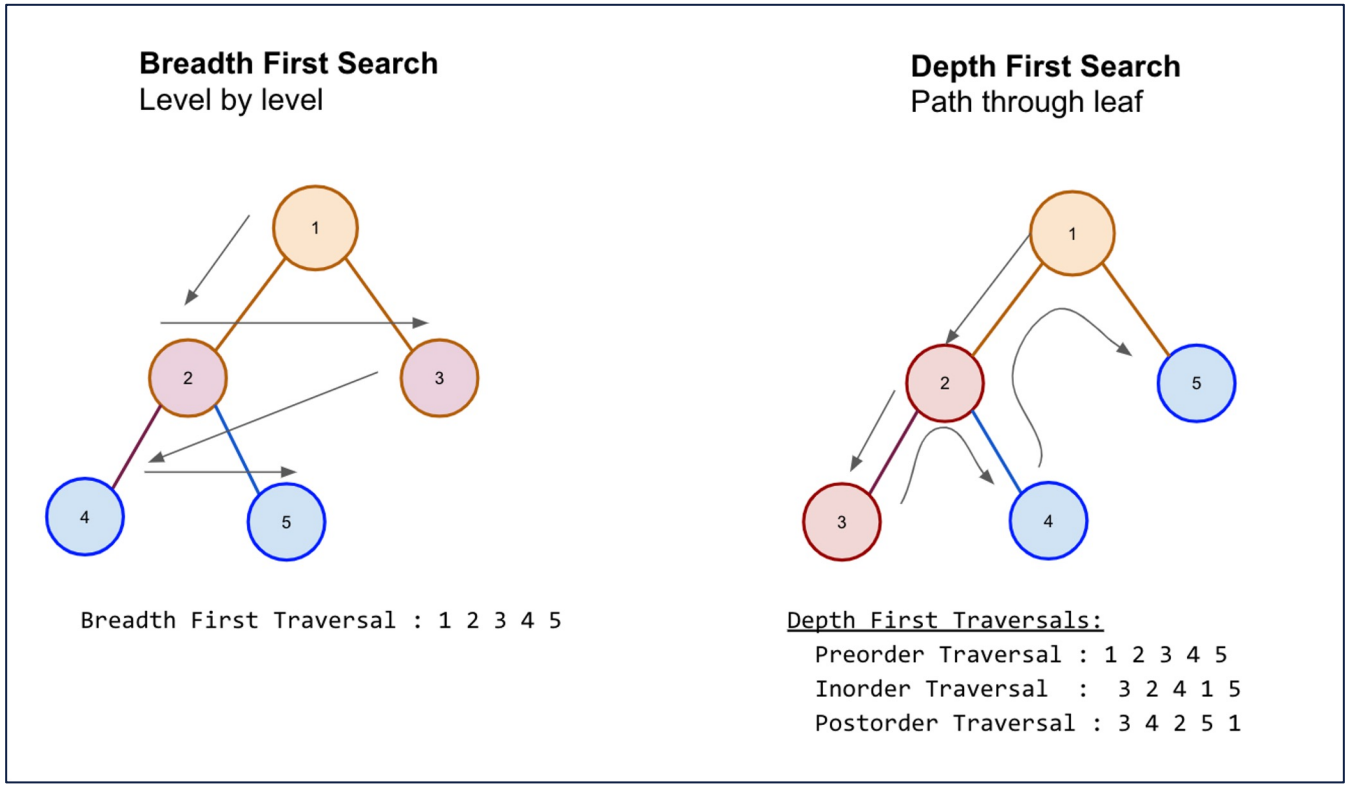


Image credit - https://libguides.brooklyn.cuny.edu/

Binary Tree Traversal

Preorder Traversal

In this traversal method, the root node is visited first, then the left subtree and finally the right subtree.

```
Until all nodes are traversed –
Step 1 – Visit root node.
Step 2 – Recursively traverse left subtree.
Step 3 – Recursively traverse right subtree.
```

Inorder Traversal

In this traversal method, the left subtree is visited first, then the root and later the right sub-tree. We should always remember that every node may represent a subtree itself.

If a binary tree is traversed in-order, the output will produce sorted key values in an ascending order.

```
Until all nodes are traversed –
Step 1 – Recursively traverse left subtree.
Step 2 – Visit root node.
Step 3 – Recursively traverse right subtree.
```

Postorder Traversal

In this traversal method, the root node is visited last, hence the name. First we traverse the left subtree, then the right subtree and finally the root node.

```
Until all nodes are traversed –
Step 1 – Recursively traverse left subtree.
Step 2 – Recursively traverse right subtree.
Step 3 – Visit root node.
```

Binary Tree Traversal

InOrder(root) visits nodes in the following order:
4, 10, 12, 15, 18, 22, 24, 25, 31, 35, 44, 50, 66, 70, 90

A Pre-order traversal visits nodes in the following order:
25, 15, 10, 4, 12, 22, 18, 24, 50, 35, 31, 44, 70, 66, 90

A Post-order traversal visits nodes in the following order:
4, 12, 10, 18, 24, 22, 15, 31, 44, 35, 66, 90, 70, 50, 25

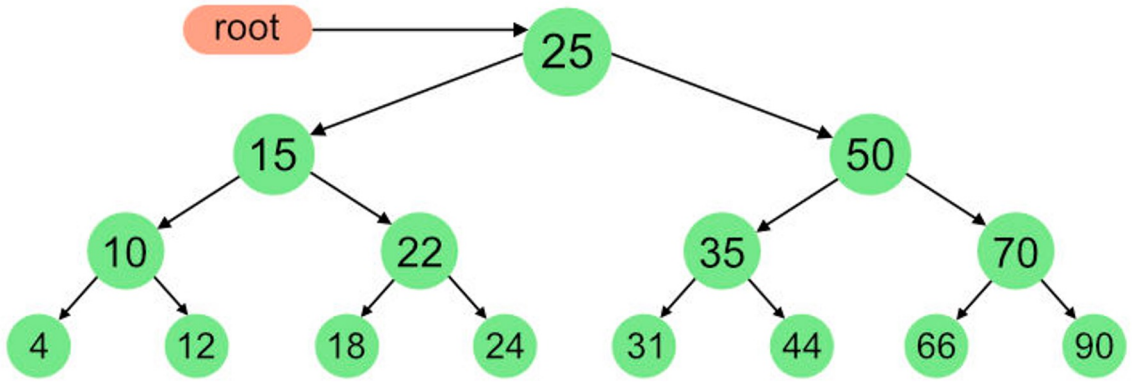


Image credit - https://www.geeksforgeeks.org/

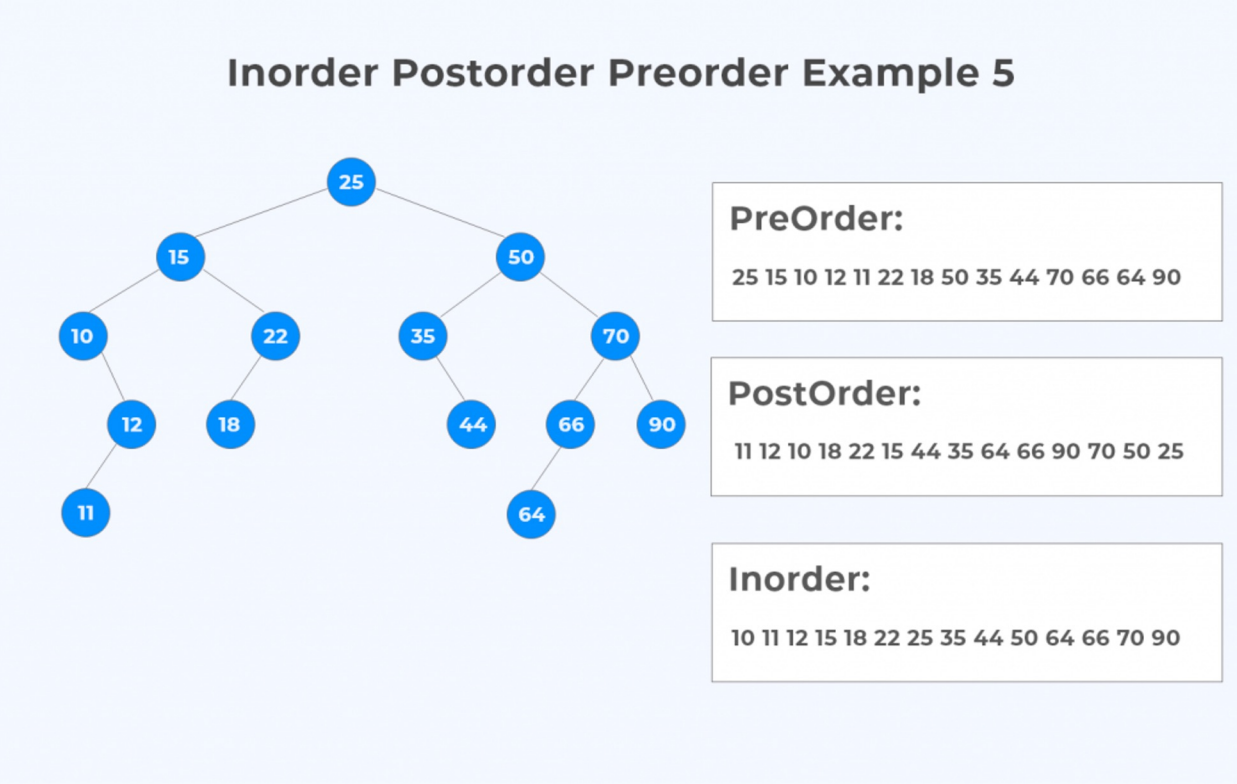


Image credit - https://prepinsta.com/

Applications of Tree Structures and Traversals

Game Development

In a 3D game, a typical scenario involves an avatar controlled by the player wandering around in a virtual environment and interacting with its surroundings. Such games usually contain many virtual entities, representing different objects. In order to have some degree of realism, the user's avatar should at least interact with these entities as if they were solid objects (instead of passing through them).
For this purpose, collision detection is necessary. It can easily be seen that as the number of entities grows larger, naively checking for collisions with every object becomes computationally infeasible. As a matter of fact, most of these checks are redundant, as the objects and the avatar are too far away.

Space partitioning trees recursively divide the space into smaller and smaller cells until a specific cell size is reached. Leaf nodes correspond to regions of the virtual environment and contain the objects that are currently within it. Thus, checking for collisions boils down to finding the cell where the avatar currently is, and checking for collisions with objects only in the neighboring cells.

This process massively reduces the number of calculations and allows real-time collision detection.

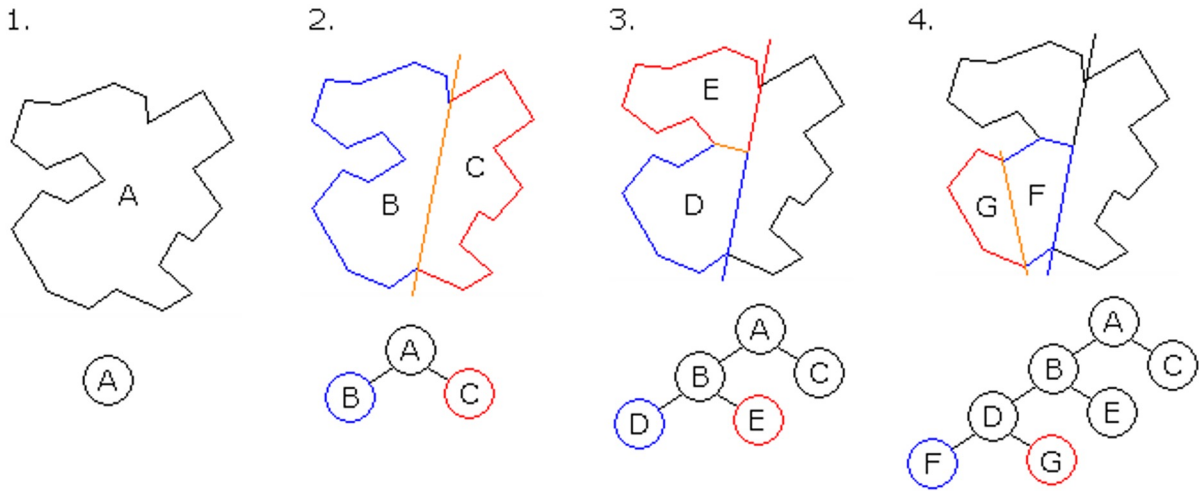


Image credit - https://www.baeldung.com/cs/tree-examples

Applications of Tree Structures and Traversals

Huffman Coding

Huffman coding is a popular algorithm used for data compression. It is widely used in various real-world applications to efficiently represent and compress data.

- Data Compression: Huffman coding is widely used in data compression algorithms, such as ZIP and GZIP, to reduce the size of files and improve data transfer efficiency over networks.
- Text Compression: Huffman coding is applied in text file compression, making it an integral part of various document storage and transmission systems.
- Image Compression: In image formats like JPEG, Huffman coding is used to compress the frequency information of image components, resulting in smaller file sizes.
- Video Compression: Huffman coding is used in video compression techniques like H.264 and H.265 to reduce the size of video files for streaming and storage.
- File Archiving: Many archiving tools, like WinRAR and 7-Zip, use Huffman coding to compress and extract files efficiently.
- Network Protocols: Huffman coding can be employed in network protocols to reduce the bandwidth required for data transmission, improving network performance.
- Resource Allocation: Huffman coding is used in applications related to resource allocation, such as assigning memory space to programs based on their memory requirements.

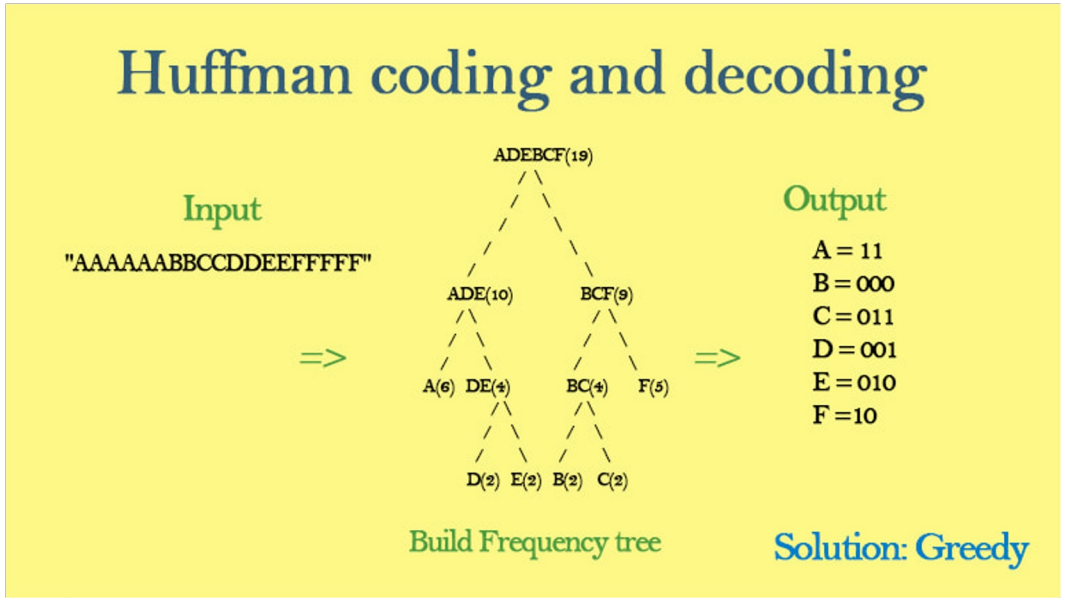


Image credit - https://www.lavivienpost.com/

Applications of Tree Structures and Traversals

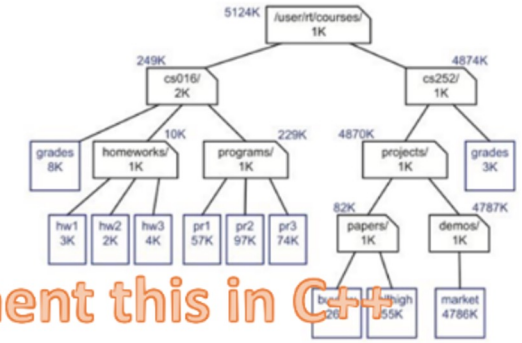
File Systems

File systems often use tree structures to represent directory hierarchies, and traversals are essential for navigating and managing files and directories efficiently.
Tree traversal is essential for various file system operations:

- Listing Files and Directories: Traversing the tree allows you to list the contents of a directory, such as listing all the files and subdirectories within the User1 directory.
- File Search: You can perform file searches by traversing the tree to find a specific file based on its name, extension, or other attributes.
- File and Directory Management: Traversals are used for creating, moving, renaming, and deleting files and directories within the file system.
- Path Resolution: When you specify a file path (e.g., C:\Users\User1\Documents\File1.txt), the file system uses traversal to navigate to the target file or directory.
- Access Control: Traversing the file system is necessary for enforcing access control permissions and determining which users can access or modify specific files and directories.
- Backup and Recovery: Backup and recovery processes involve traversing the file system to identify and save or restore files and directories.
- Disk Space Analysis: Tree traversal can be used to analyze the disk space usage within the file system, helping users and administrators manage available storage efficiently.

Example 7.7: *Consider a file-system tree T, where external nodes represent files and internal nodes represent directories (Example 7.1). Suppose we want to compute the disk space used by a directory, which is recursively given by the sum of the following (see Figure 7.9):*

- *The size of the directory itself*
- *The sizes of the files in the directory*
- *The space used by the children dire*



Homework, implement this in C++