Tree Traversal: Breadth-First Search and Depth-First Search

Depth-First Search Complexity

DFS has time and space complexity of `O(n)` .

DFS Implementations

DFS is an exhaustive search algorithm for searching tree data structures that can be implemented with a recursive approach or an iterative one.

Depth-First Search Algorithm

Our depth-first search algorithm was implemented using recursion. The implementation returns the path to the first `TreeNode` encountered with the matching target value.

```
def dfs(root, target, path=()):
  path = path + (root,)

  if root.value == target:
    return path

  for child in root.children:
    path_found = dfs(child, target, path)

    if path_found is not None:
      return path_found

  return None
```

Depth-First Search and Stacks

The most concise and efficient DFS implementations use a stack, either explicitly (iterative implementation) or implicitly (call stack from recursion).