Date:- / /
Page No.:- 15.

Title :- To convert given binary tree, into threaded binary tree.

Problem Statement :- convert given binary tree into threaded binary tree Analyze time and space complexity of the algorithm

objectives :- To study the conversion of binary tree into Threaded Binary Tree

Softwarx Requirement :-

Theory :-

In a binary tree, a thread refers to a link between a node and its in-order successor or procedessor, which can be used to traverse the tree without using recursion or a stack. A threaded binary tree is a binary tree in which every node is threaded to its in-order successer predecesser, creating a more efficient traversal algorithm

The problem Statement requires converting a given binary Tree into a threaded binary tree This can be achieved by performing an in-order traversal of the binary tree and threading each node to its in-order sucessor as it is visited The in-order traversal can be implemented recursively or iteratively using a stack and the threading can be accomplished by adding links between nodes as, they are visited

- The time complexity of this algorithm will be O(n) where n is number of codes in the binary tree

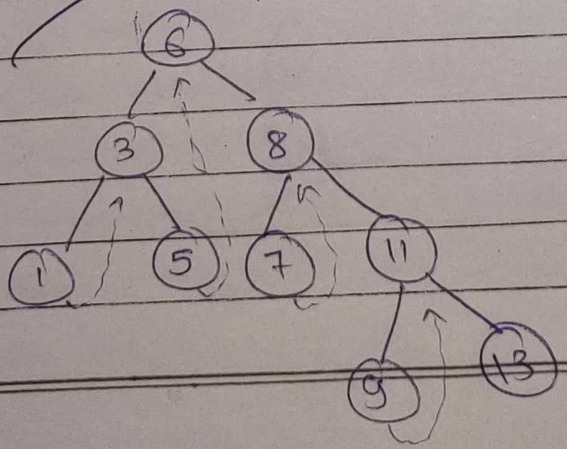
- A threaded Binary trees is in a type of binary tree data structure where the empty left and right child pointers in a binary tree are replaced with threads that link nodes directly to their in-order predecessor or successor, thereby providing a way to traverse the tree without using recursion or a stack

- Threaded binary trees can be useful when space is a concern as they can eliminate the need for a stack during traversal. However they can be more complex to implement than Standard binary trees

There are two types of threaded: binary trees
1. single threaded: where a NULL right pointers is made to point to the inorder successor
2. Double threaded: where both left and right NULL pointers are made to point to inorder predesser and inorder successor respectively. The predessor threads are useful for reverse inorder traversal and postorder traversal

The threads are also useful for fast accessing ancestors of nodes

Following diagram shows an example of single Treaded Binary tree The dotted lines represents threads.



Date:- / /
Page No.:- 17

Algorithm :-

1. Start
2. Traverse the binary tree in-order (left, root, right), keeping track of the previously visited node.
3. For each node visited, check if its left child is null. If it is, set the left child to be the previously visited node and set the left thread flag to true
4. If the previously visited node's right child is null, set the right child to be the current node and set the right thread flag to true.
5. - Repeate steps 3 & 4 until all nodes have been visited
   - The time complexity of this algorithm is o(n) where n is the number of nodes in the binary tree as it requires traversing each node once.
   - The space complexity is o(n), where h is the height of the binary tree, as the algorithm requires maintaining a stack to store the visited nodes and their parent nodes.

Conclusion :- The threaded binary allows for efficient traversal and searching, as the threads provide direct access to the predecessor and successor nodes without the need for recursion or a stack.