INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 4, Issue 3)
Available online at**:** www.ijariit.com

A review on tree traversal techniques

Raj Bala
riyainsha8026@gmail.com
Chaudhary Devi Lal University, Sirsa, Haryana

Sakshi Dhingra
sakshi24.dhingra@gmail.com
Chaudhary Devi Lal University, Sirsa, Haryana

ABSTRACT

A tree is a non-linear data structure for fast storing and retrieval of data in primary memory. It represents data in the form of hierarchical form. Data are stored in a tree i.e. called as a node, in which topmost node is called root and each node has one or more nodes lying on the left or right side of a tree. Except for root node each node has a parent node. The information can be extracted from a tree through various traversal algorithms. Tree traversal means visiting the nodes of a tree at once. In this paper, we are studying different algorithms for tree traversal

Keywords— *Tree, Node, Traversal, Path*

1. INTRODUCTION

A tree is one of the most important parts of computer sciences. A tree is a non-linear data structure (such as graphs and trees), in which data is represented in a hierarchal manner. Every element in a tree is called node. It is a collection of different nodes. When we want to represent a hierarchical relationship represented between family members, employees in company etc., the trees are very flexible, powerful and versatile data structure. In a tree, data is organized in random order [2]. The data of particular elements are stored in a node of the tree and linked to next element in the tree structure. The topmost element in a tree is called root. In a tree, except for the root node, each element has a parent node. Each parent node has zero or more children. It is called a left child or a right child. A tree has different types of terminology [8]:

Node: Each data items are represented in a tree through node.

Root node: In a tree, the root node is topmost node.

Internal/Non- Terminal Node: A node that is not a root node but has at least one child that is called non-terminal node.

External/Terminal or Leaf Node: External nodes are those nodes that have no children.

A degree of a node: Degree of a node tells indicates the number of subtrees in a given tree.

Siblings: The Descendants nodes of a given ancestors node are called siblings.

Level: A level number is assigned each to a node in a tree. The root node is always at level zero. After root node children are at level 1 and their immediate children at level 2 and it goes on up the at last node of a tree.

Path: It is a sequence of consecutive edges from the source node to destination node.

Depth/Height: How many levels or maximum level of a tree i.e. called depth or height of a tree.

Branch: Ending of a leaf node in a tree is called the branch.

When we are performing an operation on a tree for retrieval of an information, we are visiting or walk the tree i.e. called tree traversal. Different types of algorithms are used for traversal of a tree. Preorder Traversal, in Order Traversal, Post order traversal or Level Order Traversal. DFS or BFS algorithm is also used for a tree traversal. Both are used as a different process to traverse a tree.

Application of Non-Linear Data Structure: Tree

There are many applications of non-linear data structure tree, i.e. medical data, policy proposal, a non-medical data, underwriter data sources, 3D video games, file storage, space partition, specialization of image signature [6]. Also, have the property of image filtering. Min Tree and Max Tree terms are used for image filtering [13].

2. LITERATURE REVIEW

Suri Pushpa, Prasad Vinod [1] This paper has proposed some algorithm for balancing the height of the binary tree. The author gives the idea as to how we can maintain the shape of a binary tree in a proper way. And how to manage the time and space requirements can be managed. In this paper author proposed some tree balancing algorithm. These algorithms can be static (global) or dynamic in nature. Both algorithms have pros and cons. Adel's son-Velskii & Landis proposed AVL Tree (dynamic algorithm) for balancing the binary tree. Some examples of this come close to are the weight-balance tree, height-balanced tree, and B-trees. In the static algorithm, rebalancing is performed when the need arises. Most algorithms are static in nature. The static algorithm is better performing when tree size is small. The author has given the idea that how we can manage the balance of tree after insertion or deletion of a node. The main purpose of this paper is to show the height of a tree in O(log(n)), so that all the basic operation or tree perform in O(log(n)) time. Runtime complexity is O(N) of each node in a tree. Over the years

computer scientists have proposed different balancing algorithm. After the study of different types of balancing algorithms, algorithms are not given an idea that balances the tree in lesser time. So there are more opportunities for the researcher in future.

Hua Li [5], the author has discussed binary tree & its properties which is a very vital data structure in computer science. In this paper author discussed recursive & non-recursive algorithm in detail. The author wants some improvement in existing programming. The comparison made between recursive and non-recursive algorithm. A recursive algorithm is very easy and trouble-free. Through the recursive call, we made binary tree traversal very speedily. In the recursive algorithm, all traversal depend upon the pointer, when a pointer is lost the traversal process is stopped. It is calling itself in a program so that efficiency of the program is low. When the efficiency of the recursive program is low, then the author follows non-recursive traversal algorithm. A non-recursive algorithm is totally different from it. A non-recursive algorithm is very difficult binary tree traversal algorithm. It is used to stack to store the data from tree node. When the operations are performed on the tree, the outputs come one after other. In the paper, the author intends to propose the improvement in the non-recursive algorithm for better program efficiency and removes the complexity of the non-recursive algorithm.

Dr. N. Rama, Justin Sophia. I [12], the author proposed DFS & BFS searching algorithm. Data structure plays an important role in a mathematical formulation for creating good software. The author proposed DFS algorithm for full binary tree traversal (FBT) through preorder. In this, the author paper tries to find the time taken by DFS & BFS algorithm for traverse the height of a full binary tree.

In DFS traversal, the tree is traversed depth wise. When traversal process starts from the root node and traverses it, at last, to the leaf node. It is implemented using a stack. In which backtracking is performed on DFS algorithm. It is generally gets trapped into infinite loops. In a BFS traversal tree is visited as a level wise. A queue is used for implementation of BFS tree. In a BFS traversal backtracking is not required like as DFS. It is never gets trapped into infinite loops. The main purpose of the author in this paper is to calculate the time taken to search a destination node by DFS by preorder traversal algorithm.

Mr. Chandrashekahr S. Khese, Prof. Amrit Priyadarshi [6], Arrays, vectors and liked lists data structure is a linear data structure in which data are stored in sequence but in tree data structure data is stored in hierarchical form. It means one node has further one or more node. In tree data structure data are stored at random from. The shape of a tree depends on the form of data. When data is not in the form of random, the height of the tree becomes longer on one side, and the shape of the tree becomes wider and flatter. In this paper discussed worst-case analysis in BST tree (binary search tree). The aim of this paper is to show how we can maintain the height of the tree and how we can perform an operation on a tree in O(log(n)) time.

Kevin Andrusky, Stephen Vurial and Jose Nelson Amaral [3], In the paper, an attempting made to present a profiling-based analysis to decide the traversal direction of link-based tree data structures. The paper underlines the fact that the way of traversal of the tree is predefined. It means the nature of algorithm that is used in a tree to perform an operation on a tree that can be breadth first or depth first. The storage capacity of the profiling based analysis is efficient. When it used, the non instrumented code 7 percent extra memory is required. When the profiling based analysis of the tree is performed the result comes out in the form of float point. So that how we can consider the nearest vale of BFS or DFS. Those values are used by the programmer or researcher for improvement in the data structure. The author proposes to traversal of the tree not required the modification on the part of the programmer but small modification to a compiler.

V. R. Kanagavalli1, G. Maheeja [11], this paper promoted the idea for the retrieval of the information from tree data structure. In this paper, the author well defines IR (Information Retrieval) or IE (Information Extraction). Both techniques IE and IR have a different goal. IE does not set any goal but IR set a well defined a goal to satisfy the users need regarding retrieval information or data from a non-linear data structure. The man purpose of information retrieval is to answer the user's queries.

In this paper, different data structures are used for information retrieval i.e. storage data structures in information retrieval, process-oriented data structures in information retrieval, descriptive data structures in information retrieval.

Niloofar Aghaieabiane, Henk Koppelaar, and Peyman Nasehpour [10], the paper proposed an algorithm for reconstructing a binary tree. Most of the algorithms have been proposed for a binary tree from its inorder and preorder traversals. In this paper proposed an Inpos algorithm for reconstructing the binary tree from its inorder and postorder traversal i.e. takes the time O(n2) for running an algorithm. In this paper proposed an improved inpos algorithm i.e. takes time and space complexity is θn. The inpos algorithm also has been a feature that also provides the information about how many sub-nodes of a parent node that can be one, two or more. The inpos algorithm provides a matrix based structure. So we can access the information from a given tree in linear time.

Parth Patel and Deepak Garg [4], the author discussed different types of advance tree data structure. B-tree and R-tree both are the basic index structure. In this paper, the author made a comparison between B-tree and T-tree and its applications. Some factors were presented in data structure i.e. complexity, query type support, data type support, and application, on the basis of those factors comparison is made in the advance tree data structure. Proposed modified B-tree and R-tree are used in real word for the query and performance optimization. Those proposed trees; some have less time and space complexity. B-tree and R-tree support the single query and multidimensional query in that order. The author proposed a new index structure with improvement in existing data structure like a hash function. It is used by BR tree. A new index structure is proposed by using the existing two different index structure that made a change in algorithm same as hash index structure.

Jeffrey L. Eppinger[9], In this study, the author has given an idea, how we can perform an operation on tree i.e. insertion or deletion randomly. That operation performed on a binary tree by symmetric or asymmetric methods. That operation in this paper the author proposed a comparison between insertion and asymmetric deletion algorithm i.e. used for the expected internal path of a binary tree. The main aim of the author in this paper is how we can improve in insertion and asymmetric deletion algorithm that can provide a binary tree with decreased internal path length after performing operations. Because when we make an operation (insertion or asymmetric deletion) on a random binary tree, the length of the internal

path more increased that cause-effect on balance of a random binary tree. However, the author proposed a symmetric algorithm for the solution of above-discussed problems. Asymmetric deletion algorithm made a balance in which randomly binary tree when insertion or deletion operation is performed.

Navneet Kaur, Deepak Garg [7], had given an idea of how we can reduce the unnecessary research node of a tree. "Analysis of the Depth First Search Algorithms". In this paper, the author focuses on how we can improve the search in a non-linear data structure (tree). In which is used for the RHS algorithm and parallel formulation for improvement in search, better time and space management for memory.DFS algorithm has a backtracking record i.e. main drawback of this algorithm. That drawback is removed by using the RHS algorithm [13]. Which is not required for the backtracking on a node. Using RHS algorithm or parallel formulation we can improve the searching time and given the better results. Backtracking is mostly responsible for the worse performance of DFS.

Table 1: A brief overview of this literature review

No.	Types of Algorithms	Time complexity	Space complexity	Data Structure
1.	BFS	O(|n|): where n is the number of nodes	O(|n|)	Queue
2.	DFS	O(|n|)	Depends on implementation	Stack
3.	DFS(recursive implementation)	O(|n|)	O(|h|): h is the maximal depth of tree	
4.	DFS(with iterative solution)	O(|n|)	O(|n|)	Stack
5.	Recursive algorithm (Fibonacci sequence)	O(2^n)	O(nm): n is the maximum depth of recursion tree	
6.	Component tree computation algorithm (memory access with minimum degree b)	O(b+$\log_b$v)(as per memory access)	O(b+$\log_b$v)	Stack
7.	RHS algorithm for improvement in DFS algorithm[7]	O(N)	O(N)	Stack
8.	RHS (in case of complete Binary tree)[7]	O(2^n-1)	-	Stack
9.	Iterative deepening depth-first search (IDDFS) algorithm(for well-balanced tree)	O(b^d): where b is the branching factor and d is the shallowest solution	O(d)	Stack
10.	Martin & Ness's Balancing Algorithm	O(N)	The stack is used to carrying out the traversal	Stack
11.	A Colin day	O(N)	Little space is required	Contiguous memory
12.	Change & Ayengar	O(N)	Additional workspace required = size of tree	Not used Stack
13.	Stout & Warren	O(N)	Only fixed amount of space is required	-
14.	In order traversal without recursion	O(N)	-	Stack
15.	Inorder traversal using recursion & iterative algorithm	O(n)	O(n)	Stack
	Preorder traversal (iterative and non-recursive)	O(n)	O(n)	Stack
	New modified non-recursive algorithm[14]	O(N)	O(NlogN)	-
	Max-tree algorithms.[13]	Shows in table No. 2(n is the number of pixels and k the number of gray levels)		

Table 2: Max Tree Algorithm

Algorithm	Time complexity			Auxiliary space requirement		
	Small int	Large int	Generic int	Small int	Large int	Generic int
Berger	O(n log n)	**O(n log n)**	O(n log n)	n+k+O(n)	2n+O(n)	n+O(n)
Berger+rank	O(n α(n))	O(n log log n)	O(n log n)	3n+k+O(n)	4n+O(n)	3n+O(n)
Naiman and couprie	O(n α(n))	O(n log log n)	O(n log n)	5n+k+O(n)	6n+O(n)	5n+O(n)
Salembier et al.	O(nk)	O(nk) ≃ (n^2)	N/A	3k+n+O(n)	2k+n+O(n)	N/A
Nister and stewenius	O(nk)	O(nk) ≃ (n^2)	N/A	2k+2n	2k+2n	N/A
Wilkinson	O(n log n)	O(n log n)	O(n log n)	N+k+O(n)	3n	3n
Salembier non-recursive	O(nk)	O(n log log n)	O(n log n)	N+k+O(n)	3n	3n

3. CONCLUSION

In this review paper discussed various non-linear data structure tree, and also discussed different kinds of tree traversing techniques. In existing tree traversing algorithms an issue Occurs about time complexity, space complexity and height balance of a tree. The author changed the existing algorithm day by day for better performance, according to a need of time in the data structure. The algorithm that can balance a tree in less time since has not been developed. In future, best scope in the improvement in existing methods.

4. REFERENCES

[1] Suri Pushpa, Prasad Vinod, Binary Search Tree Balancing Methods: A critical Study *International Journal of Computer Science and Network Security*, Volume.7 No.8, August 2007

[2] Rubi Dhankhar, Sapna Kamra, Vishal Jangra, Tree Concept in the data structure, *International Journal of Computer Applications*, Volum.1, 2014

[3] Kevin Andrusky, Stephen Curial, and Jose Nelson Amaral. Tree Traversal Orientation Analysis

[4] Parth Patel and Deepak Garg, Comparison of Advance Tree Data Structures, *International Journal of Computer Applications*, Volume No. 2, March 2012

[5] Hua Li, Binary Tree's Recursion Traversal Algorithm and Its Improvements, May *2016*.

[6] Mr. Chandrashekhar S. Khese, Prof. Amirit, Binary Search Tree and Its Applications: Survey, *International Journal on Recent innovation Trends in Computing and Communication*, Volume 3, November 2015

[7] Navneet Kaur, Deepak Garg, Analysis of The Depth First Search Algorithms

[8] Ramesh M. Patelia, Shilpan D. Vyas, *Basic Tree Terminology*, Their Representation and Application 2015.

[9] Jeffrey L. Eppinger, An Empirical Study Of Insertion And Selection In Binary Search Trees, Volume No. 26, September 1983

[10] Niloofar Aghaieabiane, Henk Koppelaar, and Peyman Nasehpour**,** An improved algorithm to reconstruct a binary tree from it's in order and postorder traversals, April 4, 2018.

[11] V. R. Kanagavalli1, G. Maheeja, A Study on the Usage of Data Structures In Information Retrieval

[12] Dr. N. Rama, Justin Sophia. I, An Inquisitive Result in DFS Problem of Binary Trees, *International Journal of Scientific Research and Management*, Volume.5, July 2017.

[13] Edwin Clarinet and Thierry Geraud, A Comparative Review of Component Tree Computation Algorithms, Volume. 23, No.9, September 2014

[14] Nitin Arora, Pradeep Kumar Kaushik, satendra Kumar, Iterative Method for Recreating a Binary Tree form its Traversals, *International Journal of Computer Applications*, Volume. 57 No.-11, November 2012.

