International Journal of Science and Engineering Applications
Volume 9 - Issue 11, 125 - 128, 2020, ISSN:- 2319 - 7560

Securing Data by Using Tree Traversal Techniques

Girija Rani Suthoju
Assistant Professor,
Department of CSE,
Koneru Lakshamaiah Education Foundation, Hyderabad, TS.

S. Swapna
Associate Professor,
Department of CSE,
Pallavi Engineering College, Hyderabad, TS.

Farhana Begum Sattar
Assistant Professor,
Department of CSE,
Vardhaman College of Engineering, Hyderabad, TS.

Abstract: Now-a-days, securing data is a typical scenario and secure world is inviting hackers proportional to the technology. Thus, Security must be provided to data in all sides by encoding data at sender and releasing in to network, on other side at receiver, the data must be decoded with the provided credentials. In the proposed system of this paper, we are introducing binary tree traversals to secure data as a ciphering technique. The data may be extracted from a tree through numerous traversal algorithms.
Keywords: ciphers, traversals, security, trees and encoding

1. INTRODUCTION

A tree is a non-linear data structure for instant storing and retrieval of statistics in number one memory. It represents records in the form of hierarchical shape. Statistics are saved in a tree i.e., referred to as a node, in which topmost node is known as root and every node has one or extra nodes lying at the left or proper facet of a tree. Except for root node each node has a parent node. The data may be extracted from a tree through numerous traversal algorithms. Tree traversal means journeying the nodes of a tree right now. On this paper, we are reading one of a kind algorithms for tree traversal.

A tree is one of the most critical elements of computer sciences. A tree is a non-linear data structure (such as graphs and trees), wherein facts is represented in a hierarchal way. Every element in a tree is called node. It is a collection of various nodes. When we want to symbolize a hierarchical courting represented among own family participants, personnel in organization and many others, the trees are very bendy, powerful and flexible records shape. In a tree, statistics is organized in random order. The statistics of unique elements are stored in a node of the tree and related to subsequent element inside the tree shape. The topmost element in a tree is referred to as root. In a tree, except for the basis node, every detail has a figure node. Each discern node has 0 or extra children. It's miles called a left child or a proper child**.**

2. MAIN CONTRIBUTIONS

While we are acting an operation on a tree for retrieval of an data, we're traveling or walk the tree i.e., called tree traversal. Different kinds of algorithms are used for traversal of a tree. Preorder traversal, in order traversal, submit order traversal or stage order traversal. Dfs or bfs set of rules is likewise used for a tree traversal. Both are used as a specific method to traverse a tree

Here are numerous applications of non-linear data structures, tree: i.e,. Clinical statistics, coverage proposal, a non-clinical information, underwriter information resources, 3D video games, record garage, space partition, specialization of image signature. Also, have the assets of picture filtering. Min tree and max tree phrases are used for image filtering.

3. OUR SYSTEM AND ASSUMPTIONS

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)		

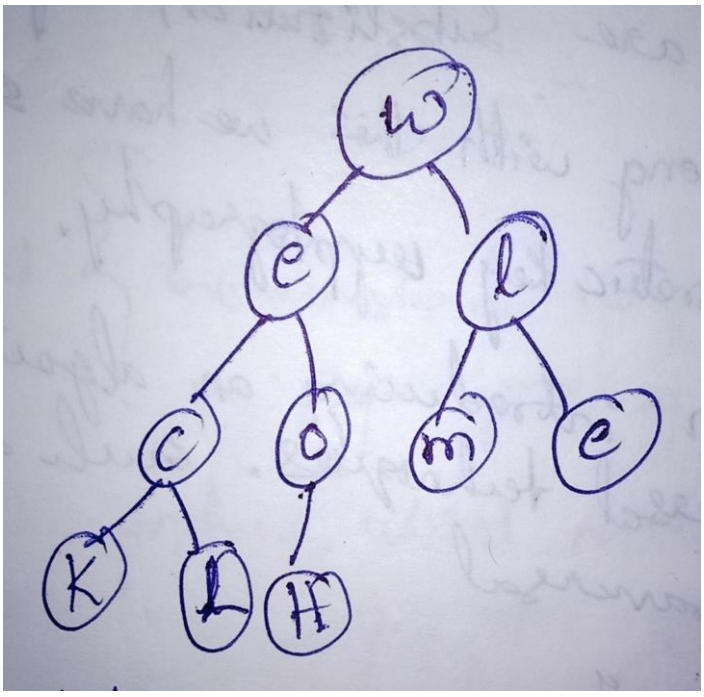
Fig .1: Complexities of Tree

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 courrie	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

Fig .2: Continuation of time and space complexities

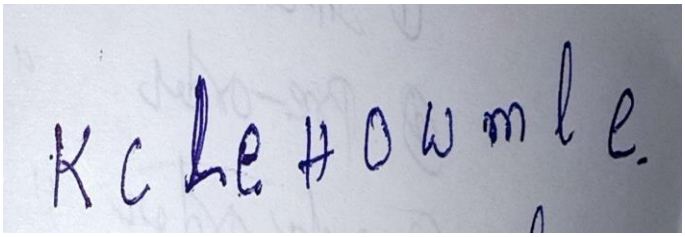
4. IMPLEMENTATION AND EXPERIMENTAL EVALUATION

Consider the tree as follows:

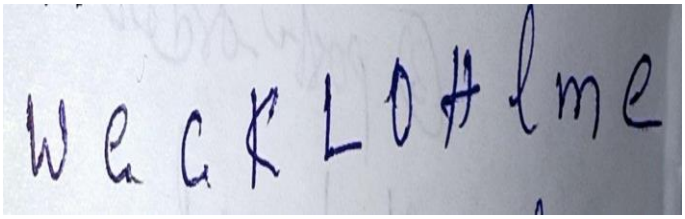


Then the tree traversals may be as follow:

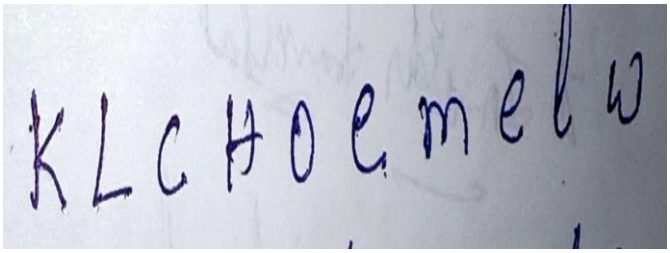
Inorder :



Preorder:



Postorder:



The related code in python can be given as:

```
class Node:
   def __init__(self, item):
      self.left = None
      self.right = None
      self.val = item


def inorder(root):
   if root:
      # Traverse left
      inorder(root.left)
      # Traverse root
      print(str(root.val) + "->", end='')
      # Traverse right
      inorder(root.right)


def postorder(root):
   if root:
      # Traverse left
      postorder(root.left)
      # Traverse right
      postorder(root.right)
      # Traverse root
      print(str(root.val) + "->", end='')
```

```
def preorder(root):
   if root:
      # Traverse root
      print(str(root.val) + "->", end='')
      # Traverse left
      preorder(root.left)
      # Traverse right
      preorder(root.right)
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)
print("Inorder traversal ")
inorder(root)
print("\nPreorder traversal ")
preorder(root)
print("\nPostorder traversal ")
postorder(root)
```

5. CONCLUSIONS

Traversing a tree approach travelling every node within the tree. You might, as an instance, want to add all of the values within the tree or locate the largest one. For a lot of these operations, you'll need to visit each node of the tree. In this assessment paper mentioned various non-linear facts shape tree, and also mentioned exclusive forms of tree traversing strategies. In present tree traversing algorithms an trouble occurs about time complexity, space complexity and top stability of a tree. The writer modified the prevailing algorithm day by day for higher overall performance, according to a need of time in the facts shape. The algorithm that can balance a tree in much less time due to the fact that has not been evolved. In destiny, quality scope in the improvement in existing methods. Linear data structures like arrays, stacks, queues, and linked listing have most effective one manner to study the facts. However a hierarchical data structure like a tree may be traversed in distinct approaches will be helpful to cryptographic concepts.

6. ACKNOWLEDGMENTS

Our thanks to the experts who have contributed towards paper work and guiding us in all aspects.

7. REFERENCES

[1] Kevin Andrusky, Stephen Curial, and Jose Nelson Amaral. Tree Traversal Orientation Analysis
[2] V. R. Kanagavalli1, G. Maheeja, A Study on the Usage of Data Structures In Information Retrieval
[3] Ramesh M. Patelia, Shilpan D. Vyas, *Basic Tree Terminology*, Their Representation and Application 2015.
[4] Edwin Clarinet and Thierry Geraud, A Comparative Review of Component Tree Computation Algorithms, Volume. 23, No.9, September 2014

8. AUTHOR

[1] Ms. GIRIJA RANI SUTHOJU pursuing Ph.D in CSE department at JNTU Hyderabad. She received Masters degree (M.Tech) in the Department of Computer Science and Engineering at Aurora's Technological and Research Institute belongs to Jawaharlal Nehru Technological University, Hyderabad in 2014 and Bachelors degree (B.Tech) in the Department of Computer Science and Engineering from the same University in 2012.
She worked as an Assistant Professor for 5years in Aurora's Technological and Research Institute affiliated to Jawaharlal Nehru Technological University, Hyderabad. At present she is working in Koneru Lakshmaiah Educational Foundation (K L deemed to be University) Hyderabad.
Ms. Girija Suthoju received Merit Certificate in her Masters from ATRI, JNT University Hyderabad for her best performance in Academics. She also received Teaching Excellence award for best teaching thrice from 2016 to 2018. At present she is working in her area of research i.e., Cryptography and network security.

[2] Ms. S. Swapna pursuing her Ph. D in GITAM University Vizag. She has received her Masters degree (M.Tech) in the Department of Computer Science and Engineering at Aurora's Technological and Research Institute belongs to Jawaharlal Nehru Technological University, Hyderabad. She worked as an Assistant Professor for 5years in Aurora's Technological and Research Institute affiliated to Jawaharlal Nehru Technological University, Hyderabad. At present she is working in her area of research i.e., Cyber security.

She also received Teaching Excellence award for best teaching thrice from 2016 to 2018.

[3] Ms. Farhana Begum Sattar, working as an Assistant professor in CSE department at Vardhaman College of Engineering, Hyderabad. She worked as an Assistant Professor for 4years in Aurora's Technological and Research Institute affiliated to Jawaharlal Nehru Technological University, Hyderabad. At present she is working in her area of research i.e., Cryptography and network security and also in Cyber security.