Implementation of Strictly Binary Tree to Calculate Factor of an Integer

Vikas J Magar[1], Rajivkumar .S Mente[2]

[1] Research Student,
Department of Computer Science,
School of Computational Sciences,
Solapur University, Solapur
(Email: Magarvjresearch@gmail.com)

[2] Head of Department,
Department of Computer Science,
School of Computational Sciences,
Solapur University, Solapur
(Email: Rajivmente@rediffmail.com)

***Abstract*—** A Binary tree is a non-linear data structure. It has several applications. Here research article represents how to evaluate the prime factor of an integer using a binary tree. To perform this task strictly binary tree is applied. An algorithm is intended to calculate the factor of an integer. By applying an algorithm each leaf node of the strictly binary tree represents the prime factor of an integer.

***Keywords*—** *Strictly binary tree, a factor of an integer, prime factor*

I. INTRODUCTION

A binary tree is a finite set of elements that are either empty or is partitioned into three displace subsets. The first subset contains a single element called the root of the tree [1]. The other two subsets are themselves binary trees, called left & right subtree of the original tree. Left or right subtree may be empty. Each element of the binary tree is called node of the tree.

A binary tree is an acyclic graph G (V, E) where V is a non-empty set of vertices & E is set of edges. A binary tree has only one root and it has at most two children. It has maximum 2^L children at level L. subsequent diagram represent the binary tree.

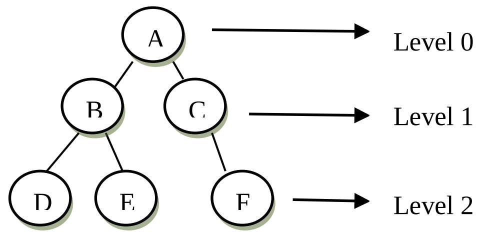


Fig.1 Binary Tree.

In fig.1, A is the root of binary tree and B is the root of its left subtree, then A is said to be the father of B and B is the left son of A. A node that has no sons (such as D, E, and F) is called leaf nodes. Two nodes are brothers if they are left & right sons of the same father [4].

The level of a node in the binary tree is defined as follows: the root of the tree has level 0, and the level of any other node in the tree is one more than its father. Maximum level of any leaf in the tree is called depth of binary tree. This equals the longest path from the root to any node [5].

In the binary tree at Level 0 maximum number of children is 2^0 =1. At level 1 the maximum number of children is $2^{1=}$ 2. Same way at level n there are 2^n children's [6]. Maximum number children at level L are= 2^{L+1}-1.

If every non-leaf node in a binary tree has non-empty left and right subtrees, then a tree is termed as a strictly binary tree. The tree in fig.2 is a strictly binary tree. Root A has two children. Node B has also two children D and E. Node C, D and E are leaf nodes.

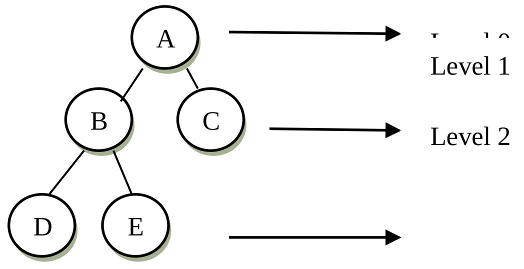


Fig. 2 Strictly Binary Tree

1.1 Dynamic representation: Fig.3 represents the dynamic structure of a binary tree where N represents NULL.

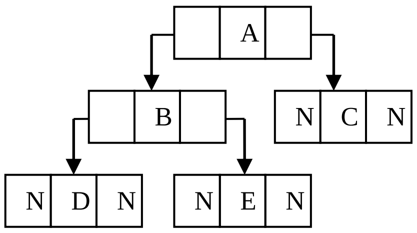


Fig.3 Dynamic Representation

1.2 Data Structure for the tree:

```
struct node
{
struct node *lchild;
int data;
struct node *rchild;
}
```

The factor of an Integer is the product of prime numbers that treat the given number. Number theory represents, prime factorization or integer factorization is the disintegration of a composite number into smaller non-trivial divisors, which when multiplied together equal the original integer. For example, consider an integer 96.

96 = 2 * 48
= 2 * 2 * 24
= 2 * 2 * 2 * 12
= 2 * 2 * 2 * 2 * 6
= 2 * 2 * 2 * 2 * 2 * 3

Thus factors of an integer 96 are
2, 2, 2, 2, 2, 3.

II. METHODOLOGY

Given an integer *n* (throughout this article, *n* refers to "the integer to be factored"), the trial division consists of systematically testing whether *n* is divisible by any smaller number. Obviously, it is only meaningful to test candidate factors less than *n* and in order from two upwards because an arbitrary *n* is more likely to be divisible by two then by three, and so on. With this ordering, there is no point in testing for divisibility by four if the number has already been determined not divisible by two, and so on for three and any multiple of three, etc. Therefore, the effort can be reduced by selecting only prime numbers as candidate factors. Furthermore, the trial factors need not go to further. If *n* is divisible by some number *p*, then $n = p \times q$ and if *q* is smaller than *p*, *n* would have earlier been detected as being prime factor of q or divisible by *q*. In this research article, trial-division method is implemented on the strictly binary tree. Following algorithm explains how a strictly binary tree is used to evaluate the factor of an integer.

III. TRAIL DIVISION METHOD

Trail division is a method which uses prime numbers to divide an integer. The movement will be in ascending order to decompose an integer. The set will contain divisor number as shown below.

Prime numbers= {2, 3, 5, 7, 11, 13, 17, 19, 23 and so on}.

I. ALGORITHM

Step 1: Input an Integer.

Step 2: Make entered integer as root of tree.

Step 3: If the entered integer is prime then print root. Go to step 6

Step 4: Now apply trial division method and try to divide number by smallest prime number. If the number is divisible then decompose it otherwise, go to step 6. Arrange the smallest integer as left child and largest as the right child.

Step 5: Now consider right child and apply step 4.

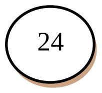
Step 6: Print all leaf nodes.

Step 7: Stop

IV. IMPLEMENTATION

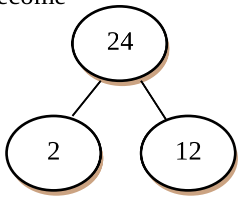
Step 1: Enter an integer 24

Step 2: Make entered integer as root. The tree will become,

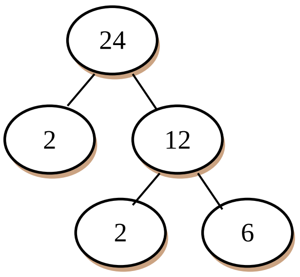


Step 3: 24 is not prime. So continue to step 4

Step 4: Apply trial division method. A number will become 24= 2*12
Now consider decomposed integer 2, 12. Smallest integer is 2. Make it as left child and 12 as of the right child. Tree will become

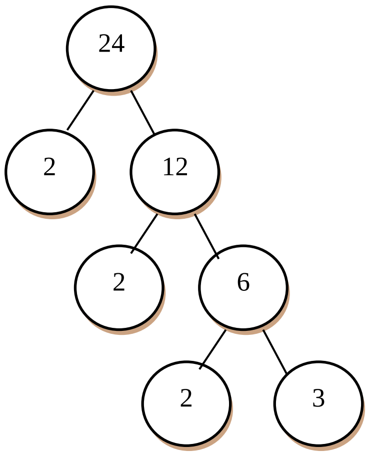


Now consider right child 12 which is not prime. Decompose 12. 12= 2* 6 smallest number is 2 make it left child and 6 as of the right child.



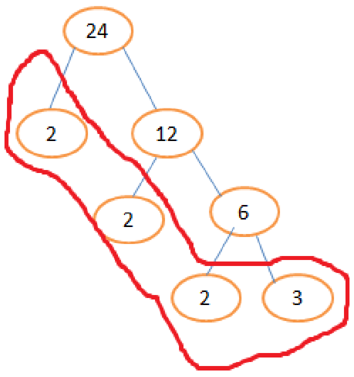
Now consider right child 6 which is not prime. Decompose 6.

6= 2*3 smallest number is 2 make it left child and 3 as for the right child.



Now consider right child 3 which is prime. So it cannot be decomposed.

Step 6: Now consider all leaf nodes of the above tree. All leaf nodes are the factor of root 24.



V. CONCLUSION:

From above algorithm hereby conclude that strictly binary tree is can be used to calculate the factor of an integer. It can be easily extended to the negative integer also. This is one more application of binary tree to calculate factors.

[1] Suri Pushpa, Prasad Vinod, “Binary Search Tree Balancing Methods: A Critical Study,” IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.8, August 2007

[2] B. Deepa, Reshmi. S, Kirthika. B, “A Survey On Different Method of Balancing A Binary Search Tree,” International Journal of Advance Engineering and Research Development Volume 4, Issue 10, October -2017

[3] Jon L. Bentley, “ Multidimensional Binary Search Trees in Database Applications,” IEEE transactions on software engineering, vol. sec-5, no. 4, July 1979

[4] Nishant Doshi, Tarun Sureja, Bhavesh Akbari, Hiren Savalia, Viraj Daxini, “Width of a Binary Tree,” International Journal of Computer Applications (0975 – 8887) Volume 9– No.2, November 2010

[5] Vinod George, “An adaptive indexed binary search tree for efficient homographic coercion resistant voting System,” The international journal of managing information Technology vol.2 No.1 Feb 2010.

[6] Suri Pushpa, “Insertion and Deletion on Binary Search Tree using Modified Insert Delete Pair: An Empirical Study,” IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.12, December 2007

[7] Vinod, P. Suri, P., and Maple, C. “Maintaining a Binary Search Tree Dynamically,” Proceedings of the 10th International Conference on Information Visualization. London, UK. 5-7 th July 2006. PP 483-488.
Stout, F. and Bette, L. W. “Tree rebalancing in

[8] Optimal Time and Space," Communication Of the ACM. 29, 1986. PP 902-908.

[9] Yi-Ying Zhang, Wen-Cheng Yang, Kee-Bum Kim, Myong-Soon Park, "An AVL Tree-Based Dynamic Key Management in Hierarchical Wireless Sensor Network," International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp.298-303, 2008

[10] T.H.Cormen, C.E. Leiserson, R. L. Rivest, C. Stein, "Introduction to algorithms", second edition, McGraw-Hill publication, 2002.

Author Profile:

.

Vikas J Magar is research student in school of Computational Sciences, Solapur University, Solapur under the guidance of Dr. Rajivkumar Mente. Artificial Intelligence, Data Structure, Data Mining is his Research interest. He has participated in various national and international level conferences.

Dr. Rajivkumar Mente is working as Assistant Professor in Department of Computer Science, Solapur University, Solapur, Maharashtra, India. He is having 23 years of teaching experience. His areas of interest are Digital Image Processing, Data Structure, DBMS, Programming Languages, CBIR etc. He has published 30 research papers in various international and national journals. He has participated and presented research papers in 14 national and international conferences