Link

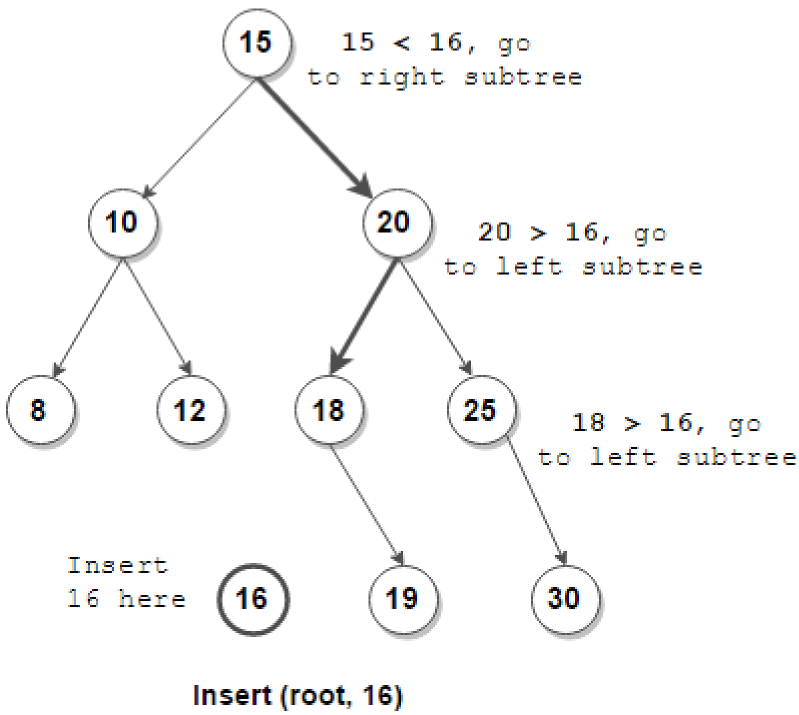
By..

Submission Date 15-Dec-2021 12:07 AM (UT+1829)
Submission ID: 4542134451
File Name-Final Project _ Analysis Document.doc (172kB)
Word Count-1269

Final Project: Analysis Document

Inserting into a BST versus a Hash Table

Data Analysis Graphs



Analysis and Summation

In computer science, a **binary tree** is a tree data structure in which each node has at most two children, which are referred to as the *left child* and the *right child*. A recursive definition using just set theory notions is that a (non-empty) binary tree is a tuple (*L*, *S*, *R*), where *L* and *R* are binary trees or the empty set and *S* is a singleton set containing the root.[1] Some authors allow the binary tree to be the empty set as well.[2]

From a graph theory perspective, binary (and K-ary) trees as defined here are arborescences.A binary tree may thus be also called a **bifurcating arborescence**[3]—a term which appears in some very old programming books,[4] before the modern computer science terminology prevailed. It is also possible to interpret a binary tree as an undirected, rather than a directed graph, in which case a binary tree is an ordered, rooted tree.[5] Some authors use **rooted binary tree** instead of *binary tree* to emphasize the fact that the tree is rooted, but as defined above, a binary tree is always rooted.[6] A binary tree is a special case of an ordered K-ary tree, where *K* is 2.

In mathematics, what is termed *binary tree* can vary significantly from author to author. Some use the definition commonly used in computer science,[7] but others define it as every non-leaf having exactly two children and don't necessarily order (as left/right) the children either.

Concrete Real-World Example

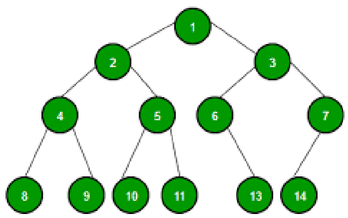
- First, as a means of accessing nodes based on some value or label associated with each node.[9] Binary trees labelled this way are used to implement binary search trees and binary heaps, and are used for efficient searching and sorting. The designation of non-root nodes as left or right child even when there is only one child present matters in some of these applications, in particular, it is significant in binary search trees.[10] However, the arrangement of particular nodes into the tree is not part of the conceptual information. For example, in a normal binary search tree the placement of nodes depends almost entirely on the order in which they were added, and can be re-arranged (for example by balancing) without changing the meaning.
- Second, as a representation of data with a relevant bifurcating structure. In such cases, the particular arrangement of nodes under and/or to the left or right of other nodes is part of the information (that is, changing it would change the meaning). Common examples occur with Huffman coding and cladograms. The everyday division of documents into chapters, sections, paragraphs, and so on is an analogous example with n-ary rather than binary trees.

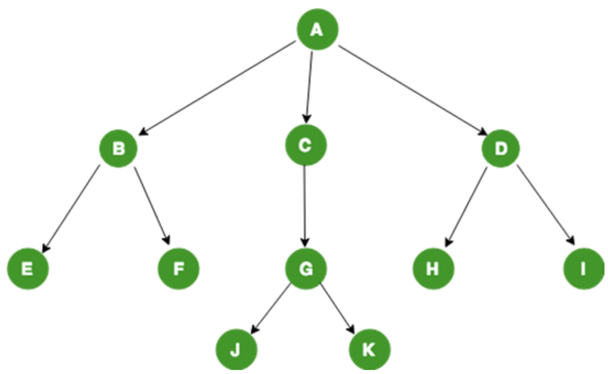
Data Structure Selection

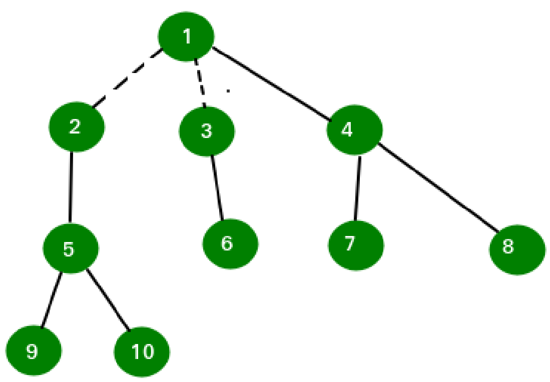
A binary tree is a **tree-type non-linear data structure** with a maximum of two children for each parent. Every node in a binary tree has a left and right reference along with the data element. The node at the top of the hierarchy of a tree is called the root node. The nodes that hold other sub-nodes are the parent nodes

Removing from a Vector at the Front versus the Back

<u>Data Analysis Graphs</u>







Analysis and Summation

To actually define a binary tree in general, we must allow for the possibility that only one of the children may be empty. An artifact, which in some textbooks is called an *extended binary tree* is needed for that purpose. An extended binary tree is thus recursively defined as:[11]

- the empty set is an extended binary tree
- if T_1 and T_2 are extended binary trees, then denote by $T_1 \bullet T_2$ the extended binary tree obtained by adding a root *r* connected to the left to T_1 and to the right to T_2[*clarification needed where did the 'r' go in the '$T_1 \bullet T_2$' symbol*] by adding edges when these sub-trees are non-empty.

Another way of imagining this construction (and understanding the terminology) is to consider instead of the empty set a different type of node—for instance square nodes if the regular ones are circles.

Concrete Real-World Example

A binary tree is a rooted tree that is also an ordered tree (a.k.a. plane tree) in which every node has at most two children. A rooted tree naturally imparts a notion of levels (distance from the root), thus for every node a notion of children may be defined as the nodes connected to it a level below. Ordering of these children (e.g., by drawing them on a plane) makes it possible to distinguish a left child from a right child.[13] But this still doesn't distinguish between a node with left but not a right child from a one with right but no left child.

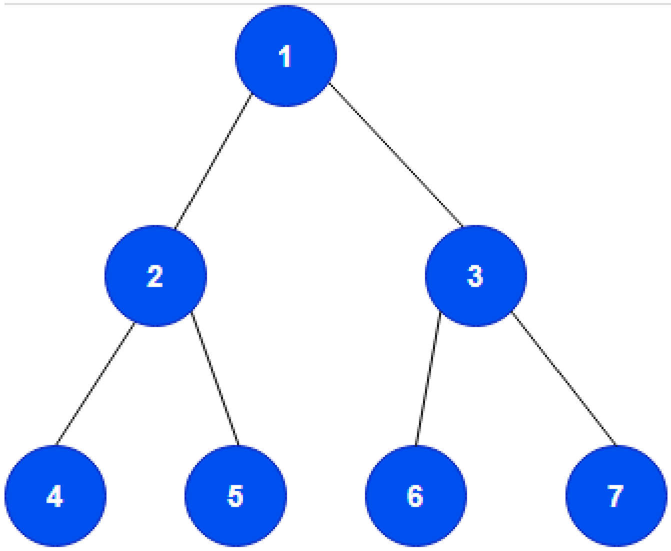
The necessary distinction can be made by first partitioning the edges, i.e., defining the binary tree as triplet (V, E_1, E_2), where $(V, E_1 \cup E_2)$ is a rooted tree (equivalently arborescence) and $E_1 \cap E_2$ is empty, and also requiring that for all $j \in \{ 1, 2 \}$ every node has at most one E_j child.[14] A more informal way of making the distinction is to say, quoting the Encyclopedia of Mathematics, that "every node has a left child, a right child, neither, or both" and to specify that these "are all different" binary trees

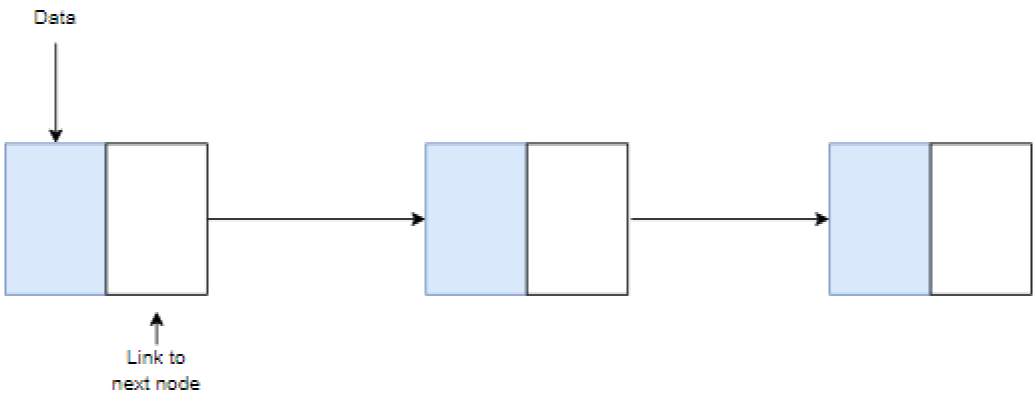
Data Structure Selection

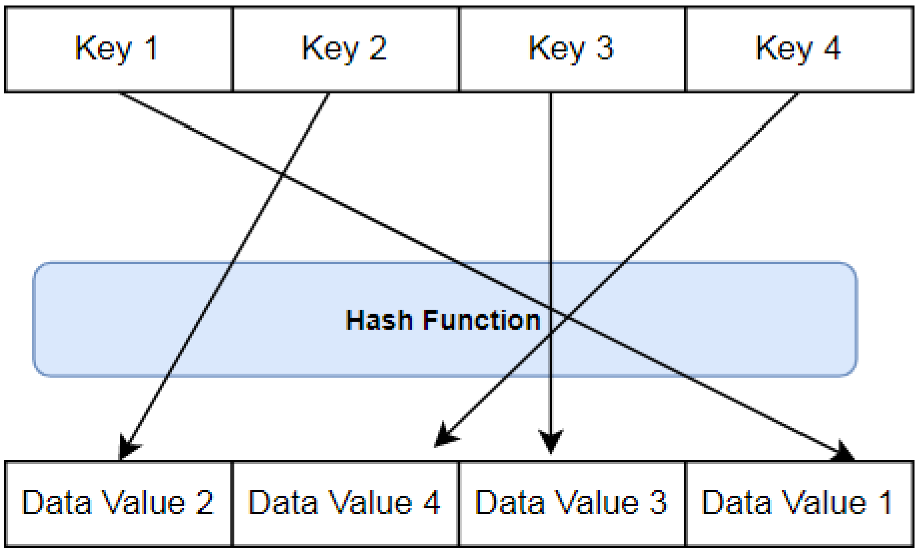
The **Tournament tree** is a complete binary tree with n external nodes and n – 1 internal nodes. The external nodes represent the players, and the internal nodes are representing the winner of the match between the two players. This tree is also known as Selection tree. There are some properties of Tournament trees

Searching a Vector, DLL, SLL, BST, and Hash Table

Data Analysis Graphs







Analysis and Summation

A binary tree is a hierarchical tree-based data structure in which each node has at most two children. The root node is the topmost node of a binary tree, while the left and right nodes are called left and right children, respectively. Furthermore, the links between nodes are known as branches, while a node without children is called a leaf node.

In a binary tree, a single node will contain a data value and a link to each of the child nodes. The following operations can be performed on binary trees: insertion, searching, and deletion. These operations can be executed in time.

Concrete Real-World Example

A few benefits to using linked lists are:

- Nodes can be easily deleted and inserted.
- A linked list can be easily implemented in most programming languages.

On the contrary, some limitations of linked lists are:

- Nodes must always be accessed sequentially, which is time consuming.
- The pointers used in linked lists require additional memory.

Data Structure Selection

Consequently, some major benefits of using hash tables are:

- Insert, delete and search operations are very fast and can be done in time.

- Hash tables can store large amounts of data.

Conversely, some limitations of using hash tables are:

- Hash functions tend to produce duplicate keys, which cause problems with storing data values, known as collisions.
- Good hash functions that produce distinct keys are expensive and difficult to implemen

proposal

ORIGINALITY REPORT

5%	3%	1%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	en.wikipedia.org Internet Source	0%
2	Submitted to Southern New Hampshire University - Continuing Education Student Paper	2%
3	cultureiq.com Internet Source	0%
4	ibrief.wagewatch.com Internet Source	3%
5	everythingwhat.com Internet Source	0%
6	www.managementstudyguide.com Internet Source	0%
7	www.researchgate.net Internet Source	0%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		