Data Structure:Theoretical Approach

Durgesh Raghuvanshi

May 19, 2022

B-Tech Department of Computer Science, IILM Academy of Higher Learning, Greater Noida, Uttar Pradesh, India
Run with accordance with significance. The first if these this paper explains about the basic terminologies used in this paper in data structure. Better running times will be other constraints, such as memory use which will be paramount. The most appropriate data structures and algorithms rather than through hacking removing a few statements by some clever coding. Data structures serve as the basis for abstract data types (ADT). "The ADT defines the logical form of the data type. The data structure implements the physical form of the data type."Different types of data structures are suited to different kinds of applications, and some are highly specialized to specific tasks. For example, relational databases commonly use B-tree indexes for data retrieval, while compiler implementations usually use hash tables to look up identifiers

1 INTRODUCTION: Data structures serve as the basis for abstract data types (ADT). "The ADT defines the logical form of the data type. The data structure implements the physical form of the data type."Different types of data structures are suited to different kinds of applications, and some are highly specialized to specific tasks. For example, relational databases commonly use B-tree indexes for data retrieval, while compiler implementations usually use hash tables to look up identifiers. Data structures provide a means to manage large amounts of data efficiently for uses such as large databases and internet indexing services. Usually, efficient data structures are key to designing efficient algorithms. Some formal design methods and programming languages emphasize data structures, rather than algorithms, as the key organizing factor in

1.1 Sequential search: When data items are stored in a collection such as a list, we say that they have a linear or sequential relationship. Each data item is stored in a position relative to the others. In Python lists, these relative positions are the index values of the individual items. Since these index values are ordered, it is possible for us to visit them in sequence. This process gives rise to our first searching technique, the sequential search. Starting at the first item in the list, we simply move from item to item, following the underlying sequential ordering until we either find what we are looking for or run out of items. If we run out of items, we have discovered that the item we were searching for was not present.

Algorithm Complexity:

cell1	cell2	cell3
cell4	cell5	cell6
cell7	cell8	cell9

search tree: It is observed that BST's worst-case performance is closest to linear search algorithms, that is (n). In real-time data, we cannot predict data pattern and their frequencies. So, a need arises to balance out the existing BST. Named after their inventor Adelson, Velski Landis, AVL trees are height balancing binary search tree. AVL tree checks the height of the left and the right sub-trees and assures that the difference is not more than 1. This difference is called the Balance Factor. Here we see that the first tree is balanced and the next two trees are not balanced In the second tree, the left subtree of C has height 2 and the right subtree has height 0, so the difference is 2. In the third tree, the right subtree of A has height 2 and the left is missing, so it is 0, and the difference is 2 again. AVL tree permits difference (balance factor) to be only 1.If the difference in the height of left and right sub-trees is more than 1, the tree is balanced using some rotation techniques. In our example, node A has become unbalanced as a node is inserted in the right subtree of A's right subtree. We perform the left rotation by making A the left-subtree of B.

2 Conclusion:

This paper covered the basics of data structures. With this we have only scratched the surface. Although we have built a good foundation to move ahead. Data Structures is not just limited to Stack, Queues, and Linked Lists but is quite a vast area. There are many more data structures which include Maps, Hash Tables, Graphs, Trees, etc. Each data structure has its own advantages and disadvantages and must be used according to the needs of the application. A computer science student at least know the basic data structures along with the operations associated with them. Many high level and object oriented programming languages like C, Java, Python come built in with many of these data structures. Therefore, it is important to know how things work under the hood. Dynamic data structures require dynamic storage allocation and reclamation. This may be accomplished by the programmer or may be done implicitly by a high-level language. It is important to understand the fundamentals of storage management because these techniques have significant impact on the behavior of programs. The basic idea is to keep a pool of memory elements that may be used to store components of dynamic data structures when needed. Allocated storage may be returned to the pool when no longer needed. In this way, it may be used and reused.

3 Reference:

1. Book of Data structures through C G. S Baluja. 2. Pieren Garry Department of computer science New York University. 3. Paul Xavier department of algorithms in c Amsterdam. 4. Surendrakumar Ahuja IItdelhi department of computer science delhi . 5. Nick jones department of data mining Australia. 6. Wikipedia sequential search.