THE ROLE OF DATA STRUCTURES IN DIFFERENT FIELDS OF COMPUTER- SCIENCE: A REVIEW

***Souyash Dutta, Aritri Chowdhury, Ishak Debnath, Riddhi Sarkar,Hritika Dey & Arjun Dutta**

Dept. of Basic Science and Humanities, I.E.M, Kolkata, India

**Email: souyash.77@gmail.com*

Abstract:

This paper deals with concise study on the applications of data-structure and algorithms associated with field of computer science. It has huge significance in the proper functioning of operating system and improvement of the throughput by resource utilization. There is an amount of essential complexity in the operations related to the data and files present in computer system. No brute-force approaches can minimize and manifest it so easily by adopting to the non-harmonized methodology of the storage and management, but it can be placed so that it doesn't get in the way and it's also easy to understand. The best such place is a data structure. Data structures have many applications in the area of system development, data base design, software coding and computer-networks. In this paper, we have presented a succinct explanation about the efficiency and application of algorithms upon different domains followed by a survey of different areas.

Keywords: *algorithm, brute-force, computer-networks, data-structures, operating system*

INTRODUCTION

The heart of a system is its data, which is entered through a system's input, and is subsequently stored in files. Data may be organized in many different ways. The logical or mathematical model of a particular organization of data is called a "Data structure". Data Structure can be defined as the group of data elements which provides an efficient way of storing and organising data in the computer so that it can be used efficiently. This particular paper focuses on the applications of various data structures in the field of computer science and system software like operating system. The most commonly performed operations on data structures include Traversing, searching, insertion, sorting, merging and deletion. Other applications include implementation of other data structures, execution of matrices and vectors, dynamic memory allocation, pointer container, control tables, evaluation of expressions, backtracking and runtime memory management. As applications are getting complexed and amount of data is increasing day by day, there arises the following problems:

Processor speed: To handle very large amount of data, high speed processing is required.

Data Search: If our application needs to search for a particular item from 106 items, it needs to traverse 106 items every time, results in slowing down the search process.

Multiple requests: If thousands of users are searching the data simultaneously on a web server, then there are the chances that a very large server can be failed during that process.

Hence in order to solve the above problems, data structures are used. Data is organized to form a data structure in such a way that all items are not required to be searched and required data can be searched instantly.

In this paper, we compose a survey beginning with the diverse applications of data structures in computer science and operating system. Section 2 focuses on the problem statements as well as solution measures mentioned above. Section 3 focuses on the functional aspects of various data structures and algorithms with relevance to their corresponding field of computer science and also to system software like operating system and their ordeal in performing its operational tasks. Finally, section 4 details the points presented in the paper in the form of conclusion. The goal of this theme is to survey the core concepts and techniques in the large subset of data-structure with its roots in computer-networks and operating systems.

DATA STRUCTURES AND ITS APPLICATIONS

Data structures are used in operational tasks of almost every program or software system. Some programming languages emphasize data structures rather than algorithms as the key organizing factor in software design. The standard categorization of data structures include: Primitive and non-primitive data structures. The following figure shows the categories of data structures and its sub divisions.

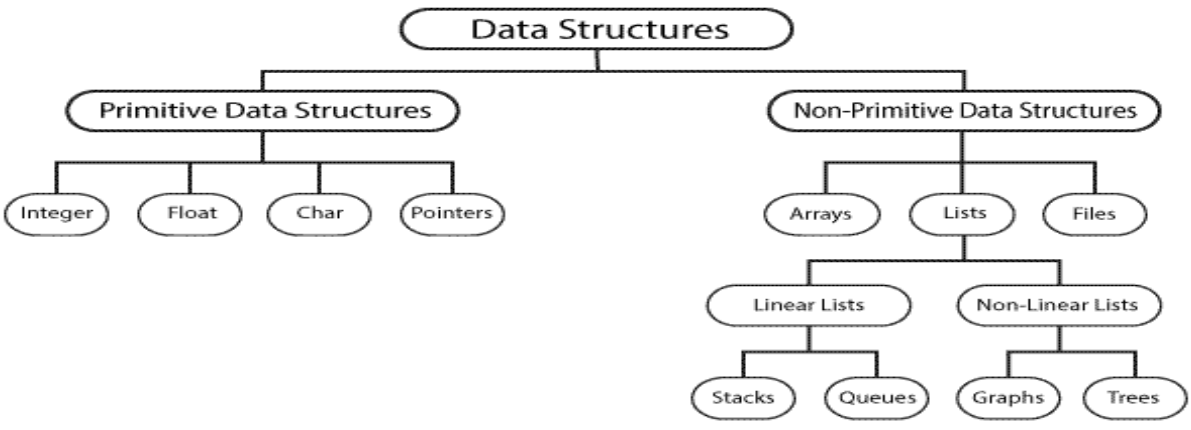

Fig.1[6]

Fig.1[6] depicts the types of data structures. There are two types of data structures primitive and non- primitive data structures. Primitive data structures can be further divided into four types: Integer, Float, Char and Pointers. While non-primitive data structures can be divided into 3 types: Array, Lists, Files. Among which the lists can be sub-divided into further two types.

A data structure is used for organizing and storing data as we mentioned in the above section. General data structure types include the array, the file, the record, the table, the tree, and so on. Any data structure is designed to organize data to suit a specific purpose so that it can be accessed and worked with in appropriate ways. Primitive data structures are the one that are directly manipulated by machine instructions. The primitive data structures include integers, float, character and pointer data. The primitive data structures define the storage structures for different types of data.

Non-primitive data structures are the one which are not directly manipulated by machine level instructions. Nonprimitive data structures are categorized into linear data structures and non-linear data structures. The relationship between linear and non-linear data structures is defined in terms of factor called principle of alignment. Some of the linear data structures include arrays, structures, unions, stacks, queues and linked lists, files etc. Linear data structures can be constructed as a continuous arrangement of data elements in the memory. It can be constructed by using array data type.

The non-linear data structures are mainly trees and graphs. Non-linear data structure can be constructed as a collection of randomly distributed set of data item joined together by using a special pointer. There are various important applications of multiple data structures associated with field of computer science which are as discussed below:

(a) ARRAYS:

Arrays are homogenous and linear data structures that systematically arrange the data objects sequentially one after another in a continuous chunk of memory. An array can hold several pieces of information of same type. Arrays are considered as building blocks of data structures. Arrays can be used to implement all the basic and advanced data structures that simplify the coding in many cases. The simplest type of data structure is a linear array, also called one-dimensional array. The size of an array cannot be increased. An array is a collection of items stored at contiguous memory locations.

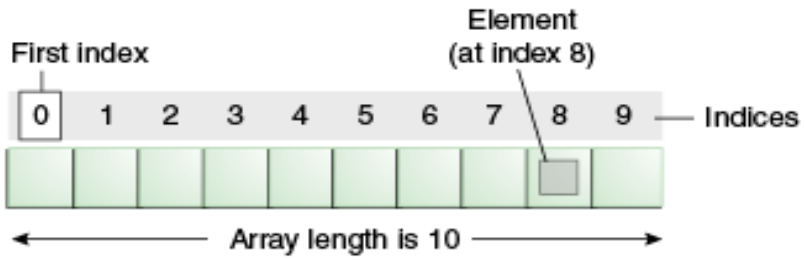

Fig.2[7]

Fig.2[7] depicts the main structure of an array. The array length is 10. The numbers from 0 to 9 are called indices. 0 is the first index and 9 is the last index. Elements in the array can be referred by their index numbers and memory address of each element in a array is also different.

Array is like a universal data structure which have its relevance towards building all other data structures. Here are some of the applications of arrays:

(i). To search and find a specified target value, such as maximum, minimum, mean, median, average, count from a group of data objects the arrays are very efficient.

(ii). Array data structure is used for arrangement of items at equally spaced and sequential addresses in computer memory makes it easier to perform operations like sorting, merging, traversal, retrievals.

(iii). Array data type, used in a programming language to specify a variable that can be indexed.

(iv). Numerical representations like matrices can be easily represented in computer's memory which enables to solve many complex mathematical problems and images processing transformations.

(b) STACKS:

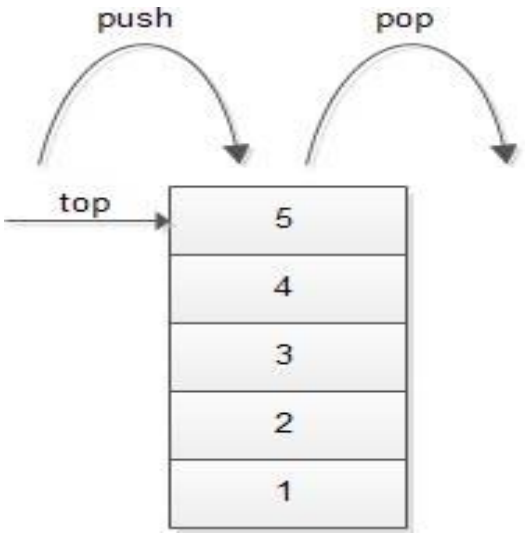

Fig.3[8]

Fig.3[8] explains the working of stack. The figure shows the operations that can be performed in the stack are push and pop. It works on the LIFO system where the element at the top position is deleted first while it has entered in the stack at the end.

Stack is a homogenous linear and recursive data structure which works on LIFO (Last In First Out) system. Insertion of an element in stack is known as PUSH operation and deletion of an element is referred to the POP operation. Elements can be inserted and removed from the top position only. To explain the working of a stack we can think of a stack of books. We can remove only the top book and add a new book at the top only. Simplest application of stack is to reverse a word.

Some of the applications of stacks in accomplishing the operating system tasks are:

(i). A stack is used for the compiler (operating system) to store local variables used inside a function block, so that they can be removed once the control comes out of the function block.

(ii) A stack can be used as an "undo" mechanism in text editors.

(iii). Stacks are useful in backtracking, which is a required when one need to access the most recent data element in a series of elements.

(iv) Stacks play an important role in parsing. For example, a compiler must parse arithmetic expressions written using infix notation:

(v). It can be used to call functions and recursion can be implemented using stack. The return values of the functions will be stored into stack and the lastly called function will first return the value.

(vi) In language processing, space for parameters and local variables is created internally using a stack. Compiler's syntax check for matching braces is implemented by using stack.

(c) QUEUES:

Queue is a linear and homogenous data structure in which insertions and deletions are done at different ends in First in first out order (FIFO). Hence, they are called FIFO lists. In queue all the elements are inserted at rear end whereas all deletions are performed at another end called as front end. General applications of queues include transport, and operations research where various entities such as data, objects, persons, or events are stored and held to be processed later. In these contexts, the queue performs the function of a buffer.

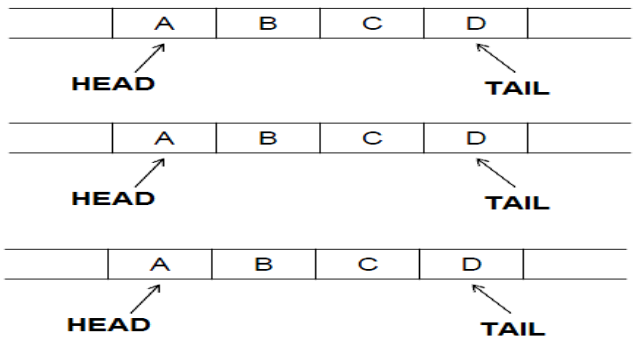

Fig.4[9]

Fig.4[9] depicts the working of queue. Queue works on FIFO system. Here insertion is done at the rear end and deletion is done at the front end. While in circular queue, both insertion and deletion can be done at both the ends using the head and tail variables.

Some important applications of queues in performing operational tasks of system include:

(i). Queues are used in programming. Common implementations are circular buffers and linked lists.

(ii). Queues can be used to store the interrupts in the operating system.

(iii). It is used to store the incoming data in a program

(iv) Queue is used to process synchronization in Operating System.

(v) Queues are used for CPU job scheduling and in disk scheduling. This data structure is involved in various features of operating system like multiprogramming platform systems and in different type of scheduling algorithms. Printer server routines and various applications software are also based on queue data structure.

Example: Round robin technique is implemented using queues. It is used especially for the time-sharing system. The circular queue is used to implement such algorithms.

(d)LINKED LISTS:

A linked list is a linear data structure consisting of a group of nodes, each node is composed of a data and a link to the next node in the sequence. A linked list organizes stored data on a storage medium based on the logical order of the data and not the physical order. The linked lists are of three types: Single, Double and Circular Linked List

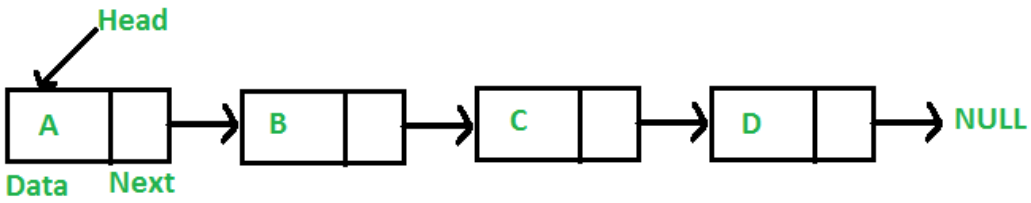

Fig.5[10]

Fig.5[10] demonstrates the structure of a single linked list. Each block in a linked list is known as a node and a node contains two parts a data and the next part. The data part stores the data

of a particular node while the next part stores the address of the next node. The head variable stores the address of the starting node. By default, the next part of the last node is null.

Linked list is most useful linear data structure. It has many applications in completing various jobs of operating system:

(i). Linked lists are used in dynamic Memory Management tasks like allocation and releasing memory at run time.

(ii). A polynomial can be represented and manipulated using linear link list to perform operations such as addition, multiplication with polynomials.

(iii) Linked lists are used in Symbol Tables for balancing parenthesis and in representing Sparse Matrix.

(iv) A linked list would be a reasonably good choice for implementing a linked list of file names, undo functionality in Photoshop.

(v). The cache in browsers allow us to hit the BACK button is implemented using a linked list of URLs.

(vi) Data structures like stack, hash table, and binary tree can be implemented using a doubly linked list.

(vii) A linked list can be used to manage the unassigned memories in the operation system.

Some common applications of linked lists include creating hash tables for collision resolution across communication channels and managing relational databases**.**

(e) TREES:

A tree data structure is a powerful tool for organizing data objects based on keys. It is equally useful for organizing multiple data objects in terms of hierarchical relationships. Tree structures make an excellent alternative to arrays, especially when the data stored within them is keyed or has internal structure that allows one element to be related to, or ``saved within" another.

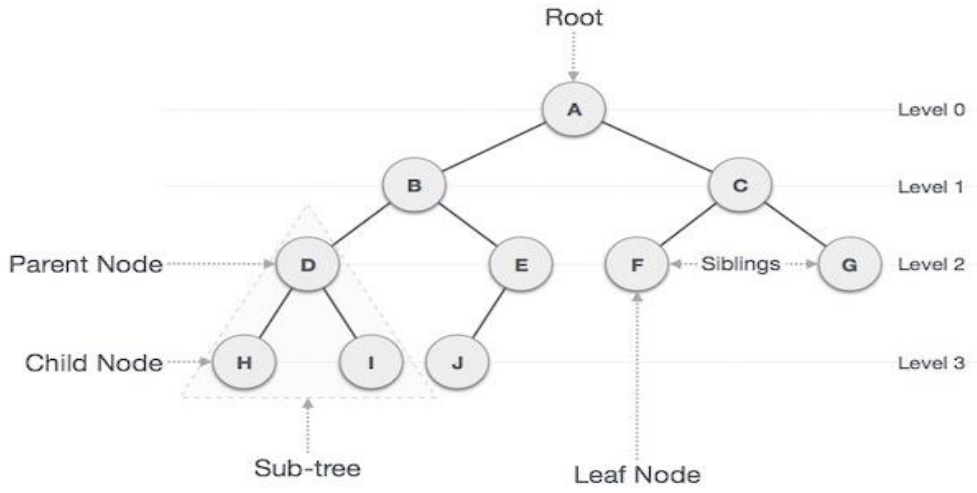

Fig.6[11]

Fig.6[7] depicts the structure of a binary tree. A tree can be divided in to many sub-trees. The last node is called the leaf node and the starting node is called the root. There are parent nodes and child nodes as well. For example, D is a parent node and H and I are its child node. H and I are also siblings. Level of a tree is the longest path between the root and the leaf note.

Some of the applications of Trees are:

(i) Trees are used to represent phrase structure of sentences, which is crucial to language processing programs. Java compiler checks the grammatical structures of Java program .

(ii) Trees are used in many search applications where data is constantly entered and deleted such as the maps and set objects of many language libraries.

(iii) An operating system maintains a disk's file system as a tree, where file folders act as tree nodes. The tree structure is used because it easily accommodates the creation and deletion of folders and files.

(iv) Tree data structures can be used for syntax validation in major compilers and as implementations of sorted dictionary.

(v) Trees are vastly used in computer networking and are used commonly in internet protocols. It can be used as every high bandwidth router for storing router-tables.

(vi) Trees can be used for searching the shortest path using Dijkstra's algorithm. They are used to find the node that are closest to the router.

(vii) Trees can be used for quick traversals and searching of directory structures in the system.

(f) GRAPHS:

A graph is a data structure of a finite set of ordered pairs, called edges or arcs, of certain entities called nodes or vertices.

A graph may be undirected or directed from one vertex to another. Many practical problems in computer science can be represented and solved by graphs. Graph theoretical ideas are highly utilized in the field of computer science. Modelling of data structures with vertices and edges to solve problems like resource allocation, scheduling, graph colouring, resource networking, database design, network topologies etc are considerably simpler and easier with graphs. Some of the practical applications of graphs in the field of computer science include.

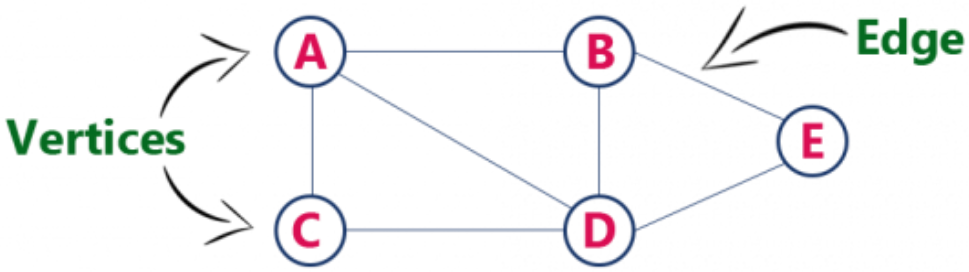

Fig.7[12]

Fig.7[12] explains the structure of graph data structure. A graph is a combination of finite sets of ordered pairs, called edges or arches, of certain entities called nodes and vertices. A graph can be undirected or directed from one vertex to another.

Applications of graphs:

1. The link structure of a website could be represented by a directed graph: the vertices are the web pages available at the website and a directed edge from page A to page B exists if and only if A contains a link to B.

2. Graph colouring concept is used in job scheduling problems of CPU, jobs are assumed as vertices of the graph and there will be an edge between two jobs that cannot be executed simultaneously.

3. Job execution between processors and sets of jobs can be simultaneously executed with the help of graphs.

   Example: A processor cannot work on two jobs simultaneously. when scheduling of file transfers occurs between processors, this problem arise. This can be modelled by considering a graph whose vertices correspond to the processes and if there is any task that has to be executed on processors i and j, then and edge to be added between the two vertices. Now the scheduling problem is to assign colours to edges in such a way that every colour appears at most once at a vertex.

4. The problem of complex time table scheduling can be dealt efficiently with graphs. A bipartite graph G where the vertices are the number of instructors say m1, m2, m3, m4, ……. mk and n number of subjects say n1, n2, n3, n4, ……. nm such that the vertices are connected by pi edges. It is presumed that at any one period each professor can teach at most one subject and that each subject can be taught by maximum one professor.

   Example: Consider a period setting the timetable for this single period corresponds to a matching of possible assignment of professors to subjects taught during that period. Hence, the solution for the time table scheduling problem will be obtained by partitioning the edges of graph G into minimum number of matching. This problem can also be solved by vertex colouring algorithm.

5. Shortest Path algorithm is the commonly used algorithm for determining the shortest path between any two points. This is practically used in applications such as network stations, computer games, maps, satellite navigation system etc.

6. Minimal Spanning Tree algorithm is used to estimate the minimal cost or cheapest way to connect any two or more point such as networks, distance between two locations in memory, distance between as two cities, telephone or electrical stations etc.

7. Optimal Graph Colouring algorithm is used to colour a map with a minimum number of colours. A vertex colouring is a labelling of the vertices of an undirected graph such that no edge has two endpoints with the same label. The colouring algorithm visits every vertex, identifying the minimal number of colours necessary to label every vertex. Example: Scheduling and resource allocation using a resource conflict graph. Vertices in a resource conflict graph represent operations to perform; edges represent pairs of operations which are in conflict because they cannot use the same resource.

8. Graphs also have its applications in searching an element in a data structure using DFS or BFS.

9. Graphs have still many more practical applications applicable to other fields and which have not discussed in the scope of this paper. Graphs have numerous applications in solving network problems.

I. Some Real Time Applications of Data Structures are:

• Determination of cities using Google maps to find population.

• Elevation and location of the data structures supports in performing operations like finding addresses on the map associated with some conditions like lookup city by name, calculates shortest path between two cities.

• To display cities within a given window, insert, delete or rename cities etc

Applications of Data Structures for Databases:

Complex data structures are essential in the design of fast algorithms for a variety of applications, including combinatorial optimization, information retrieval and Web search, databases and data mining, and geometric applications. The data structures employed in a DBMS context are B-trees, buffer trees, quad trees, R-trees, interval trees, hashing etc. Data Structures for Query Processing high-level input query expressed in a declarative language called SQL the parser scans, parses, and validates the query.

CONCLUSION:

The paper is designed in such a way that any person can gain depth knowledge on data structures and its relevance with operating systems and databases. Application of major data structures in the field of computer science have also been included. Data structures are considered as building blocks of efficient algorithms to deal with operating system functions. The importance of each data structure like memory allocation, job scheduling, establishing relationship between processes or entities and many more functions are reviewed in details. The interrelation between operating system and data structures in the field of computer science has been explained well in this paper with necessary examples.

ACKNOWLEDGEMENT:

This research paper was supported by Institute of Engineering & Management (IEM). We thank our Mathematics professor, Biswadip Basu Mallik sir who helped us immensely to make this paper successful. Moreover, I like would like to acknowledge Mr. Arjun Dutta(author) for guiding us for completing the paper.

REFERENCES:

1. “Introduction to Algorithms” by Thomas H. Corman
2. “Data structures, algorithms and applications in C++” by Satraj Sahni
3. “Data structures and algorithm” by Aho, Ullman and Hopcroft
4. Array-Wikipedia, http: //en.wikipedia.org/wiki/Array
5. Data Structures-Geeks for Geeks, www.geeksforgeeks.org/data-structures
6. Explain different types of data structures with examples, https://www.ques10.com/p/29618/explain-different-types-of-data-structures-with-ex
7. Collection Part-1:Java Arrays, https://www.helicaltech.com/java-arrays
8. C Stack using Array, https://www.zentut.com/c-tutorial/c-stack-using-array
9. Array of Queues, https://cboard.cprogramming.com/c-programming/148261-array-queues.html
10. Link list Set-1, https://www.geeksforgeeks.org/linked-list-set-1-introduction
11. Data Structures and Tree, https://www.tutorialspoint.com/data_structures_algorithms/tree_data_structure.html
12. Introduction to Graphs, http://btechsmartclass.com/data_structures/introduction-to-graphs.html