JOURNAL ON COMMUNICATIONS

ISSN:1000-436X

Scopus®

REGISTERED

www.jocs.review

A Systematic Study of Data Structure Techniques and Their Applications in Computer Science

Sunil Kumar
Assistant Professor
Department of Computer Science & Engineering
Bansal College of Engineering, Mandideep, Madhya Pradesh, India

Abstract – This paper shows the concept & application of data structure from how the computer solves the practical problems through program design, based on the analysis of the connection & difference between data structure & algorithm, the general principles followed by data structure are given, & further illustrates the practical significance of data structure application through examples. Data structure is the logical structure of data, the physical storage structure & the encapsulation of algorithm. This paper discusses how to apply data structure to solve the practical problems of non – numerical calculation from these three aspects & illustrates the application of data structure with a concrete example.

Key words: *Data structure, Program, Algorithm*

1. Introduction

Nowadays, computer is an indispensable tool for the production & life of human society. Computer is not only help us to solve many numerical problems in engineering & Scientific calculations, but also can help us to effectively solve plenty of non- numerical calculation problems such as text processing, information retrieval, data management, image processing, video processing & artificial intelligence. The mathematical models of numerical calculation are various equations & solutions, & the mathematical models of non- numerical calculation are the content of data structure application. Therefore, data structure is the foundation of computer science. From the division of disciplines, it is the core curriculum of operating system, database, compiling principle & computer networks & other kinds of professional basic course& Computer graphics, image processing & artificial intelligence & so on, which play the role of the core course in the computer course system

2. Meaning of Data Structure

Data structure is a kind of data element set which has certain logical relation, uses some kind of storage structure in computer, & encapsulates corresponding operation. Data structure contains three aspects of content, logical relationships, storage relationships & Operations. The logical structure of data can be divided into linear structure & nonlinear structure roughly. A one to one relationship exists in the data elements of linear structure, except the first & the last node; any other nod has only one direct precursor & descendant node. Linear structures include liner tables, Stacks & queues. The tree, set & graph are all nonlinear structure, in which the tree structure simulates levels, & the graph structure simulates symmetric & asymmetric relation. The research data structure in the program deigns to make the program design more robust & efficient & the development of the program more convenient.

3. Literature Survey

Data structures are one among the most fundamental concepts for a programmer. The implementation phase of any project would start off by the decision of data structure to be used. It gives an effective way for memory management and helps in analyzing the real-world problems in a much simpler way.

In [1], the author speaks about the analysis of array operations on the basis of canonical abstractions and numeric domains. The author speaks about the scalar variables and how the array can be partitioned into different elements with a particular index. In [2], Lester Leong gives a brief idea about the array and all the basic operations that can be carried out on arrays.

The author tells us about the best way of choosing the data structure for a particular problem is based on the number of steps the operations.

In [7], the author has compared stacks to a set of disks in a CD pack and queue to a real-world queue in which a new person can join only at the rear end.

According to [10], a linked list is considered to be collection of linearly arranged elements where the data elements are called as nodes. In [6] and [7], queues and stacks are compared to each

other and their applications are explained. In [8], the author throws some light on the properties of queues and their operations. It also lists some real-world applications of queues.

The basic terminologies used in trees are clearly described in [11] and [12]. In [11], Reema Thareja explains all the basic operations on trees. It also shows how trees are represented in the form of arrays and linked lists. They also tell us about the important real-world applications of trees.

In [13], Stefanie Cassiagena Navone gives a basic idea about how the graphs are visualized in this real world and the basic purpose of using this graph. She also explains the elementary types of graphs and some basic terminologies related to it. The author states that the graphs are fundamentally used to set up the connections between the different elements and also talks about how to find and analyze these elements.

4. Importance of Data Structures in Programming

Efficient data access and manipulation: Data structures enable quick access and manipulation of data. For example, an array allows constant-time access to elements using their index, while a hash table allows fast access to elements based on their key. Without data structures, programs would have to search through data sequentially, leading to slow performance.

Memory management: Data structures allow efficient use of memory by allocating and deallocating memory dynamically. For example, a linked list can dynamically allocate memory for each element as needed, rather than allocating a fixed amount of memory upfront. This helps avoid memory wastage and enables efficient memory management.

Code reusability: Data structures can be reused across different programs and projects. For example, a generic stack data structure can be used in multiple programs that require LIFO (Last-In-First-Out) functionality, without having to rewrite the same code each time.

Optimization of algorithms: Data structures help optimize algorithms by enabling efficient data access and manipulation. For example, a binary search tree allows fast searching and insertion of elements, making it ideal for implementing searching and sorting algorithms.

Scalability: Data structures enable programs to handle large amounts of data effectively. For example, a hash table can store large amounts of data while providing fast access to elements based on their key.

5. Concepts of Data Structure

A data structure is a specialized format for organizing, storing, and managing data in a computer's memory so that it can be used efficiently. It defines the relationship between data elements and the operations (like insertion, deletion, searching, and sorting) that can be performed on them.

The primary goal is to optimize a program's time and space complexity (performance and memory usage) when dealing with large amounts of data. The choice of an appropriate data structure is crucial for designing efficient algorithms and scalable software applications, from databases and operating systems to artificial intelligence and web development.

Key Concepts

- **Data Organization**: Data structures provide a systematic way to arrange data, moving beyond a simple collection of unrelated variables.
- **Efficiency**: Proper use of data structures can significantly improve a program's performance, making tasks faster and more memory-efficient.
- **Abstraction**: Data structures provide an abstract way to focus on the logical view of the data and its operations, rather than the low-level memory details.
- **Algorithms**: Data structures work hand-in-hand with algorithms (step-by-step procedures to solve problems). A data structure provides the framework, and the algorithm manipulates the data within that framework

Designing a data structure involves choosing an efficient format for organizing and storing data based on the specific operations the application will perform and the efficiency (time and space complexity) required. The right choice is crucial for program performance and scalability**.**

6. Key Steps in Data Structure Design

- **Understand the Problem:** Clearly define the problem and identify the requirements and constraints, such as the type and amount of data involved.
- **Identify Required Operations:** Determine which operations (e.g., searching, sorting, insertion, deletion, finding minimum/maximum) will be performed most frequently.
- **Evaluate Trade-offs:** Different data structures excel at different operations. Consider the time and space complexity trade-offs. For instance:
- **Hash tables** offer fast average-case lookups (O(1)) but may use more memory.
- **Sorted arrays** allow for efficient searching (O(log n)) but insertions and deletions can be slow (O(n)).
- **Linked lists** are efficient for insertions and deletions but slow for random access.
- **Choose the Appropriate Structure:** Select the data structure that best optimizes for the required operations given the constraints.
- **Implement and Optimize:** Implement the chosen structure and associated algorithms. Document the design decisions and plan for future maintenance and optimization.

7. Application Example of Data Structures and their Uses

Data structures are generally classified as linear (sequential arrangement) or nonlinear (hierarchical or interconnected arrangement).

Type	Examples	Primary Uses
Linear	Arrays, Linked lists, Stacks, Queues	Storing sequential records, managing function calls (stacks), and task scheduling (queues).
Non-Linear	Trees, Graphs, Hash	Representing hierarchical data (file systems, organization

	Tables	charts), modeling networks (social media, transportation), database indexing.

By understanding these principles and the characteristics of different data structures, you can design efficient and effective solutions to complex problems.

Conclusion

In this paper, we have explained the different types of data structure and their operations. It also clarifies depicts the implementation of each data structure with the help of diagrams. It also explains how each data structure is applied in the real world. Data structures are the best arrows to have in our quiver of computer science. They have their own advantages and disadvantages. Fortunately, the number of advantages exceeds the number of disadvantages. It is important to choose the correct type of data structure for our problem statement.

References

[1] Denis Gopan, Thomas Reps and Mooly Sagiv: “Numeric Analysis of Array Operations “ https://research.cs.wisc.edu/wpis/papers/tr1516.pdf

[2] Lester Leong: “A Guide to Arrays and Operations (Data Structures)” https://towardsdatascience.com/a-guide-to-arrays-andoperations-data-structures-f0671028ed71

3] Monika Yadav: “ARRAYS: REVIEW” http://ijirt.org/master/publishedpaper/IJIRT142671_PAPER.pdf

[4] Ahmad Shoaib Zia**: “**A Survey on Different Searching Algorithms “ https://www.irjet.net/archives/V7/i1/IRJET-V7I1275.pdf

[5] Rahul Mehta. “Research Paper on Sorting Algorithms Jariya Phongsai“. https://www.cse.iitk.ac.in/users/cs300/2014/home/~rahume/cs300A/techpaper-review/5A.pdf

[6] Nitesh, Manbir Singh, Rahul Yadav. “Research paper on stack and queue”. http://www.ijirt.org/master/publishedpaper/IJIRT101357_PAPER.pdf

[7] Venkatesh N. Baitipuli. Introduction to Data Structures Using C. 2009

[8] E Balagurusamy. Data Structures Using C. Tata McGraw-Hill Education, 2013

[9] A. Jain, U. Kumar. "research paper on queues"
http://ijirt.org/master/publishedpaper/IJIRT100828_PAPER.pdf

[10] Dr. Mahesh K Kaluti, Govindraju Y, Shashank A R, Harshith K S. "Dynamic Implementation of stacks using singly linked list"
https://www.irjet.net/archives/V5/i3/IRJET-V5I3434.pdf

[11] "Data structures using C second edition –Reema Thareja".

[12] Rubi Dhankar, Sapna Kamra, Vishal Jangra. "Tree concept in data structure"
http://ijirt.org/master/publishedpaper/IJIRT101212_PAPER.pdf

[13] Estefania Cassiagena Navone: Data Structures 101:
Graphs—A Visual Introduction for Beginners.