**Comparative Analysis of Linked-List and**

**Array Data Structures**

*Kabeer Ahmed1, Abdul Moiz2, Nizam Ali3, Muhammad Muzamil Hussain4, Muhammad Taha Raees5*

Department of Software Engineering

NED University of Engineering and Technology, Karachi, Pakistan

1. **ABSTRACT**

***The Data Structures Array and Linked list are wont to preserve the info in organized and disorganized manner respectively helping us to seek out the memory location of the weather where we've found the time complexity of the both and every arrangement have some built-in functions. We have compared here the array and linked list wherein we have gone through their types, Advantages, Disadvantages, Implementation, Insertion, Deletion, Time and space Complexity, Operations and Applications of Linked list And Array respectively here are some defined Algorithm of insertion and deletion too.***

***Keywords—Data structure, Array, Linked-list, Traverse, Insertion, Deletion ,Comparison***

1. **INTRODUCTION**

There are many data structures, one of them is a linked list. Where data not stored sequentially inside the computer memory but they are linked with each other by the help of address.It is usually utilized as a linear data structure which comprises groups of nodes in an arrangement. It is a collection of data elements called nodes. Each node has two sections, one contains data and the other has the address of the next element.

A collection of homogeneous elements is called An Array. Arrays are always stored in consecutive memory locations. This makes it simpler to figure the situation of every component by just adding an offset to a base value. It can put away various values which can be referred to by a single name.

Array is an assortment of homogenous data type components while the linked list is an assortment of unordered connected elements known as nodes.

In array the elements are stored in continuous memory location but in linked list the element can be stored anywhere in the memory.

Array worked static data structure whereas linked list worked with dynamic data structure.

Array elements are independent to each other but linked list elements are dependent to each other.

Array takes more time whereas linked list takes less time.

1. **LITERATURE REVIEW**

**3.1** Searching an element in any data structure is need of todays computing world so the searching algorithms have importance in the programming world. For most of the Software searching of required to search object in the huge database so the application which has the massive database should also require implementations of searching algorithms at many platforms Like Phyton, C++ and java. In this there is comparison of binary search and linear search algorithms it is compared on the basis of the different data structures that is Array and Linked List. By implementing three different algorithms it concluded that binary search is not efficient than the linear search. Hashing is another technique used for the search an element in different data structures.[1]

**3.2** Linked lists are the linear data structures that allocate memory separately for each element unlike an array that allocates memory for every one of its components lumped together. The components in the linked list are connected or associated through pointers every node has two hubs/nodes, one for its information and other for address of next component/element, that eventually joins two nodes. This is the ultimate feature of linked lists that saves a computer's memory. There are some built-in functions in many programming languages that can be applied on linked lists. The author introduces three steps for link formation that are allocate, link next and link head. In the allocation step, malloc() function is utilized and it allocates new node and sets its data wherever

needed. The link next step has same functionality as of pointer that assigns the new node to previous one. And after that link head step directs head pointer to recently build node to make it head node. There are some methods to demonstrate techniques mentioned above. AppendNode(), CopyList() are the some of the functions.[2]

**3.3** Data processing and storage is main slice of the number of Applications. Normal Data processing and storage use either full Array structure or full linked list for storing data, whereas, Array takes large slice of memory and linked list gives not so fast processing. So usual methods have not capability of enhancement of memory consumption and time efficiency. Whereas present System provides efficient and fast way of storing data by establishing integrated array and linked list-based structure where data are stored by Delta mechanism method and that way make us to form the location where the data is stored. In the result advantage of this is reduction in memory consumption.[3]

**3.4** NumPy is basic array programming library for python language.it provide us the most useful, easy to read syntax for operating manipulating and accessing the data in matrices, vectors, and multidimensional arrays .it plays major role in research analysis of pipelines in arenas unlike as physics, astronomy, psychology.. The entire python universe is constructed on the foundation of NumPy. NumPy is the foundation of scientific Python network**.** NumPy progressively plays the important role of an interoperability level between these new array reckoning libraries.[4]

**3.5** Linked list is a linear data structure. The linked list code uses the classic singly linked list structure**.** In the list the first node is a single head pointer. Each contains a single. next pointer to the next node. A null head pointer is known as the empty list**.** Allocate all the nodes in the heap**.** For some problems,solutions is presenting the dummy node variation temporary,but mostly code deals with linked lists in their pain from. In the next, {} are used to describe lists. The list is containing the numbers 1, 2, 3 and written as {1, 2, 3}. In a few places, Returns the number of nodes in the list.[5]

1. **COMPARISON**

**4.1 Linked-List**

A linear Data Structure, linked list is the Collection of different data elements which is not stored in neighboring memory Locations. It contains the nodes which have data and next pair, data stores data element and next stores the reference of next node’s data knows as Pointer.[6]

**4.1.1 Types of Linked-List**

***Single Linked List:*** The normal linked list is also called Single Linked List, which is the collection of data elements in the linear way alike the Array.

**DATA**

**NEXT**

**NONE**

**START**

**A**

**C**

**B**

***Circular Linked List*:** The linked list that forms the circle by connecting all the nodes. In this linked list the last pointer stores the reference of the first data element so that the cycle is continued until it is stopped.

**START**

**A**

**C**

**B**

***Doubly Linked List*:** The linked list that contain one more pointer known as previous pointer. So, the linked list contains two pointers in a node, first pointer stores the reference of the data of next node and second pointer stores the reference of the data of previous node.

**NONE**

**NONE**

**PREV**

**NEXT**

**START**

**A**

**C**

**B**

**4.1.2 Basic Operation in Linked List**

***Traverse*:** All elements of linked list is displayed one by one.

***Insertion*:** Add a node at beginning or between two node or at the End of the linked list.

***Deletion*:** Delete a node of the linked list

**4.1.3 Implementation of Algorithms:**

Some of the algorithms of linked list are implemented in python is given below;

**1. Traverse**

Traverse is defined as the visiting or touching the data element in the linked list or in any data structure and it also display all the elements of the linked list separately.

**Code:**

*class node:*

*def \_\_init\_\_(self,data):*

*self.data=data*

*self.next=None;*

*class LinkedList:*

*def \_\_init\_\_(self):*

*self.start=None;*

*def traVerse(self):*

*if self.start== None:*

*print("Empty Linked list")*

*else:*

*tr=self.start*

*while tr!=None:*

*print(tr.data,end="")*

*tr=tr.next*

*list1=LinkedList()*

*list1.traVerse()*

**2. Deletion**

Deletion is defined as the rearranging of the elements after removing the elements already present in the linked list or in any other data structure.

**Code:**

**class** node:

*def \_\_init\_\_(self,data):*

*self.data=data*

*self.next=None;*

*class LinkedList:*

*def \_\_init\_\_(self):*

*self.start=None;*

*def Delete(self):*

*if self.start==None:*

*print("Empty Linked list")*

*else:*

*self.start=self.start.next*

*list1=LinkedList()*

*list1.Delete()*

**3. Insertion**

Insertion is defined as the adding of data elements in the start or in between or in the End of the linked list or in any other data structure.

**Code:**

*class node:*

*def \_\_init\_\_(self,data):*

*self.data=data*

*self.next=None;*

*class LinkedList:*

*def \_\_init\_\_(self):*

*self.start=None;*

*def insert(self,value):*

*Nnode=node(value)*

*if self.start==None:*

*self.start=Nnode*

*else:*

*tr=self.start*

*while tr.next!=None:*

*tr=tr.next*

*tr.next=Nnode*

*list1=LinkedList()*

*list1.insert(11)*

**4.1.4 Time Complexity:**

In linked list to access and search the element we have to traverse node to node till the End. So, the Time complexity of Access and Search is O(n).

In insertion the prev node is re-assign to next node in the linked list to insert the new element in the linked list. So, the time Complexity of Insertion is O(1).

In Deletion the prev node is re-assign to the next node in the linked list to delete an element in the linked list. So, the time Complexity of Deletion is O(1).

“Worse case” is the same as the above case.

**4.1.5 Space Complexity**

In linked list the space complexity is O(n) for single as well as for doubly linked list so that Doubly linked list has the advantage over single which is that the traverse is performed in both forwards as well as backwards direction but the memory is same as the single use.

The Space complexity of linked list is O(n).

**4.1.6 Applications:**

***Computer World*:** Linked list is used in the Stacks and queues implementation, and in the implementation of the advance data structures (Fibonacci Heap). This is also used in free blocks, preforming Arithmetic operations on large integers, maintaining Directories and

Sparse Matrices representation.

***Real World*:** Linked list is used in Image Viewer to next and pervious the image, Music player to next and pervious the song and also web browser to next and previous the web page.

**4.1.7 Advantages**

They are strong data structure can grow or shorten during program implementation.

Effective memory consumption: memory is not pre assigned where it is required and unallocated when no coating is needed.

Insertion and deletion are unchallenging and well organized: provides adaptability in insertion of data at specified place and deletion from stated position.

**4.1.8 Disadvantages**

Supplementary Memory: If more the fields more memory will be required.

Entrance to a random data item is a bit awkward and also time absorbing.

Difficult to sort the elements of the linked list.

**4.2 Array**

An array is the simplest data structure; it is a sequent collection of elements of similar data type and it stores a fixed number of data items in a consecutive memory location. Indexes are used to assess their data Items in array. The index range of an array of size N starts from 0 to N-1.

Index

10 20 30 40 50 60

0 1 2 3 4 5

Value

Array length = 6

**4.2.1 Types of Array**

The Followings are the Types of Array

1- One dimensional array

2- Multi-dimensional array

1- One dimensional array

A one-dimensional array is a sequential collection of elements (often called array elements) that can be accessed specifically by specifying the position of the elements with their index values.

2- Multi-dimensional array

A multidimensional array has multiple indexes of each element in the array. The most generally used multidimensional array is the two-dimensional array or 2-D Array, it is also known as a table or matrix. A two-dimensional array has two indexes of Each element in the array. 2D array is identify by the notation (row, column) as shown in figure

|  |  |  |
| --- | --- | --- |
| **(0,0)** | **(0,1)** | **(0,2)** |
| **(1,0)** | **(1,1)** | **(1,2)** |
| **(2,0)** | **(2,1)** | **(2,2)** |

**4.2.2 Basic Operations in Array**

***Traverse*** − it will show all the elements in array one by one

***Insertion*** − it will add element in array of given Index

***Deletion*** − An element is deleted at the given index.

**4.2.3 Implementation of Algorithms:**

Some of the algorithms of Array are implemented in python is given below;

**1. Traverse**

Traversal is completed by starting with the primary element of the array and getting to the last. Traversal operation is used in accessing or printing the elements stored in arrays.

**Code:**

*import array as arr*

*y=arr.array("i",([1,2,36,5,8,56)]))*

*for i in range(6):*

*y[i]=v*

*print(v)*

**2. Deletion**

Removing an element of an array and re-arranging all elements of an array is called the Deletion operation.

**Code:**

*import array as arr*

*y=arr.array("i",([22,26,25,25,6,26]))*

*n=len(y)*

*j=n-1*

*pos=2*

*item=7*

*while j<pos:*

*y[j-1]= y[j]*

*j=j+1*

*y[pos]=item*

*n=n-1*

**3. Insertion**

Insert operation is use to put more data items in array. This new data item can be placed on any index of the array based on user need

**Code:**

*import array as arr*

*y=arr.array("i",([22,26,25,25,6,26]))*

*n=len(y)*

*j=n-1*

*pos=2*

*item=7*

*while j>=pos:*

*y[j+1]= y[j]*

*j=j-1*

*y[pos]=item*

*n=n+1*

**4.2.4 Time Complexity**

Time complexity means the time required to run a program till the end The fixed size of an array does not grow as more elements are required. The calculated time complexity for accessing or writing an array is O (1) no matter the number of elements present in array. The time complexity for insertion of an element at the start of array is O(N) and for deletion is O(N-1)

**4.2.5 Space Complexity**

Space complexity means amount of memory required for an algorithm.in case of array the space complexity may beO (1)orO(N) depending on the situation if array is created of size N then its space complexity if O (1)if the size of an array grows with the input then the space complexity will be O(N).

**4.2.6 Applications**

Arrays are implemented in database records.

Arrays can be used for CPU planning.

Arrays can be used for sorting purpose. because we can easily store and sort the item in array We can implement Adjacency list implementation of graph uses vectors using arrays.

Data structures such as map, heap, balanced binary trees and set use binary search tree and which uses are implemented using arrays.

Arrays are used to preserve multiple variables with the same name.it helps to maintain large data

stack and queues are also implemented by array

We can use array to implementvectors and lists.

A real time application of array is mobile phone which simply store all the cells in array

**4.2.7 Advantages**

We can read or access the array element in a very simple and efficient way. Because the time complexity of arrays is O(1) in both cases, best and worst. this is the reason we can easily print or read any element in array using their index

Array is a base of other data structures. For instance, other data structures like Linked List, Stack, Queue etc. are implemented using array.

Many data items of the same type can be stored in a single array. Array assigns memory in adjacent memory locations for elements.

**4.2.8 Disadvantages**

when we are using array we have to decide about the size of array in start,if we are not aware of that how many item we are going to place in array,later on it would be difficult task for adding more elements.

if we want to add more items in the array it can not be done because of the fixed size of the array.if we put less number of items than the declared size of array then the remaining owed space is wasted

Time complexity increase in insertion and deletion

***Big O Comparison of Arrays and Linked Lists***

|  |  |  |
| --- | --- | --- |
| **Operations** | **Array** | **Linked-list** |
| **Traverse** | **O (1)** | **O (N)** |
| **Insertion** | **O (N)** | **O (1)** |
| **Deletion** | **O (N)** | **O (1)** |

**CONCLUSION**

Both the data structures are available in many different programming languages and are used according to their needs. Each data structures have some built-in functions too. On looking up the pros and cons of both linear data structures, we have reached to conclusion by analyzing their properties as well. Arrays are efficiently faster than linked list if we access element by giving index and also it has fixed size. Arrays utilize the computer’s memory efficiently as they consume continuous memory locations while linked lists don’t. Accessing elements in arrays is much easier and faster as it’s time complexity is O(1) while linked list has O(N). Searching operation’s cost same time for both the data structures as it’s is O(N). Comparing insertion and deletion operation it is more time consuming and complicated because if we have to insert element at last of array it is costly due to traversing through whole the array, hence it’s O(N) while in linked list it is O(1) because we have to just make reference to first node of linked list. Well the above discussion of big O notations is based on average cases respectively. Finally, arrays are simpler to use as they consume less memory, are faster, on the other hand, linked lists have ability to shrink and extend and they are considered as strong data structures and they also consume memory effectively, they are worth adaptable.

**6. REFERENCE**

[1] V. P.Parmar and C. Kumbharana, “Comparing Linear Search and Binary Search Algorithms to Search an Element from a Linear List Implemented through Static Array, Dynamic Array and Linked List,” *Int. J. Comput. Appl.*, vol. 121, no. 3, pp. 13–17, 2015, doi: 10.5120/21519-4495.

[2] [6] B. N. Parlante, “Stantford C library,” *Rev. Lit. Arts Am.*, 2002.

[3] Patent Application Publication, "Systems and methods for Storing data in an Integrated Array and Linked List ", pn US 2020/0133942 A1, 2020

[4] C. R. Harris *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, 2020, doi: 10.1038/s41586-020-2649-2.

[5] B. N. Parlante, “Linked List Problems,” *Rev. Lit. Arts Am.*, 2001

**Biography**

**Kabeer Ahmed (4.1, 3.1)**

Kabeer Ahmed is an undergraduate student of Software Engineering from NED University of Engineering and Technology, Karachi. He is also a beginner Web developer and further he is interested in the field of Cybersecurity.

**Muhammad Muzamil Hussain (5, 3.2)**

Muhammad Muzamil Hussain hailing from Lakhi Ghulam Shah and he is studying software engineering from NED University. He has interest in tech, writing and traveling and would like to pursue a career in cybersecurity.

**Nizam Ali (1, 3.3)**

My name is Nizam Ali from Ranipur District Khairpur Mir’s Sindh Pakistan now studying in NED University of Engineering & Technology. In future INSHAALLAH i’ll be a Professional Developer.

**Abdul Moiz (4.2, 3.4)**

My name is Abdul Moiz. I am a student of software engineering in NEDUET.I am interested in becoming a professional developer specially in web and mobile applications.

**Muhammad Taha Raees (2, 3.5)**

Muhammad Taha Raees is from Hyderabad and he is

studying software engineering from NED University and this is his third semester and so far, mine experience is university life is very tuff but as well as interesting. he learned a lot here and he want to become a web developer in future.