International Journal of Recent Trends in Engineering & Research
ISSN : 2455-1457 Impact Factor : 3.344

Priority Based Sorting Algorithm

Jignesh Viradiya
PG Researcher,GTU

Abstract: There are many operations performed in Data Structures. One of the most important operation of Data Structure is sorting that makes us to search, arrange or locate the record easily and efficiently. There are many sorting algorithms having different-different efficiencies based on the number of inputs. Sorting can also be performed on priority basis which displays required record first and later on remaining records. But we don't get any changes in complexity. If we give one attribute value, it will take less time. That means the efficiency of searching a record from sorted records increases when we sort on the basis of priorities and attribute's value. If we implement this process, we can increase our main-power performance as well as tool or application because it will take less time.
Keywords: Sorting, Priority Queue, Complexity

1. INTRODUCTION

One of the operation in data structure which plays a foremost role in processing of data is "Sorting" which organizes the records in some logical order, i.e. either in ascending order or in descending order [1]. We can't say that one sorting algorithm is better or worse than other sorting algorithm because each sorting algorithm has their own pros and cons. For example, if we want to sort small list of records, then bubble sort is the efficient algorithm whereas if want to sort long list of records, then quick sort is the efficient algorithm.

The main advantage of sorting application is that it requires less time to search for a particular record compared to unsorted list of records [2]. There have been many attempts made to analyze and reduce the complexity of various algorithm. Also improved sorting algorithms have been proposed to get better efficiency in terms of time and memory. The performance of each sorting algorithm is based on the data being sorted and the machine used for sorting [3].

In general, any sorting algorithm performs two operations: first compare the two elements and second swap according to desired order of a user.These two operations proceed over and over until the entire list of record is sorted [4].

2. PROBLEM STATEMENT

The sorting of records can be done on the basis of priority which we want to display the record first and then remaining list of records. But the problem arises here is that they are displaying all the records on the basis of priorities, which consumes same amount of time. We are proposing an algorithm which will display required records only as par user's choice on the basis of priorities and attributes. So, that we can save time.

2.1 Objective

The main objectives of the proposed algorithms are:

- Propose priority based sorting algorithm.
- Apply priority based sorting on multiple attributes of records.
- Save time on the basis of number of operations.

3. PROPOSED ALGORITHM

Step 1: Insert all the Records and assign priority to each Record.
Step 2: Sort the records on the basis of attribute values.
Step 3: Display sorted records of step 2.
Step 4: Exit.

4. THEORETICAL AND EXPERIMENTAL WORK

4.1 Priority Queue:

Priority queues is an abstract data type in data structure. Unlike the simple queue that inserts and removes records in a fixed order (First-In-First-Out), priority queue is assigned a priority for each records represented by an integer value so that the record can be removed from queue on the basis of highest priority represented by the minimal integer value assigned.

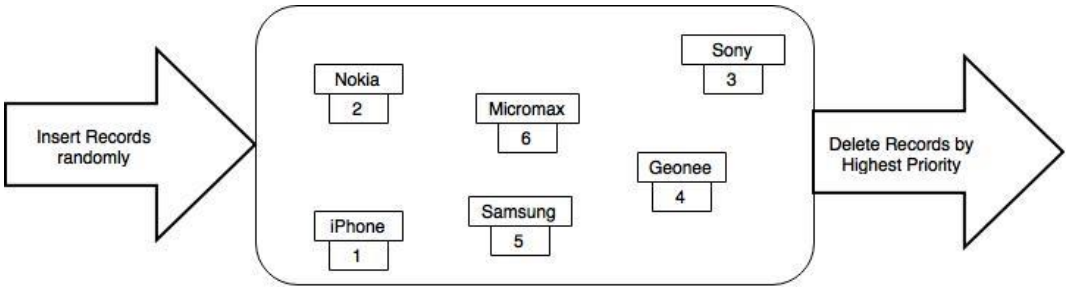

Fig 1 Priority Queue

Characteristic operations:
1) Insert
2) Find and remove the largest (or smallest) item (DeleteMax or DeleteMin).

Algorithm:
Input: a collection S storing n elements
Output: the collection S sorted
P = new PQueue()
while !S.isEmpty() do
e = S.removeFirst()
P.insert(e)
while !P.isEmpty()
e = P.removeMin()
S.addLast(e)

4.2 Insertion Sort Algorithm:

Insertion sort is one of the simple and efficient comparison methods to sort a list of records. In this algorithm, each iteration removes a record from the input list of records and inserts it into the correct position in the list being sorted. The choice of the record being removed from the input is random and this process is continued until all input records have been gone through. Both average and worst-case time is O(n2) [11].
Algorithm:
INSERTION-SORT(A)

for j ←2 to length[A]
do key ← A[j]
Insert A[j] into the sorted sequence A[1 j- 1].
i ← j- 1
while i> 0 and A[i] > key
do A[i+ 1] ← A[i]
i ← i- 1
A[i+ 1] ← key

4.3 Test cases to sort record on the basis of priorities (No attribute value):
Input Values:
Suppose we have a record of 5 employees of different designation and experience who work in a University.

Emp Name	Dept	Desig.	Priority
Rohit	CP	Prof.	1
Lopa	ECE	Ass.Prof.	3
Shankar	EP	Ass.Prof.	3
Rupal	CP	Assoc.Prof	2
Sunny	EP	Prof.	1

Table 1: Input Values

Expected Output:
Suppose, we want to display required data first (highest priority) and then remaining records (lower priority) of employee.

Emp Name	Dept	Desig.	Priority
Rohit	CP	Prof.	1
Sunny	EP	Prof.	1
Rupal	CP	Assoc.Prof	2
Lopa	ECE	Ass.Prof.	3
Shankar	EP	Ass.Prof.	3

Table 2: Expected Output for Priority based Records

4.4 : Test cases to sort records on the basis required data:
Input Values:
Consider input value of table 1.
Expected Output on the basis of priority (For One Attribute Value):
Suppose, we want to display required data only on the basis of any one attribute value, and suppose we consider “Exp” as one attribute and we want to display records of only those employees whose “Exp” is 12 to be in order on the basis of priority.

Emp Name	Dept	Desig.	Priority
Rohit	CP	Prof.	1
Shankar	EE	Ass.Prof.	3

Table 3: Expected Output for Priority based Records

4.5 Test cases to sort records on the basis required data (Multiple attributes):
Input Values:
Consider input value of table 1.1.
Expected Output:
Suppose, we want to display required data only on the basis of multiple attribute values, and suppose we consider "Desig" and"Exp" as two attributes and we want to display records of only those employees whose "Desig" is Prof and "Exp" is 10 to be in order on the basis of priority.

Emp Name	Dept	Desig.	Priority
Sunny	EE	Prof.	1

Table 4: Expected Output for Priority based Records

5. RESULTS AND ANALYSIS

5.1 Results
Given Input:
In the input, we are giving details of 6 different employees who works in University.

Name	Department	Designation	Priority
Deeksha	Computer Science	Assistant Professor	3
Swikar	Civil	Professor	1
Sirin	Chemical	Associate Professor	2
Shresht	Computer Science	Professor	1
Vidhi	Electrical	Associate Professor	2
Rohit	Electrical	Associate Professor	2

Table 5: Input Records

Output for one attribute value on the basis of priority:
Now, suppose we want to display only those records whose experience is 12 and sort them on the basis of priority. It will display sorted records priority-wise like professor first, then associate professor and at last assistant professor having experience 12 years as shown in fig 1.2, where attribute is experience only.

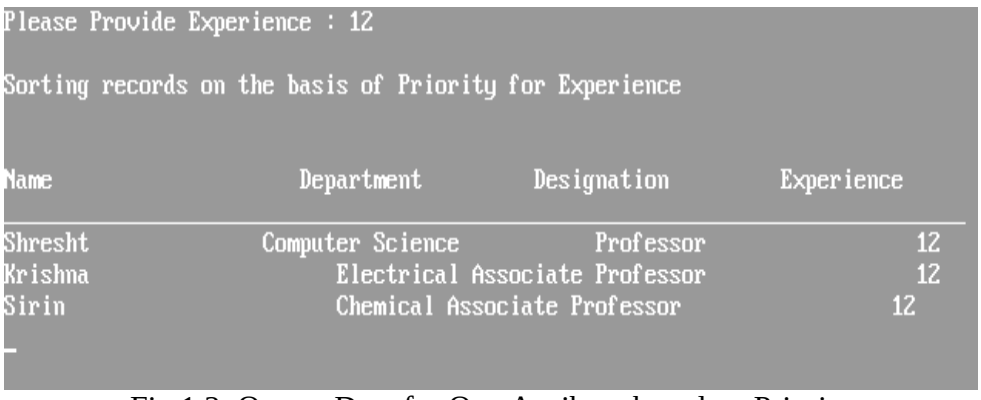

Fig 1.2: Output Data for One Attribute based on Priority

Output for two attribute values on the basis of priority:

Suppose, we want to display only Associate professors whose experience is 12 and display them in sorted form. It will display sorted records of associate professor of all departments having experience 12 years name-wise as shown below in fig 1.3, where two attributes are designation and experience.

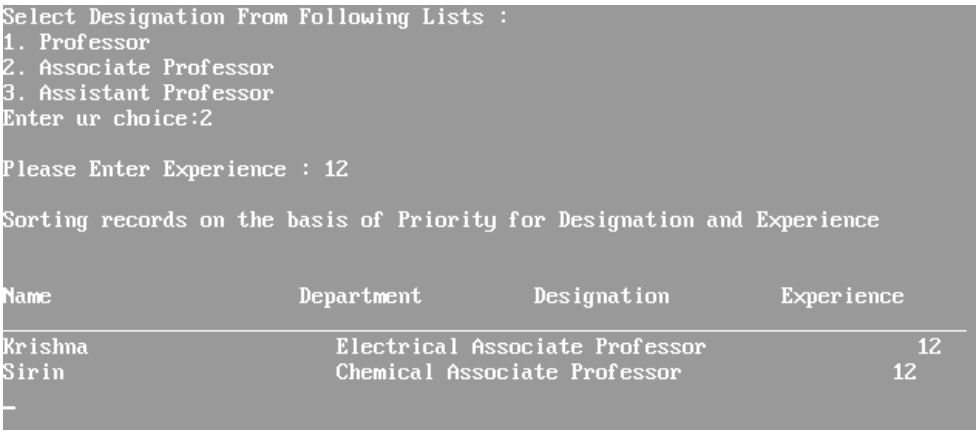

Fig 1.3: Output Data for One Attribute based on Priority

6. CONCLUSION

The researchers always try to reduce the time complexity by proposing new methods. So we have implemented proposed sorting algorithm which display only required records on the basis of priority for multiple attributes to save time by not displaying all records. So, we are saving time in terms of sorting as well as searching operations.

REFERENCES

1. Udit Agarwal, *"Data Structure Using C"*, S.K. Kataria & Sons Publications, July 2011
2. M.S. Garai Canaan.C and M. Daya, "*Popular sorting algorithms*", World Applied Programming, 1:62–71.
3. A.D. Mishra and D. Garg, "*Selection of Best Sorting Algorithm*",
4. Pankaj Sareen, IJARCSSE, 3:522–532.
5. Aditya Dev Mishra, *"Selection of Best Sorting Algorithm for a Particular Problem"*, 2009
6. Tim Bell, Bengt Aspvall, "*Sorting Algorithms as Special Cases of a Priority Queue Sort*", SIGCP'11 [8] "*A Comparative Analysis of Sorting Algorithms on Integer and Character Arrays*", The International Journal Of Engineering And Science (IJES) |
7. Ghiath Al Aqel, *"A New Priority-Sort Based Optimization Algorithm Integrated Process Planning and Scheduling"*, International Journal of Modeling and Optimization, Vol. 3, No. 2, April 2013.
8. Jehad Hammad, *"A Comparative Study between Various Sorting Algorithms",* IJCSNS International Journal of Computer Science and Network Security, VOL.1 5 No.3, March 2015.
9. Thomas H. Cormen, *"Introduction to Algorithms"*,

