Journey of Sorting technique based on Datastructure

[1]Nilesh Kumar Janghel, [2]Pavan Kumar R, [3]Shabaz Ahmed Dar, [4]Sounak Mukherjee, [5]Tenzin Choedhen

[1,2,3,4,5]Dept. of Computer Applications, Dayananda Sagar College of Arts, Science & Commerce, Bangalore

Abstract—In software industry where sorting is essential part, It makes faster and easier to locate items in a sorted list than unsorted. Sorting algorithms can be used in a program to sort an array for later writing and searching out to an ordered report or file. In this research paper, we are trying to focus on the concept of sorting techniques based on data structures.

Keywords— ADT; Bubblesort; Hashing; Monotonous; Replicate; SAN Storage ;

I. APPLICATION OF DATA STRUCTURES

Data structures can implement one or more abstract data types (ADT), which stipulate the operations that can be performed on the computational complexity of those operations and data structure. In contrast, a data structure is a concrete implementation of the specification assigned by an ADT. Different kinds of data structures are matched with different kinds of applications, and some are highly specialized to tasks. For example, relational databases typically use B-tree indexes for retrieval of data, while compiler implementations usually use hash tables to look up identifiers [4]. Data structures provide a means to achieve huge amounts of data efficiently for uses such as internet indexing services and large databases. Generally, efficient data structures are key to designing efficient algorithms. Some formal design methods and programming languages emphasize data structures, instead of algorithms, as the key organizing factor in software design. Data structures can be used to establish the storage and retrieval of information stored in both secondary memory and main memory.

II. NEED FOR SORTING

We will search the dictionary, perhaps in the middle of the pages. As 'M' is practically in the middle of the dictionary. Then we will look up at any word, let's say we found the word 'Modest' in that page, then we will look in the previous page, because 'R' comes before 'S' and accordingly in some other 2/3 steps, I will find the word 'Modern'[4]. So, that's it, generally anyone in the world will do that. Because all of us know that the words are organized in some fashion in the dictionary [10]. We basically call this method as 'Sorted'. Now, let me put some overhead on the same problem. Just think that, the dictionary is not sorted at all and may be the dictionary has the following order of words, Good, Place, Algorithm, Program, Zebra, Work, Bat, Cow, Ball, Billiards, Efficient, Alphabet, Access, Sort, Order, Monotonous, Migration, Immigration, External, Pouch. Well, that is what happens at a first glimpse, but if you think silently, you will understand that you can find the meaning of any word from this dictionary too. Only thing you got to do is, you must check all the words, and until you get the word you are searching for in the dictionary. Wait, that's basically none of use [4]. If I cannot find a meaning when I need it, what is the purpose of finding it out? Rather I would buy a new dictionary instead of using an inefficient dictionary. Yes that is what we usually want, a quicker way to make things better and get the essential element at ease [13]. That's where the exquisiteness of sorting lies. Generally, sorting helps in following two ways,

1. Easing the process of viewing up an element from the pile

2. Boosting the method of merging sequences

So, in today's world data has become enormous, so sorting basic block of data building. Sorting helps this process in an efficient way. So, to cope up with the huge data collection, we need efficient sorting process

III. ALGORITHM OF BUBBLE SORT

Bubble sort algorithm starts by comparison of the first two elements of an array and swapping if required, i.e., if you want to sort the elements of an array in ascending order and if the first element is greater than the second then, you need to swap the elements but, if the first element is smaller than second, you mustn't swap the element. Then, again second and third elements are equated and swapped if it is required and this process go on until last and second last element is swapped and compared [4]. This concludes the first step of bubble sort. If there are n elements to be sorted then, the method mentioned above should be recurring n-1 times to get mandatory result. But, for enhanced performance, in second step, last and second last elements are not compared because; the proper element is placed at last after first step automatically. Likewise, in third step, last

and second last and second last and third last elements are not compared and so on [11].

IV. ALGORITHM

1. Start
2. Define control variable i=1,j=0;
3. Define temp;
4. While (i<size) repeat loop
5. While(j<size-i-1) repeat loop
6. If(A[j]>A[j+1])
7. Temp=A[j];
8. A[j]=a[j+1];
9. A[j+1]=temp;
10. Increment I and J by 1;

V. ALGORITHM OF INSERTION SORT

Step 1: The second element of an array is compared with the elements that appear before it (only first element in this case) [8]. If the second element is smaller than first element, second element is inserted in the position of first element. After first step, first two elements of an array will be sorted.

Step 2: The third element of an array is compared with the elements that appears before it (first and second element). If third element is smaller than first element, it is inserted in the position of first element. If third element is larger than first element but, smaller than second element, it is inserted in the position of second element. If third element is larger than both the elements, it is kept in the position as it is. After second step, first three elements of an array will be sorted [4].

Step 3: Similarly, the fourth element of an array is compared with the elements that appear before it (first, second and third element) and the same procedure is applied and that element is inserted in the proper position. After third step, first four elements of an array will be sorted. If ,there are n elements to be sorted. Then, this method is recurring n-1 times to get sorted list of array.

VI. ALGORITHM

1. Start
2. Initialize pass=1;
3. While (i<=n) repeat loop
4. Key=A[pass];
5. Initialize control variable j, where j=pass-1;
6. While(j>=0 && a[i]>>key) repeat loop
7. A[j+1]=a[j]
8. Decrement j by 1;
9. A[j+1]=key;
10. Exit.

VII. INSERTION SORT

1. Insertion sort maintains a sorted sub-array, and repetitively inserts new elements into it. The process is as following:
2. Take the first element as a sorted sub-array.
3. Insert the second element into the sorted sub-array (shift elements if needed).
4. Insert the third element into the sorted sub-array.
5. Repeat until all elements are inserted.
6. The following insertion Sort() method implements insertion sort. It uses a nested loop to repetitively insert elements into the sorted sub-array.

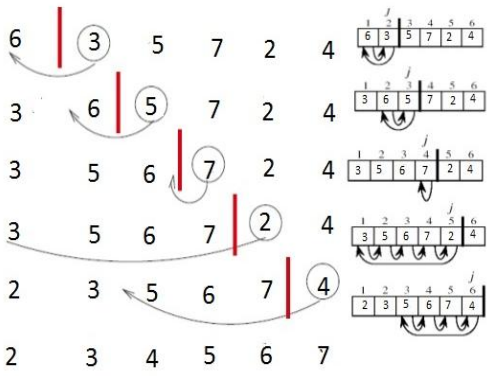

Figure 1: Example of Insertion sort

VIII. BUBBLE SORT

Bubble sort repeatedly compares adjoining pairs of elements and swaps if required. Scan the array, swapping adjoining pair of elements if they are not in relative manner. This bubbles up the greater element to the end. Scan the array again, bubbling up the second

greatest element. Replicate until all elements are in order [9]. The following bubble Sort() method applies bubble sort. It uses a nested loop to repeatedly swap elements and bubble up the greatest elements one by one.

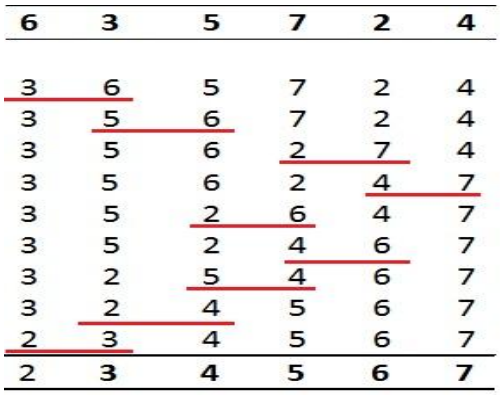

Figure 2: Example of Bubble sort

IX. DATA STORAGE

Data storage is the place where data is held in an optical or electromagnetic form for access by a computer processor in a computer [7]. Storage is frequently used to define the data and devices connected to the computer through input/output (I/O) operations -- that is, hard disk, tape systems and other forms of storage that don't include computer memory and other in-computer storage. For the enterprise, the options for this kind of storage are of a much larger variety and expense than those associated to memory. Huge variety of services around the globe is accessible by users anywhere at any point of time, thus boosting the increase of the amount of processed data in a very short span of time [4]. For example, bus route finder applications for Smartphones are astonished by data-traffic as the number of Smartphone users increases. In total, the rapidly growing Web service technologies challenge the traditional theories and approaches to databases. Total efficiency and flexibility have become more critical than the individual process result between a record and a transaction, as these days it is not just a couple of DB servers that access the SAN storage. Many claim that the current RDBMS and traditional database theories are inappropriate for handling large-scale data (big data).

X. PROCEDURE FOR SEARCHING DATA

Many relational systems store rows in "pages" and then provide various indexes within the page. There are a lot of diverse schemes for relating pages to the table to where they exist. Dissimilar structures, based on B-trees or like algorithms, may be vital to index often retrieved columns, especially foreign and primary keys. Hashing could also be used for this resolution. Page data may be squeezed to decrease storage needs but also to decrease the number of pages that must be made to resolve a query. Fast penetrating of a text column may be applied using a reversed index. One of the important concepts of a relational database system is that the query language is independent of any implementation of underlying storage and navigation structures [4]. This allows the database itself to adapt to queries by either automatically constructing the structures it requires for query performance, or to deliver statistics and references to the database administrator to add such structures [4]. The data is printed to either a file in the file system or straight to disk. The "database" is a program that reads the data from the disks and portrays in its "tabular" representation of tables, rows and columns.

XI. CONCLUSION

In this article, we are trying to emphasize on the concept of sorting techniques based on data structures. In software industry where sorting proved to be crucial part in daily schedule, It makes easier and quicker to locate items in a sorted list than unsorted list. Sorting algorithms can be used in a program to sort an array for later scripting and searching out to an ordered file or survey.

REFERENCES

[1] F. Bergholm. “Edge focusing,” in Proc. 8th Int. Conf. Pattern Recognition, Paris, France, pp. 597- 600, 1986

[2] E. Argyle. “Techniques for edge detection,” Proc. IEEE, vol. 59, pp. 285-286, 1971

[3] Kanij F. aleya, D Samanta,” Automated damaged Flower Detection using image processing”, Journal of Global Research in Computer Science (JGRCS), pp.21-24, Volume 4, No. 2, ISSN: 2229-371X.

[4] S.A. Dar, T.Choedhen, S.Mukherjee, L.K. Singh,"Admin Architecture Based on Unix", International Journal of Advanced Electrical and Electronics Engineering ,Volume 5, Issue 3, Pages 17-20, 2016.

[5] W. E. Grimson and E. C. Hildreth. “Comments on Digital step edges from zero crossings of seconddirectionalderivatives’’. IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-7, no. 1, pp. 121-129, 1985.

[6] V. Torre and T. A. Poggio. “On edge detection”. IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-8, no.2, pp. 187-163, Mar. 1986.

[7] M Mukherjee, T Paul, D Samanta,” Detection of damaged paddy leaf detection using image processing”, Journal of Global Research in Computer Science (JGRCS), pp.7-10, Volume 3, No. 10, October 2012, ISSN: 2229-371X.

[8] R. M. Haralick. "Digital step edges from zero crossing of the second directional derivatives," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-6, no. 1, pp. 58-68, Jan. 1984.

[9] E. R. Davies. "Constraints on the design of template masks for edge detection". Partern Recognition Lett.,vol. 4, pp. 11 1-120, Apr. 1986

[10] D Samanta, and G Sanyal," Development of Edge Detection Technique for Images using Adaptive Thresholding", International Journal of Information Processing (IJIP), volume 6, issue 2, 2012

[11] D Samanta, and G Sanyal," Automated Classification of SAR Images Using Moment", International Journal of Computer Science Issues (IJCSI), Vol. 8, Issue 6, pp. 135-138, 2011, ISSN (Online): 1694-0814, Indexed by Elsevier (up to 2012), Impact Factor: 0.242.

[12] L. G. Roberts. "Machine perception of 3-D solids" ser. Optical and Electro-Optical Information Processing. MIT Press, 1965 . R. C. Gonzalez and R. E. Woods. "Digital Image Processing". 2nd ed. Prentice Hall, 2002.

[13] K. R Reddy, D Samanta, "Application of digital image processing for horticulture", International Journal of Engineering Sciences & Emerging Technologies (IJESET), pp. 228-232, Volume 6 Issue 2, 2013.