IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Visual Sorting Algorithms

[1]Sanket Ashok Patil, [2]Somraj Balaso Jadhav, [3]Shubham Sunil Patil,[4]Ms. Priyanka Rajendra Jadhav

[1,2,3]Students, [4]Assistant Professor,

[1,2,3,4] Computer Science And Engineering Department,

Abstract: The Visual sorting application developed using Html, CSS, JavaScript, Bootstrap, React.js (Frontend), Node.js (Backend). A study that tested the benefits of animated sorting algorithms for teaching. To visualize four sorting algorithms, a web-based animation application was constructed and visualization of data is implemented as a bar graph, after which a data sorting and algorithm may be applied. Data Structures and Algorithm (DSA) Can play a crucial role in a visual sorting application. In this application used to visualize seven commonly sorting Algorithm: Bubble sort, Selection sort, Quick sort, Merge sort, Heap sort and Tim sort .

Keywords **-** Sorting Algorithm Visualization, Sorting Algorithm Demos, Interactive Sorting Algorithms, Array Sorting Visualization

I. Introduction

The user can adjust the size of the bar graph, the visualization speed, and the technique used to display data in the form of a bar graph through this web application. The application's architectural design, technical framework, practical use, and educational advantages are all displayed and addressed.[1]

The primary objective of the thesis was to develop a program that would act as a tool for comprehending the operation of the majority of well-known sorting algorithms. An effort was made to provide the optimal user experience. The demonstration program is designed to be easy to use and friendly to users. You can test each sorting algorithm on your data to get the most value from learning. There is a recognition that the visual aid is an essential component of learning, having encountered sorting issues in my first year of school while working on algorithm design. It was fascinating to discover new methods for sorting algorithms in-depth when working on the thesis.[2]

Readers of this text are expected to have some programming experience to know basic data structures such as arrays, lists, trees and understand recursive procedures. Also, knowledge of some simple algorithms and their implementations could be helpful. In order to understand the topic better, knowledge of linear algebra and calculus is involved. The text of the thesis describes principles of the most known sorting algorithms which are demonstrated in the computer program. It might be used as a source for learning algorithms by students. Also, the program might be easily used as a demonstration by lecturers and tutors during classes. Besides, there is programmer documentation and user guide to the provided software.[3]

OBJECTIVE

- Identify patterns and characteristics specific to each algorithm, such as Bubble Sort's adjacent element swaps or Quick Sort's partitioning technique.
- Enhance learning engagement through interactive elements that allow users to modify input data and observe algorithm adaptations
- Make complex algorithms more accessible to readers through visual representations
- Ensure the original algorithmic concepts are preserved in the visual representation.

II. SYSTEM ARCHITECTURE

The system architecture of a visual sorting algorithm involves multiple layers, each with distinct roles. The UI layer ensures user interaction and visualization, the application logic layer handles the execution of sorting algorithms and visual updates, the data layer manages data state, and the optional backend layer provides additional processing capabilities. Integration and deployment layers ensure smooth communication between components and robust deployment processes.

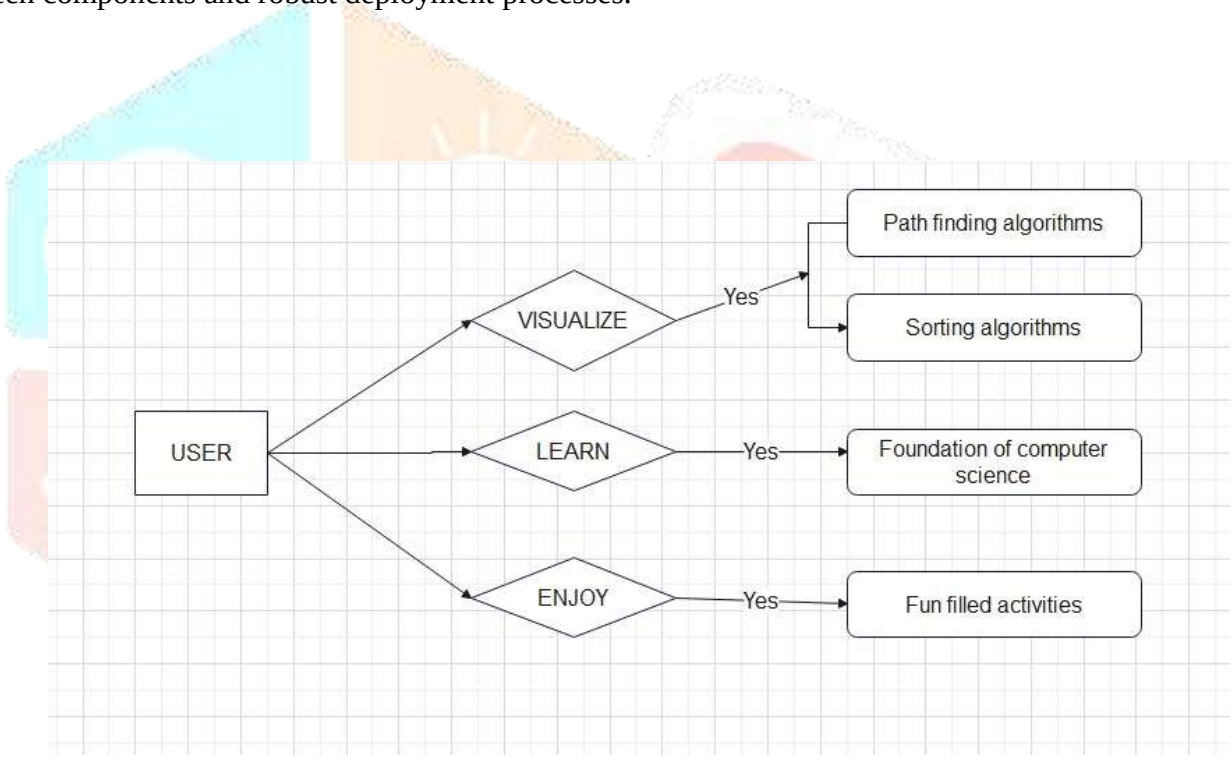


Fig 1 : System Architecture

Visual sorting algorithms provide a dynamic and interactive way to understand and analyze the behavior of various sorting techniques. By graphically representing the steps taken by algorithms like Bubble Sort, Quick Sort, and Merge Sort, these visualizations make complex processes more comprehensible and engaging. They allow users to observe the real-time progression of the sorting procedure, identifying key actions such as comparisons, swaps, and data partitioning. This visual approach is invaluable for educational purposes, as it helps learners grasp the underlying principles of sorting algorithms, recognize patterns, and compare the efficiency of different methods. Additionally, developers benefit from visual sorting algorithms by using them to debug and optimize their code, quickly identifying inefficiencies or logical errors. By making abstract concepts tangible, visual sorting algorithms enhance both learning and productivity, allowing users to gain deeper insights into algorithmic performance and choose the most suitable sorting method for their specific needs. Moreover, visual sorting algorithms are invaluable in communication and documentation. Educators can use animated visualizations to demonstrate sorting techniques in a more engaging and understandable way, capturing and retaining students' attention more effectively than static

text descriptions or code snippets. In technical documentation, visual representations help convey complex algorithmic concepts more clearly, making the documentation more accessible to readers who may not have a strong background in computer science. Modern web technologies like HTML, CSS, and JavaScript, along with libraries like D3.js, are commonly used to build these interactive tools, while backend support using server-side technologies like Node.js can handle intensive computations for more complex tasks. Real-time communication technologies like Web Sockets ensure that the visualization remains responsive and up-to-date with the ongoing sorting process. In conclusion, visual sorting algorithms bridge the gap between theoretical concepts and practical understanding, enhancing both learning experiences and real-world applications.

I. RESEARCH METHODOLOGY

3.1 Data Flow Digram

DFDs are applicable to various types of systems, including information systems, business processes, and software systems. They aid in visualizing and analyzing data flow, identifying areas of congestion and inefficiency, and effectively conveying system design to others. data flow diagram can be created using different notations, such as the Gane-Sarson notation and the Yourdon-DeMarco notation, depending on the designer's preferences and conventions.

3.1.1 Bubble sort

The foundation of bubble sort is the concept of swapping two neighboring elements in case their order is incorrect. The larger items tend to migrate or quote to the right quote; when the algorithm steps through each element from left to right. The algorithm is known as Bubble Sort for this reason.

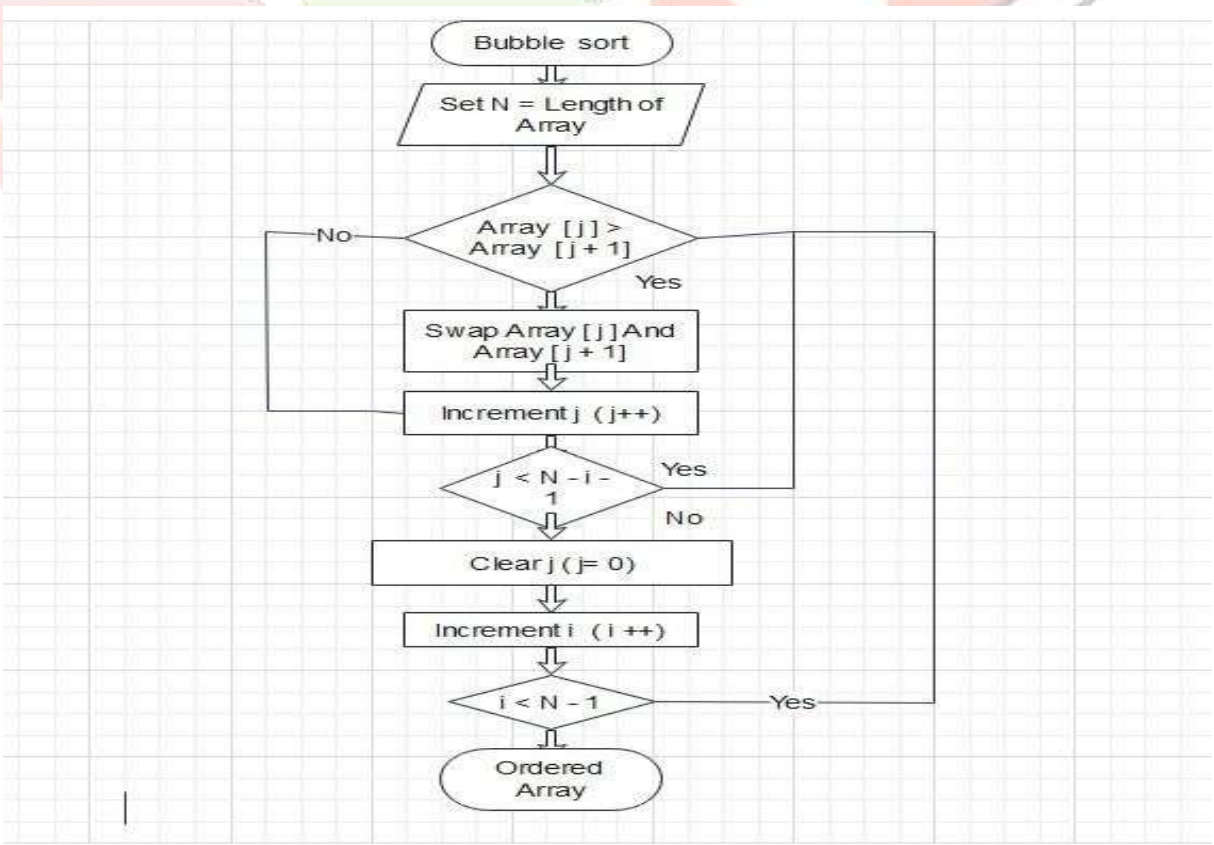


Fig 2 : Data Flow Diagram

IV. RESULTS AND DISCUSSION

4.1 Bubble Sort

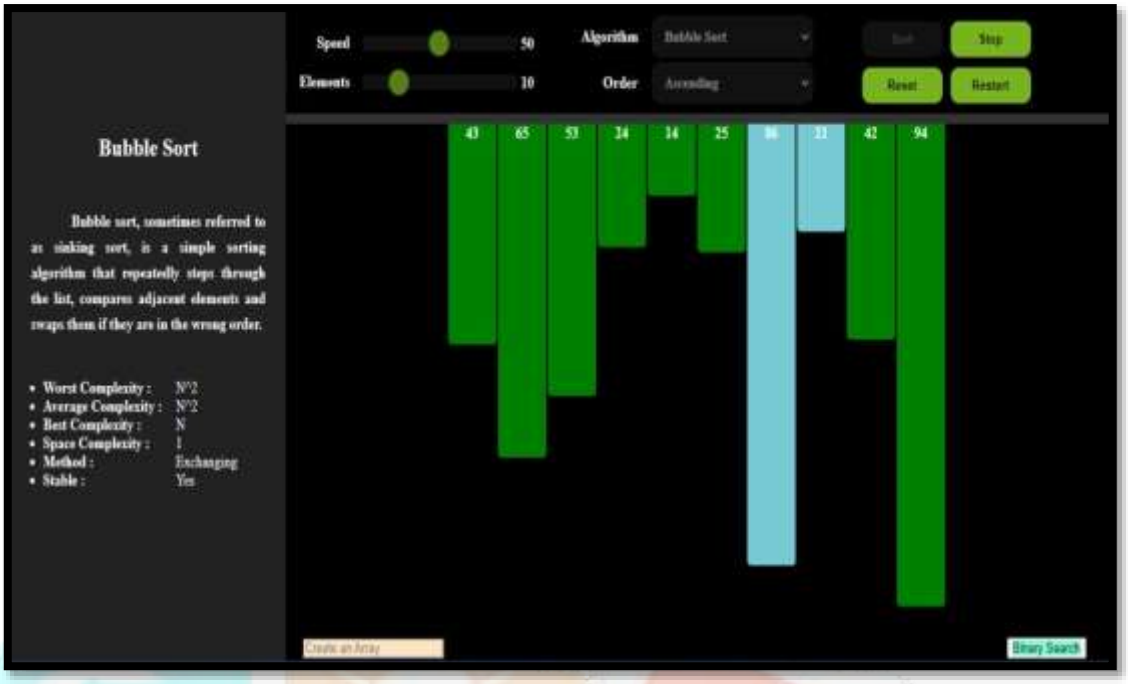


Fig 4: Bubble Sort

- Here's a basic explanation of how it works:

➢ Start with the first two elements of the list.

➢ Compare them, and if they are in the wrong order, swap them.

➢ Move to the next pair of elements and repeat step 2.

➢ Continue this process until you reach the end of the list.

➢ Repeat the process for each element in the list until no more swaps are needed

4.2 Shell Sort

- Here's a basic outline of how Shell sort works:

➢ Gap Sequence Selection: Choose a sequence of gap values. Commonly used sequences include the Knuth sequence (3x + 1) and the Sedgewick sequence.

➢ Iteration: Starting with the largest gap, perform several passes over the array, comparing and swapping elements that are separated by the current gap.

➢ Reducing Gap Size: After each pass, reduce the gap size. This can be done by applying the chosen gap sequence.

➢ Final Pass: Eventually, the gap size becomes 1, and the algorithm performs a final pass using insertion sort. By this point, the array has been partially sorted, making the insertion sort step more efficient.

4.3 **Selection Sort**

- Here's a basic outline of how selection sort works:

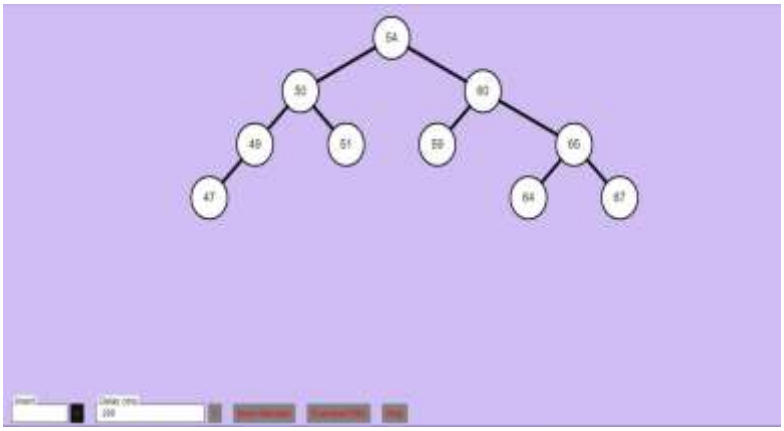
➢ Selection of Minimum (or Maximum) Element: Find the smallest (or largest) element in the unsorted sublist.

➢ Swap with First Unsorted Element: Swap the found minimum (or maximum) element with the leftmost unsorted element.

➢ Shrink Unsorted Sublist: Move the sublist boundary one element to the right, effectively reducing the size of the unsorted sublist by one.

➢ Repeat: Continue the process until the unsorted sublist becomes empty, indicating that the entire list has been sorted.

4.4 Binary Search Tree



- Here's how binary search tree typically works:

➢ Here's Binary Tree Structure: Each node in a binary search tree contains a key (value) and references to its left and right children (which can be null if there are no children).

➢ Ordering Property: The keys in a binary search tree are arranged in a specific order. For any node in the tree, all keys in its left subtree are less than its own key, and all keys in its right subtree are greater than its own key.

➢ Efficient Searching: Due to the ordering property, binary search trees allow for efficient searching. When searching for a key, the algorithm starts at the root and recursively navigates left or right based on a comparison of the search key with the keys of nodes in the tree until the key is found or the search reaches a null pointer (indicating the key is not present).

➢ Insertion and Deletion: Insertion and deletion operations in a binary search tree maintain the ordering property. When inserting a new key, the algorithm traverses the tree to find the appropriate position to place the new node. When deleting a key, the algorithm rearranges the tree to maintain the ordering property while removing the node.

III. CONCLUSION

The visual sorting algorithm project has successfully bridged the gap between theoretical knowledge and practical understanding, providing a comprehensive tool that enhances learning, debugging, optimization, and communication. By making abstract sorting concepts tangible and interactive, the project has significantly contributed to the field of computer science education and practice, offering lasting benefits for a wide range of users.

REFERENCES

[1]Veena R, D. Ramesh, *"Automatic text summarization–A systematic literature review"*, World Journal of Advanced Engineering Technology and Sciences, March 2023.

[2]Prerna Mishra, Kartik Garg, "Video-to-Text Summarization using Natural Language Processing", International Journal of Advanced Research in Science, Communication and Technology (IJARSCT), April 2023.

[3]Kapil Hande, Hrushi Karlekar, "NLP based Video Summarisation using Machine Learning", International Journal of Scientific Research in Science, Engineering and Technology, April 2023.

[4]Ramya R. S, M Shahina Parveen, *"A Survey on Automatic Text Summarization and its Techniques"*, International Journal of intelligent systems and applications in engineering, May 2023.

[5]Mengli Zhang,Gang Zhou, *"A Comprehensive Survey of Abstractive Text Summarization. Based on Deep Learning"*, State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou, China Guilin University of Electronic Technology, Guilin, China, August 2022.

[6]M. F. Mridha, aklima akter lima, *"A Survey of Automatic Text Summarization Progress, Process and Challenges"*, Department of Computer Science and Engineering, Bangladesh University of Business and Technology, November 2021.

[7]Nidhi Patel, "*Abstractive vs. Extractive Text Summarization (Output based approach) - A Comparative Study*", IEEE International Conference for Innovation in Technology (INOCON), May 2021.

[8]Ahmed Emad, Fady Bassel, "Automatic Video summarization with Timestamps using natural language processing text fusion", Authorized licensed use limited to: Robert Gordon University, May 2021.

[9]Ishitva Awasthi, Kuntal Gupta, *"Natural Language Processing (NLP) based Text Summarization - A Survey"*, Sixth International Conference on Inventive Computation Technologies [ICICT], June 2021.

[10]Sanjana R, Vedhavathi K R, "Video Summarization using NLP", International Research Journal of Engineering and Technology (IRJET), August 2021.