

Sorting Algorithm Visualizer

Ashish Kitukale¹ Noorin Sheikh² Prof. Abhijeet Thakare³

^{1,2}Author ³Professor

^{1,2,3}Department of Computer Engineering

^{1,2,3}SRPCE, Nagpur, Maharashtra, India

Abstract — Computer Science is a field of not just generating solutions to day-to-day activities but also to enhance is to the fullest. Algorithms are the basic practical unit for improving any activity that offer work with specific productivity, and to obtain increase the productivity of a task. Sorting is the key component of Data structures and Algorithms. However, it is complicated and we couldn't able to recognize the whole algorithm and code for the first time. So the use of this application has many superior understanding on education. The key element of this project is to help beginners to be able to visualize the sorting algorithms so our brain can understand faster and remember better. And also get better understanding of underlying operations. This paper outlines a study that tested the benefits of animated sorting algorithms for teaching. To visualize four sorting algorithms, a web-based animation application was constructed. A visualization of data is implemented as a bar graph, after which a data sorting and algorithm may be applied. The resulting animation is then performed either automatically or by the user, who then sets their own pace. This is a research on the computer science curriculum's approach to learning algorithms. The experiment featured a presentation and a survey, both of which asked students questions which may illustrate improvements in algorithm comprehension. These findings and reactions are catalogued in this document and compared to earlier investigations.

Keywords: Sorting Algorithms, React Visualizer, Selection Sort, Merge Sort, Bubble Sort, Insertion Sort, Heap Sort

I. INTRODUCTION

How do you get something done? You don't have to be extremely complex in solving the problem, for example, if your car's headlight is broken (although nowadays, manufacturers are trying the patience of the community with their increasingly abstract, space-age designs). The main issue is figuring out the best way to go about it. To locate step-by-step directions in your car's handbook, you conduct research, or do you use instinct to find someone who knows how to do it? In short, my instinct tells me that I am a visual learner and hence more suited to acquire topics by watching them than by reading about them. In this case, I found that seeing the data move to its rightful spot as the result of an algorithm is MUCH easier to follow than looking at the source code and trying to figure out where the data was supposed to go. My project was born out of my curiosity about sorting algorithms, which inspired the idea for this paper, which details an online tool I built that explains how sorting algorithms transform and organize sets of data. It is possible to organize a list of people, for example, by their age in ascending order using different methods. To aid my visualization, I created a histogram of numerical data to represent four of the well-known examples. Each number is depicted as a bar, and the height of each bar represents the value of that number. It is being shifted by the algorithm from

its original, unordered location to its final ordered place, making it distinct from the rest of the data. Selection Sort, Bubble Sort, Insertion Sort, and Merge Sort are the four sorting algorithms. 2 Let's imagine that you have printed each person's age on a separate index card. Bring the youngest card to the front and then sort the cards by age. To discover the next smallest item, identify the age that has already been ordered and position it behind the already ordered age. Index cards full of ages will be at the end of the pile. Selection Sort works in the same way as this. In this case, to sort a set of data, you select the smallest first, and then the next smallest, and so on until you've sorted all of the data. This technique is quite simple to explain to someone in conversation, but more advanced sorting algorithms, such as Quick Sort, which requires the data to be moved around a pivot point, are not easy to grasp using text alone. I wanted the animation to appeal to a wide spectrum of individuals utilizing various technology media, and so I had it made in a web-based format. Instead of requiring the user to install extra software or attempt to organize setups to use the tool, this helps to remove this source of anxiety. It uses HTML5 (Hypertext Markup Text Language) JavaScript, and CSS for the website's layout (Cascading Style Sheets).

II. LITERATURE SURVEY

The paper "Algorithm Animation," by A. Kerren and J. Stasko [3] is a step-by-step guide to analyzing the environment, means, and available coding methods to use a sorting animation. Many different types of software are listed to be used for animation, one of which was BALSAs, which pioneered the interesting event approach [3, p. 3]. BALSAs was created by Marc Brown and the interesting event approach was coined to determine what part of the sorting algorithm was significant to both clearly see and understand how the algorithm performs. Additional software that followed this principle were Zeus, CAT, Tango, and Samba, to name a few [3, p. 2]. These developments prove the continuing interest in creating animation tools. For example, the interesting event approach I took focused on animating the movement of data as a visual description of the algorithm. 5 For a direct analysis of how students respond to sorting animation, the paper "Do Algorithm Animations Assist Learning? An Empirical Study and Analysis" [6] provides an in-depth view. A post-test study was used to gather information on comparing the results of students who only had textbook resources to those who had a textbook as well as an animation for assistance. The post-test was the same for each group of students, which covered a comprehensive view of the topic. The study found no clear support that an animation would help students with the material significantly. The group of students that had the animation tool in addition to the textbook material averaged correctly answering two more questions than the control group [6]. The paper concluded that visualizing algorithms sounds good, but may

not achieve the desired results when implemented. Another article I found is Stasko's "Using Student-Built Algorithm Animations as Learning Aids." [5] It initially stated the same fault found in the previous article, where showing an animation does not promote understanding as much as desired. In an interesting twist, however, students were given assignments to build the animations themselves, rather than use some already made as a means of better understanding. The students were introduced to the visual programming tool Samba. After a few introductory assignments to get the knack for using the software, they were given the option to animate an algorithm, keeping in mind that a person who did not know the material could understand it. The results showed positive feedback and overall better understanding of how the algorithms worked. Also, some students found that they had misconceptions about the material when implementing them in an animation.

III. METHODOLOGY

As mentioned above, we already have an existing application with existing functionality of basic algorithm visualization of both path-finding and sorting algorithms, mazes and patterns and displaying time complexity [1]. Now, to make the visualizations all the more realistic and easier to comprehend we decided to include a few additional features in our application. The additional features to be included are as follows: The flow chart below in Figure 1 describes the logic of the application. It begins with opening the application, then the user can see customizable settings in the application UI. They can be changed, or the default setting can be used in which case it will generate the steps of sorting for the default sorting algorithm. To visualize the sorting process, all navigation buttons except the "reset" button can be pressed or the "start" button for automatic visualization. The reset button will generate the random array again. After the visualization is complete, the process can be repeated, or the application can be closed.

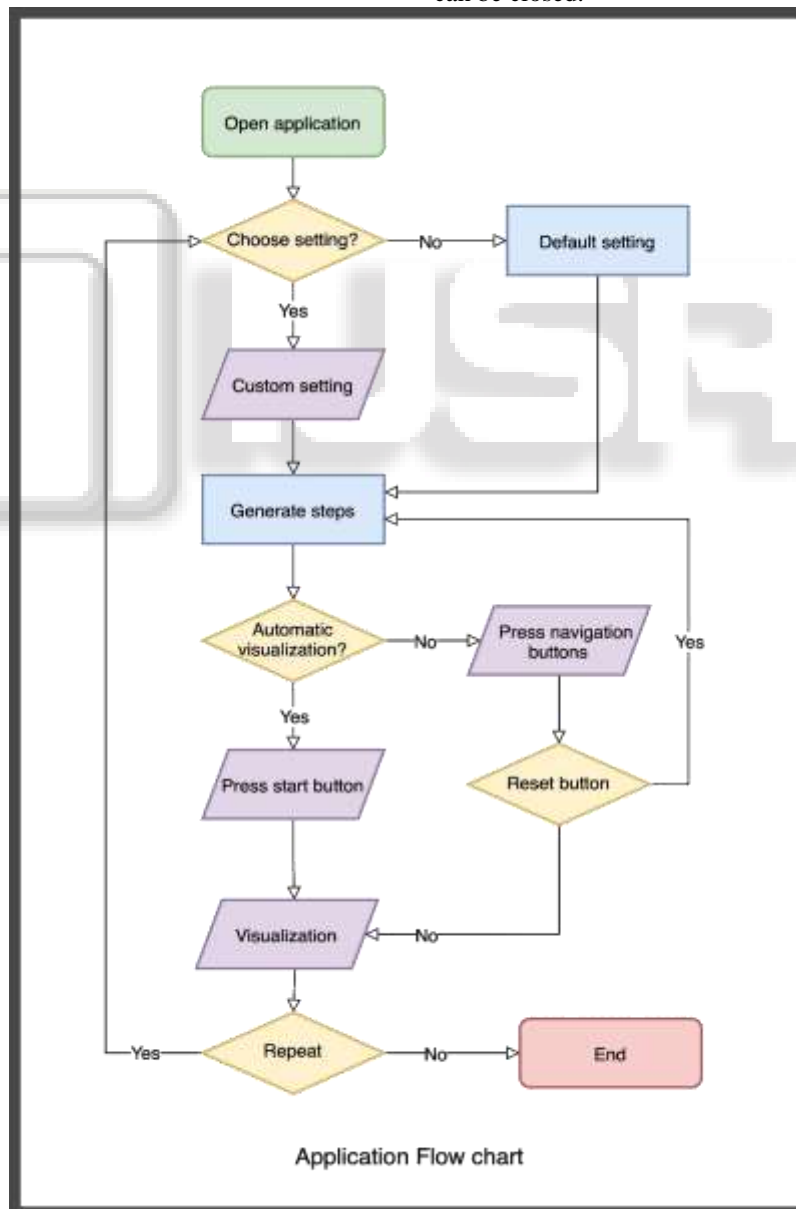


Fig. 1: Application Flow chart

IV. TECHNOLOGIES

The HTML5 (Hypertext markup language), PHP (Hypertext preprocessor), JS(JavaScript) libraries and CSS (Cascading Style Sheets) is used for building the project. Code contain in one Html file i.e. preprocessed by PHP language. We have used Speech Recognition library of js(Speech Recognition) for storing data/note in the form of audio i.e. by speaking. However, we modified, user not only store information in speech-to-text form but also user can store information in written form. For visualization we have used Standard Recognition library. We have used four object: Represent information and algorithm related to sorting as well as searching.

- Visualization.
- we can size the array according to our choice.
- How much time is required to sort the particular array we able to know that also (time required). We have used waterfall methodology to work more efficiently through all the phases of our project. We have merge both the modules i.e. visualization and all algorithms. In this we integrated both and created the final result by generating array automatically of particular sorting and then press on the sorting button, start to sort numbers one by one and also we have built time requirement module to understand how much time is required to sort the particular array.

V. PROPOSED SOLUTION

The proposed system involves the simulation of the different type of sorting algorithms codes. We created six sorting

algorithm which are bubble sort, insertion sort, selection sort, heap sort, merge sort and quick sort. When we perform particular sorting the step explanation of that sorting is done through audio. Share section will also available so the user can share and in about section our information will be available so user can contact us. The implementation of all sorting techniques is available in this code section and also we can manually insert an array of our choice and their code too. The user can select any of these sorting algorithms to see the visualization of how that algorithm works. We can see visualization of particular array data of sorting is done in graphical representation (i.e. bar graph). It can also lend a different perspective on how the algorithms perform semi-sorted data compared to the given ordering options Allow the user to choose between sorting input data that is already in order, or in reverse and random orders. The default is in sorted order. All sorting algorithm are unlock user need not to buy for practice any of the sorting algorithm.

VI. IMPLEMENTATION

This section will describe the project implementation, coding practices, vision of the project in detail.

A. Building the application

The goal of the project was to create a visualization platform to visualize various sorting algorithms in the form of a web application. There are abundant choices of tools for creating web applications. But the tools mentioned in the previous section are the ones used while creating a project.

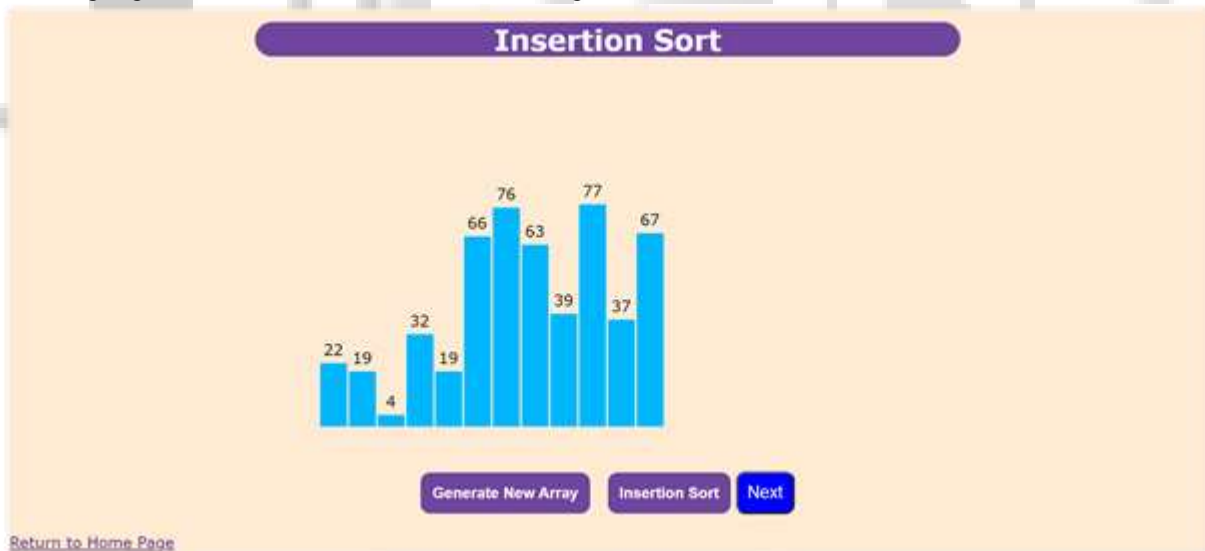


Fig. 2: Insertion sort

VII. APPLICATION

- Student can learn sorting of their choice by their own whenever he/she get time.
- Teacher can use this application for teaching or explaining the concept of sorting.
- We can use this application in institute, coaching center, colleges, etc.

VIII. CONCLUSION

The project titled “App based sorting algorithm visualizer”, has been completed. The system has been designed with great care and error-free, while still being effective and time-saving.

This platform as well as practical experiences with system we believe it helps to improve the education

quality in the stream and contribute to the solution for some of the problems in higher education.

However, strong mindset help us to research and generate animation to improve learning in the classroom. Hence, from this project we can easily understand the working of sorting and searching through their geometric graphical visualization or representation and their explanation. It is easy to know all sort and search and also effectively beneficial and efficient.

IX. FUTURE SCOPE

Our intentions here include development of new plug-in modules from the area of sorting algorithms and more complex data structure. In present the data which we have store in database in the form of audio and written in future we will see the stored data on the side of information page. Need to give more thought on how to optimize the code so that it can work with multiple people using it. User can split the screen in which half of the screen will show the visualization of sorting and other half will show the code of particular sorting and as well as searching.

REFERENCES

- [1] T. Bingmann. "The Sound of Sorting - 'Audibilization' and Visualization of Sorting Algorithms." Panthemanet Weblog. Impressum, 22 May 2013. Web. 29 Mar. 2017
- [2] Bubble-sort with Hungarian ("Cs'ang'o") Folk Dance. Dir. K'atai Zolt'an and T'oth L'aszl'o. YouTube. Sapientia University, 29 Mar. 2011. Web. 29 Mar. 2017.
- [3] Moreno, E. Sutinen, R. Bednarik, and N. Myller. Conflictive animations as engaging learning tools. Proceedings of the Koli Calling '07 Proceedings of the Seventh Baltic Sea Conference on Computing Education Research - Volume 88, Koli '07 (Koli National Park, Finland), pages 203-206
- [4] J. Stasko. Using Student-built Algorithm Animations As Learning Aids. Proceedings of the Twenty-eighth SIGCSE Technical Symposium on Computer Science Education. SIGCSE '97 (San Jose, California), pages 25-29.
- [5] J. Stasko, A. Badre, and C. Lewis. Do Algorithm Animations Assist Learning?: An Empirical Study and Analysis. Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, CHI-93 (Amsterdam, the Netherlands), pages 61-66.
- [6] Kerren and J. T. Stasko. (2002) Chapter 1 Algorithm Animation. In: Diehl S.(eds) Software Visualization. Lecture Notes in Computer Science, vol 2269. Springer, Berlin, Heidelberg.