Educational Administration: Theory and Practice
2024,30(01), 5363-5368
ISSN:2148-2403
https://kuey.net/
Research Article

Algorithm Visualiser for Coin Changing Problem

Atharva Rajmane[1], Vatsal Upadhyay[2], Dhruv Shah[3], Neha Agarwal[4]

[1] Information Technology Department, Dwarkadas J Sanghvi College ofEngineering, Mumbai -56, Maharashtra, India
Email: atharvarajmane53@gmail.com
[2] Information Technology Department, Dwarkadas J Sanghvi College ofEngineering, Mumbai -56, Maharashtra, India
Email: vatsalu2003@gmail@gmail.com
[3] Information Technology DepartmentDwarkadas J Sanghvi College of Engineering, Mumbai -56, Maharashtra, India
Email: dhruvshah6903@gmail.com
[4]Information Technology Department, Dwarkadas J Sanghvi College of Engineering, Mumbai -56, Maharashtra, India
Email: nehagarg60@gmail.com

Citation: Atharva Raj mane et al. (2024), Algorithm Visualizer for Coin Changing Problem, *Educational Administration: Theory and Practice*, *30*(01), 5363-5368,
Doi: 10.53555/kuey.v30i1.8943

ARTICLE INFO **ABSTRACT**

The goal of this research project is to create an interactive analytical and teaching tool in the form of an algorithm visualizer application. The main objective is to improve users' comprehension of the differences between the Greedy Method and the Dynamic Programming Method when addressing the coin change problem by providing a visual depiction of each method's step-by-step implementation. The primary task at hand is developing a user-friendly and educational interface that facilitates the selection of algorithms, customization of input data or graphs, and real-time observation of algorithmic processes by users. Through interactive visualization approaches, the project's output is anticipated to be a useful tool for educators and learners alike, aiding in the understanding of intricate algorithmic principles.

keywords: Dynamic programming, Algorithm Visualiser, Greedy, Method, Dave, Mavis, Piton, Dom

I. INTRODUCTION

It is generally acknowledged that algorithm animation and program visualization are distinct concepts informally. External representations that are closely aligned with program source code are referred to by the former name. The latter phrase describes external representations of a program's abstract behavior, usually an algorithm. Program visualizations are often generated automatically due to their lower abstraction level, while algorithm animations require human participation due to their higher abstraction level. It is important to continue investigating alternative approachesfor program visualization because one of the primary excuses given by educators for not utilizing visualization software in the classroom is effort. [1]

In the field of computer science and algorithmic problem solving, it is crucial to comprehend and apply various algorithmic procedures. Among these strategies, greedy methods and dynamic programming (DP) are two that are especially crucial for solving optimization challenges. One example of a problem like this is the coin change problem, which entails figuring out how many coins are absolutely necessary to make a specific quantity of change. Although both the Dynamic Programming and the Greedy techniques provide solutions for this issue, their methods and results are very different.This research project will use an algorithm visualizer application as an analytical and instructional tool to help users better comprehend the distinctions between the Greedy Method and the Dynamic Programming Method for addressing the coin change problem. The purpose of this application is to give users access to an interactive environment where they can watch these algorithms beingexecuted step-by-step in real time.

The primary task is to design an intuitive and instructive user interface that lets users select algorithms, modify input data or graphs, and see algorithmic processes in real time. By bridging the gap between theoretical principles and real- world application, this visualizer seeks to empowerenthusiasts and learners of computer science. It accomplishes this by enabling more in-depth understanding of how dynamic programming and greedy tactics function.

II. RELATED WORKS

Different algorithm visualization systems are suggested as effective and alternate learning settings for beginning programming courses. They have dynamic aspects that are based on animation techniques and are meant to encourage students to experiment and build algorithmic knowledge by showing how simple algorithms behave. The web-based dynamic algorithm visualization environment known as DAVE is presented in this paper with the goal of assisting secondary education students in their study of fundamental algorithms. DAVE makes it easier for students to experiment with array techniques by enabling code and data editing. Presentation of early findings from an assessment study demonstrated the system's usability and shown its potential to help students build effective mental models of fundamental array algorithms.[6]

Up until recently, Java was the most often used implementation technology for antivirus software; however, as HTML5 and JavaScript have become more popular, Java's significance on the web is waning. HTML5-based
antivirus programs have been developed in the last few years. Let's examine a few current AVs. JAWAA 1 is both an animation authoring language and an animation visualization system. Certain data structures and graphical primitives can be used in animations. A learning environment including algorithm simulation exercises is called TRAKLA2 2. Students are required to simulate an algorithm's steps in order to build an AV for the activities. The program includes several exercises, supports various data types, evaluates student answers automatically, and provides visual feedback. An online collection of AVs is called VisuAlgo 3. When learning algorithms, users can observe how they are implemented in a slideshow, for example, or by investigating operations on a data structure,such a binary heap. [2].

A virtual lecture is delivered in a distant learning setting after the user accesses a particular website. An instructor and a class of pupils make up a virtual classroom, just like any other. It should be possible for users to communicate, work together, and exchange information. The sole distinction from a traditional classroom is that students mayattend class from different locations.

The majority of algorithm visualization technologies are not intended for usage in a typical remote learning setting. We introduce a new conceptual model for algorithm visualization in the context of a virtual classroom in this research. Additionally, we introduce the Multi-level Algorithm VIsualization System (MAVIS), a system designed to implement this model.[7]

It is impossible to make any assumptions about the processing power of the clients when a virtual lecture and visualization are sent via the Internet. Furthermore, communication demands may differ significantly across different links. Delays may ensue from this, making synchronization a challenging operation. Therefore, a key focus of our system is to dynamically support the expectedheterogeneity.

Rather than matching an algorithm with a single visualization, our paradigm supports multi-level visualizations that dynamically adapt to the network's heterogeneity [8]. In other words, the visualization system ought to be able to generate a wide range of visualizations for each algorithmic event, as opposed to only one.[9]

Our work expands upon the framework of PITON, an earlier tool for teaching programming from the same institution and written by the same authors. With the extra benefit of a program visualization (PV) tool, PITON serves as a typical programming workspace aimed primarily at beginner programming students. Students can see how their code is being executed using this PV feature, which helps with comprehension when they run into problems. Although the goal of DS-PITON is data structure implementation learning, whereas PITON's goal is helping introductory programming, PV functionality is an essential part of DS-PITON.
DS-PITON targets data structures particularly, extending the capabilities of PITON. Seven common data structures are now supported for visualization: queues (array and linked list implementations), linked lists, stacks (both arrayand linked list implementations), arrays, and priority queues with linked lists as a means of implementation.[4]

Our work makes effective and interactive data structure visualization possible by utilizing JavaScript as the primarytechnology. A number of current tools and frameworks demonstrate the usefulness of JavaScript in educational andvisualization applications[10], even though they are not functionally connected.

III. METHODOLOGY

A. System Architecture

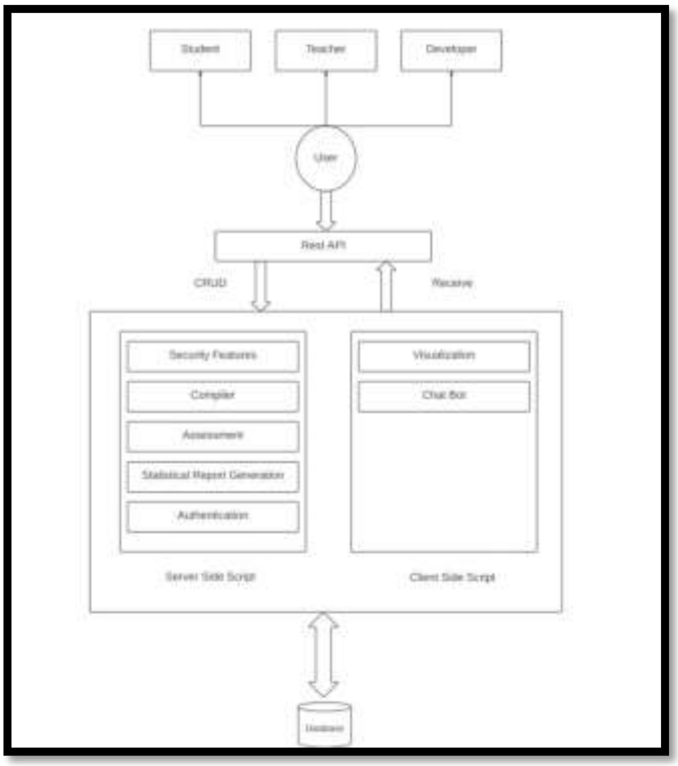


Fig 1. System Architecture

In Fig.1 ,this architecture is structured to cater to the diverse needs of three primary user roles: students, teachers, and developers. Each role possesses distinct privileges and access levels within the application ecosystem.
The REST API is the backbone of our system, acting as the interface layer that facilitates smooth user-applicationcommunication. The REST API offers a standardized set of protocols and functions by abstracting away the system's intricacies, making tasks like data creation, retrieval, updating, and deletion (CRUD) over HTTP protocols easier.
Important server-side processing tasks are handled by our server-side script on the backend. This component is made up of multiple important modules: CRUD operations to manipulate data efficiently; strong security features to protect sensitive data, such as encryption and access controlmechanisms; and a compiler module to make code compilation easier for the execution of custom scripts or educational goals. The server-side script also includes an authentication mechanism to confirm user identity prior to allowing access to system resources, a statistical report producing module to examine usage metrics, and an assessment system to rate user performance.
Our application's frontend controls client-side processing and communicates with the backend, assisted by the client- side script. This part uses visualization technologies to present information or results graphically and improve user understanding and interpretation. It also offers a chat bot function that lets users ask questions of an automated systemor utilize it to accomplish particular tasks inside the application.
Since the database houses all application data, it is the primary component of our data management strategy. The REST API can be used by the server-side script to simply communicate with the database, enabling user-initiated actions for data retrieval,updating,andstoring.
In our approach, user requests are sent using the REST API to start the interaction loop. The server-side script does the required processing tasks by giving relevant answers to these queries. The client-side script then either renders the results or sends them back to the user. Furthermore, the server-side script communicates with the database to enable effective data management, encompassing storage and retrieval functions.

A. Tech Stack used for Algorithm Visualiser

With the use of a well-considered tech stack, the algorithm visualizer project offers users an entertaining and instructiveexperience. Let's examine the main elements:

1. JavaScript (JS):

Core Logic: The visualizer's foundation is JavaScript.Because of its adaptability, sorting algorithms can be used with success.

Real-Time Interaction: JavaScript runs directly in users' browsers since it is a client-side programming language.Step-by-step visualization of the sorting processes is made possible by this real-time execution.

2. **HTML (Hypertext Markup Language):**
   Core Logic: JavaScript is the backbone of the visualizer. Sorting algorithms are useful because of their versatility.
   Real-Time Interaction: Because JavaScript is a client-sideprogramming language, it operates directly in users' browsers. This real-time execution enables step-by-step visualization of the sorting processes.
3. **CSS (Cascading Style Sheets):**Visual Styling: The visual elements are the responsibilityof CSS. It guarantees a beautiful, immaculate user interface.Responsive Design: CSS improves user experience byadjusting layouts to various screen sizes and devices.

B. Tech Stack for Assessment

This section describes how we created the assessment partof our algorithm visualizer using a TypeScript and Firebase platform similar to LeetCode.

1. Assessment Platform Design:
Using a question bank approach, the evaluation platform groups and saves different coding issues according to the data structures they contain and how complex they are. Justas in the LeetCode interface, users may browse and choosewhich issues to work on.

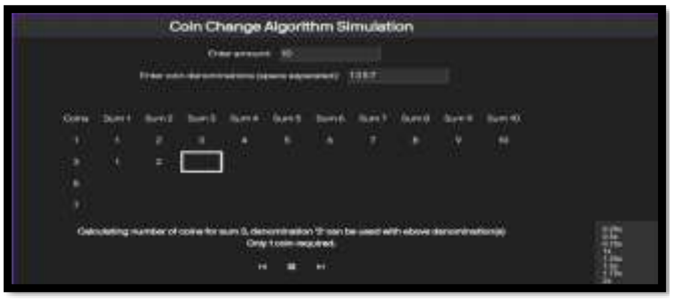
2. Development Stack:
Frontend: The robust type system of TypeScript facilitates error detection, code comprehension, and development code maintenance.
Backend: Because of its scalability, ease of integration, andfeatures like database administration and user authentication, Firebase is used as the backend solution.

3. Functionality Breakdown:
User Interface (UI): Users can select problems, edit code, and submit them with ease because to the UI's straightforward design. This comprises traits such as:
List of Issues: shows the issues that are currently available along with information about the data formats, level of complexity, and title.It provides a detailed explanation of the issue statement, anticipated inputs, and intended results.
Code Editor: With the help of this tool, users can write their solutions in Python, C++, or Java. For smooth codeexecution and visualization, this editor should ideally connect with the visualizer component.
Submission and Assessment: Users may choose to send in their code for assessment. The platform ought to offer inputregarding the accuracy and effectiveness of the solution, potentially through automated testing techniques.
Visualization: To help in understanding the code's execution flow, the visualizer component should be developed upon the already-existing visualizer functionalities. A dynamic representation of the data structures and operations utilized in the user's solution isrequired.

4. Integration with Visualizer:
The assessment instrument effortlessly incorporates with the current algorithm visualizer. The platform retrievescode contributed by users and feeds it into the visualizer component. The code is then executed step-by-step by the visualizer, which dynamically updates the display in response to actions taken on the data structure. By enabling users to observe how their code impacts the data structures in real time, this aids in their understanding of debugging and helps them get better at it.

5. Security and Data Management:
Firebase provides a range of integrated security features forhandling authorization and user authentication. Moreover, the platform needs to provide safe data storage procedures for user code and evaluation outcomes in the Firebase database.
This method illustrates how we used a TypeScript and Firebase platform similar to LeetCode to build the algorithm visualizer's evaluation component, providing users with an enjoyable and productive learning environment.

IV. VISUALIZATION OF ALGORITHM

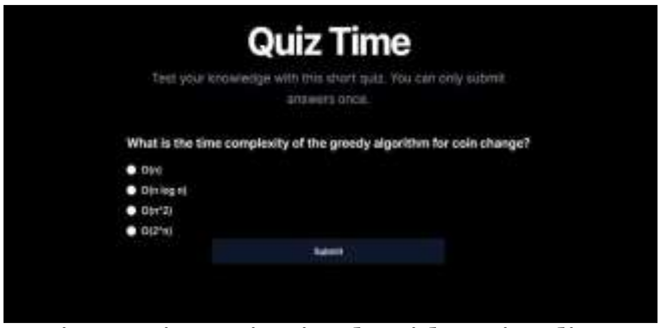


Calculating number of coins for sum 3, denomination '3' can be used with above denomination(s)
Only 1 coin required.

Fig 2. Algorithm Visualiser for Coin change Problem

The visualizer in Fig. 2 has an extremely user-friendly userinterface (UI). The steps that each algorithm takes to solvethe coin flip problem are displayed in an interactive dynamic grid. A real-time message section that provides comprehensive updates on the simulation's state enhances the visual representation. Through the usage of this grid, users can better understand the underlying concepts by seeing how different currency denominations assist them reach the required total amount. These are useful recommendations that provide an in-depth analysis of the decision-making process used in each program.

The visualizer's control buttons have a simple design that further enhances user engagement and control[12]. These thoughtfully positioned options allow users to progress through the simulation at their own pace by pausing, fast- forwarding, or reversing the processes. By selecting the speed modification option, users can additionally adjust the animation's speed to better fit their desired learning style. This feature allows for a wide variety of learning styles andpreferences, ensuring a tailored learning experience.


Quiz Time

Test your knowledge with this short quiz. You can only submit answers once.

What is the time complexity of the greedy algorithm for coin change?

- ● O(n)
- ● O(n log n)
- ● O(n^2)
- ● O(2^n)

Fig 3. Quiz Section in Algorithm Visualiser

Figure 3 displays Students can use the interactive "Quiz Time" feature of the Algorithm Visualizer to assess their comprehension of algorithm complexity. The tool time complexity analysis in this section concentrates on how wellthe greedy algorithm handles the coin change problem.
Following a brief title and introduction, participants are required to complete a brief quiz to determine their level ofknowledge; the fact that only one submission is allowed emphasizes the need of thoughtful responses.

To find the temporal complexity of the greedy algorithm, participants must select an answer from a set of multiple- choice options after answering a question. Each option has a selectable circle next to it so that users can select one and then click the "Submit" button to confirm. Students will learn more about theoretical concepts and how to apply them in practical settings by finishing this quiz. This method of learning algorithms encourages introspection andcritical thinking.

Coin Change Algorithm

[illegible]

Dynamic Programming Approach

[illegible]

Algorithm

[illegible]

Fig 4 . “Ask AI” section of Algorithm Visualise

The AlgoViz website's "Ask AI" section offers details on
the mathematical underpinnings of several algorithms. Userscan start a smooth dialogue with the AI system by typing their questions into the large search box in the top right corner. After that, they will receive meaningful answers and explanations. In an effort to make the section on requesting an AI system for assistance as user-friendly as possible, it is divided into tabs marked "Home," "Simulation," "Coding," "Quiz," and the main "Ask AI" page.

The "Ask AI" section's main article provides readers with in-depth details on specific algorithms, like the "Coin Change Algorithm." This comprises a description of the goals, procedures, and, on occasion, even the mathematical formulas used in the program. The simple visual style, with awhite background and readable black lettering, improves the user experience. With the help of the Gemini API, the "Ask AI" function processes user requests using state-of-the-art technology and delivers precise results. Because of this, it is a very helpful tool for algorithmic research, which includes theoretical reasoning.

V. CONCLUSION

The recommended approach significantly enhances users' educational experiences by offering personalized practice, interactive visualization, and detailed feedback, thus strengthening algorithm comprehension and problem-solving skills. The user-friendly interface and secure, welcoming atmosphere make learning enjoyable. Expanding the platform to include dynamic programming and greedy algorithms will broaden algorithmic exploration beyond the coin change problem. Bridging the gap between theory and practice through real-world examples and interactive features will further improve learning. Collaborative tools like discussion boards and real-time coding sessions can foster community and knowledge sharing. Lastly, the "Ask AI" section can be enhanced with advancements in machine learning and naturallanguage processing for more relevant and tailored explanations.

ACKNOWLEDGMENT

We extend our heartfelt gratitude to the open-source community, volunteers, advisors, mentors, and participants for their invaluable input into this study. Their collaboration, critiques, and recommendations have all beenextremely beneficial in shaping and refining our proposed response. We also thank our friends, family, and colleagues for their unwavering inspiration and support during ourjourney. Since their combined help has been crucial to the success of our project, we truly thank them for their contributions.

REFERENCES

[1] J. Angel ´ Vel´azquez-Iturbide, Antonio P´erez-Carrasco and Jaime Urquiza-Fuentes, “A Design of Automatic Visualizations for Divide- and-Conquer Algorithms”, ScienceDirect, 2009.
[2] Aditya Thakkar, Kavita, Sonali Dash, Sanjeev Kumar Joshi, "Sorting Algorithm visualizer", IEEE Xplore, 2022.
[3] Barnini Goswami , t Barnini Goswami , Akash Gupta, Antriksh Gupta-"Algorithm Visualizer: Its features and working ", IEEE, 2021.
[4] Rossevine Artha Nathasya, Oscar Karnalim , Mewati Ayub- "Integrating program and algorithm visualisation for learning data structure implementation", ScienceDirect, 2019.
[5] Nikhil Yadav, Karishma Dhameja, Prakhar Chaubey-"Path Finding Visualizer Application for Shortest Path Algorithm", IEEE, 2021.
[6] “Design and evaluation of a web-based dynamic algorithm visualization environment for novices”, Euripides Vrachnos, Athanassios Jimoyiannis, ScienceDirect 2014
[7] MAVIS: A multi-level algorithm visualization system within a collaborative distance learning environment Igal Koifman a, Ilan Shimshoni b, Ayellet Tal
[8] B. Goswami, A. Dhar, A. Gupta and A. Gupta, "Algorithm Visualizer: Its features and working," 2021 IEEE 8th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON), Dehradun, India, 2021, pp. 1-5, doi: 10.1109/UPCON52273.2021.9667586.
[9] A. Kumari, M. Mittal, V. Jha, A. Sahu, M. Kumar, N. Sangwan and N. Bohra, Algorithm Visualization - Modern WebBased Visualization of Sorting and Searching Algorithms, Advances and Applications in Mathematical Sciences, Volume 21, Issue 5 (March 2022), 2721-2736.
[10] T. L. Naps, Jhave: Supporting algorithm visualization, IEEE Computer Graph ´ ics and Applications 25(5) (2005), 49-55
[11] Haaranen LAhrenberg LHellas A(2023)Decades of Striving for Pedagogical and Technological AlignmentProceedings of the 23rd Koli Calling International Conference on Computing Education Research10.1145/3631802.3631809(1-8)Online publication date: 13-Nov-2023
[12] Yin MWang ENg CXiao R(2024)Lies, Deceit, and Hallucinations: Player Perception and Expectations Regarding Trust and Deception in GamesProceedings of the CHI Conference on Human Factors in Computing Systems10.1145/3613904.3642253(1-15)Online publication date: 11-May-2024