Vizalgo

Km Abha Dohre[1], Prof. Kaveri Kari [2], Rucha Barbole[3],
Rutvik Sutar[4], Manvi Saini[5]
[1, 2, 3, 4, 5]Dept of Information Technology Engineering
[1, 2, 3, 4, 5] Genba Sopanrao Moze College of Engineering,Balewadi.

Abstract- *Algorithm visualization illustrates how algorithms work in a graphical way. It mainly aims to simplify and deepen the understanding of algorithms operations. Within the paper, we discuss the possibility of enriching the standard methods of teaching algorithms, with algorithm visualizations. As a step in this direction, we introduce the Algorithm visualizer platform, present our practical experiences and describe possible future directions, based on our experiences and exploration performed by means of a simple questionnaire*

Keywords- Graphical Way, Algorithms Operation

I. INTRODUCTION

Understanding Data Structures and Algorithms (DSA), which includes the arrangement of algorithm theory, is a very challenging task in the computer science field one of the most important subjects, but . DSA is due to its abstract character, it is also one of the most difficult to master. The difficulty is generally due to the algorithm, which is derived from dynamic step bystepprocedures.It is broadly seen that algorithm visualizations can give a strong opti on in contrast to static composed presentations (from course books), or verbal portrayals upheld by illustrations (from lectures). Algorithm Visualizations can improve the understanding of fundamental Data Structures and Algorithms. The purpose of this project will be to tackle these issues by extensive research, understanding and studying of various algorithms and developing a website using react and nodejs that provide a superior client experience alongside a wide scope of elements . The venture will likewise plan to end up being a superior instrument for the teaching of Data Structures and Algorithms by utilizing custom visualizations and animations. To better understand and grasp the power of algorithms, students need to understand the algorithms at a low level, which is not possible in today's learning methods. With the rise in the online education system during the pandemic.

II. SYSTEM ARCHITECTURE

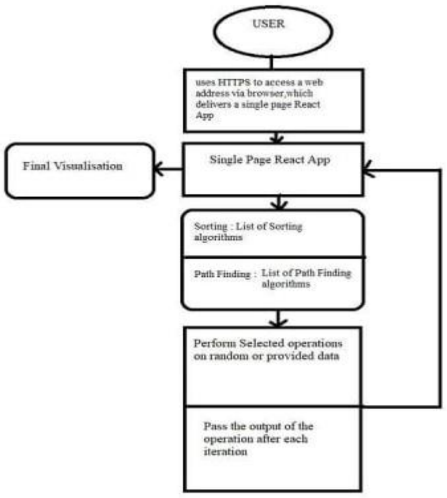

USER
uses HTTPS to access a web address via browser,which delivers a single page React App
Single Page React App
Final Visualisation
Sorting : List of Sorting algorithms
Path Finding : List of Path Finding algorithms
Perform Selected operations on random or provided data
Pass the output of the operation after each iteration

Fig: System Architecture

III. IMPLIMENTATION

Path Finding Algorithms

In Path Finding Algorithms, the user can choose from a variety of mazes available under the Generate Maze button or the user can build its own maze button or the user can build its own maze through the interactive platform.then the user can select from various.

A.Dijkstra's Algorithm

The basic process is to assign each node at a distance value, at first set to zero for the initial node and infinity for all other nodes. All nodes are marked as unvisited and the initial, node is marked as the current node. All nodes that are neighbours to the current node are examined and their distance D from the initial node is calculated through the current node is calculated. If this new distance D is less than the previously recorded distance D for that node, the new distance value replaces the old distance value for that node. When all neighbours of the current node are examined, the current node is marked as visited and will not be looked at again. The neighbour node with the lowest distance value is marked as the new current node and the process repeats until the target is marked as visited or all nodes are marked as visited without the target being found .

Algorithms Pseudocode

1. Mark the ending vertex with a distance of zero. Designate this vertex as current.
2. Find all vertices leading to the current vertex. Calculate their distances to the end. Since we already know the distance the current vertex is from the end, this will just require adding the most recent edge. Don't record this distance if it is longer than a previously.
3. Mark the current vertex as visited. We will never look at this vertex again.
4. Mark the vertex with the smallest distance as current, and repeat from step 2.

B. A* Algorithm

For every node in the graph, A* maintains three values: f(x), g(x), and h(x). g(x) is the distance, or cost, from the initial node to the node currently being examined. h(x) is an estimate or heuristic distance from the node being examined to the target.

Algorithms Pseudocode

1. Place the starting node in the OPEN list.
2. Check if the OPEN list is empty or not, if the list is empty then return failure and stops.
3. Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node n is goal node then return success and stop, otherwise
4. Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute evaluation function for n' and place into Open list.
5. Else if node n' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(n') value.
6. Return to Step 2.

C. Breadth-First Search

Breadth-first search was discovered by Moore in the context of finding paths through mazes. BFS is a The graph traversal algorithm to explore a tree or a graph efficiently. The algorithm starts with an initial node (root node) and then proceeds to explore all the nodes adjacent to it, in a breadth-first fashion, as opposed to depth-first, which goes down a particular branch till all the nodes in that branch are visited. Put simply, it traverses the graph level-wise, not moving down a level till all the nodes in that level are visited and marked. It operates on the first-infirst-out (FIFO) principle and is implemented using a queue data structure. Once a node is visited, it is inserted into a queue. Then it is recorded and all its children's nodes are inserted into the queue. This process goes on till all the nodes in the graph are visited and recorded.

Algorithms Pseudocode

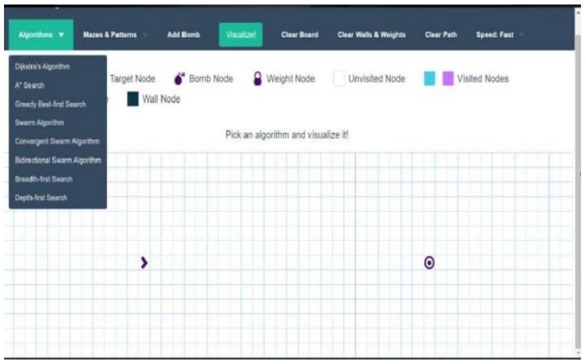
1. Declare a queue and insert the starting vertex.
2. Initialize a visited array and mark the starting vertex as visited
3. Follow the below process till the que ue becomes empty
4. Remove the first vertex of the queue.
5. Mark that vertex as visited
6. Insert all the unvisited neighbours of the vertex into the queue

D. Depth First Search

A version of the depthfirst search was investigated in the 19th century by French mathematician Charles Pierre Trémaux as a strategy for solving mazes. The DFS search begins starting from the first node and goes deeper and deeper, exploring down until the targeted node is found. If the targeted key is not found, the search pat h is changed to the path that was stopped exploring during the initial search, and the same procedure is repeated for that branch. The spanning tree is produced from the result of this search. This tree method is without the loops. The total number of node s in the stack data structure is used to implement DFS traversal.

Algorithms Pseudocode

1. Create a recursive function that takes the index of the node and visited array.
2. Mark the current node as visited and print the node.
3. Traverse all the adjacent and unmarked nodes and call the recursive function with the index of the adjacent.



IV. CONCLUSION

The Proposed System offers a complete perspective of visualizations for sorting and path Finding algorithms so far. Also, the web-based platform can run on small devices while providing the feature to visualize at one's own pace.

As a future scope, more Algorithms for Trees, Graphs, Linked List and many more can be added. To empower learning of the learner concept of community learning through forums, discussion and user-based feedback can be added. The objective of the platform is to reduce the fear level in the minds of learners especially students regarding Algorithms and Data Structures.

REFERENCES

[1] S. Hansen, N. H. Narayanan, and M. Hegarty, "Designing Educationally Effective Algorithm Visualizations", Journal of Visual Languages and Computing, vol. 13, no. 3, 2002, pp. 291-317

[2] Furcy, David. (2009). JHAVEPOP: visualizing linked-list operations in C++ and Java. Journal of Computing Sciences in Colleges. 25. 32-41

[3] Halim, S. (2011). VisuAlgo - visualising data structures and algorithms through animation. VisuAlgo. https://visualgo.net/en [12] Halim, S., Halim, F. (2011). Competitive Programming 2: This increases the lower bound of programming contests. Available: http://www.lulu.com

[4] Supli, A. A., &Shiratuddin, N. (2017). Designing algorithm visualization on mobile platforms: The proposed guidelines. https://doi.org/10.1063/1.5005943

[5] [Aniket B. Ghandge, Bhagyashree P., Hrithik R., Prateek S, ParmodB.(2021).AlgoAssist: Algorithm Visualizer and Coding Platform for Remote Classroom Learning.2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP 2021).http://dx.doi.org/10.1109/ICCCSP52374.2021.9465 503