VASIREDDY VENKATADRI
INSTITUTE OF TECHNOLOGY

(Autonomous)

ALGO-VISUALIZER

(Academic Year: 2020-21)

Project Guide

N Brahma Naidu Sir

Team Members

1. Shaik Abdul Muheet (19BQ1A05J8)

2. Vaseem Naazleen Shaik (19BQ1A05L1)

3.Thippabathina Pavan Kumar (19BQ1A05M8)

4.Vellaturi Phannindra Mouli (19BQ1A05O2)

Department of Computer Science and Engineering

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY, NAMBUR

Table of Content

PROBLEM STATEMENT

" One of the most difficult topics that students feel is GRAPH THEORY. There are several complex graph algorithms like BFS, DFS, Dijkstra's, A* etc. Our goal is to make them easy by providing beautiful visualisations. "

INTRODUCTION:

ABOUT THE PROJECT:

DEFINITION:

- Path finding algorithms build on top of graph search algorithms and explore routes between nodes, starting at one node and traversing through relationships until the destination has been reached.
- These algorithms find the cheapest path in terms of the number of hops or weight
- Weights can be anything measured, such as time, distance, capacity, or cost.

ALGORITHMS:

- BFS
- Bi-directional BFS
- DFS
- Dijkstra's
- Greedy Best first search
- A star

PROJECT NEEDS AND REQUIREMENTS:

- React Js , CSS, BootStrap

PROJECT TYPE:

- Web Application

SOFTWARE AND HARDWARE REQUIREMENTS :

Softwares used:

- NPM(Node Package Manager) or YARN
- Web Browser (for debugging and running the project)
- Suitable Editor (preferably VS Code, Notepad ++)

Hardware Requirements:

- RAM: 8GB (recommended)
- Processor: any updated 32/64 bit processor
- Diskspace: 4GB (recommended)

Additional Tools used:

- Github

IMPLEMENTATION:

MODULES :

Dependencies

- React
- Font-awesome
- React-dom
- React-popper
- React-scripts
- Reactstrap
- Bootstrap

React

We need to import react from the node-modules. Importing react from node_modules to create the components(createElement) in the jsx/js file

Font-Awesome

Font Awesome is a font and icon toolkit based on CSS and Less. Font Awesome is the most popular **way to add font icons to your website**

React-dom

ReactDOM is **a package that provides DOM specific methods** that can be used at the top level of a web app to enable an efficient way of managing DOM elements of the web page.

React-popper

react-popper is **the official React wrapper to let you use Popper in your React** projects. It exposes an API very similar to the one of the Vanilla JavaScript library, but it provides some useful additions to better integrate with React.

React-script

react-scripts is **a set of scripts from the create-react-app starter pack**. create-react-app helps you kick off projects without configuring, so you do not have to setup your project by yourself.

Bootstrap

Bootstrap is a **potent front-end framework used to create modern websites and web apps**. It's open-source and free to use

Components developed during this project

We have categorized the project work into three Folder Mainly:

- Algorithms - contains the Algorithms implementation in project
- Components - contains Components that implement features for visualization of the algorithms
- Css - Helps in styling the website

Algorithms

- Asearch.js
- Bfs.js
- biDirectionalBfs.js
- Dfs.js
- Dijkstra.js
- greedyBfs.js

Components

- GridBlockComponent.js
- GridLayoutComponent.js
- HeaderComponents.js
- PriorityQueue.js

CSS

- gridblock.css

ALGORITHMS

Asearch.js :

In this file,the A*search algorithm is implemented.

Bfs.js:

In this file, the Breadth First Search algorithm is implemented.

Dfs.js:

In this file, the Depth First Search algorithm is implemented.

Dijkstra.js:

In this file, Dijkstra's algorithm is implemented.

Bidirectional.js:

In this file, the Bi directional bfs algorithm is implemented.

greedyBfs.js:

In this file, the Greedy Best First Search algorithm is implemented.

COMPONENTS

GridLayoutComponent.js:

In this file,the code for the following functionalities are implemented: they are random wall grid generation, random weight grid generation,random start and end node selection, execution speed controller, all the connections for proper executions of algorithms and display of shortest path and visited nodes for all algorithms. These all are passed as props to the HeaderComponents.js for the proper rendering.

GridBlockComponent.js:

In this file, Grid is implemented with functionalities of the onclick events for the walls and selection of start and end nodes and icons for the start and are also defined here.

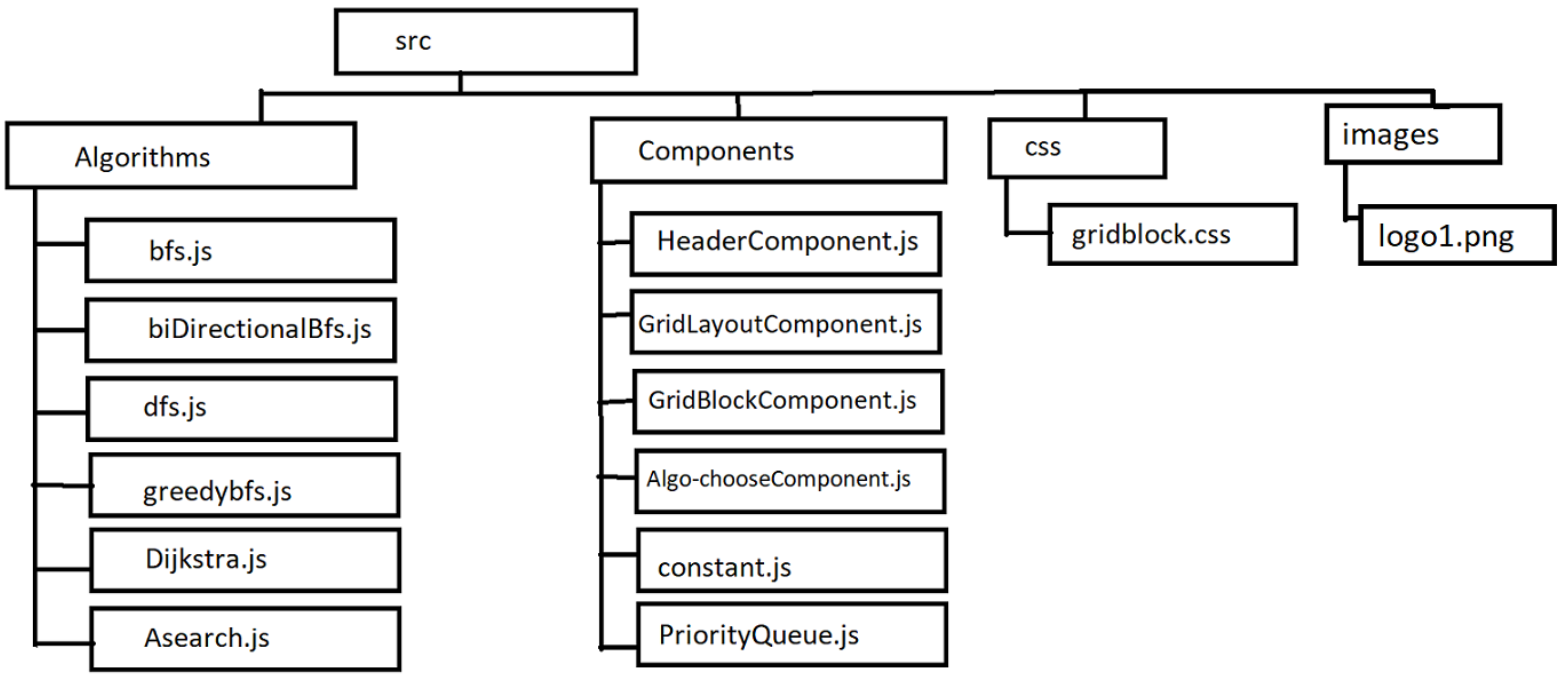
HeaderComponent.js:

In this file, the UI of the website is designed properly with functionalities passed from the GridLayoutComponents.js which includes the nav header with algorithms, random wall and weight generation,speed chooser etc.

PriorityQueue.js:

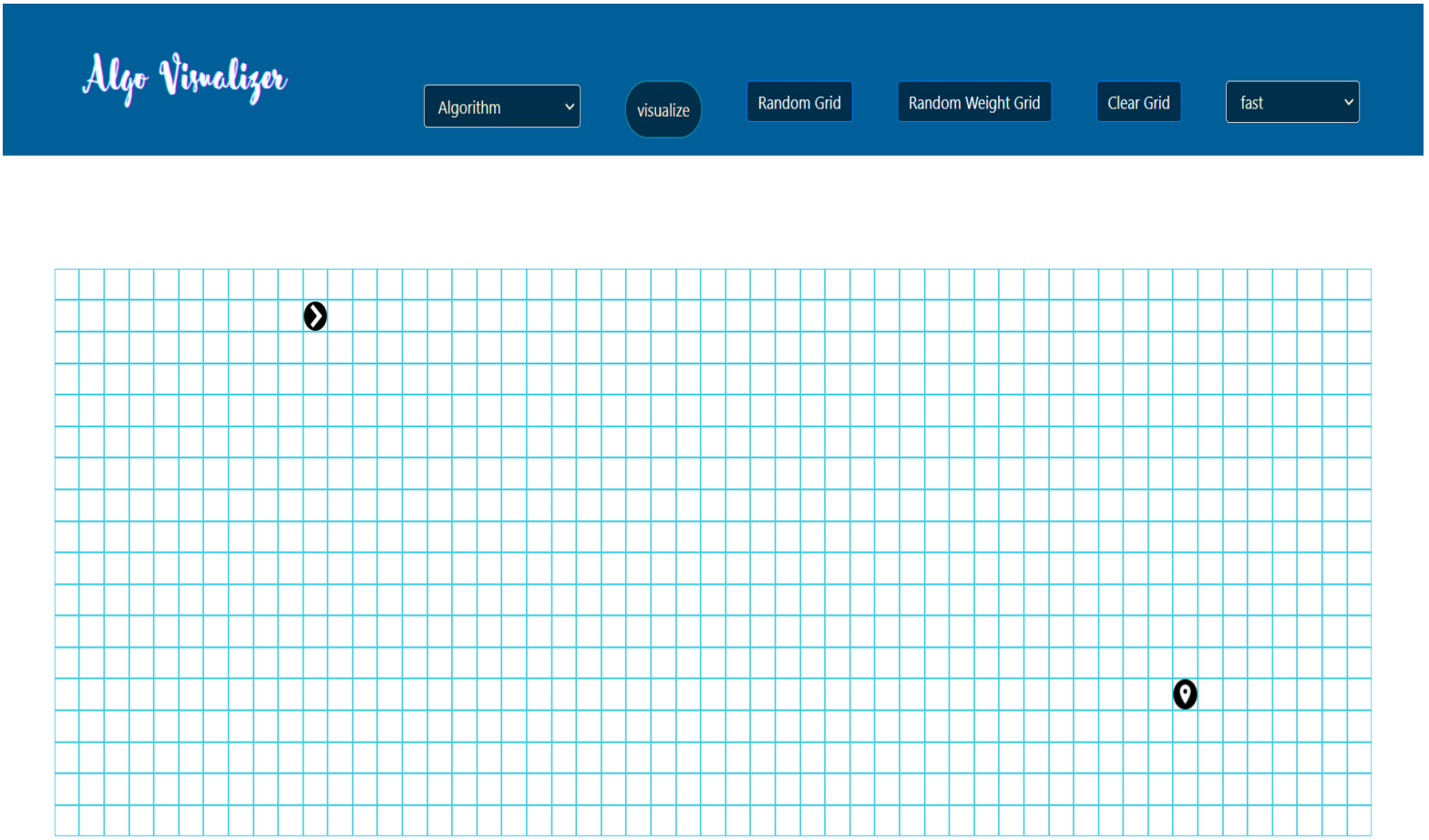
In this file, the code for the implementation of priority queue is used which is useful in the A* search algorithm.

FLOW CHART

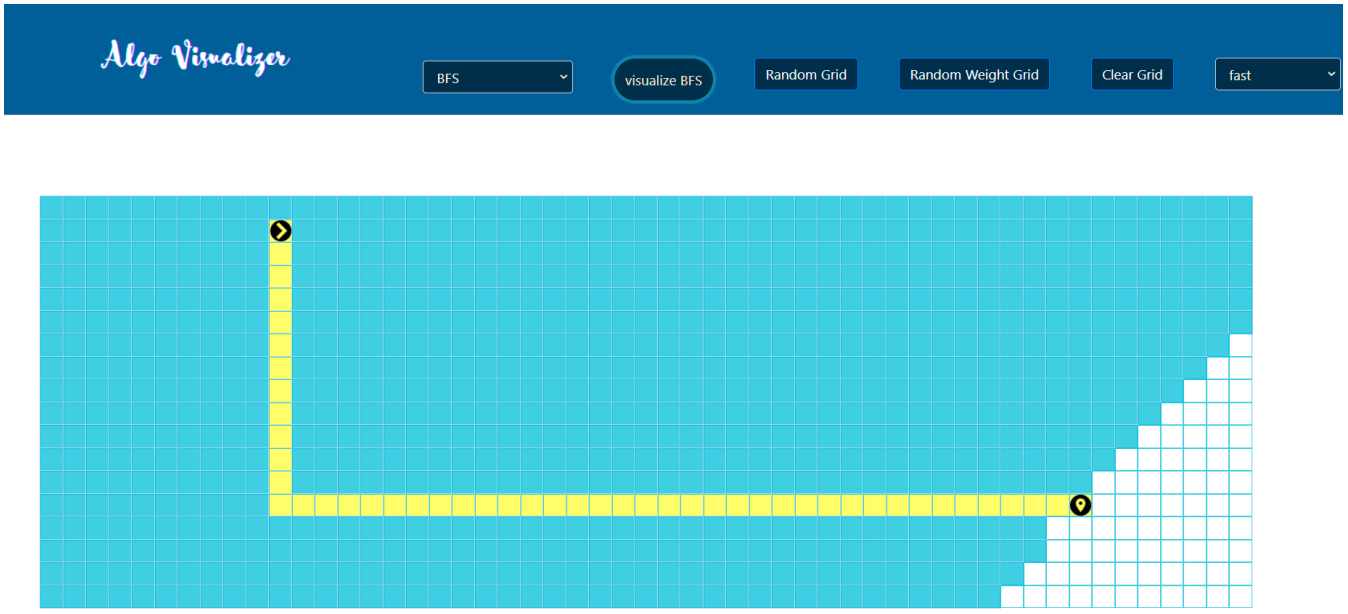


RESULT

WEBSITE MAIN PAGE

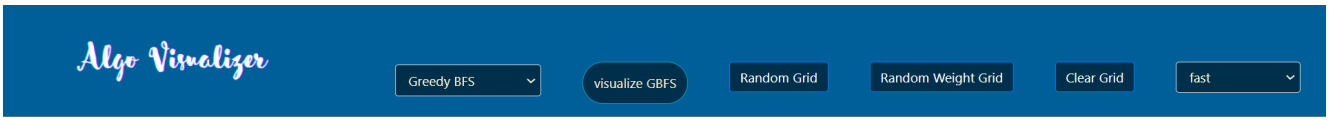


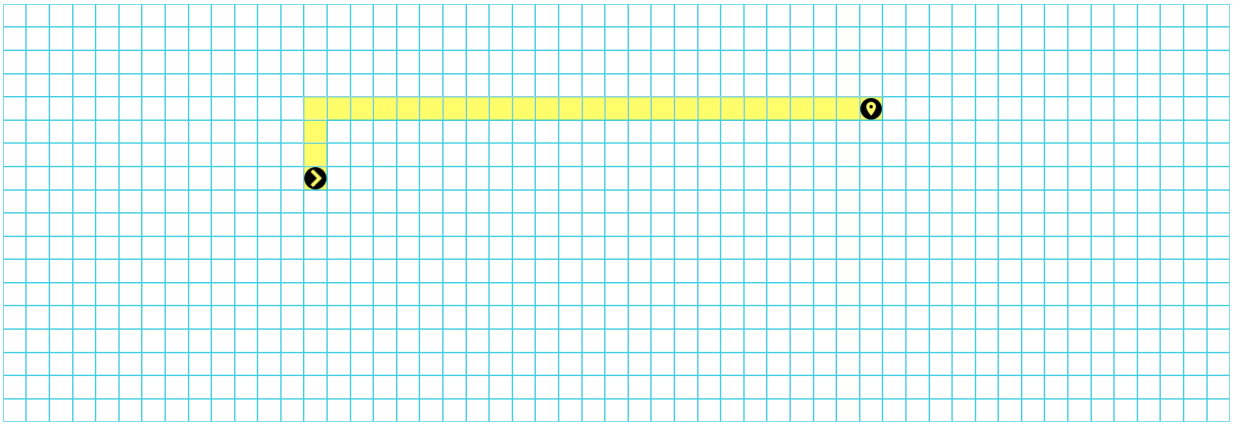
Breadth First Search



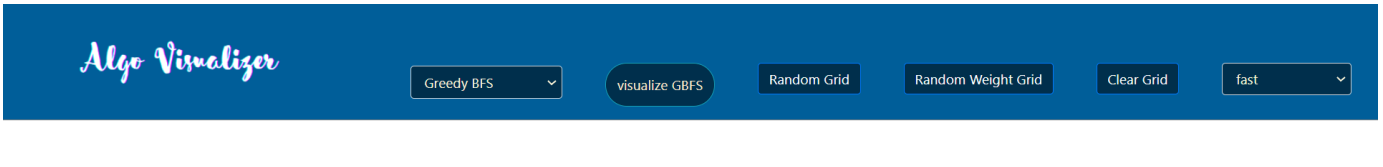
Breadth First Search with Walls

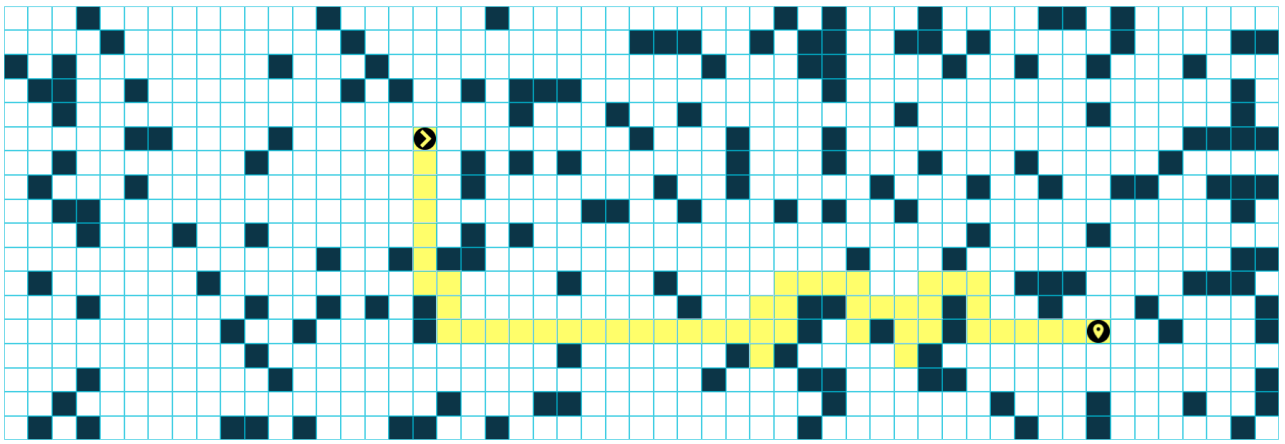
Greedy Breadth First Search



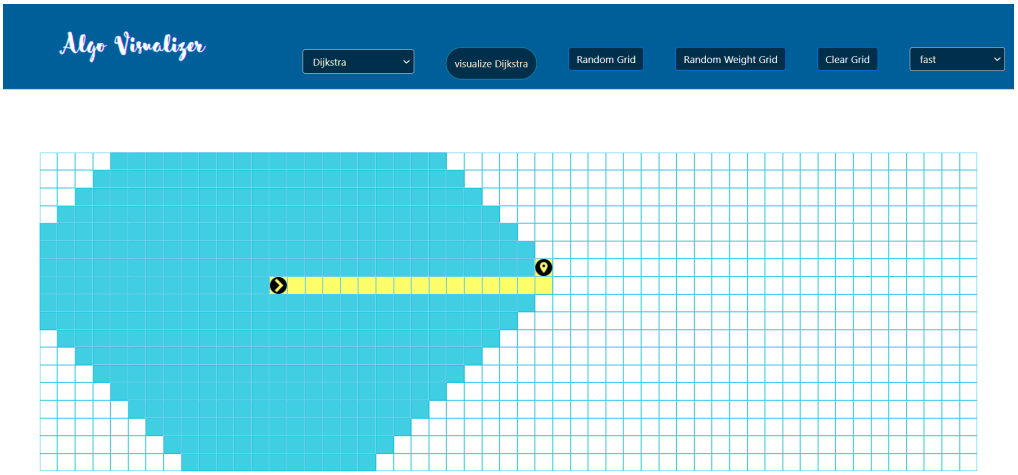


Greedy Breadth First Search with Walls

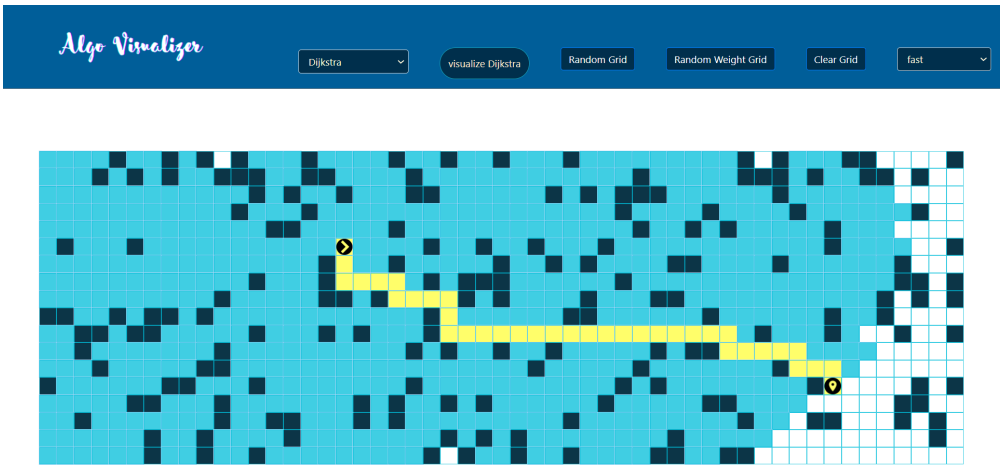




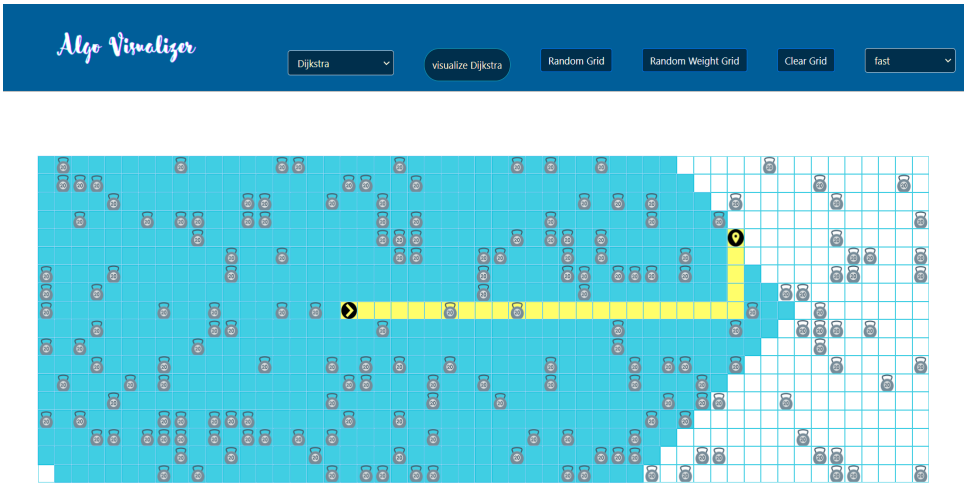
Dijkstra's



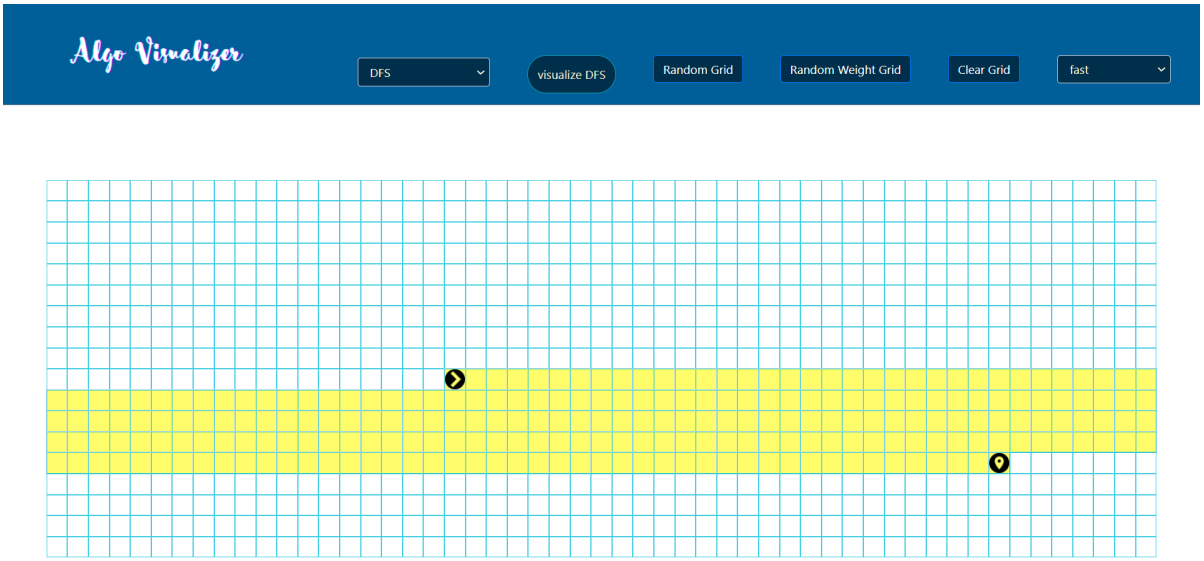
Dijkstra's Algorithm With Walls



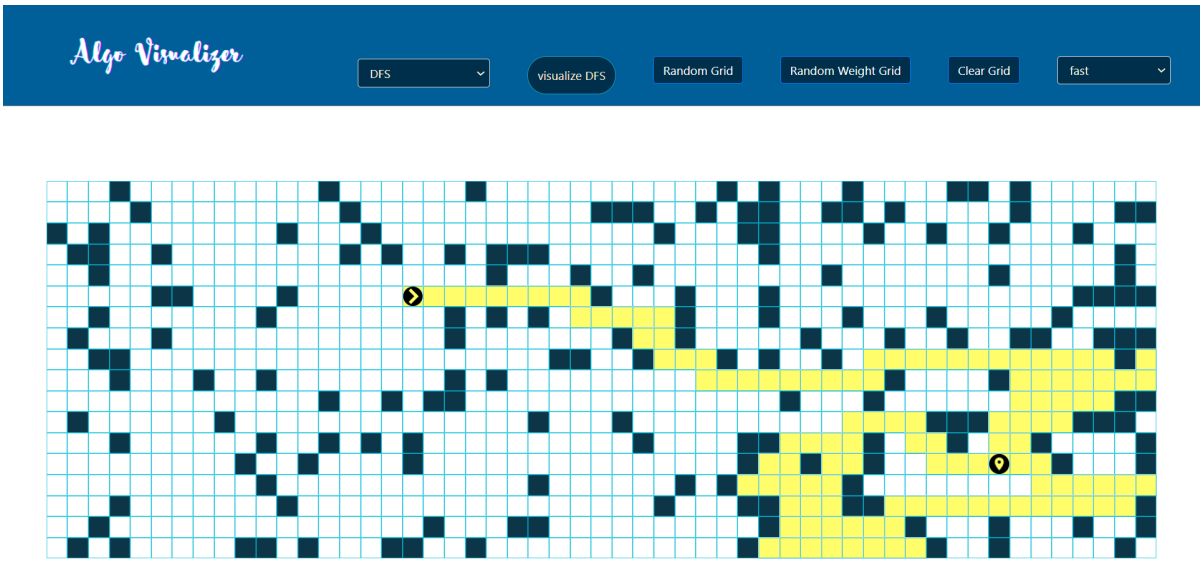
Dijkstra's Algorithm with Weights



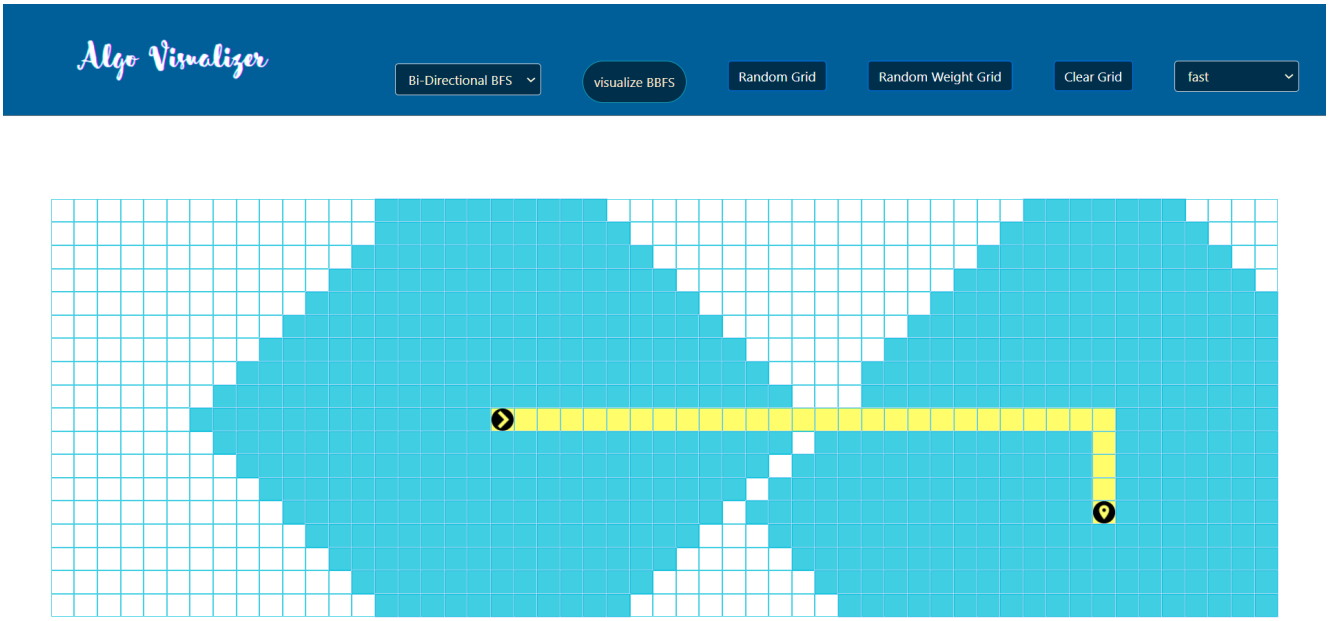
Depth First Search



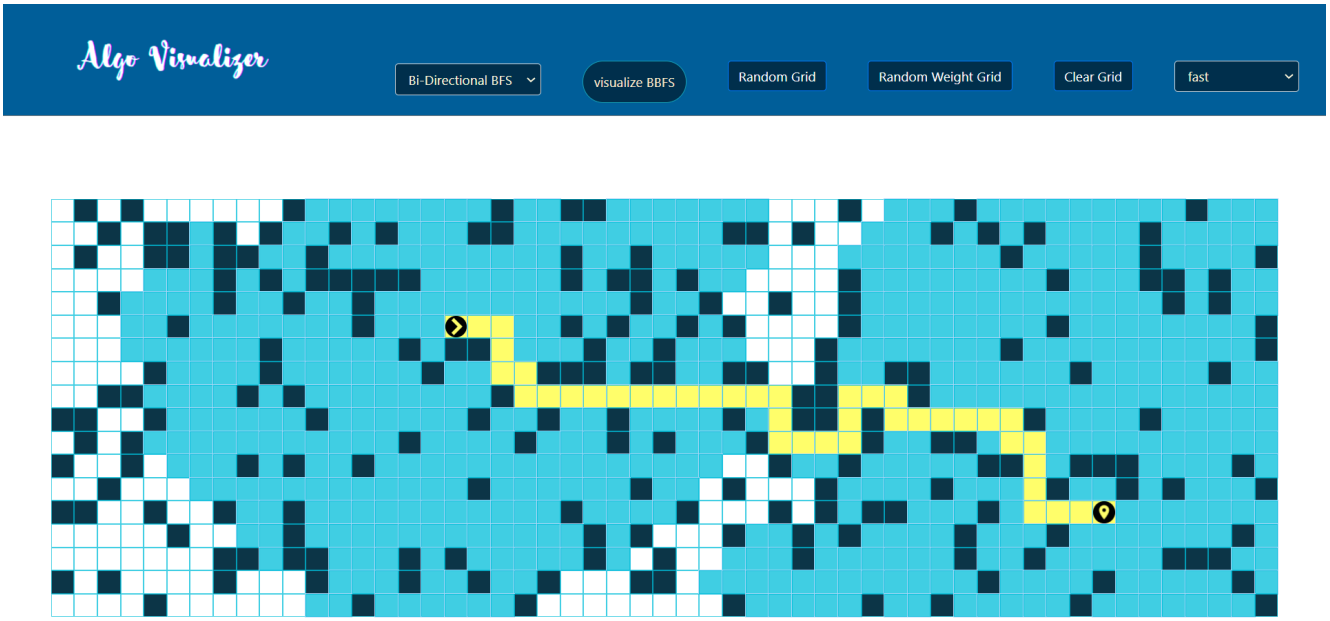
Depth First Search with Walls



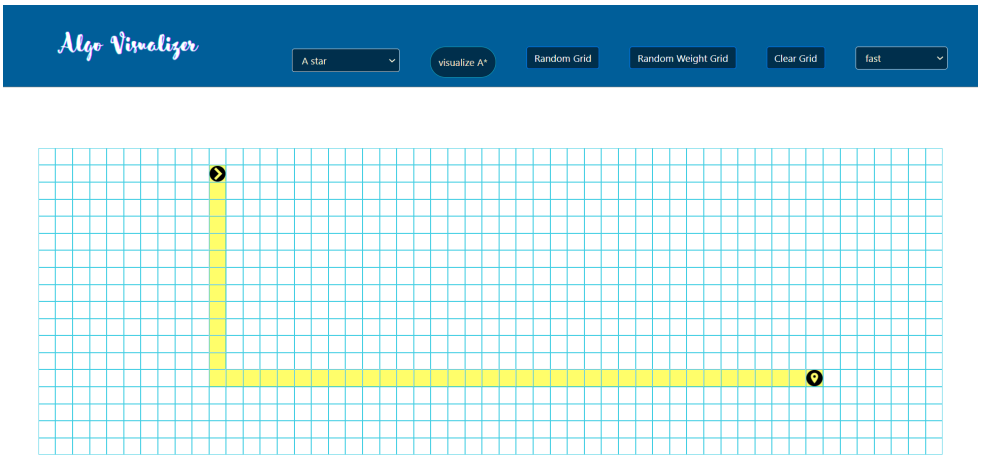
Bi-directional Breadth First Search



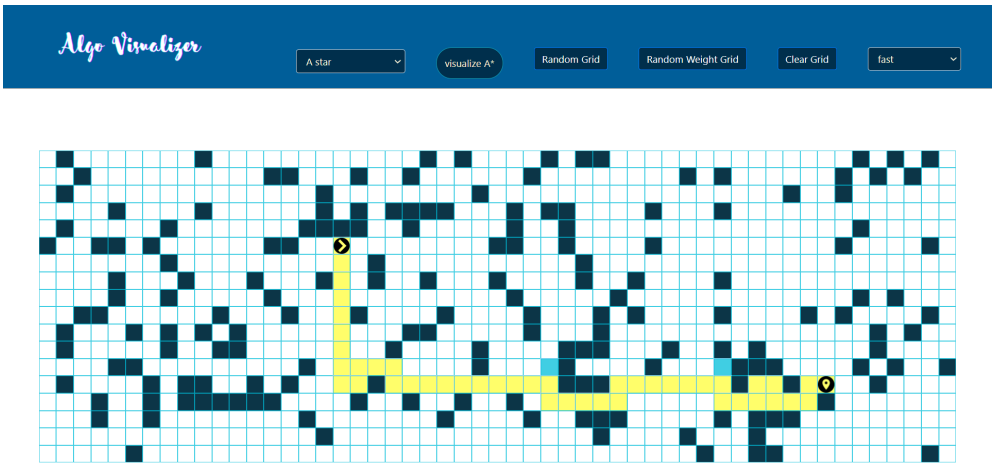
Bi-directional Breadth First Search with Walls



A Star



A Star with Walls



A Star with Weights