Team – One Click

Lowe's hackathon submission

-Internal store guidance system-

Problem Statement

- Whenever you go to a store/mall for shopping, time is wasted in finding and navigating an item/shelf. There is a need to make the navigation simple and fast.

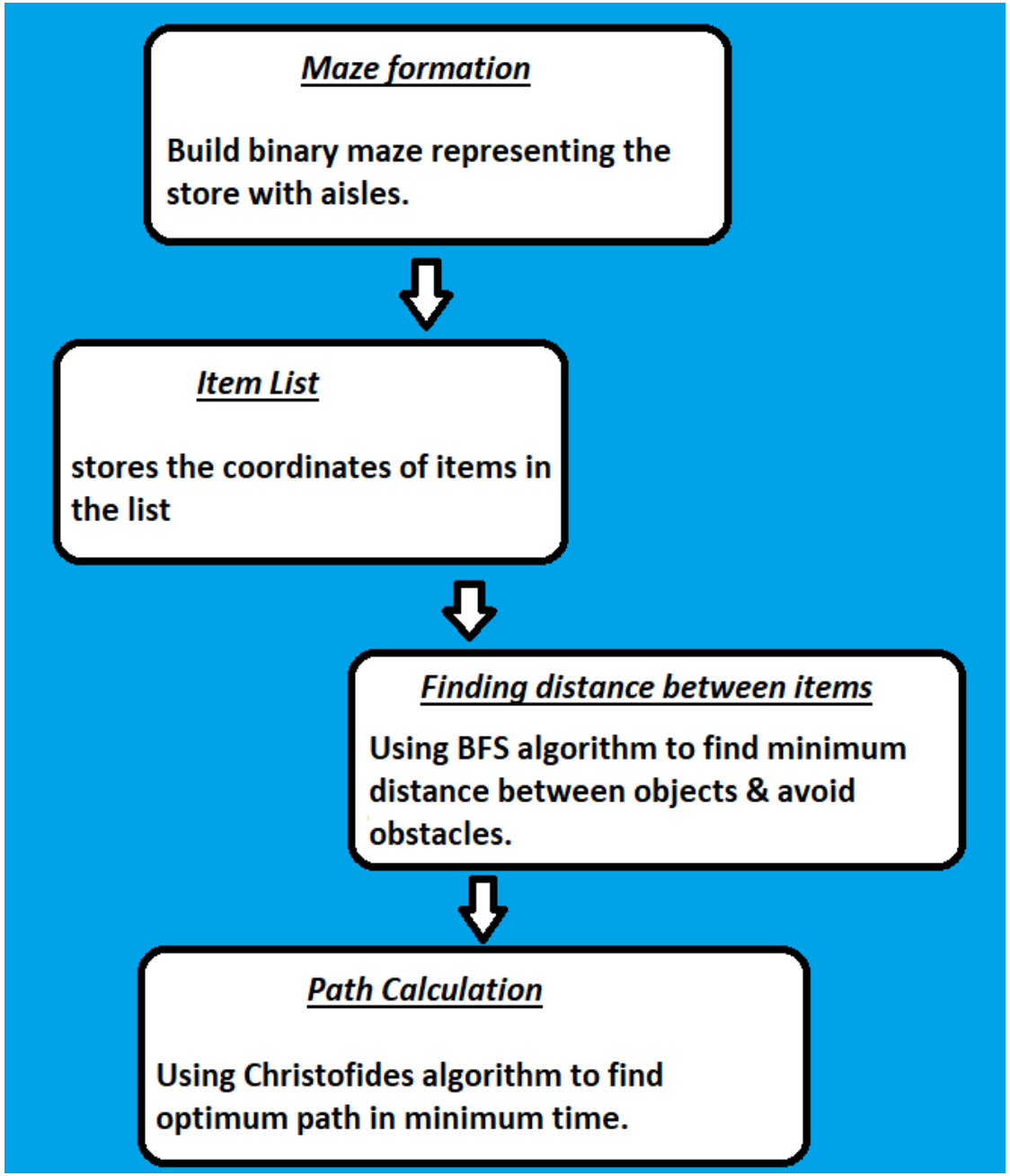
- Also, due to coronavirus many stores are operating under-capacity and often gets overcrowded which increases risk of infection.

- We need to have a solution that can enable user find items quickly so that purchases are faster, efficient and avoid overcrowding.

Solution Approach

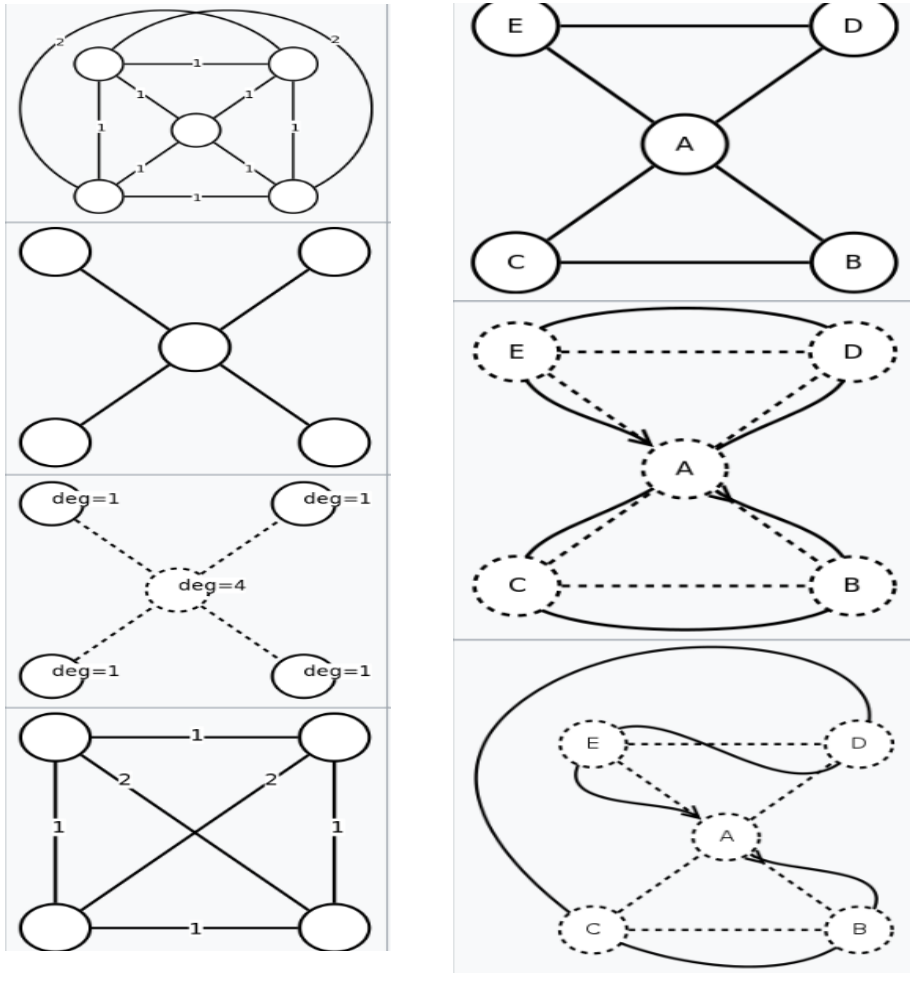
- **Christofides algorithm** finds optimal path in minimum amount of time.

- **Breadth First Search (BFS)** Algorithm find the distance in all cases in the fastest time as compared to **A*** and **DFS algorithm**.

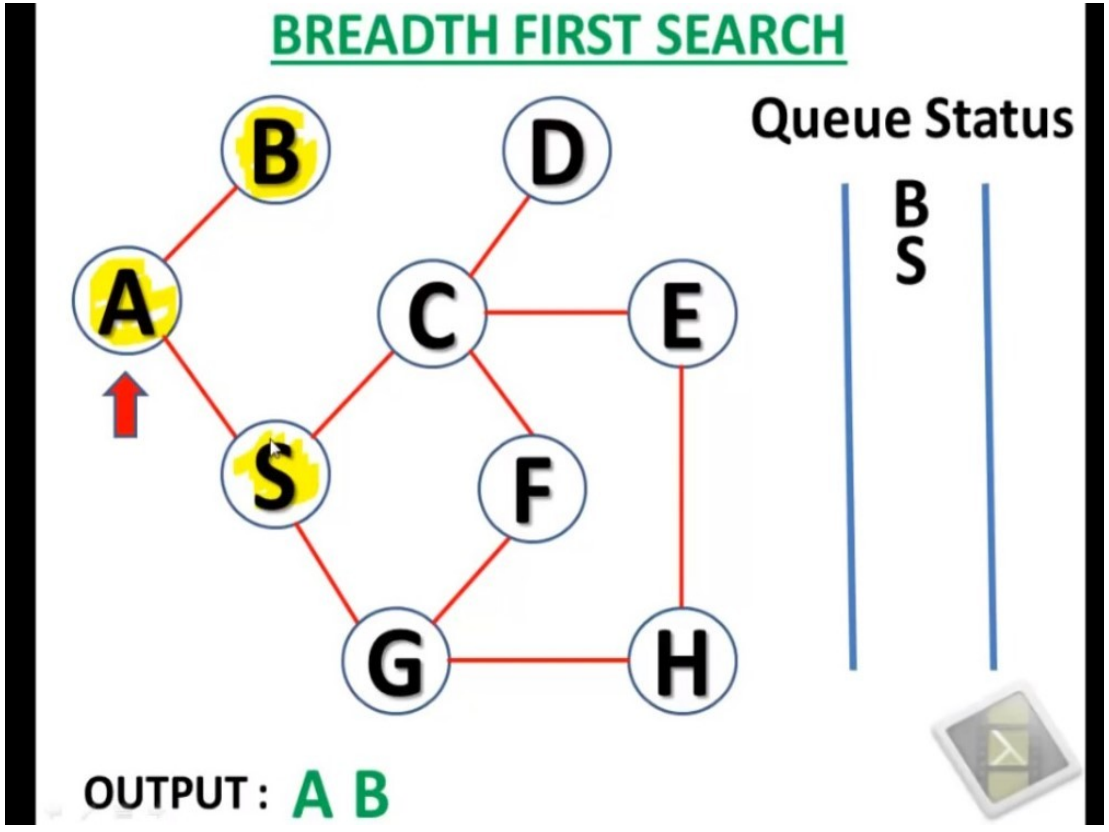


Algorithms Used

- **Christofides Algorithm**



BFS Algorithm



Additional Improvements

- ✓ Apart from the core functionality we have added *Analytics* which are useful for store owners.
- ✓ Store owners can measure :- *footfall, retention rate, demographics, lifetime value of customer, Impulse purchases and gender*.
- ✓ *Modified Christofides Algorithm* to introduce entry and exit points which makes navigation easy and optimized.
- ✓ *Improved the UI* and made the process of creating a store easy with just a mouse click.

Assumptions while designing

- Currently we support only horizontal and vertical aisles.
- Distance between consecutive aisles must be at least 1 unit to allow path between them.
- For ease of testing make a store with dimensions 35x35. But any rectangular store will do.
- While adding coordinates of items add the item one unit to left/right(for vertical aisle) or above/below(for horizontal aisle) of aisle. Do not add item directly on it.
- This only needs to be done by store manager and not by the customer.

Technology and Tools

Software used

- We have used **python** for building our solution.
- **JSON** & **SQL lite** for database.
- We chose python because it is easy to use and powerful.

Hardware requirements

- We recommend a computer with minimum **8GB RAM**.
- **1 TB HDD**, SSD would be an added advantage.

Challenges Faced

- **GUI challenge** – Using inbuilt functions(buttons) caused the program to lag and give a bad user experience.
- **Algorithm Dilemma** – Out of the many algorithms out there it was a challenge to choose one which was optimal for use.
- **Hick-ups with python -** It has call by object unlike other languages like C++ which allow us to decide when to call by value or call by reference.
- **Databases -** Tried SQLlite, config.ini but it had a problem with storing complex matrices.

Learnings from the challenges

1. **Fixed the GUI** challenge by building the functionality from scratch which doesn't lag now. Also, were able to add zoom function.

2. Learned about various algorithm and chose **Christofides and BFS algorithm** to find optimal solution.

3. Learned about **JSON** and its implementation which helped in storing and retrieving complex matrices and data.

4. Learned about the application and importance of various metrics while implementing the **analytics** part for the store.

Build Instruction

1) Download Jupyter notebook as this has all required libraries inbuilt.

2) Copy code in a new python notebook.

3) Run code

OR

You can just run the .exe file which can easily be deployed.

Link:
https://drive.google.com/open?id=1evUvJUGzpcdSeAFJMPB6LwXDqqkovL38

Team

Arnav Khandekar
B.Tech - ECE
3rd year
IIIT Nagpur

Dhanraj Mahurkar
B.Tech - ECE
3rd year
IIIT Nagpur

In a nutshell

- **Our solution allows customers to locate an item easily and navigate the store quickly.**
- It **saves time**, **avoids overcrowding** and provides **user analytics** useful for the store owner**.**
- It uses **Christofides and BFS Algorithm** to find shortest path in minimum time and avoids obstacles**.**
- **Simple UI -** Anyone can build a store easily with a mouse click.