IDEATHON

CSL2020: DATA STRUCTURES & ALGORITHMS

PROJECT TITLE

SHORTEST PATH ALGORITHMS FOR CAMPUS NAVIGATION

TEAM MEMBERS

RAHUL GARG(B22CH025)

YOGESH JAJORIA(B22ME073)

DHRUV KUMAR SINGH(B22CH010)

SACHIN CHOUDHARY(B22ME056)

INSTRUCTOR

DR. SUCHETANA CHAKRABORTY

MENTORS

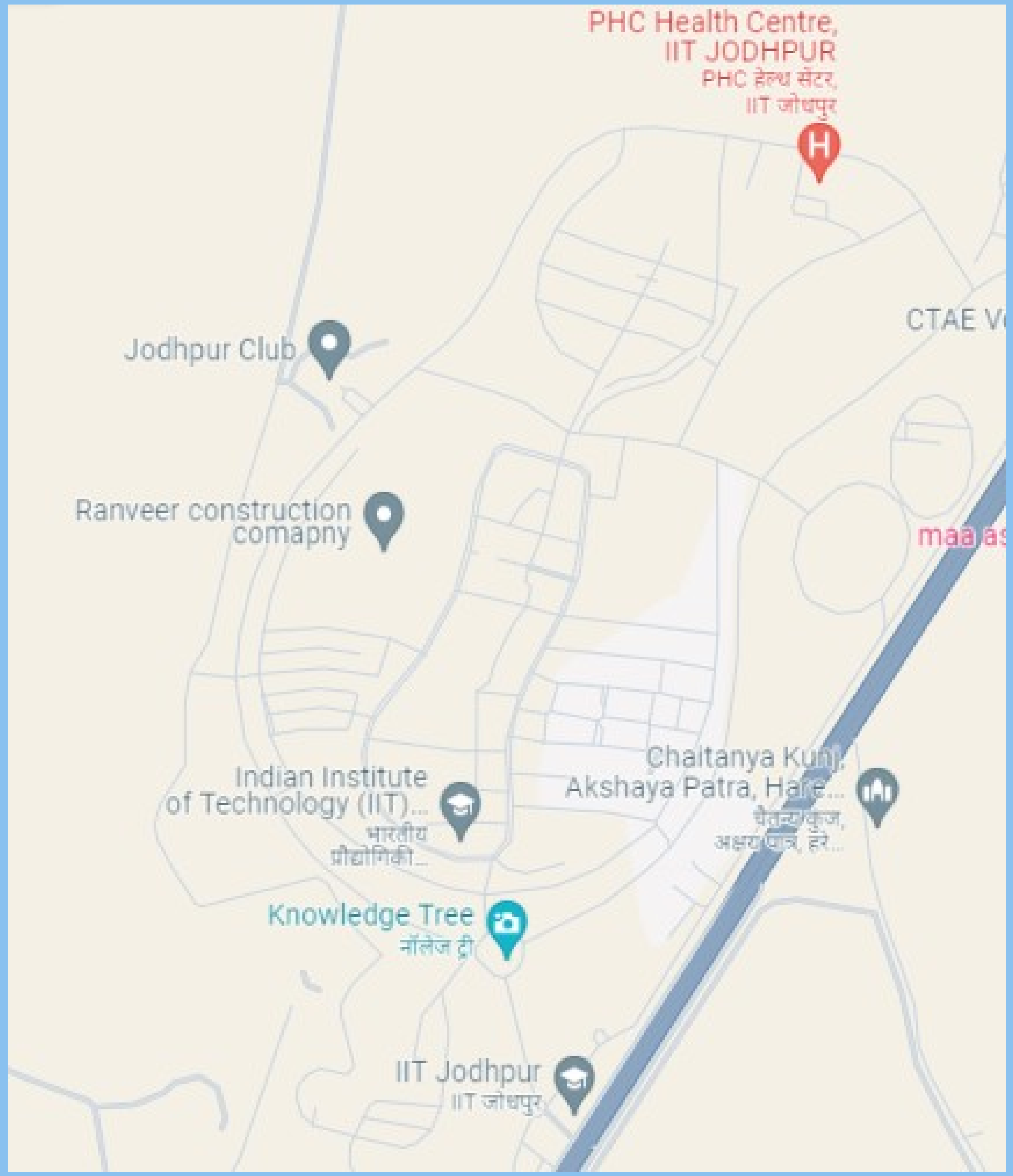
DIXIT DUTT BOHRA

SHUBHAM KUMAR

PROBLEM STATEMENT

Finding The Shortest Path For Campus Navigation

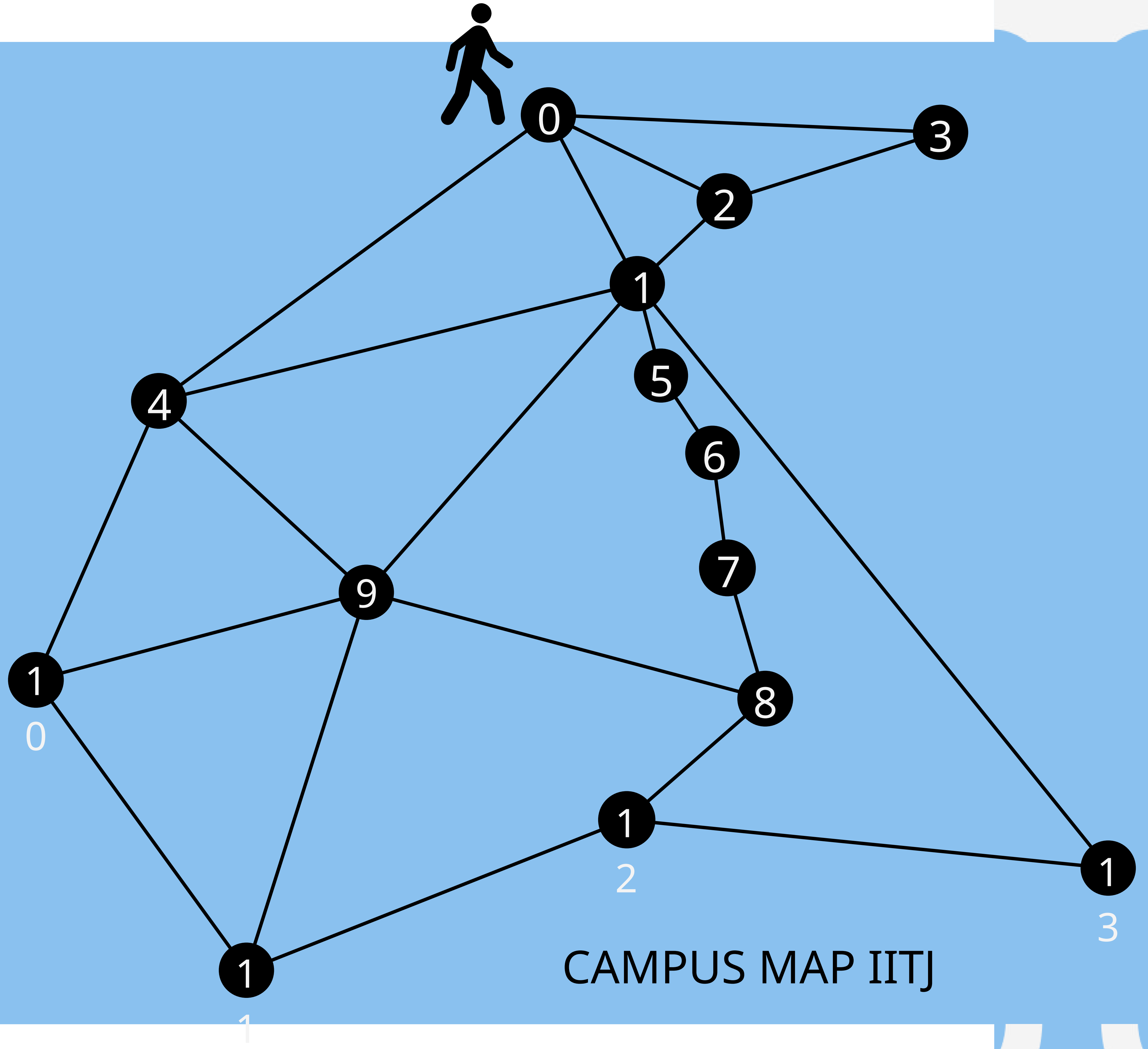
- Getting around the vast campus of IIT Jodhpur can be confusing and time-consuming for students, faculty, and visitors.
- Existing navigation options often don't give the quickest or easiest routes, and they don't consider things like crowded areas or events happening on campus.
- To make campus travel easier, we want to create a navigation system using different algorithms like Dijkstra, bellman Ford, A star, etc.
- This system will find the shortest and fastest paths between different spots on campus.
- Our goal is to simplify campus travel, save time, and make getting around IITJ smoother for everyone.



PROBLEM STATEMENT

USE CASE MAPPING

- 0: Main Gate IIT Jodhpur
- 1: Knowledge Tree
- 2: Kendriya Bhandar
- 3: School of AIDE
- 4: Sports Complex
- 5: Administrative Block
- 6: Library
- 7: Lecture Hall Complex
- 8: Department of Computer Science
- 9: Y3 Hostel (Amaltas)
- 10: Primary Health Centre
- 11: Department of Mechanical Engineering
- 12: Shamiyana
- 13: Jodhpur Club



CAMPUS MAP IITJ

PROBLEM STATEMENT

USE CASE MAPPING

```
int graph[V][V] = {
    {   0,  550,  700,  850, 1600,    0,    0,    0,    0,    0,    0,    0,    0,    0},
    { 550,    0,  600,    0, 1200,  270,    0,    0,    0, 1000,    0,    0,    0, 1400},
    { 700,  600,    0,  900,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0},
    { 850,    0,  900,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0},
    {1600, 1200,    0,    0,    0,    0,    0,    0,    0,  300,  800,    0,    0,    0},
    {   0,  270,    0,    0,    0,    0,  160,    0,    0,    0,    0,    0,    0,    0},
    {   0,    0,    0,    0,    0,  160,    0,  300,    0,    0,    0,    0,    0,    0},
    {   0,    0,    0,    0,    0,    0,  300,    0,  300,    0,    0,    0,    0,    0},
    {   0,    0,    0,    0,    0,    0,    0,  300,    0,  400,    0,    0,  300,    0},
    {   0, 1000,    0,    0,  300,    0,    0,    0,  400,    0,  900,  700,    0,    0},
    {   0,    0,    0,    0,  800,    0,    0,    0,    0,  900,    0,  750,    0,    0},
    {   0,    0,    0,    0,    0,    0,    0,    0,    0,  700,  750,    0,  300,    0},
    {   0,    0,    0,    0,    0,    0,    0,    0,  300,    0,    0,  300,    0,  650},
    {   0, 1400,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,  650,    0}
};
```

- 0: Main Gate IIT Jodhpur
- 1: Knowledge Tree
- 2: Kendriya Bhandar
- 3: School of AIDE
- 4: Sports Complex
- 5: Administrative Block
- 6: Library
- 7: Lecture Hall Complex
- 8: Department of Computer Science
- 9: Y3 Hostel (Amaltas)
- 10: Primary Health Centre
- 11: Department of Mechanical Engineering
- 12: Shamiyana
- 13: Jodhpur Club

PROBLEM STATEMENT

Importance/Relevance:- Inaccurate or inefficient navigation systems can lead to wasted time, frustration, and decreased overall campus experience.

Challenging Nature:-Campus navigation presents unique challenges due to the large and complex layout of the campus, varying levels of pedestrian traffic, dynamic event schedules, and diverse user needs. Traditional navigation methods often fail to address these challenges adequately, resulting in suboptimal routes and user dissatisfaction.

Promise of Data Driven Solutions:-This data-driven approach ensures that navigation decisions are based on up-to-date information, leading to more accurate and responsive navigation experiences for users.

CURRENT STATUS

- At present, navigating the campus of IIT Jodhpur relies primarily on traditional methods such as verbal directions, and online maps like Google Maps.
- While these tools offer some assistance, they often lack the specific details and real-time updates needed to navigate the campus efficiently.
- Additionally, there is a lack of a dedicated campus navigation system that takes into account the unique layout and requirements of IIT Jodhpur.

OUR ALGORITHM

1) A Star Algorithm:-

- A* (A-star) algorithm is a heuristic search algorithm used to find the shortest path between a start node and a goal node.
- It evaluates nodes by combining the cost of reaching the node from the start node (known as g-value) and the estimated cost of reaching the goal node from the current node (known as h-value).

- The algorithm selects nodes with the lowest f-value (sum of g-value and h-value) for expansion, prioritizing nodes that are closer to the goal.
- A* guarantees finding the shortest path if certain conditions like admissible heuristic are met.

2) Dijkstra's Algorithm:-

- Dijkstra's algorithm is a graph search algorithm used to find the shortest path from a single source node to all other nodes in a weighted graph.
- It maintains a priority queue of nodes yet to be processed, with their tentative distances from the source node.
- The algorithm iteratively selects the node with the smallest tentative distance and relaxes its outgoing edges, updating the distances of adjacent nodes if shorter paths are found.
- Dijkstra's algorithm ensures the shortest path to each node is discovered before terminating, making it ideal for finding shortest paths in a graph without negative edge weights.

OUR ALGORITHM

3) Bellman-Ford Algorithm:-

- Bellman-Ford algorithm is a single-source shortest path algorithm that can handle graphs with negative edge weights and detect negative weight cycles.
- It maintains an array of tentative distances from the source node to all other nodes, initially set to infinity.
- The algorithm relaxes all edges repeatedly for a number of iterations, updating the tentative distances if shorter paths are found.
- Bellman-Ford algorithm detects negative weight cycles by performing one extra iteration and checking for further distance updates, which indicate the presence of a cycle.

4) Floyd-Warshall Algorithm:-

- Floyd-Warshall algorithm is a dynamic programming approach used to find the shortest paths between all pairs of nodes in a weighted graph.
- It constructs a matrix where each entry represents the shortest distance between two nodes.
- The algorithm iteratively considers all possible intermediate nodes and updates the shortest path distances if a shorter path is found through the intermediate node.
- Floyd-Warshall algorithm is efficient for finding shortest paths in dense graphs or graphs with negative edge weights.

RESULT

1) Dijkstra Algorithm :-

```
Enter the Source Node: 0
Vertex          Distance from Source          Path
0               0                             0
1               550                           0 -> 1
2               700                           0 -> 2
3               850                           0 -> 3
4               1600                          0 -> 4
5               820                           0 -> 1 -> 5
6               980                           0 -> 1 -> 5 -> 6
7               1280                          0 -> 1 -> 5 -> 6 -> 7
8               1580                          0 -> 1 -> 5 -> 6 -> 7 -> 8
9               1550                          0 -> 1 -> 9
10              2400                          0 -> 4 -> 10
11              2180                          0 -> 1 -> 5 -> 6 -> 7 -> 8 -> 12 -> 11
12              1880                          0 -> 1 -> 5 -> 6 -> 7 -> 8 -> 12
13              1950                          0 -> 1 -> 13
```

2) A Star Algorithm:-

```
Enter the Source Node: 0
Shortest path from 0 to 1 is: 0 -> 1
Shortest path from 0 to 2 is: 0 -> 2
Shortest path from 0 to 3 is: 0 -> 3
Shortest path from 0 to 4 is: 0 -> 4
Shortest path from 0 to 5 is: 0 -> 1 -> 5
Shortest path from 0 to 6 is: 0 -> 1 -> 5 -> 6
Shortest path from 0 to 7 is: 0 -> 1 -> 5 -> 6 -> 7
Shortest path from 0 to 8 is: 0 -> 1 -> 5 -> 6 -> 7 -> 8
Shortest path from 0 to 9 is: 0 -> 1 -> 9
Shortest path from 0 to 10 is: 0 -> 4 -> 10
Shortest path from 0 to 11 is: 0 -> 1 -> 5 -> 6 -> 7 -> 8 -> 12 -> 11
Shortest path from 0 to 12 is: 0 -> 1 -> 5 -> 6 -> 7 -> 8 -> 12
Shortest path from 0 to 13 is: 0 -> 1 -> 13
```

3) Bellman Ford Algorithm:-

```
Enter the Source Node: 0
Vertex          Distance from Source          Path
0               0                             0
1               550                           0 -> 1
2               700                           0 -> 2
3               850                           0 -> 3
4               1600                          0 -> 4
5               820                           0 -> 1 -> 5
6               980                           0 -> 1 -> 5 -> 6
7               1280                          0 -> 1 -> 5 -> 6 -> 7
8               1580                          0 -> 1 -> 5 -> 6 -> 7 -> 8
9               1550                          0 -> 1 -> 9
10              2400                          0 -> 4 -> 10
11              2180                          0 -> 1 -> 5 -> 6 -> 7 -> 8 -> 12 -> 11
12              1880                          0 -> 1 -> 5 -> 6 -> 7 -> 8 -> 12
13              1950                          0 -> 1 -> 13
```

4) Floyd-Warshall Algorithm:-

Shortest distances between every pair of vertices:

0	550	700	850	1600	820	980	1280	1580	1550	2400	2180	1880	1950
550	0	600	1400	1200	270	430	730	1030	1000	1900	1630	1330	1400
700	600	0	900	1800	870	1030	1330	1630	1600	2500	2230	1930	2000
850	1400	900	0	2450	1670	1830	2130	2430	2400	3250	3030	2730	2800
1600	1200	1800	2450	0	1460	1300	1000	700	300	800	1000	1000	1650
820	270	870	1670	1460	0	160	460	760	1160	2060	1360	1060	1670
980	430	1030	1830	1300	160	0	300	600	1000	1900	1200	900	1550
1280	730	1330	2130	1000	460	300	0	300	700	1600	900	600	1250
1580	1030	1630	2430	700	760	600	300	0	400	1300	600	300	950
1550	1000	1600	2400	300	1160	1000	700	400	0	900	700	700	1350
2400	1900	2500	3250	800	2060	1900	1600	1300	900	0	750	1050	1700
2180	1630	2230	3030	1000	1360	1200	900	600	700	750	0	300	950
1880	1330	1930	2730	1000	1060	900	600	300	700	1050	300	0	650
1950	1400	2000	2800	1650	1670	1550	1250	950	1350	1700	950	650	0

IDEA

Implemented Idea for Campus Navigation at IIT Jodhpur

In our quest to enhance campus navigation at IIT Jodhpur, we have implemented a comprehensive solution leveraging advanced algorithms such as A* (A-star), Dijkstra, Floyd-Warshall, and Bellman-Ford. Each algorithm serves a specific purpose in calculating the shortest paths and providing node-wise routes across the campus.

A (A-star) Algorithm*: A* is used for finding the shortest path between two nodes (locations) on the campus map. It efficiently combines the advantages of both uniform cost search and greedy best-first search to provide an optimal path while considering factors such as distance, estimated cost, and heuristic information.

Dijkstra Algorithm: Dijkstra's algorithm is employed to calculate the shortest paths from a single source node (starting point) to all other nodes on the campus map. It systematically explores the neighboring nodes, updating their shortest distance from the source node until all nodes have been visited, thereby determining the optimal routes.

Floyd-Warshall Algorithm: The Floyd-Warshall algorithm is utilized to calculate the shortest paths between all pairs of nodes on the campus map. This algorithm efficiently handles scenarios where users may need to find the shortest route between any two locations without specifying a single source or destination.

Bellman-Ford Algorithm: Bellman-Ford algorithm is employed to handle scenarios involving negative edge weights or cycles within the campus map graph. It is particularly useful for detecting and resolving negative-weight cycles that may arise in certain pathfinding scenarios.

CONCLUSION

Enhanced Campus Navigation Solution:-

Through the implementation of advanced algorithms such as A* (A-star), Dijkstra, Floyd-Warshall, and Bellman-Ford, our campus navigation solution at IIT Jodhpur has achieved significant milestones in enhancing the navigation experience for students, faculty, and visitors.

By harnessing the power of these algorithms, we have successfully addressed the challenges associated with finding the shortest distances and optimal routes across the sprawling campus. Users can now access a robust system that not only calculates the shortest paths between specific locations but also provides comprehensive node-wise routes, ensuring efficient traversal from one point to another.

Moreover, our solution considers various factors such as distance, estimated cost, heuristic information, and real-time constraints like pedestrian traffic and building accessibility. This holistic approach enables users to make informed decisions about their routes, saving time and effort while navigating the campus.

CONTRIBUTION

Contribution of Team Members:-

1) Rahul Garg(B22CH025) :- Contributed in writing the A star algorithm and making the presentation of the project .
2) Yogesh Jajoria(B22ME073):- Contributed in writing the Dijkstra algorithm and making the presentation of the project.
3) Sachin Choudhary(B22ME056):- Contributed in writing the code of Bellman Ford Algorithm in this project .
4) Dhruv Kumar Singh(B22CH010):- Contributed in writing the code of Floyd Warshall algorithm in this project.

THANK YOU