Real Time traffic Navigation System

Ayush Sonika (2023csb1107)
Sumit Sharma (2023csb1165)
Brajesh Khokad (2023csb1111)
Instructor: Dr. Anil Shukla
Teaching Assistant: Mr. Deepu Yadav

October 1, 2024

Abstract

This project focuses on designing the real time traffic navigation system which contains a graph with nodes as the cities and weights of their edges determine the distance / time required between both the places, this system also uses an AVL tree for efficiently managing and updating traffic data across road segments and Dijkstra's algorithm for calculating the shortest path in terms of travel time. unlike other traditional algorithm our system integrates traffic levels as weights to dynamically adjust routes based on real-time congestion data. This system also provide the high traffic situation and also provide a the path that will used in minimum path

1 Introduction

This document describes the structure and functionality of a traffic-aware pathfinding system developed to address navigation in highly congested areas. Our approach integrates an AVL tree to store and retrieve traffic congestion data efficiently, allowing for rapid updates and access.

The primary components include a graph-based representation of cities and roads, where edges represent road segments with associated weights (distances / time). Traffic congestion data dynamically updates the effective weight of edges, allowing Dijkstra's algorithm to calculate paths optimized for current traffic conditions.

2 Equations

Traffic conditions affect travel time by increasing edge weights, which represent road segments in the graph. The total travel time T_{total} for a path from source S to destination D is formulated as:

$$T_{total} = \sum_{i=1}^{n} (d_i + c_i)$$

where:

- d_i: distance of segment i,
- c_i: congestion cost for segment i.

Dijkstra's algorithm is used to minimize T_{total} by choosing the path with the lowest weight, adjusted for congestion.

3 System Components

3.1 Graph Representation

The road network is represented by a graph $G(V, E)$, where V is the set of vertices (cities) and E is the set of edges (road segments). Each edge e_{uv} between vertices u and v has a weight w_{uv}, representing travel distance / total time . Traffic congestion data is stored separately between one place to another and dynamically updates the effective weight.

3.2 AVL Tree for Traffic Data

The AVL tree is used to store and retrieve traffic congestion data efficiently, with each node representing a unique road segment. The AVL tree nodes contain:

- **key**: unique identifier for each road segment (edge),
- **traffic**: congestion level, classified as low, moderate, high, or very high.

3.3 Tables

Traffic Level	Congestion Impact
Low	+2 km
Moderate	+4 km
High	+6 km
Very High	+8 km

Table 1: Traffic congestion levels and their impact.

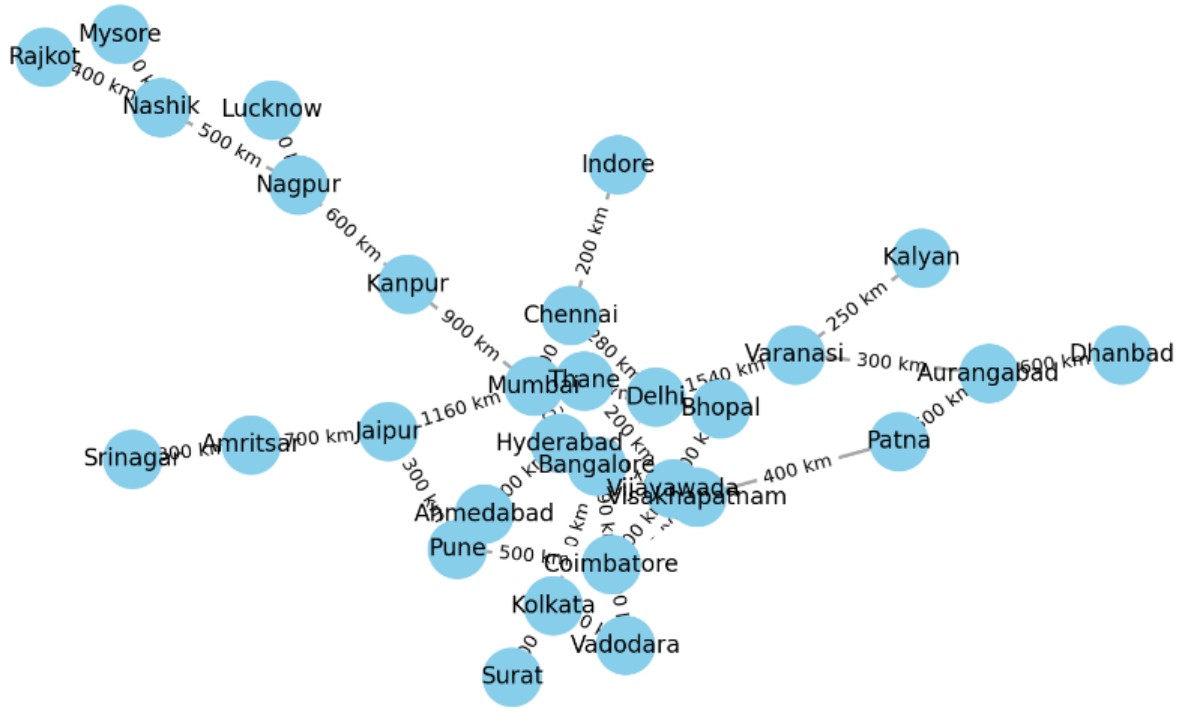

Figure 1: Graph representation with cities as nodes and road segments as edges.

3.4 Algorithms

Algorithm 1 Traffic-Aware Dijkstra's Algorithm

Initialize distances from the source to all cities as infinity.
Set the source city's distance to 0 and add it to the priority queue.
while the queue is not empty **do**
    Extract the city u with the smallest distance.
    for each adjacent city v of u **do**
        Retrieve traffic congestion c for the edge (u, v) from the AVL tree.
        if $\text{dist}[u] + w_{uv} + c < \text{dist}[v]$ **then**
            Update $\text{dist}[v]$.
            Set $\text{parent}[v] = u$ to reconstruct the path.
        end if
    end for
end while
Construct the path from source to destination using parent pointers.

4 Results

Testing was conducted on a garph of 30 cities with dynamically changing traffic congestion data. The AVL tree used to do efficient updates, while Dijkstra's algorithm provided optimal routes based on the latest traffic conditions. The system detected and avoided highly congested routes, demonstrating potential for real-world applications.

5 Conclusion

This project successfully integrates AVL trees with graph-based pathfinding to create a system that dynamically adjusts for traffic congestion. Future enhancements may include incorporating machine learning to predict congestion patterns and refine route efficiency.

References

[1] E.W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik*, 1959.

[2] J. Adelson-Velsky and E.M. Landis, "An Algorithm for the Organization of Information," *Soviet Math. Dokl.*, 1962.

[3] Papageorgiou, A., and W. K. K. Van Zuylen. "Dynamic Traffic Management Systems."