Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Project Report

on

"Traffic simulation using Queue"

[Code No: COMP 202]

(For partial fulfillment of 2nd Year/ 1st Semester in DSA)

Submitted by

Sakar Dhimal(22)

Submitted to

Mr.Rupak Ghimere

Department of Computer Science and Engineering

Submission Date: 2025/12/27

Acknowledgement

I would like to express my sincere gratitude to my course instructor, Mr. Rupak Ghimire, for assigning this work as part of the COMP 202 course. This assignment provided me with a valuable opportunity to strengthen my understanding of the subject through independent effort and practical application. His guidance, course structure, and emphasis on conceptual clarity played an important role in shaping my approach to this work. I am also thankful to Kathmandu University and the Department of Computer Science and Engineering (DoCSE) for providing the academic environment and resources necessary to complete this assignment successfully. I welcome any feedback or suggestions that may help me improve my work and understanding further.

Regards,
Sakar Dhimal

Title:Traffic Light Simulator using Queue

Course: Data Structure and Algorithms (COMP202)
Student Name: Sakar Dhimal
Roll no: 22
Assignment: Implementing Queue for Solving the Traffic Light Problem
Submission: GitHub Repository – Traffic Light Simulator

1. Summary of Work

This project implements a traffic junction simulator to demonstrate the practical application of linear data structures (Queue) in solving a real-world traffic management problem. The simulator models a four-road intersection where vehicles arrive, wait in queues, and are dispatched based on traffic light conditions.
The system supports:

- Normal traffic conditions, where all lanes are served fairly.
- High-priority conditions, where a designated priority lane (AL2) is served immediately when congestion exceeds a threshold.

The project also includes a visual simulation using SDL, allowing real-time observation of vehicle movement, lane congestion, and traffic light behavior.

2. Problem Statement

At a traffic junction connecting four major roads (A, B, C, and D), vehicles must be managed efficiently to avoid congestion and deadlock. Each road consists of three lanes, with one lane designated as a priority lane. The system must:

- Maintain fairness during normal conditions.
- Detect congestion in the priority lane.
- Dynamically adjust traffic light priority using queue-based logic.

3. Objectives

- Apply queue data structures to a real-world traffic management scenario.
- Simulate and visualize queue behavior using graphics.
- Implement priority handling using queue size thresholds.
- Avoid deadlock by ensuring only one road is served at a time.

4. System Design

4.1 Roads and Lanes

- Total Roads: 4 (A, B, C, D)
- Lanes per Road: 3
    - Lane 1: Incoming lane
    - Lane 2: Controlled by traffic light
    - Lane 3: Free left-turn lane
- Priority Lane: AL2

Each incoming lane maintains a queue of vehicles waiting to cross the junction.

5. Data Structures Used

Data Structure	Implementation	Purpose
Queue	Array-based / Struct-based	Store vehicles waiting in each lane
Priority Queue (Logical)	Condition-based queue selection	Give priority to AL2 during congestion
Struct	Vehicle, Lane, Traffic Light	Represent simulation entities

6. **Core Data Structures**

6.1 Vehicle Structure

```
typedef struct {
    int id;
    int road;
    int lane;
    float position;
} Vehicle;
```

This structure represents each vehicle uniquely and stores its lane and road information.

6.2 Lane Queue Structure

```
typedef struct {
    Vehicle vehicles[MAX_VEHICLES];
    int front;
    int rear;
    int count;
} Queue;
```

Each lane is represented as a queue where vehicles are enqueued on arrival and dequeued when the light turns green.

7. Project Structure

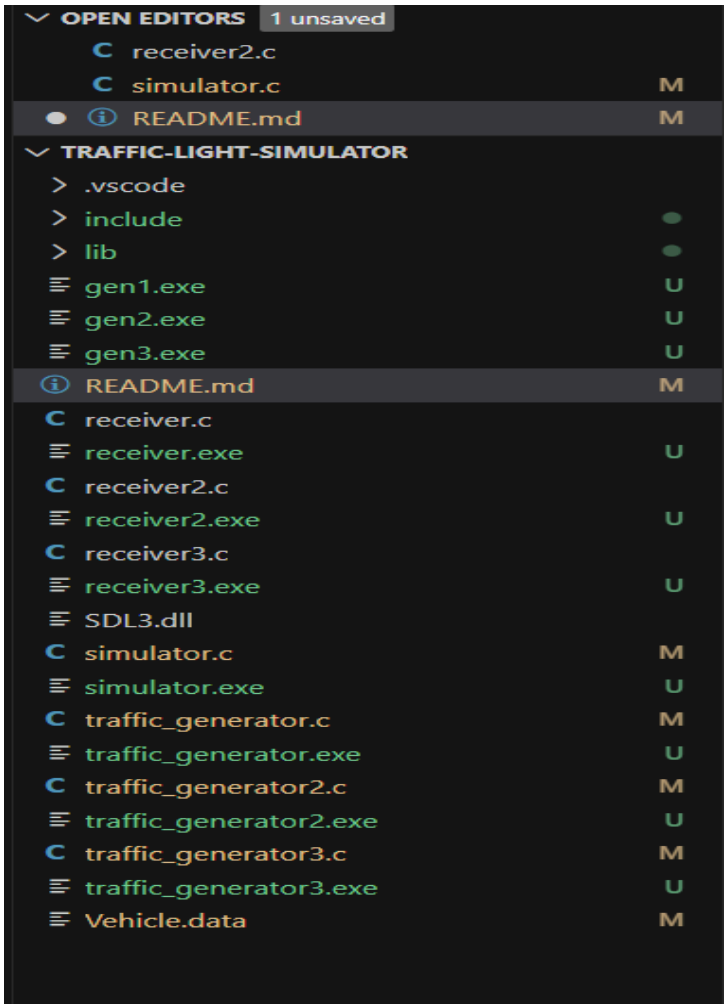

fig:Project Structure

8. Algorithms Used

8.1 Normal Traffic Algorithm

1. Calculate the average number of vehicles across all normal lanes.
2. Estimate green-light duration using:

   $V = (|L1| + |L2| + |L3|) / n$

   $\text{Green Time} = V \times t$

8.2 Priority Lane Algorithm

1. Continuously monitor AL2 queue size.
2. If queue size > 10:
   - Immediately grant green light to AL2.
3. Continue serving AL2 until queue size < 5.
4. Resume normal scheduling afterward.

This ensures reduced waiting time for high-priority traffic.

9. Traffic Light Logic

1. State 1 (Red): Vehicle stop
2. State 2 (Green): Vehicle allowed to move

Only one road can have a green signal at any given time to avoid deadlock.

10. Visualization

The simulation uses SDL to render:

1. Roads and lanes
2. Vehicles as moving rectangles
3. Traffic lights with red/green states

The visualization helps in understanding queue behavior and system performance.

11. Time Complexity Analysis

Operation	Complexity
Enqueue	O(1)
Dequeue	O(1)
Priority Check	O(1)
Lane Selection	O(n), n = number of lanes

Overall system performance is efficient and suitable for real-time simulation.

12. Limitations

1. Priority queue is condition-based rather than heap-based.
2. Vehicle generation is simplified.
3. No real-world traffic randomness modeling.

13. Future Enhancements

1. Implement true heap-based priority queue.
2. Add adaptive traffic light timing.
3. Introduce emergency vehicle handling.
4. Improve realism using probabilistic traffic generation.

14. Conclusion

This project successfully demonstrates how queues can be used to model and solve traffic management problems. By combining data structures with graphical simulation, the system provides both conceptual clarity and practical insight into queue-based scheduling.

15. Execution Video

testing_code.mp4

16. Source Code

GitHub Repository:

https://github.com/007Sakar/Traffic-Light-Simulator