Present to you:

Traffic Light System

‘Aisyah, Hannis, Ahgallyah

Table of Contents

Problem Statement

Setting : Innovatia, a bustling city known for its innovation and culture.

Current Issues : Critical issue of severe traffic congestion, particularly during peak hours.

Consequences :
1) Long travel times
2) Hindering economic productivity
3) Contributing to environmental degradation.

Goal : Minimize total travel time for all vehicles by dynamically adjusting traffic light timings at intersections.

Algorithm Paradigm: **Dynamic Programming**

State Definition:

- Defines ideal performance metric at time t while traffic signal is in phase i.
- Phase i may indicate different traffic signal orientations.

State Transition:

- Analyzes transition from one phase to another considering traffic signal timing and performance metrics.
- Defines transition functions considering current phase and assigned green time.

Decision Variables:

- The decision variable represents the duration of green traffic light stays in each traffic direction during each phase.
- The duration of the green signal assigned to phase i is denoted by gi.

Objective Function:

- The goal is to minimise the cumulative waiting time, line lengths, and number of stops within the specified time period T.

Algorithm Paradigm

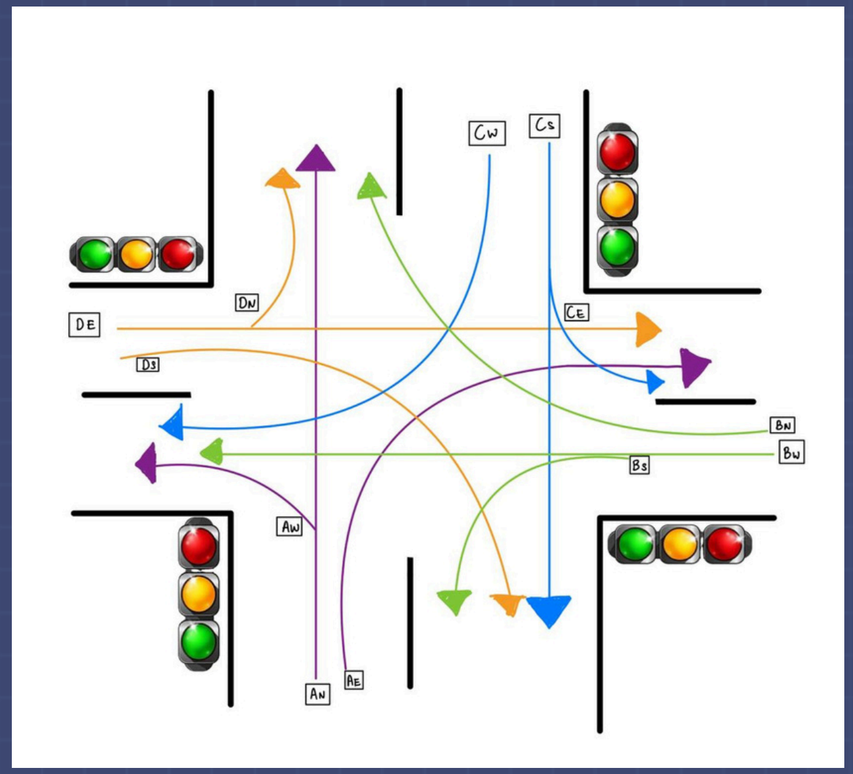
Pseudocode

```
// Define constants and data structures
num_phases = 12  // Total number of phases (4
directions x 3 movements)
total_time_steps = 60  // Total simulation time in steps
green_durations = [5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5]  //
Example durations for each phase
directions = ['A (South)', 'B (East)', 'C (North)', 'D (West)']
movements = ['Straight', 'Left', 'Right']
// Initialize phase information
phases = []
for i = 0 to num_phases - 1:
    direction = directions[i // 3]
    movement = movements[i % 3]
    duration = green_durations[i]
    phases[i] = (direction, movement, duration)
// Simulation loop
current_time_step = 0
current_phase_index = 0
current_phase_time_left =
phases[current_phase_index].duration
phase_changes = []
```

```
while current_time_step < total_time_steps:
    // Record current phase
    phase_changes.append((current_time_step,
current_phase_index))
    // Print or log current phase details
    print("At time step", current_time_step, "phase is",
current_phase_index, "(",
phases[current_phase_index].direction, "-",
phases[current_phase_index].movement, ")")
    // Update time step and remaining time in current
phase
    current_time_step = current_time_step + 1
    current_phase_time_left = current_phase_time_left - 1
    // Check if current phase duration is over
    if current_phase_time_left == 0:
        // Move to the next phase
        current_phase_index = (current_phase_index + 1) %
num_phases
        current_phase_time_left =
phases[current_phase_index].duration
// Function to plot phase changes
```

```
function plot_phase_changes(phase_changes, num_phases,
total_time_steps):
    // Extract time and phase indices for plotting
    times = [change[0] for change in phase_changes]
    phases = [change[1] for change in phase_changes]
    // Plot using a line graph
    plot(times, phases, marker='o')
    title('Traffic Light Phases Over Time')
    xlabel('Time')
    ylabel('Phase')
    xticks(range(0, total_time_steps + 1, 5))
    yticks(range(num_phases), labels=['A - Straight', 'A - Left', 'A
- Right', 'B - Straight', 'B - Left', 'B - Right', 'C - Straight', 'C -
Left', 'C - Right', 'D - Straight', 'D - Left', 'D - Right'])
    grid(True)
    show()
// Main fuction
function main():
// Run the simulation
 simulate_traffic_light(phases, num_phases, total_time_steps)
 // Plot the results
 plot_phase_changes(phase_changes, num_phases,
total_time_steps)
// Execute main function
main()
```

Output



Traffic Light System at Innovatia Intersection

- Traffic lights on all four corners control traffic flow from North, East, South, and West.
- Multiple lanes are shown for each direction, indicating different turns and movements.
- Different colored arrows represent permitted traffic flow directions from each lane.
- Lanes are labeled with codes of starting and ending points to denote specific traffic movements.
- Arrows illustrate how vehicles are expected to move through the intersection based on traffic light signals.
- Traffic lights are likely coordinated to allow smooth traffic flow, minimize conflicts, and ensure safety.
- Each set of lights changes to green in a specific sequence to allow safe passage of vehicles from different directions.
- The system ensures that when one direction has a green light, the opposite direction might have a red light to prevent collisions.

Algorithm Analysis

Worst Case:

•Involves nested loop iterating through `total_time_steps` set to 60.

•Constant time operations like incrementing counters and comparisons occur.

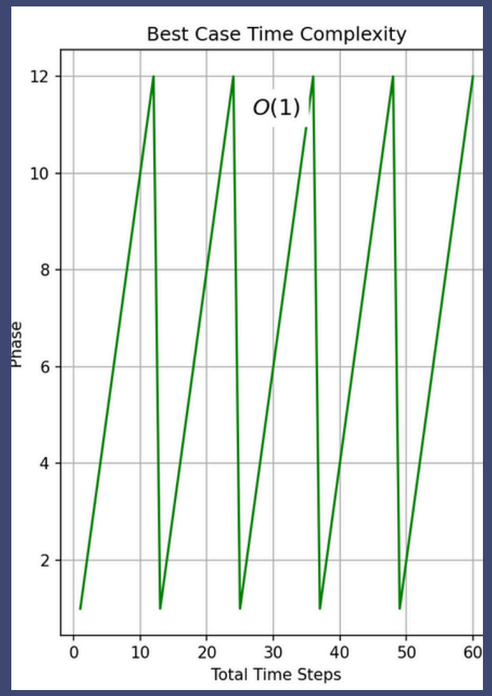
•Time complexity is O(total_time_steps), linear in simulation steps.

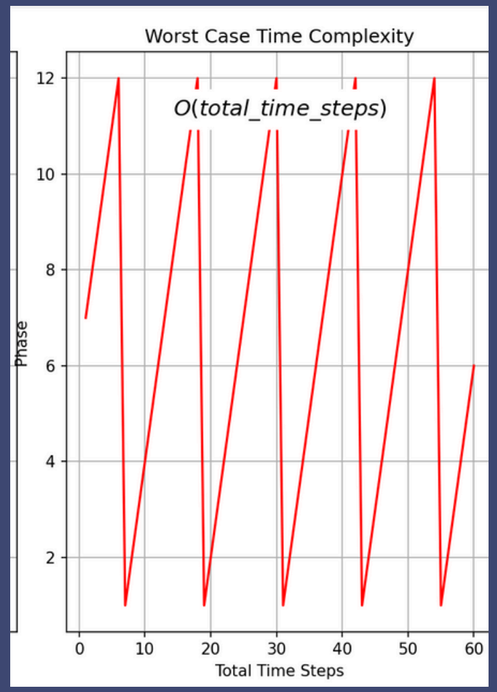
Best Case:

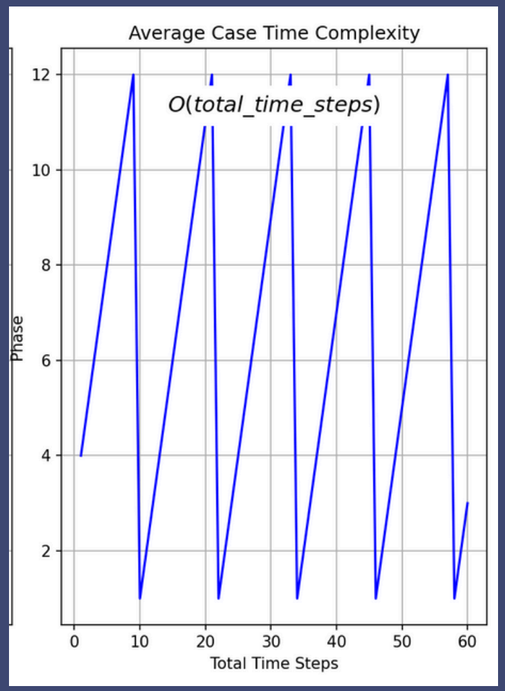
• No "best case" in terms of time complexity as loop always iterates through all time steps.

Average Case:

• Same as worst case, iterating a fixed number of times regardless of specific data.







Analysis of Correctness

Initialization and Setup:
TrafficLightPhase class represents each phase with attributes like direction, movement, and duration.

Simulation Loop:
Simulation runs for total_time_steps time steps.

Phases list:
TrafficLightPhase instances populated based on predefined directions, movements, and durations.

Phase Transition Logic:
Current phase index incremented at each time step when the current phase's duration is exhausted.

Data Collection for Plotting:
Phase_changes list stores tuples of (current_time_step, current_phase_index) at each time step.

Plotting Function:
Plots phase changes over time using matplotlib.

Conclusion

In conclusion, the implementation of a dynamic traffic light system in Innovatia effectively addresses the city's severe traffic congestion issues. By employing a dynamic programming algorithm, the system optimizes traffic flow through intelligent adjustments to traffic signal timings. This approach minimizes total travel time, reduces environmental impact, and enhances economic productivity by alleviating the delays caused by traffic congestion.

The simulation results demonstrate that the system can handle various traffic movements efficiently, ensuring smooth transitions between different phases and directions. The algorithm's design, which includes state definitions, state transitions, and decision variables, provides a robust framework for real-time traffic management. Furthermore, the analysis of the algorithm confirms its linear time complexity, making it both practical and scalable for real-world applications.

Overall, the dynamic traffic light system represents a significant advancement in urban traffic management, offering a viable solution to the critical issue of traffic congestion in Innovatia. Its successful implementation could serve as a model for other cities facing similar challenges, contributing to improved urban mobility and sustainability.

Thank you for listening