Team Member Details
Raj Tripathi 2019B4A70869H
Satvik Omar 2019B4A70933H
Pranjal Jasani 2019B4A70831H

Traffic Management System

OVERVIEW

All major cities today face the issue of constant traffic jams. This does not just increase the travel time but also frustrates the passengers. Traffic lights have at least two states, usually red to signal "stop", and usually green to signal that cars can proceed through. With the ever increasing traffic in the world, it becomes more important to try and find ways to reduce this waiting time.

GOAL

To optimize the traffic light schedule of a city and minimize the time spent in traffic thus allowing vehicles to reach their respective destinations faster.

SPECIFICATIONS

To solve this problem, we make certain assumptions. These assumptions allow us to effectively model the city plan, traffic lights and vehicles while keeping the scenario fairly close to the practical one.

CITY PLAN

The city plan is a collection of streets with intersections among them.

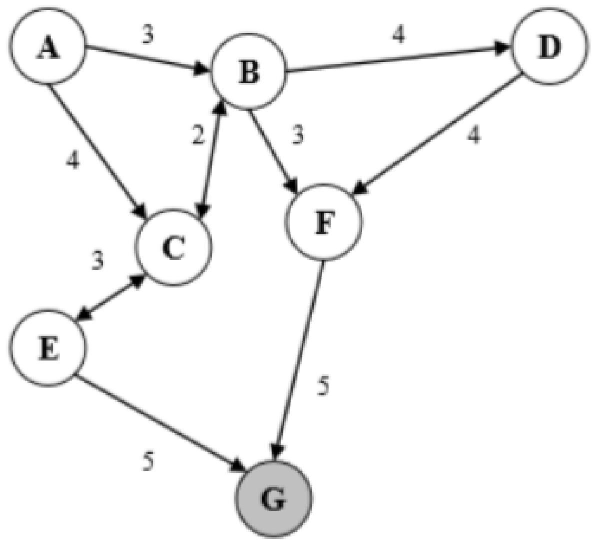
The streets are assumed to be unidirectional (Since two-way streets can also be easily modeled in this way).

A street connects two intersections and does not contain any intersection in between. (In a practical scenario, imagine using an overpass to cross streets without intersection).

Let **S** be the expected time of crossing a given street. For simplicity, we assume that this **S** is fixed for all the vehicles.

To avoid redundancies, we shall never have two streets connected intersections in the same direction.

We can label the intersections with some index(e.g. A,B,C...) and label the streets with some name(e.g. S1,S2). Then a city plan is described by a list of the 4-tuples **(street_index, start_point, end_point, S)**, where start_point and end_point are the indices of the respective intersections.



TRAFFIC LIGHTS

We assume that there are lights at the end of every street. At any fixed time, there can only be at most one green light at a given intersection.

- **QUEUES**

  When a given traffic light is red, then the vehicles start to form a queue at the end of that street. Assuming that **I** vehicles can cross an intersection every second(same for all intersections), only the first **T*I** vehicles can cross the intersection if the green light turns on for **T** seconds.

- **SCHEDULE**

  Our goal is to set a traffic light schedule for each intersection. This schedule tells us when each light is supposed to go green and for how long. We can consider this to be a list of an ordered pair (**street_index, duration)**, where Duration(in secs) tells us for how long the light at the end of Street street_index goes green. If the light goes green at time t, then it automatically becomes red at t+duration. For simplicity we assume that this schedule is periodic and not varying, and for that we assume that a street_index only appears once in the list.

VEHICLES

We can model a vehicle using just a sequence of paths. Although in real life a vehicle can even repeat a street, this will not be particularly useful. We acknowledge this possibility and may try to implement this if time permits, but for now we assume that no intersections are repeated(as is usually the case). At **t=0**, we assume all vehicles are **at the end** of the first street in their path. These vehicles follow the traffic rules and reach their destination. When they reach the end of the last street in their path, then they are removed from the simulation.

This is the variable that allows us to test our scheduling. Our system will take these paths as one of the inputs(If two vehicles start at the same street, the order of input is the order in the queue).

OPTIMIZING

We shall try to form a metric to measure the effectiveness of the scheduling. This metric will be based on the number of vehicles that reach their destination in a given time period. The details are yet to be determined.