CSE 6730: Modeling and Simulation: Fundamentals & Implementation
Spring 2019

Project 2: Checkpoint
PEACHTREE STREET CASE STUDY: A SIMULATION APPROACH

Somdut Roy, Yashovardhan Jallan and Han Gyol Kim

Background and Motivation of Study

The recent success of transportation agencies in managing the traffic during the Super Bowl 2019 in Atlanta screamed the necessity of transportation studies. Transportation can be studied majorly in two fronts: transportation planning and traffic engineering. While transportation planning primarily deals with macroscopic aspects (like travel-demand patterns of different Traffic Analysis Zones, etc.), traffic engineering has the capability of tackling more microscopic issues in a roadway network. Primarily traffic engineering concerns with operations and safety. The uniqueness of traffic studies is that, once you make a decision about a project and build infrastructure based on calculations, there are certain unpredictable real-life factors that come into play. So there are aspects that can not be determined in a deterministic way. The probabilistic aspects can however be evaluated with the aid of simulation. With the advent of connected and autonomous vehicle technologies, traffic studies are gaining more and more voices and interest in the world of technology and simulation is a very powerful tool in propelling us into the next generation of transportation.

Our current study involves the Peachtree St NE corridor from 10th St through to 14th St. This is a part of the heart of Atlanta Midtown area. Factors like existence of more points of attraction, temporal travel pattern variations, etc makes studying this corridor even more intriguing. Making traffic simulator helps us have control on all driving factors in the simulation. We can play with different factors and look at the traffic response to it.

Simulation Approaches Implemented

As per the problem statement of this project, we are to simulate traffic in one direction for this corridor. We had to attack the problem in two fronts: (1) Queueing Model, (2) Corridor

Model. We concern ourselves with traffic in the northbound direction starting from 10th street. We have used Python 3 for implementing all simulations.

(1) Queueing Model

In the queueing model, the intersection signal acts as the server as it "serves" the purpose of allowing vehicles into the next corridor segment through itself and each vehicle acts as the "client". Once a vehicle crosses the intersection, it becomes a client for the next intersection. Vehicle, that takes a turn from an intersection, changes travel direction and hence gets excluded from our scope of study. In this model, we limit ourselves to study the queueing pattern at the intersections and do not look into speed-acceleration variations of a particular vehicle while traversing through the corridor. The queues are modelled by two different Worldviews, viz. (a) Event-Oriented, (b) Activity-Scanning. Details about the world views and underlying concepts for building the models are discussed elaborately later.

(2) Corridor Model

To model the corridor, we use the Cellular Automata (CA) model. For this each vehicle is alloted a cell which act as slot for each vehicle at a given time-stamp. This model vividly captures inter-vehicular interactions, variation in vehicle speed and acceleration due to preceding vehicle in a corridor. Numerous studies have been done with success to model traffic as CA in the past few decades and this projects provided us with an opportunity to implement the established knowledge in this regard to have the "cars travelling in a road" scenario come to life.

Traffic Data and Input Analysis

We aim to base our model on NGSIM data provided to us. We have trajectory data of the corridor in both directions (updated every one tenth of a second for each vehicle) with detailed microscopic information like speed, acceleration, headway, spacing, etc. Alongside trajectory data, we also have the signal phase cycle information for all the intersections. The aim is to use this data as the basis for simulation inputs and then verify and validate the gathered simulation output with this data to measure the performance (like sensitivity, stability, believability, etc) of each simulation model. In the following section, we discuss how we incorporate the provided data into different simulation models, assumptions made in each model and detailed underlying concept for each.

Queueing Model

To maintain uniformity and to have the possibility of comparative analysis of different world views in future, we ensure that we use the same methodology for input analysis and assumptions for unknown parameters for queue modelling with both world views.

Input Analysis

In the queueing model, the inter-arrival time is modelled from the inter-arrival times of all the northbound vehicles in the corridor. An empirical distribution of inter-arrival time for northbound traffic is created as shown in figure 1 below. Using the cumulative distribution trendline (red line) of the figure, we draw random samples of inter-arrival times.

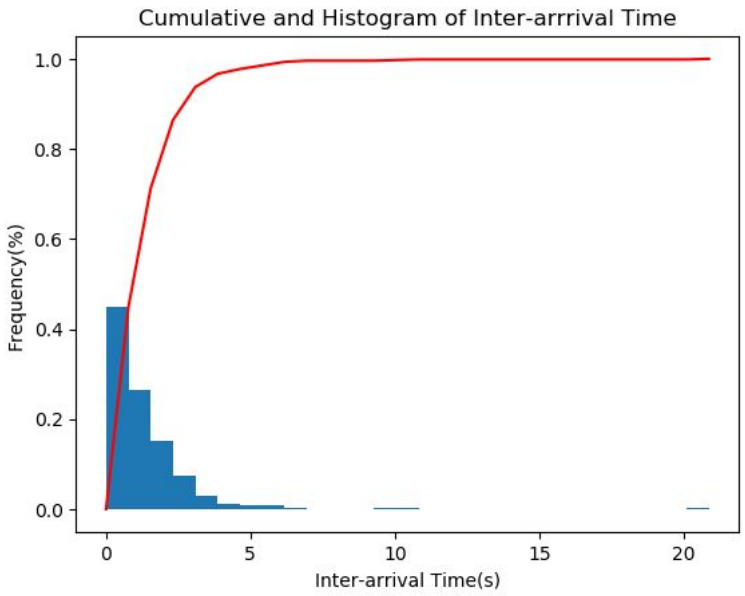

Figure 1: Distribution of inter-arrival time data

As we cumulatively add up the inter-arrival time samples, we build up a Future Event List (FEL) for arrival of vehicles into the corridor.

Task details and Model assumptions

For the time being, we have three intersections, 10th, 11th and 12th St (in the order mentioned). We have considered only a single lane running through, without turn movements for vehicles. We allow 0.1-0.3 secs lag time (uniform random sample) for each vehicle to process signal and pass. This works as the variable "service time" for the "server" intersections. Once a vehicle passes one intersection, we add a 10s to 15s (uniformly sampled travel time) to the vehicle before it joins the queue in the next intersection. The time is chosen, based on the fact the corridors are of approximate length of 0.1 miles and a vehicle moving at 20 odd mph takes 10 odd seconds to travel that length before being in the discussion for next queue. This operation also syncs the vehicles to the correct signal phase for the next intersection making the scenario more realistic.

Different World Views: Underlying Concept/ Pseudocode

(a) Event-Oriented

The underlying concept can be explained as follows:

Variables:

- Now: current simulation time
- InTheCorridor: number of vehicles entering or waiting to enter corridor
- InTheIntersection: number of vehicles in the intersection
- SignalFree: Boolean, true if signal is not occupied by a vehicle
- Signal: RED (No-Go), GREEN (Go), YELLOW (Cautiously Go/ Cautiously No-Go)

Arrival Event:

```
InTheCorridor := InTheCorridor+1;
If (Signal!=RED and SignalFree):
        SignalFree:=FALSE
        Schedule IntersectionEntry event @Now+TravelTime;
```

Intersection Cross Event:

```
:InTheCorridor:=InTheCorridor-1;
InTheIntersection:=InTheIntersection+1;
Schedule InterSectionCrossing event @ Now+ServiceTime;
If (InTheCorridor>0) Schedule IntersectionEntry event @Now+TravelTime;
Else SignalFree := TRUE;
```

InterSectionCrossing Event:

```
InTheIntersection := InTheIntersection - 1;
```

As instructed, we maintained a Future Event List (priority queue) to handle events.

(b) Activity-Scanning

We can do simple time-stepped model for this case. However as suggested in lectures, it is advisable to adopt a more time-efficient activity-scanning method.

- Maintain a Future Event List (priority queue) for activities
- Simulation consists of a set of activities, events
- B-events: Time-stamped events of Vehicle activities: Stored in Future Event List (FEL): Initially contains arrival times of all vehicles
- C-events: About Signal Activity

While (simulation not complete)

- Remove smallest timestamp B-event from FEL and all others with the same timestamp
- Advance simulation clock to timestamp of removed events
- Process all B-events removed from FEL; create and schedule triggered B-events from them (waiting, Getting green, crossing intersection)
- For each C-event: if (pre-condition is true) process event and create and schedule B-events triggered from them
- Repeat above two steps until no events remain that can be processed

Initial Observation and Future goals

With current architecture, run-time was not an issue with both world views. An interesting study will be the difference in performance with more intersections and complicated lane structure, like multiple lane and actuated signals, we may be able to study a performance difference of the two world views. We will be adding lanes for turn movements and assigning turns to vehicles based on the ratios given in the dataset.

Corridor Model

We are given signal information for through and left for 10th street and 14th street and only through movement for the rest. So we build CA model initially with a through and a left lane before 10th St and a through lane before 11th St. We have not used the volume information from NGSIM yet but we have implemented the signal phase information based on the provided data.

Task details and model assumptions

Assuming a car of 6-6.5 feet, we create a cell of length 8 feet (including ideal spacing between cars). Both corridors (i.e. before 10th St and before 11th St) are approximately 0.1 mile long. So we have 66 cells of 8 feet to build a 0.1 mile corridor. Figure 2 below shows the schematics of current model. We provide 2 cells for the intersection. Therefore, to build the two corridors, we had (66+2 for 10th St+66+2 for 11th St)=136 cells to model the entire though corridor for current network. We have two lanes, one through, and one Left-Turn only before 10th St and one through-only lane before 11th St. The maximum speed based on NGSIM data is 56 ft/s. Based on that we can see the maximum speed is approximately 7 cells per second. With a 0.05 probability, a vehicle is assigned to turn left after Corridor 1 and hence will start and stay on the left-turn lane. Once the left-turned vehicles leave corridor 1, we drop those vehicles from our study. We generate vehicle at 0.3 veh/second, by assigning 0.3 probability of assigning a new vehicles into the corridor at any given time-stamp. We assign 2-6 cells/s speed (chosen uniformly) to each newly "created" vehicles. New influx of vehicles into corridor-2 is caused stochastically at 0.25/sec during the (green+yellow) phase for EB left-turn.

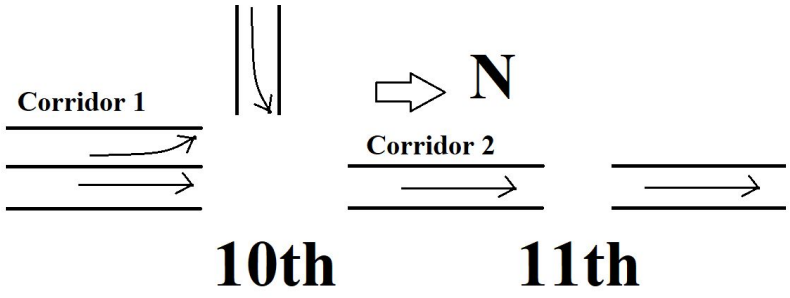

Figure 2: Schematics of current CA model

Rules for Speed assignment and Position Change

There are four steps movement in the simplest rule set, which leads to a realistic behavior, has been introduced in 1992 by Nagel and Schreckenberg. We introduce the signal logic into that to create the following five-step process.

Step 1. All the vehicles whose velocity has not reached the maximum Vmax will accelerate by one unit. $v \rightarrow v+1$

Step 2. Assume a car has m empty cells in front of it. If the velocity of the car (v) is bigger than m, then the velocity becomes m. If the velocity of the car (v) is smaller than m, then the velocity stays v. ($v \rightarrow \min[v, m]$)

Step 3. The velocity of the car may reduce by one unit with the probability p. For current model, p=0.05 during GREEN time, p=0.2 during YELLOW time (more cautious).

Step 4. The velocity needs to be reduced so as to not cross the intersection at a RED.
$V \rightarrow \min[v, signalhead\_cell - current\_cell]$

Step 5. After 4 steps, the new position of the vehicle can be determined by the current velocity and current position. ($xn' \rightarrow xn + vn$)

Initial Observation and Future goals

With current model, run time was not an issue. The model prepared for 600 simulation seconds ran almost instantly. In near future, we will add more lanes and extend the corridor to all four intersections. With added lanes, we will also have to add the lane-switching logic for each vehicle. In addition, we will also incorporate volume-input based on NGSIM data (unlike the current model). To get better results, we will incorporate the vehicle type (with varying size). To incorporate vehicles of varied size, we are going to use smaller cells and associate multiple to a vehicle at a time. Having smaller cells will also make a smoother and more precise simulation.

Big Picture (as of now)

If things work out in time, we could also explore a hybrid model that incorporate CA model into a queueing model to create a more realistic behaviour. After verification and validation, we can subject the resultant trajectory data to estimate aggregated energy and emission figures of the corridor and make observations from them. Other factors like simulation run time comparison can also be studied. This section is subject to change based whatever new and interesting findings we stumble upon while completing the missing pieces of the simulation.

Script running Instruction

The scripts and truncated data (used for processing) can be found at:
https://github.gatech.edu/sroy86/CSE6730_Spring-2019_Project-2

Get the entire folder.

Use Python 3.6

CA model:
python3 CA_model.py
(Trajectory.csv file is created as output for 600 simulation seconds)

Queueing Model:
python3 Event_QSimDriver.py ``for Event-Oriented simulation
python3 Activity_QSimDriver.py ``for Activity-Scanning simulation

While running the two scripts above, you will be prompted to ask for printing output (in the form of vehicle information) at the end of each intersection:
If you want output printed: type True
To opt out: type False