Modeling and simulation

Project One Report

CSE 6730

Group 15

Zongwan Cao

Rahul Nihalani

Yaohong Xi

Description of the problem

Our object for this project is to develop a simulation model to compare the average travel time for vehicles to traverse a portion of Peachtree Street (from 10th to 14th) when using synchronized compared to unsynchronized traffic signals. Traffic signal synchronization is a method of timing groups of traffic signals along an arterial to provide for the smooth movement of traffic with minimal stops. By calculating the arrival time for a group of vehicles at specific speed, we can time the traffic signals to turn green just as those vehicles arrive at each intersection.

A more realistic application for this project is we can optimize the travel process and come out with the most effective traffic light pattern to reduce the traffic congestion.

The NGSIM Dataset provides Peachtree Street Trajectory and Signal Time data. We can use it to get the geometric information about the Peachtree. Figure 1 shows the geometric structure of the study area. In this area, there are five intersections that divide the street into four sections.

Figure 1 Study Area Schematic
Source: NGSIM Peachtree Street Dataset

Conceptual model

This traffic modeling system can be abstracted as queueing network conceptual model. A lane loading cars are regarded as a queue. Each car is an element of a queue. The time cars need to cross an intersection depends on the traffic signal status and cars' direction. When a car crosses an intersection, it will enter a new queue and leave the old one.

1. Input and Output
The first thing we need to figure out is the input and output. Based on our objects, we need input attributes for each car including carID, starting point, terminal point,
direction and starting time. After running the model, ending time will be obtained from the model. Ending time minus starting time is each car's traveling time. All the input data are produced randomly.

2. Simplifications
Some simplifications are made during the establishment of conceptual model.

It is both unnecessary and difficult to take some details into consideration in our simulation. Therefore we make the following four simplifications.

(1) Each lane has unlimited capacity to hold vehicles. It is unlikely that the number of cars on Peachtree Street is over its capacity. Thus we neglect the capacity boundary.
(2) There are two lanes on each side. We do not want to make things to complicated, so we use two lanes each side to represent the reality.
(3) The traffic light has three states, namely red (all cars need to stop), green (cars can only go straight and turn right) and left (cars can only turn left). We picked three key states which can reflect cars main behaviors.

Due to the purpose of our simulation, we focus on the time that cars consumed on the signals of intersections. Therefore we made the following simplifications. We ignore the time for cars starting, stopping, waiting in the stop signs and driving on the lane. Also we regard the time for crossing each intersection as a constant.

(4)The car can start and stop immediately
(5) There is no stop sign in each direction.
(6) For all cars, the time for crossing each intersection is fixed. By referring to the picture below, we take the speed of each car as constant 30ft/s. Also from the NGSIM data, we take average length of each intersection as 90 feet, which means the time for crossing each intersection is 3 sec.

Figure 2 Illustration of the velocity

Source: Trajectpory Data Description Page 20

(7) Time for cars driving on the lanes is ignored.
(8) The total car number in our system is a constant.

3.Assumptions

Some data about our simulation is hard to collect. So we need to make some assumptions to process our model.
(1) The distribution of vehicle on the minor roads is derived from the data of major road (the Peachtree Street). We lack this part of data, so we assume based on what we already have.
(2) There is also no record for drivers' behaviors. We treat them as unaggressive drivers, and then we induce no one will pass the cars before.
(3) Entering time for each car, entering intersection for each car and exiting intersection for each car all obey uniform distribution.

4.Entities and Events

Then we move to the content of our conceptual model.
For entities, we consider cars, lanes and junctions.
After analyzing this modelling system, we decide four events
(1) Enterlane, occurring when a car cross an intersection and enter a new lane

(2) Exitlane, standing for a car cross an intersection and leave the old lane
(3) Checksignal, in order to get the status of individual traffic light
(4) Changesignal ,to change the state of individual signal

Literature survey

Traffic simulation is modeling of transportation system that can help in plan design and operate transportation. The traffic simulation models can be classified according to discrete and continuous time, state and space [1]. The queuing model has long been used to study discrete event simulation [2~5]. Priority queue is a queue data structure for each element has an associated priority. In a priority queue, an element with high priority can be popped before an element with low priority. Queuing model had good performance in simple traffic micro-simulation [6]. It is possible to use such a dynamic, event-based approach to simulate transportation with affordable data and hardware. For example, Vandaele and his team developed some queuing models based on traffic counts and modeled the behavior of traffic flows as a function of some of relevant factors [7]. Furthermore, for the computation of the large-scale it can also be speed up by parallel implementation [8]. In general the queue model is simpler and easier to improve than the flow models of DYNASMART(Dynamic Traffic Assignment and Simulation for Advanced Network Informatics)[9], DynaMIT[10], and the cell transmission model[11]. The disadvantage of queuing model is that the speed in traffic jam condition cannot be realistically modeled; the advantage is higher speed in computation.

[1] Pursula, Matti. "Simulation of traffic systems-an overview." Journal of Geographic Information and Decision Analysis 3.1 (1999): 1-8.
[2] Henriksen, James O. "An improved events list algorithm." Proceedings of the 9th conference on Winter simulation-Volume 2. Winter Simulation Conference, (1977).
[3] Kingston, Jeffrey H. "Analysis of tree algorithms for the simulation event list."Acta Informatica 22.1 (1985): 15-33.
[4] McCormack, William M., and Robert G. Sargent. "Analysis of future event set algorithms for discrete event simulation." Communications of the ACM 24.12 (1981): 801-812.
[5] Chi, Sung-Do, Ja-Ok Lee, and Young-Kwang Kim. "Discrete event modeling and simulation for traffic flow analysis." Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century., IEEE International Conference on. Vol. 1. IEEE, 1995.

[6] Simon, Patrice M., Jörg Esser, and Kai Nagel. "Simple queueing model applied to the city of Portland." International Journal of Modern Physics C 10.05 (1999): 941-960.
[7] Vandaele, Nico, Tom Van Woensel, and Aviel Verbruggen. "A queueing based traffic flow model." Transportation Research Part D: Transport and Environment5.2 (2000): 121-135.
[8] Cetin, Nurhan, Adrian Burri, and Kai Nagel. "A large-scale agent-based traffic microsimulation based on queue model." IN PROCEEDINGS OF SWISS TRANSPORT RESEARCH CONFERENCE (STRC), MONTE VERITA, CH. 2003.
[9] Mahmassani, H. S., T. Hu, and R. Jayakrishnan. "Dynamic traffic assignment and simulation for advanced network informatics (DYNASMART)." Urban traffic networks: Dynamic flow modeling and control. Springer, Berlin/New York(1995).
[10] Ben-Akiva, Moshe, et al. "DynaMIT: a simulation-based system for traffic prediction." DACCORS Short Term Forecasting Workshop, The Netherlands. 1998.
[11] Daganzo, Carlos F. "The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory." Transportation Research Part B: Methodological 28.4 (1994): 269-287.

Random Number Generator

In this project, we assumed the time of vehicle entering the system is uniform distribution. The minimum and the maximum of it is 0 and simulation time. We wrote a uniform generator to get the time list and then sort it to be our time schedule. The source and sink of each vehicle are also uniform distribution. They were assigned a value randomly from lane0 to lane9 expect for lane1 and lane2, since no vehicle can come out and disappear in the center of the system. The uniform random number we used is based on c++ rand() and srand() function. We first generated a random number x between 0 and 1, then change the scale by x = x*(max-min)+min. In order to test the validation of our generator, we produced hundred thousand random numbers between 0 and 1. Then use Matlab Chi-square goodness-of-fit test function to test them. The result shows below, p-value is probability of observing the given result. H = 0, that means we can accept our assumption that the random number is uniform distribution. The histogram also shows below.

Hypothesis	0
P-value	0.7912

Test result of uniform random number generator

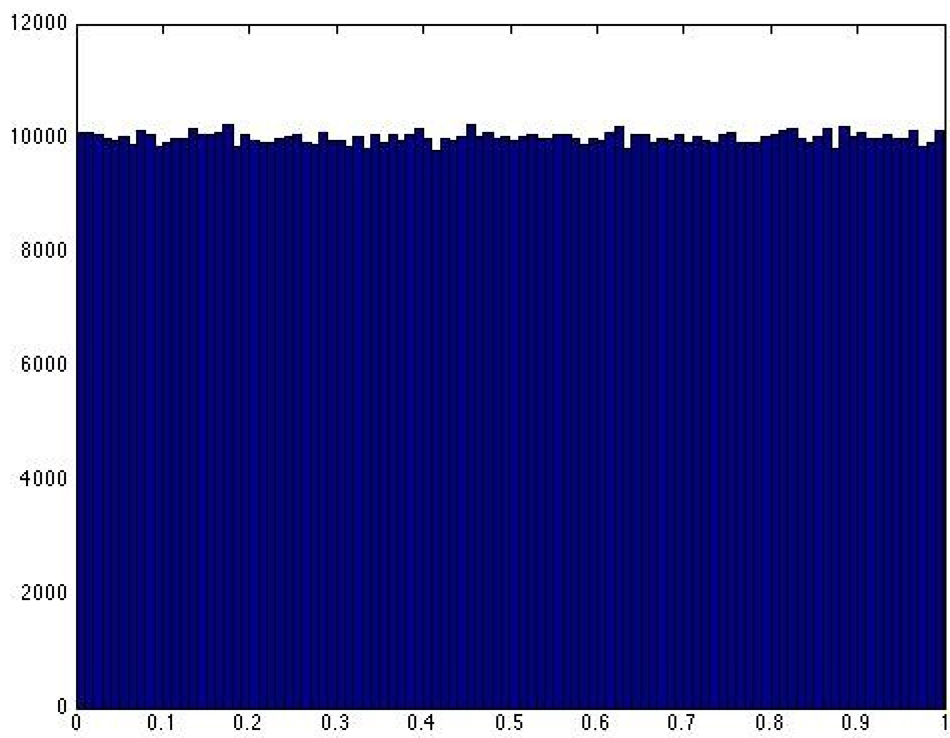


Figure 3 The histogram of uniform random number

Simulation Model

The diagram shown below is a simplified version of peachtree street, which we simulate. The lane numbers and signal numbers are marked. Our model consists of objects for cars, lanes and traffic intersections. Interaction between these objects is handled by four events, which are briefly described as follows:

1) EnterLane: EnterLane is called whenever a car tries to enter a lane. If the car is the first car in the lane (i.e. the lane was empty before that), it schedules SignalCheck for the front signals, which then remains active till the queue again becomes empty.

```
EnterLane (Car c, Time t, Lane l) {
        If l was empty before c entered {
                Schedule event SignalCheck for front signal at time t
        }
        Push car c in Lane l
}
```

2) SignalCheck: It is fired by the first car in any lane, and then remains active for that lane till it becomes empty again. It's job is to see if the front car can pass to it's next lane. If yes, it schedules the entry and exit for that car by giving a delay t_d, which is the time a car takes to pass any intersection. If not, it schedules another SignalCheck after some wait.

```
SignalCheck (Signal s, Time t, Lane l) {
        If l is empty {
                exit()
        }
        If the front car in l can pass {
                Schedule entry for that car at time t + t_d
                Schedule exit for that car at time t.
                Schedule SignalCheck at time t+t_d
        }
        else {
                Find the time for next green light.
                Schedule SignalCheck at that time
        }
}
```

3) ExitLane: Very similar to entry lane, it is called when a car wants to leave a lane

4) SignalChange: It is scheduled at times when the signals at an intersection need to be changed. It is called periodically. Each signal goes through the cycle LLRR

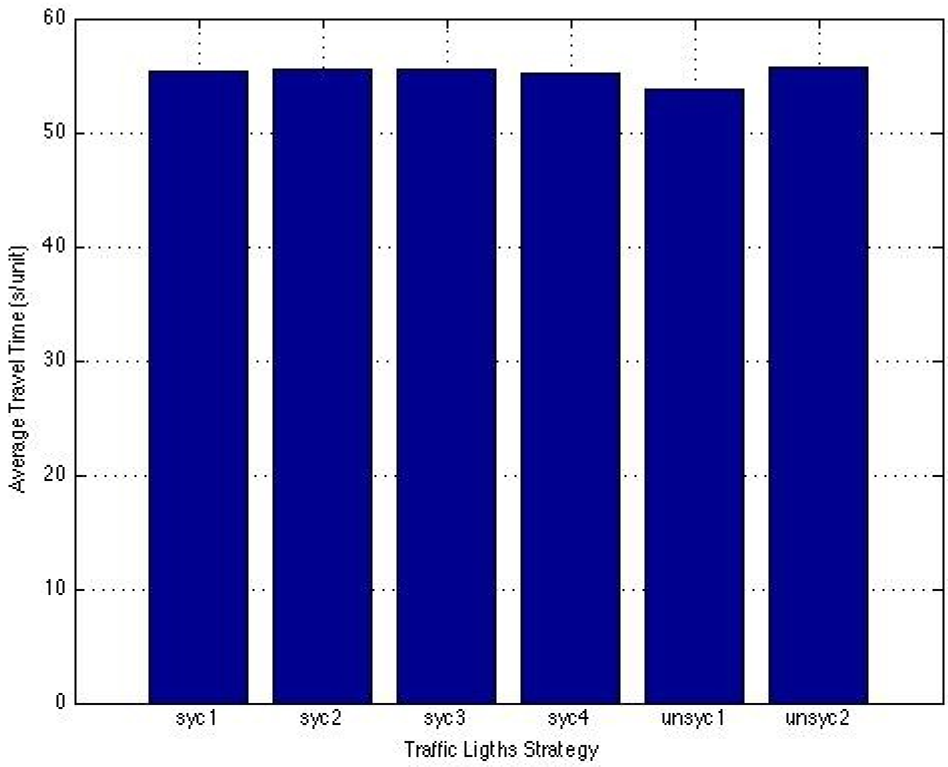
→ GGRR → RRLL → RRGG → LLRR, where L represents the left green light (cars can turn left), G represents the straight green light (cars can go straight) and R represents red where the cars stop. We assume that the cars can turn right at all times.

The simulation code implements the four events, the three objects and an event handler to queue and process all the events.

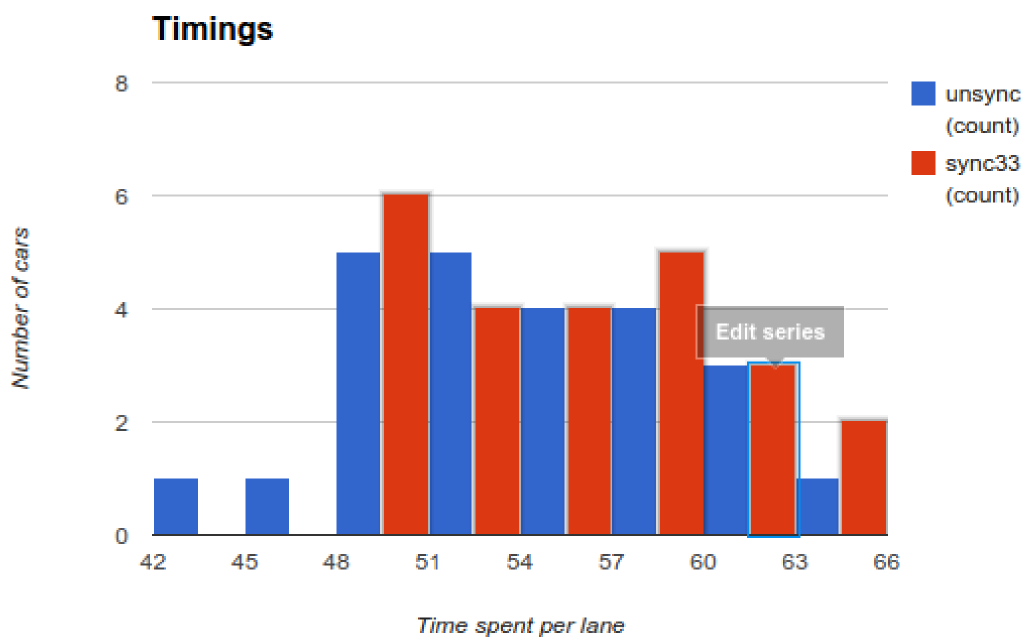
The code is implemented using C++. For running the code, please refer to the readme file.

Experiments

In our system, we consider a total of 350 cars. We ran the simulation for unsynchronized settings and various synchronization settings. Lets define sync-ij to be a synchronization scheme where signal 1 is delayed by I time units from signal 0, and signal 2 is delayed by j time units from signal 1. We tried schemes sync-00, sync-22, sync-33, sync-24. In addition we ran the unsynchronized scheme twice. Each scheme testing consisted of around 20 runs. The following picture shows the average times spent by cars on each lane.



As it is evident, there is no significant difference between the different schemes. We compare one synchronized and one unsynchronized run more carefully in the following picture.



The graph analyzes the average time for each run in the two schemes. The x axis represents the average time spent by a car in one lane, and the y axis represents the number runs in one scheme that had the particular average time for a given scheme. As it shows, the difference between the two schemes is not significant (though the unsynchronized run is slightly better off).

Conclusions

In this project, we modeled a section of peachtree street and simulated some traffic on that to find out the best signaling scheme for maximum throughput through the street. We experimented with various signaling schemes and the runs don't demonstrate any trivial choice. One reason might be that we are modeling a very small piece of road and the small variations that we have tried don't affect the traffic flow much. One future improvement might be to play with the values of signal delays and number of cars, to see if we get any significant difference.