Traffic Simulator : Project Documentation

Tommi Gröhn, Kaisa Kärkkäinen, Akseli Oinaanoja, Stas Tatun

1. Overview

Our project topic is the Traffic Simulator. The project includes basic features listed in the topic assignment as well as some additional ones. We have a working city loader that loads an open street map file (file.osm) and builds the city, the whereabouts of the buildings and the two laned roads, for our simulation based on the given file. You can also change the osm file during the running of the program.

The different types of buildings are randomly picked to be either residential, industrial or commercial buildings. The route preferences of people changes during the day. For example in the morning people prefer driving from their home to the office which is not a common thing to do in the evening. When a car reaches its destination it is destroyed. The cars in the city are also created at different speeds depending on the time of the day, for example during the night there are not many cars around. The randomization in these occasions is created with poisson distribution and bernoulli distribution and it works best with the file map7.osm.

We chose to only create cars, because the existence of pedestrians would not in our opinion add a lot of value to the simulation. We created nodes to represent the links between roads and the ending points of roads. The cars from a previous road will not be able to move forward to the next road if that road is full. The capacity of a road is defined by the length of the road. This is the way the simulation accumulates traffic jams.

The graphical user interface has a visualization of the simulation showing a map with buildings and roads based on the osm file you have chosen to upload. You can also see the time of the day and the total amount of cars on the map at all times. The traffic is shown by changing the colour of the streets. If there are more cars on the road than 25% of the maximum capacity, the street turns into a darker green. If there are more cars on the road than 50% of its maximum capacity, the street turns brown. If the road is filled with more than 75% of its maximum capacity it turns red and finally if the road is filled with its maximum capacity and therefore creates a traffic jam it turns dark red. This makes it easy to spot congestion and possible future congestion right away.

There are a few additional properties implemented. For example in the menu bar you can see different options. You can pause the simulation and then choose to continue it. You can choose to upload a different osm file. You can choose to create a histogram showing how many cars have on average driven through the streets during each hour of the simulation and you can choose to specify to analyse only one road by picking a number between 1 and the total number of roads in the map. The chosen road will be highlighted. It is possible for you to save your histogram in png format.

One of the main shortcomings of our simulation is that it doesn't have incoming/leaving traffic out of the map, i.e. the roads leading to the map borders are mostly empty. This proved difficult to implement because of discrepancies in the xml file.

2. Software structure

CITY:
Essentially, the class mainwindow will first call class city which will add to itself all the needed information based on real-world data. We use data from Openstreetmap and basically our program works for any map taken as long as it is taken from a city and it is not too heavy for simulation. The city also counts optimal routes for cars using

Dijkstra's algorithm. Even a small map will contain hundreds of roads because the class road actually represents smaller pieces of roads. This property ensures that the simulation describes the traffic quite realistic on a real world data.

MAINWINDOW:
Later on, mainwindow will create a for-loop of many small ticks. During one tick happens a lot of random events: new cars are generated on roads and cars on road finally find their way to the final destination. One single tick is a responsibility of the class city which is the reason why mainwindow calls the city during one loop.

CITY:
City is one of the most important classes of this project because it controls the classes road, car, node and building. In the class city, it is defined parameters for randomness that occurs in the city. For example, cars are generated more likely in the morning than in the evening and the distributions follow Poisson distribution which has a parameter lambda that changes every hour. For example if lambdas value for current hour is 0.3, it means on average 0.3 people are generated on the map per the time instant, which is in our case 10 seconds. Furthermore, people route preferences change during the day. For example in the morning many people drive to work from their homes but in the evening that event is unlikely. The distributions defining people's preferences can be changed easily in city.hpp.

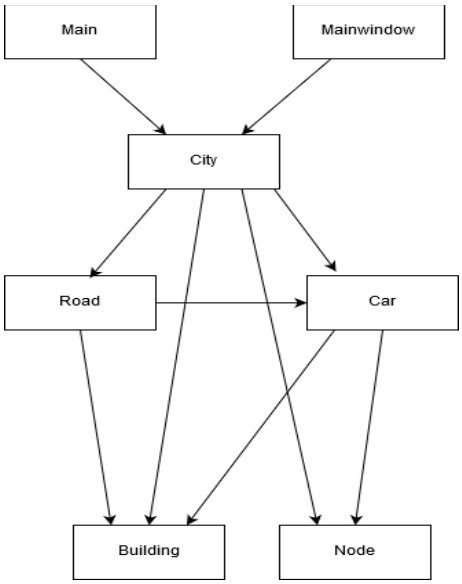
ROADS:
After, the random generations the city will create new instances of the class car and later add new cars on roads. The city does not know itself which cars are driving on which roads. This is information that belongs to the class road. The road controls the sequence of cars driving on the road. The road will also ask from the next road if a new car can be added, meaning the next road cannot be full.

CARS:

On contrary, the cars do not get much information about the flow of traffic. An instance of a class car is simply created by a city randomly generated starting point and destination and it will only keep on track on its route. After the car arrives to its final destination, it will get destroyed.

NODES and BUILDINGS:

Nodes and buildings are one of the simplest classes of the city and they do not have any complicated methods. The concept of a building is simple to grasp because buildings represent homes, offices and free time activities (f.e. stores, gyms). Nodes are everything that is needed to connect small pieces of roads together: f.e. Crossroads and end points of roads. Buildings are attached to the route map with non-visible roads that will not show up in the real map.



3. Instructions for building and using the software

How to compile the program:

Build the program by running "make" in /src/. Then run the executable named "simulation". The program uses Qt and rapidxml libraries. Qt is installed on Aalto Linux machines, and rapidxml is included in the project under /lib/

How to use the software: a basic user guide

1. Having first downloaded the software you should check that you have at least one open map source file, that you can choose to run our simulation on.
    a. You can simulate your own maps by downloading them from www.openstreetmap.org/export and moving the *.osm files to /src/. If there are not enough buildings connected by roads in the map, the simulation will exit and print an error message.
    b. Our implementation is computationally extremely expensive, so larger maps with a lot of roads and buildings will probably crash/freeze. Several maps with good performance are included in /src/.

2. Having started the simulation you are asked what file do you choose to open. You should write the name of the file in the textbox in the following format “file.osm”.
    a. Now if you chose a file that exists in the /srs/ you should see a window with a map pop out. If the file did not exist the simulation will end and you can try again by running the simulation again and inputting an existing file.

3. The simulation starts running. On the right you can see a clock telling what time it is in the simulation and under the clock you can see the total amount of cars on the move at that exact moment. The streets turn from green -> darker green ->

red -> dark red, depending on how many cars are on the road at the moment from its maximum capacity. The roads that are marked dark red are jammed.

4. Now you can choose from different tools in the menu bar.
    a. You can choose to pause and continue the simulation
    b. Set a different time
        i. When setting time you should only use integers between 0 and 8640. Those numbers give you access to every hour of the day.
        ii. For example, if you would like to see the simulation run at 1 am, set the time at 360 decaseconds, or at 12 am, then set the time at 4320 decasecond.
        iii. If the time is changed in the middle of the simulation, the histograms won't show correctly from the time before the change.
    c. Choose a different open street map to analyse
        i. If you have paused the simulation, it does not change the map before you have continued the simulation.
    d. Draw a histogram
        i. Based on the average amount of cars on all the roads per hour
        ii. Based on a specific road and draw the amount of cars per hour
            1. Additional advice: qo to the menu bar and click file and then choose road. Then it will ask you to input a number from 1 to the amount of roads total on the map. When you have chosen a number you can see the histogram on a different window and the road you have chosen has been highlighted blue.
        iii. You can choose to save the histogram in png format.

5. When you are done with the simulation you can choose to exit the traffic simulation from the menu bar, or you can press ctrl + w.

4. Testing

Quality assurance was done manually, no class/function specific tests were implemented. (Looking back, some automated tests would've saved a ton of time and effort. Checking if the simulated map matches the original XML file by hand was tedious.)

5. Work log

Division of work:
Stas made the base for the simulation based on the whole groups thoughts and plans. Tommi focused on the randomization and on how to make the simulation as close to the reality of traffic in a city. Akseli and Kaisa were in charge of the GUI implementation and divided work so that Akseli focused on making the traffic simulator heat map and Kaisa on making the histogram.

Time schedule and time spent:
During the first few weeks we mainly focused on planning what kind of city format we were going to use and how to create a simulation. At first we decided on a grid format, but then later on chose to leave it behind and challenge ourselves with the open street map format. We had already figured out the main classes and how they would work with the grid type city during the first weeks, so that took us a few steps back and gave us a bit tighter schedule to work with, but we managed our best.

During the last two weeks all of the group members used a lot of their time to make the project happen, ranging from 40 to 60 hours.