Senior Design Report

Network Simulator

Document Prepared by:

Andy Do
William Fong
Daniel Romo
Y Hoang
Dibakar Barua
Zifan Yang

Advisor:

Zilong Ye

1 May 2018

CALIFORNIA STATE UNIVERSITY LOS ANGELES
Los Angeles, California

Table of Contents

I. Introduction

II. Technologies

III. System Architecture

IV. Results and Conclusions

V. References

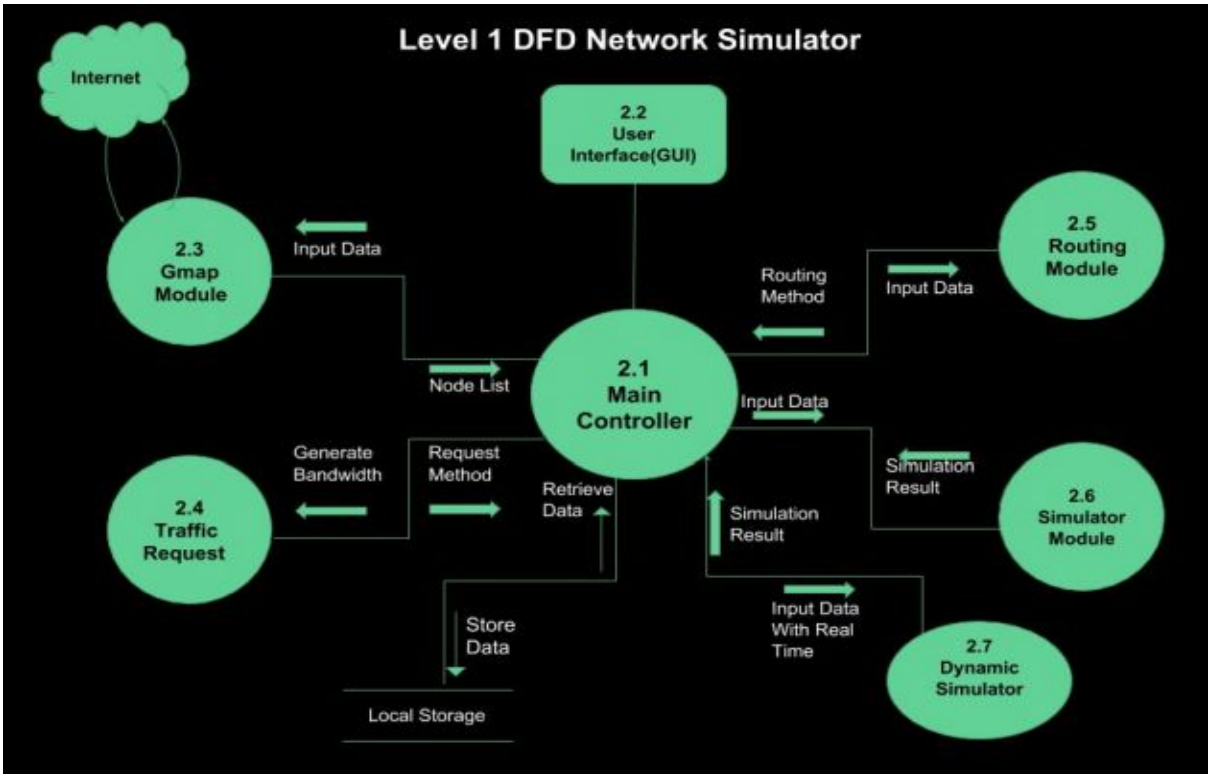
I. Introduction

The Network Simulator is an application that is capable of simulating various networking algorithms in order to find out the best algorithm. In addition, it is capable of displaying Google Maps onto the NSFNET topology. The Google Map view is interactive, letting the user add new nodes into the map which adds a new node into the NSFNET topology. The user is able to simulate an earthquake, which will disable nodes and links that are inside the earthquake range. Network Simulator will also let users select different given traffic requests and routing methods which uses the random algorithm and the routing method of user choice. In the end of each simulation, a table will appear displaying the results and will be stored locally.

II. Technologies

The Network Simulator application is mostly coded with Java programming language. For the graphical user interface, it is mostly coded with JavaFX. JavaFX is designed to provide sophisticated GUI features such as smooth animation, web views, and styles. A design tool called SceneBuilder was used to build the layout of the application. This tool generates FXML code, which is an XML based markup language that is used to create user interfaces. To allow users to create network diagrams or visualize network traffic, a map component was added to the network simulator application using the GMapsFX API.

III. System Architecture



GMap Module - This module allows users to select points or nodes on different locations on Google Maps that correspond to their latitude and longitude and give ability to add a polyline in between the nodes to build network diagram.
Traffic Request Module - This module allows users to select a randomization method, such as gaussian and uniform, that the simulator uses to generate a random bandwidth for requests.
Routing Module - This module allows users to choose between 6 different algorithm paths, such as Shortest Path First and Least Used First, that the simulator uses to determine the number of transponders.
Simulator Module - This module generates the bandwidth process based from the traffic method and routing algorithm that is chosen.
Dynamic Simulator Module - This module is more likely real world simulation, which takes three variable input from user - time interval, request count, and maximum bandwidth.

There are three inputs for the simulator: topology, traffic request method, and routing method. The topology is a file that is stored locally and is able to simulate a network from the nodes, links, and link distances specified in the file. There are three traffic request methods to choose from: random, gaussian, and uniform. These are randomization methods the simulator uses to generate a random bandwidth when it is generating the requests. There are six routing methods to choose from: SPF, LUF, MUF, OPT, MUX, Hybrid. These are shortest path methods the simulator uses to determine the number of transponders needed. SPF determines the shortest path between the starting and destination node using Dijkstra's shortest path algorithm. Both LUF and MUF initially use the K-shortest path algorithm to determine a set number of shortest paths that is specified. Then, LUF picks the path that has the least used node while MUF picks the path that has the most used node. Both OPT and MUX determines the number of transponders based on a direct path between the starting and destination node. Hybrid uses a threshold system, where if the bandwidth of a request is under a threshold, SPF is run; otherwise, OPT is run.

One of the key functions that was implemented is the dynamic simulator. The dynamic simulator allows for a more real-world simulation. It uses three variables for its inputs: time interval, request count, and maximum bandwidth. Each request will be generated with a random start time and a random end time within the indicated time interval. It will also have a random bandwidth that is within the maximum bandwidth. The dynamic simulator will check each request to see if there is enough available bandwidth before putting it into the scheduler. If there is not enough bandwidth, the request will be dropped and logged. Otherwise, it will run for the duration of the request and release its bandwidth when finished. The dynamic simulator can only be used for the Shortest Path First, Least Used First, and Most Used First routing algorithms. The GUI will display the option to enable the dynamic simulator when selecting one of these routing algorithms.

An additional key function that was implemented is the dynamic interaction between the software and the user. Google Maps was integrated into the user interface to have a cleaner representation of a NSFNET topology. Each marker on the Google Maps represents a node, and each line connecting the markers are links. Using Google Maps, the distance between two points are found to design the NSFNET topology. The user interface contains a left panel in order to display information, such as the location of each marker. Users can click directly onto the map to add or remove additional markers, earthquakes, and/or connections. The user interface is designed to make the interaction as simple as the click of a button. When the user is done designing a new NSFNET topology, the user can save the topology structure into a text file directly to their computer. The user can later use the same topology even after they close the application.

IV. Results and Conclusions

Network Simulator performs simulations of various infrastructures and network traffic. It is mainly coded in JavaFX with the Google Maps API. There are three types of inputs are accepted by the simulator: topology, traffic request method, and routing method. Three options are available for users to pick for traffic request method, they are: random, gaussian, and uniform. For the routing method, there are several routing algorithms available including SPF, LUF, MUF, OPT, MUX, and Hybrid. A dynamic simulator was also implemented to simulate a more "real world" scenario. Compared to a regular simulator, the dynamic simulator has three more variables: time interval, request count, and max bandwidth. The time interval and bandwidth is generated randomly. Moreover, there is a super user-friendly GUI that embeds Google Maps into the simulator. This allows users to add or remove nodes or edges by simply clicking on the map directly. Lastly, there is a feature that allows users to save their topologies. The node coordinates and edge connections can be saved to a text file. The simulator can read this text file and load the nodes and edges onto the map automatically. A feature to implement in the future would be the use of earthquake data to simulate natural disasters. The purpose of the earthquake events is to test the survivability of each routing method after disconnections caused by these disasters.

V. References

GMapsFX

http://rterp.github.io/GMapsFX/

Gluon Scene Builder

http://gluonhq.com/products/scene-builder/