Route Finding

Programming Project 2 Report

Luke Hutton [sc17ljth]

Contents

[Project Summary 2](#_Toc508618057)

[Design Plan 2](#_Toc508618058)

[Large scale design plan 2](#_Toc508618059)

[Aim of the project 2](#_Toc508618060)

[Program input 2](#_Toc508618061)

[Program output 2](#_Toc508618062)

[Dijkstra’s algorithm 2](#_Toc508618063)

[Priority queue implementation 2](#_Toc508618064)

[Medium scale design plan 2](#_Toc508618065)

[Code modules 2](#_Toc508618066)

[Test Plan 2](#_Toc508618067)

[Schedule 2](#_Toc508618068)

# Project Summary

A graph or network is collection of points (nodes) connected by a series of lines (edges). Graphs are used in an area of maths called discrete maths to model optimisation problems that involves finding an efficient solution to a problem. Solving an optimisation problem means finding the best solution out of many feasible solutions.

This project will involve creating an application that will find the shortest path between two nodes on a network. The network will be given in the form of a file; the application will then read this and output the shortest path. In order to create such an application, the following things need to be taken into consideration: importing the data, the data structure used to store the network and the algorithm used to determine the shortest path.

# Design Plan

## Large scale design plan

### Aim of the project

The ultimate aim of the project is to find the shortest path between two input nodes. This can be done using Dijkstra’s algorithm in O(E log V) time using a priority queue with a binary heap. The program must be able to accept an XML input from a file, read the nodes and edges from such file and then output the shortest path in a graphical way to the user.

### Program input

The programs input will consist of an XML file containing the data used to represent the network. Some of this data will be used and other parts won’t. A plugin called libxml2 can be used to read and parse XML files s I will therefore use this in my project to read the data file.

Another input from the user will the 2 points to find the shortest path between. The user will enter these once prompted through the command line.

### Program output

The program will output a graphical network consisting of nodes and edges. I will do this using GNUplot as we have already used it before and it is an easy to use application. With gnuplot you can change the line colour of elements. Therefore, it will be easy to display the shortest path calculated by the program.

### Dijkstra’s algorithm

Djikstra’s algorithm can be used to calculate the shortest path between two nodes on a network. This is the pseudo code I will base my implementation off of in c.

"Find shortest path"

FOR x IN numberOfNodes

distance(x) ← infinity

processed(x) ← false

parent (x) ← null

ENDFOR

distance(SourceNode) ← 0

WHILE (there are still nodes left to process)

Let 'sNode' be a node which hasn't been processed that has the smallest distance(node)

ENDWHILE

IF (sNode ← destinationNode) THEN

Exit While Loop

ENDIF

processed(sNode) ← true

FOR Each unprocessed parent node, dNode

IF (distance(node) + weight(sNode, dNode) < distance(dNode)) THEN

distance(dNode) ← distance(sNode) + weight(sNode, dNode)

parent(dNode) ← sNode

ENDIF

ENDFOR

# Find path

node ← destinationNode

WHILE node != sourceNode

Append node to the beginning of the path list

node ← parent(dNode)

ENDWHILE

Append sourceNode to the beginning of the path list

In order to implement this algorithm efficiently we need to use a priority queue, an implementation of which is described below.

### Priority queue implementation

## Medium scale design plan

### Code modules

# Test Plan

# Schedule