COS 226 Programming Assignment Checklist: Map Routing

Special collaboration policy for Spring '03: for this assignment, you are permitted to work jointly with one classmate. If you choose to do so, you and your partner should submit your code and the readme.txt jointly (under one login name only). You and your partner will receive the same grade.

Frequently Asked Questions

What if there are multiple shortest paths? Print out any one you like.

What's a good way to check my answers? Use the client program `distances.c` which only prints out the shortest path distances. You can `diff` the results against the ones produced by our reference solutions.

What should I do if the two vertices are not connected? Print that their distance is "infinity" or print out that they are not connected.

Is it OK to print the shortest path in reverse order. Yes. One hack to print it in the correct order is to switch `s` and `d`. Otherwise you could use a stack. But don't worry about it on this assignment.

The turtle graphics output doesn't view properly with `turtle.c`. Please use the version of `turtle.c` supplied in the `map` directory. We've customized it to print things out in landscape.

I can't see the blue dots that are supposed to represent the vertices that the algorithm examines. You might need to zoom in. Or you can change the value of `SIZE` in `point.h`.

How fast should my algorithm be? Fast.

Input, Output, and Testing

Input and output. Here are a number of sample input files.

Extra credit. Create your own map file. If it's sufficiently interesting, we'll award extra credit and post them for other students to use. It can be of the US, Princeton, or any other recognizable region. For example, delete all roads in Nebraska from the US map (perhaps, for drivers who have an unpaid speeding ticket in the state).

Submission and readme

Submit everything along with the client program distances.c. (Don't submit paths.c or plotit.c.) We'll jointly compile with "`gcc *.c`" and execute with "`a.out < usa-5000.txt`". We may also substitute a different client for distances.c, e.g., paths.c.

Here is a template readme.txt file. It should contain the following information:

- A high-level description for the design of your algorithm, and what influenced your decisions.

Unix users may find the shell script `timeit.csh` useful for computing a table of CPU times.

Getting started

- Download the directory `map`. This directory is mirrored on arizona at `/u/cs226/pub/map`. It contains a reference implementation and sample input files.

- Compile our bare bones implementation with

```
% gcc plotit.c graph.c pqindex.c point.c -o plotit
% gcc turtle.c -o turtle -lm
% plotit < usa.txt | turtle > usa.eps
```

The client `plotit.c` peforms a single shortest path query and plots the results in turtle graphics. You can view the results with your favorite PostScript viewer.

- When processing a single shortest path query, the bare bones implementation is essentially optimal since the majority of time is spent reading in the input and plotting it. The client program `paths.c` processes many shortest path queries and prints the results as text instead of turtle graphics.

```
% gcc paths.c graph.c pqindex.c point.c -o paths
% paths < usa-5000short.txt
```

The program is painfully slow. Your goal is to optimize `GRAPHsp()` so that you get the same output, but faster.

COS 226 Home Page
wayne@cs.princeton.edu
Last modified: April 10, 2003