Project Proposal

Team Members:

Raghuveer Raghavendra (rraghav2)

Prasanna Jeevan Devanur Nagabhushan (pdevanu)

Problem Description:

We are planning to implement "Save Gas Map" application for Google G1 dev phone on Android platform[1,2,5]. The application allows the user to enter multiple locations that the user is interested to visit and finds the optimal route between those locations and lists them on the GPS.

We all use Google maps to get from point A to point B. Given the address of the start location (A) and the end location (B), Google maps plots out a nice route for the trip and gives an estimate of the travel time. But if the trip includes multiple points, the user has to specify the sequence of locations himself before Google maps can give a route. It would be interesting if Google maps can give the shortest path for a set of locations. We propose to find an optimal path for a given set of locations and plot the sequence of location on Google maps.

Proposal:

Input:

The user inputs the locations by typing in the addresses, as one would do in Google maps (maps.google.com). Along with the locations, the user can specify a relative order among his locations. He can also fix the order for a part of his journey. An example will make the idea clear.

Let's say I want to start from home (A), visit the library (B) to scan/print some documents, mail them at the post office (C), buy some groceries (D) and get back home(A). It does not matter (mostly) if I buy the groceries first or go to the library first. However, I do not want to go to the post office before going to the library. We will allow the user to specify constraints for his trip, wherein he specifies a relative order among a subset of the locations.

There may be strict constrains too. Consider a situation when you want to send a post (A), buy groceries (B), and re-fuel (C). The post office may close in short duration, so you may want to go there first. We allow the user to specify such constraints too.

Output:

The output displayed is a map with a route. The route is most likely the shortest route possible (see section Feasibility).

Feasibility:

This is similar to traveling salesman problem (TSP) which is NP Complete and NP Hard. Starting from a location, for covering N locations and arriving back at a starting location, there are N! possible routes. Also, in order to decide the shortest route, there needs to be square (N) calls to the Google maps system. Each call to the Google maps system querying the distance b/w two locations is deemed expensive. We want to minimize this. Hence we make an assumption that for two locations that are geographically closer (in terms of their latitudes and longitudes), there exists roads connecting the locations that are proportionally closer. With this assumption, we just have to make N Google maps API calls just to query the positions of the locations (latitude, longitude). The longitude and the latitude can be obtained using the API's provided by the Android such as getLatitudeE6 () and getLongitudeE6 (). Once the information is available, a complete graph of N+1 nodes is created and a brute force algorithm calculates the shortest path (We also like to consider other heuristics and approximation algorithms that are used to solve the TSP[3]). Since generally N will be small in practical cases the brute force methods works fine. The downside of the assumption is that, two locations may be geographically close but the roads connecting the locations may be round about, making the path computed by our algorithm not the minimal one.

The application can be made to feed the path to existing GPS software for good software integration.

Project web page:

http://www4.ncsu.edu/~pdevanu/Site/Project.htm

References:

1. http://source.android.com/
2. http://developer.android.com/
3. http://en.wikipedia.org/wiki/Traveling_salesman_problem#Heuristic_and_approximation_algorithms
4. http://iphoneapplicationlist.com/
5. http://androidcommunity.com/