CSE 5400 Interdisciplinary CS

Assignment 2:Navigation

Instructor:Dr.Chan

Student name:Lingfeng Zhang

ID:9017490960

In the class, we talked about how to get a shortest way in a graph for one single source. In this time, each link has a weight and all of them are different.So we studied and analyzed the Dijkstra algorithm. And we also understand that why the Dijkstra algorithm can guarantee the path which is found is the shortest path.

Back to the navigation, since we are doing something like google map,so what we do is not only find the shortest path,but also give a guide to the user. Thus, we need to determine which direction is to go at every intersection. So that we came up with several methods to solve this problem and implement it in our source code.Finally, in order to be similar to google map, we also need to downsize the out put and ignore some useless information.

A. As you know, a cell phone or car is highly mobile and could change locations frequently. Hence,localization needs to be efficient so that the location can be updated frequently to be accurate.

i. By using these variables: x1,y1,d1;x2,y2,d2;x3,y3,d3,we have three formulas:

$(x1-x)^2+(y1-y)^2=d1^2$;$(x2-x)^2+(y2-y)^2=d1^2$;$(x3-x)^2+(y3-y)^2=d1^2$;

Step 2:

recount these formulas,we can get another three formulas,

$x1^2-2x1*x+x^2+y1-2y1*y+y^2=d1^2$ .....1 and

$x2^2-2x2*x+x^2+y2-2y2*y+y^2=d2^2$.....2 and

$x3^2-2x3*x+x^2+y3-2y3*y+y^2=d3^2$ .....3 and

Step 3:

Use equation 1 - 2, we can get the equation:

$(x1^2-x2^2)-2x*(x1-x2)+(y1^2-y2^2)-2y*(y1-y2)=d1^2-d2^2$.....4

and use equation 1 - 3, we can get the equation:

$(x1^2-x3^2)-2x*(x1-x3)+(y1^2-y3^2)-2y*(y1-y3)=d1^2-d3^2$.....5

Step 4:

Then we have the unknow x and y, but we have two equations.

If x1!=x2,x1!=x3,y1!=y2,y1!=y3, then we can simplify the equations 4 and 5.

$2x*(x1-x2)+2y*(y1-y2)=(x1^2-x2^2)+(y1^2-y2^2)-(d1^2-d2^2)$.....6

$2x*(x1-x3)+2y*(y1-y3)=(x1^2-x3^2)+(y1^2-y3^2)-(d1^2-d3^2)$.....7

Then change equation 6 as

$2x=[(x1^2-x2^2)+(y1^2-y2^2)-(d1^2-d2^2)-2y*(y1-y2)]/(x1-x2)$.....6

take the value of 2x into equation 7,then we have

$2y*(y1-y3)=(x1^2-x3^2)+(y1^2-y3^2)-(d1^2-d3^2)-$
$[(x1^2-x2^2)+(y1^2-y2^2)-(d1^2-d2^2)-2y*(y1-y2)]*(x1-x3)/(x1-x2)$

Then we can get value of y:

$y=\{(x1^2-x3^2)+(y1^2-y3^2)-(d1^2-d3^2)-[(x1^2-x2^2)+(y1^2-y2^2)-(d1^2-d2^2)-2y*(y1-y2)]*(x1-x3)/(x1-x2)\}/2*(y1-y3)$.....8

take the y into equation 6,then we can get the value of x:

$x=[(x1^2-x2^2)+(y1^2-y2^2)-(d1^2-d2^2)-2y*(y1-y2)]/2*(x1-x2)$.....9

So that,by using the equation 8 and 9 we can get x value and y value.

ii、How many arithmetic operations are needed to calculate x and y?

To calculate y, go back to the equation 8,we will have 32 arithmetic operations.

While after we calculate the y, we will have 16 arithmetic operations to calculate x.

B. Compare the two algorithms:

i Quality of output

By using Dijkstra algorithm,we can get the correct answer,that means the program can output all the shortest paths.

But,since I use the float to store all the variables,so if the distance between two vertexes is an irrational number,the calculation of the distance will be slight different,and it will influence the shortest path while it outputs.Here is the example of the problem:in the NewYork.txt,if you set the start node as “Austin St” house number“46”,and destination node as “63rd Ave” house number“42”,program will out put 3 shortest ways; if you set the start

node as “Austin St” “60”, problem will just out put 2 shortest ways.So it’s the problem about the float calculation,which affect the shortest path.

However, by using the Greedy algorithm, the program will output three probabilities:1.repones error-means can not find the path to the destination. 2.A path to the destination-which is not the shortest path.3.Out put the shortest path-only output one path.

ii Time/speed

For Dijkstra algorithm, because we have a set to store the visited nodes, and this set is increasing all the time.For each time we need to visit all the nodes in this set and check if there are some potential nodes to be the new visited node. So under the worst situation,the calculation will be 1+2+3+...+(n-1) n is the total number of the nodes in the graph.Time complexity will be $O(n^2)$.

For the Greedy algorithm, since every time we will find a shortest link to the present node,and go through this link to the next node.So that, under the worst situation, it will go over all the nodes in the graph.Then the time complexity will be O(n).n is the number of nodes.

iii Space/memory

For the Dijkstra algorithm, since we use the adjacency list to store the neighbors for each node,so we need to store all the nodes and all

the links. So the space complexity will be O(E+n). E represents the number of links,and n represents the number of nodes.

While to the Greedy algorithm, I use the same adjacency list to store the nodes information,so its space complexity will be O(E+n).