CSCI 2270 Lecture Notes

4/17/19

Dijkstra’s Algorithm

* BFT finds shortest path in an unweighted graph, but does not work for weighted graphs
* Instead, we can use an algorithm such as Dijkstra’s

Need to update the vertex struct:

struct vertex{

string key;

vector<edge> edges;

bool solved; //similar to visited

int distance; //total distance from starting distance

vertex \*parent; //later retrace steps

}

struct edge{

vertex \*v;

int weight;

}

vertex Dijkstra(start, end){

//given the starting an ending vertices, find the shortest path and return the ending vertex

vStart = search(start)

vEnd = search(end)

vStart.solved = true

//create a list to store solved vertices

solvedList.add(vStart)

while(!vEnd.solved){

//set a min distance to some huge value

minDist = INT\_MAX

// a pointer to keep track of solved node

solved = null

//iterate across the list of solved vertices

for x = 0 to solvedList.end{

s = solvedList(x)

// now iterate s’s adj list

for y = 0 to s.edges.end{

if(!s.edges[y].v.solved)

calc distance from vStart

dist = s.distance + s.edges[y].weight

//check if this distance is less than smallest distance found so far

if(dist<minDist)

solvedV = s.edges[y].v

minDist = dist

s.edges[y].parent = s

}

}

solvedV.distance = minDist

solvedV.parent = parent

solvedV.solved = true

solvedList.add(solvedV)

}

}