CSCI 115 Lab

Week 17 - Dijkstra's shortest path algorithm

- Professor: Dr. Matin Pirouz
  Email: mpirouz@csufresno.edu

- TA: Shreeja Miyyar
  Email:shreejarao12@mail.fresnostate.edu

Table of Contents

Dijkstra's shortest path

- It is an algorithm which computes the shortest path from a single source vertex to all the other vertices in a weighted directed graph.
- The algorithm does not work for negative edge weights.

Approach:

- Create an empty list initially to track all the vertices included in the shortest path.
- Initialize the distance from source to all the vertices as infinity and distance from source to source as 0.
- Iterate V times:
  - If the vertex is not there in the tracking list then get the vertex with the least distance.
  - Update the cost of this vertex from the source vertex.
  - Iterate all its adjacent vertices and do the same step for those as well.

Dijkstra's Algorithm

```
dijkstra(G, V, s) {
  // Let the distance_list denote the array to store the distance from the source to rest of the vertex
  // Let tracking_list represent an array to track the vertex which has already been visited
  Iterate V times {
     distance_list[i] = infinity
     tracking_list[i] = false
  }
  distance_list[s] = 0
  for each vertex {
    if tracking_list[v] is false then find minimum distance among all vertices and return that vertex v
    tracking_list[v] = true
    for adjacent vertex u {
          if vertex u and v are connected and distance_list[u] + G[u][v] < distance_list[v] {
               distance_list[v] = distance_list[u] + G[u][v]
          }
     }
  }
// Print the shortest distance from source to rest of the vertices
  for i iterating the number of vertices {
     print i and distance_list[i]
}
main () {
  // Let the input adjacency matrix be denoted as G
  // Let V represent the total number of vertices
  // Let s represent the source vertex
   dijkstra(G, V, s)
}
```

Lab Assignment

Hints and Coding Guidelines:

- Create a main function which accepts number of vertices, source vertex and adjacency matrix as inputs.
- Create a function dijkstra which accepts the 3 inputs defined above as arguments. Create the necessary temporary variables as shown in the algorithm.
- To output the distance from source vertex to rest of the vertices in tabular format, here is code snippet:

```
cout << Vertex << "\t\t" << Distance from source vertex << endl

for (int i = 0; i < V; i++) {

   cout << i << "\t\t" << dis[i] << endl;

}
```

Questions?