Shortest Paths

ICT Officers

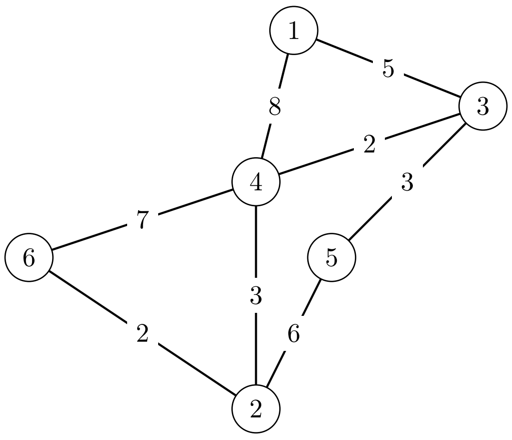
March 2, 2018

1 Introduction

Graphs are useful when finding the connectivity between multiple states. Oftentimes, when we are given a graph, we want to find the shortest path between one vertex to the next. As discussed in the last lecture, one can use breadth-first search (BFS) to find the shortest path from a source vertex to a given vertex. There are other algorithms that can help us give the shortest path.

2 Dijkstra's Algorithm

Dijkstra's algorithm gives the shortest path starting from a source node to a vertex.



Dijkstra's algorithm consists of the following:

1. We first set all the distances from the source vertex as infinity, except for the source vertex, which will be set to 0.
2. As we iterate though, we choose the vertex with the smallest distance from the source and has not been visited yet.
3. We go through all its neighbors and update their distances from the source vertex by finding the min(current distance from source, current vertex's distance + weight).

When we implement Dijkstra's algorithm, we often use a PriorityQueue to store the vertices and remove the vertex with the least value. The runtime for this algorithm is about $O(V^2)$ with V representing the number of vertices in the graph

Algorithm 1 Dijkstra

```
1: for all vertices i in the graph do
2:     source(i) = infinity
3:     visited(i) = false
4: source(0) = 0                    ▷ Let the source be vertex 0 but can be any vertex
5: while there are still vertices unvisited do
6:     vertex v = vertex with the minimum distance from source and is unvisited
7:     visited(v) = true
8:     for neighboring vertices n of v do
9:         source(n) = min(source(n), source(v) + weight)
```

3 Floyd Warshall

Sometimes, we want to find the minimum distance between any pair of vertices. Using Dijkstra's, we would have to repeat the process letting each vertex be a source node. This will run in $O(N3)$, but it involves recreating the paths each time. Floyd Warshall algorithm provides a much easier way to find the shortest distance between each node without determining the paths.

The Floyd Warshall algorithm considers the following. Given a vertex A and vertex B with some distance, we can find an intermediary vertex C, which is connected to both vertex A and B. If the sum of the distance from A to C and C to B is less than the current distance A to B, replace. By computing distances using these intermediate vertices, we can eventually find the shortest distance for each pair of vertices.

Floyd Warshall's algorithm consists the following:

1. Create an adjacency matrix and set the distances between vertices directly next to each other as the weight and set the rest as INF.
2. Iterate through all the vertices as a possible intermediate vertex c. Then, go through all possible vertex a and vertex b pairs, checking whether dist(a,c) + dist(c, a) is less than the current dist(a, b)

Algorithm 2 Floyd Warshall

```
1: adjmatrix[V][V]                 ▷ V represents the number of vertices
2: for all vertices a in the graph do
3:     for all vertices b in the graph do adjmatrix[a][b] = weight(a, b)
4: for all vertices c in the graph do
5:     for all vertices a in the graph do
6:         for all vertices b in the graph do
7:             if adjmatrix[a][c] + adjmatrix[c][b] < adjmatrix[a][b] then adjmatrix[a][b] = adjmatrix[a][c] + adjmatrix[c][b]
```

4 Problems

- Ada the Ladybug loves trips. She travels around world taking photos and souvenirs. This week she went to Buganda. Common Tourist would surely travel around main city and some conurbations, but Ada has different politics. She wants to go as far as possible (because photos from outlying places are much more valuable).

Problem is, that Buganda is very large so she can barely guess such places. Luckily, you are around so she asked you for help. Can you tell her, how far and how many cities are most distant (if the shortest path is used)?

Input:The first line will contain three integers $1 \leq N \leq 5*105$, $0 \leq M \leq 106$, Q, the number of cities in Buganda, the number of roads and number of queries (possible arrival cities).

Then M lines follow, with three integers $0 \leq A$, $B < N$, $0 \leq L \leq 10$, A, B are cities, which the (bidirectional) road connets and L is length of the road.

Afterward, Q lines follow, each with number $0 \leq qi < N$, meaning the city of arival.

You are assured that max(N,M)*Q will be always lesser/equal than 107

Warning: Since we are in real world and not in some "graph theory", multiedges and self-edges are completely valid!

Output For each query print two numbers: The distance of most distant place(s) and number of such places.

- Consider a very strange barn that consists of N stalls (N < 2500). Each stall has an ID number. From each stall you can reach 4 other stalls, but you can't necessarily come back the way you came.

  Given the number of stalls and a formula for adjacent stalls, find any of the 'most central' stalls. A stall is 'most central' if it is among the stalls that yields the lowest average distance to other stalls using best paths (USACO 1996).

- After eating too much fruit in Farmer John's kitchen, Bessie the cow is getting some very strange dreams! In her most recent dream, she is trapped in a maze in the shape of an N×M grid of tiles (1≤N,M≤1,000). She starts on the top-left tile and wants to get to the bottom-right tile. When she is standing on a tile, she can potentially move to the adjacent tiles in any of the four cardinal directions.
  Each tile has a color, and each color has a different property:
  If a tile is red, then it is impassable.
  If a tile is pink, then it can be walked on normally.
  If a tile is orange, then it can be walked on normally, but will make Bessie smell like oranges.
  If a tile is blue, then it contains piranhas that will only let Bessie pass if she smells like oranges.
  If a tile is purple, then Bessie will slide to the next tile in that direction (unless she is unable to cross it). If this tile is also a purple tile, then Bessie will continue to slide until she lands on a non-purple tile or hits an impassable tile. Sliding through a tile counts as a move. Purple tiles will also remove Bessie's smell. (If you're confused about purple tiles, the example will illustrate their use.)

  Please help Bessie get from the top-left to the bottom-right in as few moves as possible.
  INPUT FORMAT (file dream.in): The first line has two integers N and M, representing the number of rows and columns of the maze.
  The next N lines have M integers each, representing the maze:
  The integer '0' is a red tile. The integer '1' is a pink tile. The integer '2' is an orange tile. The integer '3' is a blue tile. The integer '4' is a purple tile. The top-left and bottom-right integers will always be '1'.
  OUTPUT FORMAT (file dream.out): A single integer, representing the minimum number of moves Bessie must use to cross the maze, or -1 if it is impossible to do so. (USACO Gold 2015)