Graph Implementation in Scala

Sumeth SIRIPAKORN 6080037

1 Introduction

- The goal of this project is to provide a graph representation using Scala.

2 Implementation

2.1 Vertex Representation

- I use a Char to label each Vertex(Node), eg. V1 = ‘A’, V2 = ‘B’

2.2 Edge Representation

- I use an array of size 2 to represent an edge. The first index of an edge is the source, and the second index is the destination(aka. from, to).
- An edge also has a weight attribute to represent the weight from source to destination. The initial value of the weight is set to 0. The Edge class used by both the Undirected Graph Class and Directed Graph Class.

2.3 Graph Representation

2.3a Undirected Graph

- The Undirected Graph has two main attributes, *vertices and edges*. The field *vertices* is represented by an array that holds all of the vertices in the graph. The field *edges* is represented by an array that holds all edges(bi-directional) eg. if there is an edge from ‘A’ to ‘B’, the edges will also contain an edge from ‘B’ to ‘A’.

2.3b Directed Graph

- The Directed Graph Class also has three main attributes, *vertices, edges, and in-degree*. Both *vertices* and *edges* representation are the same as the Undirected Graph Class. The in-degree is represented by a Map, the key is a vertex, and the value is the degree coming in(number of edges coming in). The field in-degree is mainly used for finding a topological ordering.

3 Sample Algorithm

3.1 Breadth-first search (BFS)

- The method bfs() is available in both Undirected Graph and Directed Graph. It takes in one input parameter, a source vertex. The function is going to return an array of vertices visited by starting at the source vertex traversing the graph.

3.2 Shortest Reach

- The method shortestReach() is available in the Undirected Graph Class. It takes in one input parameter, a source vertex. The function is going to return a Map of key->vertex and value->distance from the source to that vertex. This method assumes the default weight between two vertices is one unit distance.

3.3 Topological Sorting

- The method topSort() is available in the Directed Graph class. It takes in zero input. It makes use of the field *in-degree*. This function is going to return a topological ordering of a graph represented by an array. This function is going to return only one out of possibly many topological ordering.