Sarah Kirby

Module 7 Assignment

Results:

Top 5 by eigen centrality:

vp[personid->29] 0.16888846591827394

vp[personid->39] 0.16350272420954787

vp[personid->141] 0.16291247250083749

vp[personid->155] 0.14725453571254818

vp[personid->175] 0.14583741148325877

Top 5 by degree centrality:

vp[personid->39] 0.08620689655172414

vp[personid->141] 0.07931034482758621

vp[personid->25] 0.07931034482758621

vp[personid->281] 0.07586206896551724

vp[personid->111] 0.07586206896551724

Resulting graphml file is included in Module7.zip under “GraphDatabases”

Program:

package assignments;  
  
import graph.GraphUtils;  
import graph.MGraph;  
import org.apache.commons.math3.linear.EigenDecomposition;  
import org.apache.commons.math3.linear.SparseRealMatrix;  
import org.apache.tinkerpop.gremlin.process.traversal.dsl.graph.GraphTraversalSource;  
import org.apache.tinkerpop.gremlin.structure.Vertex;  
import org.apache.tinkerpop.gremlin.tinkergraph.structure.TinkerGraph;  
  
import java.util.List;  
  
import static org.apache.tinkerpop.gremlin.process.traversal.Order.*decr*;  
  
public class Module7 {  
  
 private String GRAPH\_OUTPUT = "GraphDatabases\\students2.graphml";  
  
 private String GRAPH\_INPUT = "GraphDatabases\\students.graphml";  
  
 private final String DEGREE = "degree";  
  
 private final String EIGEN\_CENT = "eigenCentrality";  
  
 private final String DEGREE\_CENT = "degreeCentrality";  
  
 public Module7() {  
 TinkerGraph tGraph = GraphUtils.*readGraphML*(GRAPH\_INPUT);  
 GraphTraversalSource g = tGraph.traversal();  
  
 //Adjacency  
 MGraph mGraph = new MGraph(tGraph, false);  
 SparseRealMatrix adjacency = mGraph.getAdjacency();  
 EigenDecomposition eigenDecompAdj =  
 new EigenDecomposition(adjacency);  
 double[] eigenAdj = eigenDecompAdj.getRealEigenvalues();  
 int maxIndex = MyUtils.*getMaxIndex*(eigenAdj);  
 double[] principalEigenVec =  
 eigenDecompAdj.getEigenvector(maxIndex).toArray();  
  
 //Calculate and save centrality measures  
 for (Vertex v : g.V().fold().next()) {  
  
 //save eigenvector centrality  
 int index = mGraph.getVertexIndexFromID(v.id().toString());  
 v.property(EIGEN\_CENT, principalEigenVec[index]);  
  
 //save degree centrality  
 double degree =  
 ((Long)v.property(DEGREE).value()).doubleValue();  
 v.property(DEGREE\_CENT,  
 degree / (MyGraphUtils.*countVertices*(tGraph) - 1));  
 }  
  
 //get top 5  
 System.*out*.println("\nTop 5 by eigen centrality:");  
 List<Vertex> top5Eigen =  
 g.V().order().by(EIGEN\_CENT, *decr*).next(5);  
 top5Eigen.iterator().forEachRemaining(  
 s -> printProperty(s, EIGEN\_CENT));  
  
 System.*out*.println("\nTop 5 by degree centrality:");  
 List<Vertex> top5Degree =  
 g.V().order().by(DEGREE\_CENT, *decr*).next(5);  
 top5Degree.iterator().forEachRemaining(  
 s -> printProperty(s, DEGREE\_CENT));  
  
 //save graphml file  
 GraphUtils.*saveGraphML*(tGraph, GRAPH\_OUTPUT);  
  
 }  
  
 public void printProperty(Vertex v, String property) {  
 System.*out*.println(v.property(  
 "personid") + " " + v.property(property).value());  
 }  
  
 public static void main(String[] args) {  
 new Module7();  
 }  
}

}