Geir Matti Järvelä (geirmj11@ru.is)
Þorgeir Auðunn Karlsson (thorgeirk11@ru.is)

# Programming Assignment 4: Properties of Social Networks

For the assignment we created three classes, explained further below. All of the classes implement an intelligent lazy solution to each problem. In effect this means that all calculations are done at most once, and only when they are needed. Each method checks first if a solution exists, and if so simply returns a previously calculated solution. This method costs a little bit more in memory, since previous solutions need to be kept, but makes up for it with constant calculating times when duplicating previous calculations, or when two functions use the same results.

## Part I – The Centrality class

public Centrality(Graph G)
public int degree(int v)
public int ecc(int v)
public int effEcc(int v)
public double closeness(int v)
public int popularVertex()
public int center()
public int effCenter()
public int closest()

### Efficiency of methods

The numbers in the table on the right show the complexity of each method, for the first time each method is called. However some of the methods call other methods for every point, for example the closest() method calls the closeness() function for every point in the graph, and since the results are stored, they can be accessed directly thereafter.

| Method | Initial Complexity | Subsequent |
|---|---|---|
| Constructor | 10 | constant |
| Degree | $\sim V$ | constant |
| Eccentricity | $\sim 2V + E$ | constant |
| Effective Eccentricity | $\sim V(\log V)$ | constant |
| Closeness | $\sim 2V + E$ | constant |
| Most Popular Vertex | $\sim V^2$ | constant |
| Center | $\sim V$ | constant |
| Effective Center | $\sim V(V(\log V))$ | constant |
| Closest Vertex | $\sim V(V + E)$ | constant |

Complexity table: $V$: Number of Vertices
$E$: Number of Edges.

## Part II – The SymbolCentrality class

The SymbolCentrality class is simply an extention of Centrality. It creates a Symbolgraph class instance, and uses it to create indexed integer keys for each actor.
This allows the class to use the methods of the Centrality class to calculate all necessary values.

### Efficiency

Since each function calls the sg.index() method, this adds log(n) complexity to each operation of the Centrality class. In addition more memory is needed because of the SymbolGraph that stores the indices.

## Part III – The ExtendedBreadthFirstPaths class

### Efficiency