**Implementing, testing and experimental analysis of centrality concepts**

Unzip the folder team-5.zip. You will see the following files:

* driverFunctionSpring2020CSE6331.py
* formatInputFiles.py
* Analysis.py
* Heuristics folder

Capabilities:

* DriverFunctionSpring2020CSE6331.py: This is a driver function that takes the configuration(contains layer name, list of nodes and edges in a layer) and analysis(contains expressions state the centrality measures to run on different layers) text file as inputs and produces 2 output files listing nodes and hubs depending upon the layers and expressions in the input file.
* FormatInputFiles.py: takes a text file containing list of edges and gives a list of nodes and edges in appropriate format for driver function input
* Analysis.py: takes as input list of layers(list of nodes and edges in the format same as the one used for driver function)
* **To use Driver function**

The driver function takes as arguments 2 files, the config file and the analysis file. Lines starting with ‘#’ are ignored. The formats of which are:

i) Parameter 1: Configuration file format:

HeMLN = <name-of-the-layer\_1>,<name-of-the-layer\_2>,…,<name-of-the-layer\_n>

BASE = <path-to-base-directory>

<Layer\_Name>= <path-to-layer-in-the-base-directory>

**Example:**

HeMLN = F

BASE =C:\Datasets\

F = Layers\centralityTestData\Facebook.txt

**Note**: format of ‘Layers\centralityTestData\Facebook.txt’ in will be

<name of the layer>

<number of nodes>

<number of edges>

<list of nodes >

<list of edges> -> (node1,node2,weight)

**Example:**

Facebook

2

1

Id1

Id2

Id3

Id1,Id2,1.0

ii) Parameter 2: Analysis file:

OUTPUT\_DIR = <path-to-the-output-directory>

SingleLayerHub,<centrality\_measure>,<name-of-the-layer>

**Example:**

OUTPUT\_DIR = C:\outputs\Analysis\Facebook\_output

SingleLayerHub,Degree,F

OUTPUT\_DIR = C:\outputs\Analysis\Facebook\_output

SingleLayerHub,Close,F

In this case, 2 output files will be created in the output directory(‘C:\outputs\Analysis\Facebook\_output’) for each expression in the analysis file.

Output files created in this case are:

Expression1F\_Degree.txt

Expression1F\_Degree\_hubs.txt

Expression1F\_Close.txt

Expression1F\_Close\_hubs.txt

**3. Output format**

a. Type 1-All nodes: name of the file:

expression<#><layer\_name>\_<centrality\_measure>

i. Name of the layer

ii. Time taken to calculate centrality values for all nodes

iii. Node id and centrality value

**Example:**

Facebook

Id1 0.5

Id2 0.5

Id3 0

b. Type 2-Hubs: name of the file:

File name format: expression<#><layer\_name>\_<centrality\_measure>\_hubs

Each line of the outputs is described below.

i. Name of the layer

ii. Line 2,3,4 describe the fraction of the total number of nodes that are

hubs

iii. Line 5 and 6 describe the average value, min value, max value, sum for

the given centrality measure for all the nodes

iv. Line 8 and onwards describes hubs characterized by its node id and

centrality value.

**Example:**

Facebook

X= Number of nodes

Y = Total number of nodes

x/y = 2/3

Average Value, Min Value, Max Value, Sum

0.5, 0, 0.5, 1

Id1 0.5

Id2 0.5

To run the driver function: python <path\_toDriverfile>/DriverFunctionSpring2020CSE6331.py configuration.txt analysis.txt

* **To use format an input file (use** formatInputFiles.py**)**
  1. Set the path in the code
  2. Run python <path\_to\_file>/formatInputFiles.py <path\_to\_file\_you\_want\_to\_format> <name\_of\_the\_layer> <separator>
  3. Separator can be set to either space or comma, depending upon how nodes are separated in the list of edges file.

**Example:**

If the edges in the input file are represented as:

Id1 Id2 4

Then set separator as space,

python C:\formatInputFiles.py C:\facebook\_edges.txt Facebook space

If the edges in the input file are represented as:

Id1,Id2,4

Then set separator as comma,

python C:\formatInputFiles.py C:\facebook\_edges.txt Facebook comma

* **To use Analysis file**

1. Change the output directory path in the code
2. To run the code python <path\_to\_analysis\_file>\Analysis.py <list of files containing layers> <output file name>

**Example:**

Python C:\Analysis.py Gplus.txt Delta.txt output

4 Output files will be generated:

output\_between.txt, output\_eigen.txt, output\_degree.txt, output\_close.txt

Example outputs:

**Eigen file**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Layer | No. of nodes | avg\_eigen | degree assortivity coefficient | density | clustering coefficient | number of hubs | ratio of hubs |
| Gplus | 344 | 0.029946 | 0.13941463 | 0.056986914 | 0.546231376 | 105 | 0.305 |
| Delta | 213 | 0.045537 | -0.664868907 | 0.030206396 | 0.617613732 | 71 | 0.333 |

**Closeness file**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Layer | nodes | avg\_closeness | graph\_avg\_distance | clustering coefficient | density | graph diameter |
| Delta | 213 | 0.45645 | 2.21893 | 0.617614 | 0.030206 | 3 |
| Gplus | 344 | 0.351554 | 2.877587 | 0.546231 | 0.056987 | 6 |

* **To use Heuristic Code**

1. Make sure the following packages are installed:
   1. sys
   2. statistics
   3. time
   4. networkx
2. cd Heuristics
3. The description of the files are as follows:
   1. American - sample input file.
   2. Delta - sample input file.
   3. American\_AND\_Delta - sample AND composition.
   4. betweeness\_heuristic.py - Heuristics code.
   5. centrailty\_measures.py - code to get node betweenness values.
   6. centrailty\_measuresv3.py - code to get edge betweenness values.
   7. CtoPython.py - file to run C++ code in Python.
   8. layerComposerConfiguration – configuration file for AND composition.
   9. layerComposerv2.cpp - C++ file for AND composition code.
   10. Results comparison – file comparing the various heuristics.
4. The file for layer 1 is the file containing the nodes and edges of the graph in the format as described below:
   1. Name
   2. Number of nodes
   3. Number of edges
   4. Nodes in the graph with each node on one line
   5. Edges of the graph in the format <node,node,weight> with each edge on a single line.
5. The file for layer 2 has the same format.
6. Make sure all the files are in the same folder.
7. The code for generating AND composition was taken from Abhishek Santra and used as it as.
8. The layerComposerConfiguration file should be edited on lines 5 and 6 with the names of the layers to find the AND composition of.
9. Call the file betweeness\_heuristic.py with the command line arguments <name of layer 1> <name of layer 2>. For e.g. python betweeness\_heuristic.py American Delta. Make sure to do step 8 before step 9.