Project 1 Report

Student Name: Jaskeerat Singh
Student ID: 47610039
Student Email: Jaskeerat.singh@uqconnect.edu.au

This report describes the program implemented to calculate the Betweenness centrality and the PageRank centrality of an undirected and an unweighted graph created by reading a text file containing a list of edges between the nodes of Facebook data. This undirected graph at the end consists of 88234 edges which connect 4039 nodes. The final graph created is a connected graph with only one major component.

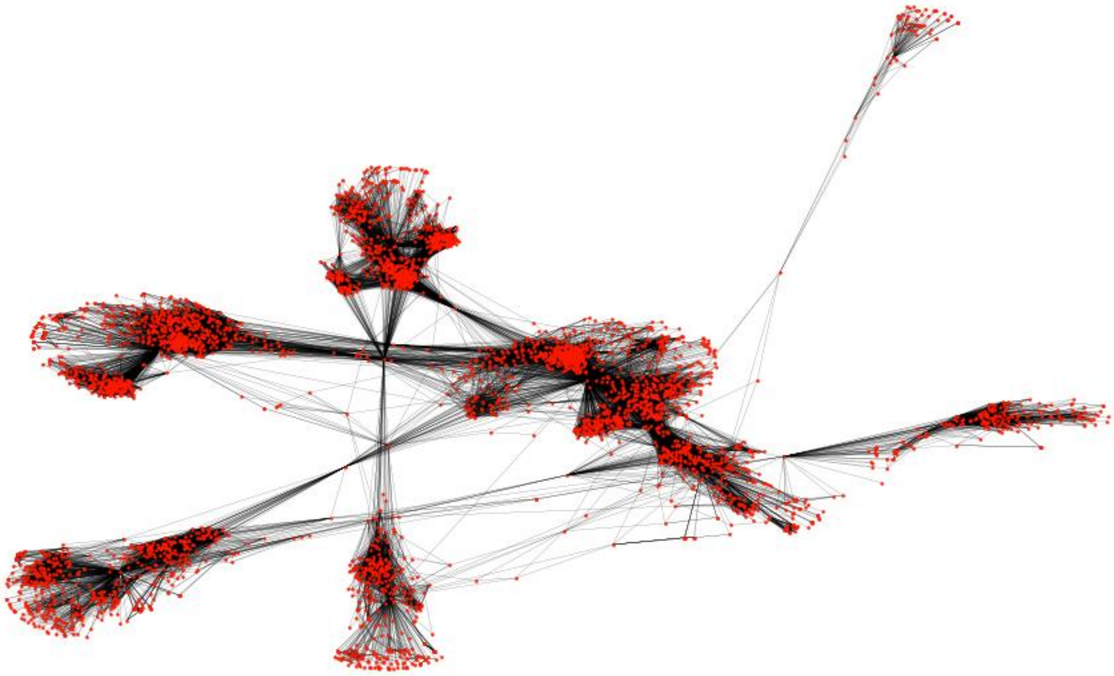


Figure 1. Undirected Graph consisting of 4039 nodes with 88234 edges

Task 1 - Betweenness Centrality

Betweenness centrality is a centrality measure used to evaluate the importance of nodes in a graph. The algorithm works on the principle that a node is crucial if it lies between the shortest path between any other two nodes. This betweenness centrality of a particular node depends on the number of times a node will come in the shortest paths between any other pair of nodes. To get the shortest paths in an unweighted graph, we have employed Breadth-First Search (BFS) algorithm.

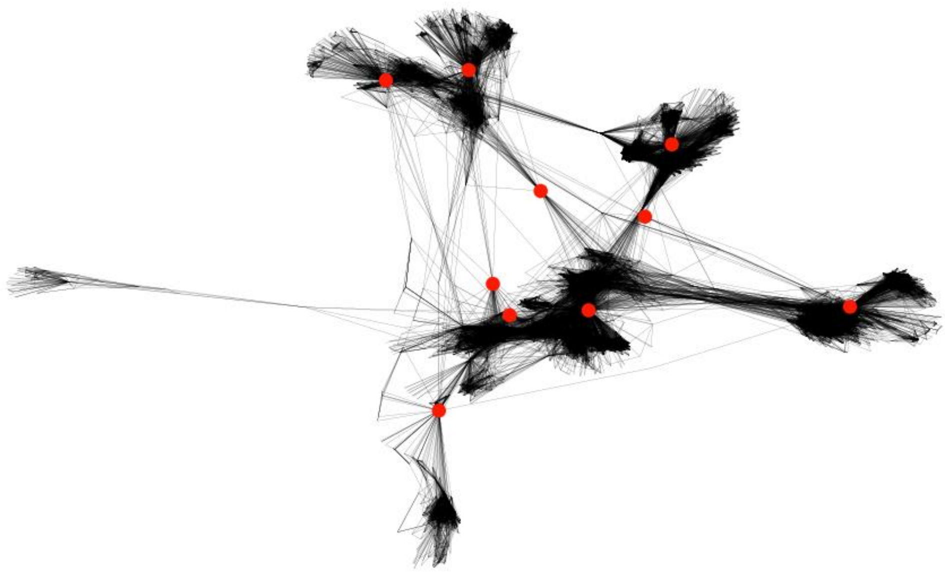
$$\boxed{C_b(v_i) = \sum_{s \neq t \neq v_i} \frac{\sigma_{st}(v_i)}{\sigma_{st}}}$$

The betweenness centrality of a node is basically a summation of the ratio of the shortest path passes through that node. Here, the numerator signifies the number of shortest paths from node s to node t that pass from node v. The denominator signifies the total number of shortest paths that exist between node s and node t.

Algorithm Followed –

- Creating a Queue for maintaining the order in which nodes will be visited
- Creating a Set for maintaining the sequence of visiting nodes
- While(queue is not empty):
    Read the nodes from queue in a FIFO order
    Add the node to the visited set
    Add the unvisited neighbors of the node to the queue if not in the queue

Results –



RANK	1	2	3	4	5	6	7	8	9	10
NODE	107	1684	3437	1912	1085	0	698	567	58	428
CENTR ALITY	78331 20.3	55065 73.4	38490 12.3	37378 36.4	24291 55.5	23849 92.2	188004 8.49	156999 3.81	137518 9.97	104832 8.14

Task 2 – Pagerank Centrality

Pagerank is a centrality measure used to evaluate nodes' importance in a graph. The algorithm works on the simple principle that a node is more important if it has more incoming links from other nodes. The centrality of all the nodes in the graph is calculated using the power iteration method.

To implement the PageRank algorithm, we start by referencing the algorithm formula followed in tutorials and lectures. Using the values of alpha = 0.85 and beta = 0.15 (or 1-alpha) defined in the task, we will be substituting the value in the given formula:

$$\boxed{\mathbf{C}_p = \beta(\mathbf{I} - \alpha A^T D^{-1})^{-1} \cdot \mathbf{1}}$$

After the undirected graph has been created, we will first calculate the values of the adjacency matrix, transpose of adjacency matrix, degree matrix and inverse of degree matrix. Here, I will be the identity matrix, and 1 will be the ones array of size equal to the total number of independent nodes/vertices in our undirected graph.

Algorithm Followed –

- Set the value of size = n, pre_value = 0, current_ value = 1/n
- While(|pre_value - current_ value| > condition):
  pre_value = current_ value
  current_ value = **beta(I–alpha*transpose_adjacency*inverse_diagnol)^-1*1**
  normalize current_value
- Return last normalized current_value

alpha and beta are the params where beta equals 1 – alpha. Hence, alpha = 0.85 & beta = 0.15

Results –

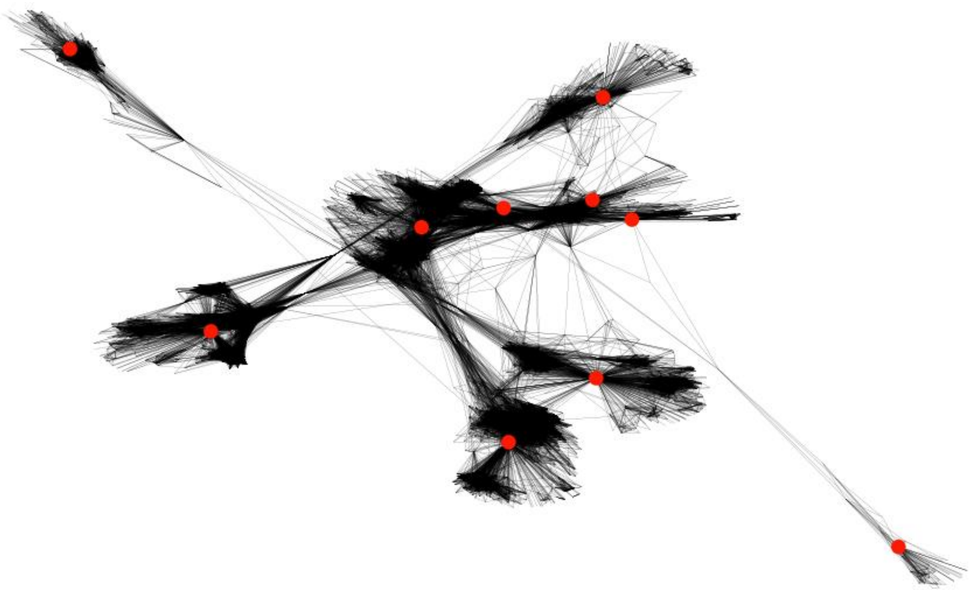


Figure 3. The top 10 nodes ranked according to their PageRank centrality values.

RANK	1	2	3	4	5	6	7	8	9	10
NODE	3437	107	1684	0	1912	348	686	3980	414	483
CENTRALITY	0.0075746	0.0068884	0.0063085	0.0062247	0.0038166	0.0023174	0.0022168	0.0021566	0.00178229	0.00129417

Summary

Through this assignment, we have implemented code and algorithm to process an undirected and unweighted graph. In this graph, we have applied PageRank and node Betweenness centrality measures to produce a text file that presents two lists of the top 10 nodes from each category. We have used the BFS shortest algorithm to calculate the shortest paths between nodes to implement betweenness centrality. We have used pandas, numpy and network modules of Python to implement this whole project.

Reference

[1] Database, N. (2022, September 14). *How to compute the betweenness centrality against open source graph database NebulaGraph*. Medium. https://itnext.io/how-to-compute-the-betweenness-centrality-against-open-source-graph-database-nebulagraph-f922d196dce

[2] Drawing basics. (n.d.). Retrieved from https://networkx.guide/visualization/basics/

[3] Sardar, A. (2020, July 1). Custom PageRank implementation in Python and verification in MS Excel. Retrieved from https://medium.com/@arpanspeaks/custom-pagerank-implementation-in-python-and-verification-in-ms-excel-9ab6c690aaf5

[4] Qu, L. (n.d.). Tutorial Week 4. School of ITEE, The University of Queensland. https://learn.uq.edu.au/