Nick Petty

Z23296080

**Homework 1**

**Question 1 (2.0 points: 0.25/each):** Please use your own language to briefly explain the following concepts:

Social networks: a social structure composed of individuals or groups connected by social interactions or shared traits.

Undirected graph: a graph (vertices connected by edges) whose edges have no orientation. The edge (u, v) is equivalent to the edge (v, u).

Adjacency matrix: a square grid whose axes are all nodes in a graph, and the value of any cell (a, b) is the weight of the edge between nodes a and b, or 0 if no edge exists.

Network Diameter: for all pairs of nodes in a graph, each pair has a minimum distance between them. The network diameter is the maximum of all these minimum node pair distances. It describes how far a network reaches and how long it takes to cross the network.

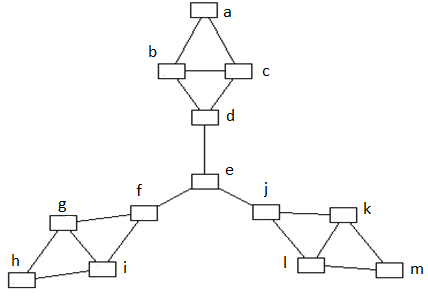
Centrality score of a node: the ratio of a node’s degree to the number of other nodes in a graph. It describes how important a node is to the network.

Random Walk: a sequence of incident edges in a graph, but no selection rule is followed when a choice of edges is presented. A path through a graph that choses each next step randomly.

Random Graph: a set of vertices with edges added at random. These usually have a Poisson degree distribution – a few nodes of very high and low degree, most nodes of similar degree.

Power-Law Distribution: the degree distribution often seen in social networks – a large number of low-degree nodes, with frequency dropping exponentially as degree increases linearly. This majority being in the low-degree group is also visualized as the long tail in a graph of a power-law distribution.

**Question 2 (1.5 points):** In the following network, please calculate the Betweenness Centrality scores [0.5 pt], Closeness Centrality score [0.5 pt], and Eigen Vector based centrality scores [0.5 pt] for every nodes in the network (please show your solutions)

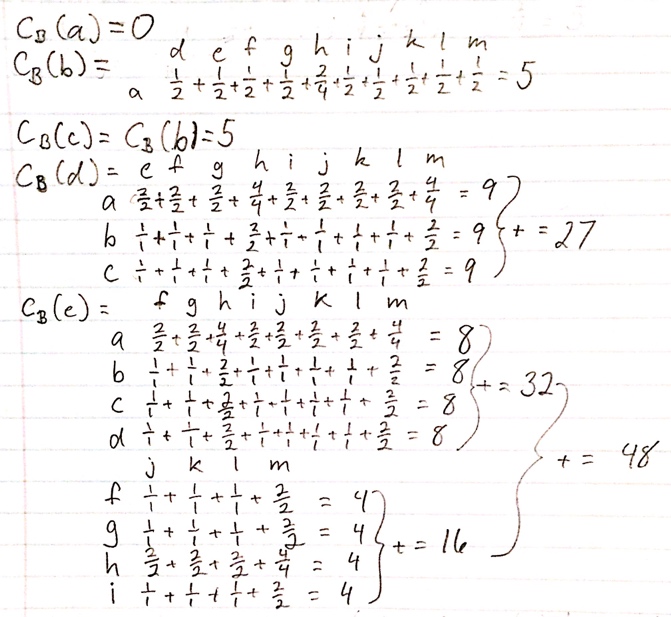


The graph is symmetrical around node e, so only scores for nodes a, b, c, d, and e need to be calculated. This means that a = h =m, d = f = j, and b = c = g = i = k = l. These results were verified with Gephi.

Betweenness Centrality

|  |  |  |
| --- | --- | --- |
|  | Score | Normalized |
| a | 0 | 0.000 |
| b | 5 | 0.076 |
| c | 5 | 0.076 |
| d | 27 | 0.409 |
| e | 48 | 0.727 |
| f | 27 | 0.409 |
| g | 5 | 0.076 |
| h | 0 | 0.000 |
| i | 5 | 0.076 |
| j | 27 | 0.409 |
| k | 5 | 0.076 |
| l | 5 | 0.076 |
| m | 0 | 0.000 |

For each node, this the sum, across all other node pairs, of ratios of shortest paths through the node to total shortest paths. Calculations for nodes a, b, c, d, and e are shown. The normalization factor is 2/(n-1)(n-2) = 1/66.



Closeness Centrality

The score is the average of all shortest paths from a node to all other nodes, that is, the sum of shortest paths divided by the total number of nodes minus one (Sum/12). This is normalized by taking the inverse of the score.

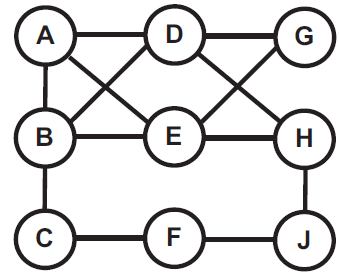
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | d | e | f | g | h | I | j | k | l | m |
| a | 0 | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 5 | 4 | 5 | 5 | 6 |
| b | 1 | 0 | 1 | 1 | 2 | 3 | 4 | 5 | 4 | 3 | 4 | 4 | 5 |
| c | 1 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 4 | 3 | 4 | 4 | 5 |
| d | 2 | 1 | 1 | 0 | 1 | 2 | 3 | 4 | 3 | 2 | 3 | 3 | 4 |
| e | 3 | 2 | 2 | 1 | 0 | 1 | 2 | 3 | 2 | 1 | 2 | 2 | 3 |
| f | 4 | 3 | 3 | 2 | 1 | 0 | 1 | 2 | 1 | 2 | 3 | 3 | 4 |
| g | 5 | 4 | 4 | 3 | 2 | 1 | 0 | 1 | 1 | 3 | 4 | 4 | 5 |
| h | 6 | 5 | 5 | 4 | 3 | 2 | 1 | 0 | 1 | 4 | 5 | 5 | 6 |
| i | 5 | 4 | 4 | 3 | 2 | 1 | 1 | 1 | 0 | 3 | 4 | 4 | 5 |
| j | 4 | 3 | 3 | 2 | 1 | 2 | 3 | 4 | 3 | 0 | 1 | 1 | 2 |
| k | 5 | 4 | 4 | 3 | 2 | 3 | 4 | 5 | 4 | 1 | 0 | 1 | 1 |
| l | 5 | 4 | 4 | 3 | 2 | 3 | 4 | 5 | 4 | 1 | 1 | 0 | 1 |
| m | 6 | 5 | 5 | 4 | 3 | 4 | 5 | 6 | 5 | 2 | 1 | 1 | 0 |
| **Sum** | 47 | 37 | 37 | 29 | 24 | 29 | 37 | 47 | 37 | 29 | 37 | 37 | 47 |
| **Score** | 3.917 | 3.083 | 3.083 | 2.417 | 2.000 | 2.417 | 3.083 | 3.917 | 3.083 | 2.417 | 3.083 | 3.083 | 3.917 |
| **Normalized** | 0.255 | 0.324 | 0.324 | 0.414 | 0.500 | 0.414 | 0.324 | 0.255 | 0.324 | 0.414 | 0.324 | 0.324 | 0.255 |

Eigenvector Centrality

**This was calculated with the Online Matrix Calculator from bluebit.gr, using the adjacency matrix below. Calculation was also done in Gephi with 10,000 iterations.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **k** | **l** | **m** |  | **Eigenvalue** |  | **Eigenvector** | **Gephi** |
| **a** | **0** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |  | **2.732** | **a** | 0.198 | 0.592 |
| **b** | **1** | **0** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |  | **b** | 0.280 | 0.838 |
| **c** | **1** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |  | **c** | 0.280 | 0.838 |
| **d** | **0** | **1** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** |  | **d** | 0.316 | 0.945 |
| **e** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **0** | **0** | **0** |  | **e** | 0.335 | 1.000 |
| **f** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |  | **f** | 0.316 | 0.945 |
| **g** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **1** | **0** | **0** | **0** | **0** |  | **g** | 0.280 | 0.838 |
| **h** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** |  | **h** | 0.198 | 0.592 |
| **i** | **0** | **0** | **0** | **0** | **0** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** |  | **i** | 0.280 | 0.838 |
| **j** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **1** | **1** | **0** |  | **j** | 0.316 | 0.945 |
| **k** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **1** |  | **k** | 0.280 | 0.838 |
| **l** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **1** | **0** | **1** |  | **l** | 0.280 | 0.838 |
| **m** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **1** | **0** |  | **m** | 0.198 | 0.592 |

**Question 3 (2.5 pts):** In the following network, please explain how to use adjacency matrix and the power of adjacency matrix to find diameter of the network (show your solution 1 pt). Please draw degree distribution of the network [0.25 pt], calculate clustering coefficient for every node in the network [0.25 pt]. Please also calculate the edge density [0.25 pt] and the clustering coefficient of the whole network [0.25]. Please explain why clustering coefficient is smaller than the edge density [0.25 pt]. Please find the node with the highest betweenness score (please show your solution [0.25 pt])



Network diameter

Using the adjacency matrix of a network, the diameter is the smallest power of that matrix such that each cell has had a non-zero value at least once. When an adjacency matrix is raised to a power *p*, each cell *(i, j)* shows the number of paths of length *p* between the nodes *i* and *j*. Once all node pairs have had at least one path between them, the shortest path for each pair can be determined, and the largest shortest path is the network’s diameter. The diameter calculation is shown below, with *(F, G)* being the diameter. Results were verified with Gephi.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | A | B | C | D | E | F | G | H | J |  | **2** | A | B | C | D | E | F | G | H | J |
| A | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |  | A | 3 | 2 | 1 | 1 | 1 | 0 | 2 | 2 | 0 |
| B | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |  | B | 2 | 4 | 0 | 1 | 1 | 1 | 2 | 2 | 0 |
| C | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |  | C | 1 | 0 | 2 | 1 | 1 | 0 | 0 | 0 | 1 |
| D | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |  | D | 1 | 1 | 1 | 4 | 4 | 0 | 0 | 0 | 1 |
| E | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |  | E | 1 | 1 | 1 | 4 | 4 | 0 | 0 | 0 | 2 |
| F | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |  | F | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 0 |
| G | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |  | G | 2 | 2 | 0 | 0 | 0 | 0 | 2 | 2 | 0 |
| H | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |  | H | 2 | 2 | 0 | 0 | 0 | 1 | 2 | 3 | 0 |
| J | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |  | J | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 2 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **3** | A | B | C | D | E | F | G | H | J |  | **4** | A | B | C | D | E | F | G | H | J |
| A | 4 | 6 | 2 | 9 | 9 | 1 | 2 | 2 | 2 |  | A | 24 | 24 | 7 | 14 | 14 | 4 | 18 | 20 | 3 |
| B | 6 | 4 | 5 | 10 | 10 | 0 | 2 | 2 | 3 |  | B | 24 | 31 | 4 | 14 | 14 | 8 | 20 | 23 | 2 |
| C | 2 | 5 | 0 | 1 | 1 | 3 | 2 | 3 | 0 |  | C | 7 | 4 | 8 | 12 | 12 | 0 | 2 | 2 | 6 |
| D | 9 | 10 | 1 | 2 | 2 | 2 | 8 | 9 | 0 |  | D | 14 | 14 | 12 | 36 | 36 | 1 | 4 | 4 | 11 |
| E | 9 | 10 | 1 | 2 | 2 | 2 | 8 | 9 | 0 |  | E | 14 | 14 | 12 | 36 | 36 | 1 | 4 | 4 | 11 |
| F | 1 | 0 | 3 | 2 | 2 | 0 | 0 | 0 | 3 |  | F | 4 | 8 | 0 | 1 | 1 | 6 | 4 | 7 | 0 |
| G | 2 | 2 | 2 | 8 | 8 | 0 | 0 | 0 | 2 |  | G | 18 | 20 | 2 | 4 | 4 | 4 | 16 | 18 | 0 |
| H | 2 | 2 | 3 | 9 | 9 | 0 | 0 | 0 | 4 |  | H | 20 | 23 | 2 | 4 | 4 | 7 | 18 | 22 | 0 |
| J | 2 | 3 | 0 | 0 | 0 | 3 | 2 | 4 | 0 |  | J | 3 | 2 | 6 | 11 | 11 | 0 | 0 | 0 | 7 |

Degree distribution

The degrees of each node are shown along with a frequency distribution graph, verified with Gephi.

|  |  |
| --- | --- |
| Node | Degree |
| A | 3 |
| B | 4 |
| C | 2 |
| D | 4 |
| E | 4 |
| F | 2 |
| G | 2 |
| H | 3 |
| J | 2 |

Clustering coefficient per node

For each node, the number of neighbors is listed, along with how many edges these neighbors share and how many edges are needed to make the neighbors into a complete graph. The clustering coefficient is the ratio of existing edges to total edges required for the neighbors to be complete.

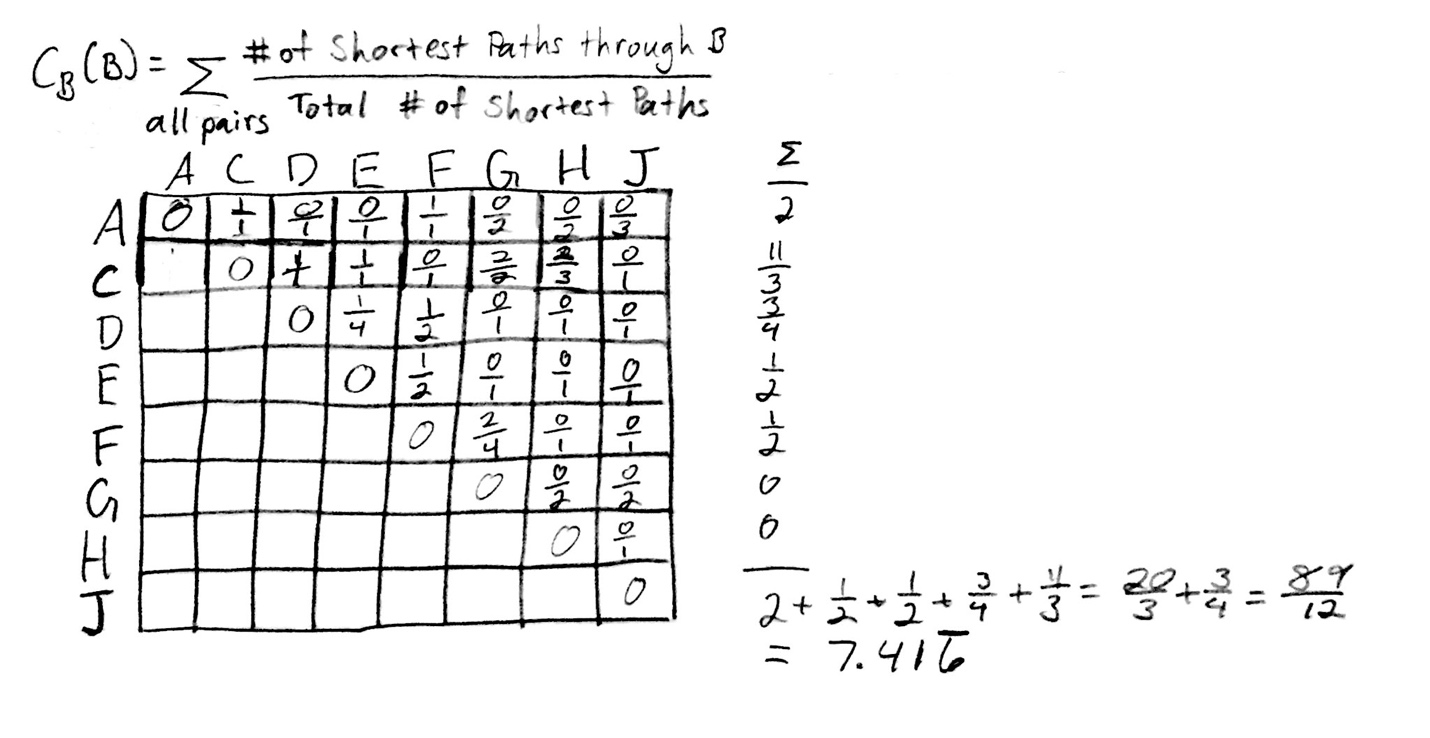
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Node | Neighbors | Edges | Complete | Clustering coefficient |
| A | 3 | 2 | 3 | 0.667 |
| B | 4 | 2 | 6 | 0.333 |
| C | 2 | 0 | 1 | 0.000 |
| D | 4 | 1 | 6 | 0.167 |
| E | 4 | 1 | 6 | 0.167 |
| F | 2 | 0 | 1 | 0.000 |
| G | 2 | 0 | 1 | 0.000 |
| H | 2 | 0 | 1 | 0.000 |
| J | 2 | 0 | 1 | 0.000 |

Network edge density and clustering coefficient

The edge density is 2|E| / |V|(|V| - 1) = 2\*13 / 9\*8 = 13/36 = 0.361. The clustering coefficient for the network is the average of clustering coefficients, that is (1/n) ∑Ci = (4/3)/9 = 0.148.

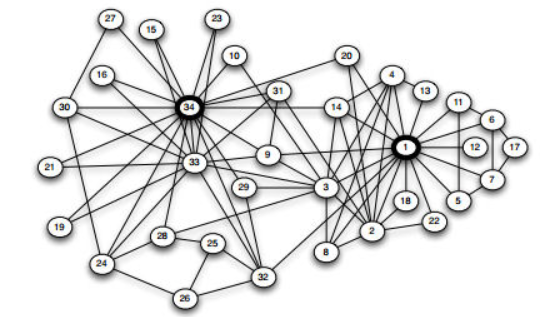
In this graph, the clustering coefficient is smaller than the edge density because node subgroups are very rarely complete compared to the completeness of the entire graph.

Highest betweenness score

Node B has the highest betweenness score at 7.417. This was found through Gephi and verified manually below

**Question 4 (2.0 pts)** The following network shows a small benchmark “[Zachary's karate club](http://www-personal.umich.edu/~mejn/netdata/karate.zip)” social network which contains “friendships between 34 members of a karate club at a US university in the 1970s”.

1. Please calculate the edge density and the clustering coefficient of the whole network, and analyze why clustering coefficient is larger (or smaller) than the edge density [0.5 pt].
2. Please calculate the average distance between any two pairs of nodes [0.5 pt], and report the Diameter of the network [0.5 pt].
3. Please find the node(s) with the highest closeness centrality score [0.5 pt]



Zachary’s karate club network

**Question 5 (2.0 points)** The following URL points to a “coauthorship network” of scientists working on network theory and experiment.

<http://networkdata.ics.uci.edu/data.php?id=11>

A brief description of the network is given in the “netscience.txt”. In “netscience.paj” file (these are text files), you can find nodes and edges between nodes. The names of the scientists (which correspond to the nodes of the networks) are given in “netscience.gml”. Please download the dataset and use any program tools you are familiar with to build a network and finish the following tasks:

1. Please report and draw the degree distributions of the whole network [0.25 pt]. Convert the figure to log-log space and validate whether it complies with the power-law distributions [0.25 pt].
2. Please report cumulative degree distribution of the network [0.25 pt], and convert it to log-log space and validate whether it complies with the power-law distributions [0.25 pt].
3. Please report Rank-Degree distribution of the network [0.25 pt], and convert it to log-log space and validate whether it complies with the power-law distributions [0.25 pt].
4. Please report clustering coefficient and diameter of the network [0.5 pt].

**Tips for programming:**

In order to calculate the average distance between a pair of nodes, and calculate the diameter of the network, you will need to implement algorithms which calculate shortest path between any two nodes. Example of algorithms include

Dijkstra’s algorithm (<http://en.wikipedia.org/wiki/Dijkstra's_algorithm>)

Breath First Search algorithm (<http://en.wikipedia.org/wiki/Breadth-first_search>)

Alternatively, you may consider following programming tools/packages which are specifically designed for network and graph data. These packages have algorithms for finding shortest path and network diameter.

**Gephi: The open Graph Viz Platform**

[**https://gephi.org/**](https://gephi.org/)

**(Please note Gephi also has API functions to support user programming. You can check API functions for the following URL:**

[**https://gephi.org/docs/api/**](https://gephi.org/docs/api/)

**Java Platform: JUNG (Java Universal Network/Graph Framework)**

<http://jung.sourceforge.net/>

<http://www.datalab.uci.edu/papers/JUNG_tech_report.html>

**Python: NetworkX (High-productivity software for complex networks)**

<http://networkx.github.com/>

**.Net: NodeXL (Open source template for Microsoft tools)**

<http://nodexl.codeplex.com/>