Network Analysis and Modelling - CSCI 5352
Problem Set 1

Santhanakrishnan Ramani

Tuesday 6th September, 2016

Problem 1

(a) The adjacency matrix of network (A) is

A	1	2	3	4	5
1	0	1	0	0	1
2	0	0	1	0	0
3	1	0	0	0	0
4	1	0	0	0	1
5	0	0	1	1	0

(b) The adjacency list of network (A) is

A
1 → {2,5}
2 → {3}
3 → {1}
4 → {1,5}
5 → {3,4}

(c) Adjacency matrices of One Mode Projections of Network (B) are

A	1	2	3	4	5
1	0	1	1	1	1
2	1	0	0	1	1
3	1	0	0	0	0
4	1	1	0	0	0
5	1	1	0	0	0

A	1	2	3	4	5	6
1	0	1	1	0	1	0
2	1	0	1	0	0	1
3	1	1	0	1	1	0
4	0	0	1	0	0	0
5	1	0	1	0	0	0
6	0	1	0	0	0	0

Problem 2

Given, **A** the adjacency matrix of a simple graph and **1** the column vector whose elements are all 1

(a) The vector **K** whose elements are the degrees k_i of the vertices is given by

$$K = A.1$$

As the row sum of the adjacency matrix gives the degree of the node corresponding to that row.

(b) The number m of edges in the network is,

$$m = \frac{1}{2}(A.1)1^T$$

Since it's a simple graph, the number of edges is half the sum of degrees of all the nodes in the graph.

(c) The matrix **N** whose elements N_{ij} is equal to the number of common neighbour of vertices i and j is

$$N = AA^T$$

There will be a common neighbour for vertices i and j only if there is a vertex k such that $A_{ik} = A_{kj} = 1$

(d) Total number of triangles in the network are,

$$Traingle\_Count = \frac{1}{6} * Tr((AA^T)A)$$

Since, AA^T gives the number of common neighbours of vertices i and j, triangles could be formed if there is an edge between i and j.
As it's a simple graph, each edge is considered 2 times, and each of the 3 vertices forming the triangle adds up individually, the triangle count is found out by taking the trace of common neighbours and adjacency matrix and dividing it by 6 (2*3).

Problem 3

Given a bipartite graph with n_1 vertices of type 1 and n_2 vertices of type 2, prove that the mean degrees c_1 and c_2 of the two types are given by,

$$c_2 = \frac{n_1}{n_2} c_1$$

Proof:

As the graph is bipartite, there can be edges only between two nodes of different type, therefore

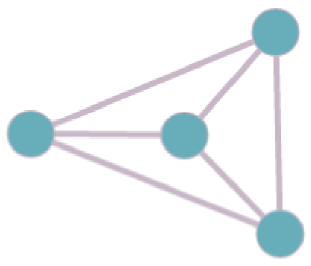
$$\text{Mean degree } c_1 \text{ of type } 1 = \frac{\text{No of edges in the graph}}{n_1} \tag{1}$$

$$\text{Mean degree } c_2 \text{ of type } 2 = \frac{\text{No of edges in the graph}}{n_2} \tag{2}$$

$$\text{From (1) and (2), we can see that, } c_2 = \frac{n_1}{n_2} c_1$$

Problem 4

(a) K-core network is obtained by repetitively removing nodes whose degree are less than K. In the end we are left with the subset of graph whose vertices have a minimum degree of K. 3-core in network (A) is,



(b) Reciprocity (r) of a network is given by the formula, $r = \frac{1}{m} Tr(A^2)$ where A is the adjacency matrix.

The Adjacency matrix of network(B) is,

A	1	2	3	4
1	0	1	1	0
2	1	0	0	1
3	1	0	0	1
4	1	1	0	0

The Reciprocity of network(B) is, $r = \frac{1}{8} Tr(A^2) = \frac{1}{8} * 6 = \frac{3}{4}$

(c) The cosine similarity of i and j is the number of common neighbors the two vertices have divided by the geometric mean of their degrees. The cosine similarity between vertices A and B in network(C) is,

$$\sigma_{AB} = \frac{n_{AB}}{\sqrt{k_A k_B}} = \frac{2}{\sqrt{4*5}} = \frac{1}{\sqrt{5}}$$

Problem 5

- To Prove in a Cayley tree number of vertices reachable in d steps from the central vertex is $k(k-1)^{d-1}$ for $d \geq 1$.

  Since each vertex except leaves is connected to k other vertices, in the first step k vertices can be reached from the central vertex. Now as these k vertices will be connected to k-1 new vertices and this continues till we reach the level of leaves. So, the number of vertices that can be reached in d steps is $k(k-1)^{d-1}$.

- Diameter(D) of a network is the longest geodesic path that exist between any pair of vertex. Diameter in terms of k and the number of vertices n is given below,

  The total number of vertices in a network is the summation of one (the central vertex) plus the number of nodes that can be reached each step until we reach the leaves. The total number of steps will be half the diameter since the tree is symmetric, which can be mathematically return as,

$$n = 1 + \sum_{d=1}^{D/2} k(k-1)^{d-1}$$

  On expanding the geometric series, $\sum_{d=1}^{D/2} k(k-1)^{d-1}$ where $a = k$, and $r = k - 1$,

$$n = 1 + k\frac{((k-1)^{D/2} - 1)}{k-1-1}$$

$$\frac{(n-1)(k-2)}{k} + 1 = (k-1)^{D/2}$$

$$\frac{n(k-2)+2}{k} = (k-1)^{D/2}$$

  Taking log on both sides,

$$D = 2 * [\log_{k-1}(\frac{n(k-2)+2}{k})]$$

- From the above relation for Diameter of the network, we can conclude that the given network displays the "small-world effect", as the diameter(D) increases as O($\log n$) or slower.

Problem 6

(a) Derive an expression for $< k_v >$ average degree of the neighbour in terms of the average squared-degree $< k^2 >$ and the average degree $< k >$, we know that

$$< k_v > = \frac{1}{2m}\sum_{u=1}^{n}\sum_{v=1}^{n} k_v A_{uv}$$

$$< k > = \frac{1}{n}\sum_{i=1}^{n} k_i = \frac{2m}{n} \tag{3}$$

$$< k^2 > = \frac{1}{n}\sum_{i=1}^{n} k_i^2 \tag{4}$$

Since adjacency matrix is symmetric as the given network is a simple graph, $< k_v >$ can be return as

$$< k_v > = \frac{1}{2m}\sum_{u=1}^{n}\sum_{v=1}^{n} k_v A_{vu}$$

Now interchanging the order of summation and bringing $< k_v >$ out of first summation,

$$< k_v > = \frac{1}{2m}\sum_{v=1}^{n} k_v \sum_{u=1}^{n} A_{vu}$$

Collaborated with Ruhi Saraf, Irene Beckman on Problem 5 & Problem 6. We discussed our ideas and proofs for these problems.

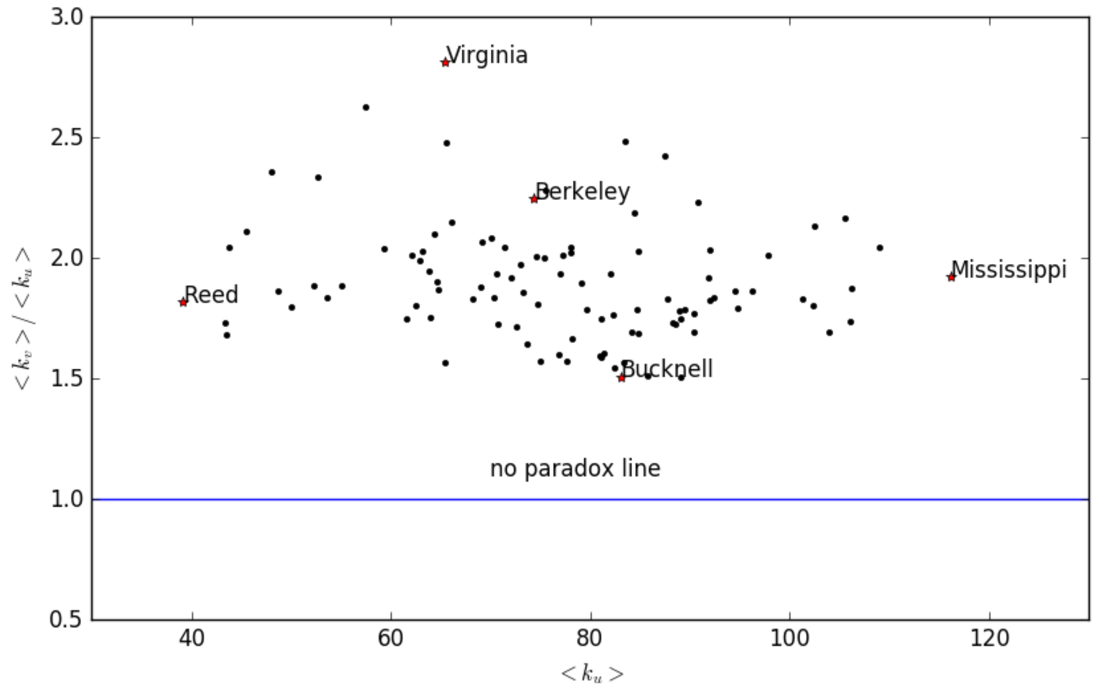
As summation of adjacency matrix row wise gives the degree of each vertex, $< k_v >$ can be return as,

$$< k_v >= \frac{1}{2m}\sum_{v=1}^{n} k_v k_v = \frac{1}{2m}\sum_{v=1}^{n} k_v^2$$

Substituting (3) and (4) in the above equation we get,

$$< k_v >= \frac{< k^2 >}{< k >}$$

(b) • The scatterplot below shows the ratio $\frac{< k_v >}{< k_u >}$ as a function of the mean degree $< k_u >$ of all 100 of the FB100 networks. The code is attached at the end.



From the above graph we could clearly see that, the friend paradox is observed in all 100 of the FB100 networks and there is no dependency between size of the paradox (the MND value) and the networks mean degree.

• The friendship paradox in the networks can be attributed the variance in the number of nodes each node is connected to. For eg., there might be a node which might be connected to a large number of nodes in the network, and completely shifts the average neighbour degree. The condition at which friendship paradox is not seen is when all the nodes in the network are connected to equal number of nodes. Mathematically,

$$< k^2 > \leq < k >< k >$$

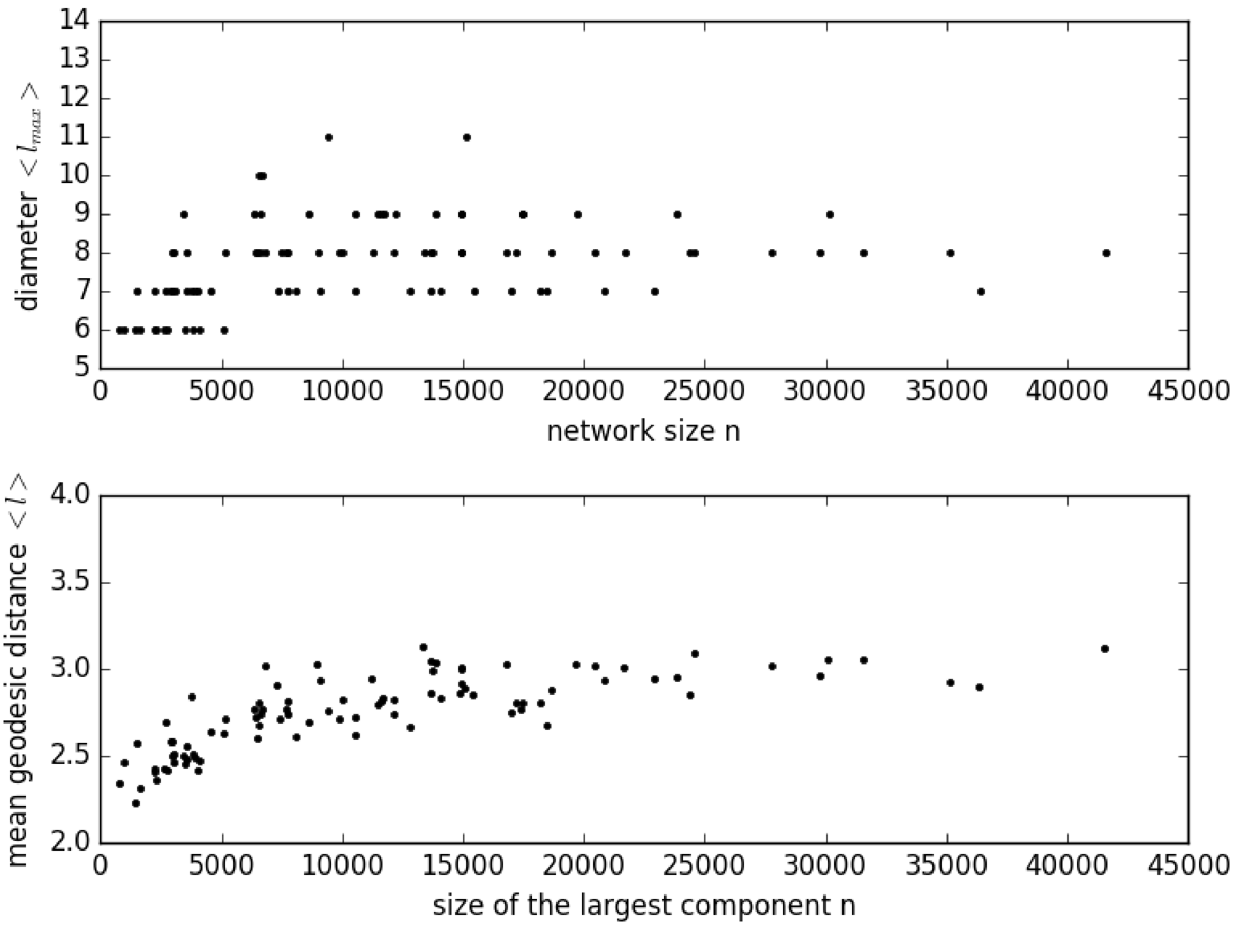
(c) The majority illusion is possible only if the vertexes having x=1 as the binary value of their vertex level property have higher mean degrees than the vertexes having x=0. In order to strengthen our intuition for the cause of majority illusion lets look at the below example.

Consider a bipartite graph, let the *type* 1 vertices have the property x=1, and the *type* 2 have the property x=0 such that numbers of vertices in *type* 1 is less than the ones in *type* 2. This satisfies the given fact that the fraction of vertices (q) that exhibit this property is less than 0.5. From problem 3, we know the relation between mean vertices of the two types c_1 & c_2 in a bipartite graph as

$$c_2 = \frac{n_1}{n_2}c_1$$

since the ratio of $\frac{n_1}{n_2} < 1$, the mean degree of c_1 must be greater than c_2 in order to satisfy the above relation, which implies that the nodes possessing this property should have higher degrees and therefore the majority of a nodes neighbors, on average, exhibiting that property will be greater than 0.5 (as the number of nodes in *type* 2 is greater than *type* 1). The same reasoning could be generalised to graphs that aren't bipartite.

(d) • The figure [1] below shows two plots, one showing l_{max} versus $network\, size\,(n)$ and another showing $mean\, geodesic\, length < l >$ versus the $size\, of\, largest\, component\,(n)$.



We could conclude form the l_{max} versus $network\, size\,(n)$ graph above that most of the 100 networks aren't well within the six degrees of separation, this could be attributed to the lower number of connections per person in 2005.

• As we know that the average degree of separation between two people is 6 in facebook at present, the diameter of facebook has decreased relative to the these values in 2005. The reason being the number of people in facebook has increased along with the number of connections per individual in recent years. As diameter D with respect to the network size (n) and average degree of a person (k) can be approximated as below,

$$D = \frac{\log n}{\log k}$$

increase in the value of (k) in recent years has kept the diameter D of the network in check and well within a limit, even though the network size has increased considerably.

[1] The graphs are plotted using only 92 of the total networks. As the code hasn't completed even after running it for full 4 days

Code - Problem 6B

Listing 1: To write data to a file

```
import re
import os
import networkx as nx

dataPath = "/home/santa/Dropbox/NAM/Problem_Set_1/Data/facebook100txt/"

file = open("/home/santa/Dropbox/NAM/Problem_Set_1/Code/data.txt", "w")
for filename in os.listdir(dataPath):
    if filename.endswith(".txt") and not filename.endswith("attr.txt") and
                                                   filename.find("readme") == -1:
        lines = [line.rstrip('\n') for line in open(dataPath+filename)]

        G = nx.Graph()
        for line in lines:
            vertexes = line.split("\t")
            G.add_edge(vertexes[0], vertexes[1])

        listDegree = []
        for node in G.nodes():
            listDegree.append(G.degree(node))

        average_degree = sum(listDegree) / (1.0 * G.number_of_nodes())
        average_squared_degree = sum([i ** 2 for i in listDegree]) /
                                                        (1.0 * G.number_of_nodes())
        mean_neighbour_degree = average_squared_degree / (1.0 * average_degree)
        name = filename[:re.search("\d", filename).start()]
        file.write(name + "," + str(average_degree) + "," +
                              str(mean_neighbour_degree/(1.0 * average_degree)) + "\n")
file.close()
```

Listing 2: To plot the data

```
import matplotlib.pyplot as plt

filename = "/home/santa/Dropbox/NAM/Problem_Set_1/Code/data.txt"
lines = [line.rstrip('\n') for line in open(filename)]
names = ["Reed","Bucknell", "Mississippi", "Virginia", "Berkeley"]

for line in lines:
    pts = line.split(",")
    if pts[0] in names:
        plt.annotate(pts[0], xy=(float(pts[1]),float(pts[2])))
        plt.plot(float(pts[1]),float(pts[2]),'*',color='red')
    else:
        plt.plot(float(pts[1]),float(pts[2]),'.',color='black')

plt.annotate("no_paradox_line",xy=(70,1.1))
plt.plot([30,130],[1,1])
plt.axis([30,130,0.5,3])
plt.xlabel(r'$<k_u>$')
plt.ylabel(r'$<k_v>/<k_u>$')
plt.show()
```

Code - Problem 6D

Listing 3: To write data to a file

```
import os
import re
import networkx as nx

dataPath = "/home/santa/Dropbox/NAM/Problem␣Set␣1/Data/facebook100txt/"
for filename in os.listdir(dataPath):
    if filename.endswith(".txt") and not filename.endswith("attr.txt") and
                                                  filename.find("readme") == -1:
        print filename
        lines = [line.rstrip('\n') for line in open(dataPath+filename)]
        G = nx.Graph()
        for line in lines:
            vertexes = line.split("\t")
            G.add_edge(vertexes[0], vertexes[1])

        graphs = sorted(nx.connected_component_subgraphs(G), key=len, reverse=True)
        largest_component = graphs[0]
        n = G.number_of_nodes()
        n_comp = largest_component.number_of_nodes()
        diameter = nx.diameter(largest_component)
        mean_length = nx.average_shortest_path_length(largest_component)

        name = filename[:re.search("\d", filename).start()]
        file = open("/home/santa/Dropbox/NAM/Problem␣Set␣1/Code/ec.txt", "a")
        file.write(name + "," + str(diameter) + "," + str(n) + "," + str(mean_length) +
                                                  "," + str(n_comp) + "\n")
        file.close()
```

Listing 4: To plot the data

```
import matplotlib.pyplot as plt

filename = "/home/santa/Dropbox/NAM/Problem␣Set␣1/Code/ec.txt"
lines = [line.rstrip('\n') for line in open(filename)]

plt.subplot(211)
for line in lines:
    pts = line.split(",")
    plt.plot(float(pts[2]),float(pts[1]),'.',color='black')

plt.axis([0,45000,5,14])
plt.ylabel(r'diameter␣$<l_{max}>$')
plt.xlabel('network␣size␣n')

plt.subplot(212)
for line in lines:
    pts = line.split(",")
    plt.plot(float(pts[4]),float(pts[3]),'.',color='black')

plt.axis([0,45000,2,4])
plt.ylabel(r'mean␣geodesic␣distance␣$<l>$')
plt.xlabel('size␣of␣the␣largest␣component␣n')

plt.tight_layout()
plt.show()
```