Network Analysis Project
Social Data Management

CHEN Hang

January 2019

The objective of this project is to analyze a network dataset. I choose the dataset of "EmailNetwork" which is the email communication network at the University Rovira i Virgili in Tarragona in the south of Catalonia in Spain. Nodes are users and each edge represents that at least one email was sent. The direction of emails or the number of emails are not stored. Why I choose it because this dataset has a graph with about 1000 nodes and 5000 edges, it's not a too small model and it's not so large that my computer could not do the analyse.

1 Minimal requirements

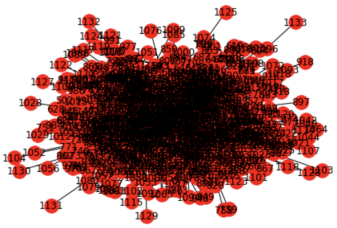
1.1 Show the number of nodes and edges in the graph

```
In [3]: len(G.nodes())
Out[3]: 1133

In [4]: len(G.edges())
Out[4]: 5451
```

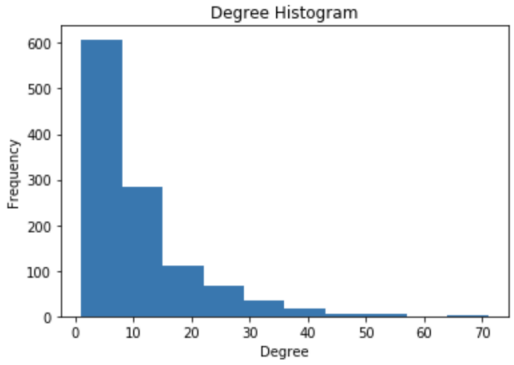
From the result of the figure, we can know that this graph has 1133 nodes and 5451 edges.

1.2 Draw the graph



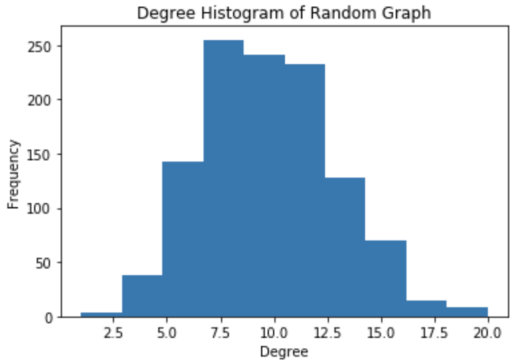
This graph shows its nodes in red and its edges in black. And we can notice that this graph is a connected graph.

1.3 Draw the histogram of degrees. Compare the distribution with the distribution for a random graph having the same average degree



We know that nodes are users and each edge represents that at least one email was sent. So the degree represents the number of other users that this user has communicated. From the figure, most of users only connect with less than 20 peoples by email. There are a few users whose connected users number is more than 20, and the most is up to 70. In my opinion, because this is the email of a university, so the former is maybe the normal student who has just communicated with his classmates and they constitute a large number of users in this system. And the latter is maybe some teachers who have usually sent the studying information to the students and they constitute a small number of users in this system.

Then we need draw the distribution for a random graph having the same average degree. First of all, we calculate the average degree of the original graph, then, we create a random graph with the same average degree.



The figure above shows the distribution of degrees of the random graph. As we could see, the random graph's degree distribution is more evenly, likes normal distribution. However, in reality the social network, there is a huge number of nodes don't have large degree and only small number of nodes will have a large degree. That means, in reality, in social network, people tend to cooperate with some important peoples.

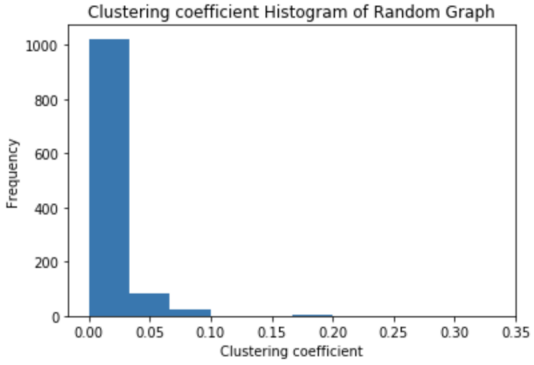
1.4 Draw the histogram of clustering coefficient, and the average clustering coefficient

In graph theory, a clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together. Evidence suggests that in most real-world networks, and in particular social networks, nodes tend to create tightly knit groups characterised by a relatively high density of ties.

There is the histogram of clustering coefficient of Email graph:

And we get the average clustering coefficient is 0.22.

For the random graph, we have the histogram of clustering coefficient:

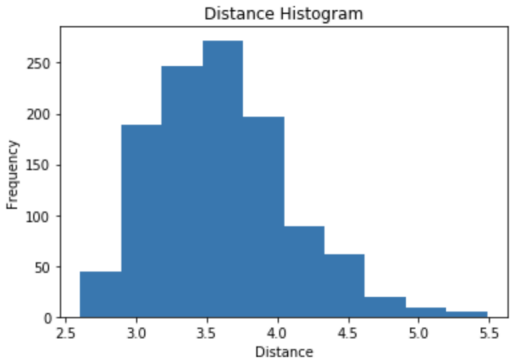


And the average clustering coefficient is 0.01.

So comparing the results, for the distribution of the clustering coefficient of the Email graph, there are a large number of nodes who has a low clustering coefficient and only very few nodes with a high clustering coefficient. But for this random graph, almost all the nodes have the low clustering coefficient. What's more, the Email graph's average clustering coefficient 0.22 is higher than the random one. So in the social network, a network group with relatively high density tends to be formed between nodes. That is to say, the real world network has a higher clustering coefficient than a network obtained randomly.

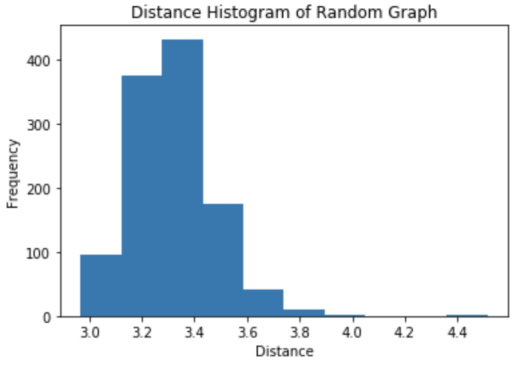
1.5 Draw the histogram of distances in the graphs, the diameter and the average distance

As we know, the distance between two nodes in a graph is the number of edges in a shortest path connecting them. So for each node, its distance is that the sum of the distance between the other nodes and itself divided by the number of other nodes. Using this method, we can get the histogram of distances in Email graph is:



And the diameter is 8, the average distance is 3.61.

For the random graph, the histogram of distances is:



And the diameter is 6, the average distance is 3.32.

As we could see from the comparisons, the diameter and the average distance in the Email graph is bigger than the random graph. In the histogram of Email network, the average distance of each node is at least 2.5 and up to 5.5, which means in the real social network, when the number of users is big enough, many users can not communicate with each other directly, there are certain users are connected via the other peoples. In the random graph, the situation is more evenly and denser, the most distance is between 3 and 4.

1.6 Analyze the degree correlations of the graph

```
In [31]: assortativity = nx.degree_assortativity_coefficient(G)

In [32]: print("The degree correlations of the graph is: " + str(assortativity))
         The degree correlations of the graph is: 0.07820096256581392
```

As we know, the degree correlation is used to detect if the nodes with similar degree tend to connect to each other. If the large degree node tends to connect to a node with a large degree of connectivity, the network is said to be positively correlated; if the large node tends to have a small degree of connectivity, the network is negatively related. The degree correlation of our graph is about 0.078, so it's positive number but a bit small. So we could say that in the real social network, the peoples with large public relations circle will tend to communicate with the same type of people, but this situation is not very usual.

2 Extra requirements

2.1 Detect the communities in the graph

```
# Algorithm of Ravasz

# compute the modelarity of a graph
def get_modularity(G,Adj,nb_edges):
    comps = nx.connected_components(G)
    mod = 0
    for com in comps:
        com_list = list(com)
        for i in range(0,len(com_list)):
            for j in range(0,len(com_list)):
                if i!=j:
                    mod += (Adj[(i,j)] - degree[i]*degree[j]/(2*nb_edges))
    mod = mod/(2*nb_edges)
    return mod

# for every loop, find the edge with max betweenness centrality and remove it
# then recompute the modularity
# output the subgraph with the max modularity
def girvan_newman(G, Adj, nb_edges):
    G_copy = G.copy()
    max_modularity = get_modularity(G_copy, Adj, nb_edges)
    while True:
        betweenness = nx.edge_betweenness_centrality(G_copy)
        max_bwtweeness = max(betweenness.values())
        for edge, value in betweenness.items():
                if float(value) == max_bwtweeness:
                    G_copy.remove_edge(edge[0], edge[1])
                    break
        modularity = get_modularity(G_copy, Adj, nb_edges)
        if modularity > max_modularity:
            max_modularity = modularity
            G_max = G_copy.copy()
        if G_copy.number_of_edges() == 0:
            break
    print("Max modularity is: ", max_modularity)
    return G_max
```

For detecting the communities, I have implemented the divisive community detection (Ravasz) as presented in the course. The main idea is: Firstly we compute the betweenness centrality, then removing the edge having the highest centrality, and recompute the betweenness, we repeat it until there is no edge. For each loop, we get the modularity, we output the largest one and record the corresponding sub-graph.

From the codes, we can get the modularity of the Email network, and when we cut the relation between the important peoples(the nodes between the nodes with high betweenness centrality), we could get a social network with highest modularity.

2.2 Count the number the triangles in the graph, and compare to a random graph

Count the number the triangles of EmailNetwork

```
In [40]: sum(nx.triangles(G).values())/3
Out[40]: 5343.0
```

Count the number the triangles of the random graph

```
In [41]: sum(nx.triangles(GR).values())/3
Out[41]: 186.0
```

In the graph, the number of triangles is 5343. In the random graph, the number of triangles is 186. It is because in the random graph, the distribution of its degree is like a Gaussian distribution, it tends to have less edges. So it has less number of triangles.

2.3 Compute and discuss other centrality measures: betweenness, PageRank

The information of betweenness is:

Betweenness

```
In [45]: print(nx.betweenness_centrality(G))
```

{'1': 0.008363102947366158, '2': 0.0075074958207875684, '3': 0.01018942047576352, '4': 0.0050322766292369 02, '5': 0.0 013481749426585538, '6': 0.003150928500621051, '7': 0.012585050914428348, '8': 0.0006021188230099114, '9': 0.00346979 0560126041, '10': 0.010695003829980886, '11': 0.004118298636384025, '12': 0.0026791056613123146, '13': 0.014128171993 53133, '14': 0.012620451155818405, '15': 0.0049010961541844 96, '16': 0.017842922172167586, '17': 0.00077541159299943 89, '18': 0.0025933903662088906, '19': 0.006839009547402535, '20': 0.002643321367328178, '21': 0.019476237901142984, '22': 0.0007671623646256435, '23': 0.03346297766608285, '24': 0.017216188448430564, '25': 0.007414285182597431, '26': 0.0, '27': 0.004361644058635199, '28': 0.0014672291128109843, '29': 0.0017810007528742 58, '30': 0.0088490049297 1497, '31': 0.003959238520546687, '32': 0.0002338797855558137, '33': 0.0002171036413761 6492, '34': 0.004034040431704064, '35': 0.0, '36': 0.0, '37': 0.0, '38': 0.009597784052551675, '39': 0.007160996499358942, '40': 0.007841404705803577, '41': 0.026499357139407435, '42': 0.02602441770604824, '43': 0.00040113642075034214, '44': 0.004036523081760 28, '45': 0.012256123860893095, '46': 0.0143818697528803, '47': 0.00042659111218912995, '48': 0.0007098347458512115, '49': 0.013 736140598631047, '50': 0.003905067329735171, '51': 0.008870843610739 08, '52': 0.022929092225047812, '53': 1.827474802 27068e-05, '54': 0.013599779600265129, '55': 0.0018980448739129721, '56': 0.007852379153291032, '57': 0.0086129414286 95862, '58': 0.019392268020770518, '59': 0.0062888722608760595, '60': 5.91264234726381 64e-05, '61': 0.002450254801313 1532, '62': 0.007604713189566376, '63': 0.0006691234610173538, '64': 0.002069526186237 2816, '65': 0.006264107899360252, '66': 0.0011776529284815715, '67': 0.005232494434534376, '68': 0.0012991776587609504, '69': 0.016084189244457682, '70': 0.008560708941258962, '71': 0.002876424202308 73, '72': 0.01827374766862954, '73': 0.007150275419269023, '74': 0.009009302811543364, '75': 0.0038173182246077305, '76': 0.030117501315083384, '77': 0.0007319274126930062, '78': 0.0 03003429824907773, '79': 0.001788726114377755, '80': 0.011581039044613023, '81': 0.009606604263024284, '82': 0.008987

Betweenness centrality is a measure of centrality in a graph based on shortest paths. In this graph, we could know that some nodes have the large betweenness centrality because in the real social network, there are some known people who is the center of the social network. For example

in our graph the node with large betweenness centrality maybe the teachers and some important students in the class because they often communicate with different peoples and there are many pairs of people connect via them. And for those node with small betweenness centrality, they are just the normal students.

The information of PageRank is:

PageRank

```
In [44]: print(nx.pagerank(G))
```

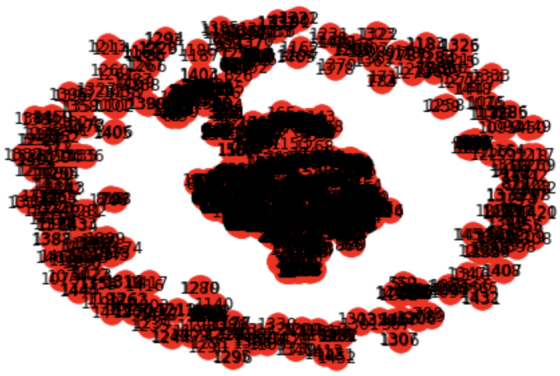

```
{'1': 0.0022458842293395166, '2': 0.0020496677120602183, '3': 0.0027268196308997805, '4': 0.001727678608983813, '5':
0.0008455030445311813, '6': 0.0015919945588248735, '7': 0.002192913529660412, '8': 0.0007244464253297192, '9': 0.0013
081310208668414, '10': 0.0026561447144534994, '11': 0.001559075821711105, '12': 0.00142361557807974, '13': 0.00203514
2140360336, '14': 0.0030892034944525557, '15': 0.0015139694580437093, '16': 0.003580479915951419, '17': 0.00073427522
56037961, '18': 0.0014372260332886425, '19': 0.0022305216499390254, '20': 0.0013428457432897276, '21': 0.003480319707
380225, '22': 0.001188106880346552, '23': 0.0039662211299081573, '24': 0.0034531438044433406, '25': 0.0016852852886525
959, '26': 0.0003354422767416305, '27': 0.0017769621397015541, '28': 0.0009994644145524087, '29': 0.0008415236326141 6
98, '30': 0.002123012824506347, '31': 0.0016580915578935452, '32': 0.0003395556034533 8113, '33': 0.000698514188226991
3, '34': 0.001552200534878613, '35': 0.0002080928292256 2144, '36': 0.0002080928292256 2144, '37': 0.0002080928292256 21
44, '38': 0.002101860554520615, '39': 0.0020557265388437963, '40': 0.0020156922227567 21, '41': 0.0038573542264763944,
'42': 0.0036459853225152077, '43': 0.0008167294775995278, '44': 0.0013505073422475553, '45': 0.0024838763765139206,
'46': 0.002682135519194975, '47': 0.0006131316154968271, '48': 0.0006068351748724713, '49': 0.0028782891991425918, '5
0': 0.001893853792910127 1, '51': 0.001849723040691 2364, '52': 0.0029204092842777 56, '53': 0.0026462131997640 82, '5
4': 0.0025374848149750657, '55': 0.0011525556273684744, '56': 0.0024542779382310705, '57': 0.0022595079695856076, '5
8': 0.0028736781735627386, '59': 0.0016444751689226 14, '60': 0.0003452246759162026, '61': 0.0009864644601287201, '6
2': 0.0019384716811935315, '63': 0.0008917896015355945, '64': 0.0013800164551711 91, '65': 0.0016203898609431166, '6
6': 0.0011260644770823985, '67': 0.0014221537717215656, '68': 0.0013146303462998 12, '69': 0.0026309223421094813, '7
0': 0.0019658074276952403, '71': 0.0011044034546311976, '72': 0.0031020000356522342, '73': 0.0018311611216117571, '7
4': 0.0017501700663784471, '75': 0.0011537853932085517, '76': 0.0033307185161354333, '77': 0.0006856681941811149, '7
```


In the figure above, as we could see, some nodes have the large pagerank. This is almost the same reason as the last question. The users usually need to communicate with some important people in their university, so that they have a larger weight in this data.

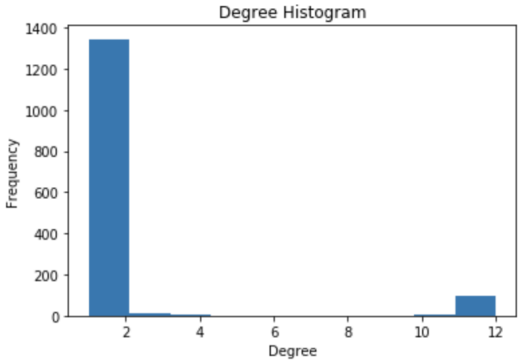
2.4 Do a comparative analysis of the social dataset and a non-social one

Here I choose the road transportation network of the Chicago region (USA). Nodes are transport nodes, and edges are connections.

The graph of this non-social dataset is:

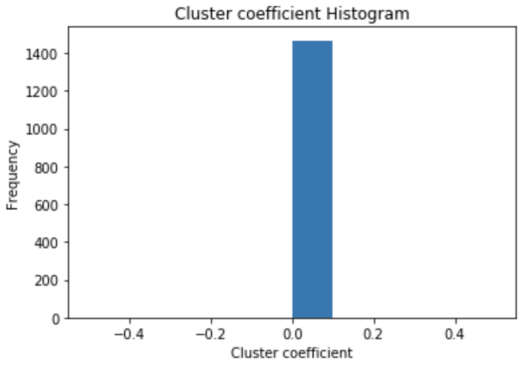


There is the histogram of degree:



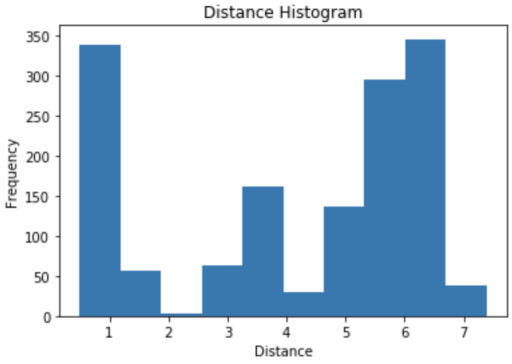
From the figure above, we can know that many nodes are isolated, only a few nodes could connect with other nodes. Maybe it is because in Chicago, there is only a small number of avenue. This is quite different in the social network, in the social network, a node can have relation with many other nodes.

There is the histogram of clustering coefficient:



The clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together. In our non-social network, we can know that almost all the nodes' clustering coefficients are 0. It means that all the nodes which connect with a same node have no connection. So the transportation roads in Chicago are isolated, which is quite different from the social network.

There is the histogram of distance:



From the figure above, we notice that for most node if they want to connect with others, they need go through many other nodes. In real world, it represents that from a station to another station, we need to go through many other stations to arrive the destination. However, in the social network, if 2 people want to get connection, the number of nodes that they go through is less than that in the non-social network.