P0-Crawl

I crawled Foursquare using python package called foursqure, which can be download from https://github.com/mLewisLogic/foursquare. I got an undirected graph with Number of nodes = 81131 and Number of edges = 153509

Inside Folder P0, fs_crawler.py is the code used to crawl Foursquare with seed '31571'. The raw data I crawled is inside the file "foursquare_output". I used the code preprocess.py to anonymize the dataset. foursquare_mapping.txt contains the mapping from usernid to numbers. foursquare_output_assigned_single.csv is the file contains only (i,j) for i < j. Since it is an undirected graph, I also created another file foursquare_output_assigned_symmetric.csv, which contains both (i,j) and (j,i) so that it can pass the MATLAB test code.

Summary:

fs_crawler.py: code used to crawl Foursquare

Preprocess.py: used to do mapping from usernames to numbers

foursquare_output: non-anonimized dataset

foursquare_mapping.txt: usernames->numbers

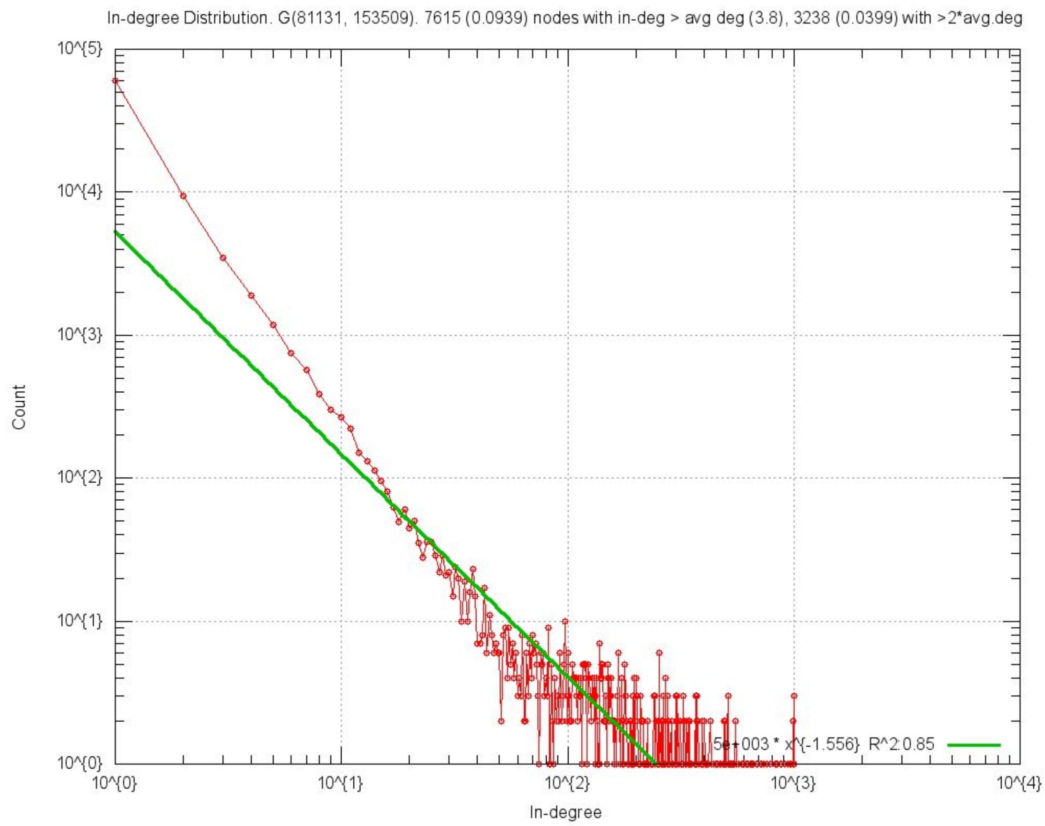
foursquare_output_assigned_single.csv: edge list i,j with i < j

foursquare_output_assigned_symmetric.csv: edge list i,j and j,i

For P1 to P3, I used Sanp for Python (http://snap.stanford.edu/snappy/)

P1-Graph Essential

(i)



From the graph, we can see that the fitted line is: y = 5000*x^(-1.556).

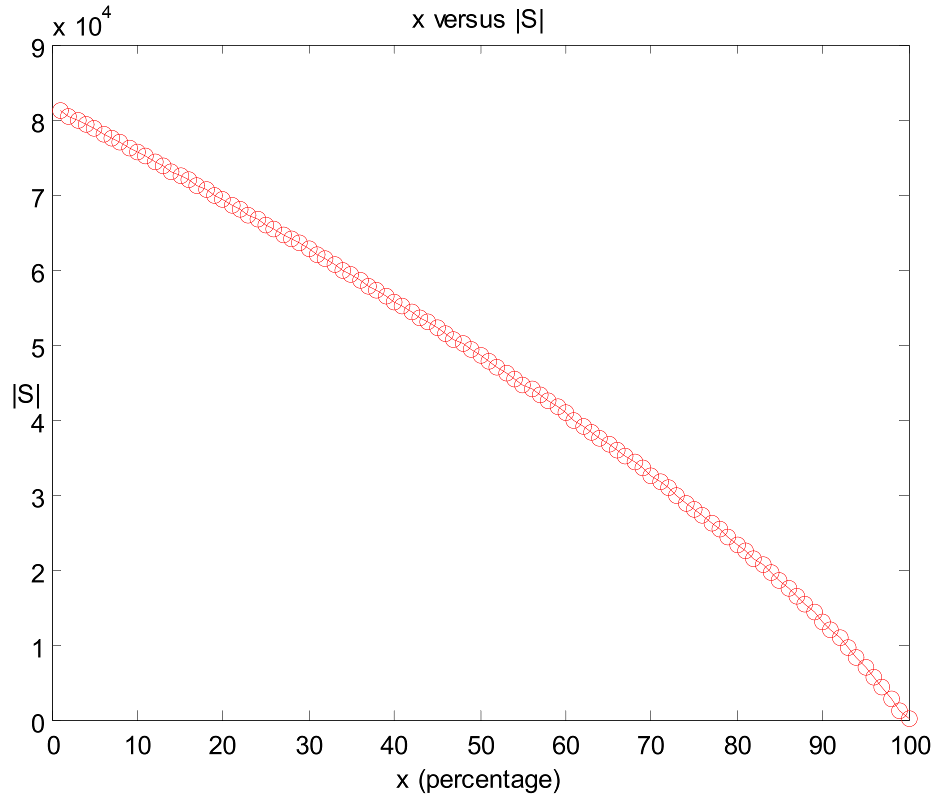
(ii) The number of bridges in the graph is 60562

(iii) The number of 3-cycles is 139482

(iv) The graph diameter is 6

(v) Select the first one

I use Snap to generate the data and plotted using MATLAB. The randIndex.txt is the randomperm generated by MATLAB, I used that to sample the graph. In the graph, x is the percentage of edges removed.



P2-Network Measure

(i) Global Clustering coefficient = 0.0137708434708

Local clustering coefficient = 0.0803507918798

We can see that the global clustering coefficient and local clustering coefficient are not very large. This is because we only crawled 1000 nodes' friend. The friends list of most of these 1000 nodes' friends are not crawled, which makes many nodes only have degree one (leaf nodes). Thus, the global clustering coefficient and local clustering coefficient are not very large.

(ii)

PageRank:

Node Number	PageRank	userID	firstName, lastName	Number of Friends
878	0.00478647071733	3875366	"p", "hollywood"	987
255	0.00432873262595	15203618	"Band of the Day", "App"	989
99	0.00398467170714	3430278	"mayu-chan"	966

Eigenvector centrality:

Node Number	Eigenvector centrality	userID	firstName, lastName	Number of Friends
408	0.179556540206	387	"Scott", "Beale"	995
590	0.154273706431	6053	"Robert", "Scoble"	1000
645	0.138893918899	388	"Lane", "Becker"	831

Degree centrality

Node Number	Degree centrality	userID	firstName, lastName	Number of Friends
644	0.0123752002958	84228	"Phil", "Jeudy"	1004
448	0.0123628743991	1758243	"Jason", "Baptiste"	1003
325	0.012325896709	11471	"David", "Weekly"	1000

After visiting the user profile, I find that in terms of number of friends, degree centrality works best, becasue the nodes it finds has the largest number of friends. In terms of the importance of friends, eigenvector centrality works best., because the friends of the nodes find by eigenvector has the more friends. Eigenvector centrality also assigns a large value to nodes that itself is not important but is connected to some import friends. Pagerank doesn't have this problem. In this sense, Pagerank works best.

(iii) Select the second one: find the two most similar nodes using Jaccard similarity

The two most similar nodes using Jaccard similarity is 192, 239, and Jaccard similarity value is 1.

This is because node 192 has only one friend, which is node 2, and node 239 has only one friend, which is also node 2. So Jaccard similarity of node 192 and 239 is 1, which is the largest since the Jaccrd similarity cannot be larger than 1.

P3-Network Models

(i) random graph

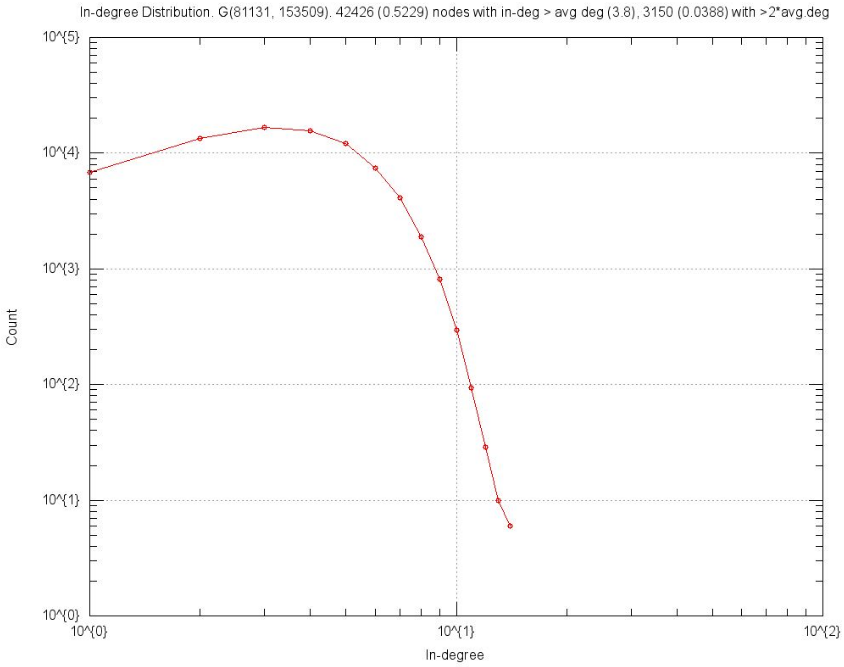
For random graph, Snap provide the Gnm model, n =81131, m = 153509

Average path length: 8.38414222814

Global clustering coefficient: 7.24254061427e-05

Local clustering coefficient: 4.51152495377e-05

The degree distribution is shown below. Since it is an undirected graph, the indgree distribution is the degree distribution. As we can see, it doesn't follow power-law distribution.



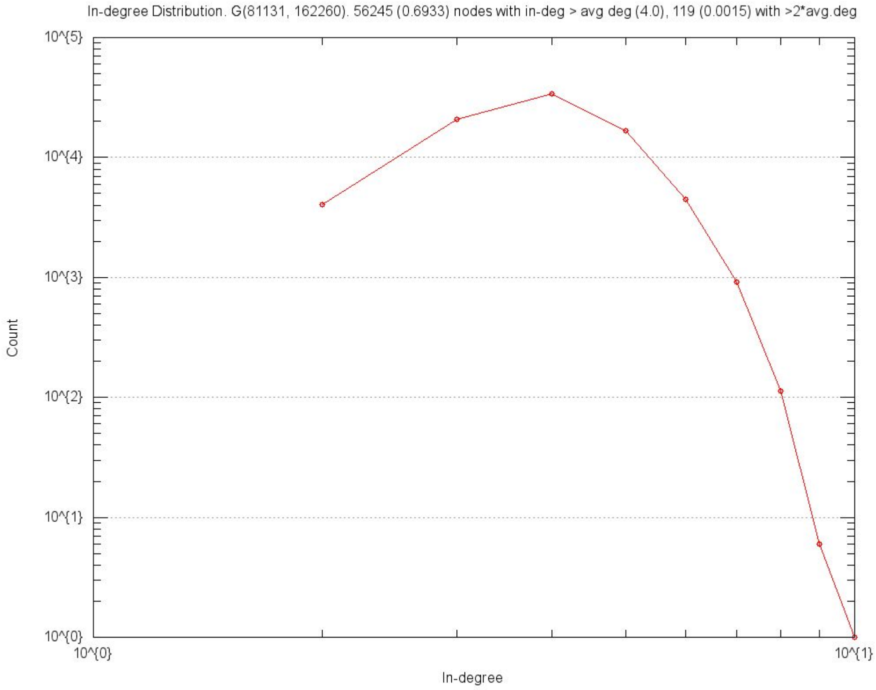
(ii) small world model

For small world model, I set n to be 81131 and use the formula C(p) = (1-p)^3C(0) with average degree c = 3.78, C(0) = 3/4 *(c-2)/(c-1), and I get p = 0.45. Typically, p is between 0.01 to 0.1. However, since I only crawled 1000 nodes and for the remaining nodes, I didn't craw their neighbors, which makes this p large.

Average path length: 9.64214940248

Global clustering coefficient: 0.0746821741387

Local clustering coefficient: 0.0896570566943



The degree distribution of the small world model is not power law distribution. As we can see, the most number of nodes has average degree as 4, which is the degree of the regular lattice.

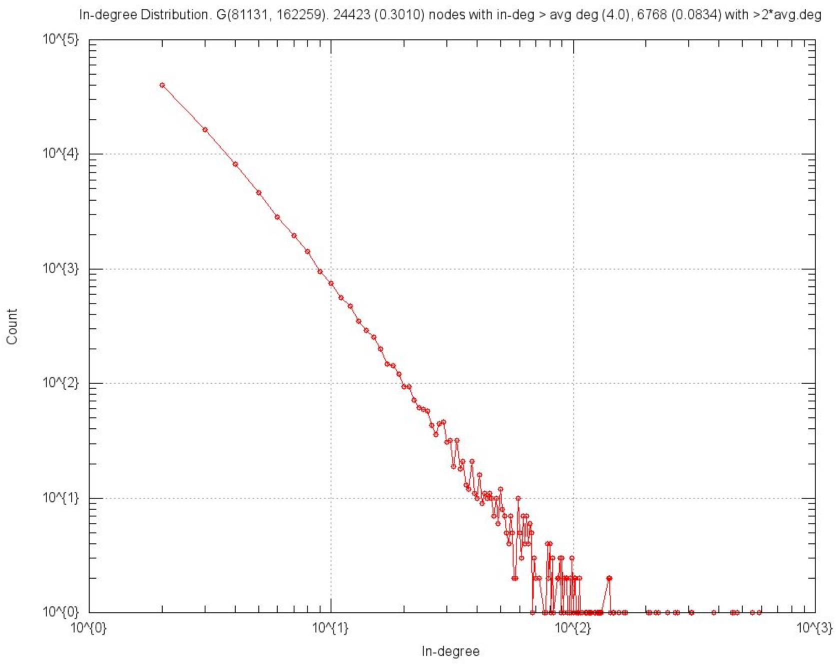
(iii) Preferential attachment model

For preferential attachment model I set n =81131 and average degree to be 4. There are no more other parameter to set for the snap.GenPrefAttach function.

Average path length: 5.8226742587

Global clustering coefficient: 0.000268620849288

Local clustering coefficient: 0.000927072301158



As we can see, the degree distribution of preferential attachment model follows a power law distribution.

Comparison

	Average Path Length	Global Clustering Coefficient	Local Clustering Coefficient	Average Degree
Original Graph	4.22646152876	0.0137708434708	0.0803507918798	3.8
Random Graph	5.38414222814	7.24254061427e-05	4.51152495377e-05	3.8
Small-world Model	5.64214940248	0.0746821741387	0.0896570566943	4.0
Preferential Attachment model	5.8226742587	0.000268620849288	0.000927072301158	4.0

From the above table, we can see that the average path length of Random Graph and Small-world model are very close to the original graph, while Preferential Attachment model has a little larger average path length. For clustering coefficient, small-world has the closest one to the original clustering coefficient, while the clustering coefficients of both random graph and preferential attachment model are very small. The degree distribution of preferential attachment model follows a power law distribution while the other two don't follow.

I also list the top 3 values of PageRank, Eigenvector Centrality and Degree Centrality for the original graph, random graph, small-wold model and preferential attachment model

PageRank

	Original Network	Random Graph	Samll-wold Model	Preferential Attachment Model
Top1	0.00478647071733	4.01300064773e-05	2.83942920421e-05	0.00215545388605
Top2	0.00432873262595	3.93606004525e-05	2.58002590347e-05	0.00137757132879
Top3	0.00398467170714	3.8307145498e-05	2.53924996561e-05	0.00120182176406

Eigenvector Centrality

	Original Network	Random Graph	Samll-wold Model	Preferential Attachment Model
Top1	0.179556540206	0.0356985389118	0.031405109976	0.609648679961
Top2	0.154273706431	0.0335264640426	0.0265582702022	0.0927107703633
Top3	0.138893918899	0.0256746006394	0.0254770243176	0.0842200450552

Degree Centrality

	Original Network	Random Graph	Samll-wold Model	Preferential Attachment Model
Top1	0.0123752002958	0.000172562553926	0.00012325896709	0.0102428201652
Top2	0.0123628743991	0.000172562553926	0.000110933070381	0.00652039935905
Top3	0.012325896709	0.000172562553926	0.000110933070381	0.00569456427955