ITCS 8010/6010: Machine Learning with Graphs and Large Networks Fall 2020

Problem Set 1

Due: 10/21/2020 at 11:59 pm via Canvas

General Instructions

This homework is easy and will get you started on tools for network analysis. If you find these questions difficult, then you might not be ready for this course.

Submission instructions: You should submit your answers via Canvas.

Submitting answers: Prepare answers to your homework in a single PDF file and submit it via Canvas. Make sure that you answer to each sub-question and include your name. Please submit your code as well. It can be a Jupyter notebook (preferred).

Questions

The purpose of these exercises is to get you started with network analysis and the NETWORKX software. For this homework, you need to install and try out the NETWORKX network analysis package.

You can find more details about this exercise on the NetworkX tutorial page:

https://networkx.github.io/documentation/stable/tutorial.html.

1 Analyzing the Wikipedia voters network [9 points]

Download the Wikipedia voting network wiki-Vote.txt.gz: http://snap.stanford.edu/ data/wiki-Vote.html.

Using one of the network analysis tools above, load the Wikipedia voting network. Note that Wikipedia is a directed network. Formally, we consider the Wikipedia network as a directed graph $G = (V, E)$, with node set V and edge set $E \subset V \times V$ where (edges are ordered pairs of nodes). An edge $(a, b) \in E$ means that user a voted on user b.

To make our questions clearer, we will use the following small graph as a running example: $G_{\text{small}} = (V_{\text{small}}, E_{\text{small}})$, where $V_{\text{small}} = \{1,2,3\}$ and $E_{\text{small}} = \{(1,2), (2,1), (1,3), (1,1)\}$.

Compute and print out the following statistics for the wiki-Vote network:

1. *The number of nodes in the network.* (G_{small} has 3 nodes.)
2. *The number of nodes with a self-edge (self-loop), i.e., the number of nodes* $a \in V$ *where* $(a,a) \in E$. (G_{small} has 1 self-edge.)
3. *The number of directed edges in the network, i.e., the number of ordered pairs* $(a,b) \in E$ *for which* $a \neq b$. (G_{small} has 3 directed edges.)
4. *The number of undirected edges in the network, i.e., the number of unique* unordered *pairs* (a,b), $a \neq b$, *for which* $(a,b) \in E$ *or* $(b, a) \in E$ *(or both).* If both (a, b) and (b, a) are edges, this counts a single undirected edge. (G_{small} has 2 undirected edges.)
5. *The number of reciprocated edges in the network, i.e., the number of unique unordered pairs of nodes* (a,b), $a \neq b$, *for which* $(a,b) \in E$ *and* $(b,a) \in E$. (G_{small} has 1 reciprocated edge.)

6. *The number of nodes of zero out-degree.* (G_{small} has 1 node with zero out-degree.)
7. *The number of nodes of zero in-degree.* (G_{small} has 0 nodes with zero in-degree.)
8. *The number of nodes with more than 10 outgoing edges (out-degree* > 10*).*
9. *The number of nodes with fewer than 10 incoming edges (in-degree* < 10*).*

Each sub-question is worth 1 point.

2 Further Analyzing the Wikipedia voters network [6 points]

For this problem, we use the Wikipedia voters network. If you are using Python, you might want to use NumPy, SciPy, and/or Matplotlib libraries.

1. Plot the distribution of out-degrees of nodes in the network on a log-log scale. Each data point is a pair (x,y) where x is a positive integer and y is the number of nodes in the network with out-degree equal to x. Restrict the range of x between the minimum and maximum out-degrees. You may filter out data points with a 0 entry. For the log-log scale, use base 10 for both x and y axes.

3 Finding Experts on the Java Programming Language on StackOverflow [5 points]

Download the StackOverflow network stackoverflow-Java.txt.gz: http://snap.stanford.edu/class/cs224w-data/hw0/stackoverflow-Java.txt.gz. An edge (a,b) in the network means that person a endorsed an answer from person b on a Java-related question.

Using one of the network analysis tools above, load the StackOverflow network. Note that StackOverflow is a directed network.

Compute and print out the following statistics for the stackoverflow-Java network:

1. *The number of weakly connected components in the network.* This value can be calculated in NETWORKX via function *weakly_connected_components*.
2. *The number of edges and the number of nodes in the largest weakly connected component.* The largest weakly connected component is calculated via function call in NETWORKX

4 Network Characteristics [40 points]:

One of the goals of network analysis is to find mathematical models that characterize real-world networks and that can then be used to generate new networks with similar properties. In this problem, we will explore two famous models—Erdos-Renyi and Small World—and compare them to real-world data from an academic collaboration network. Note that in this problem all networks are undirected.

Erdos-Renyi Random graph (G(n, m) random network): Generate a random instance of this model by using n = 5242 nodes and picking m = 14484 edges at random. Write code to construct instances of this model, i.e., do not call the NetworkX function. Compare your results (major properties like degree distribution, clustering coefficient, and diameter) with the NetworkX implementation.

Small-World Random Network: Generate an instance from this model as follows: begin with n = 5242 nodes arranged as a ring, i.e., imagine the nodes form a circle and each node is connected to its two direct neighbors (e.g., node 399 is connected to nodes 398 and 400), giving us 5242 edges. Next, connect each node to the neighbors of its neighbors (e.g., node 399 is also connected to nodes 397 and 401). This gives us another 5242 edges. Finally, randomly select 4000 pairs of nodes not yet connected and add an edge between them. In total, this will make m = 5242 · 2 + 4000 = 14484 edges. Write code to construct instances of this model, i.e., do not call a NetworkX function. Report the properties of this network.

5 Random Graphs with Clustering [40 points]

Consider the following random graph model with clustering. For n nodes, we have $\binom{n}{3}$ distinct 'triplets'. For *each* triplet, with independent probability p we connect the nodes belonging to this triplet in the graph using three edges to form a triangle, where $p = \frac{c}{\binom{n-1}{2}}$, where c is a constant.

Assume n is very large.

Question: Prove that the expected degree in this model is 2c. [Hint: expected degree of a node u in this generative model is equal to twice the expected number of triangles incident on u]

Question: What is the clustering coefficient C? What is the value of C as n tends to infinity?

Question: Implement this model to computationally derive degree distribution, diameter, and clustering coefficient.