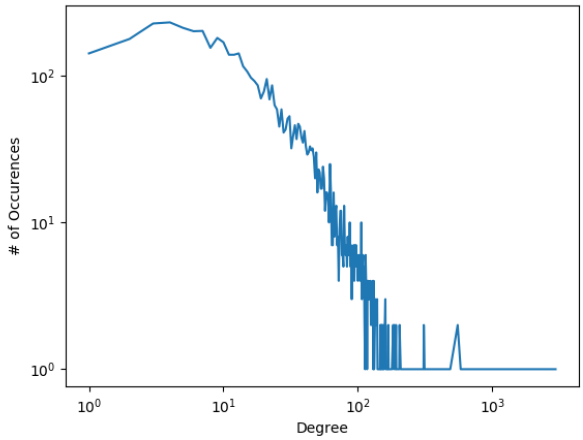
Christian Zinck
10/18/18

HW 3

1.

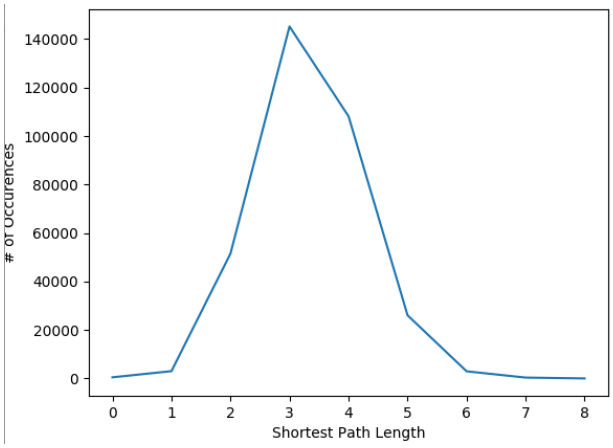
a. The PPI network appears to fit the definition of scale-free. When both axes are spaced logarithmically, the degree distribution has a strong negative linear correlation, meaning the distribution follows the power law and is therefore scale-free.



b. YGR296W has a clustering coefficient of .1 and YPL098C has a clustering coefficient of .8.Since they both have 5 interacting partners, we can conclude that the neighbours of YPL098C are much more interconnected than the neighbours of YGR296W. In other words, the neighbours of YPL098C have 8/10 of the 3-cliques needed to create a 5-clique while the neighbours of YGR296W have 1/10 of the 3-cliques needed to create a 5-clique.

c. There are 354514 triangles in the PPI network.

d. The distribution of shortest path lengths is typical of a PPI network. There are few shortest paths with length < 2 and length > 5, which is to be expected of a PPI network. I would expect the average shortest path length to be slightly lower, but since it is a sample of a larger PPI network, it does not surprise me that the average is a little high.



e. From the graph above, the diameter appears to be 8, although I ran the random sampling multiple times and got diameter values ranging from 6 to 8.

2.

a. Implemented the majority vote algorithm.

b. For my algorithm, I chose to implement weighted majority vote. It is nearly identical in theory and in code, except when I tally the votes from a node's neighbour, I don't add 1 for each vote, I add the weight between the node and its neighbour for each vote.

c. Majority vote correctly labeled 2332/5001 nodes (0.466306738652%), while weighted majority vote correctly labeled 2537/5001 nodes (0.507298540292%) using leave one out CV where a node was considered correctly labeled if the label it was assigned was one of its correct labels.

d. After tallying all the votes as in the vanilla majority vote algorithm, there are two ways I can think of for setting a threshold to decide which functional labels to assign to a node. The first way is to set a maximum number of labels a node can have, and attribute at most that many labels to the node, based on number of votes. The second way is to set a percentage threshold, meaning a certain percentage of the nodes neighbours must have a given label in order for the node to be assigned that label. In either case, the classification of a node will not always be simply right or wrong because it is possible to predict some, but not all, labels. I would propose a scoring system that gives 1 point for each correctly labeled function, -1 point for each function that applies to the node, but was not labeled accordingly, and -2 points for each function that was labeled, but does not apply to the node. We can then use this scoring system to optimize the threshold parameters.