Information Retrieval.

Network models

Student:
Maria Slanova
DMKM, 2015

In this lab I used Python network analysis tool - NetworkX. Other libraries, that were used for plotting and computation - Numpy, Matplotlib, Math.

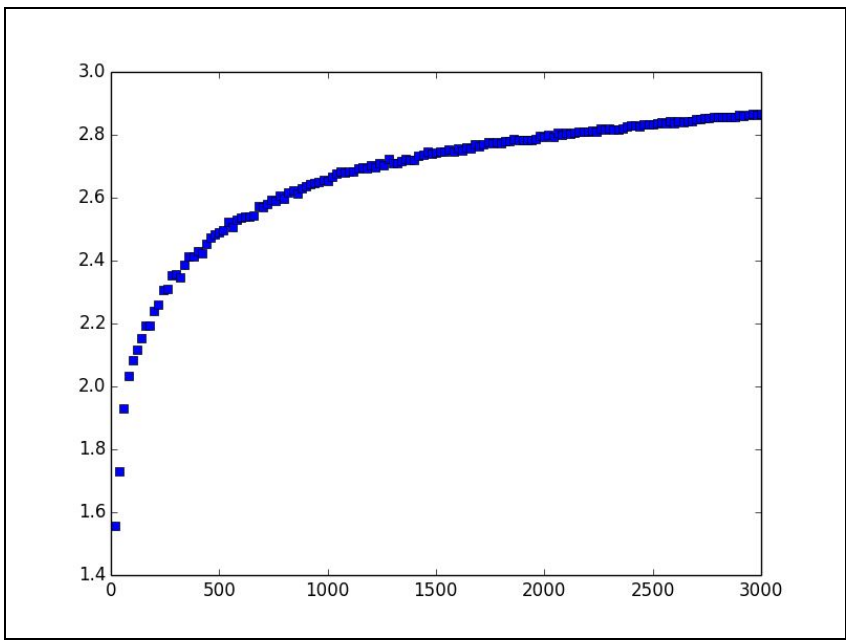
Erdos-Renyi model.

The graph for Erdos-Renyi model was created with the use of NetworkX function - *nx.erdos_renyi_graph.* The function takes as an input the number of nodes and probability *p* of creating an edge to build the graph. If the *p* is initially not specified, then there is a certain probability of creating a disconnected graph. Disconnected graphs make it complex to compute the average shortest path, since there is not one single graph, but a group of disconnected “subgraphs”. To avoid this problem, the probability is computed as a function of *n* - number of nodes. For our model the following function was used:

$$p = \frac{2.8 * ln(n)}{n}$$

The number of nodes also creates a difficulty, because when the number is high enough (8000-10000), then the computation takes a very long time. In the following example (Picture 1), the number of nodes is 3000, the step is 20, the computation time more than 15 minutes:

Picture 1.



Watts-Strogatz model.

The graph for Erdos-Renyi model was created with the use of NetworkX function - *nx.connected_watts_strogatz_graph.* The function takes as an input the number of

nodes (*n*), the number of nearest neighbours, to which each node is connected (*k*), the probability (*p)* of rewiring each edge and number of attempts to generate a connected graph (*tries*). The parameter *k* was initially set to *k = 4,* as stated in the lab description, *tries* was set to *tries = 100*.

The average shortest path and the cluster coefficient were calculated using the functions - *nx.average_shortest_path_length* and *nx.average_clustering.* Since clustering coefficient and average shortest path are of different scales, they need to be normalized before plotting them together. At first, clustering coefficient and average shortest path were normalized by dividing the values by the value obtained at the left-most point. But this kind of normalization didn't give the expected results. The graph shown on Picture 2, is the one that was build after this normalization. Here the blue graph is average shortest path, and the red one - clustering coefficient. Afterwards, the min-max normalization was used:

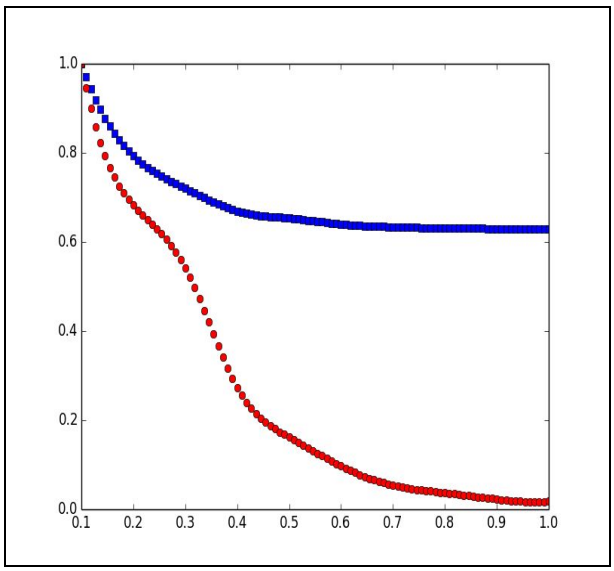
$$ASP_{normalized} = \frac{(ASP - ASP_{min})}{(ASP_{max} - ASP_{min})}$$

$$CC_{normalized} = \frac{(CC - CC_{min})}{(CC_{max} - CC_{min})}$$

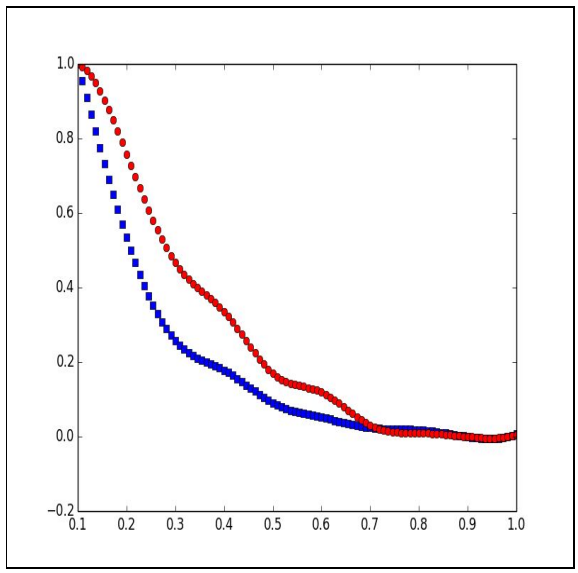
The resulted graph is shown on Picture 3.

The last step to get the same result as in lab description, was to plot the graphs using the log(nonlinear) scale. The resulted graph is shown on Picture 4.

Picture 2.



Picture 3.



Picture 4.