Social Computing (CS60017)
Autumn 2021
Assignment 2: Network centrality and community structure

Assignment given: 18th October, 2021
Deadline:, 1st November 2021, 23:55 pm
Full marks: 50

[IMPORTANT] General instructions:

- This assignment teach you the actual network structures in the real world networks
- Students should use networkx library for graph analysis: https://networkx.org/documentation/stable/index.html
- Please add one readme.txt file with instructions for running your code for each question. If we cannot run your code, there might be a severe penalty.
- Given a question Q, output_Q.txt file should be generated by your code at runtime only.
- Please submit your code files in ipynb format (jupyter notebook) and readme.txt in a single zip file named <ROLL_NO>_Assgn2.zip
- Please follow the nomenclature of the files strictly as instructed above.
- If we detect plagiarism, there will be severe penalties. So please don't copy.
- Please use CSE Moodle to submit your zip file.
- It's preferable that you use python for your code.

Q1. Download the datasets, use the following algorithms and solve the below-mentioned problems for each dataset and each method. **[(9+9) + (9+9) marks]**
Datasets: (i) Football network (*Football*) [Link, 12 communities], (ii) Polbooks network (*Polbooks*) [Link, 3 communities]

- Consider the *value* of the node mentioned in the ".gml" file as its ground truth communities. For plots, create "PLOT" folder and store the plots inside that folder.
- Use the following two "Community Detection Algorithms" (*CDA*) -- (i) Newman-Girvan method using betweenness centrality (*NEWMAN*), (ii) Greedy community detection using Clauset-Newman-Moore greedy modularity maximization to compute communities (*CLAUSET*).
- For CDA methods (i) and (ii), generate output_1_NEWMAN.txt and output_1_CLAUSET.txt respectively.
- Solve all the problems below (a to f) for each algorithm and each dataset.

[a] Report the number of communities generated using these two methods for each dataset. In case, you are bound to mention the *number of communities to be generated* as a parameter then set it equal to the *number of true communities*. Write it in the **output_1_<methodname>.txt** file. For example, print "Dataset: Football\n Number of communities using Newman-Girvan method: xxxxx\n" for Newman-Girvan method on Football dataset.

[2+2 marks]

[b] Given a CDA method, plot the distribution of the size of generated communities for each dataset. The filename of each plot should be **<datasetname>_<methodname>_dist.png**.

[2+2 marks]

[c] Plot the distribution of the size of ground-truth communities. Ground truth communities (true communities) can be found inside the dataset. Save it as **ground_truth_communities_dist_<datasetname>_<methodname>.png**.

[1+1 marks]

[d] Given a method, print the size (number of nodes, number of edges) of the top 5 communities (top communities should be computed based on the number of nodes present in it. The Largest number of nodes indicate the topmost community.). Print as "Dataset: Football\n Community i: (number_of_nodes, number_of_edges)\n" for i^{th} community in Football dataset.

[2+2 marks]

[e] Given a method, consider the top 5 communities based on the question (d), find out the coverage of each community. In the **output_1_<methodname>.txt** file, write it as "Dataset: Football\n coverage of community i = coverage_value\n" for i^{th} community in Football dataset.

[1+1 marks]

[f] Given a method, consider the topmost community and compare it with the topmost ground-truth community (follow the same logic as (d)). Compute Jaccard coefficient between *nodes of* generated topmost community and ground-truth community. In **output_1_<methodname>.txt** file, write it as "Dataset: Football\n jaccard coefficient = jaccard_value\n" for Football dataset.

[1+1 marks]

Q2. Download the dataset and solve the below-mentioned problems.
Dataset- Download here

A. Write a code to implement a method for computing normalized closeness centrality (slide, page 8) metric for a graph. Given a graph, your code has to calculate closeness centrality for each of the nodes. **[10 marks]**
   Please note the following points carefully-
   a. Your code file should be named **gen_centrality.py** and should NOT take any input arguments. Include the dataset in your submission.
   b. The format for the input file (.adj file) is as follows:
      i. The first line of the file specifies the dimensionality (rows and columns) of the adjacency matrix (i.e., number of vertices) of the network.
      ii. Each subsequent line specifies the source vertex ID, the target vertex ID, and the edge weight.
      iii. The vertices are identified by their number.
   c. The code should output a text file which contains a line for each of the nodes. Name the output file as **output_2_closeness.txt**. Each line in the output files has the format: *node_ID <white space> centrality _value*.
   d. There are already built-in functions in networkx to calculate all these centralities. However, you should **NOT** use the function. You need to implement the centrality by yourself.

B. Using built-in functions- **[2 + 2 marks]**
   a. Compute the normalized closeness centrality of each of the nodes using the library networkx. Plot a histogram of centrality values of all the vertices. Name the output file as "closeness_dist.png" and store it inside the PLOT folder.
   b. Take the top-ranked (by the closeness values) 50 nodes generated by networkx. Take the top-ranked 50 nodes generated by your implementation of the corresponding closeness centrality, in part A. Calculate and print how many nodes among the top 50 overlap. Your code should print the following lines in **output_2.txt** file: *#overlaps for Closeness Centrality: <value>*.