Assignment-5: Community Detection

In this assignment you will be implementing a community detection algorithm using a divisive hierarchical clustering (Girvan-Newman algorithm). You will be making use of 2 python libraries called networkx and community. The community library will be provided to you and you need to simply import it. The networkx is a python library which can be installed on your machines. The assignment will require making use of the betweenness function and the modularity function which are a part of the networkx and the community libraries respectively. You will also need to use the matplotlib library for plotting the communities.

Do not use the best_partition() function in the community library to obtain the partitions of the input graph.

This assignment also contains a 20% bonus section, which will be explained at the end of the assignment description.

Assignment Hints

- Read the input file into a graph using Networkx.
- Use the betweenness of the edges as a measure to break communities into smaller communities in divisive clustering.
- The result should be the set of communities that have the highest modularity.
- Use the modularity function in the community API
- Use the betweenness function from networkx
- After the best set of communities are obtained, use matplotlib and networkx functions to plot the communities with different colors. An example for this image will be uploaded.

Execution Details

The python code should take in two parameters, namely input file containing the graph and the output image with the community structure. For example:

Python pooja_anand_communities.py input.txt image.png

Input Parameters:

input.txt: This file consists of the representation of the graph. All graphs tested will be undirected graphs. Each line in the input file is of the format:
1 2
where 1 and 2 are the nodes and each line represents an edge between the two nodes. The nodes are separated by one space.

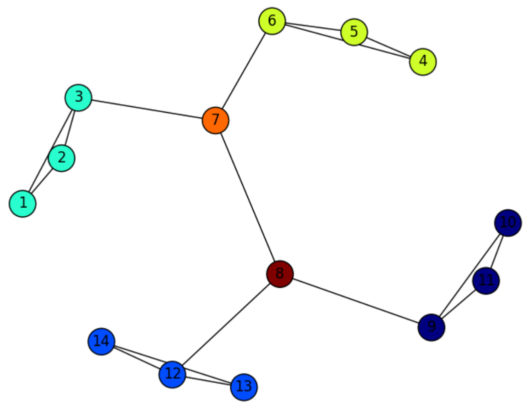
image.png: This will be a visualization of the communities detected by your algorithm. You should represent each communities in a unique color. Each node should contain a label representing the node numbers in the input file. Please refer the sample image file for a clear idea.

Output:

The Python code should output the communities in the form of a dictionary to standard output (the console). Each community should be an array representing nodes in that community. In each array, the nodes should be sorted lexicographically. For example:

```
[1,2,3,4]
[5,6,7,8]
[9]
[10]
```

These 4 arrays represent the 4 communities. Please refer the image below.



A sample input and output file will be provided with the image file.

20% Bonus Component

This is an optional component of the assignment. You should implement your own function to find the betweenness of the edges and not use any of the library functions to do the same. You should implement the Girvan-Newman algorithm to find the betweenness value of all edges. Please write this as a separate function and make sure to comment it clearly. You should submit only 1 python file. Your submission can either have your own implementation of betweenness or use the community library for the implementation. If you use your own function, you will be awarded extra 20% of your actual score. For example, if your score is 40/50 and you implemented the betweenness function, you will get a score of 48.

Submission

- Please submit your Python code with comments for each module. Try to make the code modular and easy for the graders to go through.
- If you have implemented the betweenness algorithm on your own name your file as <firstname>_<lastname>_communities_bonus.py.
- If you are using the library function to calculate betweenness name your file as <firstname>_<lastname>_communities.py
- You do not need to submit your image files or any other output files.

General Instructions:

1. Do not zip your files
2. Make sure your code compiles before submitting
3. Make sure to follow the output format and the naming format.
4. Make sure not to write the output to any files. Use standard output to print them.
5. We will be using Moss for plagiarism detection.