Assignment 1 for E0 259: Data Analytics

Chinmay Jain (SR No: 23006) [*1]

1. Introduction

This report details the implementation of two Communities detection algorithms Girvan-Newman Algorithm and Louvain Algorithm for finding better and meaningful communities.

2. Datasets

2.1. Wikipedia vote network Dataset

This dataset represents voting activity on Wikipedia administrator elections up to January 2008.

Statistic	Value
Nodes	7,066 (99.3% of total)
Edges	103,663 (100% of total)

Table 1. Statistics for LastFM Asian Users Dataset

2.2. LastFM Asian Users Dataset

This dataset represents a social network of LastFM users from Asian countries, collected in March 2020.

Statistic	Value
Nodes	7,624
Edges	27,806

Table 2. Statistics for LastFM Asian Users Dataset

2.3. Data Format

Both the dataset are provided as an edge list where each line represents a directed edge from first node to second node.Wiki-vote dataset is in .txt format and the Lastfm is in .csv format.

3. Libraries and Attributes Used

- `NumPy`: Used for numerical computations like array manipulation and matrix operations.
- `Collections (defaultdict, deque)`: Used for efficient data structures like dictionaries and queues for graph traversals and calculations.
- `Multiprocessing (optional)`: Used for parallelizing the betweenness centrality calculation in the Girvan-Newman algorithm for improved performance on larger datasets.
- `Matplotlib`: Used for generating visualizations of dendrogram.

4. `Algorithm 1`: Girvan-Newman Algorithm

- Edge Betweenness: It measures the number of shortest paths passing through an edge. High betweenness indicates the presence of a bridge between communities.
- **Algorithm**:
  1. Calculate edge betweenness for all edges.
  2. Remove the edge with the highest betweenness.
  3. Recalculate betweenness and repeat until the graph splits into communities.

  we are returning a hierarchial structure in form of a matrix where each column represents a level of division, and each element indicates the community ID for a node at that level, community ID is the lowest node index of that community.

4.1. `Optimisation:`

Use of multiprocessing is used to reduce the runtime of algorithm significantly. We can paralellize the calculation of betweenness calculation which is the performance bottleneck of the algorithm.To address this, we divided the task among multiple processes, distributing the nodes across available processors. Each process independently calculates the local betweenness centrality for its assigned nodes, and the results are subsequently aggregated.

Another possible optimisation which can be implementedis that betweenness is calculated only for a random subset of k nodes. This reduces computational load while still capturing essential network structure, making the algorithm faster.

5. `Algorithm 2`: Louvain Algorithm

In the Louvain algorithm we optimize **modularity** , a metric that measures the quality of community partitions within a network.
Phase 1: we iteratively move each node to the community of neighbor that results in the highest gain in modularity. we repeat this process till there is no improvement in the modularity.
Phase 2: Supernodes are created by aggregating the all the nodes of a community creating a new smaller community.

This algorithm returns the vector storing the community assignment of each node after one iteration.

6. Questions

6.1. Stopping criteria:

The stopping criteria would be "Modularity Value". We will stop the Grivan-Neuman algorithm when the modularity will not increase.

6.2. Best decomposition of communities:

To select the best decomposition of nodes into communities, first define modularity as the key metric for partition quality. Next, generate various community structures using algorithms like Girvan-Newman or Louvain. For each partition, calculate the modularity score to measure how well the communities are formed. Check the consistency of these partitions by running the algorithms multiple times and ensure they meet practical constraints, such as the desired number of communities and ease of interpretation. Finally, choose the partition with the highest modularity score, while also considering its practical relevance and stability.

6.3. Running time:

The running time of Grivan-Neuman algorithm till it was broken into approximately 4000 communities was 48 hours. The running time of algorithm was a few minutes.

6.4. Better Algorithm:

The Grivan-Neuman algorithm will perform better in this case because we are running only one iteration of Louvain Algorithm.

7. RESULTS

The Community at which louvain algorithm for Wiki-Vote stopped was 54 communities i.e. the modularity decreased at 54 communities and for last-fm dataset it was 32.

The stopping criteria by automated Grivan Neuman algorithm was 109 communities for wiki-vote and for last-fm it was 64 commuities.

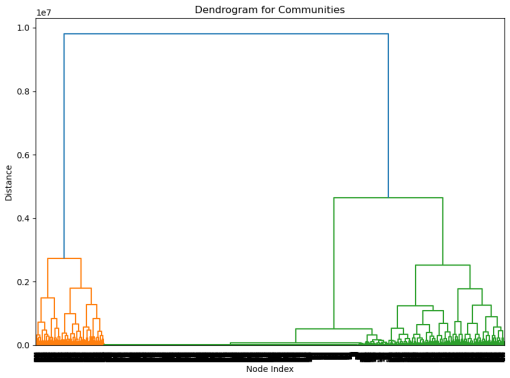

Figure 1. Results on running `Girvan-Newman Algorithm` on `Wiki-vote dataset`

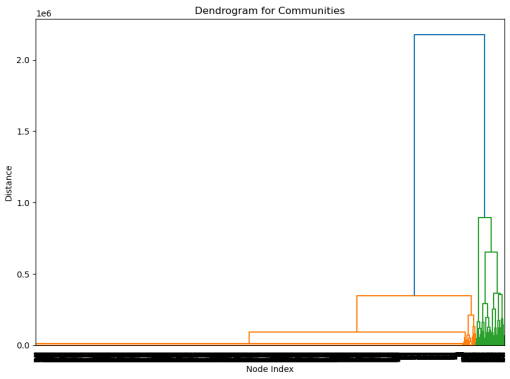

Figure 2. Results on running `Girvan-Newman Algorithm` on `last-fm dataset`