LINFO1115: Project – Opinion Graph Analysis

Peter VAN ROY Marie LONFILS

Academic Year: 2024 – 2025

Context

Opinion graphs are fundamental in various applications, including recommendation systems, reputation management, market analysis, and fraud detection. They can model e-commerce platforms, social networks, peer-to-peer networks, citation networks, etc.
In this project, you will analyse a trust-based opinion network, examining its structure and identifying key relationships between the users. By exploring the connections and trust levels within the network, you will identify how opinions propagate, detect clusters of like-minded individuals, and assess the overall trustworthiness of different users.

1 Guidelines

1.1 In practice

This project is made of 5 different questions, summing up to a total of 20 points. You do not need to have the answer to a previous question to be able to answer the next one.

This assignment must be completed by **group of two students**. If you do not manage to find a partner, please start by requesting a partnership via the Moodle forum. Then, **if and only if** you cannot find anyone else to work with, get in touch with Marie Lonfils.

For this project, you have to use the Python language. We request you to implement all algorithms for the project by yourself, only using the authorised libraries. If you used ChatGPT or any other artificial intelligence, you should clearly mention how you used it and what for you used it. A code template is available on Moodle with details about what should be returned for each question. You will also have to write a report analyzing the obtained results.

1.2 Data format

The dataset "epinion.txt" is available on Moodle and a code template shows you how to load it. Every entry of the dataset represents the opinion of one person on another one. The dataset has three columns. A line of the dataset `(u, v, w)` means that node `u` has opinion `w` on node `v`. The opinion is either positive (+1) or negative (-1). The graph is directed but is considered undirected in tasks 1,3 and 5. When representing the data as

an undirected graph, only create **one** edge between two nodes even if the edge is present multiple times in the dataset.

1.3 Deadline

The assignment is due by **Sunday, April 20th**, 2025 at 23:59. The code should be submitted on Inginious (`https://inginious.info.ucl.ac.be/course/LINFO1115`). The report must be handed in on Moodle as a **zip file containing both your report and your complete source code**. **Only one student per group of two should submit on Inginious and Moodle (the same student for both)**.

2 Tasks

2.1 Task 1: (6 points/20)(Easy)

*For this exercise, consider the graph as an **undirected** graph.*

In this first exercise, we are interested in knowing more about the general structure of the graph. Therefore, we ask you to provide:

1. The average degree of the nodes (average number of connections per node)
2. The degree distribution histogram of the graph (occurrences of each degree up to degree 20).
3. The number of bridges in the graph.
4. The number of local bridges in the graph.
5. Compute the degree of each node attached to a local bridge. Then, use a Student's t-test with a significance level of 0.05 to check if the mean of the computed degrees is equal to the average degree. The t-statistic is given by

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$$

   You can retrieve the p-value associated to the t-statistic by using the `scipy.stats.t.sf` function which is imported as `p_value`. Be sure to call `p_value` in your submission on Inginious. What can you conclude from the p-value ?

Interpret those numbers and the degree distribution histogram.

2.2 Task 2: (4 points/20)(Intermediate)

For this exercise, you must consider the graph as a directed one.

Identify the node that is the best regarded in this graph. We define the score of node p as:

$$s(p) = \sum_{n \in B(p)} w(n, p)$$

where $B(p)$ is the set of nodes pointing to p and $w(n, p)$ is the weight of the edge pointing from n to p. Compute the score of each node and report the node with the highest score. Additionally, provide a graph representing the average number of incoming edges for each score and plot the function $f(x) = |x|$. Comment on the results.

2.3 Task 3: (3 points /20)(Intermediate)

For this exercise, you must consider the graph as undirected.

Measure the distance of the shortest path between each pair of reachable nodes and generate a graph of the number of paths having a given distance value (number vs length); you can find an example of such a graph in the slides of the first course. Then, give the diameter (the longest shortest path between two nodes) of the largest connected component of the network. Interpret the obtained values.

2.4 Task 4: (4 points/20)(Intermediate)

For this exercise, you must consider the graph as a directed one.

We want to measure which nodes are trustworthy. Therefore, we want to compute the PageRank score of each node in the network.

The common PageRank (PR) score of a node p is usually computed recursively as

$$PR(p) = \frac{(1-d)}{N} + d \sum_{n \in B(p)} \frac{PR(n)}{Nout_n}$$

where N is the total number of nodes in the graph, $B(p)$ is the set of nodes pointing to p, $Nout_n$ is the number of outgoing links of node n and d is the damping factor, having a value of 0.85. After convergence, the PageRank scores need to be normalized so that the total of all PageRank scores sums to one. Note that the system can be considered as "converged" when the total sum of value updates is less than or equal to $N \cdot 10^{-6}$.

We ask you to compute the PageRank score of each node. Give the ID number of the node with the highest weighted PageRank score and indicate its value.

2.5 Task 5: (3 points /20)(Advanced)

*For this exercise, you must consider the graph as **undirected**.*

We would like to analyse further the relationship between the nodes by measuring how close the friends of a given person are to each other. First, compute the number of triangles contained in the graph. Then count the number of balanced and unbalanced triangles. Finally, compute the global clustering coefficient (GCC) defined as:

$$GCC(G) = \frac{\text{number of closed triplets}}{\text{number of open triplets}}$$

where a triplet is a set of three distinct nodes. A triplet is said to be open if the three nodes are connected by two edges and closed if the three nodes are connected by three

edges. A triangle contains three closed triplets, each centred on one of the three nodes.

Interpret your results.

3 Deliverables

The code submitted on Inginious should follow the structure shown in the template on Moodle. Note that if the code does not compile when we run it or if the code does not do exactly what is required, you will not succeed in the project.

Your report should be **maximum 4 pages** long.
For each task, we request you to explain the algorithms you implemented, comment on your quantitative results and indicate for each algorithm, the temporal complexity. You should also explain how you obtained the complexity. **This complexity should be expressed using the number of vertices V and/or the number of edges E**.
The report should have the following structure and should contain the following information:

Name and NOMA of the two students.

1. Task 1
    1.1. Results: Indicate the average degree of the nodes, the number of bridges, the number of local bridges and the p-value of the Student's t-test.
    1.2. Graph: Show the histogram of the degree distribution.
    1.3. Interpretation: Interpret the obtained values in the context of an opinion graph.
    1.4. Complexity: Indicate the time complexity of your algorithms.
2. Task 2
    2.1. Result: Give the node with the highest score (the node number + the score).
    2.2. Graph: Show the graph with the average number of incoming edges for each score, alongside the function $f(x) = |x|$.
    2.3. Interpretation: Interpret the obtained graph.
    2.4. Complexity: Indicate the time complexity of your algorithm.
3. Task 3
    3.1. Result: Indicate the diameter of the largest strongly connected connected component of the graph.
    3.2. Graph: Show the graph of the number of shortest paths between two nodes, having a given value.
    3.3. Interpretation: Interpret the results.
    3.4. Complexity: Indicate the time complexity of your algorithm.
4. Task 4

4.1. Result: Indicate the highest PageRank score you found and the id number of the related node.

4.2. Complexity: Indicate the time complexity of your algorithm.

5. Task 5

5.1. Result: Give the number of triangles, the number of balanced triangles, the number of unbalanced triangles and the global clustering coefficient of the graph.

5.2. Interpretation: Interpret the obtained values.

5.3. Complexity: Indicates the time complexity of your algorithm.