COVID-19 Contact Tracing Visualization Dashboard

LDATA2010 Project

Networks arise in increasingly many situations in data science. As introduced during the exercise session on network visualization, displaying large graphs requires care and is far from being an obvious task. Various algorithms have been designed for the sake of determining the network layout which best reveals its structure.

In this project, you will have the opportunity to create a *specialized* network visualization dashboard. This includes getting familiar with layout algorithms, enabling the user to change the color and/or size of the nodes and edges according to some network attributes or properties, allowing the user to interactively navigate in the network, depicting multiple views of the graph structure, etc. Obviously the aim of the project is *not* to develop such a complete software as Gephi. Instead you are asked to implement only the features necessary for the dataset provided, with a careful look at user experience and user interaction.

1 Contact tracing dataset

Dataset Context

Your goal is to create a dashboard for the analysis of contact tracing information in the context of covid-19. Because such data is not readily available, you will receive a simulated dataset, that you will be able to use for the visualization and testing of your application.

The dataset represents timed interactions between people, for example tracked by the "coronalert" app.

Each line in the file contains information about interactions between two people :

- `person1` (integer)
- `person2` (integer)
- `infected1` and `infected2` : whether each person was infected or not at the time of interaction
- `timestep` : the time of the interaction, an ever-increasing integer.
- `location` (in latitude, longitude) of the interaction
- `home1` and `home2` (in latitude, longitude), the home municipality of each person

1.1 Exploratory analysis

By using the software you will provide, the user should be able to answer, at the very least, the following questions :

- Who is seeing a lot of people ?
- When are interactions taking place ?
- Where are most interactions taking place ?
- Are there clusters ? Who are the links between those clusters ?
- Is everything under control ?

2 Basic Features

You will develop a software with the following features:

1. A user must be able to add/upload a new graph in your software to visualize it and interact with it. The files are provided in a CSV format.
2. You will enable the user to employ several different layout algorithms. The user will be able to change the parameters of the algorithms to observe how they change the visualization. For instance, you use one or several force-directed layout algorithms, an adjacency matrix view with a heuristic to order the nodes, a circular layout, etc. Some ideas can be found in the document `mcguffin-2012-simpleNetVis.pdf`.
3. A user will be able to display multiple views of a network using different layouts, to explore different facets of its structure.
4. You will enable the user to compute some basic properties and metrics of a network, such as betweenness centrality, the clustering coefficient, minimum spanning trees, the shortest path between a source and target node (specified by the user), communities among the nodes, etc. To compute these properties, feel free to employ relevant toolboxes, such as `networkx` in Python. The user will have the possibility to highlight these properties on the graph display, for instance by coloring the shortest path between a source and target node, coloring the communities, etc.
5. A user will be able to change the color and size of the nodes and edges *according to metrics of the network* (simply coloring all nodes with a unique color is not what is asked here).
6. The user will be able to filter some nodes and edges of the network according to some of their metrics (e.g. degree) or attributes (e.g. edge weight, time, geographic location).

User Interface

While the user interface (the placement of views, buttons, ...) is left up to you, be careful that it is an integral aspect in the design of the application.

3 User interactive aspects

In addition to the features detailed in Section 2, you will also focus on user interactive aspects while designing your software. We expect you to implement a user interface enabling navigation through the network for exploratory analysis. For instance, the Fisheye distortion enables focusing on parts of the network when moving the mouse on the screen. The document `lecture_`

`aalto.pdf`, introduced in Section 5, quickly overviews this distortion method and provides a link toward a demo. Other ideas include enabling the user to zoom in the network, to drag some nodes to change their positions using the mouse, to highlight a node selected in one view on all the multiple views of a network, to depict some node label or edge weight information when selecting the corresponding node or edge with the mouse, etc. In short, all types of user interaction aiming at facilitating network visual exploration are welcome. Any other, possibly original, feature that you could imagine may be interesting, even if it is not listed above.

4 Report

In addition to your software, you are asked to provide a small report (maximum 7 pages) detailing the features that you have implemented. In particular,

- You can write your report as a user guide for your software.
- Explain and justify your design choices.
- Cite the toolboxes, sources and papers that you employed. There is no restriction on the sources you use nor on the paper that you read, but you have to cite them.
- Reasonably detail the layout algorithms that you have employed (e.g. by providing an overview of each one of them without the practical implementation details) and justify why you chose them.
- Provide examples on how to use your software, illustrating its capabilities in terms of scaling, interaction and visualization (e.g. show a figure with the highlighted shortest path between two nodes, etc.).
- Give some ideas on how to improve your software. Which features might it be worth implementing in future versions? How could you make your software more scalable?

5 Material

To help you get started, you are provided with the following documents:

lecture_aalto: lecture slides from the course CS-E4840 at Aalto University. They quickly introduce graph visualization and navigation and give a link toward a demo.

mcguffin-2012-simpleNetVis: survey paper introducing simple algorithms for network visualization. Some basic methods are detailed. References toward further reading is provided.

6 Practical information

- **Groups:** You can complete the project alone or by groups of 2 students. Please join a group on Moodle (even if you're alone)
- **Programming language:** you can use the one you prefer (Python, Matlab, R, etc.), but Python is recommended. You can use all the toolboxes, packages, modules, etc., that you find relevant. You can use toolboxes to help you designing the interactive user interface, but you cannot just rely on an already existing interface.

- **Deadline for the project submission on Moodle:** Friday December 18, 15pm. Submit one .zip file per group, containing report and code.
- **Mid-project meet-up:** In order to start on good foundations, each group will have a (15 minute) one-on-one with the teaching assistant on the 6/11/2020 or 13/11/2020, during the exercise session. It will happen either face-to-face or on Teams, depending on circumstances and choice. During this meeting, you should already have :
  - A list of the different layouts and attributes that might be useful for exploratory analysis.
  - A sketch/draft of the user interface. This does *not* mean an already working implementation. You might for example use pen-and-paper or `https://excalidraw.com/` to draw the layout of your application.
- **Project presentation and discussion:** right after the exam. Each student will have a meeting with the teaching assistant during which you will first present the software and show the features that have been implemented by realizing a small demo. After this presentation, a short discussion will take place with the teaching assistant, who will ask a few questions on the software and during which you will explain how you could further improve it.
- Do not hesitate to ask questions to the teaching assistant before or after your planned one-on-one, for instance to define what you plan to implement, the network metrics that you could evaluate, etc.