CS490-VIZ - Fall 2020  
Programming Assignment 3: Networks

**Key Dates**

|  |  |  |
| --- | --- | --- |
| **Handed out:** | May 8, 2023 |  |
| Due date: | May 14, 2023 *(before 11:59 PM)* |  |

**Extra Credit Option**

[**Task 5**](https://www.cs.purdue.edu/CS49000-VIZ/projects/assignment3.html#filtering)**is optional and counts toward extra credit**, namely 30% of the value of this assignment or 4% of the semester grade.

**Objectives**

This third programming assignment is concerned with networks (aka graphs) and the challenges posed by their effective visualization when they are large (or dense).

**Context**

The dataset you will use for this project (known as *US Air 97*) contains the list of all US Air flights in 1997 as a undirected graph linking airports connected by US Air at the time. The individual airports are associated with 2D coordinates ("**posx**" and "**posy**"), the name of the corresponding city, state, and country, as well as the associated geospatial coordinates. In addition, each edge is associated with a weight that represents the frequency of the flights between the two cities. Though not huge, the dataset is fairly large from a visualization standpoint with 332 nodes (cities) and 4252 edges. The goal of this project is to allow you to compare several basic graph visualization strategies and to once again see the role that interactivity can play in managing the complexity of the data.

**Tasks**

**Task 1: Basic Graph Visualization (25%)**

For the first task, you will be drawing the graph contained in the dataset as nodes and edges. The main challenge in graph drawing is to create a spatial layout that offers a clear view of the connectivity. Instead you will use the graph visualization library of your choice to create a first layout, thereby selecting one that you find effective. In Python, a plethora of packages exist for that purpose ([networkx](https://networkx.github.io/), [Cytoscape](http://www.cytoscape.org/), [Graphviz](https://www.graphviz.org/), etc...) and the layout they produce can be easily imported in a visualization software like Bokeh or Plotly. In a separate visualization, you will represent the graph using the layout corresponding to the (**posx**, **posy**) coordinates associated with each city in the input dataset.

Compare the results that you obtained in each case. What are the pros and cons of the layout you obtained with and without using the provided coordinates? Which one do you find superior and why?

**Deliverables:** p3\_force\_layout.py, p3\_xy\_layout.py, and answer to questions.

**Task 2: Geospatial Visualization (25%)**

Given the geographic nature of the nodes in this particular dataset, it is quite natural to try and visualize the graph in its spatial context, namely by mapping each vertex to its geospatial location and showing the graph overlaid on a map of the United States. For this second task, I am asking you to map each city / airport in the dataset to its geospatial coordinates (included in the dataset) and to display the corresponding node / vertex at the matching location on a map. Using this layout, you will visualize the graph and assign to each node a color corresponding to its time zone. A handy package for that is found [here](https://pypi.org/project/us/). Note that certain states stretch across multiple time zones. In that case, you are free to assign to the corresponding airport the time zone of your choice among the possible choices. Your map of the United States should clearly delineate the border of each state, as shown [here](http://docs.bokeh.org/en/0.11.0/docs/gallery/choropleth.html). The basic idea consists in using the information contained in bokeh.sampledata.us\_states.

Compare the results you obtained in this task with the ones from the previous task.

**Deliverable:** p3\_geospatial.py, and answer to question above.

**Task 3: Encoding Additional Attributes (25%)**

The visualizations created so far were mostly limited to the connectivity of the graph and did not represent additional pieces of information available in the dataset. In particular, the weights associated with the edges were not visualized, and neither was the significance of each airport (see possible definitions below).

In this third task, I am asking you to encode the weight of each edge (which corresponds to the frequency of a given flight connection) using both thickness (or width) and color. In addition, you will compute an importance factor for each airport and map that value to the size of the corresponding node representation in your graph visualization. Here, the importance factor will be measured as the sum of the weights of all the edges incident to a given vertex. Note that for the color coding of the edges, you will simply vary the saturation (or opacity) of an edge to represent the associated frequency, whereby higher frequencies should correspond to higher saturation (purer color). Your visualization must include a legend showing the meaning of node sizes (importance factor), edge width and color (frequency). The nodes will be color-coded according to their associated time zone, as in the previous task.

Note that you are free to use for this question the layout that you deemed most effective in the previous tasks.

**Deliverable:** p3\_attributes.py.

**Task 4: Tooltip (25%)**

Similar to what you did for the [second programming assignment](https://www.cs.purdue.edu/CS49000-VIZ/projects/assignment2.html), you will enhance your visualization by providing a tooltip that will display for each vertex the name of the airport / city, the US state, the number of connections (i.e., the degree of the node), and the overall frequency of the connections to and from that airport. In addition, your tooltip should highlight the edges connected to the airport that is currently in focus.

**Deliverable:** p3\_tooltip.py.

**Task 5: Interactive Filtering (OPTIONAL: 30% Extra Credit!)**

Your visualizations so far will have made clear that the number of nodes and edges makes it difficult to perceive the details of the dataset. To remedy this problem, you will create two sliders that will control filters on nodes and edges, respectively. Specifically, you first slider will control the minimum importance factor for a city to be shown in the visualization. Similarly, the second slider will control the minimum frequency/weight for an edge to be visualized. Note that both filtering mechanisms must be kept consistent: the edges associated with vertices that have been filtered out by the first sliders should not be shown, regardless of their frequency. In addition, the nodes whose edges all have weight below the frequency threshold (second slider) should not be displayed (since they will have no incident edges).

Comment on the benefits and limitations of the filtering options introduced in this task. What additional control mechanisms would you wish to have to improve the visualization of this dataset?

**Deliverable:** p3\_filtering.py.

**Data Set**

The dataset is available in json format [here](https://www.cs.purdue.edu/CS49000-VIZ/projects/data/project3/USAir97v2.json).

**Submission**

Submit your solution for this project on *Brightspace* before October 10, 2020 at 11:59 pm. Refer to the instructions below.

* Include all the requested python files: p3\_force\_layout.py, p3\_xy\_layout.py, p3\_geospatial.py, p3\_attributes.py, p3\_tooltip.py.
* Include your answers to the questions asked in Tasks 1 and 2 in a separate text document (.doc, .pdf, .pages, .rtf, .txt, .md, etc...)
* Include p3\_filtering.py as well the answer to the corresponding question if you choose to complete the optional task 5.
* Include all submitted files in a single directory named **<myLogin>\_p3**, where **<myLogin>** is your Purdue login.
* Do not use absolute paths in your code.