INFO 3501/5501: Lab Assignment #2

Due on Wednesday, October 19, 2016 at 9:00am

Professor Brian Keegan
Department of Information Science
College of Media, Communication, and Information
University of Colorado, Boulder

Background & Objectives

Multiple kinds of relationships are encoded within Wikipedia data. Lab Assignment #2 will explore different approaches to identify and quantitatively analyze these data.

1. **Retrieving data via the MediaWiki API**. The replication databases in PAWS provide a lot of functionality, but the MediaWiki API is also extremely powerful (if somewhat slower). Students will use functions to retrieve the content of a Wikipedia page.
2. **Identifying relational data in Wikipedia**. Wikipedia collects multiple kinds of relationships among pages and users. Students will use functions to parse the HTML of a Wikipedia page to retrieve the hyperlinks pointing to neighboring pages.
3. **Analyzing the structure of complex networks**. The Python libraries `networkx` and `igraph` implement some powerful data structures and algorithms for describing complex networks. Students will convert relational data about page-page hyperlinks into graph objects and analyze their structure.
4. **Visualizing the structure of complex networks**. The `Gephi` software package is a powerful tool for visualizing large-scale complex networks. Students will compare graph layout algorithms, visualize node attributes, and validate numeric measures of graph structure with visual features.

Instructions

We will be be using Jupyter Notebooks running on the Wikimedia Foundation's PAWS infrastructure to retrieve public data from replica MySQL databases containing the complete history of revisions to articles. We will use the account you created in Lab 1, install a new library called `networkx` to enable us to model and analyze complex networks, and install a program called `Gephi` to visualize the complex networks.

Step 1: Install `networkx` and `igraph` in your PAWS account

1.a Log in to your PAWS account at https://paws.wmflabs.org/. Click the green "My Server" button to access your PAWS Home page.

1.b In the upper right hand corner of the Home page, click the "New ▼" button and select "Terminal".

1.c A new tab/window should launch containing a black terminal window. Enter the command:
`pip install networkx python-igraph`
then press enter. A series of status commands and download progress bar should automatically appear to install both the `networkx` and `igraph` libraries.

1.d Confirm that both libraries were successfully installed if the status "Successfully installed networkx" or "Requirement already satisfied" is printed. If you receive an error message, please contact me.

1.e Close the tab/window containing the terminal window and return to your PAWS Home page.

Step 2: Uploading a Jupyter Notebook

2.a In a different browser window or tab, download the Jupyter Notebook posted to D2L to your desktop. Note that you can also view other notebooks I will be developing for the course in http://paws-public.wmflabs.org/paws-public/User:Cuinfostudents/ but downloading the files from here apparently will not work to upload to your PAWS account.

2.b Return to the browser tab having your empty "My Server". Click the "Upload" button in the upper right.

2.c In the system pop-up box, navigate to where you saved the downloaded "Lab 1 - Revision Histories.ipynb" file on your desktop. Select the file and click "Open".

2.d Click the blue upload button next to the file name to upload it to the PAWS server.

2.e Confirm that you can access the notebook by clicking on its title in the Home directory, which should launch a new tab/window labeled "Lab 2 - Networks" and show the contents of the notebook.

2.f Execute the the cell block underneath "Import modules and setup environment" containing the Python code `import networkx as nx` runs without rror. If you receive an error message like "networkx not found", please contact me immediately.

Step 3: Visualizing networks with Gephi

3.a Go to `https://gephi.org/`, download the latest version (0.9.1) of Gephi, and complete the installation of the package. Download the `.graphml` file generated by your notebook in PAWS. Launch Gephi and open the file. A window called "Import report" should appear showing the number of nodes and edges. Click OK after confirming that these are the same numbers as reported in your notebook.

3.b Confirm that you are in the "Overview" workspace by clicking the button at top left. A mess of black circles and lines should appear. Layout the graph by going to the "Layout" sub-window in the lower left. Click the "—Choose a layout" dropdown and select "ForceAtlas 2". Change the "Scaling" parameter to 250 and select the "Stronger Gravity" and "Prevent Overlap" options. Click "Run", wait for the nodes to stabilize, then click "Stop". Resize the window by clicking the magnifying glass (hover label is "Center on Graph" in the lower left. Try other combinations of parameters or layout algorithms.

3.c Scale the node sizes by going to the "Appearance" sub-window in the upper right. Select "Nodes" and then the button with the three circles, click the "Attribute" button, select "Out-Degree" from the "—Choose an attribute" dropdown, and click "Apply". Experiment with different scaling approaches by clicking "Spline...", selecting one of the templates on the left in the "Interpolate" window, closing the window, and clicking "Apply" again.

3.d Color the nodes by sub-communty. In the "Statistics" sub-window on the right and click the "Run" button next to Modularity. In the "Modularity settings" window that pops up, keep the defaults, click "OK". Note the Modularity score in the "Modularity Report" window that appears; values closer to 1 indicate sub-communities are better-defined and values closer to 0 suggest sub-communities lack any clear boundaries. Close the "Modularity Report". In the "Appearance" sub-window, select "Nodes", the paint palette, and then "Attribute". Under the "—Choose an attribute" drop-down, select "Modularity Class", and click "Apply."

3.e Include the node labels on the graph by selecting the solid black "T" at the bottom of the "Graph" sub-window. You can either scale the label sizes manually with the slider bar at far right or automatically by the node size by selecting "Node size" under the black "A" drop-down at the bottom of the "Graph" sub-window.

3.f Save the visualization by clicking on the "Preview" workspace at top. Click the "Refresh" button on the bottom left in the "Preview Settings" sub-window. Enable the "Show Labels" option and change the Font size to something less than 10. Save the figure to disk by clicking the "SVG/PDF/PNG" button in the lower left of the "Preview Settings" sub-window, change the File Format to PNG, entering a file name, and clicking "Save".

Additional Resources

Here are some links to other resources that provide more details if you are curious and want to learn more.

networkx Official documentation for the `networkx` library: https://networkx.github.io/

Gephi Official documentation for the `Gephi` library: https://www.gephi.org

MediaWiki API Official documentation for the MediaWiki API: https://www.mediawiki.org/wiki/API:Main_page

API Sandbox Sandbox for experimenting with API calls on Wikipedia: https://en.wikipedia.org/wiki/Special:ApiSandbox

Problem 1

Identify an that you believe has interesting connections to other articles. Why did you choose this article? What kinds of articles would you expect to link from this article and why? How many other pages do you expect your initial article links out to? What would the other articles linked from the page you chose tell you about the topic? What percentage of the other articles linked from the page you chose do you think link back? To what extent do these other articles also link to each other? Write down more precise hypotheses about these questions with some justification either from work we have read in class or you knowledge about the topic.

Problem 2

Run all the code in the notebook until the `hyperlink_communities_(page title).graphml` file is created at the end. Load this file up into Gephi and visualize your graph following Step 3 in the Instructions (above). Include one PNG file following all the instructions in Step 3. Experiment with different graph layout algorithms (Fruchterman-Rheingold, Yifan Hu, *etc.*), size the nodes by different attributes or spline interpolations, and apply different coloring schemes. Include a second PNG file showing the results of your customizations. Discuss the strengths and weaknesses of each visualization: which does a better job identifying key nodes, separating sub-communities more clearly, making the data easy to interpret, *etc.*

Problem 3

Based on the descriptive statistics about the graph you computed, compare the observed values against your hypotheses. Are there more or fewer nodes and connections in the network than you expected? Is the density and reciprocity of the network higher or lower than you expected? Discuss why the observed network is or is not different from your hypotheses above.

Problem 4

Play the "Wikipedia Game" by trying to navigate using only hyperlinks in the body from one article to the next. How many steps does it take you? Play the game a few times with different articles. What was your strategy for choosing what links to click to get to your destination? The shortest path in a network is the shortest number of connections between two different nodes. How many pages in your hyperlink network had a shortest path to another page greater than 5 steps? If you increase the `path_length_threshold` value up from greater than 5,

Problem 4 continued on next page…

are there any longer paths? What is an example of a pair of nodes having an extremely long path length between them? As you look at different combinations of longest shortest paths, does there appear to be a common strategy about what kinds of pages to navigate towards and away from?

Problem 5

What are the most well-connected nodes in the network? Among these top articles, are there any common factors (topic, content, user behavior, *etc.*) that explain why they are among the most-connected article combinations? How do the top articles by in-degree versus out-degree compare with each other? Among the articles that have only 0 or 1 connections, are there any common factors that explain why they have so few connections? How do the bottom articles by in-degree and out-degree compare with each other? What is an example of an article having anomalously low number of connects? What are examples of other articles that you would recommend connecting together and why?

Problem 6

In Gephi, there should be a list of different node attributes corresponding to the different community detection algorithms (leading eigenvector, fastgreedy, infomap, *etc.*). Color the graph by the output of each of these different algorithms. Make sure that the coloring scheme is categorical rather than continuous: there should be different colored boxes with numbers next to them, rather than a single colorbar. Switch between coloring modes by clicking the paint palette button in the lower-left corner of the "Appearance" sub-window. What kinds of page sub-communities do these algorithms consistently uncover? What kinds page sub-communities do these algorithms disagree about membership? What algorithm comes closest to your interpretation of the different sub-communities within the graph and why? Include a PNG image of the network with the best layout and community detection and explain its important features in your writeup. "File > Save As" your workspace containing your best graph coloring and layout as "Lab_2.gephi" and attach it with your submission.

Acknowledgements

I want to thank the Wikimedia Foundation for the PAWS system and related Wikitech infrastructure that this workbook runs within. Yuvi Panda, Aaron Halfaker, Jonathan Morgan, and Dario Taraborelli have all provided crucial support and feedback.

Licensing