Assignment 1

mkai & sljo

September 13, 2011

This is the first mandatory assignment for the 02815. Tasks 2–6 are based on the Facebook visualization reproduced in Figure 1. You are encouraged to read about the figure in the accompanying blog post,

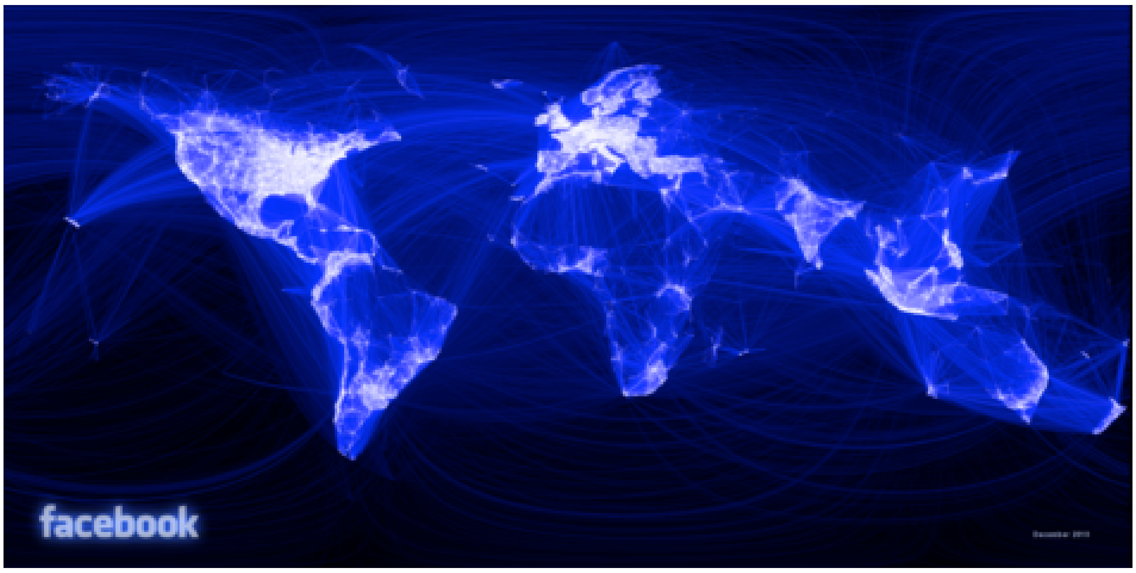

Figure 1: The inspiration for today's exercises.

which you can find below.

`http://www.facebook.com/note.php?note_id=469716398919`

Whereas the Facebook graph focuses on the whole network, we will focus on the networks surrounding individuals. The individual tasks are as follow.

1. *Design Patterns.* Identify web 2.0 architectural design patterns used by *Zynga* and *LinkedIn*—briefly analyze the potential in the their web 2.0 concepts based on what design patterns reinforce each other and how they utilize the value of network links formed between users
2. *Networks.* Using inspiration from MTSW (Chapter 4), download the network surrounding at least two prominent twitter users; the networks should include connections between the central user's friends and followers. By prominent users, we mean users with more than 2000 followers and with fewer friends than followers. The users should be located on different continents (e.g. one in Europe and one in the US). Consider choosing users from different domains? E.g. actor vs. politician or organization vs. individual (or all types, if you're feeling ambitious).
3. *Geocoding.* It is possible to add geolocation to individual tweets[1]. Since this option is opt-in,

[1]See MTSW pp 101f: if the `geo_enabled` is set to `true`, then individual tweets can have a non-`null` value for the `geo` key.

however, very few users have it turned on and for that reason, we cannot use Twitter's services for the purpose of placing people in geographical space.

Instead, we will use the `location` key to determine user's whereabouts (see MTSW p 101 for an example). The `location` key is a text string that each user specifies and can be set to anything in particular (so 'New York, NY' could be 'NY', 'Big Apple', 'The Five Boroughs', etc).

To deal with such irregularities without thinking too much, we will use the fact that someone has already solved this problem and use an API to extract latitude and longitude based on strings. The various map-APIs have rate-limits. *Bing* Maps currently have the most generous rate limits, so we'll use their API. (Note, we will explain the basics of *Bing*'s map API during the lecture).

Due to the rate limits you'll want to minimize the number of queries, thus you should do the following.

- Normalize queries by converting all strings to lower case
- Store your queries in a database (we suggest *Redis*) that maps a (normalized) location string to a lat,long.
- Before you ask an api for a location, make sure that it is not already in your database.

4. *Mapping I.* To get a flavor for a simple way to think about static maps, we'll first plot the user's 20 most important connections using *Google*'s static map API (Note, we will explain the basics of *Google*'s static map API during the lecture). You decide how to define important connections, but use what you've learned about network theory to explain your answer. It is ok to use *NetworkX* functions as part of your answer.

5. *Mapping II.* Now we're ready to connect to the Facebook visualization. In order to draw maps from python, we'll use `matplotlib`'s `basemap` toolkit. This module is part of the *Enthought* distribution, people using other distributions can download it here

   `http://matplotlib.sourceforge.net/basemap/doc/html/`.

   We will explain the basics of using `basemap` during the lecture, but there's lots of good information on line[2].

   We would like you to plot the networks surrounding the two or more users you've picked on the world map. Choose a different color for the links corresponding to each user. There are bonus points if your map looks great. How are your ego-networks different from the full network arising from the Facebook visualization?

6. *Understanding the maps.* Let's think about what we can learn from this exercise. We now have a new property for links: *distance*[3]. Using tools from network theory, consider if long-range links are different from short-range links.

   - Calculate the average distance of links for each of your ego-networks.
   - Calculate the average distance of 2000 users, selected at random from the public timeline (see MTSW p 145 for a script).
   - Compare the three results and explain the differences.
   - What is the average degree of nodes on the other end of links that connect nodes more than 1000 km apart.
   - What is the average degree if the distance is less than 200 km?
   - Explain the difference.

[2]See, for example `http://sciblogs.co.nz/seeing-data/2011/08/12/plottinggeographic-data-ona--world-map-with-python/`

[3]See `http://en.wikipedia.org/wiki/Haversine_formula`

Formalia

- The report is due September 20th, 2011.
- The report should describe your work to solve the various exercises, and may include figures, tables (e.g. illustrating database setup), as well as visualizations (including intermediary plots, etc).
- The report should be formatted according the standard *ACM SIG Proceedings Template*[4] template and take up 4 pages, including graphics, with a section per question.
- You should form groups of 2 individuals to write the reports. Groups of 3 will be allowed, but must write 6 pages. Authorship should be clearly marked.

[4]`http://www.acm.org/sigs/publications/proceedings-templates`