02815 Web 2.0 and mobile interaction E11 - Assignment 1

Corneliu Sugar
Technical University of Denmark
s111023@student.dtu.dk

Søren Ulrikkeholm
Technical University of Denmark
s112460@student.dtu.dk

ABSTRACT
(SU) This report covers the work that was performed for solving assignment 1 in the course Web 2.0 and mobile interaction. The report includes an analysis of which web 2.0 architectures have been applied by Zynga and LinkedIn, how they are profiting and how they could improve their services. Furthermore, the report describes the different techniques used for mining and representing data from Twitter in interesting ways, and also attempts to make some analysis based on the data.

General Terms
Data Mining

Keywords
Web 2.0, Twitter, Python, Basemap, NetworkX

1. DESIGN PATTERNS

1.1 Zynga

1.2 LinkedIn (SU)
The professional social network extensively uses declarative living and tag gardening as a pattern for offering a service of great relevance to their users. Its users provide vast amounts of information about themselves and their professional lives which LinkedIn in turn uses to suggest people that could be part of a personÕs network, and display other types of content that could be of interest to the user. In this way LinkedIn utilizes its community to let users both connect with friends and establish new professional contacts, and also provides to possibility for users to build conversations either via personal messages of status updates.

In addition to creating an interesting service, LinkedIn also uses the information it mines to monetize from their product by selling advertisements to recruiters and companies interested in hiring specific types of people.

LinkedIn could, and perhaps already is, using the concept of curating identity to provide a more vertical experience in which professional and academic interests can be explored and improved.

2. NETWORKS (SU)
As the original task, that asks that two users with each 2000 followers, proved to be too much to handle for the Twitter API, the task was re-formulated to use users with approximately 200 friends and followers each.

To make the results a little more relevant we chose to use Søren Ulrikkeholm (@sorenu, 114 friends, 25 followers) and an African male with approximately the same stats, i.e. age, interests (IT), and number of friends and followers. We were able to find a Twitter user with the screen name @vincente_shiala who fit the description.

The first step in collecting information about each user is to make a script that collects all the friend and follower ids of a user, and then goes on to collect all of the friend ids of each of those users - note that we get the friend ids only in order to limit the number of requests we have to make later. Then we collect detailed information for each of the friends and followers. We are particularly interested in the location information which we will be using later.

The results are stored in a Redis database, so that we do not have to repeat the queries each time we want to use the data.

Most of the Twitter and Redis logic is given to us through the twitter_util module from MTSW.

The data that we collect with the script can be represented in graphs by using the networkx Python library as shown in figure **??**. It is interesting how both users' connections seem to be highly clustered. Especially @vincente_shiala's connection are very distinctly divided into to clusters, with @vincente_shiala acting as the global bridge between the two worlds.

3. GEOCODING (SU)
The next step in the process of representing Twitter users on a map is to collect geographical data about the location of each user in our database.

To do this we firstly need some extra information about the

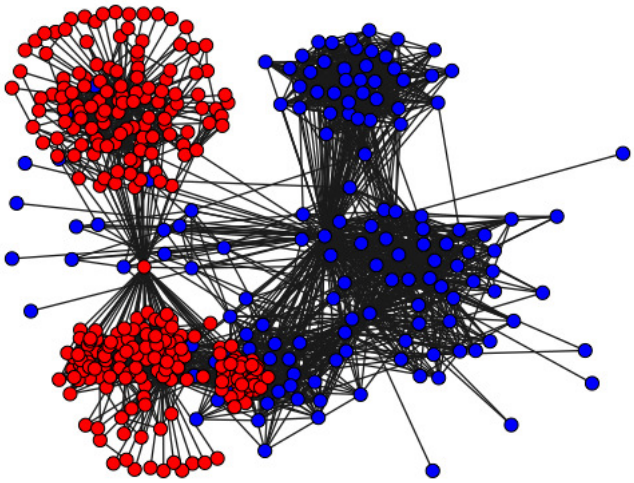

Figure 1: A graph representing two twitter users' connections and how the connections are connected. Red is @vincente_shiala - blue is @sorenu. NB: the red and blue graphs are not actually connected, but it looks cool when they are presented like this.

users in our database in addition to their friend and follower ids. The getUserInfo method in the twitter__util module lets us extract a vast amount of data, including location, for a list of user ids that we provide for it. Using this method we harvest location data for all of the users we are interested in, i.e. the friends and followers of our main user, and the friends of all of those users.

Now the locations need to be converted into longitudes and latitudes. The Bing Maps API allows you to query a location and have its coordinates returned along with a parameter stating the certainty of the result. In our script we only store the results with high certainty in our Redis database using the location name as the key. To decrease the necessary number of queries, we normalize the locations before querying by turning them all into lower case and checking the database if the location already exists before querying.

Since Twitter users tend to state strange locations it's important that the script is able handle bad responses from the Bing service.

4. MAPPING I (SU)

As Twitter is a service very much about reaching the most amount of people, we decided to define "most important contacts" as the contacts that have the highest amount of followers.

To extract the top 20 users among the main user's friends and followers we use Redis' built-in set operations to do a union of the friends and followers which we turn into a dictionary with user_id as the key and the number of followers as the value. We then turn the dictionary into a sorted list based on the values and extract the first twenty entries.

We then want to create a static Google map with the connections between the main user and his top twenty connections. To do this, we make use of the location and position data that we collected in the previous task. The Google maps API lets us draw paths on maps by calling the API with a URL containing an arbitrary number of path parameters. We create this URL based on the location and position information.

When we open the URL we get the result as seen in figure **??**.

Figure 2: @sorenu's top twenty connections.

Figure 3: The latest tweet from @vincente_shiala.