CSC 591/791 - Social Computing Spring 2015
Assignment 1
Social Analytics using Graph Databases

Part1: 11:45 PM, September 18, 2015

Part 2: 11:45 PM, September 26, 2015

Social Analytics involves **A**nalyzing, **I**nterpreting and **M**easuring interactions and associations between people, topics and ideas. The first assignment will introduce you to the basics of formal approaches in social network analysis. The assignment has been divided into two parts: -

- **Part 1:** Work interactively with the class and identify interesting characteristics of a sample dataset.
- **Part 2:** Work individually on the programming task to test your hypotheses.

Part 1a:

In part 1, students are expected to work interactively to hypothesize potential trends across a given sample dataset. Give reasons and make arguments to support or reject a hypothesis. Use the message board for all your discussion. The instructors will monitor and guide the discussion and finally summarize and vote up the best trends that emerge from the discussion. Based on the summary, you will be asked to further discuss and develop evaluation measures to test your hypothesis.

For our discussion, we will use the SNAP Social circles dataset: -

http://snap.stanford.edu/data/egonets-Facebook.html

To begin with, explore the dataset to understand the attributes that have been collected. The above URL also links to a paper for your reference. Once you have an intuition for the nature of data collected, try and reason about how you would characterize the graph and evaluate it using some properties of the social network. As an example of some standard evaluation measures, consider the following sociometric network properties for the sample dataset - density, centrality, reciprocity and number of components. Think about other measures that you would like to calculate for the graph. Try and come up with some interesting properties that you would like to evaluate on the dataset (e.g., Cliquishness monotonically increases with the number of hops in a random walk).

As part of the assignment, you will also be introduced to concepts of argumentation theory in practice. To this end, we have come up with a text-based argumentation scheme for use on our message board. For simplicity of maintenance, we request you kindly adhere to the rules given below while posting to the message board.

Argumentation Rules:

- Every new hypothesis will have a uniqueID of the form ***<unityID>-<serial number>***.
    - The *unityID* uniquely identifies the student while the serial number provides a numbering scheme to differentiate hypotheses made by a student. Remember to increment the serial number for every new hypothesis you state.
- All hypotheses **must** have a justification. To mention your justification simply prepend your statement with the text ***<Justification>***.
- A new post must be created in the **P1_hypothesis** folder for a new hypothesis. All discussions for the hypothesis must take place as replies within that post. The only exception is when ***<Replace>*** is being used.

1) To propose an original hypothesis not previously considered, prepend the unique ID with the text ***<Propose>***

- *e.g.,* ***<Propose> uniqueID***

2) To suggest an alternative hypothesis (uniqueID2) to an existing hypothesis (uniqueID1), prepend uniqueID1 with the text ***<Replace>*** and append the new uniqueID (uniqueID2)***.***

- *e.g.,* ***<Replace> uniqueID1 uniqueID2***

3) To oppose a hypothesis (uniqueID1) in support of another student's hypothesis (uniqueID2), prepend uniqueID1 with ***<Oppose>*** and uniqueID2 with ***<Support>***

- *e.g.,* ***<Oppose> uniqueID1 <Support> uniqueID2***

4) If you would like to question the validity or the claim made by a hypothesis without proposing a new hypothesis, prepend the uniqueID of the hypothesis with
<Raise>

- *e.g.,* ***<Raise> uniqueID***

5) To increase the weight of a question/suggestion raised with **<Raise> uniqueID**, use a simple **<+1>** followed by ***<Raise> <uniqueID>***

- *e.g.,* **<+1>** ***<Raise> uniqueID***

6) Finally, use a new line to start a statement that includes any of the above tags.

Part 1b:

Once the deadline (TBD) for Part 1a has passed, no further discussions will take place. Everyone will have one day to vote for their top two hypotheses. Voting will be done through a Survey (to be posted after Part 1a has finished). Everyone will be required to rank the hypotheses (from strongest to weakest) as they see fit.

The top ten hypotheses will be chosen and a final round of voting will take place. The top two hypotheses from this round will be used for **Part 2**.

Part 2:

The second half of the assignment involves programming to evaluate some of the sociometric properties decided upon in Part 1. Please note that this will be an **individual effort** on the part of the student.

You will use the same SNAP Social circles dataset to calculate the following measures: -

- Graph statistics
  - Number of Nodes
  - Number of Edges
- Sociometric network properties
  - Betweenness Centrality
  - Clustering co-efficient

You will also be required to prove/disprove the two hypotheses chosen from **Part 1**.

You may use the Snap.py for building and storing the social network graph. Instructions on how to do this can be found at http://snap.stanford.edu/snappy/index.html.

Another method of doing this is with neo4j. You may use any of the programming language APIs provided with neo4j to carry out this exercise. Instructions on how to use neo4j with Java are detailed below.

Installation of Neo4j Server and Demo

Complete the installation of Neo4j server. Neo4j is available for Windows, Linux and Mac. Once the installation is complete, follow the demo on the localhost to understand and visualize how the Neo4j graph database works. This is essential before starting the project requirements and I would encourage you to complete this for a good learning outcome.

Creation of Graph Database using Neo4j Java API

i) To Setup your JDK, Eclipse IDE and Maven, follow the steps at:
http://docs.neo4j.org/chunked/stable/tutorials-java-embedded-setup.html

ii) Download and setup Neo4j Java API
Create a new Project and configure it using Maven. Enter the dependency in the pom.xml file. Maven will download and build the necessary packages.
Note: You can manually do this if you don't want to use Maven.

iii) Import and load the SNAP database as a neo4j graph
The Neo4j Java API that you just added to the project has the provision to start, stop, add nodes, add relationship between nodes, and add properties, etc. Explore the possible choices and build the dataset into a graph database. Visualize the data using the browser if required.
Follow the documentation at http://neo4j.com/docs/stable/batchinsert.html

Resources:

Tutorials on using SNAP (Stanford Network Analysis Project)

- http://snap.stanford.edu/proj/snap-icwsm/

Neo4j Download

- http://neo4j.com/download

Neo4j API docs

- http://neo4j.com/developer/language-guides/

Neo4j Batch Inserter Examples

- http://neo4j.com/docs/stable/batchinsert-examples.html