Web Mining
Web Structure Mining

In this practical work, you will learn some basics on network data analysis. In Python, the 3 most popular librairies for network analysis are `networkx`, `igraph` (also available in `R`), and `graph-tool`. In this practical work, we will illustrate network analysis using the `networkx` library since it is easy-to-use and offers a broad range of functionalities. However, for efficiency purpose (when the network to analyze has more than 50K nodes), using the `igraph` and `graph-tool` libraries is preferable.

Note that this tutorial is guided step by step. Its objective is not showing you all the functionalities of this package but showing the most useful instead.

The network we will use today is the *Reuters terror news network*. It is based on all stories released during 66 consecutive days by the news agency Reuters concerning the September 11 attack on the U.S., beginning at 9:00 AM EST 9/11/01. The vertices of a network are words (terms); there is an edge between two words iff they appear in the same text unit (sentence). The weight of an edge is its frequency. The network has n = 13332 vertices (different words in the news) and m = 243447 edges, 50859 with value larger than 1. There are no loops in the network.

This network is not a web network *per se*. However, whatever the type of considered network, similar techniques can be applied. The way results are interpreted is obviously related to the network.

The documentation of the package is available at: `http://networkx.readthedocs.org/en/networkx-1.11/`
Important note. In this practical work, you are manipulating non native libraries. To import these libraries, please download the zip file on my

website. Then, unzip it to the `C:\\Python27` directory. To finish, your python script must start with:

```
import sys,os
sys.path.append("./libs")
sys.path.append("./libs/decorator")
import networkx
```

Introduction to the NETWORKX package

Once you have downloaded the data, they need to be read in python. Since the date is in a particular format (`Pajek` format), a particular reading function is needed. Do not forget to import the `networkx` library first.

```
M = nx.read_pajek("days.net")
```

Print some statistics about the graph such as the number of nodes, the number of edges, its density. Then print the set of edges. What observation can you make according to the result of the output? Write a Python function that convert this MultiGraph into a simple undirected graph. Remove the edges having a weight lower than 2 as well as isolated nodes. How many nodes and edges have been removed? Compare the density of this cleaned graph to the original multigraph density. Now the data are clean, we can run some more sophisticated treatments.

On the calculation of node-level indices

Let us now calculate some topological measures on the vertices. Read on the documentation of the package `networkx` and calculate for each node:

- Its degree centrality
- Its closeness centrality
- Its betweeness centrality
- Its current flow closeness centrality
- Its current-flow betweenness centrality
- Its eigenvector centrality
- Its Page Rank

When possible, you will specify the weight parameter to be the weight attribute of edges. For each centrality measure, find the 10 top nodes and compare them. Are they similar?

Tip. The following Python code sorts a dictionary (according to the attribute value) in descending order and prints the 3 top entries.

```
import operator
my_dict = {"a":10,"b":20,"c":1,"d":4,"e":12,"f":3,"g":11}
for n,c in sorted(my_dict.items(), key=operator.itemgetter(1),
    reverse = True)[:3]:
    print "%s %i"%(n,c)
```

On the calculation of graph-level indices

We are now calculating a few graph-level indices:

- The density
- The center
- The diameter
- The eccentricity
- The periphery
- The radius
- The connected components (deduce if the graph is connected)
- The average clustering coefficient
- The k-components
- The cliques
- The degree pearson correlation coefficient

Read on the documentation of the package `networkx` and calculate these measures.

On the discovery of communities within ego networks

It could be interesting to analyze if it exists several communities of words within the network. Apply the k clique community detection technique on the network with several values for k (15 and 20).