Athens University of Economics and Business Dept. of Informatics
M.Sc. Program in Business Analytics
Course: Social Network Analysis Spring 2023
Instructor: Dr Katia Papakonstantinopoulou

Homework 2:
From raw data to temporal graph structure exploration

Due 11:59pm EEST July 8, 2023

General Instructions

Your answers should be as concise as possible.

Submission instructions: You should submit a compressed directory, containing your answers and code, via `https://e-mscba.dmst.aueb.gr`.

Submitting answers: Prepare a report with your answers on this homework in a single PDF file named *p2.pdf*

Submitting code: Prepare the source file(s) with your code.

Problem

1 Twitter mention graph

Your first task is create a weighted directed graph with igraph,[1] using raw data from Twitter. You will download a compressed file with Tweets posted during July 2009. [2] The format of the file is the following:

```
T 2009-07-01 00:04:20
U http://twitter.com/greeneyed_panda
W @Dprinzessin jajajajajajaja.......... no.....
```

In the tweet above, `T` indicates the time the tweet was posted (2009-07-01 00:04:20), `U` indicates the user that posted it (*greeneyed_panda*) and `W` is the

[1]`https://igraph.org/r/`
[2]`https://drive.google.com/open?id=1RjWUg-6KrVOjJPZHHQg-h_9gSSWZUPn-`

text of the tweet (@Dprinzessin jajajajajajaja.......... no.....). Twitter subscribers can use the '@' character to make a mention to another subscriber, e.g., in the above tweet, user *greeneyed_panda* made a mention to user *Dprinzessin*.

You will first manipulate the raw data with the programming language of your choice to create a total of 5 .csv files, one for each of the first five days of July 2009, using the following format:

```
from,to,weight
user1,user2,5
user2,user1,1
user1,user3,2
...
```

Each .csv file should describe the weighted directed mention graph for the respective day, e.g., in the example above *user1* has made 5 mentions to *user2*, *user2* has made 1 mention to *user1*, and *user1* has made 2 mentions to *user3*.

In addition to the above, you will need to identify the most important topic for each user. The most important topic is defined as the mostly-used hashtag (#) for each user, across all his/her tweets. For example, user Dprinzessin, could have 5 hashtags #javascript and 1 hashtag #html. For this user the most important topic is #javascript. In case of similar scores, you may choose the most important topic at random. Bear in mind that many users may not have any hashtags, for such cases the most important topic may remain null/na.

After finding the most important topic for each user, you will create a csv for each day, so again 5 csv files in total, with the following format:

user, topic_of_interest
Dprinzessin, #javascript
greeneyed_panda, #apples

Having created the edgelist .csv files, it will be trivial to use them and create the respective igraph graphs. You will then need to update the graph vertices, so that they also have the topic_of_interest as an attribute.

Your submission should include the code you used to create the .csv files (any programming language), the code you used to create the igraph graphs (R) and the 5 (compressed) .csv files.

2 Average degree over time

Your next task is to create plots that visualize the 5-day evolution of different metrics for the graph. More specifically, you will create plots for:

- Number of vertices
- Number of edges
- Diameter of the graph
- Average in-degree
- Average out-degree

What do you notice for each of the 5 above metrics? Are there significant fluctuations during these five days?

3 Important nodes

Next, you will write to code to create and print data frames for the 5-day evolution of the top-10 Twitter users with regard to:

- In-degree
- Out-degree
- PageRank

Again, provide short comments on your findings. Do you notice variations on the top-10 lists for the different days?

4 Communities

Your final task is to perform community detection on the mention graphs. Try applying fast greedy clustering, infomap clustering, and louvain clustering on the undirected versions of the 5 mention graphs. Are you able to get results with all methods? Include a short comment on your report regarding the performance of the 3 algorithms.

Then, pick one of the three methods as well as a random user that appears in all 5 graphs and write code to detect the evolution of the communities this user belongs to. Do you spot similarities in the communities? Can you find out which were the most important topics of interest? Do the communities share any topics? (Hint: You can use the membership function to populate the vertex attributes.)

Finally, you will create a visualization of the graph using a different color for each community. Make sure to have a look at the sizes of the communities and filter out all nodes that belong to very small or very large communities, in order to create a meaningful and aesthetically pleasing visualization.

5 Hints

Managing large volume data can be troublesome if you do not follow an appropriate approach. Below you can find some tips on:

1. how to process large files, and
2. how to extract the desired information using regular expressions.

Moreover, while developing your solution you could use just a sample of the provided dataset. Such a sample can be easily created using command line utilities. The following example writes the first 10,000 lines of a compressed file named tweets2009-07.txt.gz to a new file named tweets2009-07.txt:

```
$ zcat tweets2009-07.txt.gz | head -n 10000 > tweets2009-07.txt
```

5.1 Parsing large files

You can parse compressed files directly with R. However, you must be careful not to load the entire content of the file in memory Instead, you should focus on reading the file line by line. Given the format of the file used in this project, it would be best if you process four lines at a time, as this is the space that each tweet takes in the dataset provided.

Below you can find code written in R that parses four lines at a time and examines whether a tweet was written at a specific date:

```
inputFile <- "tweets2009-07.txt.gz"
con  <- file(inputFile, open = "r")
while (length(fourLines <- readLines(con, n = 4, warn = FALSE)) > 0) {
    if (grepl("T\t2009-07-03", fourLines[2], fixed=TRUE)) {
        ...
    }
  }
close(con)
```

5.2 Using regular expressions

You can extract the desired information from a string using an appropriate regular expression. In the example written in R below you can see how we can extract the mentions present in a random tweet:

```
regex1 <- "(^|[^@\\w])@(\\w{1,15})\\b" # get strings with @
regex2 <- "[^[:alnum:]@_]"  # remove all punctuation except _ and @

tweet <- "Open graph benchmark (OGB) initiative by
```

```
Jure Leskovec @jure seems a great source for researchers
working on graph learning and graph neural networks.
Graph representation learning workshop at #NeurIPS2019 #neurips19
pip install is also available. Thanks Jure!"
tweet <- unlist(strsplit(tweet, " "))
users <- gsub(regex2, "", tweet[grep(regex1, tweet, perl = T)])
print(users)
```