Introduction to Web Science

Assignment 9

Prof. Dr. Steffen Staab
staab@uni-koblenz.de

René Pickhardt
rpickhardt@uni-koblenz.de

Korok Sengupta
koroksengupta@uni-koblenz.de

Olga Zagovora
zagovora@uni-koblenz.de

Institute of Web Science and Technologies
Department of Computer Science
University of Koblenz-Landau

Submission until: January 18, 2016, 10:00 a.m.
Tutorial on: January 20, 2016, 12:00 p.m.

Please look at the lessons 1) **Similarity of Text** & 2) **Generative Models**

For all the assignment questions that require you to write scripts, make sure to **include the scripts in the answer sheet, along with a separate python file.** Where screen shots are required, please add them in the answers directly and not as separate files.

Team Name: Echo

1 Generative models (abstract) (10 points)

In the lecture you learned about 6 potential parts you could find in research paper abstracts. Consider the following research paper abstract[1]

> Hit songs, books, and movies are many times more successful than average, suggesting that "the best" alternatives are qualitatively different from "the rest"; yet experts routinely fail to predict which products will succeed. We investigated this paradox experimentally, by creating an artificial "music market" in which 14,341 participants downloaded previously unknown songs either with or without knowledge of previous participants' choices. Increasing the strength of social influence increased both inequality and unpredictability of success. Success was also only partly determined by quality: The best songs rarely did poorly, and the worst rarely did well, but any other result was possible.

1. Name the 6 potential parts you could find in research paper abstracts.
2. Mark all parts you can find in the given abstract.

Answer
Potential parts you could find in research paper abstracts:

1. The background and the problem that is tackled in the research
2. The used methodology in the research
3. One to three precise research question that are answered in the research paper
4. Giving the "Unique idea" or the solution
5. Pointing the results
6. A conclusion with a point of impact

The potential parts marked from the given abstract:

1. The background and the problem:
   Hit songs, books, and movies are many times more successful than average, suggesting that "the best" alternatives are qualitatively different from "the rest"; yet experts routinely fail to predict which products will succeed.

2. The used methodology:
   We investigated this paradox experimentally, by creating an artificial - music market

[1] https://www.princeton.edu/~mjs3/salganik_dodds_watts06_full.pdf

3. One precise research question:
   14,341 participants downloaded previously unknown songs either with or without knowledge of previous participants' choices.

4. Unique idea:
   Increasing the strength of social influence increased both inequality and unpredictability of success.

5. Results:
   The best songs rarely did poorly, and the worst rarely did well, but any other result was possible.

6. Point of impact:
   Success was also only partly determined by quality

2 Meme spreading model (10 points)

We provide you with the following excerpt from the meme paper[2] which was already discussed at the lecture. This part of the paper contains and explanation of their basic model. Your task is to **list five model choices** that stay in conflict with reality and **discuss the conflict**.

> Our basic model assumes a frozen network of agents. An agent maintains a time-ordered list of posts, each about a specific meme. Multiple posts may be about the same meme. Users pay attention to these memes only. Asynchronously and with uniform probability, each agent can generate a post about a new meme or forward some of the posts from the list, transmitting the cor- responding memes to neighboring agents. Neighbors in turn pay attention to a newly received meme by placing it at the top of their lists. To account for the empirical observation that past behavior affects what memes the user will spread in the future, we include a memory mechanism that allows agents to develop endogenous interests and focus. Finally, we model limited attention by allowing posts to survive in an agent's list or memory only for a finite amount of time. When a post is forgotten, its associated meme become less represented. A meme is forgotten when the last post carrying that meme disappears from the user's list or memory. Note that list and memory work like first-in-first-out rather than priority queues, as proposed in models of bursty human activity. In the context of single-agent behavior, our memory mechanism is reminiscent of the classic Yule-Simon model.
>
> The retweet model we propose is illustrated in Fig. 5. Agents interact on a directed social network of friends/followers. Each user node is equipped with a screen where received memes are recorded, and a memory with records of posted memes. An edge from a friend to a follower indicates that the friend's memes can be read on the follower's screen (#x and #y in Fig. 5(a) appear on the screen in Fig. 5(b)). At each step, an agent is selected randomly to post memes to neighbors. The agent may post about a new meme with probability p_n (#z in Fig. 5(b)). The posted meme immediately appears at the top of the memory. Otherwise, the agent reads posts about existing memes from the screen. Each post may attract the user's attention with probability pr (the user pays attention to #x, #y in Fig. 5(c)). Then the agent either retweets the post (#x in Fig. 5(c)) with probability $1 - p_m$, or tweets about a meme chosen from memory (#v triggered by #y in Fig. 5(c)) with probability p_m. Any post in memory has equal opportunities to be selected, therefore memes that appear more frequently in memory are more likely to be propagated (the memory has two posts about #v in Fig. 5(d)). To model limited user attention, both screen and memory have a finite capacity, which is the time in which a post remains in an agent's screen or memory. For all agents, posts are removed

[2] http://www.nature.com/articles/srep00335

after one time unit, which simulates a unit of real time, corresponding to Nu steps where Nu is the number of agents. If people use the system once weekly on average, the time unit corresponds to a week.

Answer

1. Probability for sharing meme will increase if the meme topic is related to profession/culture of the agent.
   - Determining the subjects of all memes do not have a proper methodology because not all of them has a #hashtag. Even if we consider only #hastag memes, it will not be possible to relate each meme with the profession or culture of the meme. Even determining agent's culture/profession will not always be 100
2. An agent will share a meme of own political interest which is intentionally triggered by political/media entity.
   - This type of meme will have #hashtag as it is intentional. But determining agent's political interest is difficult. So there are some more variables to consider which are not included in the model or the part of the system.
3. Probability of sharing meme of recent phenomena will be higher than older ones.
   - It conflicts with reality because defination of 'recent phenomena' and collection of data of recent phenomena is not defined. Considering the meme is given #hastag.
4. An agent's real time personal incident will have some influence on the agent to share specific memes.
   - System can not track agent's real time personal incident in reality. So we can't model this real time incident.
5. Probability of sharing meme's which is related to the agent is higher than any random meme.
   - 'Related to the agent' can mean many things. Also, if there is not #hastag in the meme it is not possible to be sure about the topic of the meme.

3 Graph and its properties (10 points)

Last week we provided you with a graph of out-links[3] of Simple English Wikipedia which should be reused this week.

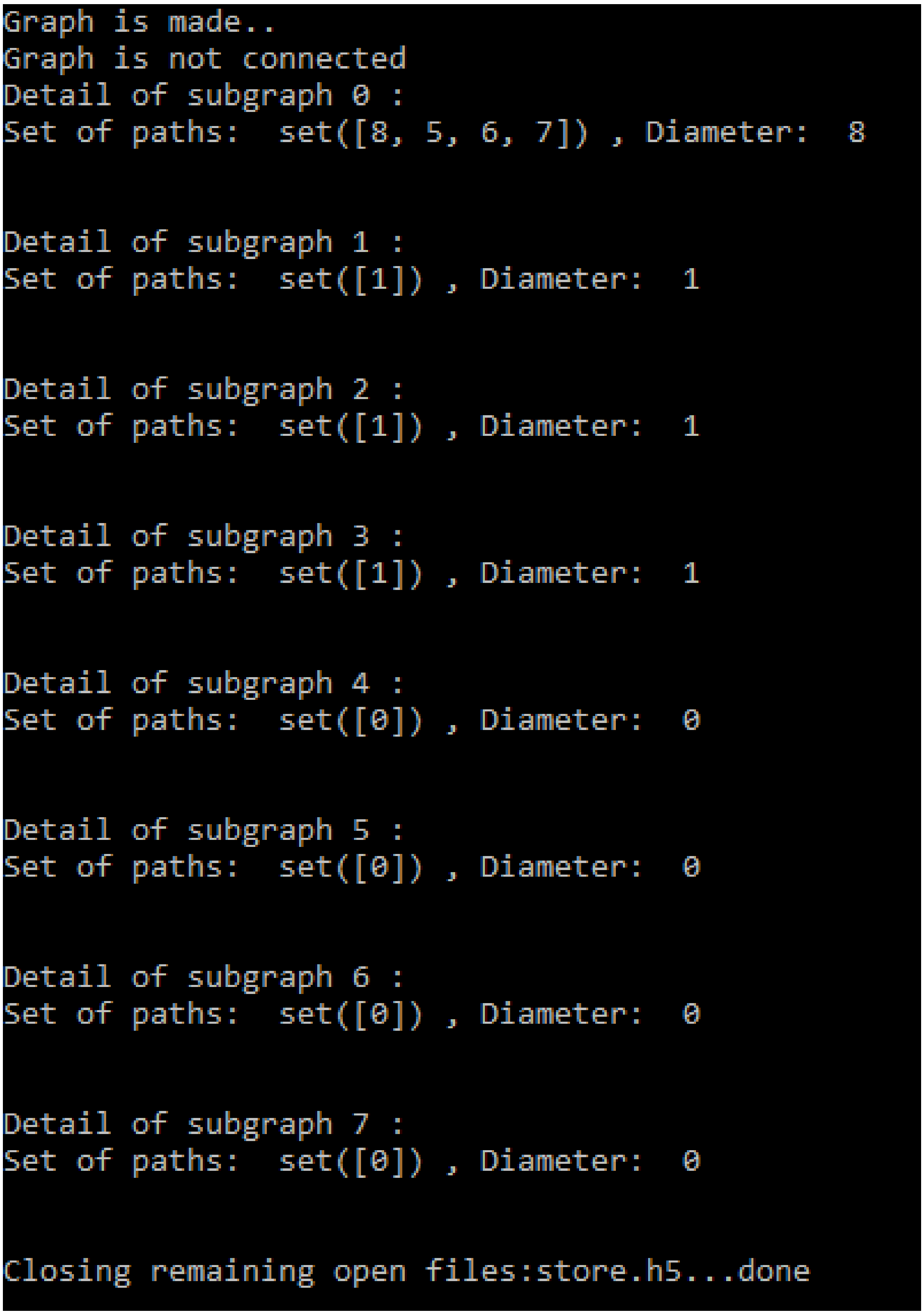
Write a function that returns the diameter of the given directed network. The diameter of a graph is the longest shortest path in the graph.

3.1 Hints

1. You can first write a function that returns the shortest path between nodes and then find the diameter.
2. Do not forget to use proper data structures to avoid a memory shortage.

Answer
There was a run time issue so the loop for nodes in subgraphs are limited to 100 for testing. The code is given bellow and here is the output.

[3] http://141.26.208.82/store.zip



```
Graph is made..
Graph is not connected
Detail of subgraph 0 :
Set of paths:  set([8, 5, 6, 7]) , Diameter:  8


Detail of subgraph 1 :
Set of paths:  set([1]) , Diameter:  1


Detail of subgraph 2 :
Set of paths:  set([1]) , Diameter:  1


Detail of subgraph 3 :
Set of paths:  set([1]) , Diameter:  1


Detail of subgraph 4 :
Set of paths:  set([0]) , Diameter:  0


Detail of subgraph 5 :
Set of paths:  set([0]) , Diameter:  0


Detail of subgraph 6 :
Set of paths:  set([0]) , Diameter:  0


Detail of subgraph 7 :
Set of paths:  set([0]) , Diameter:  0


Closing remaining open files:store.h5...done
```

```
# -*- coding: utf-8 -*-
"""
Created on Sat Jan 14 20:36:13 2017

@author: Hanadi
"""
import pandas as pd
store = pd.HDFStore('store.h5')
df2 = store['df2']

import networkx as nx

#create tuples of edges
def maketuple(x, y):
    r = []
    for i in range(len(y)):
        r.append((x, y[i]))
    return r

#add edges to the graph
def makeGraph(df):
    Graph = nx.Graph()
    for row in df.itertuples():
        Graph.add_edges_from(maketuple(row[1], row[2]))
    print 'Graph is made..'
    return Graph

G = makeGraph(df2)

#find the diameter of each sub graph
def findDiameter(G):
    for index, graph in enumerate(G):
        diameter = set()
        i = 0
        for node in graph:
            lenght = nx.single_source_dijkstra_path_length(graph, node)
            #print 'lenght: ',lenght,'\n'
            diameter.add(lenght[max(lenght, key=lenght.get)])
            #print 'lenght: ',diameter,'\n'
            i = i+1
            #Calculated for 1000 iteration because of huge run time
            if i == 1000:
                break
        print 'Detail of subgraph',index,': \n','Set of paths: ', diameter, ', Di
        #print 'Diameter in subgraph',index,': ', diameter

#if the graph is not connected make subgraphs
if(nx.is_connected(G)):
    print 'Graph is connected'
    findDiameter(G)
else:
    print 'Graph is not connected'
    subgraphs = sorted(nx.connected_component_subgraphs(G), key=len, reverse=True
    findDiameter(subgraphs)
```

Important Notes

Submission

- Solutions have to be checked into the github repository. Use the directory name `groupname/assignment9/` in your group's repository.
- The name of the group and the names of all participating students must be listed on each submission.
- Solution format: all solutions as *one* PDF document. Programming code has to be submitted as Python code to the github repository. Upload *all* `.py` files of your program! Use `UTF-8` as the file encoding. *Other encodings will not be taken into account!*
- Check that your code compiles without errors.
- Make sure your code is formatted to be easy to read.
  - Make sure you code has consistent indentation.
  - Make sure you comment and document your code adequately in English.
  - Choose consistent and intuitive names for your identifiers.
- Do *not* use any accents, spaces or special characters in your filenames.

Acknowledgment

This latex template was created by Lukas Schmelzeisen for the tutorials of ”Web Information Retrieval”.

LaTeX

Currently the code can only be build using LuaLaTeX, so make sure you have that installed. If on Overleaf, there's an error, go to settings and change the LaTeXengine to `LuaLaTeX`.