Project 2

Pranav Menon

April 25, 2024

1 Question

The project deals with the impression network formed between 143 nodes, and we analyse who would be the leader in the case
Choose the top leader by running a random walk on the graph with teleportation.

2 Algorithm

In the algorithm we use the method of random walk on the graph. We need to find out who would be the leader for the graph. For this we carry out the following steps.

- Choose a random node out of the 143 nodes which are given to you. This would be the first node from which we would begin our analysis.
- Now out of all the possible edges choose an edge uniformly at random. Continue on this edge, after dropping a coin on the current edge (incrementing the array by 1).
- If one reaches a node, where there is no out_link then we again choose the node randomly so as to continue on this journey of exploring all the nodes.

3 Code and Implementation:

3.1 Importing libraries

```
import numpy as np
import networkx as nx
import pandas as pd
import math
import matplotlib.pyplot as plt
import random
```

All these libraries will be used for computing the final leader of the impression network.

3.2 Preprocessing and making the graph

```
G = nx.DiGraph()
for index, row in df.iterrows():
row_list = row.tolist()
t = row_list[1]
t = t[:11].lower()
for i in range(2, 31):
    if(not isinstance(row_list[i], str)):
        continue
    x = row_list[i]
    x = x[-11:]
    x = x.lower()
    G.add_edge(t, x)
```

This makes the plot for the graph.

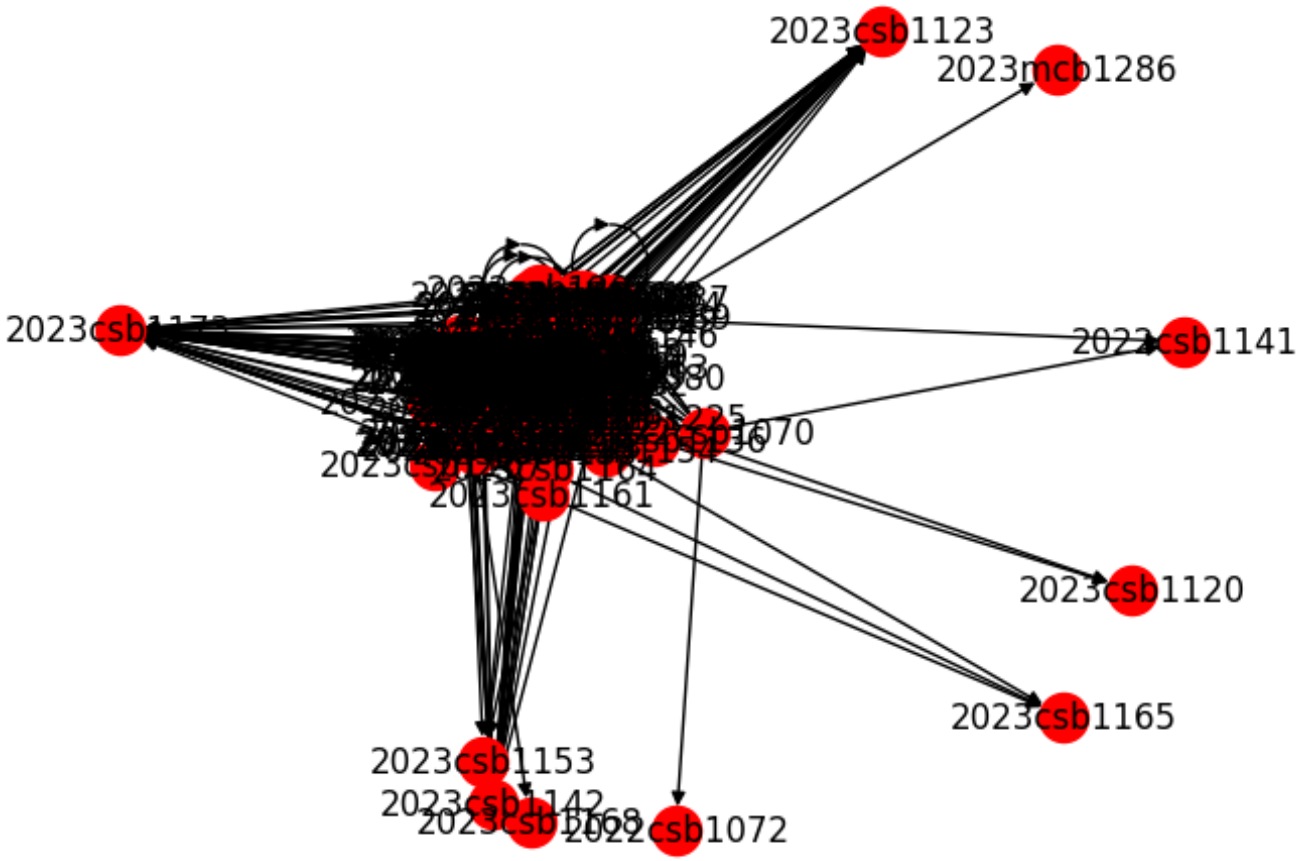

Figure 1: Image for the question.

3.3 Random Walk:

```
for i in range(143):
rand_walk[i] = 0
out_links = G.out_edges(node_to_begin)
out_ed = list(out_links)
for i in range(10000000):
    if not out_ed:
        f += 1
        current_present = random.choice(all_nodes)
        out_links = G.out_edges(current_present)
        for i in range(143):
            if(number[i] == current_present):
                rand_walk[i] = rand_walk[i] + 1
                break
    else:
        current_present_edge = random.choice(out_ed)
        current_present = current_present_edge[1]
        out_links = G.out_edges(current_present)
        out_ed = list(out_links)
        for i in range(143):
            if(number[i] == current_present):
                rand_walk[i] = rand_walk[i] + 1
                break
```

This chooses a node at random and goes to that node and finally increment a variable stored in number.

3.4 Final results

```
max = 0
val = 0
for i in range(131):
    if(rand_walk[i]>val):
        max = i
        val = rand_walk[i]
print(number[max])
```

This prints the output of the code, the one which has maximum value.

4 Result :

The value of nodes after one iteration is:

[12005, 12034, 8063, 12158, 5346, 7703, 7868, 5747, 14109, 7121, 7857, 10609, 6374, 6070, 8786, 8909, 9013, 11410, 11378, 5646, 4436, 4808, 6430, 5868, 11312, 4461, 7626, 6498, 7314, 9210, 10038, 6226, 7261, 7856, 9682, 6410, 9645, 11737, 6955, 5645, 8293, 7841, 10727, 11262, 7842, 4396, 6841, 10300, 9343, 5869, 14308, 7638, 4612, 3637, 3245, 7375, 11039, 8945, 8206, 6318, 7345, 10664, 4083, 4692, 7005, 6481, 9318, 12268, 4648, 3651, 7125, 7935, 7590, 8376, 7048, 7122, 10969, 4030, 11292, 5592, 5222, 7611, 5786, 3674, 5286, 3748, 6226, 8619, 11822, 5365, 6024, 2953, 4095, 4344, 9413, 3834, 3902, 2607, 3206, 4347, 101, 2470, 9171, 4471, 8215, 5136, 9185, 7141, 4049, 7356, 7035, 6374, 6049, 5718, 644, 6662, 10615, 5501, 3880, 4242, 10320, 10172, 10397, 9264, 7202, 6092, 8950, 5172, 8326, 9562, 7967, 6152, 996, 338, 4374, 4365, 2186, 228, 253, 486, 1098, 351, 21] 2023csb1091

The leader is : 2023csb1091