MCS-381: Social Computing (Fall 2023) Coding Assignment 2: PageRank and Link Analysis

Start: Monday, 11/6/2023
Due: Monday, 11/27/2023 by the start of the class
10% grade deduction penalty per day late

In this assignment you will implement the PageRank algorithm and run it on a number of datasets.

You are to do this coding assignment in a team of two.

When you typed some keywords in the Google search engine, a list of Web Pages will be displayed, but how does the search engine know which page is to be shown fist to the user? To solve this problem an algorithm called PageRank was developed at Stanford University by Larry Page and Sergey Brin (whom, according to rumors, used that money to buy a ninja style castle as his house) in 1996. The PageRank algorithm uses probabilistic distribution to calculate the rank of a Web Page.

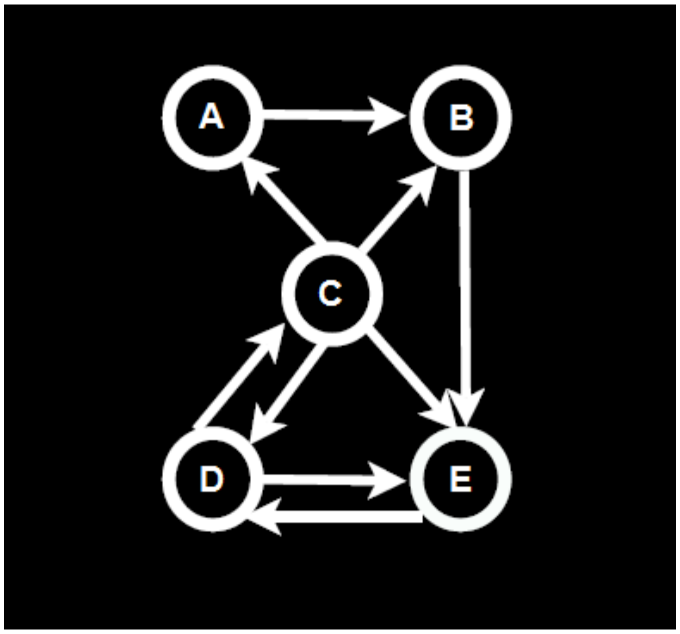
I will give you a very brief description of the PageRank algorithm. You will need to do your own research to truly understand it and implement it on your own. The original PageRank algorithm was described as:

```
PR(A) = (1-d) + d (PR(W1)/C(W1) + ... + PR(Wn)/C(Wn))
```

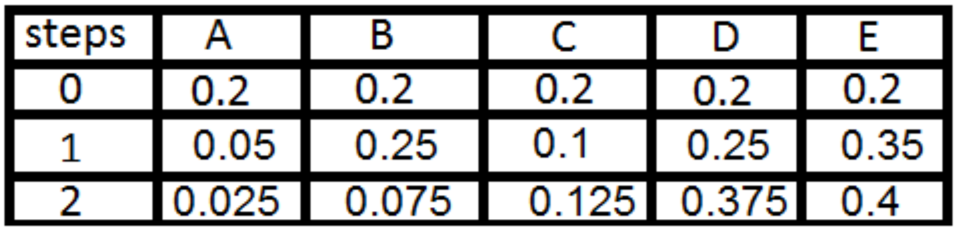
Where :

`PR(A)` – Page Rank of page `A`
`PR(Wi)` – Page Rank of pages `Wi` which link to page `A`
`C(Wi)` - number of outbound links on page Wi
`d` - damping factor which can be set between 0 and 1 (usually set at 0.85)

To calculate PageRank for the n Webpages, first we initialize all Webpages with equal page rank of `1/n` each. Then Step by Step we calculate Page Rank for each Webpage one after the other. Let's look at one example:



There are 5 Web pages represented by Nodes *A, B, C, D, E*. The hyperlink from each webpage to the other is represented by the arrow head.



steps	A	B	C	D	E
0	0.2	0.2	0.2	0.2	0.2
1	0.05	0.25	0.1	0.25	0.35
2	0.025	0.075	0.125	0.375	0.4

At the *0*th Step we have all Webpages' PageRank values set to 0.2, which is *1/5, or 1/n*. At the first step, to get the PageRank value of Webpage *A*, consider all the incoming links to *A*. So we have *1/4th* the Page Rank of *C* is pointed to A. So it will be *(1/5)*(1/4)* which is *(1/20)* or *0.05* the Page Rank of *A*. Similarly the Page Rank of *B* will be *(1/5)*(1/4)+(1/5)*(1/1)* which is *(5/20)* or *0.25*.

For step 2, the iteration continues with the results from step 1. In general, we consider the *(N-1)*th step's PageRank values when we calculate the PageRank values for the *N*th Step.

The algorithm will keep running until an equilibrium point. That is, there is no more updating to be done from the previous iteration. Once stable, the PageRank value for each node is its final PageRank value. One can then sort them to get the most important webpage to be displayed as part of the search results.

Your job is to write a program that takes as input a data file formatted in a way described below, runs the PageRank analysis on the graph extracted from the input file and outputs the individual items (dolphin names, etc.) ranked in descending order of their computed PageRank along with the PageRank value and the rank.

The program, pageRank.java (for example, if you are using Java) shall take as input the file name. It may also (if you want) take as input a flag specifying whether the dataset you are reading in represents a directed or undirected graph (some of you may find it useful, but it is not absolutely necessary so I'll leave it up to you).

Your program should parse the input, create a graph structure from it in main memory and run a version of PageRank ranking algorithm to rank the nodes in the graph. I will give more information during lecture as to what I expect as outputs.

There is a number of datasets available for this assignment. Your implementation shall run in adequate time on any of the datasets from the "small datasets" list provided below. The results and the time it took to run your implementation on these datasets must be put in your report.

1. KARATE: this dataset describes a small social network consisting of members of a university karate club.
2. DOLPHINS: a social network of a group of dolphins as observed by researchers over a period of time.
3. POLITICAL-BLOGS: a graph of political blogs connected by their citations of each other on the eve of the 2004 Presidential election in the US.

To simplify parsing, all small datasets are available to you in a uniform data format. The generic data format is:

```
<Node1>, <Node1-Value>, <Node2>, <Node2-Value>
```

Each row in the format above represents a single edge in the graph. Here, `<Node1>` and `<Node2>` are unique ids of two nodes forming an edge. `<Node1-Value>` and `<Node2-Value>` are two numeric labels that can be associated with an edge. In most of the datasets, one, or both of these values are set to 0. If non-zero values are present, they can be used to determine the direction of a link (if the dataset represents a directed graph).

Specific instructions regarding the data format for each of the datasets are included below.

KARATE: the dataset represents an undirected graph of social inter- actions. (If *X* interacted with *Y*, then *Y* interacted with *X*). There are two CSV files: `karate.csv` and `karateDir.csv`. The former represents each interaction as two rows in the CSV file in order to maintain symmetry. The latter represents each interaction using a single row. The node values are always 0. The node labels are integers from 1 to 34.

DOLPHINS: same type of graph, data format and file availability (`dolphincs.csv` and `dolphinsDir.csv`) as for the KARATE dataset. The only difference is that node labels (names of individual dolphins) are strings enclosed in double quotes.

POLITICAL-BLOGS: the dataset represents a directed graph of citations among political blogs. Citations are modeled as incoming edges in the graph. There is one data file available, `polblogs.csv`. Node labels are integer ids. The first node label is the citing blog; the second node label is the blog being cited. The first node value is always 0, the second is always 1, encoding the fact that each row represents an outgoing edge for the first node and an incoming edge for the second.

Additionally, all datasets are available in GML (Graph Markup Language) format. GML format allows for some flexibility in representing graphs (a variety of means can be used to encode meta-data about the graph, and node and edge labels). GML representations may contain extra information about the graphs (e.g., the GML file of the `POLITICAL-BLOGS` dataset contains a "liberal" or "conservative" label for each blog and identifies it). You are welcome, but are not required, to use the GML files provided to you.

Grading guide:

Total: 100 points:

- 10 points: you had described in your report the type of PageRank algorithms you had used in your implementation.
- 70 points: your PageRank algorithm produces the correct result for all three datasets and you had provided the results (as specified) in your report.
- 10 points: you had provided the time it took to run your implementation on each of the Datasets and the time is reasonable.
- 10 points: your algorithm works for both directed and undirected graph and produces (the correct) result in a reasonable running time.