PageRank on an Airpots Network

Gabriela HERNÁNDEZ, Maíra LADEIRA

INTRODUCTION

This work consisted on applying the "PageRank" algorithm to an airports and routes network. The network was built taken the airports as nodes and the routes between them were represented by directed weighted edges. The weight in an edge from airport i to airport j depends on the number of routes between these airports. [1]

DEVELOPMENT

Implementation

The treatment of the data, to construct the network, and the implementation of the Page Rank algorithm were done in Python 3, using the libraries Pandas 0.17.0 and Numpy 1.9.1.

For cleaning the data, we first removed the airports with duplicated IATA codes and those that had no IATA code from the airports file. In the routes file, we deleted the routes that did not have an airport from the airports file either in the origin airport or in the destination airport. Throughout this work we will refer to airports as the nodes (taken from the airports file) and routes as the edges (taken from the routes file).

The following assumptions were taken into consideration to have an efficient solution:

1. Dead-end nodes: Nodes without out-going routes. To identify them we checked in routes those airports that does not appear as origin airports. These kind of nodes break the PageRank algorithm, in the literature we can find that they suggest to add edges to all the nodes including itself, nevertheless for this purpose is not a good idea to add them "physically" because it would completely affect the performance of our program. For solving this issue, in each iteration of the PageRank algorithm we performed the following:

1) Take the probabilities of all the dead-end nodes.
2) Divide each probability by the total number of nodes and sum them. We considered that each dead-end node had only one route to the other nodes, hence the weight was 1.
3) Add this "sum" to each airport-node.

[1]For further details about the code organization and how to run it, please read the file "README.txt"

In this manner, when we were computing the PageRank for each airport i, we used the equation 1.

$$Q[i] = l* \left[\left[\sum_{j,i \epsilon edge} \frac{P[j] * w_{j,i}}{out_j} \right] + sum_{from\_DeadEnds} \right] + \frac{1-l}{n} \tag{1}$$

It is worth mentioning that the total number of dead-end nodes found were 2453, which would considerably affect the performance of the algorithm if the physical nodes would have been added.

2. Not incoming routes: We identified the airports without incoming routes by checking those airports that did not appear in the destination of the routes. As we know, these nodes will have the same PageRank value, due to they are not being pointed by other airports, they will only take the "default" sum coming from the dead-end endpoints. We separate these nodes each iteration and assigned them the same PageRank value computed as shown in equation 2.

$$q[i] = l * [sum_{from\_DeadEnds}] + \frac{1-l}{n} \tag{2}$$

It is worth mentioning that the total number of nodes with not incoming routes found were 2444.

3. Network: Instead of working with a nxn matrix (where n is the number of airport-nodes), we represented the network as a data frame (pandas data frame) that could be used as a hash table as well. The final data frame contains only the "destination" airports that have incoming routes to have less iterations to compute the PageRank of each node, as explained above the nodes without incoming routes will get the same probability. We processed the data in such a way that we ended up with a data frame as the one shown in the table I, where we included only two examples. In this table I, $Airport_i$ is a node and $Airports_j$ are the indexes of the airports j pointing the airport i, i.e. the routes from the $airportsj$ to the $airporti$, $Weight_{j,i}$ represents the weight of each edge and Out_j the total number of outgoing routes from each $airport_j$. We can easily notice that in this "graph" the total number of edges is $O(n)$ and not $O(n * n)$ as in a matrix. Thus, the complexity of applying the PageRank algorithm to this data frame is $O(n)$ obtaining a good and efficient performance.

TABLE I
EXAMPLE OF THE NETWORK REPRESENTATION

$Airport_i$	$Airport_i$ IDX	$Airpots_j$ IDX	Weight_j,i	Out_j
AAE	74	[221, 2861, 3142, 3566, 3573]	[1, 1, 2, 1, 2]	[81, 144, 135, 29, 208]
AAN	81	[242, 346, 810, 3696]	[1, 1, 1, 1]	[113, 239, 35, 30]

Performance and convergence

The convergence and computation time of the PageRank in a network is very important and may depend on some aspects like the number of nodes, damping factor and tolerance of error (floating point precision errors). However if the network is lineal and we can access to the information of each node linearly, we have found that the number of nodes does not play an important role for determining the number of iterations, it only affects the speed of computation of all node's PageRank.

Beginning with the damping factor l, the smaller l is, the faster the convergence, but the smaller l is, the less the true hyperlink structure of the web is used to determine web page importance. Moreover, as $l \rightarrow 1$, not only does convergence slow drastically, but sensitivity issues begin to surface as well [1]. Thus, the choose of the damping. factor is crucial, in the literature the most common value used for the damping factor is 0.85.

In [1], they proposed a rough estimate of the number of iterations depending on the damping factor l and the tolerance error τ (floating point precision errors), shown in equation 3. Hence, For $\tau = 1x10^{-6}$ = and $l = .85$, one can expect roughly 85 iterations until convergence to the PageRank vector. For $\tau = 1x10^{-8}$, about 114 iterations and for $\tau = 1x10^{-10}$. about 142 iterations.

$$Iterations = \frac{log_{10}\tau}{log_{10}l} \tag{3}$$

In table II, we summarized our results for the convergence and performance of the PageRank in the airports depending on the damping factor and on the tolerance error.

TABLE II
CONVERGENCE AND PERFORMANCE OF THE AIRPORTS' PAGERANK

Damping Factor l	Tolerance Error τ	Number of iterations	Computation Time
0.9	1.00E-08	95	54.19 seconds
0.9	1.00E-05	30	17.88 seconds
0.9	1.00E-03	4	2.3 seconds
0.85	1.00E-08	62	37.36 seconds
0.85	1.00E-05	19	11.52 seconds
0.85	1.00E-03	4	2.05 secomds
0.8	1.00E-08	45	28.12 seconds
0.8	1.00E-05	14	8.22 seconds
0.8	1.00E-03	3	1.68 seconds

We also have found, that the outcome results are the same for τ = 1E-8 and $\tau = 1E-5$ for each damping factor. Thus we can roughly say that if we want to apply a damping factor of 0.9 to compute the PageRank in our airports network, the total number of iterations required are 30 and with a damping factor of 0.85 are 19. For more detailed information about it please refer to the file "PageRank_performance.txt" included with the deliverables.

Main difficulties

The main difficulty of this lab work was the construction of the network to make the PageRank algorithm as fast as possible and to have only $O(n)$ edges, where n represents the number of airports.

RESULTS

Given the above information, we decided to obtain two results when the damping factor $l = 0.9$ and $\tau = 1E-5$, and when the damping factor $l = 0.85$ and $\tau = 1E-5$, both included in the deliverable files as Results_90.csv and Results_85.csv, respectively. The head of these results are shown in figure 1 and figure 2, respectively.

Airport_Name	Airport_Rank
Los Angeles Intl	0.006228
Chicago Ohare Intl	0.006212
Denver Intl	0.005985
Heathrow	0.005078
Charles De Gaulle	0.004920
Capital Intl	0.004843
Frankfurt Main	0.004785
Changi Intl	0.004697
Hartsfield Jackson Atlanta Intl	0.004687
John F Kennedy Intl	0.004426
Schiphol	0.004390
Dallas Fort Worth Intl	0.004139

Fig. 1. Results when $l = 0.9$ and $\tau = 1E-5$

Airport_Name	Airport_Rank
Chicago Ohare Intl	0.005591
Los Angeles Intl	0.005585
Denver Intl	0.005561
Heathrow	0.004365
Hartsfield Jackson Atlanta Intl	0.004287
Charles De Gaulle	0.004242
Capital Intl	0.004214
Changi Intl	0.004213
Frankfurt Main	0.004117
Sydney Intl	0.003957
Dallas Fort Worth Intl	0.003864

Fig. 2. Results when $l = 0.85$ and $\tau = 1E-5$

.

REFERENCES

[1] LANGVILLE, Amy N., MEYER, Carl D. *Deeper inside pagerank*. Internet Mathematics, 2004, vol. 1, no 3, p. 335-380.