UNIVERSITAT DE BARCELONA

MSc Fundamental Principles of Data Science

Project 3 NLA: Page Rank

Authors:

Jordi Segura

22 December 2022

1 Discussion of results

1.1 Computing the PageRank Vector Storing Matrices

PageRank is a well-known algorithm developed by Google that determines the importance of a webpage in the context of the World Wide Web. It was one of the key factors in the success of Google as a search engine, and it remains an important part of the company's ranking system today. In this report, we will discuss the basics of the PageRank algorithm, its various applications, and different ways in which it has been implemented. We will also examine the strengths and limitations of different PageRank implementations and how these implementations behaves depending on the dampling factor, *m*.

The first part of this project was to code the PageRank(*PR*) vector of the matrix M_m using the power method. One important aspect to consider when implementing the power method is how to handle the "dangling" webpages, which are those that have no outgoing links. The damping factor(*m*) typically has a value of 0.15, which means that there is a 85% probability that the surfer will follow a link on the current webpage, and a 15% probability that they will continue to a random webpage. This helps to ensure that the algorithm does not get stuck in a cycle of pages, and that the PageRank values eventually converge to a steady state.

This algorithm then reduces to iterate:

$$x_{k+1} = (1-m)GDx_k + ex^t x_k$$

until

$$||x_{k+1} - x_k||_\infty < tolerance$$

In the figure 1 we can see the time of the algorithm for different values of the dampling factor. Remark again that a higher value of the damping factor may lead to faster convergence, as the surfer is more likely to jump to random webpages and not follow the links as closely. However, a higher value of the damping factor may also result in a less accurate ranking of webpages, as the surfer is less likely to follow the links as closely. This is what we can observe in the plot below, a decreasing time execution plot with an increasing *m*:

1.2 Computing the PageRank Vector Without Storing Matrices

In this exercise, our goal was to compute the PageRank (*PR*) vector of the matrix M_m using the power method, without storing the full M_m matrix in memory. This can be useful in situations where the matrix is too large to fit in memory, or when we only need to compute a few elements of the *PR* vector and don't need to store the entire matrix.

To solve this problem, we can use a technique called "sparse matrix-vector multiplication". This involves only storing the non-zero elements of the matrix, and using these values to compute the *PR* vector iteratively.

To implement this algorithm in code, we can use a data structure called a "sparse matrix" to store only the non-zero elements of M_m. We can then iterate the equation above until the *PR* vector converges, using the sparse matrix to compute the matrix-vector multiplication efficiently.

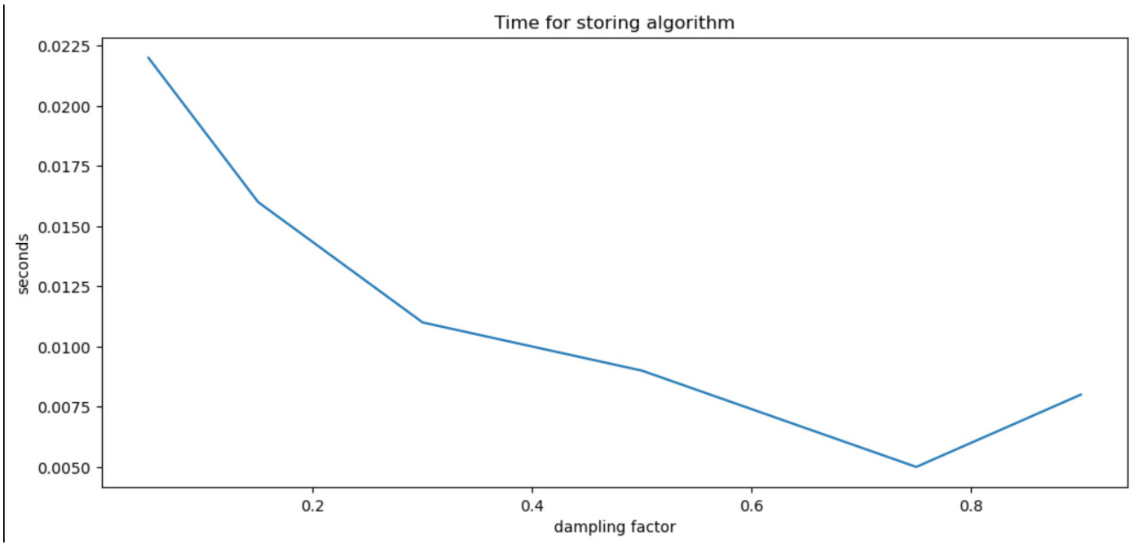

Figure 1: Results of the storing algorithm respect to the dampling factor, m

In this case, we can see how that without storing the matrices the time is way higher than in the first case(Figure 2). We can also observe how the slope of the elbow for small values of m is more vertical, so it plays a crucial role here the dampling factor.

Moreover, it is interesting to see the difference between the results of both algorithms, which can be found in the Figure 3. In this case, the magnitude is $10e^{-5}$, but seems to decrease while m increases, which theoretically should not be that way as we are allowing more randomness.

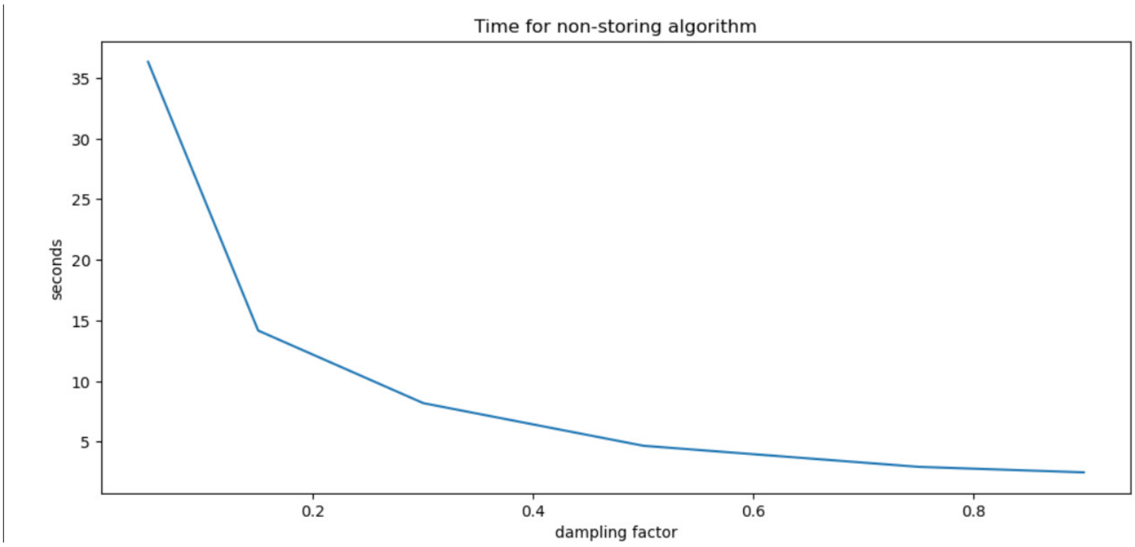

Figure 2: Results of the no-storing algorithm respect to the dampling factor, m

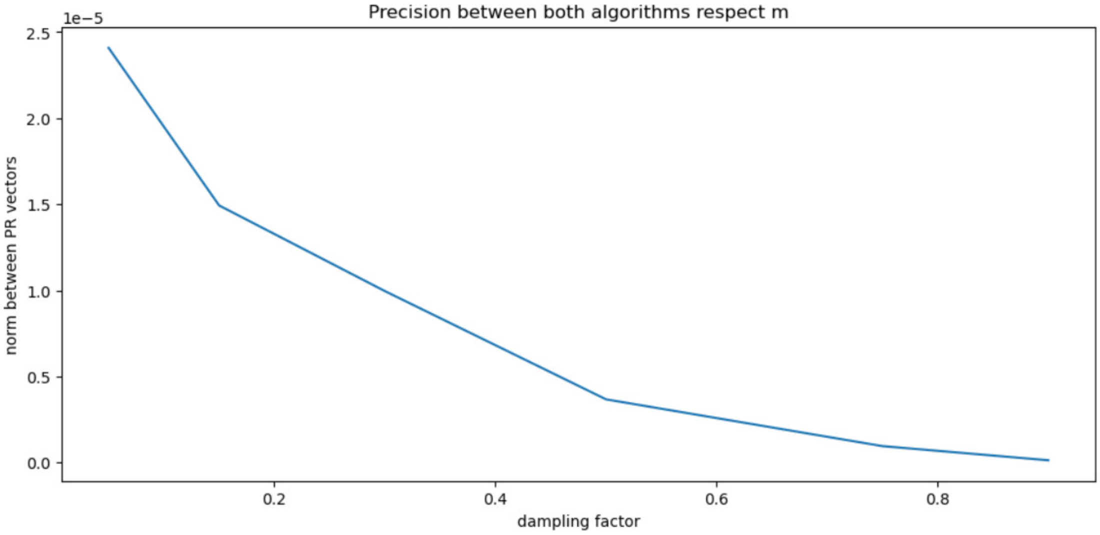

Figure 3: Difference of results between both techniques respect to the dampling factor, *m*

2 Conclusions

The PageRank project is a complex and interesting topic to analyze and understand the structure and importance of web pages. Some challenges I encountered in the project have been to understand the underlying concepts and algorithms and implementing the calculations efficiently, which in a first step were done with numpy.

I enjoyed the opportunity to see the results of the analysis and understand how it can be used to inform search engine rankings or other applications.

Overall, I think it is from the 3 projects the one I liked most for its direct applications.