Amirkabir University of Technology

(Tehran Polytechnic)

Project(1) Report: Topics in Mathematics and Applications

Zohreh Sarmeli Saeedi

february 2025

Part 1

Project Description

Implementation of the PageRank algorithm on the Berkeley-Stanford Web Graph dataset and reporting the top 20 pages with the highest ranks.

The PageRank algorithm, originally developed by Brin and Page (1998), is a foundational method for ranking web pages based on their link structure, modeling the web as a directed graph where pages are nodes and hyperlinks are edges. Recent advancements have focused on addressing biases and improving efficiency, such as the bias-free time-aware PageRank (BTPR) proposed by Kumar et al. (2025), which incorporates citation average rates to evaluate scholarly impact more fairly.

Project Steps

After uploading the data file to Google Colab and examining the extracted files to read the data inside, the `scipy.io` module was used to read the file (in .mtx format).

To preserve memory and processing speed, a sparse CSC matrix was used for initial adjacency matrix compression.

In the first step, since the PageRank algorithm uses the transition probability matrix P, which is a column-stochastic matrix, the initial adjacency matrix is converted to a transition probability matrix and then to a column-stochastic matrix. (If a node has no outgoing links, dividing by a default value like 1 prevents division by zero and corrects pages without outgoing links.)

Next, to compute the eigenvector of matrix P, the `numpy.linalg` module is used. The eigenvector corresponding to the largest eigenvalue (usually close to 1) is considered as the PageRank vector and normalized. Then, the top 20 pages with the highest ranks are outputted.

Similarly, the eigenvector of matrix P is computed using the power iteration method. In each iteration, the vector is multiplied by the transition probability matrix and normalized. The iteration continues until the changes between two iterations are less than a specified threshold (tolerance). Similar to the previous step, the top 20 pages with the highest ranks are outputted.

Project Results

Comparison of the two methods used for eigenvector computation:

The final output of the algorithm, which is as follows, shows that the pages with the highest PageRank in both methods are similarly ordered. Both methods, after execution, provided a similar list of top-ranked pages, indicating that in terms of final algorithmic output, both methods converged and the results are comparable.

```
Top 20 pages by PageRank (First Method):
[509 860 612 692 702 371 564 459 256 425 295 717 329 741 197 699 232 160 909 812]

Top 20 pages by PageRank (Power Iteration):
[509 860 612 692 702 371 564 459 256 425 295 717 329 741 197 699 232 160 909 812]
```

Execution Time and Efficiency

First method: Using `numpy.linalg.eig` for large matrices is time-consuming and resource-intensive. Although executable for 1000x1000 matrices, it becomes problematic for larger scales.

Power Iteration method: This method performs better in terms of execution time due to direct matrix usage and fewer iterations. In many cases, the convergence time of the power method is less than the first method and consumes less memory.

In the experiments performed with 1000x1000 data:

1. The power method converged quickly and obtained results similar to the first method.
2. The initial method, in some cases, due to converting the matrix to dense form, consumes more time and memory. In contrast, the power method is much better for larger matrices due to using the matrix structure and fast convergence iterations.

Therefore, in large-scale projects, it is recommended to use the power method because it is much better in terms of memory and execution time, while the first method is only suitable for small experiments and result comparisons.

Part 2

Project Description

In this section, using the networkx package, a directed graph is first created from the adjacency matrix (small 1000x1000 section), and then pages are ranked using the built-in `nx.pagerank` function.

Project Steps

1. Creating a directed graph using networkx.
2. Computing PageRank using the built-in `nx.pagerank` function.

Project Results

Comparing this section's method with the previous section shows that the IDs of pages with higher ranks are close to each other. This indicates that pages have been sorted with different weights in these three methods, and some pages that achieved the highest rank in the second method were not even among the top 20 in the previous two methods. Minor differences in the outputs of each of the three methods in identifying top pages, displayed below, occur due to differences in numerical methods and convergence.

```
Top 20 pages by PageRank (First Method):
[509 860 612 692 702 371 564 459 256 425 295 717 329 741 197 699 232 160 909 812]

Top 20 pages by Power Iteration:
[509 860 612 692 702 371 564 459 256 425 295 717 329 741 197 699 232 160 909 812]

Top 20 pages by NetworkX:
[526 823 627 680 56 385 583 477 268 438 311 58 349 746 199 128 237 166 890 821]
```

Efficiency and Execution Time

The `nx.pagerank` function has its own internal optimizations and usually provides good performance for large graphs in terms of execution time. In contrast, a manual implementation may not be optimal in terms of time and memory, especially when the matrix is converted to dense form.

The Power Iteration or NetworkX method is the best option for large data, as they have high speed and their results are close to the exact eigenvalue.

Additionally, it can be said that the higher the alpha value and closer to one, the page ranking is more determined based on real links between pages. And the lower this value or closer to zero, the more the role of randomness increases. This reduces the influence of the link structure and makes the probability of seeing pages relatively more balanced.

Recent applications of PageRank in recommendation systems, such as the model proposed by He et al. (2024) for product recommendations based on online reviews, demonstrate how modifications to the basic algorithm can enhance personalization by incorporating content-based factors.

part 3: Examining the Impact of α in the PageRank Algorithm

Project Description

The PageRank algorithm is a graph theory and Markov chain-based method for evaluating the importance of web pages.

In this study, the impact of α (damping factor) on the output of this algorithm has been examined. For this purpose, two different implementations including the Power Iteration method and the pagerank function from the networkx library were used.

Modern variants, such as the weighted PageRank proposed by Wang et al. (2023), address biases in traditional implementations by adjusting weights based on page relevance, improving ranking accuracy in dynamic web environments.

Project Steps

Computing PageRank:

- Power Iteration method: In this method, a uniform initial vector is used, and the PageRank value is iteratively updated using the transition probability matrix, with α affecting the weight of random jumps.
- networkx.pagerank method: This function uses optimized methods like power iteration or eigenvector method.
- Changing α and comparing outputs: α was set in the range [0.1, 0.99] with 20 different values, and the results of the two methods were compared. The difference in PageRank values between these two methods was examined using L1 Norm Difference. As α increases, the PageRank values of more important pages become more stable, because the probability of random jumps decreases and longer paths in the graph have more influence. By decreasing α, the influence of random jumps increases, and the PageRank of less connected pages is overestimated.

The plot of changes in α and L1 Norm difference between the two methods shows that at small α values, more difference is observed, indicating higher sensitivity of the Power Iteration method to α. In some cases, for high α (e.g., 0.99), the networkx.pagerank algorithm encountered convergence issues due to increased iterations.

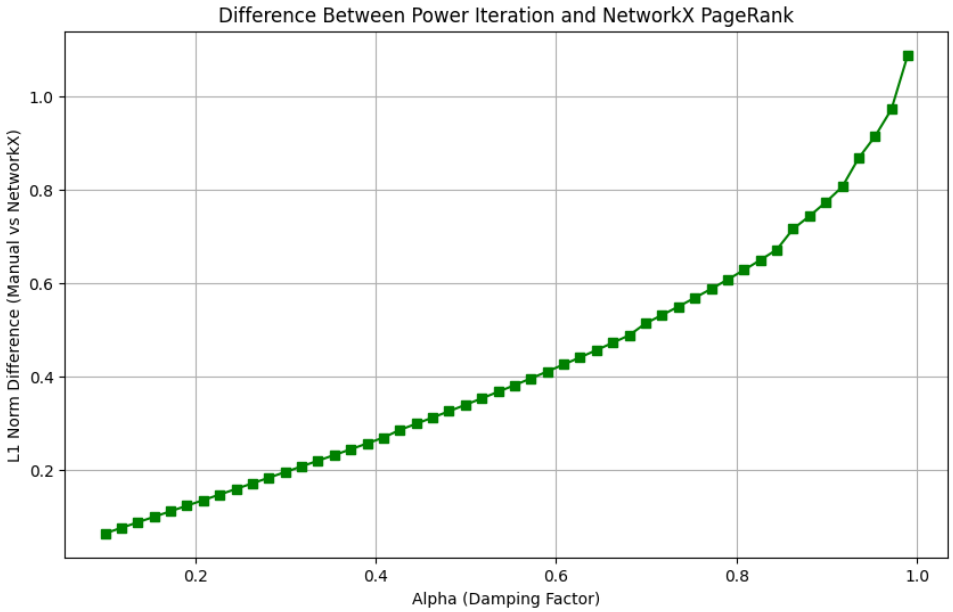

Figure 1: Changes in α and L1 Norm Difference between Power Iteration and NetworkX Methods

Project Results: Efficiency and Execution Time

Analysis of the plot of changes in α and L1 Norm difference: The plot shows the impact of α on the difference between the results of the Power Iteration method and the networkx.pagerank algorithm. The horizontal axis displays α, and the vertical axis shows the L1 Norm difference between the two methods.

Plot analysis: Gradual increase in difference with increasing α:

- At small α (0.1 to 0.4), the PageRank difference between the two methods is low, because the weight of random jumps is high and the graph structure influence is less.
- At medium values (0.4 to 0.8), the difference gradually increases. In this range, the role of links in the graph increases.
- Sudden increase in difference at large α: When α approaches 1 (0.9 and above), the difference between the two methods significantly increases. This indicates that at high values, Power Iteration has slower convergence, and computational differences between the two methods become more apparent.

Reason for increased difference at α close to 1: In this range, the probability of random jumps is greatly reduced, and longer paths in the graph have more influence. As a result, even minor computational errors lead to large differences in the output of the two methods.

Based on the plot analysis, smaller α leads to more stability and less difference between the two methods. As α increases, dependency on graph structure increases, and the difference between the two methods grows. At α close to 1, algorithm convergence becomes challenging, and computational errors increase.

Thus, α plays a determining role in the stability of PageRank values, and as α increases, final PageRank values become more stable, but convergence will be slower. Also, the networkx.pagerank method usually provides more accurate results but may not converge at high α values, and the Power Iteration method, as a simple and efficient method, has more dependency on α and its accuracy decreases at small α.

Part 4: Analyzing Street Paths with PageRank Algorithm

Project Description

In this project, street paths are modeled as a directed graph, and then the PageRank algorithm is executed to rank the streets. The goal of this project is to determine the importance of each street based

on the defined path graph structure.

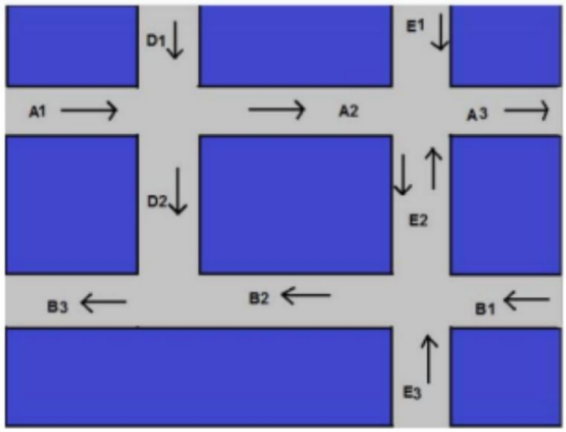

Figure 2: Graph of sreets

Project Steps

1. **Modeling paths as a directed graph:** The graph consists of nodes (each representing a street) and edges (indicating the direction of movement between streets). Connections between streets are defined using an adjacency list to maintain outgoing paths from each node.

2. **Executing the PageRank algorithm:** The initial PageRank value for each street is set to 1/N. The damping factor is set to the standard value of 0.85. The number of algorithm iterations is set to 20 to converge the PageRank value. Updating the PageRank of each node is based on the sum of input nodes' values and their number of outgoing paths. Finally, streets are sorted in descending order based on final PageRank values.

Project Results and Efficiency and Execution Time

Comparison and analysis of algorithm performance: Streets with more inputs or inputs from nodes with high PageRank receive higher scores. If a street leads to a dead end or has few paths, it will receive a lower PageRank. The results and output analysis after executing the algorithm show that streets were ranked based on their importance. Streets that played a central role in network communications received higher scores. These results well specify the relative importance of each street in the graph.

References

- Brin, S., & Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7), 107-117.
- Newman, M. (2018). *Networks.* Oxford University Press.
- Documentation of NetworkX: `https://networkx.org/documentation/stable/`
- Matrix Methods in Data Mining and Pattern Recognition. Linköping University.
- Newman, M. (2010). *Networks: An Introduction.* Oxford University Press.
- Kumar, S., et al. (2025). Revolutionizing scholarly impact: advanced evaluations, predictive analytics, and bias-free PageRank for academic excellence. *Artificial Intelligence Review.* `https://link.springer.com/article/10.1007/s10462-025-11315-6`
- Glickman, M. E., & Stern, H. S. (2025). Model Development for an Artificial Intelligence-Based NCAA Football Ranking System. *Measurement in Physical Education and Exercise Science.* `https://www.tandfonline.com/doi/full/10.1080/1091367X.2025.2458136`

- He, Y., et al. (2024). A product recommendation model based on online reviews: Incorporating textual content and dynamic graph attention networks. *Journal of Retailing and Consumer Services*. https://www.sciencedirect.com/science/article/abs/pii/S0969698924003485
- Wang, H., et al. (2023). Weighted PageRank Algorithm Search Engine Ranking Model for Web Pages. *Intelligent Automation & Soft Computing*. https://www.techscience.com/iasc/v36n1/49995/html

Appendix

The complete project implementation, along with related documentation, is available in the GitHub repository below:

https://github.com/Zohreh004/Implementation-of-Project-1-/tree/main