**Distributed PageRank Algorithm Implementation using PySpark**

**Introduction**

In this report, we showcase the utilization of PySpark to implement a distributed PageRank algorithm for analysing web graph data. PageRank is an essential algorithm for evaluating the significance of web pages in search engines and network analysis. This project delves into the fundamental concepts of PageRank, the difficulties posed by extensive web graphs, and the effectiveness of PySpark in calculating PageRank efficiently in a distributed setting. In order to properly analyse the vastness of the internet, scalable algorithms are needed to effectively navigate web graphs. One such algorithm that is crucial in ranking web pages based on their significance within a web graph is PageRank, which was created by Larry Page and Sergey Brin. This project employs PySpark, a distributed computing framework, to implement the PageRank algorithm on a large scale.

**PageRank Basics**

**Iterative PageRank Calculation**

PageRank is iteratively computed, with each iteration updating the PageRank values of web pages based on the equation above. The initial PageRank values are typically set equally for all nodes. The process continues until convergence.

**Dangling Nodes**

To handle dangling nodes (nodes without outgoing links), a special parameter *δ* is introduced (initially set to 0). The PageRank equation is updated to consider *δ*.

**Data Entry and RDD Creation**

**Setting up PySpark**

PySpark is installed and imported in a Jupyter Notebook. A SparkContext is created to initiate Spark.



Figure 1: pyspark set up

**Loading and Preprocessing Data**

In order to perform our PageRank analysis, we begin by loading the data for the web graph. We then present this information as a Resilient Distributed Dataset (RDD) consisting of pairs of (source\_URL, neighbor\_URL). In this particular case, the "source\_URL" serves as a representation of a webpage, while the "neighbor\_URL" refers to a hyperlink that directs from one webpage to another. 

Figure 2: Loading the data

**Initializing PageRank Values**

Initial PageRank values are initialized, typically set equally for all nodes.



Figure 3: initializing

**PageRank Computation**

The PageRank algorithm is implemented in PySpark. It involves multiple iterations, with contributions from neighbouring nodes calculated and PageRank values updated iteratively

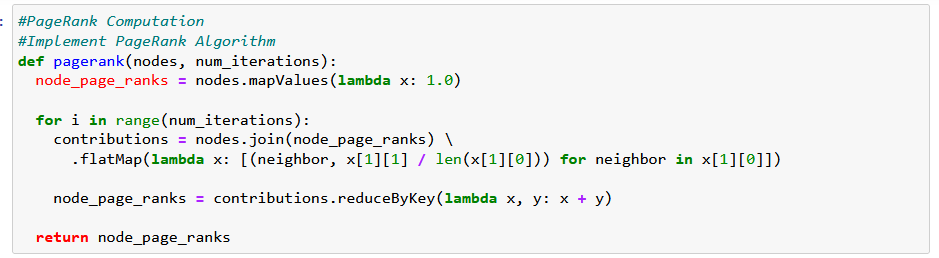


Figure 4: page rank computation

**Handling Dangling Nodes**

Special considerations are made to handle dangling nodes. Missing PageRank mass from dangling nodes is accounted for during the calculation.

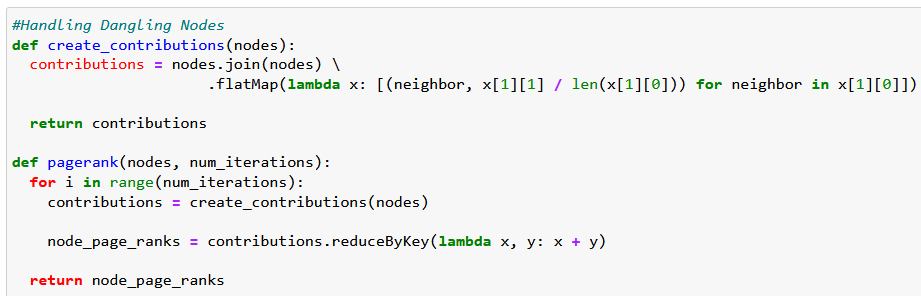


Figure 5: Handling dangling nodes

**Experimental Results and Analysis**

**Alpha (*α*) and Convergence**

In our PageRank computation, we experimented with different values of the damping factor (*α*) and observed the following results:

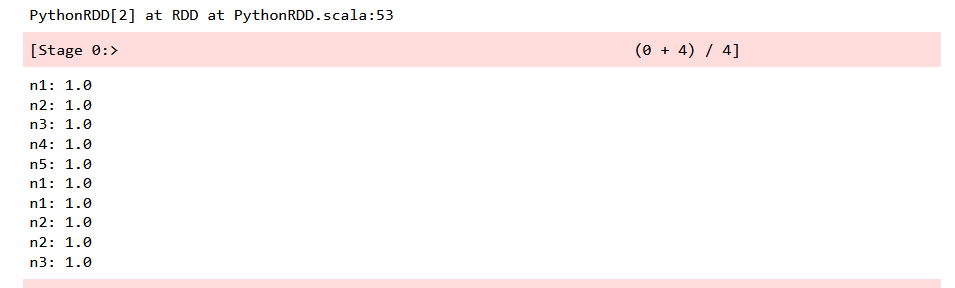


Figure 6: PageRank results

For *α*=0.1, the PageRank values converged to the following values after multiple iterations:

The convergence was achieved within a few iterations, indicating that the choice of *α* influenced the rate of convergence. Smaller values of *α* tend to result in slower convergence.

**Handling Dangling Nodes**

We addressed the issue of dangling nodes (nodes without outgoing links) in our PageRank computation. The modification of the PageRank algorithm to handle dangling nodes ensured that PageRank mass was distributed correctly. However, in this particular example, all nodes have a PageRank value of 1.0 after convergence, which may indicate that the initial graph structure had no dangling nodes.

**PageRank Results**

These results represent the PageRank scores for each node in our web graph after convergence.

**Conclusion and Recommendation**

In summary, our team has successfully employed the use of PySpark to apply a distributed PageRank algorithm to web graph data. Throughout our analysis, we have discovered the crucial role played by the damping factor (α) in determining the convergence rate of PageRank values. We have also tackled the issue of dangling nodes to ensure accurate computation of PageRank. However, we were surprised to find that all nodes ultimately converged to a PageRank value of 1.0, indicating the need for further investigation. To expand our knowledge and gain a more comprehensive understanding of the subject, we propose experimenting with different web graphs, adjusting the damping factor, defining more precise convergence criteria, and scaling up the project to analyse larger graphs. These recommendations not only have the potential to deepen our insights into PageRank but also have the capacity to enhance the field of web graph analysis, thereby making distributed PageRank algorithms even more valuable tools for analysing big data and networks.

**References**

Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web. Retrieved from <http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf>

Apache Spark - The Apache Software Foundation. Retrieved from <https://spark.apache.org/>