# 1. Introduction

The PageRank algorithm quantify the importance of each web page based on the link’s structure of the whole network. The main idea that allows PageRank to work is that links that point to a page from other pages considered important, make it more important. Pratically each page “vote” the pages linked to it. This problem, is mathematically solved by calculating the eigenvector corresponding to the greatest eigenvalue of a stochastic matrix. This process is completed by a centralized algorithm which given the big dimension of the network (more than 8 billion of pages) takes a week to finish. In order to solve the problem was proposed an approach based on randomized distributed algorithms. This algorithm is characterized by 3 important aspects:

1. Each page can calculate its PageRank locally communicating with the pages directly linked to it
2. Each page decides to start the communication with the others at a random time.
3. The workload requested to each page is very mild

This algorithm is related to the consensus theory in such way that some nodes exchange their values with their neighbours in order to reach the consensus i.e. all nodes reach the same value.

# 2. PageRank

Let's consider a network of web pages indexed from 1 to n. The network is represented by a graph where V is the set of nodes (web pages) and E is the set of edges (links between pages) which belongs to is an edge is it exist a link that lead from page i to page j. The pagerank of a page i is is a real number in [0,1] denoted by . In practice a page i has greater importance than a page j if

The value of each page(PageRank) if computed as Where is the set of the pages which has an entering link in i, is the number of links which are going out from the node j. It's a good practice to normalize the values such that .

Let's rewrite values in a vectorial form:

Using this notation we can rewrite the problem in the following form:

e where the matrix A = ( is called link matrix and their element are defined as:

Notice that the vector x\* is the eigenvector corresponding to the eigenvalue 1 of the matrix A. In general this eigenvector exist and is unique if the graph is strongly connected. Since the actual network is not strongly connected in the following we will assume that every page are somehow connected for simplicity. So the matrix A becomes column stochastic and will always exist the eigenvalue 1. To guarantee uniqueness of the eigenvalue 1 we introduce a modified version of the matrix A, defined as following:

With m parameter with typical value of 0.15. M is positive and stochastic and, for Perron Theorem, primitive, and in particular the eigenvalue 1 is unique. So we redefine the vector x\* as following:

As we told, compute the eigenvector for large dimension matrixes is complex, for this reason we will use recursion:

Where Is a probability vector and notice that . This formula is important because allow us to use directly the sparse matrix A (with several zeros)

Using the lemma 2.3 from the documentation the vector converges always to the consensus value for .

# 3. A distributed randomized approach

Now we are going to show a distributed approach to the calculus of the vector . The base scheme of the protocol is the following:

At time instant k, the page I starts the update of the PageRank value sending its value to the other pages connected to it. To implement the distributed scheme, we assume that the pages that start the update, are determined in a random manner. This is determined by the aleatory process . Each node can start the process with equal probability, that is

In particular, we consider the distributed update scheme in the following form:

With distributed link matrices and the parameter so that the PageRank values are computed through the time average of the state x defined as follows:

The time average converge to the consensus value in the mean square sense i.e.

## 3.1 Distributed link matrices

We define the distributed link matrices in such way that:

* The i-th row and column is equal to the same row and column of A
* The elements on the diagonal are 1 - l=1…n e li
* All the other elements are 0

Formally:

These matrices are stochastic because the original matrix is stochastic. To clarify the distributed link matrices’ property, let’s consider the following simple update scheme:

So, we are interested to observe the average dynamics. We define as the average matrix and the average of as and its recursion is

**Lemma 3.2** The matrix has the following properties:

* There exist an eigenvector corresponding to the eigenvalue 1 for both matrices A and

## 3.2 Mean square convergence of the distributed update scheme