**Assignment 10**

Page Ranking and Web Mining

**Name: Dean D’souza**

**H.U. ID: 168424**

# **Solutions**:

**Following is a link to one of their speakers, try take a listen to on (**<https://www.youtube.com/watch?v=-mUI1g5PZXI>**)**

**(**<https://www.youtube.com/watch?v=4c3DAxQXzLI>**)**

**(you may explore some more clips as well) ... and respond to the questions below:**

1. **Explain the math behind the Pagerank process.**

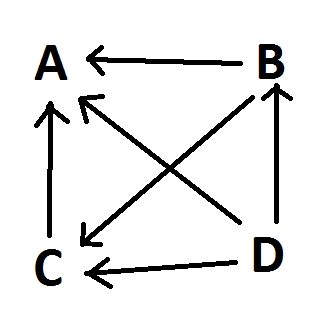
PageRank, the algorithm used by Google Search to rank websites, is an algorithm which outputs a probability distribution which represents the likelihood of a person clicking at random arriving at a specific page. This algorithm can work by computing this probability distribution either iteratively or algebraically, but in general the iterative method is preferred and is viewed as a power iteration method or power method.

We can show the math involved in PageRank algorithm through the following example:

1. Assume that there are four pages A, B, C, and D. PageRank initializes the probability distribution of each page to the same value (in this case it will be 0.25 as there are four only four pages under consideration). In general, the probability each page is initialized to is:

**PR(pi;0) = 1/N**

1. Now we assume the following scenario seen in the image below.



The PageRank algorithm then works to confer the page rank from a given page to the target of its outbound link (shown by the directed arrows) by dividing it equally among all outbound links. In this case it would mean that for the page A:

**PR(A) = PR(B)/2 + PR(C)/1 + PR(D)/3**

Or more generally:

**PR(A) = PR(B)/L(B) + PR(C)/L(C) + PR(D)/L(D)**

Where L() can be thought of as a function which gives the number of outbound links for that page.

We can further generalize this equation as:

1. However, to consider that the web surfer or user will eventually stop clicking, a damping factor (which measures the probability that at a given stage the user will continue) is added to the equation. Many studies were conducted and in general the best damping factor is around 0.85, but for the equation we represent it as *d*. So the overall formula for iteratively finding the PageRank of a page is:

**)/N + *d***

Where M(pj) is a set of pages that link to pi.

This can further be represented in Matrix notation as:

**1**

Where R() is an eigen vector of the Page Ranks (a page rank vector) of all the web pages, M is a matrix of all the links between pages and 1 is a column vector of length N. M is defined as:

Mij = 1/L(pj) if j links to i

Or Mij = 0 otherwise

The power method makes use of the matrix notation to iteratively improve on the page rank for each page. In general number of iterations would be around 100 (considered to be a good number for the algorithm)

1. **Write an algorithm to implement the Pageranking.**

A simple version of the PageRank algorithm is as follows:

1. Let N <- number of Pages
2. Let d <- 0.85 //damping factor
3. Let A[][] <- the adjacency matrix of size NxN (0 for no outgoing link, 1 for outgoing link)
4. Let L() <- function that gives the number of outbound links
5. For i from 1 to N:

PR[i] <- (1/N) // Assigning initial PageRanks

1. Let iter <- 0 // number of iterations (initially 0)
2. While iter != (N-1):
3. For i from 1 to N:

Let temp[i] <- PR[i] // storing existing Page rank values to be used later

1. For i from 1 to N:

PR[i] <- (1-d)/N

For j from 1 to N:

PR[i] <- PR[i] + (A[j][i] \* d \* (temp[j]/L(j))) //calculating new Page rank value ()

1. iter <- iter+1
2. return PR[]

The above algorithm should return a page rank vector containing the page rank for each page (the page number will act as an index to obtain the rank). The number of iterations can be adjusted by adjusting the condition for iterations but for this algorithm it was set to be 1 less than the number of pages.

If we consider similar notation for the adjacency matrix as shown in the algorithm, then L() would be of the form of:

1. let num <- the required page for which number of outbound links need to be calculated
2. let N <- the number of pages
3. let A[][] <- the adjacency matrix of size NxN (0 for no outgoing link, 1 for outgoing link)
4. let count <- 0
5. for i in 1 to N:

count <- count + A[num][i] //iteratively adding all outbound links

1. return count

# **References:**

[1] “PageRank”, Wikipedia, (<https://en.wikipedia.org/wiki/PageRank>)

[2] “What is Google Pagerank Algorithm & How It Works”, Mukesh Mali (youtube uploader), (<https://www.youtube.com/watch?v=-mUI1g5PZXI>)

[3] “Inside Google: Page Rank | Rachid Guerraoui”, Wandida.com (youtube uploader), (<https://www.youtube.com/watch?v=4c3DAxQXzLI>)

[4]