# PageRank Notes

February 12, 2019

Reference : **PageRank** Sergey Brin and Larry Page, PhD students from Stanford University, at Seventh International World Wide Web Conference (WWW7) in April, 1998.

Google realized that with the ever growing inter-connectivity of the web, it can be modeled as a directed graph. A link from on page to another is a directed edge in the graph. With this model, we can make the assumption that nodes with higher number of incoming links, are more likely to be important. Thus, Google decided to rank the pages by the link structure. Page Rank (PR) was used to simulate where a Web surfers, starting at a random pages, could congregate by following links from one page to another. PR is essentially a vote by all other pages on the Web about the important another page.

- A page is more important if its has more in-coming links (those are the ones the page cannot control).
- Are all incoming links equal? Nah links from more important pages count more

## 1 Page Rank Formulation

Suppose that a page $P_j$ has $L_j$ links. If one of those links is to page $P_i$, then $P_j$ will pass on $\frac{1}{L_j}$ of its importance to $P_i$.

The importance ranking of $P_i$ is then the sum of all the contributions made by the pages linking to it. That is, if we denote the set of pages linking to $P_i$ by $B_i$, then :

$$PR(P_i) = \sum_{P_j \in B_i} \frac{PR(P_j)}{L_j}$$

The PR is usually computed iteratively (vs algebraically). The transition matrix M (n by n matrix, where n is the number of pages) represents the probability distribution of the location of a random surfer step j. The vector V represents the principle eigenvector of M.

The probability $x_i$ that a random surfer will be at node i at the next step:

$$x_i = \sum M_{ij} V_j$$

$V_j$ is the probability that the surfer was at node j at the previous step

$M_{ij}$ is the probability that a surfer at node j will move to node i at the next step.
If we surf any of the n pages of the Web with equal probability

1. The initial vector $v_0$ will have $\frac{1}{n}$ for each component
2. After one step, the distribution of the surfer will be $Mv_0$
3. After two steps, $M(Mv_0) = M_2 v_0$ and so on
4. Multiplying the initial vector $v_0$ by M a total of i times gives the distribution of the surfer after the $i^{th}$ steps

The distribution of the surfer approaches a limiting distribution v that satisfies v = Mv provided two conditions are met:

- The graph is strongly connected : It is possible to get from any node to any other node
- There are no dead ends : Dead ends = nodes that have no outgoing links

To avoid dealing with dead ends and non-strongly connected graphs, we modify the calculation of PageRank to allow each random surfer a small probability of teleporting to a random page rather than following an out-link from their current page. The iterative step, where we compute a new vector estimate of PageRanks v' from the current PageRank estimate v and the transition matrix M is

$$v' = \beta Mv + \frac{e(1-\beta)}{n}$$

Where,

$\beta$ is a chosen constant (usually in the range 0.8 to 0.85)

e is a vector with 1s for the appropriate number of components

n is the number of nodes in the Web graph

The term $\beta Mv$ represents the case where the random surfer decides to follow an out-link from their present page ( with probability $\beta$). The term $(1-\beta)/n$ is a vector representing the introduction, with probability 1-, of a new random surfer at a random page.

**Question 1:** Consider that you are calculating PageRank values for web pages. There are 10 Billion web pages and you have created a 10 Billion x 10 Billion transition matrix M. As a part of iterative computations, you use the MapReduce computing framework without Taxation. The $k^{th}$ iteration of the MapReduce job will create a vector $v^k$ with 10 Billion items. The $j^{th}$ item in $v^k$ is calculated using the following formula:

$$v_j^{(k+1)} = \sum_j m_{ij} v_j^k$$

What are the values $m_{ij}$ stored in the transition matrix M?

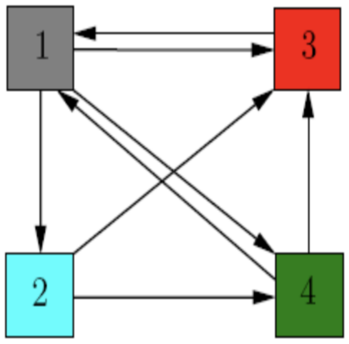
- A: The total number of times that web page i has been visited
- B: The probability that page i is to be visited from the $j^{th}$ page
- C: The page i's page rank value after $j^{th}$ iteration
- D: Random number generated by server

**Question 2**. What are the values $v_j$ for the $k^{th}$ iteration?

- A. The average PageRank value of the page j after the $k^{th}$ step
- B. The probability that the surfer was at the node j at the $(k-1)^{th}$ step
- C. The highest PageRank value of the page j after the $k^{th}$ step
- D. The lowest PageRank value of the page j after the $k^{th}$ step

## 2 Example

Assuming the following directed graph, lets compute PageRank.



Lets begin by creating the tranition matrix M of this graph:

$$M = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix}$$

Suppose that initially the importance is uniformly distributed among the 4 nodes, each getting $\frac{1}{4}$. Denote by v the initial rank vector, having all entries equal to . Each incoming link increases the importance of a web page, so at step 1, we update the rank of each page by adding to the current value the importance of the incoming links. This is the same as multiplying the matrix M with v . At step 1, the new importance vector is $v_1$ = M$v_0$. We can iterate the process, thus at step 2, the updated importance vector is $v_2$ = M$v_1$. Numeric computations give:

$$v_0 = \begin{bmatrix} \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \\ \frac{1}{4} \end{bmatrix}$$

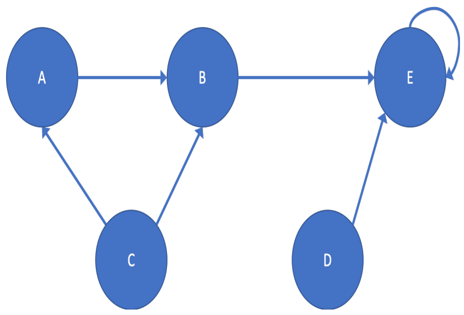
$$v_1 = Mv_0 = \begin{bmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{bmatrix}$$

$$v_2 = Mv_1 = \begin{bmatrix} 0.43 \\ 0.12 \\ 0.27 \\ 0.16 \end{bmatrix}$$

$$v_3 = Mv_2 = \begin{bmatrix} 0.35 \\ 0.14 \\ 0.29 \\ 0.20 \end{bmatrix}$$

. . .

$$v_8 = Mv_7 = \begin{bmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{bmatrix}$$



Given the above directed graph, compute the first 3 iterations of PageRank with taxation. Assume $\beta$ is 0.85.