

# **PAGE RANK ALGORITHM**

---

A Project Report Submitted to Amrita School of Engineering in partial fulfilment of the Requirements for the Degree of Bachelor of Technology in Computer Science and Engineering (Artificial Intelligence):

## **MATHEMATICS FOR INTELLIGENT SYSTEM 1 (19MAT105)**

Submitted by **Harish .B (20021)**

Under the Guidance of **Dr. Ranjith Kumar**

**Designation**

**Department of Mathematics**



**Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham**

**Chennai – 601 103, Tamil Nadu, India.**

**February, 2021.**

# **ACKNOWLEDGEMENT**

We would like to express our gratitude to our professor Dr. Ranjith Kumar who allowed working on such an amazing research-based project. We would also like to extend our gratitude towards director Shri. Manikandan, principal Dr. Shankar, and Chairperson Dr. Jaykumar have always encouraged us.

We would like to thank our parents for helping us in doing this project.

## Table of Contents

<b>1. Acknowledgement</b>	<b>2</b>
<b>2. Abstract</b>	<b>4</b>
<b>3. Introduction</b>	<b>5</b>
<b>4. Page Rank</b>	<b>6</b>
4.1. Why it is important? -----	6
4.2. History -----	8
4.3. How Search Engine Works -----	9
<b>5. Markov Chain</b>	<b>11</b>
<b>6. Stochastic Matrix</b>	<b>13</b>
<b>7. Page Rank Algorithm</b>	<b>14</b>
7.1. Principle of Algorithm -----	15
7.2. Simplified Algorithm -----	15
7.3. Procedural Steps -----	16
7.4. Example -----	16
7.5. Calculating Eigen Vector 'I' -----	18
<b>8. How to Handle Dangling node</b>	<b>19</b>
<b>9. Damping Factor</b>	<b>23</b>
<b>10. Mat-lab Implementation</b>	<b>29</b>
<b>11. How do Google Page rank Works</b>	<b>30</b>
11.1. Advantages -----	30
11.2. Disadvantage -----	31
<b>12. Conclusion</b>	<b>31</b>
<b>13. Bibliography</b>	<b>32</b>

# ABSTRACT

The importance of a website is an inherently subjective issue that depends on the interests, knowledge and attitude of the reader. However, much can still be said objectively about the relative importance of websites. This article describes PageRank, a method of objectively and mechanically evaluating web pages that effectively measures people's interest and attention to them. We compare PageRank to an idealized, random internet user. We show you how to efficiently calculate PageRank for a large number of pages. We show you how to apply PageRank to user search and navigation.

**Keywords:** Google, Search Engine, PageRank, Markov chains, power method, convergence, stationary vector, updating Result, Page Ranking.

# INTRODUCTION

Google's success in recent years is strongly related to the delivery of precise search results by an Internet search engine. As the use of search engines increases, it becomes more and more important for website developers and SEO marketers that their website scores well on these various search engines in order to reach their desired audience. Knowledge of page ranking algorithms is essential to optimize your sites and web pages for search engines such as Google and Bing.

For example, if a search engine takes into account how often search terms occur in full text on the Internet, web content should, whenever possible, use the most relevant keywords on its web pages to rank the webpage in the top position. For search engine users, the data on ranking algorithms are additionally necessary to assess the robustness of the results. Accordingly, users should understand the algorithms for estimating the hardness and thus the reliability of the search engine results.

Understanding any ranking algorithm will help website owners estimate the usefulness of the results for their search reason. For example, a user trying to find the latest match score should be shown a result in preference to date, while any user searching for the weather forecast should be shown a result based on nearby locations rather than on random locations.

This document is designed as follows. First, a summary of common search engine ranking algorithms. Then updates were released from time to time to improve the competition of the search results as well as the user experience. And finally, all the results are described. In the concluding section, our observations and interpretation of the results are presented and an overview of future research is suggested.

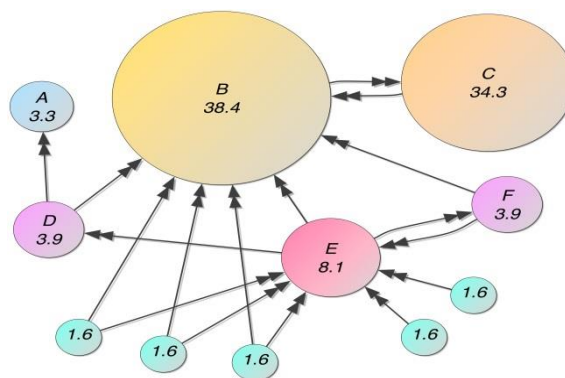
# PAGE RANK

**PageRank (PR)** is an algorithm used by Google Search to rank web pages in their search engine results. PageRank is a way of measuring the importance of website pages.

According to Google: PageRank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

Currently, PageRank is not the only algorithm used by Google to order search results, but it is the first algorithm that was used by the company, and it is the best known.

It is a process that allows for the evaluation of web pages using an algorithm based on their incoming backlink links. The expression "PageRank" originates from Larry Page, who developed this algorithm together with Sergey Brin at Stanford University and patented it in 1997.



## So why is page rank important?

Page rank is important because it's one of the factors a search engine like Google takes into account when it decides which results to show at the top of its search engine listings – where they can be easily seen. (In fact, PageRank is a Google trademark – but other search engines use similar techniques.)

It's not the only or even the most important factor. To start with, your page needs to be relevant to whatever the search is – if someone's searching for plumbers, your page about online banking isn't going to feature no matter how high its page rank may be. But all other things being equal, page rank does affect search engine rankings.

Getting links to your website (link building) is an important part of search engine optimization (SEO). Furthermore, it's quality, not quantity that matters most – you want relevant links from well-known, respected websites.

That's why search engine optimization consultants offer link building services to their clients. Whether you're using a specialist or trying to do it yourself, link building should be part of your SEO strategy.

- ❖ In simple terms, page rank is a measure of how 'important' a web page is. It works on the basis that when another website links to your web page, it's like a recommendation or vote for that web page.
- ❖ Each link increases the web page's page rank. The amount it increases depends on various factors, including how important the voting page is and how relevant it is.
- ❖ For example, suppose you've written a web page about online banking. If one of the major banks decides that you've written a brilliant explanation that it wants its customers to see, they might decide to provide a link to your web page. That link could increase your page rank a lot – because it's from an important and relevant site.
- ❖ By contrast, if your mother decides to include a link from her website of home recipes, that's probably not going to have much effect on your page rank. It's a vote from a less-known website that isn't relevant to the subject.

## History of Page Rank

The big innovation of the late 1990s is the development of search engines, which began with Alta Vista at DEC's Western Research Lab and reached its modern pinnacle with Google, founded by Stanford graduate students Larry Page and Sergey Brin in 1998.

The heart of the Google search engine is the PageRank algorithm, which was described in the paper you read for today's class, written by Larry Page, Sergey Brin, Rajeev Motwani (who drowned in a tragic accident in 2009), and Terry Winograd.



The first PageRank patent was filed on September 1, 1998, and became the original algorithm that Google used to calculate the importance of a web page and rank these.

In short, Google was formed based upon Sergey Brin's idea that information on the web could be ranked based upon a page's link popularity, that the more links point to a page, the higher it ranks.

And if we take a look at the paper that introduced Google, we can see PageRank referenced when explaining the search engine's features:

**"The Google search engine has two important features that help it produce high precision results. First, it makes use of the link**



structure of the Web to calculate a quality ranking for each web page. Second, Google utilizes links to improve search results. “

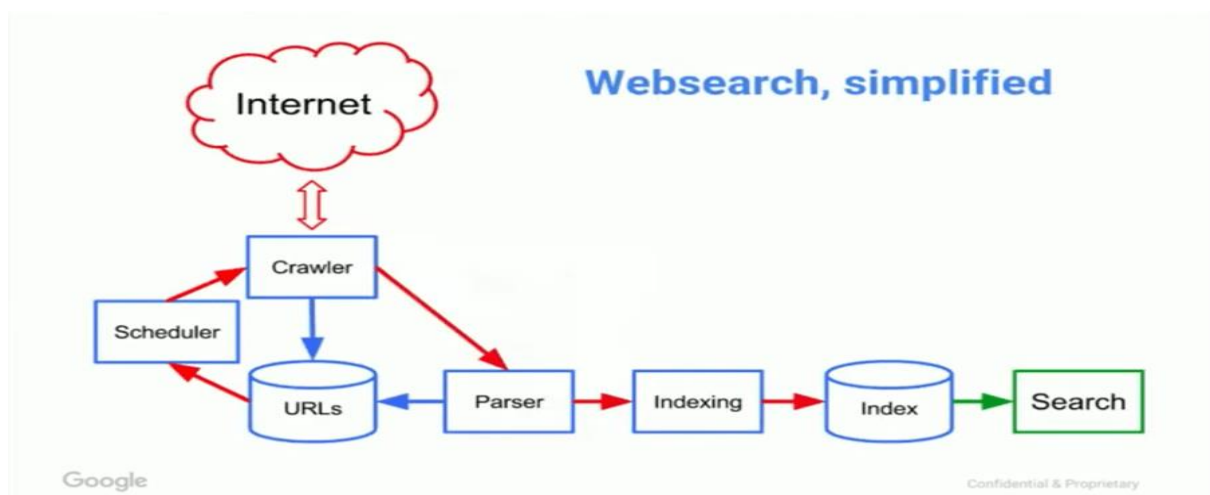
PageRank is literally what made Google so unique. The paper goes on to explain that, "The citation (link) graph of the web is an important resource that has largely gone unused in existing web search engines."

## How do search engines work?

Search engines work by taking a list of known URLs, which then go to the scheduler. The scheduler decides when to crawl each URL. Crawled pages then go to the parser where vital information is extracted and indexed. Parsed links go to the scheduler, which prioritizes their crawling and re-crawling.

When you search for something, search engines return matching pages, and algorithms rank them by relevance.

Here's a diagram from Google showing this process:



We'll cover ranking algorithms shortly. First, let's drill deeper into the mechanisms used to build and maintain a web index to make sure we understand how they work. These are scheduling, crawling, parsing, and indexing.

## **Scheduling:**

The scheduler assesses the relative importance of new and known URLs. It then decides when to crawl new URLs and how often to re-crawl known URLs.

## **Crawling:**

Google searches the web with automated programs called crawlers, looking for pages that are new or updated. Google stores those page addresses (or page URLs) in a big list to look at later. We find pages by many different methods, but the main method is following links from pages that we already know about. When a search engine like Google re-crawls that page, it downloads the content of the page with the recently-added links.

The crawler then passes the downloaded web page to the parser.

## **Parsing:**

The parser extracts links from the page, along with other key information. It then sends extracted URLs to the scheduler and extracted data for indexing.

## **Indexing:**

Indexing is where parsed information from crawled pages gets added to a database called a search index. Google visits the pages that it has learned about by crawling, and tries to analyze what each page is about. Google analyzes the content, images, and video files on the page, trying to understand what the page is about. This information is stored in the Google index, a huge database that is stored on many computers.

## Markov chain

It is a stochastic model that describes the sequence of possible events in which the probability of each event depends only on the state attained in the previous event. A countably infinite sequence, in which the chain moves state at discrete time steps, gives a discrete-time Markov chain. A continuous-time process is called a continuous-time Markov chain. It is named after the Russian mathematician Andrey Markov.



Markov chains has many applications such as **statistical models of real-world processes**, studying **cruise control systems** in motor vehicles, **queues or lines** of customers arriving at an airport, currency exchange rates, and **animal population dynamics**.

Google's random surfer is an example of a Markov process, in which a system moves from state to state, based on probability information that shows the likelihood of moving from each state to every other possible state.

Whenever you give a query on Google, you will get the web pages in an order based on their Page Rank. PageRank algorithm is the one, which is behind this ordering of search results. Google page rank is the objective way of rating web pages. We will compute a vector called PageRank Vector defined as the Eigen Vector of the Google Class matrix.

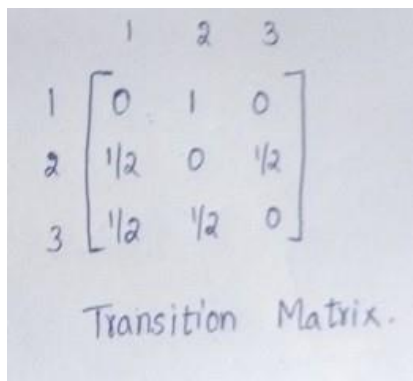
**PageRank Vector:** This is the vector that contains PageRank Values.

Page rank is based on the random surfer. Let's say we are surfing on the internet by clicking on random links. This can be interpreted as a Markov Chain.

Markov Chain helps in predicting the behavior of the system which is in the transition from one state to another by considering only the current state. At each time the system moves from state 'i' to 'j' with probability  $P_{ij}$ .  $P_{ij}$  is called the transition probability. The transition probability helps to find out what is the next state of the object by considering only the current state and not any previous ones.

### A Markov chain contains:

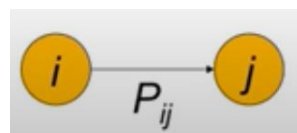
- i) 'N' number of states
- ii) ' $N \times N$ ' matrix formed from transition probability.



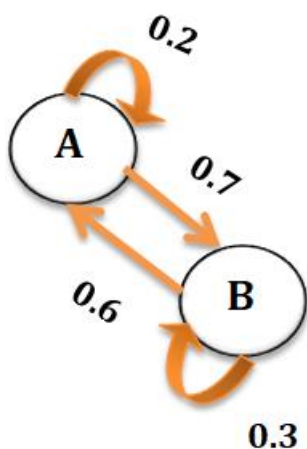
Handwritten transition matrix for a 3-state Markov chain. The states are labeled 1, 2, and 3. The matrix is:

$$\begin{array}{c|ccc}
 & 1 & 2 & 3 \\
 \hline
 1 & 0 & 1 & 0 \\
 2 & 1/2 & 0 & 1/2 \\
 3 & 1/2 & 1/2 & 0
 \end{array}$$

Transition Matrix.



Every matrix entry  $P_{ij}$  in transition probability matrix (T) tells us,  $P_{(j|i)}$  the probability of 'j' is the next state from the current state 'i'.



The diagram represents a two-state Markov process, with the states labelled A and B. Each number represents the probability of the Markov process changing from one state to another state, with the direction indicated by the arrow.

For example, if the Markov process is in state A, then the probability it changes to state B is **0.7**, while the probability it remains in state A is **0.2**.

# Stochastic Matrix

In mathematics, a stochastic matrix is a square matrix used to describe the transitions of a Markov chain. Each of its entries is a non-negative real number representing a probability. It is also called a probability matrix, transition matrix, substitution matrix, or Markov matrix.

A Google matrix is a particular stochastic matrix that is used by Google's PageRank algorithm. The matrix represents a graph with edges representing links between pages.

The PageRank of each page can then be generated iteratively from the Google matrix using the power method. However, for the power method to converge, the matrix must be stochastic, irreducible, and aperiodic.

A square matrix  $A$  is stochastic if all of its entries are nonnegative, and the entries of each column sum to 1. A matrix is positive if all of its entries are positive numbers.

A stochastic matrix is a matrix describing the transitions of a Markov chain. It is also called a Markov matrix.

❖ A **Right Stochastic Matrix** is a real square matrix, with each row summing to 1.

$$\begin{pmatrix} 0.5 & 0 & 0.5 \\ 0.5 & 0.25 & 0.25 \\ 1 & 0 & 0 \end{pmatrix}$$

❖ A **Left Stochastic Matrix** is a real square matrix, with each column summing to 1.

$$\begin{pmatrix} 0.5 & 0.5 & 1 \\ 0.5 & 0.25 & 0 \\ 0 & 0.25 & 0 \end{pmatrix}$$

- ❖ A **Doubly Stochastic Matrix** is a square matrix of non-negative real numbers with each row and column summing to 1.

$$\begin{pmatrix} 0.5 & 0 & 0.5 \\ 0.5 & 0.25 & 0.25 \\ 0 & 0.75 & 0.25 \end{pmatrix}$$

## PAGE RANK ALGORITHM

- ❖ The PageRank algorithm gives each page a rating of its importance, which is a recursively defined measure whereby a page becomes important if important pages link to it. This definition is recursive because the importance of a page refers back to the importance of other pages that link to it.
- ❖ One way to think about PageRank is to imagine a random surfer on the web, following links from page to page. The page rank of any page is roughly the probability that the random surfer will land on a particular page. Since more links go to the important pages, the surfer is more likely to end up there.
- ❖ The behavior of the random surfer is an example of a Markov process, which is any random evolutionary process that depends only on the current state of a system and not on its history.
- ❖ The objective is to estimate the popularity, or the importance, of a webpage, based on the interconnection of the web.
- ❖ The rationale behind it is
  - (i) A page with more incoming links is more important than a page with fewer incoming links,
  - (ii) A page with a link from a page that is known to be of high importance is also important. In this note, we study the convergence of the PageRank algorithm from the matrix's point of view. In practice the number of web pages is huge.

## **Principle of PageRank algorithm**

- ❖ The PageRank algorithm evaluates web pages according to the principle "the more links, the more important the website".
- ❖ Search engines like Google use algorithms as the basis for their rankings. However, today, many more factors influence the ranking of the website, which is why PageRank loses its importance.
- ❖ The PageRank algorithm is based on the number of incoming links and the weighting of the linking page. Irrespective of the content of a page, it is better evaluated when important other pages link to it.
- ❖ The ranking of a site is therefore recursive to the assessment of the linking page. Thus, the complete link structure of the internet is involved.
- ❖ The link following the basis follows the Random Surfer Model - a user, who is randomly surfing the **WWW** and therefore accesses many different pages.

## **Simplified PageRank Algorithm**

The algorithm which results in the probability distribution used to represent the person clicking the links randomly will arrive at any page particularly. It has been calculated for the collections of documents of any size. The PageRank computations require several passes, called "Iterations", through the collection to adjust approximate PageRank values to more closely reflect the theoretical true value.

The connections between pages are represented by a graph. A node represents a webpage and an arrow from page A to page B means that there is a link from page A to page B.

The number of out-going links is an important parameter. We use the notation “out-degree of a node” to stand for the number of out-going links contained in a page. This graph is usually referred to as the web graph.

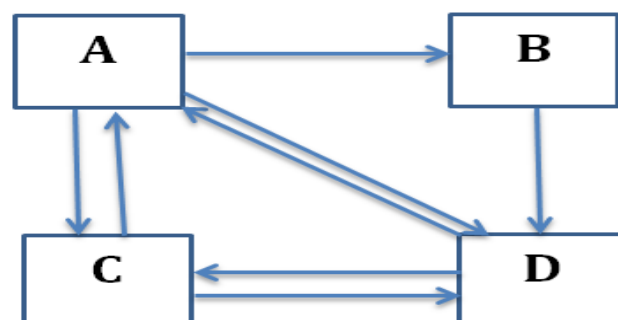
Each node in the graph is identified with a page. We will use the term “node” and “page” interchangeably. And let  $L(p)$  be the number of out-going links on a page  $p$ .

## Procedural Steps

- Start with a set of pages.
- Crawl the web to determine the link structure.
- Assign each page an initial rank of  $1 / N$ .
- Successively update the rank of each page by adding up the weight of every page that links to it divided by the number of links emanating from the referring page.
- Apply this redistribution to every page in the graph.
- Repeat this process until the page ranks stabilize.
- In practice, the Page Rank algorithm adds a damping factor at each stage to model the fact that users stop searching.

### Example1:

There are four pages. Page A contains a link to page B, a link to page C, and a link to page D. Page B contains one single link to page D. Page C points to pages A and D, and page D points to pages A and C. They are represented by the following graph. We have  $L(A) = 3$ ,  $L(B) = 1$  and  $L(C) = L(D) = 2$ .





Let  $N$  be the total number of pages. We create an  $N \times N$  matrix  $A$  by defining the  $(i, j)$ -entry as

$$a_{ij} = \begin{cases} \frac{1}{L(j)} & \text{if there is a link from } j \text{ to } i, \\ 0 & \text{otherwise.} \end{cases}$$

In the given example, matrix  $A$  is the  $4 \times 4$  matrix:

$$\begin{bmatrix} 0 & 0 & 1/2 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1 & 1/2 & 0 \end{bmatrix}$$

Note that the sum of the entries in each column is equal to 1. In general, a matrix is said to be column-stochastic if the entries are non-negative and the sum of the entries in each column is equal to 1.

The matrix  $A$  is by design column-stochastic, provided that each page contains at least one out-going link.

The simplified PageRank algorithm is: Initialize  $x$  to an  $N \times 1$  column vector with non-negative components, and then repeatedly replace  $x$  with the product  $Ax$  until it converges. We call the vector  $x$  the PageRank vector.

Usually, we initialize it to a column vector whose components are equal to each other. We can imagine a bored surfer who randomly clicks the links. If there are  $k$  links on the page, he simply picks one of the links randomly and goes to the selected page.

After a sufficiently long time, the  $N$  components of the PageRank vector are directly proportional to the number of times this surfer visits the  $N$  web pages. For example, let the components of the vector  $x$  be  $x_A$ ,  $x_B$ ,  $x_C$ , and  $x_D$ .

**Initialize  $\mathbf{x}$  to be the all-one column vector,**

$$\mathbf{x} = \begin{bmatrix} x_A \\ x_B \\ x_C \\ x_D \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The evolution of the PageRank vector is shown in the following table,

iteration	$x_A$	$x_B$	$x_C$	$x_D$
0	1	1	1	1
1	1	0.3333	0.8333	1.8333
2	1.3333	0.3333	1.25	1.0833
3	1.667	.4444	.9861	1.4028
4	1.1944	0.3889	1.0903	1.3264
5	1.2083	0.3981	1.0613	1.3322
6	1.1968	0.4028	1.0689	1.3316
7	1.2002	0.3989	1.0647	1.3361

We observe that the algorithm converges quickly in this example. Within 7 iterations, we can see that page D has the highest rank. Page D has 3 incoming links, while the others have either 1 or 2 incoming links. It conforms to the rationale of the PageRank algorithm that a page with larger incoming links has higher importance.

## Calculating the Eigenvector ‘ $\mathbf{I}$ ’

There are different ways of calculating the eigenvectors. The research papers show that on average a web page has 10 links going out; meaning almost all but 10 entries in each column is 0.

Let us consider the power method for calculating the eigenvector. In this method, we begin by choosing a vector  $\mathbf{I}^{(0)} = (\mathbf{1}, \mathbf{1}, \mathbf{1}, \dots, \mathbf{1})'$  as a candidate for  $\mathbf{I}$  and then produce a sequence of vectors  $\mathbf{I}^{(k)}$  such that

$$\mathbf{I}^{(k+1)} = \mathbf{H}\mathbf{I}^{(k)}$$

There are issues regarding the convergence of the sequence of vectors  $\mathbf{I}^{(n)}$ . The matrix under consideration must satisfy certain conditions.

For the web described in Example 1, if  $\mathbf{I}^{(0)} = (1, 1, 1, 1)'$  power method shows that

$$\mathbf{I}^{(0)} = (1, 1, 1, 1)'$$

$$\mathbf{I}^{(1)} = (1, 0.3333, 0.8333, 1.8333)'$$

$$\mathbf{I}^{(2)} = (1.3333, 0.3333, 1.25, 1.0833)'$$

$$\begin{bmatrix} - \\ - \\ - \end{bmatrix}$$

$$\mathbf{I}^{(6)} = (1.1968, 0.4028, 1.0689, 1.3316)'$$

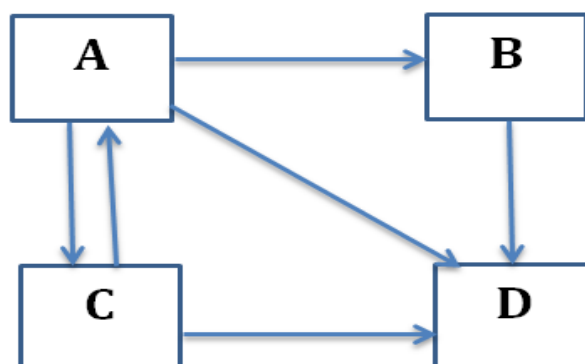
$$\mathbf{I}^{(7)} = (1.2002, 0.3989, 1.0647, 1.3361)'$$

These numbers give us the relative measures for the importance of pages. Hence we multiply all the popularities by a fixed constant to get the sum of popularities equal to 1.

## How to handle dangling node

A node is called a dangling node if it does not contain any out-going link, in the sense if the out-degree is zero.

For instance, node D in the below example is a dangling node.



In this example, all other web pages except D have an out-going link. So the web page D is said to be the dangling node.

**The associated matrix A is:**

$$\begin{bmatrix} 0 & 0 & 1/2 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1 & 1/2 & 0 \end{bmatrix}$$

We note that the entries in the last column are all zero; hence this matrix is not column-stochastic. The simplified PageRank algorithm collapses if there is a dangling node in the web graph. As an example, if we initialize the vector  $x$  to the all-one vector, the simplified PageRank algorithm gives

iteration	$x_A$	$x_B$	$x_C$	$x_D$
0	1	1	1	1
1	0.5	0.3333	0.8333	1.8333
2	0.1667	0.1667	0.1667	0.6667
3	0.0833	0.0556	0.0556	0.3056
4	0.0278	0.0278	0.0278	0.1111
5	0.0139	0.0093	0.0098	0.0509

The PageRank vector will converge to zero ultimately. One remedy is to modify the simplified PageRank algorithm by replacing any all-zero column by

$$\begin{bmatrix} 1/N \\ 1/N \\ \vdots \\ 1/N \end{bmatrix}.$$

This adjustment is justified by modeling the behavior of a web surfer, who after reading a page with no out-going link; will jump to a random page. He simply picks one of the N pages randomly with equal probability. This model may not be realistic, but it simplifies the computation of the algorithm. In matrix format, we create a matrix, whose column is the same as A except for the columns corresponding to the dangling pages. Formally, we define the entries of the matrix  $\bar{A}$  by

$$\bar{a}_{ij} = \begin{cases} \frac{1}{L(j)} & \text{if there is a link from node } j \text{ to node } i, \\ \frac{1}{N} & \text{if node } j \text{ is a dangling node,} \\ 0 & \text{otherwise,} \end{cases}$$

Where 'N' is the total number of pages.

The matrix  $\bar{A}$  is:

$$\begin{bmatrix} 0 & 0 & 1/2 & 1/4 \\ 1/3 & 0 & 0 & 1/4 \\ 1/3 & 0 & 0 & 1/4 \\ 1/3 & 1 & 1/2 & 1/4 \end{bmatrix}.$$

In the PageRank algorithm, we use  $\bar{A}$  instead of A and replace  $x$  by  $\bar{A} x$  instead. In the example, if we use  $\bar{A}$ , we get

iteration	$x_A$	$x_B$	$x_C$	$x_D$
0	1	1	1	1
1	0.75	0.5833	0.5833	2.0833
2	0.8125	0.7708	0.7708	1.6458
3	0.7969	0.6823	0.6823	1.8385
4	0.8008	0.7253	0.7253	1.7487
5	0.7998	0.7041	0.7041	1.7920

We can see that page D has the highest ranking.

### **Example 2:**

Assume an example of four web pages: **A, B, C, and D.**

Links from a page to itself are ignored. Multiple outbound links from one page to another page are treated as a single link. PageRank is initialized to the same value for all pages.

In the original form of PageRank, the sum of PageRank over all pages was the total number of pages on the web at that time, so each page in this example would have an initial value of **1**. However, later versions of PageRank, and the remainder of this section, assume a probability distribution between **0 and 1**. Hence the initial value for each page in this example is **0.25**.

The PageRank transferred from a given page to the targets of its outbound links upon the next iteration is divided equally among all outbound links.

If the only links in the system were from pages **B, C, and D to A**, each link would transfer **0.25** PageRank to **A** upon the next iteration, for a total of **0.75**.

$$\mathbf{PR(A) = PR(B) + PR(C) + PR(D)}$$

Suppose instead that page **B** had a link to pages **C and A**, page **C** had a link to page **A**, and page **D** had links to all three pages. Thus, upon the first iteration, page **B** would transfer half of its existing value, or **0.125**, to page **A** and the other half, or **0.125**, to page **C**. Page **C** would transfer all of its existing value, **0.25**, to the only page it links to, **A**. Since **D** had three outbound links, it would transfer one-third of its existing value, or approximately **0.083**, to **A**.

After this iteration, page **A** will have a PageRank of approximately **0.458**.

$$PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3}$$

In other words, the PageRank conferred by an outbound link is equal to the document's PageRank score divided by the number of outbound links  $L()$ .

$$PR(A) = \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)}$$

The PageRank value for any page  $u$  can be expressed as:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)},$$

Where the PageRank value for a page  $u$  is dependent on the PageRank values for each page  $v$  contained in the set  $B_u$  (the set containing all pages linking to page  $u$ ), divided by the number  $L(v)$  of links from page  $v$ .

## Damping Factor

The PageRank theory holds that an imaginary surfer who is randomly clicking on links will eventually stop clicking. The probabilities, at any step, that the person will continue is a damping factor  $d$ . various studies have tested different damping factors, but it is generally assumed that the damping factor will be set around 0.85.

The damping factor is subtracted from 1 (and in some variations of the algorithm, the result is divided by the number of documents ( $N$ ) in the collection) and this term is then added to the product of the damping factor and the sum of the incoming PageRank scores. That is,

$$PR(A) = \frac{1-d}{N} + d \left( \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \dots \right)$$

Where  $L(X)$  represents the number of outgoing links from page  $X$ . In the above equation, it is assumed that pages  $B, C, D$ , etc. all include exactly one link to page  $A$ .

The damping factor  $d$ , which is the click-through probability, is included to prevent sinks from "absorbing" the Page Ranks of those pages connected to the sinks. It is easy to see that an infinite surfer would have to end up in a sink given enough time, so the damping factor allows a heuristic to offset the importance of those sinks. That is why the first term of the PageRank equation,  $(1-d)/N$ , is included. It is the chance of being on a random page after a restart, while the second term is normalized so that all PageRank's sum to one.

You can see that if  $d=1$ , the person clicking will click forever and they will always end up in a sink. In this case, you discard the first term. The second term, given an infinite number of iterations to convergence, is equivalent to finding the steady-state of the Markov chain representing pages and links.

On the other hand, if the click-through probability is  $d=0$ , then all clicks are random restarts, which are uniformly distributed by definition. So, a damping factor  $0 < d < 1$  is a sort of weighted average between the two extremes.

The damping factor adjusts the derived value downward. The original page rank formula with the summation:

$$PR(A) = 1 - d + d \left( \frac{PR(B)}{L(B)} + \frac{PR(C)}{L(C)} + \frac{PR(D)}{L(D)} + \dots \right)$$

Where,

- ❖  $PR(A)$  is the page rank of  $A$ ; it is a kind of recursive formula because it depends on other page's page rank.
- ❖  $PR(B)$  is the page rank of page  $B$  which links to page  $A$ .
- ❖  $L(B)$  number of out-bounds links on a given  $B$  page.



❖ D is the damping factor in the range 0 and 1.

We have to initialize the page rank at the beginning; all pages are given equal page rank  $\frac{1}{N}$ , where N is the number of pages.

That's why we have to make several iterations until convergence.

The difference between them is that the PageRank values in the first formula sum to one, while in the second formula each PageRank is multiplied by N and the sum becomes N.

A statement in Larry Page and Brin's paper that "**the sum of all PageRank's is one**" and claims by other Google employees support the first variant of the formula above.

Page and Brin confused the two formulas in their most popular paper "**The Anatomy of a Large-Scale Hypertextual Web Search Engine**", where they mistakenly claimed that the latter formula formed a probability distribution over web pages.

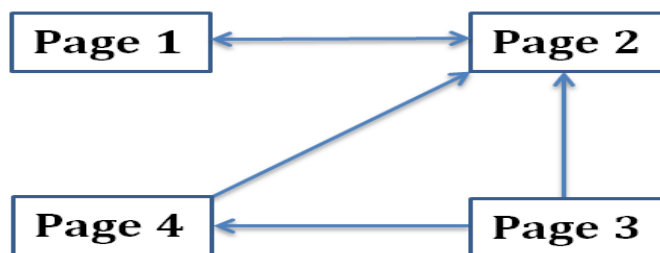
Google recalculates PageRank scores each time it crawls the Web and rebuilds its index. As Google increases the number of documents in its collection, the initial approximation of PageRank decreases for all documents. When calculating PageRank, pages with no outbound links are assumed to link out to all other pages in the collection. Their PageRank scores are therefore divided evenly among all other pages. In other words, to be fair with pages that are not sinking, these random transitions are added to all nodes in the Web. This residual probability, d, is usually set to 0.85, estimated from the frequency that an average surfer uses his or her browser's bookmark feature. So, the equation is as follows

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

Where  $p_1, p_2, \dots, p_N$  are the pages under consideration,  $M(p_i)$  is the set of pages that link to  $p_i$ ,  $L(p_j)$  is the number of outbound links on page  $p_j$ , and  $N$  is the total number of pages.

### **Example 3:**

The WWW hyperlink forms a huge directed graph where the nodes represent web pages + directed edges are the hyperlinks.



This is an example, if we need to find the page rank of page 2, we need to calculate the incoming link's from other pages and divide by the number of out-going links to the respective page.

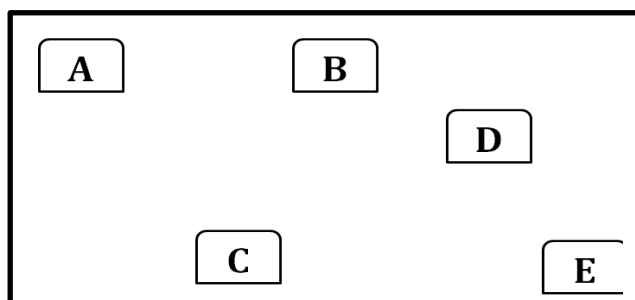
Here on page 2, the incoming links are from pages 1, 4, and 3. The out-going links of page 1 are 1; from page 4 is 1 and from page 3 is 2;

So the final page rank of Page 2 will be

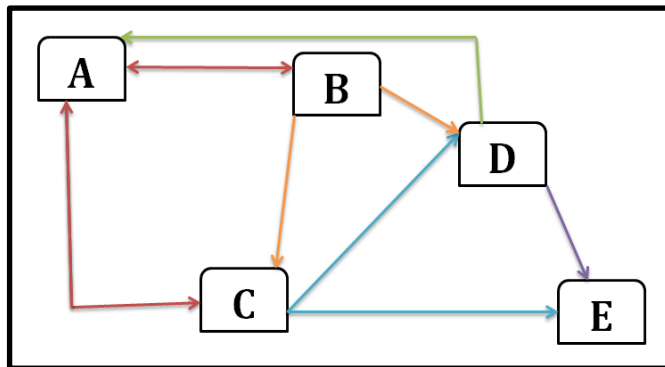
$$\mathbf{PR\ (2)} = \frac{\mathbf{PR(1)}}{\mathbf{1}} + \frac{\mathbf{PR(4)}}{\mathbf{1}} + \frac{\mathbf{PR(3)}}{\mathbf{2}}$$

### **Example 4:**

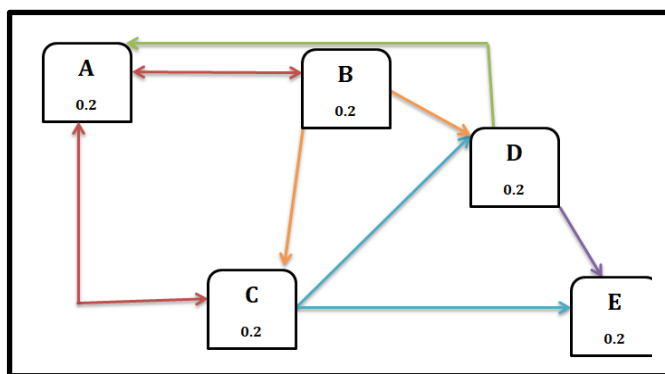
1. Start with a set of pages.



2. Crawl the web to determine the link structure.



3. Assign each page an initial rank of  $\frac{1}{N}$ .



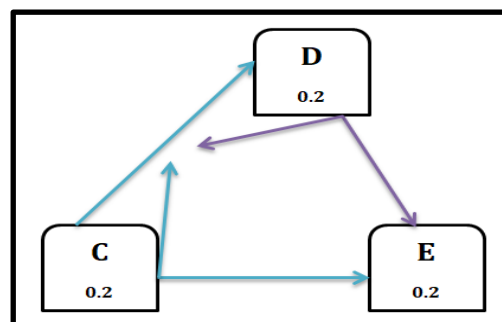
4. Successively update the rank of each page by adding up the weight of every page that links to it divided by the number of links emanating from the referring page.

❖ In the current example, page E has two incoming links, one from page C and one from page D.

❖ Page C contributes  $\frac{1}{3}$  of its current page rank to page E because E is one of three links from page C. Similarly, page D offers  $\frac{1}{2}$  of its rank to E.

❖ The new page rank for E is

$$\text{PR}(E) = \frac{\text{PR}(C)}{3} + \frac{\text{PR}(D)}{2} = \frac{0.2}{3} + \frac{0.2}{2} = 0.17$$

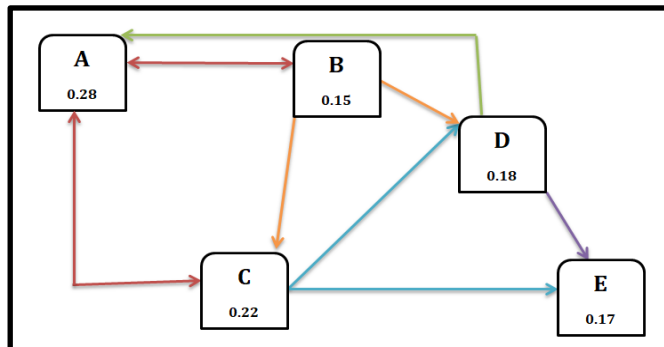


5. If a page (such as E in the current example) has no outward links, redistribute its rank equally among the other pages in the graph.

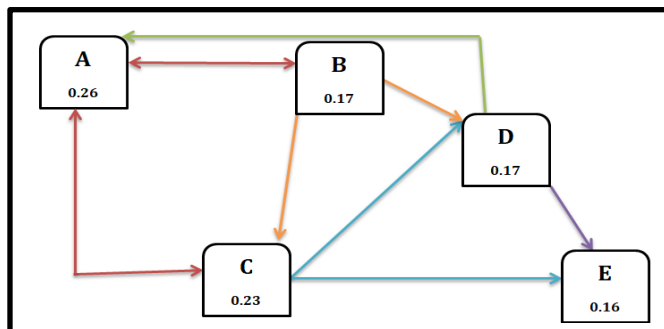
❖ In this graph,  $\frac{1}{4}$  of E's page rank is distributed to pages A, B, C, and D.

❖ The idea behind this model is that users will keep searching if they reach a dead end.

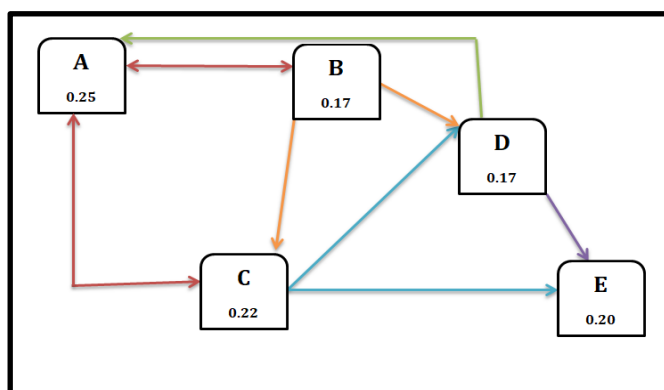
6. Apply this redistribution to every page in the graph.



7. Repeat this process until the page ranks stabilize.



8. In practice, the Page Rank algorithm adds a damping factor at each stage to model the fact that users stop searching.



## Mat-lab Implementation

```

s = [1 1 2 2 2 3 3 3 4 4];
t = [2 3 1 3 4 1 4 5 1 5];
names = {'A', 'B', 'C', 'D', 'E'};
G = digraph(s,t,[],names)

H=plot(G,'Layout','layered', 'NodeLabel',{'A','B','C','D', 'E'}, 'LineWidth',1)

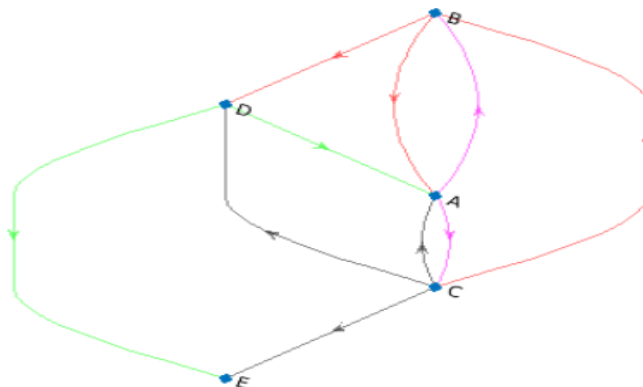
highlight(H,[1 1],[2 3],'EdgeColor', 'm')
highlight(H,[2 2 2],[1 3 4],'EdgeColor','r')
highlight(H,[3 3 3],[1 4 5],'EdgeColor','black')
highlight(H,[4 4],[1 5],'EdgeColor','g')

PR = centrality(G,'pagerank','FollowProbability',0.85)
G.Nodes.PageRank = PR;
G.Nodes.InDegree = indegree(G);
G.Nodes.OutDegree = outdegree(G);
G.Nodes

```

G =  
digraph with properties:

Edges: [10x1 table]  
Nodes: [5x1 table]



PR = 5x1  
0.2457  
0.1681  
0.2157  
0.1724  
0.1981

ans = 5x4 table

	Name	PageRank	InDegree	OutDegree
1	'A'	0.2457	3	2
2	'B'	0.1681	1	3
3	'C'	0.2157	2	3
4	'D'	0.1724	2	2
5	'E'	0.1981	2	0

## How does Google PageRank work?

We assume page A has pages  $T_1 \dots T_n$  which point to it. The parameter  $d$  is a damping factor that can be set between 0 and 1. We usually set  $d$  to 0.85. There are more details about  $d$  in the next section. Also,  $C(A)$  is defined as the number of links going out of page A. The PageRank of page A is given as follows:

$$PR(A) = (1 - d) + d \left( \frac{PR(T_1)}{C(T_1)} + \dots + \frac{PR(T_n)}{C(T_n)} \right)$$

Note that the PageRank's form a probability distribution over web pages, so the sum of all web pages PageRank's will be one.

Google mainly consider three factors when calculating the PageRank of a web page, which are:

- ❖ The quantity and quality of inbound linking pages;
- ❖ The number of outbound links on each linking page;
- ❖ The PageRank of each linking page.

## Advantages of PageRank Algorithm

- Since it pre computes the rank score it takes less time and hence it is fast.
- It is more feasible as it computes rank score at indexing time, not at query time.
- It returns important pages as Rank is calculated based on the popularity of a page.
- The algorithm is robust against Spam since it's not easy for a webpage owner to add links to his page from other important pages.
- PageRank is a global measure and is query independent.
- PageRank gives the information very efficiently according to the topmost ranking of the webpage.

## **Disadvantages of PageRank Algorithm**

- The main disadvantage is that it favors older pages, because a new page, even a very good one, will not have many links unless it is part of an existing web site.
  - The relevancy of the resultant pages to the user query is very less as it does not consider the content of the web page.
  - Other problems exist in the form of Dangling links which occurs when a page contains a link such that the hypertext points to a page with no outgoing links.
  - It leads to the Rank sinks problem that occurs when in a network pages get in infinite link cycles.
  - Dead Ends are possible pages with no outgoing links.
- Another problem in PageRank is Spider Traps. A group of pages is a spider trap if there are no links from within the group to outside the group.
- If you have circle references on your website, then it will reduce your front page's PageRank.

## **Conclusion**

PageRank is a global ranking of all websites based on their location in the structure of web graphs. PageRank uses information external to websites - backlinks. Backlinks from important pages are more important than backlinks from average pages. The structure of the web graph is very useful for information retrieval tasks. PageRank has left its mark on the world by helping Google become a search giant and remains part of its search algorithm.

However, be aware that there are hundreds of other ranking factors out there.

What determines a higher page ranking is a combination of factors that may be

different for each page. It makes no sense to focus on just one factor. Instead, take a holistic approach to SEO efforts to find the optimal medium.

## **Bibliography**

- ❖ [https://en.wikipedia.org/wiki/PageRank#:~:text=PageRank%20\(PR\)%20is%20an%20algorithm,how%20important%20the%20website%20is.](https://en.wikipedia.org/wiki/PageRank#:~:text=PageRank%20(PR)%20is%20an%20algorithm,how%20important%20the%20website%20is.)
- ❖ <https://www.mathworks.com/content/dam/mathworks/mathworks-dot-com/moler/exm/chapters/pagerank.pdf>
- ❖ <https://in.mathworks.com/help/matlab/math/use-page-rank-algorithm-to-rank-websites.html>
- ❖ <http://home.ie.cuhk.edu.hk/~wkshum/papers/pagerank.pdf>
- ❖ <https://personalpages.manchester.ac.uk/staff/yanghong.huang/teaching/MAT H36032/html/labdemo16.html>
- ❖ <https://personalpages.manchester.ac.uk/staff/yanghong.huang/teaching/MAT H36032/html/labdemo16.html>
- ❖ <https://www.google.com/search/howsearchworks/algorithms/>
- ❖ <https://towardsdatascience.com/pagerank-algorithm-fully-explained-dc794184b4af>
- ❖ [https://en.ryte.com/wiki/Page\\_Rank](https://en.ryte.com/wiki/Page_Rank)