Google PageRank Algorithm

Author: Satchit Dahal, Chris Ding, and Ben Valdebenito

TMATH 308 Matrix Algebra with Application

June 5, 2020

Abstract

In this paper, we will explain how Google's famous search algorithm known as the Page Rank Algorithm works. Aside from presenting and explaining the equation, we will also be using matrix algebra to provide various examples in an attempt to dive deep into the algorithm and provide a better insight about its core functionality. This paper hopes to help people understand the simplicity and the complexity of the search engine, Google.It is intended to provide a glimpse of the blueprints and behind the scenes technicalities that happen when users type in a query on Google and get valid results in less than a second.

Introduction

Google has become the "go-to" for all of people's queries and has assumed the role of the gateway that welcomes people to the Internet. It has been able to provide millions of search-related answers in a fraction of a second. So how does Google know which result is the most relevant to our particular query? It's thanks to the Page Rank algorithm.

In its beginning, Google wasn't as reliable as it is now. People could still use google and get instant results but those results wouldn't be the most relevant. Thanks to the new algorithm, we don't have to worry about getting anything but the best from Google. Google is the most reliable and fastest search engine in the world amongst other search engines such as Yahoo and Bing. Google's page rank algorithm is the magic behind it's lightning fast search results. However, speed isn't the only factor that makes Google the best of its kind. It is the relevance of the results that makes it stand out.

At its simplest, the page rank algorithm is a mathematical formula that google uses to rank how important a page is based on the search query. The famous formula is,

$$\boldsymbol{PR(P) = (1 - d) + d(PR\frac{T1}{C}(T1) + \ldots. + PR(Tn)/C(Tn))}$$

For further understanding of the formula, below is an index.

P	Page that goes into the formula.
(Tn)	The incoming links to a page.
C	The outgoing links from a page.
d	A damping factor that can be set between 0 and 1.

This formula calculates the rank of the page that is being evaluated and ultimately determines which pages to put on the top of the search results. Simply put, the rank of a page is based on the sum of the number of links coming into that page divided by the sum of the number of links going out from that page. For a better insight, we looked at the formula a bit closely.

In this formula, P refers to the page that is the input into the PR formula. Therefore, PR is a function of Page.

T of n is a measure of the inbound link coming into a page. The more inbound links coming into a page, the more value that page has. Here, n is a positive integer that represents the number of the link. For example, n = 1 refers to the first link that comes into the page, n = 100 represents the 100th incoming link.

C is a measure of how many outbound links are going from one page to another. For example, let’s consider three webpages, A, B and C. Assume that there is one link going from A to page B and one link coming in from page B to page A. Lastly, let’s assume C has two links coming in. Here, page C would have a value higher than A or B since it has the highest number of links coming in and the least number of links going out.

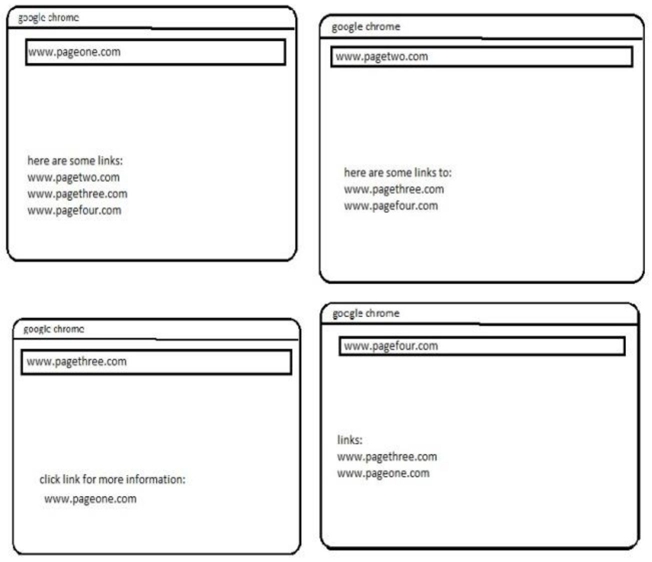
How the algorithm knows where to start is due to what’s called a random surfer. It chooses web pages randomly. Web pages that had a lot of incoming links were more likely to be chosen by the random surfer. Since it is random, there is also a possibility that it might start all over again from a completely different random page. To work around this problem a damping factor, d was introduced into the equation.
D is the probability at which the surfer might choose to follow another link. The value of d can be anywhere from 0 - 1, however the preferred value for d is set to 0.85.

History

Before we do a deep dive into what and how Pagerank works, let's take a look at how things were done before Google's amazing pagerank algorithm was put into use. Back in the 90s, most search engines would use text based ranking systems. To decide what query would be placed in the very top and what would be placed at the very bottom. At first glance, this seems like a reasonable approach. You write something in the search bar that you're interested in knowing more about and Google searches through all queries (articles) that have the most instances of those words. For example, if you want to learn more about the amusement park “Disney World", you could place that in the searchbar and the search engines back in the days would find the articles with the most instances of the word “Disney World” and place those at the very top. An issue that quickly arose from this type of search method is that an article published by Disney with lots of informative details of the parks may not always appear at the very top. That is because they may not be mentioning “Disney World" as often as a poem who mentions “Disney World” in every sentence. This would cause queries that arent the best suitable website to show up before better qualified sites to give out information on certain topics. Another issue that arose, was that companies began “hacking” the previous algorithm. Companies realized they could use third party companies to build thousands of links connecting to certain keywords to make them the top choice. Due to all of this, Google decided to remove the PageRank indicator from the toolbar and improve their algorithm to the following.

Common Model

The PageRank algorithm outputs a probability distribution based on whether a person who randomly clicks a link will reach any particular page. To better understand this algorithm and combine it with what we learned in class, suppose we have an isolated small internet system of just 4 web pages: www.pageone.com (website1), www.pagetwo.com (website2), www.pagethree.com (website3), and www.pagefour.com (website4), which link to each other, relations shown as image below: (Bryan, 2006)



here are some links:
www.pagetwo.com
www.pagethree.com
www.pagefour.com

here are some links to:
www.pagethree.com
www.pagefour.com

click link for more information:
www.pageone.com

links:
www.pagethree.com
www.pageone.com

Figure i: website connections

In order to get a better view of the algorithm, we set up our own model of four different nodes, one for each web site. When site A links to site B, we add a “vector” from A to B in the graph. (Tanase, 2009)
For example, in figure 1, website1 links to all of the other websites 2,3, and 4. Therefore, in the diagram below, node 1 has 3 outgoing “vectors” to all these nodes.

Similarly, website 3 has only one link going to website 1, therefore only one vector goes from node 3 to node 1.
Here is the diagram we get from the image above:

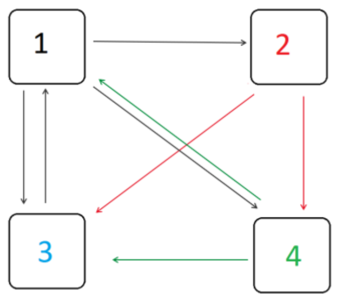


Figure ii
In our model, every incoming link in the website should be identically important. It is like a president election; every vote needs to be identical to be fair. Thus, according to Figure ii, $x_1 = 2, x_2 = 1, x_3 = 3, x_4 = 2$. However, by this calculation, it is unfair for website one who has the same amount of incoming links as website four but more outgoing links in real life scenario, a website might have hundreds of links going out much more than links coming in.(Bryan, 2006) It will make websites like msn.com which has more outgoing links than incoming, less important than the website they link to. So, instead of adding the amount of incoming links, let each website only have “one vote”, and all these links within the website share this “one vote”. In the other word, ratio between number of links outgoing from page m to page n to the total number of outgoing links of page m. (“PageRank.”) For instance, there are 2 links coming out of website 2, so each link weights ½; Node 3 has only one outgoing link, so it will pass 1 to website 1. In conclusion, if there are n links coming out of website A, we say each link weights 1/n. Here is the graph after improvement: (Tanase, 2009)

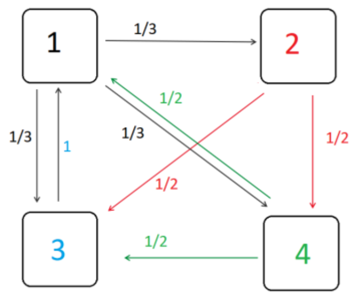


Figure iii

From Figure iii, we can get $x_1 = x_3 + \frac{x_4}{2}$ $x_2 = \frac{x_1}{3}, x_3 = \frac{x_1}{3} + \frac{x_2}{2} + \frac{x_4}{2} x_4 = \frac{x_1}{2} + \frac{x_2}{2}$. Assume we have a modified adjacency matrix resized so that each column adds up to one. The

PageRank values are eigenvector R of the matrix, for this particular example: ("PageRank.")

$$R = \begin{bmatrix} PR(P_1) \\ PR(P_2) \\ PR(P_3) \\ PR(P_4) \end{bmatrix} = \begin{bmatrix} (1-d)/N \\ (1-d)/N \\ (1-d)/N \\ (1-d)/N \end{bmatrix} + d \begin{bmatrix} i_1 j_1 & i_2 j_1 & i_3 j_1 & i_4 j_1 \\ i_1 j_2 & i_2 j_2 & i_3 j_2 & i_4 j_2 \\ i_1 j_3 & i_2 j_3 & i_3 j_3 & i_4 j_3 \\ i_1 j_4 & i_2 j_4 & i_3 j_4 & i_4 j_4 \end{bmatrix} R_i$$

Ri is the initial probability for each node. $i_n \; j_m$ indicate weights of links, n is the start node and m is the end node. For instance: (Bryan, 2006)

$$R = \begin{bmatrix} (1-0.85)/4 \\ (1-0.85)/4 \\ (1-0.85)/4 \\ (1-0.85)/4 \end{bmatrix} + 0.85 \begin{bmatrix} 0 & 0 & 1 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix}$$

And by then solving this equation we get the following,

$$R = \begin{bmatrix} 0.35625 \\ 0.10833 \\ 0.32083 \\ 0.21458 \end{bmatrix}$$

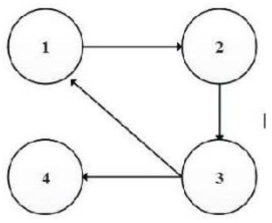
as the solution for the model we have built so far. Vector v indicates that the votes for website 1 is 0.35625 instead of 1 and for website 3 it's 0.32083 instead of 3. Compared to the previous calculation, it ranks website 1 higher than 3, which is more reasonable since the link from website 3 is the only link it comes out.

Dangling Node

A risk to the method described above is, what occurs when we reach a website with no outgoing links? The process it would follow in figure 3 would be that it would give website 4 an importance score of 0. This will later go on to change our importance scores for website 3 which is connected to website 4 which will then go on to change the ones connected to those and so on. At the end, we will end up with having all websites have an importance score of 0 which gives us no meaningful solution.

A visual representation of this would be the following:

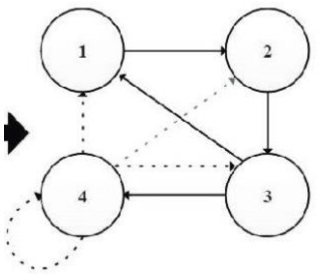
Example 2:



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

https://www.researchgate.net/figure/The-procedure-to- handle-dangling-
nodes_fig1_321056905

Even though 0 is a possible answer, it is not one we would like to see. For that reason, we will make it so its importance is neutral and non affecting others but non zero at the same time. To do this, what we want to do is to assume that website 4 is pointing to every website including itself. By doing this, our random surfer will be able to leave website 4 and continue on its journey on visiting other websites.



https://www.researchgate.net/figure/The-procedure-to-handle-dangling-nodes_fig1_321056905

Conclusion

In the late 1900’s, pageRank algorithms may have seemed like something impossible. Something that wasn’t going to be available in their lifetime. This all comes to show how useful matrices are. Matrices in this example improved speed and accuracy on information that is valuable to normal people that are trying to help grow their business, plan a trip or even learn more about a certain topic by doing a quick google search. In general, we discussed how PageRank was developed in history, and what makes up the famous formula. We also researched the pagerank algorithm and matrix algebra behind it by setting up a common model and applying it. Furthermore, we extend dangling nodes as a specific situation, and the solution with it.

Bibliography

Bryan, Kurt, and Tanya Leise. “The $25,000,000,000 Eigenvector: The Linear Algebra behind Google.” SIAM Review, August 3, 2006. https://doi.org/10.1137/050623280.

“PageRank.” Wikipedia. Wikimedia Foundation, May 22, 2020. https://en.wikipedia.org/wiki/PageRank.

Tanase, Raluca, and Remus Radu. “PageRank Algorithm - The Mathematics of Google Search.” PageRank Algorithm - The Mathematics of Google Search..2009. http://pi.math.cornell.edu/~mec/Winter2009/RalucaRemus/Lecture3/lecture3..

“Google Pagerank, Simplified: A Guide For Seo Beginners”. Dave Davis, February 25, 2020
https://www.searchenginejournal.com/google-pagerank-explained/350630/#close

“Pagerank Explained Correctly with Examples.” Princeton University. The Trustees of Princeton University. Accessed June 6, 2020.
https://www.cs.princeton.edu/~chazelle/courses/BIB/pagerank.htm.