Google's PageRank Algorithm

Angela Yang, Darin Ershov, Stanley Yang

June 2023

1 Introduction

Throughout the last few decades, extensive research has been conducted about the mathematics behind ranking. Numerous ranking methods are created and widely applied to various fields like sports. Examples of these ranking methods include the Massey ratings and Colley matrix used by the Bowl Championship Series to rank football teams [BCS 2011]. However, among all of these methods, one of the most famous and popular ranking algorithms today is the Google PageRank algorithm, which has revolutionized the way search engines sort and organize search results.

The Google PageRank algorithm is incredibly sufficient and powerful that it plays a crucial role in helping Google become the "leading search engine on the internet" [Zack et al. 2012]. The algorithm uses multiple factors to determine the ranking of webpages, one of which is Google's *PageRank score*, a score that Google assigns to webpages based on their usefulness and relevance.

The PageRank score of a webpage is not determined by evaluating the actual contents written in the webpage, but is rather influenced by other webpages that link to it. Webpages that either have lots of other webpages link to them, or have another highly ranked webpage link to them, will be considered "important" [Zack et al. 2012]. Such important webpages are viewed to have a high possibility to be accessed by a random web surfer, and will therefore be assigned a high PageRank score.

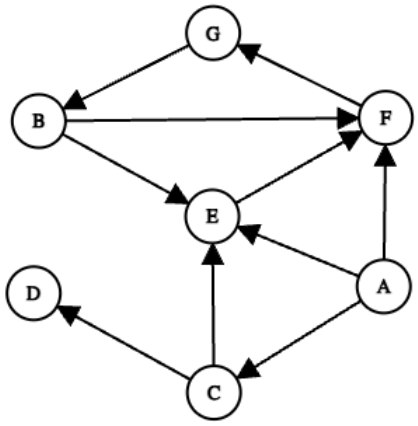
The main idea behind the algorithm is to consider how a modeled random web surfer will follow the link from one webpage to another while looking for information. By considering this "jumping to another source" behavior and the relationship between webpages, Google is able to pick out the most valuable and relevant sources. These sources, with a high PageRank score, will in principle be prioritized and put onto the top of one's search result.

The significance of Google's PageRank algorithm is that it greatly saves the users' time

when they are browsing for useful information. By providing users with a list of search results sorted in the order of their relevance, users are able to quickly and easily "filter out" non-useful sources and information, which considerably improves the user experience. This is one of the biggest reasons behind Google's popularity and dominance of the search engine market.

2 Mathematics Behind the Algorithm

Imagine we have a small network, with 7 webpages labeled A–G linked to each other. This is denoted by the graph below.



For example, page A is linked to pages C, E, and F, while page B is linked to E and F. Additionally, D does not point to anything. Note that these links are unidirectional. This means that from page A, one can then directly access pages C, E, and F by clicking the respective hyperlinks. For D, it is a dead-end, and normally it would stop there.

This graph is called a *directed graph*. All webpages are known as *nodes* and the *edges* between them are *links*. For cases like D, it is a *dangling node* – the node without any outgoing links.

To represent this directed graph in linear algebra, we use an *adjacency matrix*

$$H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Let h_{ij} be 1 if node i links to (points to) node j, and 0 otherwise. h_{12} will then be 0 since A does not link to B, but h_{13} will be 1 because A links to C. Since a node can not point to itself, h_{ii} is always 0.

Then, we wish to turn H into a *row stochastic matrix* S, which is a matrix with its rows summing up to 1. We want this since we now want to express the matrix as a probability of reaching a certain spot. To do this, we will take each row and divide it by the sum of all the elements in the row, but since each element is either 0 or 1, the denominator is then equivalent to the number of nonzero elements of each row.

When we have a dangling node, it has no outlinks, so it is represented by a row of 0's in the matrix. But, in practice, we should still be able to access another page from any arbitrary pages, so we assign each node an equal probability of being reached from the dangling node. So in the matrix, we will represent this by having each element in the dangling node's row equal to $\frac{1}{n}$, where n is the number of elements (when n is large this becomes very small so this does not impose much influence to the overall graph). So in our example, node D's row became all $\frac{1}{7}$.

The resulting stochastic matrix is shown below.

$$S = \begin{pmatrix} 0 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Now we want to add a probability of "*teleporting*" to a random outlink from each webpage, which means that each node has a small chance of being sent to any node, including itself, so we will incorporate that in our new matrix G – the *transition matrix*. Let α be a constant determining when to teleport, meaning that α is the probability of following the links and $1-\alpha$ is the probability of teleporting to a random node. Then, the new formula for G is

$$G = \alpha S + (1-\alpha) \cdot \frac{1}{n} I_n,$$

where I_n is the identity matrix and n is the number of elements in each row. In the case of our dangling node, everything still works out since $\alpha \cdot \frac{1}{n} + (1-\alpha) \cdot \frac{1}{n} = \frac{1}{n}$, so it stays the same. In our example, we will use $\alpha = 0.85$ for the sake of simplicity, but it can theoretically be any value between 0 and 1, depending on the use. So in our example, we get our new matrix $G = 0.85S + 0.15 \cdot \frac{1}{n} I_n$ to be

$$G = \begin{pmatrix} \frac{3}{140} & \frac{3}{140} & \frac{32}{105} & \frac{3}{140} & \frac{32}{105} & \frac{32}{105} & \frac{3}{140} \\ \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{25}{56} & \frac{25}{56} & \frac{3}{140} \\ \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{25}{56} & \frac{25}{56} & \frac{3}{140} & \frac{3}{140} \\ \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} & \frac{1}{7} \\ \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{61}{70} & \frac{3}{140} \\ \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{61}{70} \\ \frac{3}{140} & \frac{61}{70} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} \end{pmatrix}$$

Let $\mathbf{w}$ be a row vector with 1 in the first entry and 0 in the others.

Because of the way G is defined, $\mathbf{w}G$ will be equivalent to the probability of being at a certain node after one iteration, starting from the first node. The probability of being at a certain node after two iterations will be $(\mathbf{w}G)G = \mathbf{w}G^2$. What we want is the probability of being at a certain node after a lot of iterations, so we are looking for the vector when the number of inerations t is really large. So we want to see if $\mathbf{w}G^t$ converges.

In our example, define a *state* to be a node. Additionally, a *state space* is the set of all states. Ari Freedman defines in **Definition 1.1** in his article "Convergence Theorem for Finite Markov Chains" that a *Finite Markov Chain* with a finite state space Ω and $|\Omega| \times |\Omega|$ transition matrix P is a sequence of random variables $X_0, X_1, \ldots$ where

$$\mathbf{P}\{X_{t+1} = y \mid X_t = x\} = P(x, y)$$

In the equation, $\mathbf{P}\{X_{t+1} = y \mid X_t = x\}$ is the probability of hitting state y given that the previous one was x, and $P(x, y)$ is the xth row and yth column of P. We can define a Finite Markov Chain with a state space of each of the nodes and transition matrix G, since the element at the x row and y column is the probability of going from node x to node y in each iteration. Freedman defines in **Definition 1.5** that a Markov chain is *irreducible* if for all states $x, y \in \Omega$, there exists a $t \geq 0$ such that $P^t(x, y) > 0$. What this means is that each node has a chance of reaching every other node after a certain amount of iterations. Since each node has a chance of "teleporting" to another node, each state has a chance of being sent to every other node, so the chain is irreducible. Ari Freedman defined in **Definition 4.1** from the above article that the *period* of a Markov Chain is the greatest common divisor of the number of "steps" in the chain for a node to reach itself. And because of "teleportation", each state has a chance of being "sent" to itself, and since the smallest possible divisor of any number is 1, the chain has a period of 1 by definition, and by **Definition 4.3** from the above article (which states that a chain is *aperiodic* if and only if the period is 1), this means that the chain is aperiodic.

A distribution $\mathbf{v}$ is called a *stationary distribution* of a Markov chain P if $\mathbf{v}P = \mathbf{v}$ [Freedman 2017]. **Theorem 3.3** in article "Convergence Theorem for Finite Markov Chains" by Ari Freedman states that if P is irreducible, then it has a unique stationary distribution $\mathbf{v}$. [Freedman 2017]. **Theorem 4.9** in the above article states that if P is irreducible and aperiodic, with stationary $\mathbf{v}$, then there exists constants $0 < \alpha < 1$ and $C > 0$ such that

$$\max_{x \in \Omega} \left|\left|P^t(x, \cdot) - \mathbf{v}\right|\right|_{TV} \leq C\alpha^t$$

[Freedman 2017]. What $||\mathbf{X} - \mathbf{Y}||_{TV}$ means is the *total variation* between the two, which is the max distance between two components of the vector. $P^t(x, \cdot)$ is the xth row of P^t, which means the probability vector of being at a certain state after t iterations given that we started from the xth row. In our case, we are just considering what happens after the first node, which is just $\mathbf{w}G^t$. What this equation means is that as t gets larger and larger, the right hand side goes to 0, which means the left hand side must go to 0, so $P^t(x, y)$, which is the vector after t iterations, approaches to the stationary distribution vector, meaning that it converges. Since the chain is both irreducible and aperiodic, there is a unique vector $\mathbf{v}$ that $\mathbf{w}G^t$ will converge to. Since it converges, each successive iteration after the vector is reached will be the same; in other words, $\mathbf{v}G = \mathbf{v}$. Thus, $(\mathbf{v}G)^T = \mathbf{v}^T$, $G^T\mathbf{v}^T = \mathbf{v}^T$, so $\mathbf{v}^T$ is an eigenvector for G^T with eigenvalue 1 with elements all adding up to 1. Using this, we can calculate the final stationary distribution vector $\mathbf{v}$, and the ith entry of $\mathbf{v}$ gives us the PageRank score for webpage i. Additionally, the vector will then represent the probability of being at a certain node after many iterations.

In our example, we calculate the eigenvalue of G^T to find this vector, with an eigenvalue of 1. The first vector we got is one with magnitude of 1. We find this eigenvector by doing row reduction on matrix $G^T - I_7$, where I_7 is the 7×7 identity matrix, to get the following:

$$\text{rref}(G^T - I_7) = \text{rref} \begin{pmatrix} \frac{-137}{140} & \frac{3}{140} & \frac{3}{140} & \frac{1}{7} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} \\ \frac{3}{140} & -\frac{137}{140} & \frac{3}{140} & \frac{1}{7} & \frac{3}{140} & \frac{3}{140} & \frac{61}{70} \\ \frac{32}{105} & \frac{3}{140} & -\frac{137}{140} & \frac{1}{7} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} \\ \frac{3}{140} & \frac{3}{140} & \frac{25}{56} & -\frac{6}{7} & \frac{3}{140} & \frac{3}{140} & \frac{3}{140} \\ \frac{32}{105} & \frac{25}{56} & \frac{25}{56} & \frac{1}{7} & -\frac{137}{140} & \frac{3}{140} & \frac{3}{140} \\ \frac{32}{105} & \frac{25}{56} & \frac{3}{140} & \frac{1}{7} & \frac{61}{70} & -\frac{137}{140} & \frac{3}{140} \\ \frac{3}{140} & \frac{3}{140} & \frac{3}{140} & \frac{1}{7} & \frac{3}{140} & \frac{61}{70} & -\frac{137}{140} \end{pmatrix}$$

$$= \begin{pmatrix} 1.0000 & 0 & 0 & 0 & 0 & 0 & -0.1059 \\ 0 & 1.0000 & 0 & 0 & 0 & 0 & -0.9559 \\ 0 & 0 & 1.0000 & 0 & 0 & 0 & -0.1358 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 & -0.1636 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 & -0.5998 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 & -1.0519 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Where rref is the row reduced echelon form of the matrix. So the eigenvector, which we are denoting $\mathbf{u}$, will be

$$\boldsymbol{u^T} = \begin{pmatrix} 0.1059 \\ 0.9559 \\ 0.1358 \\ 0.1636 \\ 0.5998 \\ 1.0519 \\ 1 \end{pmatrix}$$

We have that if a vector $\mathbf{a}$ is an eigenvector of a matrix A, then $\alpha\mathbf{a}$ is also an eigenvector of the same eigenvalue of A, since $A(\alpha\mathbf{a}) = \alpha A\mathbf{a} = \alpha\lambda\mathbf{a} = \lambda(\alpha\mathbf{a})$ where λ is the eigenvalue of $\mathbf{a}$. So, this means that to get the actual stationary distribution vector, we need to make this vector have a sum of 1, which is achieved by dividing the vector by the total sum, which causes each element to be divided by the total sum, and from above this will still be an

eigenvector in exactly the same way, and now it will follow the other requirements. Doing this, we get a new vector

$$\boldsymbol{v^T} = \frac{\boldsymbol{u^T}}{\sum_i \boldsymbol{u} \cdot \boldsymbol{e_i}} = \begin{pmatrix} 0.02638 \\ 0.23820 \\ 0.03385 \\ 0.04077 \\ 0.14947 \\ 0.26216 \\ 0.24920 \end{pmatrix}$$

Where $\boldsymbol{e_i}$ is the standard basis vector. So this means that the probability of being at A after a lot of iterations given that we started from A is 0.02638, and the probability of being at B is 0.2382, C is 0.0339, D is 0.04077, E is 0.14947, F is 0.26214, and G is 0.24920.

We can then sort this vector in descending order to get this:

$$\boldsymbol{v^T_{sorted}} = \begin{pmatrix} 0.26214 \\ 0.24920 \\ 0.23820 \\ 0.14947 \\ 0.04077 \\ 0.03385 \\ 0.02638 \end{pmatrix}$$

Which suggests that the webpage rankings listed from most important to least important, is: F, G, B, E, D, C, A.

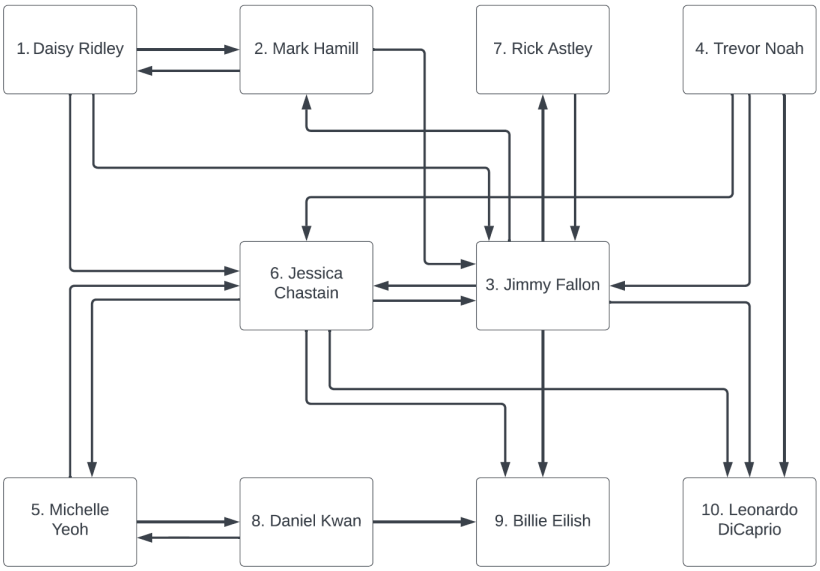
Looking back to the graph, this actually makes sense because F is pointed to by so many edges. G is then the second highest because it receives a direct edge from F with the highest ranking value. A has the least ranking value because it is not pointed to by any edges. Notice that despite D being a dangling node, it is not the lowest rank because it is still connected by a edge from C. Rank scores from C will then be partially transferred to D, raising up D's score to some extents. Therefore, being a dangling node doesn't necessarily mean that it will have the lowest scores even though it is very likely.

3 Example

We decided to apply the PageRank method to develop a hypothetical method to improve the "suggestions for you to follow" function on Instagram. When an Instagram user just starts to follow a new account, the Instagram "suggestions for you to follow" function will activate a pop-up bar on the user's screen to provide the user with a list of Instagram accounts that the user might also be interested in. Although Instagram is not currently using the PageRank algorithm to realize this function in its product, it can be an alternate solution to the problem. Therefore, we will discuss how the PageRank algorithm can be a possible approach to it as an example in this paper.

Recognizing that people tend to follow individuals they interact with, the idea is simple: if person A just followed person B on Instagram, then by looking at the list of people followed by B and B's followings, it is reasonable to infer that A might be also interested in following the accounts from B and his followings' following list. These accounts are likely to include individuals from the same environment (for example, in the same school or industry) as B, which are the range of people that A is interested in.

We decided to take a sample of celebrities and determine a hypothetical example for the application of the PageRank algorithm. The celebrities we chose were: (1) Daisy Ridley, (2) Mark Hamill, (3) Jimmy Fallon, (4) Trevor Noah, (5) Michelle Yeoh, (6) Jessica Chastain, (7) Rick Astley, (8) Daniel Kwan, (9) Billie Eilish, and (10) Leonardo DiCaprio. We collected the data from their official Instagram accounts to see if one is following another, and determined that their network graph to be:



Giving an adjacency matrix

$$H = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

This can be changed to a stochastic matrix

$$S = \begin{pmatrix} 0.000 & 0.333 & 0.333 & 0.000 & 0.000 & 0.333 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.500 & 0.000 & 0.500 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.200 & 0.000 & 0.200 & 0.000 & 0.200 & 0.000 & 0.000 & 0.200 & 0.200 \\ 0.000 & 0.000 & 0.333 & 0.000 & 0.000 & 0.333 & 0.000 & 0.000 & 0.000 & 0.333 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.500 & 0.000 & 0.500 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.250 & 0.000 & 0.250 & 0.000 & 0.000 & 0.000 & 0.250 & 0.250 \\ 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.500 & 0.000 & 0.000 & 0.000 & 0.500 & 0.000 \\ 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 \\ 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 \end{pmatrix}$$

Which can be turned into our new matrix G:

$$G = \begin{pmatrix} 0.015 & 0.298 & 0.298 & 0.015 & 0.015 & 0.298 & 0.015 & 0.015 & 0.015 & 0.015 \\ 0.440 & 0.015 & 0.440 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 \\ 0.015 & 0.185 & 0.015 & 0.185 & 0.015 & 0.185 & 0.015 & 0.015 & 0.185 & 0.185 \\ 0.015 & 0.015 & 0.298 & 0.015 & 0.015 & 0.298 & 0.015 & 0.015 & 0.015 & 0.298 \\ 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.440 & 0.015 & 0.440 & 0.015 & 0.015 \\ 0.015 & 0.015 & 0.228 & 0.015 & 0.228 & 0.015 & 0.015 & 0.015 & 0.228 & 0.228 \\ 0.015 & 0.015 & 0.865 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 \\ 0.015 & 0.015 & 0.015 & 0.015 & 0.440 & 0.015 & 0.015 & 0.015 & 0.440 & 0.015 \\ 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 \\ 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 & 0.100 \end{pmatrix}$$

Lastly, by calculating the eigenvector of G^T with an eigenvalue of 1, we do row reduction on matrix $G^T - I_{10}$, to get the following:

$$\text{rref}(G^T - I_{10})$$

$$= \text{rref}\begin{pmatrix} -0.985 & 0.440 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.100 & 0.100 \\ 0.298 & -0.985 & 0.185 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.100 & 0.100 \\ 0.298 & 0.440 & -0.985 & 0.298 & 0.015 & 0.228 & 0.865 & 0.015 & 0.100 & 0.100 \\ 0.015 & 0.015 & 0.185 & -0.985 & 0.015 & 0.015 & 0.015 & 0.015 & 0.100 & 0.100 \\ 0.015 & 0.015 & 0.015 & 0.015 & -0.985 & 0.228 & 0.015 & 0.440 & 0.100 & 0.100 \\ 0.298 & 0.015 & 0.185 & 0.298 & 0.440 & -0.985 & 0.015 & 0.015 & 0.100 & 0.100 \\ 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & 0.015 & -0.985 & 0.015 & 0.100 & 0.100 \\ 0.015 & 0.015 & 0.015 & 0.015 & 0.440 & 0.015 & 0.015 & -0.985 & 0.100 & 0.100 \\ 0.015 & 0.015 & 0.185 & 0.015 & 0.015 & 0.228 & 0.015 & 0.440 & -0.900 & 0.100 \\ 0.015 & 0.015 & 0.185 & 0.298 & 0.015 & 0.228 & 0.015 & 0.015 & 0.100 & -0.900 \end{pmatrix}$$

$$= \begin{pmatrix} 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.6288 \\ 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -0.7458 \\ 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & 0 & -1.5044 \\ 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & 0 & -0.5676 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 & -0.8738 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & -1.2780 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & 0 & -0.3119 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & 0 & -0.6832 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0000 & -1.1295 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

So the eigenvector, which we are denoting as $\mathbf{u}$, will be

$$\boldsymbol{u^T} = \begin{pmatrix} 0.6288 \\ 0.7458 \\ 1.5044 \\ 0.5676 \\ 0.8738 \\ 1.2780 \\ 0.3119 \\ 0.6832 \\ 1.1295 \\ 1 \end{pmatrix}$$

Changing its sum to 1 we get:

$$\boldsymbol{v^T} = \begin{pmatrix} 0.0721 \\ 0.0855 \\ 0.1725 \\ 0.0651 \\ 0.1002 \\ 0.1465 \\ 0.0358 \\ 0.0783 \\ 0.1295 \\ 0.1146 \end{pmatrix}$$

We can then sort this vector in descending order to get this:

$$\boldsymbol{v^T_{sorted}} = \begin{pmatrix} 0.1725 \\ 0.1465 \\ 0.1295 \\ 0.1146 \\ 0.1002 \\ 0.0855 \\ 0.0783 \\ 0.0721 \\ 0.0651 \\ 0.0358 \end{pmatrix}$$

Here $\boldsymbol{v^T}$ will be final stationary distribution vector. Therefore, the account ranking listed from the most likely to be followed by the user to the least likely to be followed by the user will be, (3) Jimmy Fallon, (6) Jessica Chastain, (9) Billie Eilish, (10) Leonardo DiCaprio, (5) Michelle Yeoh, (2) Mark Hamill, (8) Daniel Kwan, (1) Daisy Ridley, (4) Trevor Noah, and (7) Rick Astley. From the graph we can see that, (3) Jimmy Fallon and (6) Jessica Chastain are right in the middle of the graph, having a tons of connections with other celebrities, so they will have a much higher priority of being suggested to follow, because by following them, the users can then follow all other celebrities. (7) Rick Astley has the least score because it is the only one that has only one edge pointing to and from (3) Jimmy Fallon, then it will not receive a high priority. Note that (9) Billie Eilish and (10) Leonardo DiCaprio is

two dangling nodes but their rankings are the third and the fourth. This also shows that dangling nodes doesn't necessarily imply a low ranking.

4 Conclusion

The development of the Google PageRank algorithm is revolutionary as it greatly improved the way search engines sort and organize search results. By modeling the behaviors of a random web surfer and assigning PageRank score to webpages based on their importance and relevance, the algorithm allows search engines to sort search results from the most useful source to the least useful source. This considerably saves the users' time when they are looking for information.

The application of the algorithm is not limited to building search engines. The PageRank algorithm is also largely applied, for example, in scientific research such as analyzing protein networks in biology. The "suggestions for you" function mentioned in this paper is another potential use of the algorithm. As the PageRank algorithm becomes more and more completely developed, it can be used as a powerful tool in numerous fields and help people in need.

5 References

1. [BCS 2011] Bowl Championship Series, "Bowl championship series official website", webpage, 2011, http://www.bcsfootball.org.

2. [Freedman 2017] *Freedman, Ari.* "Convergence Theorem for Finite Markov Chains", 2017, math.uchicago.edu/ may/REU2017/REUPapers/Freedman.pdf.

3. [Zack et al. 2012] Zack, Laurie, et al. "An Application of Google's Pagerank to NFL Rankings." *Involve, a Journal of Mathematics*, vol. 5, no. 4, 2012, pp. 463–471, https://doi.org/10.2140/involve.2012.5.463.