PageRank Algorithm

CS-101
Discrete Mathematics

2022-23

PREPARED BY:
SHIVAM BHAGAT - 2022CSB1123
TARUSHI TANEJA - 2022CSB1135
ANANYA SETHI - 2022CSB1066
RHEA SANJAY - 2022CSB1112

Indroduction

We use Google or any other search engine every day and browse through a series of interlinked web pages. But how does the engine know which webpage, out of the 1.13 billion that exist in the world is the one we are looking for?, and how does it know which page is better than the other?

Larry Page first, the founder of Google, answered this question in 1998 when he came up with the first **PageRank Algorithm.** It was the first algorithm used by Google and has undergone several improvements and still stands tall as one of the algorithms used by Google to rank its webpages upon web searches.

In this project, we will be exploring the intricacies of the PageRank algorithm, studying its underlying principles and seeing how it revolutionised the searching technology. We will also be exploring how discrete mathematics comes up in this process and visualise its importance.

This project will contain the proper history, intuition, deep explanation, working, analysis and a self made example of the page rank algorithm. We have made our sincere efforts in making this project and hope that you'll appreciate them by the end of this project.

History of pagerank algorithm

Considered the most powerful company in the world- Google has its empire today which started back in 1998 with the PageRank algorithm. Google used the PageRank algorithm to rank pages on its search engine.
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.

In search results, pages with more links should therefore appear higher especially if those links originate from well-liked pages or pages with high PageRank scores.

Two key characteristics of the Google search engine enable it to deliver highly accurate results. To begin with, it uses the Web's link structure to determine each webpage's quality ranking. PageRank is the name of this ranking. Second, Google makes use of links to enhance search outcomes.

The PageRank algorithm generates a probability distribution that is used to illustrate the possibility that any given page would be reached by a random link clicker. Any number of documents in a collection can be used to calculate PageRank.

Several research publications make the supposition that, at the start of the computational process, the distribution is distributed equally among all documents in the collection. In order for the approximation PageRank values to more accurately reflect the theoretical actual value, the PageRank computations must make numerous passes, referred to as "iterations," across the collection.

The Introduction of the Google Toolbar

In 2000, Google released the Google Toolbar. This was one of the most critical periods in the development of PageRank. because it allowed users to view the rating of any page.
As a result, SEOs were fixated on increasing PageRank as a way to improve rankings.
Naturally, this resulted in the trade of links for money in order to manipulate PageRank. Links were widely dispersed in artificial locations.

The Retreat Of PageRank

Since the significance of a website was determined not only by its content but also by a kind of "voting system" produced by links to the page, Google's algorithm was once thought to be "unspam-able" internally.
However, Google's assurance did not last.

As the backlink industry expanded, PageRank started to cause issues. Google removed it from the public's view but still using it in its ranking algorithms.
By 2016, the PageRank Toolbar had been discontinued, and finally, PageRank was no longer accessible to the general public. But by this point, in particular, Majestic (an SEO tool) had been able to quite closely connect its own calculations with PageRank.

Until January 2017, Google actively discouraged SEO experts from manipulating links through its "Google Guidelines" literature and guidance from its Matt Cutts-led spam team.

During this period, Google's algorithms were also evolving.
The business was depending less on PageRank, and in 2014, Google began to index the world's information differently after acquiring MetaWeb and its proprietary Knowledge Graph (known as "Freebase" at the time).

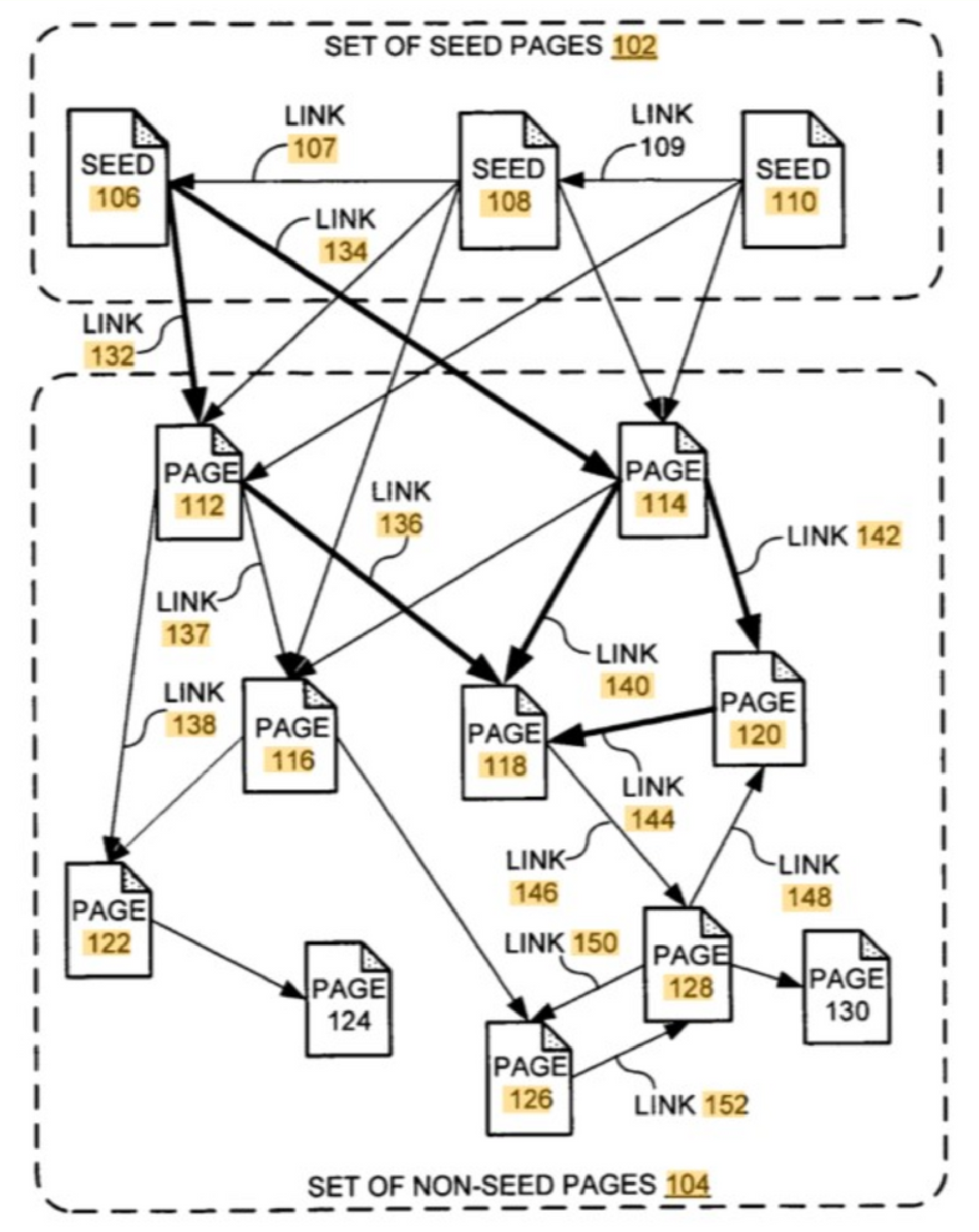
An Updated PageRank Patent

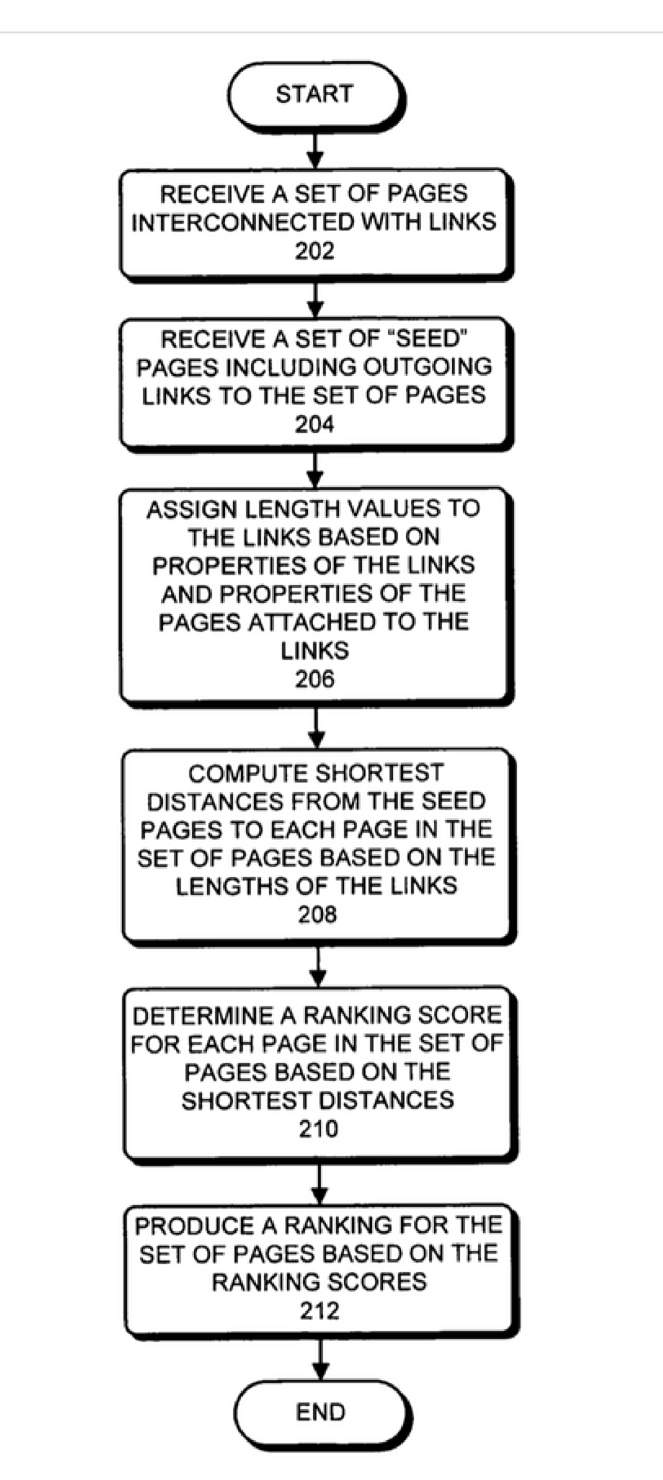
The original PageRank patent from 1998 expired in 2018 and, to the surprise of many, wasn't renewed.
Around this time, a former Google employee confirmed that the original algorithm hadn't been used since 2006.
The original patent was seemingly replaced by the new one. Which Google filed in 2006.

The system receives a set of pages to rank during operation, and the collection of pages is linked together. A set of seed pages with incoming links to the set of pages are also provided to the system. The algorithm then determines the lengths of the links based on the attributes of the links and the attributes of the pages to which they are related.The algorithm then calculates the shortest paths between each page in the set of pages and the set of seed pages based on the lengths of the links connecting the pages. The system then calculates a ranking score based on the computed shortest distances for each page in the set of pages. Based on the ranking scores for the set of pages, the system then generates a ranking for the set of pages.

In a variant of this embodiment, the system calculates a function of the number of outgoing connections from the source page of the link before assigning a length to the link.

The function is a monotonic non-decreasing function of the number of outgoing links from the source page in another form of this implementation, increasing the length of the connection as the number of outgoing links from the source page increases.
In a variant of this embodiment, the system calculates the shortest path between a seed page and a provided page by adding the lengths of each individual link in the path.


SET OF SEED PAGES 102
SEED 106
LINK 107
SEED 108
LINK 109
SEED 110
LINK 134
LINK 132
PAGE 112
PAGE 114
LINK 136
LINK 142
LINK 137
LINK 140
LINK 138
PAGE 116
PAGE 118
PAGE 120
LINK 144
LINK 146
LINK 148
PAGE 122
LINK 150
PAGE 128
PAGE 130
PAGE 124
PAGE 126
LINK 152
SET OF NON-SEED PAGES 104


START
RECEIVE A SET OF PAGES INTERCONNECTED WITH LINKS
202
RECEIVE A SET OF "SEED" PAGES INCLUDING OUTGOING LINKS TO THE SET OF PAGES
204
ASSIGN LENGTH VALUES TO THE LINKS BASED ON PROPERTIES OF THE LINKS AND PROPERTIES OF THE PAGES ATTACHED TO THE LINKS
206
COMPUTE SHORTEST DISTANCES FROM THE SEED PAGES TO EACH PAGE IN THE SET OF PAGES BASED ON THE LENGTHS OF THE LINKS
208
DETERMINE A RANKING SCORE FOR EACH PAGE IN THE SET OF PAGES BASED ON THE SHORTEST DISTANCES
210
PRODUCE A RANKING FOR THE SET OF PAGES BASED ON THE RANKING SCORES
212
END

Working Principle of PageRank

$$PR(u) = (1 - d) + d \times \sum \frac{PR(v)}{N(v)}$$

PR(u) is the page rank of the web page u

PR(v) is the page rank of the arbitrary webpages v pointing to u

N(v) is the total number of outgoing links of the arbitrary webpage v

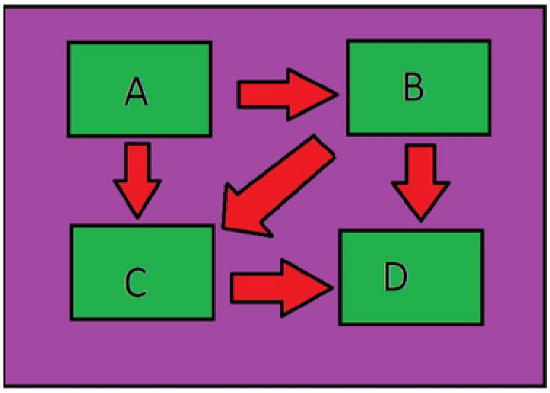
d is the damping factor

In the first iteration, this algorithm works by initially assigning a value to each webpage called the estimated PageRank score. The PageRank is then divided by the number of links on the page, and that results in a smaller fraction. This rank is then distributed to other linked pages. The process is carried out on other pages on the internet as well.

In the next iteration, a new estimate of PageRank is given as a sum of all the fractions of pages that link to each given page.

The damping factor is considered when a person who is surfing the web might stop surfing suddenly.

Before the next iteration, the proposed PageRank is reduced by the damping factor. The algorithm continues until it reaches equilibrium, and then the value is scaled down from 0 to 10 for convenience.



A sample graph for explaining the algorithm

Let us assume that the damping factor d is 0.6 then the formula stands as:-

P(N)=(1-d) +d* ∑P(A)/C(A)

P(A)= probability of the node a coming

C(A)=No of outgoing links from node A

P(A)=(1-0.6)=0.4

P(B)=(1-0.6) + 0.6*(0.4/2) = 0.52

//probability of getting A is 0.4 and no. of outgoing links from it is 2.

P(C)=(1-0.6) +0.6*(0.4/2 +0.52/2)=0.676

//There are 2 incoming links at C- A and B, with probabilities of 0.4 and 0.52, and the number of outgoing links from these is 2 and 2, respectively.

P(D)=(1-0.6) +0.6*(0.4/2 +0.52/2+0.676/1)=1.0816

//There are 3 incoming links at D- A, B, and C, with probabilities 0.4, 0.52, and 0.676 and No. of outgoing links from them are 2,2 and 1 respectively.

This is the first iteration of the calculations and this calculation continues until the average of all the PageRank is 1.0 .

PageRank Equations through Matrices

If we denote the set of pages with a hyperlink to p by pa_p , then each PageRank equation can be written in summation notation as

$$PR(p) = \sum_{q \in pa_p} \frac{PR_q}{|O_q|}$$

After kth iterations,

$$PR_p^{(k+1)} = \sum_{q \in pa_p} \frac{PR_q^k}{|O_q|} \text{ for } k = 0,1,2,\ldots,$$

Here,

pa_p is the set of webpages with a hyperlink to p.

Oq is the number of forward links of page q.

The above equations can be expressed using a single matrix equation.

To understand the matrix representation, let's familiarize ourselves with a few terminologies.

MARKOV CHAIN

It predicts the behavior of a system that moves from one state to another by considering only the current state.

It is a random process applied by a system that, at any moment "t," is in one of the limited number of states. That is at each time t the system moves from state v to u with probability P_{uv} that does not depend on the time t. P_{uv} is called the transition probability; it decides the next state of the object by considering only the current state and not the previous states.

Information for a Markov chain can be organized using a transition matrix

TRANSITION MATRIX

Transition Matrix T is an n x n stochastic matrix formed from the transition probability over one transition period of the Markov process, where n represents the number of states. Each entry in the transition matrix Tuv is equal to the probability of moving from state u to state v in one time slot. So, $0 \leq Tuv \leq 1$ must be true for all u, v = 1, 2, …, n.

Properties of a transition matrix-

1. a square matrix with non-negative entries wherein each column or row represents a state.
2. all entries are transition probabilities. Thus, their value should lie between 0 and 1 and the sum of the entries in any row must be equal to 1.

We need a way to represent the probability distribution among the states at a particular point in time. This is done using a stationary vector.

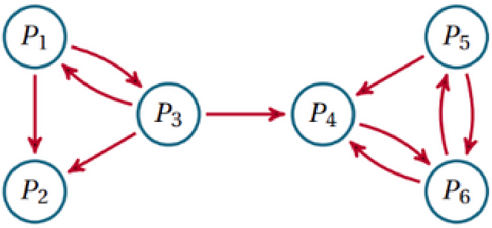
STATIONARY VECTOR
A stationary vector is a row matrix with only one row, it has a column for each state. The entries show the distribution by state at a given point in time.

For n webpages, we create a nxn hyperlink matrix A

A_{pq}= 1 , if hyperlink exists from q to p
A_{pq}= 0 , otherwise

Each entry in matrix A is then divided by the sum of the row to give the row-normalized hyperlink matrix H. H represents the transition matrix.

Consider the following mini web



The hyperlink matrix A and row-normalized hyperlink matrix H for the above Mini web will be

$$A = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \rightsquigarrow H = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

Matrix H can be viewed as a transition matrix where each entry $H_{pq,}$ denotes the probability of moving from web page p to q by clicking a hyperlink.

The basic page rank equation can be represented by this matrix equation:

$V^{k+1} = V^{K} H$, after k iterations

This will converge to give V'=V'H.

Where V represents the page rank vector. It denotes the probability distribution among the web pages at a particular iteration. V is represented as a row Matrice with only one row.

V’ is the limiting page rank vector.

Initially the probability of picking a web page is considered equal for all n pages.
That is for the above mini web,

$V^0 = \{1/6, 1/6, 1/6, 1/6, 1/6, 1/6\}$

There is a slight flaw in the above representation.

THE DANGLING WEB PAGE PROBLEM

A dangling webpage occurs when a webpage has no out links. On reaching a dangling web page a WebSurfer has nowhere to go and is stuck.
In the above H matrix, the second row consists of only zeroes, indicating that web page P2 is a dangling web page I.e., it has no out links.

Whenever there exists a dangling webpage, we observe that the limiting page rank vector V’ is not a probability vector I.e., the row sum is not equal to one.

For the above mini web finding the limiting vector using Matrix H gives us
$V' = \{0, 0, 0, 1/5, 2/15, 4/15\}$
The row sum is not equal to 1 implying that the limiting vector does not give us a probability vector.

This problem can be overcome by declaring that when a web surfer comes to a dangling web page, he can jump to ANY other web page at random.
Thus we get a new matrix S called the Stochastic Matrix.

For the above mini web, we get matrix S as

$$H = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix} \quad \rightsquigarrow \quad S = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}$$

Now the matrix representation becomes,

$V^{k+1} = V^K S$, after k iterations

This will converge to give $V' = V'S$.

We observe that the V’ obtained from this representation is a probability vector.

For the above mini web finding the limiting vector using Matrix S gives us
$V' = \{0, 0, 0, 1/3, 2/9, 4/9\}$
V’ is a probability vector I.e., the row sum is equal to 1.

HOW TO GUARANTEE THE EXISTENCE OF A LIMITING PAGE VECTOR?

Limiting vector won't exist when all the web pages are not covered.

Consider the situation in which the web surfer gets bored of following the hyperlinks and randomly opens a new web page and continues until he gets bored again only to repeat the process over and over.

To accommodate for the above possibility, DAMPING FACTOR 0<d<1 is taken.
Let d be the probability with which he follows hyperlinks, then the WebSurfer jumps into a random web page with the probability 1-d.

Each row S_i in the matrix gets modified into $(d*S_i + (1-d) *1*(1/n))$
Here 1 denotes the column matrix with n rows with all entries equal to 1.

Applying this to all rows of S we get a Matrix G, this is popularly known as the google matrix G

For the above mini web, the google matrix G will be

$$G = d\begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix} + (1-d)\begin{pmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \end{pmatrix}.$$

Usually, d=0.85 which gives us the google matrix G

$$G = \begin{pmatrix} \frac{1}{40} & \frac{9}{20} & \frac{9}{20} & \frac{1}{40} & \frac{1}{40} & \frac{1}{40} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{37}{120} & \frac{37}{120} & \frac{1}{40} & \frac{37}{120} & \frac{1}{40} & \frac{1}{40} \\ \frac{1}{40} & \frac{1}{40} & \frac{1}{40} & \frac{1}{40} & \frac{1}{40} & \frac{7}{8} \\ \frac{1}{40} & \frac{1}{40} & \frac{1}{40} & \frac{9}{20} & \frac{1}{40} & \frac{9}{20} \\ \frac{1}{40} & \frac{1}{40} & \frac{1}{40} & \frac{9}{20} & \frac{9}{20} & \frac{1}{40} \end{pmatrix}.$$

Now the matrix representation of the page rank algo becomes,
$V^{k+1} = V^{K} G$, after k iterations

The stochastic google matrix G guaranties the existence of the limiting vector V' which is used to rank the web pages.
i.e., it guarantees that after a finite number of iterations,
V'=V'G
The above is a direct implication of the PERRON FROBENIUS THEOREM.

Also, the limiting vector V' does not depend on the starting page rank vector $V^{0.}$
This can be explained by the MARKOV CHAIN THEORY.
It states that, given any arbitrary initial value, the chain will converge to the equilibrium point provided that the chain is run for a sufficiently long period of time.

Thus, we conclude that the matrix representation of the page rank algorithm guarantees a unique limiting page rank vector for any initial page rank vector.

Perron Frobenius Theorem

This theorem states that if a n*n matrix has all positive entries then it has a unique maximal eigenvalue and that its eigenvectors have a positive value.
This theorem helps to prove that if all entries of Markov matrix A are positive then it has a unique equilibrium that says that there is only one eigenvalue 1, all other eigenvalues of the Markov matrix will be less than 1.
We will illustrate this using a 2X2 and a 3X3 matrix.
Consider a 2X2 matrix of the general type

$$A = \begin{bmatrix} a & b \\ 1-a & 1-b \end{bmatrix}$$

A matrix's trace gives the sum of its eigenvectors so if one eigenvector is 1 and the sum is less than 2, the next eigenvector e will be less than 1. So the maximal eigenvector will be 1 for any Markov matrix.

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

Let's consider an example matrix where the sum of the trace is ½+2/3<2
If one eigenvector is 1, the other eigenvector will be less than 1. Thus, it proves that it is true for any 2X2 Markov matrix.

$$A = \begin{bmatrix} a & b & c \\ d & e & f \\ 1-a-d & 1-b-e & 1-c-f \end{bmatrix}.$$

Let's consider a 3X3 matrix,
the sum of the trace is 1+(a-c)+ (e-f) and the determinant of the matrix is
a(e-f)+b(f-d)+c(d-e).
If one eigenvector is 1, then the sum of the remaining eigenvector is (a-c)+(e-f).
So the absolute value of the remaining eigenvectors is less than 1. Thus it's valid for 3X3 matrix also.
A significant application of this theorem is used in the page rank algorithm.
The Google matrix is represented as G= dA+ (1-d)E, where d is the damping factor; 0<d<1; A is the Markov matrix obtained from the adjacency matrix by scaling the rows to become a stochastic matrix. E is the matrix that satisfies Eij=1/n for all i,j.
According to this theorem, the equation is then
[dA +(1-d)E]v=v
The concept used here is of eigenvalues and eigenvectors with the equation
Av=μv implies (A-μI)v=0 which replies that det(A-μI)=0,
where A is a matrix and μ is a scalar and v is a vector.