PageRank

Gea Janković, Nikola Henezi

December 21, 2015

Contents

1 Introduction

With the web growing very fast, retrieving relevant information from it is becoming more and more challenging. Every day there are new pages added and current ones deleted or modified. With the total number of more than 4.3 billion pages on the Internet, the number of the ones containing incorrect or redundant information is boosting up. It became highly important for web search engines to distinguish one from another and present quality material to the web surfer. According to Amy N. Langville and Carl D. Meyer "an average user enters very short queries, doesn't make use of system feedback to revise the query, seldom performs a search using advanced search options, and generally views only the top 10 to 20 documents returned by the search."

In the early 90's, the first search engines were using *text based ranking systems* to find pages relevant to the given query. The idea was to find pages that contain the highest number of repeated words given in the query. There were numerous problems with this approach, since the quantity of the repeated words doesn't say much about quality of the web pages links given as a result. The usefulness of a search engine is depending on the relevance and quality of the results which it gives back.

Today, the situation is a bit different. Many of today's search engines use a two-step process to retrieve the pages related to a user's query. First, they do a traditional text processing to collect all the pages and documents containing the query terms, or being related to them by semantic rules. This method groups and locally ranks pages according to traditional information retrieval methods[1]. In order to make that list manageable, search engines are sorting it by some ranking criterion. *Link analysis* has become the most popular way to rank pages, using the information about the hyper linking structure of the web. There are few well known methods, like PageRank, HITS and SALSA.

One of the most successful *link-based ranking system* is **PageRank**, the ranking system invented by Google's employees Sergey Brin and Larry Page in 1998 and used by Google. They came to a conclusion that web documents are extremely poor at their self description, so the links often provide more complete information about documents than the actual text in the documents. The importance of the web page is estimated by looking at the pages that link to it. If there are a lot of pages j linking to the web page i, i is considered important. Also, if there is only one, but important web page j that is linking to a web page i, it is said that j shares its PageRank, or transfers its authority to web page i. In other words, i is also important.

Using a hyperlink structure of the web, they were able to perform search on documents that can't be indexed by text based ranking systems also, like images, databases and programs. Besides the Google's popularity and success, PageRank is well preferred due to its query independence and immunity to spamming. The method itself is beautiful in its simplicity, and extremely interesting for its capacity for tinkering. This paper focuses solely on the PageRank method.

[1]For example, one of the most common mathematical algorithms to do so is a Latent Semantic Indexing, which is able to define semantic associations between words. Also, one can use formatting information to create relevancy set, such as font, the size, positioning on the page etc.

2 The Basic PageRank Model

2.1 How does it work?

The original model uses the hyperlink structure of the web to build a Markov chain [2] with a primitive [3] transition probability matrix **P**. The irreducibility of the Markov chain guarantees that the PageRank vector π^T exists, and the power method applied to a primitive matrix P will converge to exactly this stationary vector. That way set, the convergence is not in question. What is there to play with is the convergence rate which can be manipulated in a few ways described in the following chapters.

2.2 The Markov Model of the Web

The web can be represented as a directed graph in which the nodes stand for web pages, and the directed arcs stand for hyperlinks. We can show an example of the small collection consisted of 6 web pages and hyperlinks between them as following :

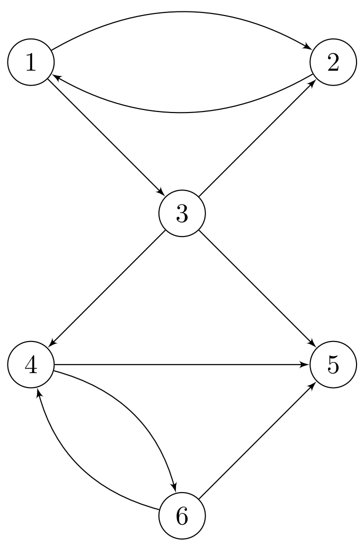

Figure 1: Directed graph representing a subsection of the web

The Markov model represents such directed graph with a square matrix **P**. Each element p_{ij} stands for the probability of going from state i to state j in

[2]A **Markov chain** is a random process that undergoes transitions from one state to another on a state space. It must possess a Markov property that is usually characterized as *memorylessness* : the probability distribution of the next state depends solely on the current state and not on the sequence of event that preceded it.

[3]A matrix is **irreducible** if its graph shows that every node is reachable from every other node. A non-negative, irreducible matrix is **primitive** if it has only one eigenvalue on it's spectral chain. *Frobenius test for primitivity* : The matrix $A \geq 0$ is primitive if and only if $A^m > 0$ for some $m > 0$

one step. Any suitable probability distribution is allowed to be used in the rows, considering that some pages may be more likely to be chosen than the others.

One way to create the matrix P would give the following:

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

The problem with this representation are the rows that contain all zeros. These rows represent the so-called **dangling nodes**, nodes that contain no outlinks. That means that the matrix is not stochastic [4]. It can be fixed by replacing the 0^T rows with $\frac{1}{n}e^T$, where n is the order of the matrix and e is an all-ones vector. Nevertheless, the uniform vector $\frac{1}{n}e^T$ can be replaced with a general probability vector $v^T > 0$.

$$\bar{P} = \begin{bmatrix} 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 1/3 & 1/3 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 1/2 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \\ 0 & 0 & 0 & 1/2 & 1/2 & 0 \end{bmatrix}$$

In order to guarantee the existence of the PageRank vector the irreducible chain is needed, so there is one more adjustment made to the matrix $\bar{P}$.

$$\bar{\bar{P}} = \alpha\bar{P} + (1-\alpha)ee^T/n$$

Matrix $\bar{\bar{P}}$ is now stochastic and irreducible, every node is connected to every other node, so the chain is irreducible.

Theorem 1 (Perron-Frobenius) *If a square matrix A is positive, stochastic and irreducible, then it has a simple eigenvalue equal to 1, with a corresponding positive eigenvector. All other eigenvalues are smaller in modulus.*

The probability of changing from state to state where an actual hyperlink doesn't exist is very small, but it's a nonzero. Because of this adjustment, matrix $\bar{\bar{P}}$ is also primitive, which assures that the power method will converge to the stationary vector π^T.

Bearing in mind the dimensions of the matrices P, $\bar{P}$ and $\bar{\bar{P}}$, and the time and memory costs of calculating each one of them, P is written in another way. The stochasticity fix to the matrix P can be modeled with the construction of a vector a where for $i \in 1, ..., n$:

$$a_i = \begin{cases} 1 & \text{if row } i \text{ of P corresponds to a dangling node} \\ 0 & \text{if row } i \text{ of P corresponds to a non-dangling node} \end{cases}$$

[4]**Stochastic** matrix is used to describe the transitions of a Markov chain. Each of its entries is a non-negative real number representing a probability. Sum of each row is 1.

Then $\bar{P} = P + av^T$ and $\bar{\bar{P}}$:

$$\begin{aligned} \bar{\bar{P}} &= \alpha \bar{P} + (1-\alpha) e v^T \\ &= \alpha P + (\alpha a + (1-\alpha) e) v^T \end{aligned} \tag{1}$$

How to interpret the matrix $\bar{\bar{P}}$? It models the **random surfer model** : most of the time, the surfer will follow the hyperlinks from a page i and go the neighbours of i. But there is a percentage of time, though much smaller, when a surfer will stop following hyperlinks, type the desired address in the address bar and get transported to it.

3 Solving the PageRank problem

The stationary vector π^T of the Markov chain can be computed solving the **eigenvector problem**

$$\pi^T \bar{\bar{P}} = \pi^T$$

with an additional normalization equation $\pi e^T = 1$ which insures that π^T is a probability vector. Due to issues of storage, computation time and complexity, **the Power Method** is the usual method of choice.

3.1 The Power Method

The Power Method is notoriously slow iterative method for computing the eigenvector for a given eigenvalue and matrix.

For any starting vector $x^{(0)T}$, usually $x^{(0)T} = e^T/n$

$$\begin{aligned} x^{(k+1)T} &= x^{(k)T} \bar{\bar{P}} \\ &= \alpha x^{(k)T} \bar{P} + (1-\alpha) x^{(k)T} e v^T \\ &= \alpha x^{(k)T} P + (\alpha x^{(k)T} a + (1-\alpha)) v^T \end{aligned} \tag{2}$$

Despite the sluggishness, the power method is yet most preferable. Why is that so?

It's possible to compute iterations using the extremely sparse matrix P instead of completely dense $\bar{\bar{P}}$. That means that there's no need to form or store big matrices $\bar{P}$ and $\bar{\bar{P}}$. Furthermore, at each iteration, the power method requires storing only one vector, the current iterate. Because of the sparsity of P, each vector-matrix multiplication required by the method is computed in $nnz(P)$ flops, where $nnz(P)$ stands for the number of non zeros in P. Since each row contains 3-10 non zero elements in average, all in all $O(nnz(P)) \approx O(n)$. Also, the method converges quickly. More about the convergence rate will be explained in the following chapters.

3.1.1 The mathematics behind

The irreducibility of the matrix $\bar{\bar{P}}$ guarantees the **existence** of the unique stationary vector π^T.

Since $\bar{\bar{P}}$ is primitive and stochastic, the spectral radius $\rho(\bar{\bar{P}})$ is 1 and it has one eigenvalue on the unit circle, meaning that the power method applied to P

will converge to that unique dominant eigenvector (the stationary vector π^T for the Markov matrix / the PageRank vector for the Google matrix).

Convergence rate:
As been said, convergence is not questionable. But, considering just the dimensions of matrix and vectors used, it's obvious that convergence rate is a critical issue. Kamvar and Haveliwala have proven that regardless of the value of the personalization vector v^T, the subdominant eigenvalue is equal to the scaling factor α for a reducible hyperlink matrix P. The web by itself is reducible so the rate of convergence of the power method applied to $\bar{\bar{P}}$ is the rate at which $\alpha^k \to 0$.

Tolerance:
Using the traditional termination criterion, measuring the residual $x^{(k+1)T} - x^{(k)T}$, the expected number of iterations needed to converge to a tolerance level τ is $\frac{log_{10}\tau}{log_{10}\alpha}$.
Brin and Page reported success using 50 - 100 iterations. Knowing that they used $\alpha = 0.85$, it implies that they were seeking tolerance in between 10^{-3} to 10^{-7}.

So, the smaller α is, the convergence rate is faster, but the hyperlink structure of the web used for computing the PageRank vector is lesser true.

3.2 The Linear System Formulation

The linear system formulation is another way of computing the PageRank vector, but we are not going to focus much on that one. The eigenvalue problem

$$\pi^T(\alpha\bar{P} + (1-\alpha)ev^T) = \pi^T$$

can be rewritten as

$$\pi^T(I - \alpha\bar{P}) = (1-\alpha)v^T$$

The system is accompanied with $\pi^T e = 1$.

4 Playing with PageRank

4.1 Behavior of α

When creating the PageRank, Brin and Page reported that they used $\alpha = 0.85$ for computing the PageRank vector. Why exactly that choice for α? As explained before, with $\alpha = 0.85$, we can expect a fast convergence. For tolerance 10^{-6} convergence is reached in approximately 85 iterations. For $\tau = 10^{-8}$ that would be about 114 iterations. Any α higher than this, getting too close to 1, slows the convergence time drastically. Considering that Google matrix works on data of 4.3 billion pages, every iteration counts. Choosing $\alpha = 0.99$ and $\tau = 10^{-8}$ the expected number of iterations jumps to 1833.

The greater α should represent the model much better, but doesn't really represent the real world better. With α we produce the model for **web surfing behavior** and $\alpha = 0.85$ represents it quite correctly. That means that an average surfer spends 5/6 of his time on the internet jumping to pages through hyperlinks, and only 1/6 of the time he uses to actually type the desired web page address into an address bar and then get teleported there. Choosing $\alpha = 0.99$ not only slows the convergence of the method, but also pays too much attention to the hyperlink structure of the web and ignores the time that surfer spend on teleportation. Alternatively, choosing significantly smaller α doesn't represent the real situation because of giving too much attention to the teleportation tendencies which is not really accurate.

Different α can give tremendously different PageRank vectors. Their difference would occur the most at the end of the ranking list and then decrease as we go up to the top.

4.2 The personalization vector v^T

In the Power Method, instead of using $\frac{1}{n}ee^T$, one can use ev^T, where $v^T > 0$ stands for a probability vector that contains all positive elements. Therefore, the matrix $\bar{\bar{P}}$ is still irreducible since every node is connected to every node. What is the difference from using $\frac{1}{n}ee^T$?

$\frac{1}{n}ee^T$ means that the probabilities of teleportation are uniformly distributed, and by using v^T probabilities can be distributed depending on users preferences. The interesting observation that can be seen from Equation (2) is that in order to compute a personalized PageRank for a particular user, the only thing that needs to be modified is exactly that constant personalization vector v^T.

The idea of personalization search is contrary with the idea of query independent searching, but it's natural to think of using it. Not every surfer has the same interests, and the same query might have slightly different meanings for different users. However, having a personalized vector for each surfer is way beyond possible considering the computational time needed for computing Pagerank, but grouping users into interests groups might have some advantages. Some predict that the personalized search engines are the future of searching, but at the moment it's computationally impossible.

How did Google use it? Recently Google used the personalization vector to prevent spamming from linking farms. A link farm is any group of web sites that all hyperlink to every other site in the group. They are used to falsely increase the rank of the client's pages. Google found a way to use the personalization vector to annihilate the ranks of link farms and their clients.

5 Implementation

We've implemented the Power method to compute the PageRank vector. Our intention was to find the PageRank vector for a subsection of the web containing our math departments' web page https://www.math.pmf.unizg.hr/.

In the process of collecting all nodes and hyperlinks we encountered storage issues, so we had to reduce our first intentions and limit the data to 33 182 web pages (nodes of a Markov chain) and 166 048 hyperlinks (directed arcs in a chain).

22 145 out of 33 182 nodes are dangling nodes and it makes 66.7% of all nodes.
141 252 out of 166 048 links are unique and it makes 85% of all the hyperlinks.

5.1 Results

α	Iteration
0.5	14
0.75	31
0.8	40
0.85	54
0.9	86
0.95	200
0.98	672
0.99	1047

Figure 2: Relations between the chosen α and the number of iterations needed for convergence with the toleration $\tau = 10^{-6}$.

Node	Outgoing links
http://web.math.pmf.unizg.hr/%7Eigaly/EH43/EHpregled.htm	555
http://web.math.hr/glasnik/classindex.html	490
http://web.math.hr/ krcko/radovi/mag/mag.html	286
http://web.math.hr/%7Eduje/sitemap.html	207
https://www.math.pmf.unizg.hr/hr/diplomski-sveuÄŤiliĹˇni-stu...	174
http://web.math.pmf.unizg.hr/nastava/la/kolokviji.html	166
https://www.math.pmf.unizg.hr/hr/integrirani-preddiplomski-i-diplom...	160
https://www.math.pmf.unizg.hr/hr/osoblje	149
https://www.math.pmf.unizg.hr/hr/preddiplomski-sveuÄŤiliĹˇni-studij...	148
http://web.math.hr/nastava/metodika/materijali.php	144
https://www.math.pmf.unizg.hr/hr/nastavnici-i-mentori-0	144
https://www.math.pmf.unizg.hr/hr/preddiplomski-sveuÄŤiliĹˇni-studij...	143
https://www.math.pmf.unizg.hr/hr/diplomski-sveuÄŤiliĹˇni-studij...	142

Figure 3: Nodes sorted by the number of outgoing links

Node	Incoming links
https://www.math.pmf.unizg.hr/	1471
http://web.math.pmf.unizg.hr/nastava/raspored/	1437
https://www.math.pmf.unizg.hr/hr/kolokviji-i-ispiti-0	1432
https://www.math.pmf.unizg.hr/en	1430
https://www.math.pmf.unizg.hr/sites/default/files/kalendar_nastave_15_16.pdf	1426
https://www.math.pmf.unizg.hr/sites/default/files/natjecaj_fer.pdf	1426
https://www.math.pmf.unizg.hr/sites/default/files/zaba_programer.pdf	1426
https://www.math.pmf.unizg.hr/hr	1426
https://www.math.pmf.unizg.hr/sites/default/files/orange_logic.pdf	1426
https://www.math.pmf.unizg.hr/sites/default/files/altima_tester.pdf	1426
https://www.math.pmf.unizg.hr/sites/default/files/oglas_pbf.pdf	1426
https://www.math.pmf.unizg.hr/sites/default/files/rp_2015_16-pmf-mo.pdf	1426
https://www.math.pmf.unizg.hr/sites/default/files/altima_developer.pdf	1426

Figure 4: Nodes sorted by the number of incoming links

Node	Incoming links
https://www.math.pmf.unizg.hr/	0.00869860795165
http://web.math.hr/ bbasrak/?Research%2FCV	0.00798506298156
http://web.math.pmf.unizg.hr/nastava/ip/?Literatura	0.00516418616732
http://web.math.hr/nastava/vms/zivkovic/1/vibracije_files/	0.00506143410346
http://web.math.hr/nastava/rp1/zadace/testdata/2015/	0.00344797996808
https://www.math.pmf.unizg.hr/sites/default/files/natjecaj_fer.pdf	0.0033881887635
http://web.math.pmf.unizg.hr/nastava/rnm/	0.00320885348904
http://web.math.hr/nastava/iter/slideovi/rjesenja/	0.00311270847752
http://web.math.hr/nastava/	0.00310970218653
http://web.math.hr/ borismil/os0203/	0.00300029421308
http://web.math.pmf.unizg.hr/%7Eigaly/EH43/EHpregled.htm	0.00298459124255
http://web.math.hr/ bbasrak/?%26nbsp%3B%26nbsp%3B	0.00298228604749
https://www.math.pmf.unizg.hr/hr/kategorije/naslovnica	0.00298228604749
http://web.math.pmf.unizg.hr/ duje/index.html	0.00298228604749
https://www.math.pmf.unizg.hr/hr/zdravstvena-za%C5%A1tita	0.00298228604749
http://web.math.hr/ bbasrak/?Contact_%2F_Other_info	0.00298228604749
http://web.math.hr/%7Eduje/index.html	0.00298228604749
http://web.math.pmf.unizg.hr/nastava/vms/	0.00298228604749
https://www.math.pmf.unizg.hr/sites/default/files/altima_tester.pdf	0.00282052490226
https://www.math.pmf.unizg.hr/sites/default/files/altima_developer.pdf	0.00281636078816
https://www.math.pmf.unizg.hr/hr/kolokviji-i-ispiti-0	0.00281026226071
https://www.math.pmf.unizg.hr/hr/obrasci-za-molbe-0	0.00281008414875
https://www.math.pmf.unizg.hr/hr	0.00280651196952
http://web.math.pmf.unizg.hr/racunski.centar/?Obavijesti	0.0027729141965
http://web.math.hr/ duje/tors/tors.html	0.00276031075439
https://www.math.pmf.unizg.hr/en	0.00275687176235
http://web.math.pmf.unizg.hr/nastava/	0.00274721923933
http://web.math.pmf.unizg.hr/nastava/raspored/	0.00274721923933
https://www.math.pmf.unizg.hr/hr/sredi%C5%A1nja-matemati%C4%8...	0.00274721923933
https://www.math.pmf.unizg.hr/hr/osoblje	0.00274721923933
https://www.math.pmf.unizg.hr/sites/default/files/rp_2015_16-pmf-mo.pdf	0.00274721923933
https://www.math.pmf.unizg.hr/sites/default/files/zaba_programer.pdf	0.00274721923933
http://web.math.hr/nastava/fm/pismeni.html	0.00224729431354
http://web.math.pmf.unizg.hr/nastava/vms/vjezbe7ana/	0.00209270894788
http://web.math.pmf.unizg.hr/nastava/rp1/zadace/testdata/2015...	0.0019760780502
http://web.math.pmf.unizg.hr/nastava/rp4/?Obavijesti	0.00176640688047
http://web.math.pmf.unizg.hr/nastava/vekt/	0.00170739873377
https://www.math.pmf.unizg.hr/hr/kolokvijski-razredi-zimski-semestar-0	0.00159873770649

Figure 5: PageRank vector. $\alpha = 0.85$, $\tau = 10^{-6}$. Sample size: top 13000 nodes, sorted by number of inlinks.

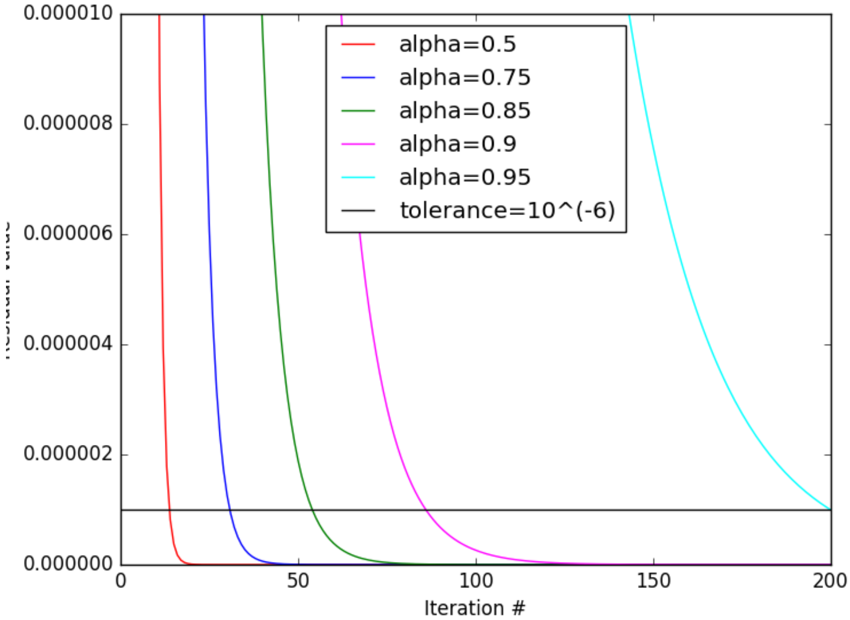

Figure 6: How convergence depends on α

6 References

[1] Amy N. Langville and Carl D. Meyer. ”Deeper inside PageRank”

[2] Kristen Thorson. ”Modeling the Web and the Computation of PageRank”

[3] https://en.wikipedia.org/