PageRank Algorithm

Ha Le

April 2021

1 A Representation of the Web

1.1 Introduction to Graph

A graph is an ordered pair $G = (V, E)$, where V is the set of vertices in the graph, and $E = \{(x, y)|x, y \in V, x \neq y\}$.

The edge of the graph can be directed or undirected, and we denoted it by drawing an arrow from one node to another to represent the direction of the edge. An edge can also have weight, which can be extremely useful to represent more complex structure, for example the map of different cities.

Over the years, graph theory has been studied by various mathematicians as well as computer scientists, and many algorithms have been developed to solve multiple problems including shortest path, spanning tree, connected problems, ... As we experience a leap in technology, the studies of large-scale graphs and their computation become more crucial, as they represent different social structures. In the next chapter, we will see how enormous parse graph can be use to represent the connection between different webpages.

1.2 PageRank as a Directed Graph

Since the Web is consisted of multiple html pages linked together, it is natural to use a directed graph as a data structure to store their relationship. We denote each page as a node in the graph, and if there is a link from page A to page B, we add a directed edge (A, B) to the graph.

Directed graph can also be used to represent different structures as well, for example a citation network[2]. However, in the case of a citation network, the graph cannot contain any cycles, since papers are written chronologically, an old paper cannot cite a new one.

2 The PageRank Algorithm

In 1996, two graduate students, Sergey Brin and Lawrence Page, at Stanford University has conducted research on the new kinds of search engine. At that moment, many search engine has been developed, including WWWW, AltaVista, WebCrawler, ... and they claimed to received millions of queries per day. However, most algorithms at the time face serious quality and complexity issues, which would prove to be a big problem as the popularity of the World Wide Web was increasing rapidly. Brin and Page's research on a new type of search engine that uses link structure and text anchor to improve search quality, especially in the academic realm. They came up with Google, which overtime has proved itself as the most prominent search engine, and is currently irreplacable in our daily life. One of the two main features of Google engine is the PageRank algorithm [3], which was designed to bring order to the Web.

2.1 The PageRank value of a webpage

In this section, let's introduce a simple, iterative version of PageRank. We define the PageRank of a page A as follow:

Definition 1 *Let an arbitrary webpage A has n pages linked to it, denoted $T_1, T_2, \cdots T_n$. Let C_n be the number of links going out of page A. The PageRank value of page A, let's call it $PR(A)$, is the following:*

$$PR(A) = \frac{PR(T_1)}{C(T_1)} + \frac{PR(T_2)}{C(T_2)} + \cdots + \frac{PR(T_n)}{C(T_n)} \tag{1}$$

From the definition above, we can derive the basic iterative algorithm to calculate the PageRank value of the entire network.

2.2 The Iterative Algorithm

We introduce the Basic PageRank Algorithm [4] for a network at step k as follow:

Algorithm 1 *We imagine the PageRank value as a "fluid" that travel through the network, passing through nodes by edges, and nodes with most amount of fluids run into are considered important. The process goes as follows:*

- *Let n be the number of nodes in our network. Initially, each page receive the PageRank value of $1/n$*
- *Choose a number of step k*
- *At each step, a node A pass $1/C(A)$ amount of PageRank that it currently possess through it outgoing link, into other nodes in the network. If it does not have any outgoing links, all its PageRank value remains.*

The algorithm seems to work for most cases, however, under a certain kind of graph, this algorithm will create serious problem, (for example a graph with isolated nodes, or strong connected components with no outlinks) which prompts us to add some new features to the definition of PageRank and the iterative algorithm.

3 The Scaling of PageRank

3.1 The Damping Factor

To avoid cases where Basic PageRank fails, we introduce the damping factor α, and a new definition of PageRank.

Definition 2 *Let an arbitrary webpage A has n pages linked to it, denoted* $T_1, T_2, \cdots T_n$*, and* C_n *be the number of links going out of page A. Let* α *be the damping factor of the algorithm, such that* $0 < \alpha < 1$*. The PageRank value of page A, let's call it* $PR(A)$*, is the following:*

$$PR(A) = (1-\alpha) + \alpha(\frac{PR(T_1)}{C(T_1)} + \frac{PR(T_2)}{C(T_2)} + \cdots + \frac{PR(T_n)}{C(T_n)}) \tag{2}$$

Usually, we should set α *to be between 0.8 and 0.9. To maximize computation efficiency, Brin and Page chose* $\alpha = 0.85$*.*

3.2 An equivalent model for PageRank

Since our current PageRank is an iterative algorithm, one might wonder whether the PageRank value of all webpages fluctuate as the number of step k changes. In this section, we will look at a similar model of PageRank, the Random Surfer Model[1]:

Theorem 1 *Imagine a surfer who:*

- *Start randomly at one point in the graph.*
- *He can move from one node A to other node B within one click if* $(A, B) \in E$*.*
- *Given a probability p, he can randomly move to another random node on the graph within one step.*

In their book[4], David Easly and Jon Kleinberg have shown that the random surfer model and PageRank model are equivalent:

Claim 1 *The probability of being at a page X after k steps of this random surf is precisely the PageRank of X after k applications of the Basic PageRank Update Rule.*

The Fundamental Theorem of Markov Chain [5] states that:

Theorem 2 *Let* P *be the transition probability matrix for a connected Markov chain. For a connected Markov chain there is a unique probability vector* π *satisfying* $\pi P = \pi$*.*

which can be interpreted as follow: The probability that the surfer is at a particular page X in k steps converges to a fixed number independent of the starting point. Hence, as k gets large, the PageRank value of all nodes in our graph will eventually converge as well.

The Random Surfer Model shows us an interesting way of proving the convergence of PageRank, and it also gives us more insights into the model.

4 The Linear Algebra of PageRank

4.1 The adjacency matrix

Any simple graph can be expressed using a square adjacency matrix, and for our PageRank directed graph, we can construct the adjacency matrix that represents the link between different nodes as follow: for each entry (i, j) in the matrix A:

$$A(i,j) = \begin{cases} 1, & \text{if } (i,j) \in E \\ 0, & \text{otherwise} \end{cases}$$

We can modify each non-zero entry of matrix A so it can accurately illustrate the flows of PageRank value from each node through its outlinks. For each entry $(i, j) \in M'$, let n be the number of outlinks from node i, we have:

$$M'(i,j) = \begin{cases} \frac{1}{n}, & \text{if } (i,j) \in E \\ 1, & \text{if } n = 0 \text{ and } i = j \\ 0, & \text{otherwise} \end{cases}$$

We can also easily get the adjacency matrix for the scaling version of PageRank, by adding the damping factor to every non-zero entry of M'. For each entry $(i, j) \in M$, let n be the number of outlinks from node i, we have:

$$M(i,j) = \begin{cases} \alpha \cdot M'(i,j) + \frac{1-\alpha}{n}, & \text{if } M'(i,j) \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

4.2 PageRank spectral analysis

Let $\vec{v}^k$ be the PageRank vector of all webpages, where v_i^k denotes the PageRank value of node i after k iteration. Hence, we can represent the PageRank Update Rule as:

$$v_i^{k+1} = M_{1i} \cdot v_1^k + M_{2i} \cdot v_2^k + \cdots + M_{1n} \cdot v_n^k$$

where n denotes the total number of nodes in our graph. The equation above can also be written as:

$$\begin{aligned} v_{k+1} &= M^T \cdot v_k && (3) \\ &= (M^T)^{k+1} \cdot v_0 && (4) \end{aligned}$$

If we believe the PageRank values of all nodes will eventually converges, then the problem is boiled down to finding the vector v such that $v = M^T \cdot v$, which is the eigenvector of M^T with the corresponding eigenvalue of 1. Since M^T is a transitional matrix, all entries will be positive, Perron's theorem [4] guarantees there exists such vector v and that v is unique:

Theorem 3 *Let A be a positive square matrix with spectral radius ρ, then A will have the following properties:*

- *A has an eigenvalue ρ such that $|\rho| > |\rho'|$ for all other eigenvalue ρ'*
- *There is an eigenvector v with positive real coordinates corresponding to the largest eigenvalue ρ, and v is unique up to multiplication by a constant.*
- *If the largest eigenvalue ρ is equal to 1, then for any starting vector $x \neq 0$ with non- negative coordinates, the sequence of vectors $A^k x$ converges to a vector in the direction of v as k goes to infinity.*

Perron's theorem does not only prove that the PageRank vector v converges as k goes to infinity, it also proves the uniqueness of v.

We have reduced the computation of PageRank from an iterative method to matrix multiplication, and now we have shown that PageRank computation is equivalent to the problem of finding the eigenvector for the tranpose of the transitional matrix M.

4.3 Complexity of PageRank

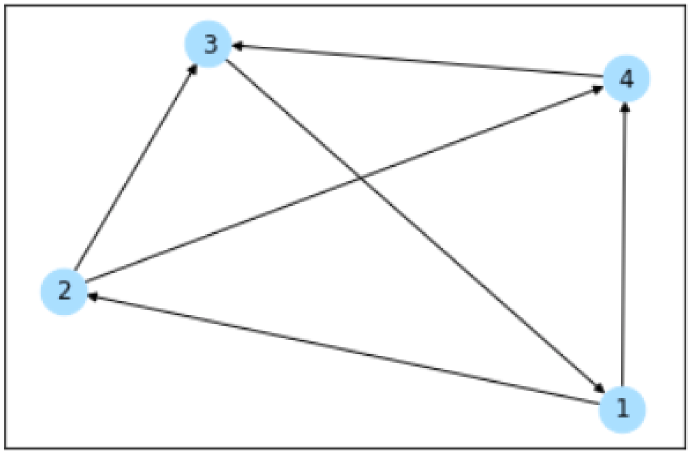
There are multiple algorithms for doing fast matrix multiplication, for example Strassen's algorithm [7] runs in $\mathcal{O}(n^{2.8})$, and since the graph for the Web is very parse, there are plenty other algorithm that has been proved to run better. Using divide and conquer to solve matrix exponentiation, the overall time complexity for matrix exponentiation is $\mathcal{O}(n^{2.8} \cdot log(k))$, with k be the number of iteration.

However, in 1996, Watts and Strogatz's work on the "small-world model" [9] has made significant impact on our understanding of social networks structure, and one of its impact is to show that the number of steps needed for PageRank vector to converge is negligible. In their original paper, Brin and Page claims that there algorithm converges within 52 steps, using a network with more than 300 millions links.

Many research have been conducted to improve the computation time of eigenvector and matrix multiplication. A recent paper [8] has shown that by using in-memory computing with cross-point resistive memory arrays, it can reduce the time complexity of computing eigenvector to $\mathcal{O}(1)$.

5 Experiment

In this section, we consider the simple network of 4 webpages as follow:



Here is the pseudocode for PageRank algorithm using matrix multiplication with k steps

Algorithm 1: PageRank using Matrix Multiplication

```
Given transitional matrix $M$, number of steps $k$;
$v \longleftarrow (\frac{1}{n}, \frac{1}{n}, ..., \frac{1}{n})$;
$M_k \longleftarrow I$;
while $k > 0$ do
    if $k = 1$ then
        $M_k = M_k \cdot M$;
    else
        $M_k = M_k \cdot M_k$;
        $k = k/2$;
    end
end
$v = M_k \cdot v$;
return $v$;
```

When we run the algorithm above to our miniature network, we can see that the PageRank value converges very well after 30 steps. The result is shown in the table below:

Node	1	2	3	4
1	0.25	0.25	0.25	0.25
...				
30	0.30	0.15	0.30	0.23
31	0.30	0.15	0.30	0.229

which has shown that our algorithm works accordingly to our analysis.

6 Conclusion

Google has proved to be an important aspect of our everyday life, and its existence has changed the way we live and search for data for the past two decades. PageRank algorithm, together with anchor text, being the two main features of Google, still proves to be an interesting research problem. Multiple related research questions have been posed, and many algorithms has been inspired by it, including the famous HITS algorithm [6]. Above all, PageRank algorithm has shown us that by using only simple linear algebra and a bit knowledge on graph theory, one could open an entire new research field and change the way people experience technology once and for all.

References

[1] Avrim Blum, TH Hubert Chan, and Mugizi Robert Rwebangira. A random-surfer web-graph model. In *2006 Proceedings of the Third Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*, pages 238–246. SIAM, 2006.

[2] John Adrian Bondy, Uppaluri Siva Ramachandra Murty, et al. *Graph theory with applications*, volume 290. Macmillan London, 1976.

[3] Sergey Brin and Lawrence Page. Reprint of: The anatomy of a large-scale hypertextual web search engine. *Computer networks*, 56(18):3825–3833, 2012.

[4] David Easley, Jon Kleinberg, et al. *Networks, crowds, and markets*, volume 8. Cambridge university press Cambridge, 2010.

[5] Charles J Geyer. Practical markov chain monte carlo. *Statistical science*, pages 473–483, 1992.

[6] Jon Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 163–170, 2000.

[7] Volker Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969.

[8] Zhong Sun, Giacomo Pedretti, Elia Ambrosi, Alessandro Bricalli, and Daniele Ielmini. In-memory eigenvector computation in time o (1). *Advanced Intelligent Systems*, 2(8):2000042, 2020.

[9] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.