A Review on Pagerank and Personalized Pagerank Algorithms

Roshni K[1], Dr. Unnikrishnan K[2]

[1]*Student, Master of Technology, Computer Science &Engg, RIT Engineering College Kottayam, Kerala, India*
[2]*Professor, Dept. of Computer Science & Engg, RIT Engineering College Kottayam, Kerala, India*

---***---

Abstract - *PageRank is an algorithm which is widely used to estimate reputations for webpages and social networks. It assigns each vertex of a graph with a rank that signifies the importance of the vertex in the graph. Personalized PageRank is a variation of PageRank used by Twitter and other services to provide personalized search results and recommendations by computing PageRank relative to a particular vertex or set of vertices. Personalized PageRank (PPR) is a widely used node proximity measure in graph mining and network analysis. Given a source node s and a target node t , the PPR value p(s, t) represents the probability that a random walk from s terminates at t, and thus indicates the bidirectional importance between s and t. The main aim of this paper is to discuss the various existing page ranking algorithms, personalized pagerank algorithm techniques.*

Key Words: **PageRank, Personalized PageRank, Graph, Webpage, Recommendation System, Twitter**

1. INTRODUCTION

PageRank is an algorithm originally developed to rank the importance of webpages by using the quantity and quality of links to a webpage. PageRank uses the hyperlink structure of the web to build a Markov chain with a primitive transition probability matrix. The irreducibility of the chain guarantees that the long run stationary vector, known as the PageRank vector, exists[1]. The values corresponding to each page in this vector gives the PageRank score of the page. Over the years, PageRank score has been widely adopted the relative importance of vertices in various graph based scenarios.

Personalized PageRank is a variation of PageRank used by many services to provide personalized search results and recommendations by computing PageRank relative to a particular vertex or set of vertices. It uses random walks to determine the importance or authority of vertices in a graph from the point of view of a given source node. Given a fixed termination probability at each step, the Personalized PageRank score of a vertex with respect to the source vertex represents the probability that a random walk from the source terminates at this vertex. This has widespread applications in areas like web search, spam detection, social networks and graph neural networks.

Personalized PageRank (PPR) is the personalized version of the PageRank algorithm which was important to Google's initial success. On any graph, given a starting node s whose point of view here take, Personalized PageRank assigns a score to every node t of the graph. This score models how much the user s is in interested in t, or how much s trusts t. More generally can personalize to a distribution over starting nodes, for example in web search we can create a distribution with equal probability mass on the web pages the searching user has bookmarked. If we personalize to the uniform distribution over all nodes, the score is no longer personalized, and recover the standard (global) PageRank score.

PPR has widespread applications in the area of data mining, including web search[2], spam detection[3], social networks[4], graph neural networks[5], and graph representation learning[6], and thus has drawn increasing attention during the past years. Studies on PPR computations can be broadly divided into four categories: 1) single-pair query, which asks for the PPR value of a given source node s and a given target node t ; 2) Single source query, which asks for the PPR value of a given source node s to every node t 2 V as the target; 3) single-target query, which asks for the PPR value of every node s V to a given target node t . 4) all-pairs query, which asks for the PPR value of each pair of nodes. While single-pair and single-source queries have been extensively studied, single-target PPR query is less understood due to its hardness[7].

2. PAGE RANK ALGORITHM

Brin and Larry Page[8, 9] developed a ranking algorithm used by Google, named *PageRank (PR)* after Larry Page (cofounder of Google search engine), that uses the link structure of the web to determine the importance of web pages. Google[10] uses PageRank to order its search results so that documents that are seem more important move up in the results of a search accordingly. This algorithm states that if a page has some important incoming links to it then its outgoing links to other pages also become important. Therefore, it takes backlinks into account and propagates the ranking through links. Thus, a page obtains a high rank if the sum of the ranks of its backlinks is high.

The PageRank algorithm considers more than 25 billion web pages on the WWW to assign a rank score [10]. When some query is given, Google combines precomputed PageRank scores with text matching scores [11] to obtain an overall ranking score for each resulted web page in response to the query. Although many factors are considered while determining the overall rank but PageRank algorithm is the heart of Google.

A simplified version [8] of PageRank is defined in Eq. 1:

$$PR(u) = c \sum_{v \in B(u)} \frac{PR(v)}{N_v} \qquad (1)$$

where o represents a web page, *B(u)* is the set of pages that point to o, *PR(u)* and *PR(v)* are rank scores of page o and v respectively, N, denotes the number of outgoing links of page *v, c* is a factor used for normalization.

In PageRank, the rank score of a page (say *p)* is equally divided among its outgoing links. The values assigned to the outgoing links of page *p* are in turn used to calculate the ranks of the pages pointed to by *p.*

Later PageRank was modified observing that not all users follow the direct links on WWW. The modified version is given in Eq. 2:

$$PR(o) = (1-d)dI \frac{PR(V)}{N} \qquad (2)$$

where *d* is a dampening factor that is usually set to 0.85. *d* can be thought of as the probability of users' following the direct links and (1 — **d)** as the page rank distribution from non- directly linked pages.

Power Method is an iterative method for computing single source and single-target PPR queries [45]. Recall that, at each step, an α-discounted random walk terminates at the current node with α probability or moves to a randomout-neighbor with (1−α) probability. This process can be expressed as the iteration formula with single-source PPR vector is shown in Eq. 3:

$$\underset{\pi\ s}{\rightarrow} = (1-\alpha) \underset{\pi\ s}{\rightarrow} .P + \alpha . \underset{e\ s}{\rightarrow} \qquad (3)$$

where $\underset{\pi\ s}{\rightarrow}$ denotes the PPR vector with respect to a given source node s, $\underset{e\ s}{\rightarrow}$ denotes the one-hot vector with $\underset{e\ s}{\rightarrow}$ (s) = 1, and P denotes the transition matrix is given in Eq. 4:

$$P(i,j) = \begin{cases} \frac{1}{dout(Vt)}, if\ Vj \in Nout(Vi) \\ 0, otherwise \end{cases} \qquad (4)$$

Power Method can be used to compute the ground truths for the single-source and single-target query. After l= $log_{1-\alpha}$ (ε) iterations, the absolute error can be bounded by $(1-\alpha)^l$= ε. Since each iteration takes O(m) time, it follows that the Power Method computes the approximate single-target PPR query with additive error in O(m.log$\frac{1}{\varepsilon}$) time. Note that the dependence on the error parameter ε is logarithmic, which implies that the Power Method can answer single-target PPR queries with high precision. However, the query time also linearly depends on the number of edges, which limits its scalability on large graphs.

2.1 Weighted Page Rank Algorithm

Wenpu Xing and Ali Ghorbani proposed an algorithm called weighted page rank (WPR). This weighted page rank algorithm is different from the traditional page rank algorithm in the fact that each outlink page has a page rank value proportional to its importance (number of inlinks and outlinks) instead of dividing it equally [12].

Win (v, u) = weight of link (v, u) or importance of web page due to inlinks
Wout(v, u) = weight of link (v, u) or importance of web page due to outlinks is in Eq. 5:

$$W^{in}_{(v,u)} = \frac{I_u}{\sum_{p \in R(v)} I_p} \qquad (5)$$

where, I_u and I_p denote the no. of inlinks of page u and page p respectively.
R(v) represents the reference page list of page v is in Eq. 6:

$$W^{out}_{(v,u)} = \frac{Ou}{\sum_{p \in R(v)} Op} \qquad (6)$$

where, Ou and Op denote the no. of outlinks of page u and page p respectively
R(v) represents the reference page list of page v
After calculating the importance of web pages, the modified page rank formula is given in Eq. 7:

$$PR(u) = (1-d) + d \sum_{v \in B9u)} PR(v) W^{n}_{(v,u)} W^{out}_{(v,u)} \qquad (7)$$

This Weighted Page Rank algorithm solves the problem of ranking web pages based on their relevancy or importance by considering the weight factor. But the problem of query independency and calculation of page ranks at indexing time still remain with WPR and with the traditional Page Ranking algorithm.

2.2 Iterative Page Rank Algorithm

It is easy to solve the equation system, to determine page rank values, for a small set of pages but the web consists of billions of documents and it is not possible to find a solution by inspection method. In iterative calculation, each page is assigned a starting page rank value. These rank values are iteratively substituted in page rank equations to find the final values. In general, many iterations could be followed to normalize the page ranks.

3. PERSONALIZED PAGE RANK

Personalized PageRank (PPR), as a variant of PageRank [8], focuses on the relative significance of a target nodewith respect to a source node in a graph. Given a directed graph G = (V, E) with n nodes and m edges, the PPR value π(s, t) of a target node t with respect to a source node s is defined as the probability that an α-discounted random walk from node s terminates at t . Here an α-discounted random walk represents a random traversal that, at each step, either terminates at the current node with probability α, or moves to a random out-neighbor with probability 1 − α. For a given source node s, the PPR value of each node t sum up to $\sum_{t\in V}\pi(s,t) = 1$, and thus π(s, t) reflects the significance of node t with respect to the source node s. On the other hand, PPR to a target node can be related to PageRank: the summation of PPR from each node s ∈ V to a given target node t is $\sum_{s\in V}\pi(s,t) = n \cdot \pi(t)$ where π(t) is the PageRank of t [8]. Large π(s, t) also shows the great contributions made for t 's PageRank, the overall importance of t . Therefore, π(s, t) indicates bidirectional importance between s and t .

3.1 Reverse Push

One local variation on Power Iteration starts at a given target node t and works backwards, computing an estimate p^t(s) of π_s(t) from every source s to the given target. This technique was first proposed by Jeh and Widom [2], and subsequently improved by other researchers [14]. The algorithms are primarily based on the following recurrence relation for π_u:

$$\pi_s(t) = \alpha e_s + \frac{(1-\alpha)}{N^{out(s)}} . \sum_{v\in N^{out(s)}} \pi_v(t)$$

Intuitively, this equation says that for s to decide how important t is, first s gives score α to itself, then adds the average opinion of its out-neighbors, scaled by 1 − α. Andersen et. al. [13,14] present and analyze a local algorithm for PPR based on this recurrence. This algorithm can be viewed as a message passing algorithm which starts with a message at the target. Each local push operation involves taking the message value (or \residual") r^t[v] at some node v, incorporating rt[v] into an estimate p^t[v] of π_v[t], and sending a message to each in-neighbors u ∈ N^{in}(v), informing them that p^t[v]]has increased. Because we use it in our bidirectional algorithm, we give the full pseudo-code here as Algorithm 1.

Algorithm 1 : Reverse Push(t, r_{max}, α)[14]

```
Inputs : graph G with edge weights $(W_{u,v})u,v \in V$, target node t, maximal residual $r_{max}$, teleport probability α
Initialize (sparse) estimate-vector $p_t = \vec{0}$ and (sparse) residual-vector $r_t = e_t$
  (i.e $r_t$ (v) =1 if v = t; else 0)
while ∃v ∈V s.t. $r_t > r_{max}$ do
    for u ∈ $N^{in}(v) do$
        $r_t$(u) += (1- α)$W_{u,v}$ $r_t(v)$
    end for
    $p_t(v) += \alpha r_t$(v)
    $r_t$(v) = 0
end while
return ($p_t$, $r_t$)
```

3.2 Forward Push

An alternative local version of power iteration starts from the start node s and works forward along edges. Variations on this were proposed in [15] and others, but the variation most useful for our work is in Andersen et. al. [16] because of the analysis they give. Because we use it a variation of our bidirectional algorithm, we give the full pseudo-code here as Algorithm 2.

Algorithm 2: Reverse Push(G,s, r_{max}, α) [16]

```
Inputs : graph G , maximal residual $r_{max}$, teleport probability α, start node s
Initialize (sparse) estimate-vector $p_s = \vec{0}$ and (sparse) residual-vector $r_s = e_t$
  (i.e $r_s$ (v) =1 if v = s; else 0)
while ∃v ∈V s.t. $\frac{r_{s(u)}}{d_u} > r_{max}$ do
    for v ∈ $N(u) do$
        $r_s$(v) += (1- α)$\frac{r_{s(u)}}{d_u}$
    end for
    $p_s(u) += \alpha r_s$(u)
    $r_s$(v) = 0
end while
return ($p_s$, $r_s$)
```

To our knowledge, there is no clean bound on the error || $p_s - \pi_s$|| as a function of r_{max} for a useful choice of norm. The difficulty is illustrated by the following graph: we have n nodes, (s,t,v_1, ……………..v_{n-2}), and 2(n-2) edges, (s, vi) and (vi,t) for each i. If we run ForwardPush on this graph starting from s with r_{max}= 1/n-2, then after pushing from s, the algorithm halts, with estimate at t of p_s (t) = 0. However, π_s(t) = (1), so the algorithm has a large error at t even thoughr_{max} is getting arbitrarily small as n → ∞.

The loop invariant in [16] does give a bound on the error of ForwardPush, but it is somewhat subtle, involving the personalized PageRank personalized to the resulting residual vector.

3.3 Monte-Carlo

The Monte-Carlo algorithm [17] computes the approximate single-source PPR query by sampling abundant random walks from source node s and using the proportion of the random walks that terminate at t as the estimator of π(s, t). According to Chernoff bound, the number of random walks required for an additive error ε is O $(\frac{1}{\varepsilon^2})$, while the number of random walks required to ensure constant relative error for all PPR larger than δ is O $(\frac{1}{\delta})$. This simple method is optimal for single-source PPR queries with relative error, as there are at most $O(\frac{1}{\delta})$ nodes t with PPR π(s, t) ≥ δ. However, the Monte-Carlo algorithm does not work for single-target queries, as there lacks of a mechanism for sampling source nodes from a given target node. Moreover, it remains an open problem whether it is possible to achieve the same optimal $O(\frac{1}{\delta})$ complexity for the single-target query.

4. CONCLUSIONS

In this paper, we discussed the various algorithms and techniques mainly used by search engines in ranking web pages on the internet. That mainly deals with the traditional pagerank algorithm, personalized pagerank algorithms and its different techniques. With the course of time the traditional page rank algorithm has been modified by adding many different factors. Google Page Rank Algorithm computes the page ranks of web pages only at the time of indexing and weighted pagerank algorithm is a modification of the google's pagerank algorithm. But these modification are not sufficient to cope with the increasing data or information on every web page day-by-day. There is a need of some kind of modified algorithm that can give results at the time of indexing as well as at the time of user query is called personalized pagerank algorithm. The existing algorithms may consider the bookmarked web pages in calculating the Page Rank of web pages. The Page Ranking algorithms are now finding applications not only in ranking web pages but extensively used in ranking research papers, suggesting user accounts to follow and in many other fields. Here personalized pageranking techniques includes reverse push, forward push and finally monte-carlo.

REFERENCES

[1] A. N. Langville and C. D. Meyer, "Deeper inside pagerank", Department of Mathematics, Center for Research in Scientific Computation, 2002.

[2] Glen Jeh, Jennifer Widom, "Scaling personalized websearch", In WWW, Pages 271-279, 2003.

[3] Reid Andersen, Christian Borgs, Jennifer Chayes, John Hopcroft, Kamal Jain,Vahab Mirrokni, and Shanghua Teng, "Robust pagerank and locally computablespam detection features", In Proceedings of the 4th international workshop on Adversarial information retrieval on the web, pages 69–76, 2008.

[4] Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Zadeh, "Wtf: The who to follow service at twitter", In WWW, pages 505–514.

[5] Johannes Klicpera, Stefan Weiasenberger, and Stephan Gaijnnemann, "Diffusion improves graph learning", 2019.

[6] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu, " Asymmetric transitivity preserving graph embedding", In SIGKDD, pages 1105–1114, ACM,2016.

[7] Hanzhi Wang, Zhewei Wei, Junhao Gan, Sibo Wang, "Personalized PageRank to a Target Node, Revisited", In Proceedings of the 26th Conference on Knowledge Discovery and Data mining, August 23-27, 2020.

[8] L. Page, S. Brin, R. Motwani, and T. Winograd, "The Pagerank CitationRanking: Bringing order to the Web", Technical report, Stanford DigitalLibraries SIDL-WP-1999-0120, 1999.

[9] C. Ridings and M. Shishigin, "Pagerank Uncovered". Technical report, 2002.

[10] http://WWW.webrankinfo.com/english/seo-news/topic-16388.html.

[11] http://www.goog1e.com/technology/index.html, Our Search: Google Technology.

[12] Wenpu Xing and Ali Ghorbani, "Weighted PageRank Algorithm",Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSR'04), 2004 IEEE.

[13] D. Fogaras, B. Racz,,T. Sarlces, "Towards Scaling fully Personalized Pagerank: Algorithms,lower bounds, and experiments", Internet Mathematics, 2005.

[14] R. Andersen, C. Borgs, J. chayes, J. Hopcraft, V. S. Mirrokni, "Local computation of pagerank contributions", In Algorithms and models for the web-graph, Springer, 2007.

[15] P. Berkhin, "Bookmark-coloring algorithm for personalized pagerank computing", Internet Mathematics, 3(1):41, 2006.

[16] R. Andersen, F. Chung and K.Lang, "Local graph partitioning using pagerank vectors", In Foundations of Computer Science 2006, FOCS'06 47th Annual IEEE Symposium on 2006.

[17] D. Carmel, N. Zwerdling, I. Guy, S. Ofek Koifman, N. Har'EI, I.Ronen, E. Uziel, S. Yogev and S. chernov, "Personalized social network", In proceedings of the 18th ACM conference on information and knowledge management, Pages 1227-1236, 2009.