PAGE RANK IMPLEMENTATION IN SEARCH ENGINE

Enroll no: 101223

Name of student: Umang Prateek

Name of supervisor: **Dr. Pardeep Kumar**

May-2014

Project Report submitted in partial fulfillment of the degree of

Bachelor of Technology

in

Computer Science & Engineering

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY,

WAKNAGHAT

.

(I)

TABLE OF CONTENTS

CERTIFICATE

This is to certify that project report entitled "**Page rank Implementation in Search Engine**", submitted by "**Umang Prateek"** in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor

Name of Supervisor **Dr. Pardeep Kumar**

Designation **Assistant Professor (Senior Grade)**

Date

ACKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of the project.

I would like to show my greatest appreciation to Dr. Pardeep Kumar. I can't say thank you enough for this tremendous support and help. I feel motivated and encouraged everytime I attended his meeting. Without his encouragement and guidance this project would not have materialized.

Signature of the student

Name of Student Umang Prateek

Date

SUMMARY

A user does not have the ability or patience to scan through all pages that contain the given query words. One expects the relevant pages to be displayed within the top 20-30 pages returned by the search engine and hence some page rank algorithm is required.

Modern search engines employ methods of ranking the results to provide the "best" results first that are more elaborate than just plain*text ranking*. One of the most known and influential algorithms for computing the relevance of web pages is the Page Rank algorithm used by the Google search engine. It was invented by Larry Page and Sergey Brin while they were graduate students at Stanford, and it became a Google trademark in 1998. The idea that Page Rank brought up was that, the importance of any web page can be judged by looking at the pages that link to it. If we create a web page *i* and include a hyperlink to the web page *j*, this means that we consider "*j"* important and relevant for our topic. If there are a lot of pages that link to *j*, this means that the common belief is that page *j* is important.

The web is very heterogeneous by its nature, and certainly huge, so we do not expect its graph to be connected. Likewise, there will be pages that are plain descriptive and contain no outgoing links. So, the solution was given by Page and Brin. We see that PageRank does not rank web sites as a whole, but is determined for each page individually. Further, the PageRank of page A is recursively defined by the PageRanks of those pages which link to page A.The PageRank of pages Ti which link to page A does not influence the PageRank of page A uniformly. Within the PageRank algorithm, the PageRank of a page T is always weighted by the number of outbound links C(T) on page T. This means that the more outbound links a page T has, the less will page A benefit from a link to it on page T.

The weighted PageRank of pages Ti is then added up. The outcome of this is that an additional inbound link for page A will always increase page A's PageRank.

Finally, the sum of the weighted PageRanks of all pages Ti is multiplied with a damping factor d which can be set between 0 and 1. Thereby, the extend of PageRank benefit for a page by another page linking to it is reduced.

Signature of Student	**Signature of Supervisor**
Name: Umang Prateek	**Name: Dr.Pardeep Kumar**
Date:	**Date:**

Chapter-1

Introduction

The World Wide Web creates many new challenges for information retrieval. It is very large and heterogeneous. Current estimates are that there are over 150 million web pages with a doubling life of less than one year. More importantly, the web pages are extremely diverse, ranging from"What is Joe having for lunch today?" to journals about information retrieval.

In addition to these major challenges, search engines on the Web must also contend with inexperienced users and pages engineered to manipulate search engine ranking functions. However, unlike "at" document collections, the World Wide Web is hypertext and provides considerable auxiliary information on top of the text of the web pages, such as link structure and link text. In this paper, we take advantage of the link structure of the Web to produce a global importance" ranking of every web page. This ranking, called PageRank, helps search engines and users quickly make sense of the vast heterogeneity of the World Wide Web.

In order to measure the relative importance of web pages, we propose PageRank, a method for computing a ranking for every web page based on the graph of the web. PageRank has applications in search, browsing, and traffic estimation.

PageRank is a link analysis algorithm and it assigns a numerical weighting to each element of a hyperlinked set of documents, such as the World Wide Web, with the purpose of "measuring" its relative importance within the set. The algorithm may be applied to any collection of entities with reciprocal quotations and references. The numerical weight that it assigns to any given element *E* is referred to as the*PageRank of E* and denoted by $PR(E)$.

Chapter-2

Background Material

The idea of formulating a link analysis problem as an eigenvalue problem was probably first suggested in 1976 by Gabriel Pinski and Francis Narin, who worked on scientometrics ranking scientific journals. PageRank was developed at Stanford University by Larry Page and Sergey Brin in 1996 as part of a research project about a new kind of search engine. Sergey Brin had the idea that information on the web could be ordered in a hierarchy by "link popularity": a page is ranked higher as there are more links to it. It was co-authored by Rajeev Motwani and Terry Winograd. The first paper about the project, describing PageRank and the initial prototype of the Google search engine, was published in 1998: shortly after, Page and Brin founded Google Inc., the company behind the Google search engine. While just one of many factors that determine the ranking of Google search results, PageRank continues to provide the basis for all of Google's web search tools.

The name "PageRank" plays off of the name of developer Larry Page, as well as the concept of a web page. The word is a trademark of Google, and the PageRank process has been patented . However, the patent is assigned to Stanford University and not to Google. Google has exclusive license rights on the patent from Stanford University. The university received 1.8 million shares of Google in exchange for use of the patent; the shares were sold in 2005 for $336 million.

PageRank was influenced by citation analysis, early developed by Eugene Garfield in the 1950s at the University of Pennsylvania, and by Hyper Search, developed by Massimo Marchiori at the University of Padua. In the same year PageRank was introduced (1998), Jon Kleinberg published his important work on HITS. Google's founders cite Garfield, Marchiori, and Kleinberg in their original papers.

A small search engine called "RankDex" from IDD Information Services designed by Robin Li was, since 1996, already exploring a similar strategy for site-scoring and page ranking. The technology in RankDex would be patented by 1999 and used later when Li founded Baidu in China. Li's work would be referenced by some of Larry Page's U.S. patents for his Google search methods.

Chapter-3

Directed Web Graph

The webgraph describes the directed links between pages of the World Wide Web. A graph, in general, consists of several vertices, some pairs connected by edges. In a directed graph, edges are directed lines or arcs. The webgraph is a directed graph, whose vertices correspond to the pages of the WWW, and a directed edge connects page X to page Y if there exists a hyperlink on page X, referring to page Y.

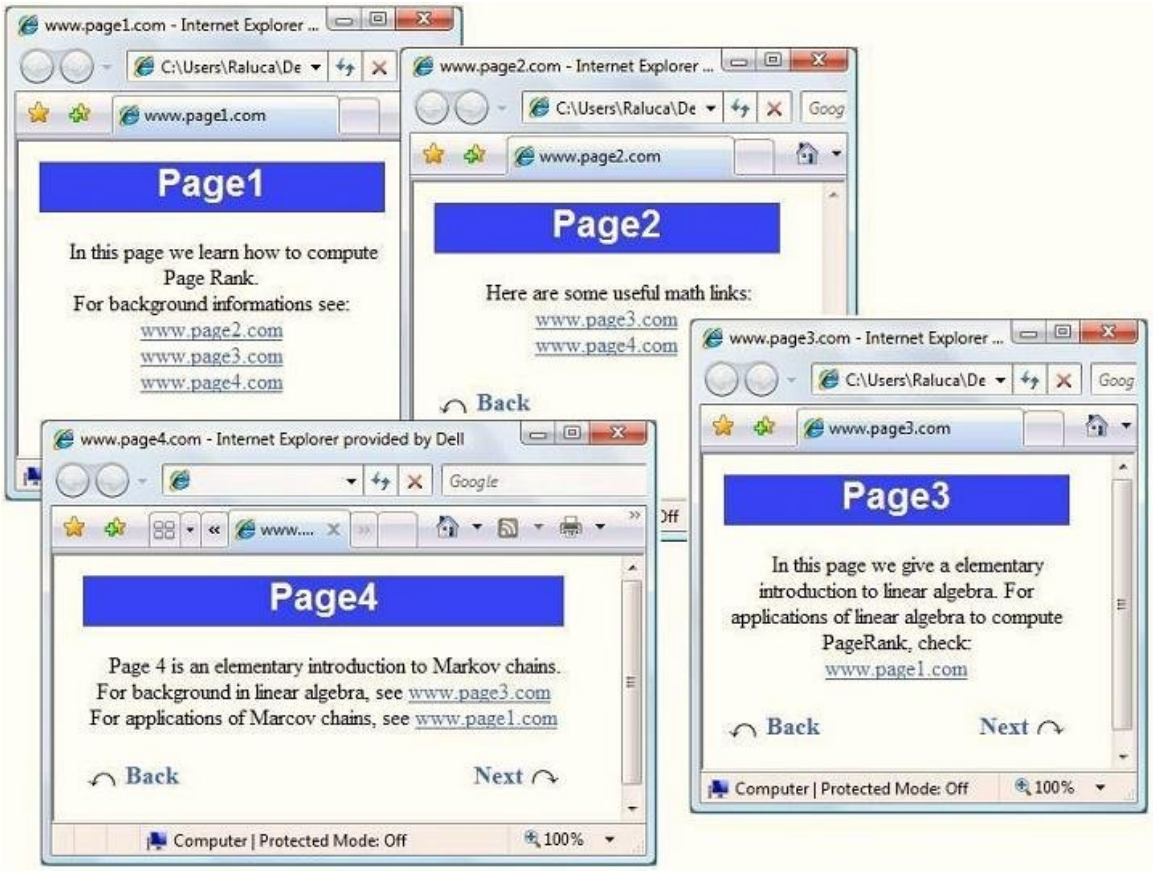

Fig.1

So, the web graph for the above web pages looks like:

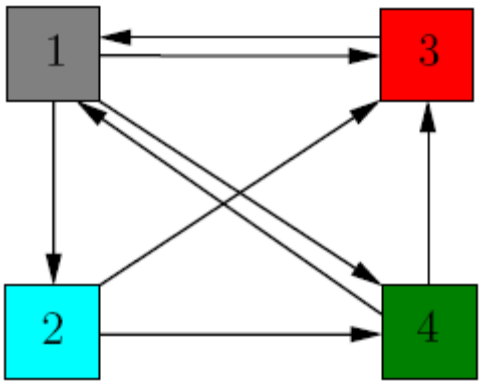

Fig.2

3.1 Properties

- The degree distribution of the webgraph strongly differs from the degree distribution of the classical random graph model, the Erdős–Rényi model:In the Erdős–Rényi model, there are very few large degree nodes, relative to the webgraph's degree distribution. The precise distribution is unclear, however: it is well described by a lognormaldistribution, as well as the Barabási–Albert model for power laws.
- The webgraph is an example of a scale-free network.

3.2 Applications

- The webgraph is used for computing the PageRank of the WWW pages.
- The webgraph is used for computing the personalized PageRank.
- The webgraph can be used for detecting webpages of similar topics, through graph-theoretical properties only, like co-citation.
- The webgraph is applied in the HITS algorithm for identifying hubs and authorities in the web

Chapter-4

Page Rank Algorithm

The idea that Page Rank brought up was that, the importance of any web page can be judged by looking at the pages that link to it. If we create a web page *i* and include a hyperlink to the web page *j*, this means that we consider "*j*" important and relevant for our topic. If there are a lot of pages that link to *j*, this means that the common belief is that page *j* is important. If on the other hand, *j* has only one backlink, but that comes from an authoritative site *k*, (like www.google.com, www.cnn.com, www.cornell.edu) we say that *k* transfers its authority to *j*; in other words, *k* asserts that *j* is important. Whether we talk about popularity or authority, we can iteratively assign a rank to each web page, based on the ranks of the pages that point to it.

To this aim, we begin by picturing the Web net as a directed graph, with nodes represented by web pages and edges represented by the links between them.

Suppose for instance, that we have a small Internet consisting of just 4 web sites www.page1.com, www.page2.com, www.page3.com, www.page4.com, referencing each other in the manner suggested by the picture above:

We "translate" the picture into a directed graph with 4 nodes, one for each web site. When web site *i* references *j*, we add a directed edge between node *i* and node *j* in the graph. For the purpose of computing their page rank, we ignore any navigational links such as back, next buttons, as we only care about the connections between different web sites. For instance, Page1 links to all of the other pages, so node 1 in the graph will have outgoing edges to all of the other nodes. Page3 has only one link, to Page 1, therefore node 3 will have one outgoing edge to node 1. After analyzing each web page, we get the following graph:

In our model, each page should transfer evenly its importance to the pages that it links to. Node 1 has 3 outgoing edges, so it will pass on $\frac{1}{3}$ of its importance to each of the other 3 nodes. Node 3 has only one outgoing edge, so it will pass on all of its importance to node 1. In general, if a node has *k* outgoing edges, it will pass on $\frac{1}{k}$ of its importance to each of the nodes that it links to. Let us better visualize the process by assigning weights to each edge.

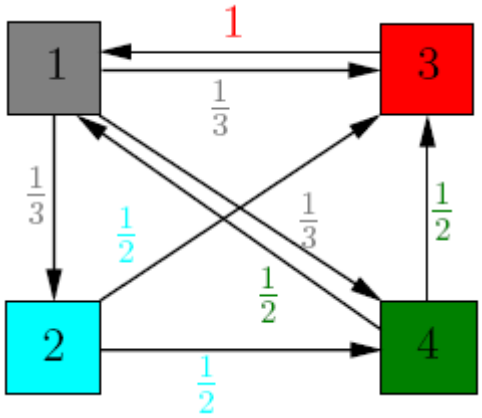

Fig.3

Let us denote by A the transition matrix of the graph, $A = \begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix}$.

Dynamical systems point of view:

Suppose that initially the importance is uniformly distributed among the 4 nodes, each getting ¼. Denote by v the initial rank vector, having all entries equal to ¼. Each incoming link increases the importance of a web page, so at step 1, we update the rank of each page by adding to the current value the importance of the incoming links. This is the same as multiplying the matrix A with v . At step 1, the new importance vector is $v_1 = Av$. We can iterate the process, thus at step 2, the updated importance vector is $v_2 = A(Av) = A^2v$. Numeric computations give:

$$v = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix}, \quad Av = \begin{pmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{pmatrix}, \quad A^2 v = A\,(Av) = A \begin{pmatrix} 0.37 \\ 0.08 \\ 0.33 \\ 0.20 \end{pmatrix} = \begin{pmatrix} 0.43 \\ 0.12 \\ 0.27 \\ 0.16 \end{pmatrix}$$

$$A^3 v = \begin{pmatrix} 0.35 \\ 0.14 \\ 0.29 \\ 0.20 \end{pmatrix}, \quad A^4 v = \begin{pmatrix} 0.39 \\ 0.11 \\ 0.29 \\ 0.19 \end{pmatrix}, \quad A^5 v = \begin{pmatrix} 0.39 \\ 0.13 \\ 0.28 \\ 0.19 \end{pmatrix}$$

$$A^6 v = \begin{pmatrix} 0.38 \\ 0.13 \\ 0.29 \\ 0.19 \end{pmatrix}, \quad A^7 v = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}, \quad A^8 v = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}$$

We notice that the sequences of iterates $v, Av, \ldots, A^k v$ tends to the equilibrium value $v^* = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}$. We call this the PageRank vector of our web graph.

Linear algebra point of view:

Let us denote by x_1, x_2, x_3, and x_4 the importance of the four pages. Analyzing the situation at each node we get the system:

$$\begin{cases} x_1 = 1 \cdot x_3 + \frac{1}{2} \cdot x_4 \\ x_2 = \frac{1}{3} \cdot x_1 \\ x_3 = \frac{1}{3} \cdot x_1 + \frac{1}{2} \cdot x_2 + \frac{1}{2} \cdot x_4 \\ x_4 = \frac{1}{3} \cdot x_1 + \frac{1}{2} \cdot x_2 \end{cases}$$

This is equivalent to asking for the solutions of the equations $A \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$.

We know that the eigenvectors corresponding to the eigenvalue 1 are of the form $c \cdot \begin{bmatrix} 12 \\ 4 \\ 9 \\ 6 \end{bmatrix}$. Since PageRank should reflect only the relative importance of the nodes, and since the eigenvectors are just scalar multiples of each other, we can choose any of them to be our PageRank vector. Choose v^* to be the unique eigenvector with the sum of all entries equal to 1. (We will sometimes refer to it as the probabilistic eigenvector corresponding to the eigenvalue 1). The eigenvector $\frac{1}{31} \cdot \begin{bmatrix} 12 \\ 4 \\ 9 \\ 6 \end{bmatrix} \sim \begin{bmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{bmatrix}$ is our PageRank vector.

Probabilistic point of view:

Since the importance of a web page is measured by its popularity (how many incoming links it has), we can view the importance of page i as the probability that a random surfer on the Internet that opens a browser to any page and starts following hyperlinks, visits the page i. We can interpret the weights we assigned to the edges of the graph in a probabilistic way: A random

surfer that is currently viewing web page 2, has ½ probability to go to page 3, and ½ probability to go to page 4. We can model the process as a random walk on graphs. Each page has equal probability ¼ to be chosen as a starting point. So, the initial probability distribution is given by the column vector $[¼\ ¼\ ¼\ ¼]^t$. The probability that page *i* will be visited after one step is equal to Ax, and so on. The probability that page *i* will be visited after *k* steps is equal to A^kx. The sequence $Ax, A^2x, A^3x, ..., A^kx, ...$ converges in this case to a unique probabilistic vector v^*. In this context v^* is called the stationary distribution and it will be our Page Rank vector. Moreover, the i^{th}entry in the vector v^* is simply the probability that at each moment a random surfer visits page *i*. The computations are identical to the ones we did in the dynamical systems interpretation, only the meaning we attribute to each step being slightly different.

The Page Rank vector v^* we have computed by different methods, indicates that page 1 is the most relevant page. This might seem surprising since page 1 has 2 backlinks, while page 3 has 3 backlinks. If we take a look at the graph, we see that node 3 has only one outgoing edge to node 1, so it transfers all its importance to node 1. Equivalently, once a web surfer that only follows hyperlinks visits page 3, he can only go to page 1. Notice also how the rank of each page is not trivially just the weighted sum of the edges that enter the node. Intuitively, at step 1, one node receives an importance vote from its direct neighbors, at step 2 from the neighbors of its neighbors, and so on.

Changing the web graph might lead to certain problem.

4.1 Implementation:

```
import java.io.*;

import java.math.*;

public class PgRank {


    public static void main(String[] args)throws IOException {

        try

        {

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

int n,i,j;

double k;

int c,count=0;
```

```
System.out.println("Enter the no of web pages");
n=Integer.parseInt(br.readLine());
double a[][]=new double[n][n];
for(j=0;j<n;j++)
{
        System.out.println("Enter no. of outlinks for node:"+(j+1));
         k=Double.parseDouble(br.readLine());
         for(i=0;i<n;i++)
         {
        System.out.print("Is there a link b/w node:"+(j+1)+(i+1));
         System.out.println("press 1 or 0");
         c=Integer.parseInt(br.readLine());
         if(c==1)
         a[i][j]=(double)(1/k);
         }
}
System.out.println("Transition Matrix is:");
for(i=0;i<n;i++)
{
         for(j=0;j<n;j++)
         System.out.print(a[i][j]+"\t\t\t\t ");
         System.out.println();
}
System.out.println();
System.out.println();
```

```
double v[][]=new double[n][1];
double c1[][]=new double[n][1];
double c2[][]=new double[n][1];
double c3[][]=new double[n][1];
double c4[][]=new double[n][1];
System.out.println("The initial page rank vector is:");
for(i=0;i<n;i++)
{
        for(j=0;j<1;j++)
        {
        v[i][j]=(double)(1.0/(double)n);
        System.out.println(v[i][j]+" ");
}
}
System.out.println();
System.out.println("After iteration no:1 the page rank vector is");

for(i=0;i<n;i++)
         {
         c1[i][0]=0;
         for(j=0;j<n;j++)
        {
     c1[i][0]+=(a[i][j]*v[j][0]);}
    System.out.println(c1[i][0]);
        }
```

```
System.out.println();
System.out.println();
System.out.println();
 count=0;
        j=0;
        boolean count1=false;
        for(i=0;i<n;i++)
        {
        count1=check(c1,v);

        }
        while(count1)
                {
                for(j=0;j<n;j++)
                {
                c3[j][0]=c1[j][0];
                }
                for(i=0;i<n;i++)
                {
                c2[i][0]=0;
                for(j=0;j<n;j++)
                {
                c2[i][0]+=(a[i][j]*c1[j][0]);}}
for(i=0;i<n;i++)
{
```

```
c1[i][0]=c2[i][0];
}

for(i=0;i<n;i++)
        {
        System.out.print(c1[i][0]+"\t\t\t\t\t");
        v[i][0]=c3[i][0];
        System.out.println(v[i][0]+" ");
        }
System.out.println();
System.out.println();
for(i=0;i<n;i++)
{
count1=check(c1,v);
}



}
private static boolean check(double[][] c1, double[][] v) {
        int i=0;
        int f=0;
        for(i=0;i<c1.length;i++)
        {
            if(c1[i][0]!=v[i][0])
            {
```

```
                    f=1;
                return true;

                    }

    }
        if(f==0)
        {
        return false;
        }
        return false;
    }
```

Test run:

Enter the no of web pages

4

Enter no. of outlinks for node:1

3

Is there a link b/w node:1&1press 1 or 0

0

Is there a link b/w node:1&2press 1 or 0

1

Is there a link b/w node:1&3press 1 or 0

1

Is there a link b/w node:1&4press 1 or 0

1

Enter no. of outlinks for node:2

2

Is there a link b/w node:2&1press 1 or 0

0

Is there a link b/w node:2&2press 1 or 0

0

Is there a link b/w node:2&3press 1 or 0

1

Is there a link b/w node:2&4press 1 or 0

1

Enter no. of outlinks for node:3

1

Is there a link b/w node:3&1press 1 or 0

1

Is there a link b/w node:3&2press 1 or 0

0

Is there a link b/w node:3&3press 1 or 0

0

Is there a link b/w node:3&4press 1 or 0

0

Enter no. of outlinks for node:4

2

Is there a link b/w node:4&1press 1 or 0

1

Is there a link b/w node:4&2press 1 or 0

0

Is there a link b/w node:4&3press 1 or 0

1

Is there a link b/w node:4&4press 1 or 0

0

Transition Matrix is:

$$\begin{bmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{bmatrix}$$

The initial page rank vector is:

$$\begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix}$$

After iteration no:1 the page rank vector is

$$\begin{pmatrix} 0.375 \\ 0.083 \\ 0.33 \\ 0.208 \end{pmatrix}$$

Comparing the two matrices…

$$\begin{pmatrix} 0.4375 \\ 0.125 \\ 0.270 \\ 0.166 \end{pmatrix} \qquad \begin{pmatrix} 0.375 \\ 0.083 \\ 0.33 \\ 0.208 \end{pmatrix}$$

$$\begin{pmatrix} 0.354 \\ 0.145 \\ 0.291 \\ 0.208 \end{pmatrix} \qquad \begin{pmatrix} 0.4375 \\ 0.125 \\ 0.270 \\ 0.166 \end{pmatrix}$$

$$\begin{pmatrix} 0.395 \\ 0.118 \\ 0.295 \\ 0.190 \end{pmatrix} \qquad \begin{pmatrix} 0.354 \\ 0.145 \\ 0.291 \\ 0.208 \end{pmatrix}$$

$$\begin{pmatrix} 0.390 \\ 0.131 \\ 0.286 \\ 0.190 \end{pmatrix} \qquad \begin{pmatrix} 0.395 \\ 0.118 \\ 0.295 \\ 0.190 \end{pmatrix}$$

$$\begin{pmatrix} \boldsymbol{0.381} \\ 0.130 \\ 0.291 \\ 0.196 \end{pmatrix} \qquad \begin{pmatrix} \boldsymbol{0.390} \\ 0.131 \\ 0.286 \\ 0.190 \end{pmatrix}$$

$$\begin{pmatrix} 0.389 \\ 0.127 \\ 0.290 \\ 0.192 \end{pmatrix} \qquad \begin{pmatrix} 0.381 \\ 0.130 \\ 0.291 \\ 0.196 \end{pmatrix}$$

$$\begin{pmatrix} 0.386 \\ 0.129 \\ 0.289 \\ 0.193 \end{pmatrix} \qquad \begin{pmatrix} 0.389 \\ 0.127 \\ 0.290 \\ 0.192 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{0.386} \\ \mathbf{0.128} \\ \mathbf{0.290} \\ \mathbf{0.193} \end{pmatrix} \qquad \begin{pmatrix} \mathbf{0.386} \\ \mathbf{0.128} \\ \mathbf{0.290} \\ \mathbf{0.193} \end{pmatrix}$$

4.2 Result:

We notice that the sequences of iterates v tends to the equilibrium value

$v^* = \begin{pmatrix} 0.38 \\ 0.12 \\ 0.29 \\ 0.19 \end{pmatrix}$. We call this the Page Rank vector of our web graph.

So, the page **1** has the **highest** rank followed by page 3,page 4 and page 2

Chapter 5

Dangling Node

Nodes with no outgoing edges **(dangling nodes)**

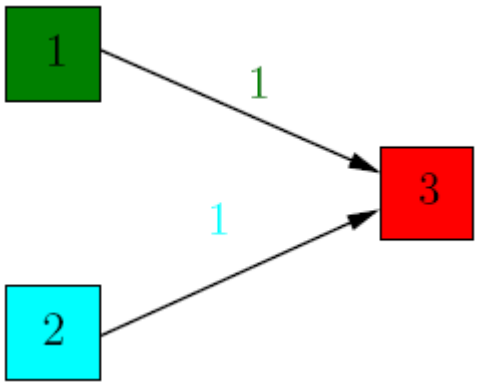

Fig.4

We iteratively compute the rank of the 3 pages:

$$v_0 = \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix}, \quad v_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{3} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \frac{2}{3} \end{bmatrix}, \quad v_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ \frac{2}{3} \end{bmatrix} =$$

.

$$= \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

So in this case the rank of every page is 0. This is counterintuitive, as page 3 has 2 incoming links, so it must have some importance!

Disconnected components

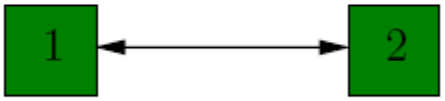

Fig.6

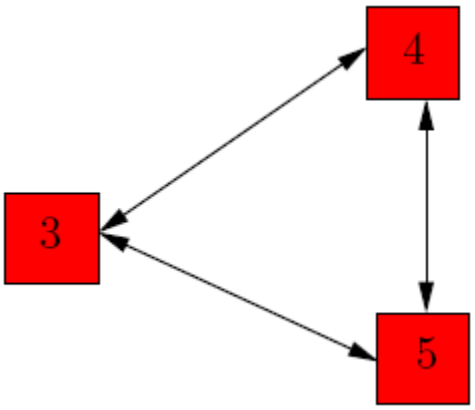

Fig.7

A random surfer that starts in the first connected component has no way of getting to web page 5 since the nodes 1 and 2 have no links to node 5 that he can follow. Linear algebra fails to help as well. The **transition matrix** for this graph is $A = \left[\begin{array}{cc|ccc} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{array}\right]$. Notice that $v = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $u = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$ are both eigenvectors corresponding to the eigenvalue 1, and they are not just trivially one the a scalar multiple of the other. So, both in theory and in practice, the notation of ranking pages from the first connected component relative to the ones from the second connected component is ambiguous.

The web is very heterogeneous by its nature, and certainly huge, so we do not expect its graph to be connected. Likewise, there will be pages that are plain descriptive and contain no outgoing links. What is to be done in this case? We need a non ambiguous meaning of the rank of a page, for any directed Web graph with *n* nodes.

5.1 Dangling node fix and the damping factor:

The original PageRank algorithm was described by **Lawrence Page** and **Sergey Brin** in several publications. It is given by

PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn)) **…..**Eqn. 1

where

- PR(A) is the PageRank of page A,
- PR(Ti) is the PageRank of pages Ti which link to page A,
- C(Ti) is the number of outbound links on page Ti and
- **d** is a **damping factor** which can be set between **0** and **1**.

So, first of all, we see that PageRank does not rank web sites as a whole, but is determined for each page individually. Further, the PageRank of page A is recursively defined by the PageRanks of those pages which link to page A.

The PageRank of pages Ti which link to page A does not influence the PageRank of page A uniformly. Within the PageRank algorithm, the PageRank of a page T is always weighted by the number of outbound links C(T) on page T. This means that the more outbound links a page T has, the less will page A benefit from a link to it on page T.

The weighted PageRank of pages Ti is then added up. The outcome of this is that an additional inbound link for page A will always increase page A's PageRank.

Finally, the sum of the weighted PageRanks of all pages Ti is multiplied with a damping factor d which can be set between 0 and 1. Thereby, the extend of PageRank benefit for a page by another page linking to it is reduced.

5.2 The Random Surfer Model

In their publications, Lawrence Page and Sergey Brin give a very simple intuitive justification for the PageRank algorithm. They consider PageRank as a model of user behaviour, where a surfer clicks on links at random with no regard towards content.

The random surfer visits a web page with a certain probability which derives from the page's PageRank. The probability that the random surfer clicks on one link is solely given by the number of links on that page. This is why one page's PageRank is not completely passed on to a page it links to, but is devided by the number of links on the page.

So, the probability for the random surfer reaching one page is the sum of probabilities for the random surfer following links to this page. Now, this probability is reduced by the damping factor d. The justification within the Random Surfer Model, therefore, is that the surfer does not click on an infinite number of links, but gets bored sometimes and jumps to another page at random.

The probability for the random surfer not stopping to click on links is given by the damping factor d, which is, depending on the degree of probability therefore, set between 0 and 1. The higher d is, the more likely will the random surfer keep clicking links. Since the surfer jumps to another page at random after he stopped clicking links, the probability therefore is implemented as a constant (1-d) into the algorithm. Regardless of inbound links, the probability for the random surfer jumping to a page is always (1-d), so a page has always a minimum PageRank.

5.3 A Different Notation of the PageRank Algorithm

Lawrence Page and Sergey Brin have published two different versions of their PageRank algorithm in different papers. In the second version of the algorithm, the PageRank of page A is given as

PR(A) = (1-d) / N + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn)) …….Eqn.2

where N is the total number of all pages on the web. The second version of the algorithm, indeed, does not differ fundamentally from the first one. Regarding the Random Surfer Model, the second version's PageRank of a page is the actual probability for a surfer reaching that page after clicking on many links. The PageRanks then form a probability distribution over web pages, so the sum of all pages' PageRanks will be one.

Contrary, in the first version of the algorithm the probability for the random surfer reaching a page is weighted by the total number of web pages. So, in this version PageRank is an expected value for the random surfer visiting a page, when he restarts this procedure as often as the web has pages. If the web had 100 pages and a page had a PageRank value of 2, the random surfer would reach that page in an average twice if he restarts 100 times.

As mentioned above, the two versions of the algorithm do not differ fundamentally from each other. A PageRank which has been calculated by using the second version of the algorithm has to be multiplied by the total number of web pages to get the according PageRank that would have been caculated by using the first version. Even Page and Brin mixed up the two algorithm versions in their most popular paper "The Anatomy of a Large-Scale Hypertextual Web Search Engine", where they claim the first version of the algorithm to form a probability distribution over web pages with the sum of all pages' PageRanks being one.

In the following, we will use the first version of the algorithm. The reason is that PageRank calculations by means of this algorithm are easier to compute, because we can disregard the total number of web pages.

The Characteristics of PageRank

The characteristics of PageRank shall be illustrated by a small example.

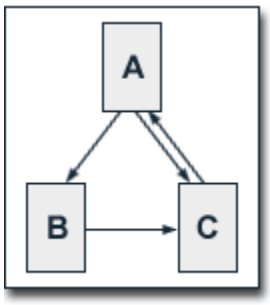

Fig.7

We regard a small web consisting of three pages A, B and C, whereby page A links to the pages B and C, page B links to page C and page C links to page A. According to Page and Brin, the damping factor d is usually set to 0.85, but to keep the calculation simple we set it to 0.5. The exact value of the damping factor d admittedly has effects on PageRank, but it does not influence the fundamental principles of PageRank. So, we get the following equations for the PageRank calculation:

PR(A)=0.5+0.5(PR(C) …Eqn.3
PR(B)=0.5+0.5(PR(A)/2) …Eqn.4
PR(C) = 0.5 + 0.5 (PR(A) / 2 + PR(B)) …Eqn.5

These equations can easily be solved. We get the following PageRank values for the single pages:

PR(A)=14/13=1.07692308
PR(B)=10/13=0.76923077
PR(C) = 15/13 = 1.15384615

It is obvious that the **sum** of all pages' **PageRanks** is **3** and thus equals the total number of web pages. As shown above this is not a specific result for our simple example.

For our simple three-page example it is easy to solve the according equation system to determine PageRank values. In practice, the web consists of billions of documents and it is not possible to find a solution by inspection.

5.4 Implementation:

```
System.out.println("Lawrence Page and Sergey Brin Page Rank Algo....");
        double damp=0.5,determinant=0.0;
        double pr1=0,pr2=0,pr3=0,n1;
        double matA[][]=new double[n][n];
        double matX[][]=new double[n][1];
        double matB[][]=new double[n][1];
        double matA1[][]=new double[n][n];
        double mattp[][]=new double[n][n];
        pr1=damp+damp*(pr3/1.0);
        pr2=damp+damp*(pr1/2.0);
        pr3=damp+damp*(pr1/2.0+pr2);
        System.out.println("Enter coefficient matrix..");
        for(i=0;i<n;i++)
        {
                for(j=0;j<n;j++)
                {
                        n1=Double.parseDouble(br.readLine());
                        matA[i][j]=n1;
                }

        }

        for(i=0;i<n;i++)
        {
```

```
        for(j=0;j<1;j++)
                {
                matB[i][j]=0.5;


                }
        }
        for(i=0;i<n;i++)
            determinant = determinant +
            (matA[0][i]*(matA[1][(i+1)%n]*matA[2][(i+2)%n] - matA[1][(i+2)%n]*matA[2][(i+1)%n]));
         for(i=0;i<n;i++){
            for(j=0;j<n;j++)
            {
                  matA1[i][j]=matA[(i+1)%n][(j+1)%n] * matA[(i+2)%n][(j+2)%n] - (matA[(i+1)%n][(j+2)%n]*matA[(i+2)%n][(j+1)%n]);



            }
            System.out.println();
                }
          //"Taking Transpose...."
          for(i=0;i<n;i++)
          {
                for(j=0;j<n;j++)
                {
                        mattp[i][j]=matA1[j][i]/determinant;
```

```
        }
}

System.out.println("Page rank vector is");
System.out.println();

for(i=0;i<n;i++)
                {

        c4[i][0]=0;

        for(j=0;j<n;j++)

          {

          c4[i][0]+=(mattp[i][j]*matB[j][0]);

          }
      System.out.println(c4[i][0]);

        }
```

5.5 Test run:

With Eqn No.1…..

Lawrence Page and Sergey Brin Page Rank Algo....

Enter coefficient matrix terms...

1

0

-0.5

-0.25

1

0

-0.25

-0.5

1

Inverse of matrix is:

$$\begin{pmatrix} 1.0 & 0.25 & 0.375 \\ 0.25 & 0.875 & 0.5 \\ 0.5 & 0.125 & 1.0 \end{pmatrix}$$

Page rank vector is

$$\begin{pmatrix} 1.07 \\ 0.76 \\ 1.15 \end{pmatrix}$$

With Eqn.2:

The other variation of Page and Brin..

Enter coefficient matrix terms..

1

0

-0.5

-0.25

1

0

-0.25

-0.5

1

Page rank vector in this case is

$$\begin{pmatrix} 0.179 \\ 0.128 \\ 0.192 \end{pmatrix}$$

Chapter-6

Page Rank Calculation

6.1 The Iterative Computation of PageRank

Because of the size of the actual web, the Google search engine uses an approximative, iterative computation of PageRank values. This means that each page is assigned an initial starting value and the PageRanks of all pages are then calculated in several computation circles based on the equations determined by the PageRank algorithm. The iterative calculation shall again be illustrated by our three-page example, whereby each page is assigned a starting PageRank value of 1.

Iteration	PR(A)	PR(B)	PR(C)
0	1	1	1
1	1	0.75	1.125
2	1.0625	0.76562	1.1484375
3	1.074218	0.76855	1.15283203
4	1.076416	0.76910	1.15365601
5	1.076828	0.76920	1.15381050
6	1.076905	0.76922	1.15383947
7	1.076919	0.76922	1.15384490
8	1.076922	0.76923	1.15384592
9	1.076922	0.76923	1.15384611
10	1.076923	0.76923	1.15384615
11	1.07692307	0.76923	1.15384615
12	1.07692308	0.76923	1.15384615

We see that we get a good approximation of the real PageRank values after only a few iterations. According to publications of Lawrence Page and Sergey Brin, about 100 iterations are necessary to get a good approximation of the PageRank values of the whole web.

Also, by means of the iterative calculation, the sum of all pages' PageRanks still converges to the total number of web pages. So the average PageRank of a web page is 1. The minimum PageRank of a page is given by (1-d). Therefore, there is a maximum PageRank for a page which is given by dN+(1-d), where N is total number of web pages. This maximum can theoretically occur, if all web pages solely link to one page, and this page also solely links to itself.

Iterative

At $t = 0$, an initial probability distribution is assumed, usually

$$PR(p_i;0) = \frac{1}{N}.$$

At each time step, the computation, as detailed above, yields

$$PR(p_i;t+1) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j;t)}{L(p_j)} \quad \text{,….Eqn.6}$$

or in matrix notation

$$\mathbf{R}(t+1) = d\mathcal{M}\mathbf{R}(t) + \frac{1-d}{N}\mathbf{1}, \quad (*) \quad \text{….Eqn.7}$$

where $\mathbf{R}_i(t) = PR(p_i;t)$ and $\mathbf{1}$ is the column vector of length N containing only ones.

The matrix $\mathcal{M}$ is defined as

$$\mathcal{M}_{ij} = \begin{cases} 1/L(p_j), & \text{if } j \text{ links to } i \\ 0, & \text{otherwise} \end{cases}$$

i.e.,

$$\mathcal{M} := (K^{-1}A)^T,$$

where A denotes the adjacency matrix of the graph and K is the diagonal matrix with the outdegrees in the diagonal.

The computation ends when for some small ϵ

$$|\mathbf{R}(t+1) - \mathbf{R}(t)| < \epsilon,$$

i.e., when convergence is assumed.

6.2 Algebraic

For $t \to \infty$(i.e., in the steady state), the above equation (*) reads

$$\mathbf{R} = d\mathcal{M}\mathbf{R} + \frac{1-d}{N}\mathbf{1}. \quad (**) \qquad \text{…….Eqn.8}$$

The solution is given by

$$\mathbf{R} = (\mathbf{I} - d\mathcal{M})^{-1}\frac{1-d}{N}\mathbf{1}, \qquad \text{..……Eqn.9}$$

with the identity matrix $\mathbf{I}$.

The solution exists and is unique for $0 < d < 1$. This can be seen by noting that $\mathcal{M}$is by construction a stochastic matrix and hence has an eigenvalue equal to one as a consequence of the Perron–Frobenius theorem.

Power Method

If the matrix $\mathcal{M}$is a transition probability, i.e., column-stochastic with no columns consisting of just zeros and $\mathbf{R}$is a probability distribution (i.e., $|\mathbf{R}| = 1$, $\mathbf{E}\mathbf{R} = 1$where $\mathbf{E}$is matrix of all ones), Eq. (**) is equivalent to

$$\mathbf{R} = \left(d\mathcal{M} + \frac{1-d}{N}\mathbf{E}\right)\mathbf{R} =: \widehat{\mathcal{M}}\mathbf{R}. \qquad \text{……Eqn.10}$$

Hence PageRank $\mathbf{R}$is the principal eigenvector of $\widehat{\mathcal{M}}$. A fast and easy way to compute this is using the power method: starting with an arbitrary vector $x(0)$, the operator $\widehat{\mathcal{M}}$is applied in succession, i.e.,

$$x(t+1) = \widehat{\mathcal{M}}x(t),$$

until

$$|x(t+1) - x(t)| < \epsilon.$$

Note that in Eq. (***) the matrix on the right-hand side in the parenthesis can be interpreted as

$$\frac{1-d}{N}\mathbf{I} = (1-d)\mathbf{P}\mathbf{1}^t,$$

where $\mathbf{P}$ is an initial probability distribution. In the current case

$$\mathbf{P} := \frac{1}{N}\mathbf{1}.$$

Finally, if $\mathcal{M}$ has columns with only zero values, they should be replaced with the initial probability vector $\mathbf{P}$. In other words

$$\mathcal{M}' := \mathcal{M} + \mathcal{D},$$

where the matrix $\mathcal{D}$ is defined as

$$\mathcal{D} := \mathbf{P}\mathbf{D}^t,$$

with

$$\mathbf{D}_i = \begin{cases} 1, & \text{if } L(p_i) = 0 \\ 0, & \text{otherwise} \end{cases}$$

In this case, the above two computations using $\mathcal{M}$ only give the same PageRank if their results are normalized:

$$\mathbf{R}_{\text{power}} = \frac{\mathbf{R}_{\text{iterative}}}{|\mathbf{R}_{\text{iterative}}|} = \frac{\mathbf{R}_{\text{algebraic}}}{|\mathbf{R}_{\text{algebraic}}|}$$

Chapter-7

Variations in Page Rank Algorithm

PageRank of an undirected graph:

The PageRank of an undirected graph G is statistically close to the degree distribution of the graph G,] but they are generally not identical: If R is the PageRank vector defined above, and D is the degree distribution vector

$$D = \frac{1}{2|E|}\begin{bmatrix} deg(p_1) \\ deg(p_2) \\ \vdots \\ deg(p_N) \end{bmatrix}$$

where $deg(p_i)$ denotes the degree of vertex p_i, and E is the edge-set of the graph, then, with $Y = \frac{1}{N}\mathbf{1}$, by:

$$\frac{1-d}{1+d}\|Y - D\|_1 \leq \|R - D\|_1 \leq \|Y - D\|_1,$$

that is, the PageRank of an undirected graph equals to the degree distribution vector if and only if the graph is regular, i.e., every vertex has the same degree.

Distributed Algorithm for PageRank Computation

There are simple and fast random walk-based distributed algorithms for computing PageRank of nodes in a network. They present a simple algorithm that takes $O(\log n/\epsilon)$ rounds with high probability on any graph (directed or undirected), where n is the network size and ϵ is the reset probability ($1-\epsilon$ is also called as damping factor) used in the PageRank computation. They also present a faster algorithm that takes $O(\sqrt{\log n}/\epsilon)$ rounds in undirected graphs. Both of the above algorithms are scalable, as each node processes and sends only small (polylogarithmic in n, the network size) number of bits per round. For directed graphs, they present an algorithm that has a running time of $O(\sqrt{\log n/\epsilon})$, but it requires a polynomial number of bits to processed and sent per node in a round.

Google Toolbar

The Google Toolbar's PageRank feature displays a visited page's PageRank as a whole number between 0 and 10. The most popular websites have a PageRank of 10. The least have a PageRank of 0. Google has not disclosed the specific method for determining a Toolbar PageRank value, which is to be considered only a rough indication of the value of a website.

PageRank measures the number of sites that link to a particular page. The PageRank of a particular page is roughly based upon the quantity of inbound links as well as the PageRank of the pages providing the links. The algorithm also includes other factors, such as the size of a page, the number of changes, the time since the page was updated, the text in headlines and the text in hyperlinked anchor texts.

The Google Toolbar's PageRank is updated infrequently, so the values it shows are often out of date.

SERP Rank

The search engine results page (SERP) is the actual result returned by a search engine in response to a keyword query. The SERP consists of a list of links to web pages with associated text snippets. The SERP rank of a web page refers to the placement of the corresponding link on the SERP, where higher placement means higher SERP rank. The SERP rank of a web page is a function not only of its PageRank, but of a relatively large and continuously adjusted set of factors (over 200),. Search engine optimization (SEO) is aimed at influencing the SERP rank for a website or a set of web pages.

Positioning of a webpage on Google SERPs for a keyword depends on relevance and reputation, also known as authority and popularity. PageRank is Google's indication of its assessment of the reputation of a webpage: It is non-keyword specific. Google uses a combination of webpage and website authority to determine the overall authority of a webpage competing for a keyword. The PageRank of the HomePage of a website is the best indication Google offers for website authority.

After the introduction of Google Places into the mainstream organic SERP, numerous other factors in addition to PageRank affect ranking a business in Local Business Results.

Google directory PageRank

The Google Directory PageRank was an 8-unit measurement. Unlike the Google Toolbar, which shows a numeric PageRank value upon mouseover of the green bar, the Google Directory only displayed the bar, never the numeric values. Google Directory was closed on July 20, 2011.

False or spoofed PageRank

In the past, the PageRank shown in the Toolbar was easily manipulated. Redirection from one page to another, either via a HTTP 302 response or a "Refresh" meta tag, caused the source page to acquire the PageRank of the destination page. Hence, a new page with PR 0 and no incoming links could have acquired PR 10 by redirecting to the Google home page. This spoofing technique, also known as 302 Google Jacking, was a known vulnerability. Spoofing can generally be detected by performing a Google search for a source URL; if the URL of an entirely different site is displayed in the results, the latter URL may represent the destination of a redirection.

Manipulating PageRank

For search engine optimization purposes, some companies offer to sell high PageRank links to webmasters. As links from higher-PR pages are believed to be more valuable, they tend to be more expensive. It can be an effective and viable marketing strategy to buy link advertisements on content pages of quality and relevant sites to drive traffic and increase a webmaster's link popularity. However, Google has publicly warned webmasters that if they are or were discovered to be selling links for the purpose of conferring PageRank and reputation, their links will be devalued (ignored in the calculation of other pages' PageRanks). The practice of buying and selling links is intensely debated across the Webmaster community. Google advises webmasters to use the nofollow HTML attribute value on sponsored links. According to Matt Cutts, Google is concerned about webmasters who try to game the system, and thereby reduce the quality and relevance of Google search results.

The intentional surfer model

The original PageRank algorithm reflects the so-called random surfer model, meaning that the PageRank of a particular page is derived from the theoretical probability of visiting that page when clicking on links at random. A page ranking model that reflects the importance of a particular page as a function of how many actual visits it receives by real users is called the *intentional surfer model*. The Google toolbar sends information to Google for every page visited, and thereby provides a basis for computing PageRank based on the intentional surfer model. The introduction of the nofollow attribute by Google to combat Spamdexing has the side effect that webmasters commonly use it on outgoing links to increase their own PageRank. This causes a loss of actual links for the Web crawlers to follow, thereby making the original PageRank algorithm based on the random surfer model potentially unreliable. Using information about users' browsing habits provided by the Google toolbar partly compensates for the loss of information caused by the nofollow attribute. The SERP rank of a page, which determines a page's actual placement in the search results, is based on a combination of the random surfer model (PageRank) and the intentional surfer model (browsing habits) in addition to other factors.

Other uses

Personalized PageRank is used by Twitter to present users with other accounts they may wish to follow.

A version of PageRank has recently been proposed as a replacement for the traditional Institute for Scientific Information (ISI) impact factor, and implemented at Eigenfactor as well as at SCImago. Instead of merely counting total citation to a journal, the "importance" of each citation is determined in a PageRank fashion.

A similar new use of PageRank is to rank academic doctoral programs based on their records of placing their graduates in faculty positions. In PageRank terms, academic departments link to each other by hiring their faculty from each other (and from themselves).

PageRank has been used to rank spaces or streets to predict how many people (pedestrians or vehicles) come to the individual spaces or streets. In lexical semantics it has been used to perform Word Sense Disambiguation and to automatically rank WordNet synsets according to how strongly they possess a given semantic property, such as positivity or negativity.

A Web crawler may use PageRank as one of a number of importance metrics it uses to determine which URL to visit during a crawl of the web. One of the early working papers [44] that were used in the creation of Google is *Efficient crawling through URL ordering*,[45] which discusses the use of a number of different importance metrics to determine how deeply, and how much of a site Google will crawl. PageRank is presented as one of a number of these importance metrics, though there are others listed such as the number of inbound and outbound links for a URL, and the distance from the root directory on a site to the URL.

The PageRank may also be used as a methodology to measure the apparent impact of a community like the Blogosphere on the overall Web itself. This approach uses therefore the PageRank to measure the distribution of attention in reflection of the Scale-free network paradigm.

In any ecosystem, a modified version of PageRank may be used to determine species that are essential to the continuing health of the environment.

Chapter-8

Comparision with Some other Page Ranking Algorithms

8.1 HITS Algorithm

In the HITS algorithm, the first step is to retrieve the most relevant pages to the search query. This set is called the *root set* and can be obtained by taking the top n pages returned by a text-based search algorithm. A *base set* is generated by augmenting the root set with all the web pages that are linked from it and some of the pages that link to it. The web pages in the base set and all hyperlinks among those pages form a focused subgraph. The HITS computation is performed only on this *focused subgraph*. According to Kleinberg the reason for constructing a base set is to ensure that most (or many) of the strongest authorities are included.

Authority and hub values are defined in terms of one another in a mutual recursion. An authority value is computed as the sum of the scaled hub values that point to that page. A hub value is the sum of the scaled authority values of the pages it points to. Some implementations also consider the relevance of the linked pages.

The algorithm performs a series of iterations, each consisting of two basic steps:

- **Authority Update**: Update each node's *Authority score* to be equal to the sum of the *Hub Scores* of each node that points to it. That is, a node is given a high authority score by being linked to pages that are recognized as Hubs for information.
- **Hub Update**: Update each node's *Hub Score* to be equal to the sum of the *Authority Scores* of each node that it points to. That is, a node is given a high hub score by linking to nodes that are considered to be authorities on the subject.

The Hub score and Authority score for a node is calculated with the following algorithm:

- Start with each node having a hub score and authority score of 1.
- Run the Authority Update Rule
- Run the Hub Update Rule
- Normalize the values by dividing each Hub score by square root of the sum of the squares of all Hub scores, and dividing each Authority score by square root of the sum of the squares of all Authority scores.
- Repeat from the second step as necessary.

HITS, like Page and Brin's PageRank, is an iterative algorithm based on the linkage of the documents on the web. However it does have some major differences:

- It is query dependent, that is, the (Hubs and Authority) scores resulting from the link analysis are influenced by the search terms;
- As a corollary, it is executed at query time, not at indexing time, with the associated hit on performance that accompanies query-time processing.
- It is not commonly used by search engines. (Though a similar algorithm was said to be used by Teoma, which was acquired by Ask Jeeves/Ask.com.)
- It computes two scores per document, hub and authority, as opposed to a single score;
- It is processed on a small subset of 'relevant' documents (a 'focused subgraph' or base set), not all documents as was the case with PageRank.

8.2 Tag Rank Algorithm

A novel algorithm named as TagRank for ranking theweb page based on social annotations is proposed by Shen Jie,Chen Chen,Zhang Hui,Sun Rong-Shuang,Zhu Yan andHe Kun. This algorithm calculates the heat of the tags by using time factor of the new data source tag and the annotations behavior of the web users. This algorithm provides a better authentication method for ranking the webpages. The results of this algorithm are very accurate and this algorithm index new information resources in a better way. Future work in this direction can be to utilize conccurrence factor of the tag to determine weight of the tag and this algorithm can also be improved by using semantic relationship among the co-occurrence tags.

8.3 EigenRumor Algorithm

As the number of blogging sites is increasing day by day,there is a challenge for service provider to provide good blogs to the users. Page rank and HITS are very promising in providing the rank value to the blogs but some limitations arise, if these two algorithms are applied directly to the blogs The rank scores of blog entries as decided by the page rank algorithm is often very low so it cannot allow blog entries to be provided by rank score according to their importance. To resolve these limitations, a EigenRumor algorithm is proposed for ranking the blogs. This algorithm provides a rank score to every blog by weighting the scores of the hub and authority of the bloggers depending on the calculation of eigen vector.

8.4 Distance Rank Algorithm

An intelligent ranking algorithm named as distance rank is proposed by Ali Mohammad Zareh Bidoki and Nasser Yazdani . It is based on reinforcement learning algorithm. In this algorithm,

the distance between pages is considered as a punishment factor. In this algorithm the ranking is done on the basis of the shortest logarithmic distance between two pages and ranked according to them.The Advantage of this algorithm is that it can find pages with high quality and more quickly with the use of distance based solution. The Limitation of this algorithm is that the crawler should perform a large calculation to calculate thedistance vector, if new page is inserted between the two pages.

8.5 Time Rank Algorithm

An algorithm named as TimeRank, for improving the rank score by using the visit time of the web page is proposed by H Jiang et al. Authors have measured the visit time of the page after applying original and improved methods of web page rank algorithm to know about the degree of importance to the users. This algorithm utilizes the time factor to increase the accuracy of the web page ranking. Due to the methodology used in this algorithm, it can be assumed to be a combination of content and link structure. The results of this algorithm are very satisfactory and in agreement with the applied theory for developing the algorithm.

Chapter-9

Applications Of Page Rank Algorithm

Estimating Web Traffic

Because PageRank roughly corresponds to a random web surfer (see Section 2.5), it is interesting to see how PageRank corresponds to actual usage. We used the counts of web page accesses fromNLANR [NLA] proxy cache and compared these to PageRank. The NLANR data was from several national proxy caches over the period of several months and consisted of 11,817,665 unique URLs with the highest hit count going to Altavista with 638,657 hits. There were 2.6 million pages in the intersection of the cache data and our 75 million URL database. It is extremely difficult to compare these datasets analytically for a number of different reasons. Many of the URLs in the cache accessdata are people reading their personal mail on free email services. Duplicate server names and page names are a serious problem. Incompleteness and bias a problem is both the PageRank data and the usage data. However, we did see some interesting trends in the data. There seems to be a high usage of pornographic sites in the cache data, but these sites generally had low PageRanks. We believe this is because people do not want to link to pornographic sites from their own web pages. Using this technique of looking for differences between PageRank and usage, it may be possible to fnd things that people like to look at, but do not want to mention on their web pages. There are some sites that have a very high usage, but low PageRank such as netscape.yahoo.com. We believe there is probably an important backlink which simply is omitted from our database (we only have a partial link structure of the web). It may be possible to use usage data as a start vector for PageRank, and then iterate PageRank a few times. This might allowing in holes in the usagedata. In any case, these types of comparisons are an interesting topic for future study.

PageRank as Backlink Predictor

One justification for PageRank is that it is a predictor for backlinks. In [CGMP98] we explore the issue of how to crawl the web efficiently, trying to crawl better documents first. We found on tests of the Stanford web that PageRank is a better predictor of future citation counts than citation counts themselves.

The experiment assumes that the system starts out with only a single URL and no otherinformation, and the goal is to try to crawl the pages in as close to the optimal order as possible.The optimal order is to crawl pages in exactly the order of their rank according to an evaluationfunction. For the purposes here, the evaluation function is simply the number of citations, givencomplete information. The catch is that all the information to calculate the evaluation function is not available until after all the documents have been crawled. It turns out using the incomplete data, PageRank is a more effective way to order the crawling than the

number of known citations.In other words, PageRank is a better predictor than citation counting even when the measure is the number of citations! The explanation for this seems to be that PageRank avoids the local maxima that citation counting gets stuck in. For example, citation counting tends to get stuck in local collections like the Stanford CS web pages, taking a long time to branch out and highly cited pages in other areas. PageRank quickly finds the Stanford homepage is important, and gives preference to its children resulting in an efficient, broad search.This ability of PageRank to predict citation counts is a powerful justification for using PageRank. Since it is very difficult to map the citation structure of the web completely, PageRank may even be a better citation count approximation than citation counts themselves.

User Navigation: The PageRank Proxy

We have developed a web proxy application that annotates each link that a user sees with itsPageRank. This is quite useful, because users receive some information about the link before they click on it. In Figure 9 is a screen shot from the proxy. The length of the red bars is the log of the URL's PageRank. We can see that major organizations, like Stanford University, receive a very high ranking followed by research groups, and then people, with professors at the high end of the people scale. Also notice ACM has a very high PageRank, but not as high as Stanford University.

Interestingly, this PageRank annotated view of the page makes an incorrect URL for one of theprofessors glaringly obvious since the professor has a embarrassingly low PageRank. Consequently this tool seems useful for authoring pages as well as navigation. This proxy is very helpful for looking at the results from other search engines, and pages with large numbers of links such as Yahoo's listings. The proxy can help users decide which links in a long listing are more likely to be interesting. Or, if the user has some idea where the link they are looking for should fall in the importance" spectrum, they should be able to scan for it much more quickly using the proxy.

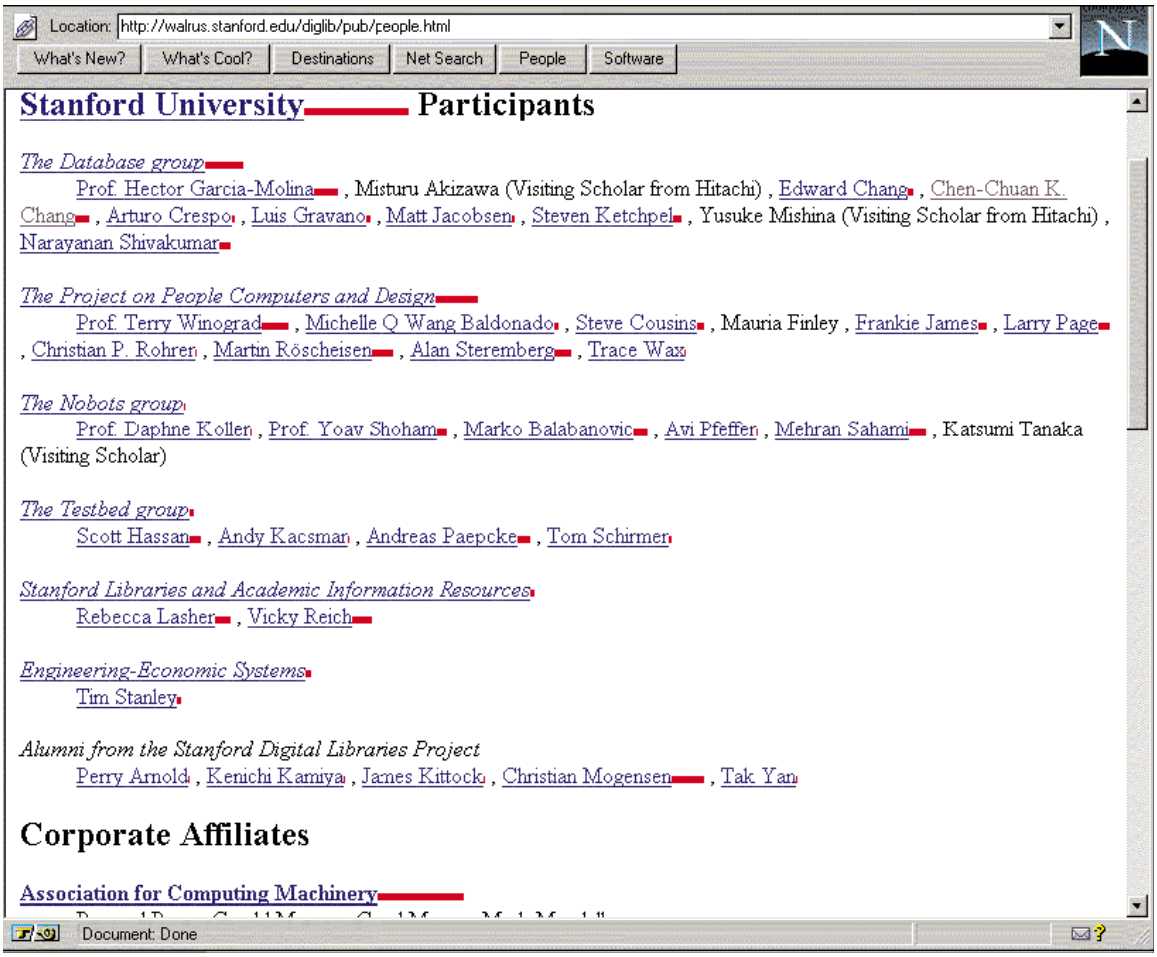

Fig.9

Page Rank Proxy

Chapter-10

Conclusion

We have taken on the audacious task of condensing every page on the World WideWeb into a single number, its PageRank. PageRank is a global ranking of all web pages, regardlessof their content, based solely on their location in the Web's graph structure.

Using PageRank, we are able to order search results so that more important and central Webpages are given preference. In experiments, this turns out to provide higher quality search results tousers. Th intuition behind PageRank is that it uses information which is exernal to the Web pagesthemselves - their backlinks, which provide a kind of peer review. Furthermore, backlinks from important" pages are more significant than backlinks from average pages. This is encompassed in the recursive definition of PageRank.

PageRank could be used to separate out a small set of commonly used documents which cananswer most queries. The full database only needs to be consulted when the small database is notadequate to answer a query. Finally, PageRank may be a good way to help find representative pages to display for a cluster center.

We have found a number of applications for PageRank in addition to search which include trafficestimation, and user navigation. Also, we can generate personalized PageRanks which can create a view of Web from a particular perspective.

Overall, our experiments with PageRank suggest that the structure of the Web graph is veryuseful for a variety of information retrieval tasks.

REFERENCES:

1.[BP] Sergey Brin and Larry Page. Google search engine. http://google.stanford.edu.

2.[CGMP98] Junghoo Cho, Hector Garcia-Molina, and Lawrence Page. Efficient crawling through url ordering. In To Appear: Proceedings of the Seventh International Web Conference(WWW 98), 1998.

3.[Gar95] Eugene Gareld. New international professional society signals the maturing of scientometrics and informetrics. The Scientist, 9(16), Aug 1995. http://www.the-scientist.library.upenn.edu/yr1995/august/issi_950821.html.

4.[Gof71] William Gorman. A mathematical method for analyzing the growth of a scientific discipline. Journal of the ACM, 18(2):173{185, April 1971.

5.[Kle98] Jon Kleinberg. Authoritative sources in a hyperlinked environment. In Proceedings of the Nineth Annual ACM-SIAM Symposium on DiscreteAlgorithms1998.