Google Page Rank Algorithm

Computational Physics (P346) Project

Ashlin V Thomas

Instructor : Dr. Colin Benjamin

School of Physical Sciences
National Institute of Science Education and Research

The Problem

- The internet contains billions of pages, making it challenging to determine the most relevant results for a search query.
- Traditional ranking methods often rely solely on keyword matching, which can lead to suboptimal results.

What happens when you do a google search?

- **Query module** : Natural language → Machine language
- **Ranking module** : Ranks relevant pages.

Google Page Rank Algorithm

- Developed by Sergey Brin and Larry Page.
- It ranks web pages based on link structure.
- It does so by evaluating the quantity and quality of links to a page.

Graphs

Graph: A graph G is a pair of sets $G = (V, E)$, where -

- Elements of the set V are called vertices and
- E is a set of unordered pairs of distinct vertices, called, edges.

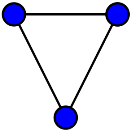


Figure 1: An example of a graph with three vertices and three edges.

Directed graph

- $G = (V, E)$ is said to be a directed graph if all pairs of vertices in E are ordered.
- For such graphs, the edges are called directed edges.

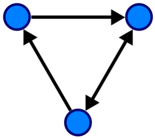


Figure 2: A directed graph with three vertices and four directed edges.

Internet : A directed graph

Internet can be thought of as a directed graph, where -

- Vertices $\rightarrow$ Webpages
- Edges $\rightarrow$ Hyperlinks between pages

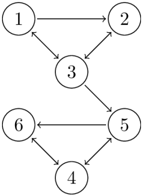


Figure 3: Internet as a directed graph.

Probability Vector

- **Probability Vector:** A probability vector is a column vector whose entries are non-negative and sum to one.
- It can be represented as $\mathbf{p} = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{pmatrix}$ where $p_i \geq 0$ and $\sum_{i=1}^{n} p_i = 1$.

Stochastic Matrix

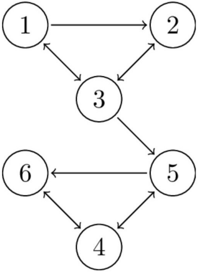
- **Stochastic Matrix:** A square matrix is said to be stochastic if each column of the matrix is a probability vector.
- It can be represented as $S = \begin{pmatrix} s_{11} & s_{12} & \cdots \\ s_{21} & s_{22} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$ where $s_{ij} \geq 0$ and $\sum_i s_{ij} = 1$ for each column j.

Hyperlink matrix

- $|P_i|$ is defined as the number of outlinks from the webpage P_i.
- Hyperlink matrix can be defined as -

$$H_{ij} = \begin{cases} \frac{1}{|P_j|} & \text{if there is a link from } P_j \text{ to } P_i \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

An Example



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

Figure 4: A network of webpages and corresponding hyperlink matrix.

PageRank vector

- Page rank of P_i can be recursively defined as -

$$r(P_i) = \sum_{Q \in \mathcal{B}_i} \frac{r(Q)}{|Q|} \tag{2}$$

where $\mathcal{B}_i$ is the set of inlink pages to P_i.

- Now all the page ranks can be written together as a vector, which we call $\mathbf{x}$.
- By definition, PageRank vector would be a probability vector.

Iterative scheme

1. Initialize all the page ranks to be equal.

$$\mathbf{x_0} = \frac{1}{N}\mathbb{1} \qquad \text{where, } \mathbb{1} \text{ is a vector with all entries 1.}$$

2. Iteratively update $\mathbf{x}$ using the equation -

$$\mathbf{x_{n+1}} = H\mathbf{x_n}$$

   where, H is the hyperlink matrix. One can view the iterations as a person visiting a page and then following a link at random.

3. Continue the iterations still a steady state is reached, which gives the page ranks.

Drawbacks

1. The question of CONVERGENCE!!!
2. - On an average, a google search requires hundreds of thousands of pages to be ranked, i,e, N would be of that order.

Number of iterations	Operation count
1	$2N^2$
m	$2mN^2$

   - This makes the iterative scheme very time consuming.

Foundational lemmas

Lemma 1

The hyperlink matrix corresponding to a a network of web-pages, where each page has at least one outlink, is stochastic.

Lemma 2 (Perron Frobenius Theorem)

If A is a stochastic $n \times n$ matrix, then:

- A will have n linearly independent eigenvectors.
- The largest eigenvalue of a stochastic matrix will be $\lambda_1 = 1$ with geometric multiplicity 1.
- The smallest eigenvalue will always be nonnegative: $0 \leq |\lambda_n| < 1$.

Foundational lemmas (continued)

Lemma 3

Let A be an $n \times n$ matrix with n linearly independent eigenvectors $v_1, v_2, \ldots, v_n$ and associated eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. Then for any initial vector $x \in \mathbb{R}^n$, we can express $A^k x$ as:

$$A^k x = c_1 \lambda_1^k v_1 + c_2 \lambda_2^k v_2 + c_3 \lambda_3^k v_3 + \cdots + c_n \lambda_n^k v_n$$

where $c_1, c_2, \ldots, c_n$ are constants found by expressing x as a linear combination of the eigenvectors.

Note: We can assume the eigenvalues are ordered such that $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|$.

Main Theorem

Theorem

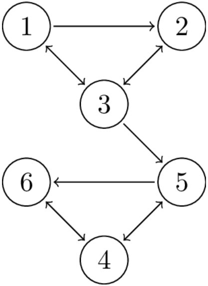
If the hyperlink matrix H of certain network of pages is stochastic with $v = (v_1, v_2, \cdots, v_n)^T$ being its dominant right eigenvector. Then the iterative scheme, previously defined, converges to -

$$\lim_{k\to\infty} H^k x_0 = \left(\frac{1}{\sum_{i=1}^{n} v_i}\right) v \tag{3}$$

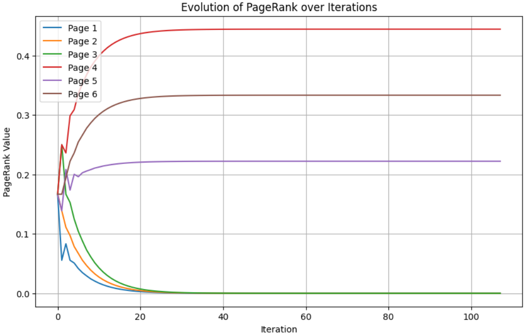
Python code

```
import numpy as np
import matplotlib.pyplot as plt

def pagerank_iterative(H, max_iter=1000, tol=1e-9):
    N = H.shape[0]
    # Initialize PageRank vector
    x = np.ones(N) / N
    ranks_history = [x.copy()]

    for k in range(max_iter):
        x_new = H @ x  # Update using the hyperlink matrix
        ranks_history.append(x_new.copy())

        # Check for convergence
        if np.linalg.norm(x_new - x, 1) < tol:
            break
        x = x_new

    return x, ranks_history
```

Results



(a) Network of pages



(b) Variation of page rank over iterations

Figure 5: Evolution of page ranks over iterations for the given network.

We find that for the shown network, the page ranks converge to - [5.9697e-10, 9.539e-10, 1.4976e-09, 4.4444e-01, 2.2222e-01, 3.3333e-01].
This is in agreement with the theorem, as the dominant eigenvector of the hyperlink matrix is -

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{4}{9} \\ \frac{2}{9} \\ \frac{1}{3} \end{pmatrix} \approx \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0.444 \\ 0.222 \\ 0.333 \end{pmatrix}$$

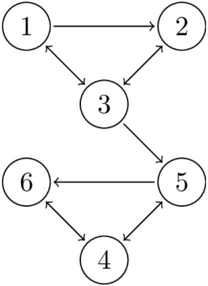
The Algorithm

- It models a person randomly clicking on links in a web page.
- Add hypothetical links between pages, such that the person can jump to any page from any page.
- Probability associated with a hypothetical link is half the probability of following a real link.

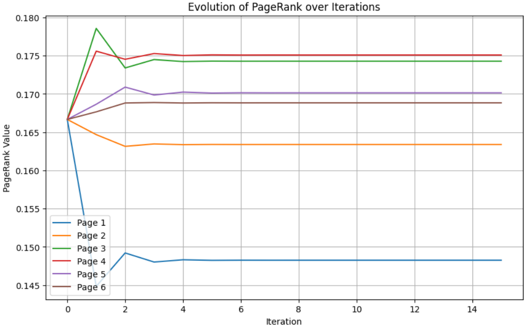
Implementation

```
def positive_entry_pos(inlist):
  pos = []
  for i in range(len(inlist)):
    if inlist[i]>0:
      pos.append(i)
  return pos

def random_surfer(H, max_iter=1000, tol=1e-9):
  N = H.shape[0]
  random_surfer_H = []
  for i in H.T:
    pos = positive_entry_pos(i)
    k = N + len(pos) - 1
    random_surfer_H.append([2/k if j in pos else 1/k for j in range(N)])
  random_surfer_H = np.array(random_surfer_H).T
  for i in range(N):
    random_surfer_H[i][i] =0

  return pagerank_iterative(random_surfer_H,max_iter, tol)
```

Results



(a) Network of pages



(b) Variation of page rank over iterations

Figure 6: Evolution of page ranks over iterations for the given network using random surfer algorithm.

Summary

- PageRank algorithm is a powerful tool to rank web pages based on link structure.
- The algorithm is based on the concept of stochastic matrices and their eigensystem.
- The algorithm can be implemented using an iterative scheme.
- The random surfer algorithm is a modification of the PageRank algorithm, which models a person randomly clicking on links.

References

[1] Sergey Brin and Lawrence Page.
The anatomy of a large-scale hypertextual web search engine.
Computer Networks and ISDN Systems, 30(1-7):107–117, 1998.

[2] Amy N Langville and Carl D Meyer.
A survey of eigenvector methods for web information retrieval.
SIAM review, 47(1):135–161, 2005.

[3] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd.
The pagerank citation ranking: Bringing order to the web.
Stanford InfoLab, 1999.

[4] Michael Sullivan.
The google page rank algorithm, 2023.
Accessed: 2023-10-01.

Thank you for your attention!!!

Any questions?