PageRank Algorithm: Recommending Top Webpages for Users

Students: Changyuan Qiu, Haoxuan Shan, Ruiyi Wang Supervisor: Professor Martin Strauss

MICHIGAN ENGINEERING, UNIVERSITY OF MICHIGAN
LSA MATHEMATICS, UNIVERSITY OF MICHIGAN

MATH 371, F20
Numerical Methods
Term Project

Introduction

The well-known PageRank Algorithm[1] provides a way to estimate the importance of webpages by evaluating the number of links between the individuals in the network.

This project aims at applying the PageRank Algorithm to develop a program to generate a pool of recommended relevant webpages from a gathering of webpages given user's keyword. First, our program will automatically collect the webpages that the base page links to within a certain depth to form a large pool of pages. Then the program will calculate the ranking of each page within the pool by the modified PageRank algorithm, sort the links according to their rankings and offer a recommendation.

Mathematical Model

The webpages can be abstracted to nodes and the connections between the nodes can be presented as a directed weighted graph. For a node j, we denote $L(j)$ as the number of nodes j is pointing towards. Then we construct an $N \times N$ matrix $\overline{A}$ from the graph, where each entry satisfies:

$$\bar{a}_{ij} = \begin{cases} \frac{1}{L(j)} & \text{if there is a link from webpage } j \text{ to } i \\ \frac{1}{N} & \text{if webpage } j \text{ is a dangling webpage} \\ 0 & \text{otherwise} \end{cases}$$

To handle the problem of spider traps where no links from within a group of pages point to the outside of the group we add a damping factor d to our final matrix M:

$$M = d\overline{A} + \frac{1-d}{N}\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 1 & 1 & \dots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \dots & 1 \end{bmatrix}$$

To represent the ranking of each node in the graph, we introduce a $N \times 1$ vector $x = [x_A, x_B, ...]$ which is initialized to be $[1/N, 1/N, \dots, 1/N]$. The simplified PageRank algorithm repeatedly perform linear transformation M on x until the vector x converges.

Procedure

- **Collecting webpages from Wikipedia**

  We collected webpages for the keyword "Linear Algebra" under the Wikipedia website into a page pool using the *wikiCrawler*[1] function. The links between the pages and their subpages were gathered and categorized into normal links and dangling links. To reduce the number of irrelevant pages, we dig into webpages of *depth* at most 4. (Here the *depth* is defined as the maximum length of the path starting from the source page).

- **Constructing the Iteration Matrix**

  Then we construct the iteration matrix M according to categories of pages and links between them as in the mathematical model illustrated before. After several testings we find the optimal damping factor $d = 0.85$ and generate the final iteration matrix M using the pageRank function.

- **Performing the PageRank algorithm and visualizing the results**

  We repeatedly iterated the ranking vector x by taking $x = Mx$ until it converges using the *pageRank*[2] function. Convergence is measured by comparing the Frobenius norm of the difference of the two vectors $||x' - x||_{\mathbb{F}}$ to a small constant ϵ and when $||x' - x||_{\mathbb{F}} < \epsilon$ we state that the convergence is reached. Here we set $\epsilon = 10^{-3}$ and after repeated iterations, we obtain the final page ranking vector x.

Results

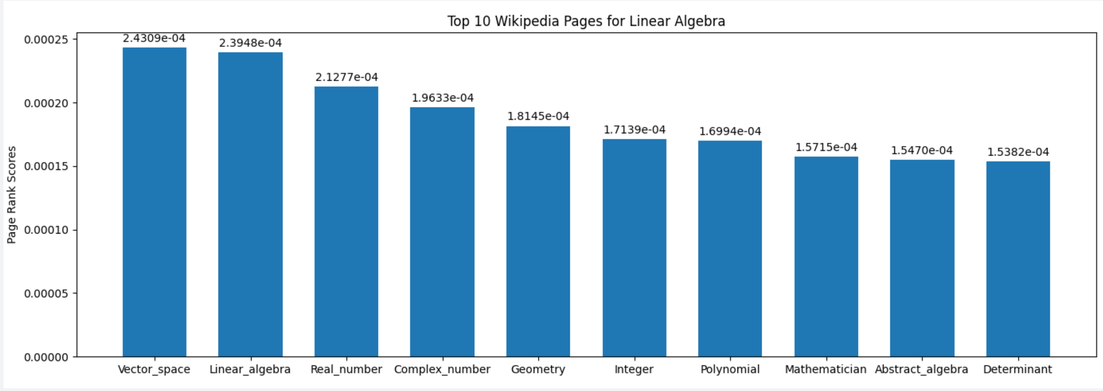

Figure 1. Bar chart of the top 10 wiki pages for "Linear Algebra" w.r.t. the PageRank scores

After obtaining the final page rank vector x, we sorted it and plotted the above bar chart, which shows the "Pages Rank Scores" of the "Top 10 Wikipedia Pages" for our keyword "Linear Algebra".

The wiki page with the highest "Page Rank Score" is the page "Vector_space", which has a very high score around 2.4309×10^{-4}. And by looking at the page names we obtained, we can verify that the topics are closely related to Linear Algebra, which meets our expectation.

Algorithms

[1]**wikiCrawler**: Collecting wiki pages from the source wiki website with specified depth.

Algorithm 1 The WikiCrawler algorithm

Input: $\begin{cases} url\text{: url of the source wiki page} \\ depth\text{: depth of pages to crawl} \end{cases}$

Output: Print link relations of all pages from the source page within the given depth

function WIKICRAWLER($url, depth$)
  if $depth = 0$ **then**
    return
  Crawl the information from the source page url
  Match all url' with form '$/wiki/ < keyword >$'
  for all url' matched **do**
    Print $depth$ number of "-" to mark current $depth$
    Print $< keyword >$
    Recursively crawl url' it by calling $wikiCrawler(url', depth - 1)$
  return

[2]**pageRank**: Constructing the iteration matrix from the links, performing the PageRank iteration on the vector until it converges and visualizing the result.

Algorithm 2 The PageRank algorithm

Input: $\begin{cases} L\text{: List of length } N\text{, representing links from each website} \\ d\text{: the damping factor} \\ \epsilon\text{: the convergence error bound} \end{cases}$

Output: The $N \times 1$ PageRank vector x

function PAGERANK(L, d, ϵ)
  Construct the $N \times N$ matrix $\overline{A}$ from L
  $M \leftarrow d\overline{A} + \frac{1-d}{N} I_N$
  Initialize the $N \times 1$ rank vector $x = [1/N, 1/N, ..., 1/N]$, $x' \leftarrow x$
  while $||x' - x||_{\mathbb{F}} \geq \epsilon$ **do**
    $x \leftarrow x'$
    $x' \leftarrow Mx'$
  return x

References

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

[2] Haveliwala, Taher. Efficient computation of PageRank. Stanford, 1999.

[3] Bianchini, Monica, Marco Gori, and Franco Scarselli. "Inside pagerank." *ACM Transactions on Internet Technology (TOIT)* 5, no. 1 (2005): 92-128.