Search Engine based on Page Rank Algorithm

Aditya Menon, G Arshad, Paras Dahiya, and Prakkash Manohar
Indian Institute of Information Technology, Sri City, India
{adityaharidas.m,arshad.g,paras.d,prakkash.m}16@iiits.in

ABSTRACT

In today's era of information overload, search engines play a significant role catering to the information needs of millions of users. Under the hood, all these search engines have different complex algorithms and techniques to retrieve relevant results from the huge corpus of documents within a few seconds. Hence, basic knowledge and working of a search engine is a necessity. This project implements a search engine which will retrieve the documents for a given search query and order them based on their relevance score. Before calculating the relevance score, all the dangling links are removed from the corpus. Then, the relevance score is calculated as a weighted sum of different criteria, one of which is the page rank of a document (calculated as per original page rank algorithm). So, different criteria are given different weights and then all these weighted criteria are summed up to yield a relevance score which is then used to indicate the relevance of a document. Based on this relevance score, top 10 most relevant documents are returned as the results to the given search query.

Keywords

Search Engine, Page Rank Algorithm, Dangling Links, Relevance Score

1 INTRODUCTION

The data set is the corpus of 100 Wikipedia pages and is obtained by crawling Wikipedia breadth wise, starting with the Wikipedia page of "Computer Programming". So, all the 100 documents (pages) are related to computing. The following attributes are associated with the corpus:

1. Each document with a url mapped to a unique url id.
2. Each word mapped to a unique word id.
3. Each word id mapped to a url id and a position id, which indicates the position of that word in each document.
4. Each word id mapped to a url id, indicating if the word is present in the url or not.
5. Each url id (document 1) mapped to another url id (document 2), indicating that there is a link going from document 1 (with the first url) to the document 2 (with the second url).

All the following relations are stored in a MySQL database (in different tables). These relational tables serve as the index for the search engine. For a given query, this index is used for finding the different attributes like position of all the query words in the different documents, frequency of the query words in each document, etc. Using this index, the following criteria (attributes related to the query words) are found out and used for the calculation of the relevance score:

1. Term frequency for a query word
2. Position of a query word in each document
3. Distance between successive occurrences of a query word in a document
4. Total number of incoming links to a document
5. Presence of a query word in url

Now, page rank is found out for all documents (as per the original page rank algorithm). After this, we assign different weights to these 5 criteria and the calculated page rank. Then, we sum up all the weighted criteria to calculate the relevance score for each document. Based on this relevance score, first 10 documents with the highest relevance scores are returned as the search results.

2 METHODOLOGY

We start by removing all the dangling links in the corpus. Dangling links are simply links that point to any document with no outgoing links. Because dangling links do not affect the ranking of any other page directly, we simply remove them from the system. We keep removing all dangling links iteratively until we have no leaf node (in a graph showing all documents and the associated links).

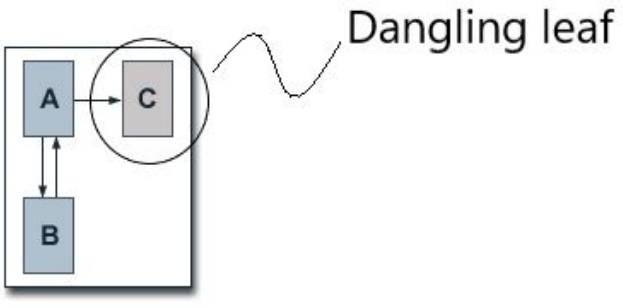

Figure 1: C is a dangling leaf node with no outbound links

After this, calculating relevance score for each document is the main aspect. We calculate term frequency (for all the query words) i.e. total number of hits (encountering a query word) in each document. We know that query word frequency in a document is directly proportional to its relevance score so we give

this criterion a high weight in relevance score calculation. Then, we normalize it (by dividing by the maximum value of total hits). Let us call this value ***hf***.

Now, we calculate word position (of first occurrence) of a query word i.e. the first hit in each document. We can get an intuition that documents with query words appearing in the initial part of a document will be more relevant. So, lesser the word position, more is the relevance. We normalize this word position (by dividing the minimum value of first hit by the current value). Let us call this value ***fhp***.

Next, we calculate the sum of difference of successive hits. We normalize it (by dividing by the maximum value). Let us call this value ***sdsh***.

Now, we calculate the total number of links to the document under consideration (incoming links). Let us call this value ***il***.

Next, for each document, we check if a query word appears in the document or not. Intuitively, if it appears in the document url, the document will be highly relevant. So, we calculate **qw** as follows:

$$qw = \sum_j a_j$$

$a_j = 1$, if the query word j appears in the url

$a_j = 0$, if the query word j does not appear in the url

Now, we normalize **qw**.

Now, we have to calculate page ranks for all the documents. Page Rank (PR) is an algorithm used by Google Search to rank web pages in their search engine results. Page Rank is a way of measuring the importance of website pages. According to Google, Page Rank works by counting the number and quality of links to a page to determine a rough estimate of how important the website is. The underlying assumption is that more important websites are likely to receive more links from other websites.

We can see that in Figure 2, Page C has a higher Page Rank than Page E, even though there are fewer links to C; the one link to C comes from an important page and hence is of high value. If web surfers who start on a random page have an 85% likelihood of choosing a random link from the page they are currently visiting, and a 15% likelihood of jumping to a page chosen at random from the entire web, they will reach Page E 8.1% of the time. (The 15% likelihood of jumping to an arbitrary page corresponds to a damping factor of 85%.) Without damping, all web surfers would eventually end up on Pages A, B, or C, and all other pages would have PageRank zero. In the presence of damping, Page A effectively links to all pages in the web, even though it has no outgoing links of its own.

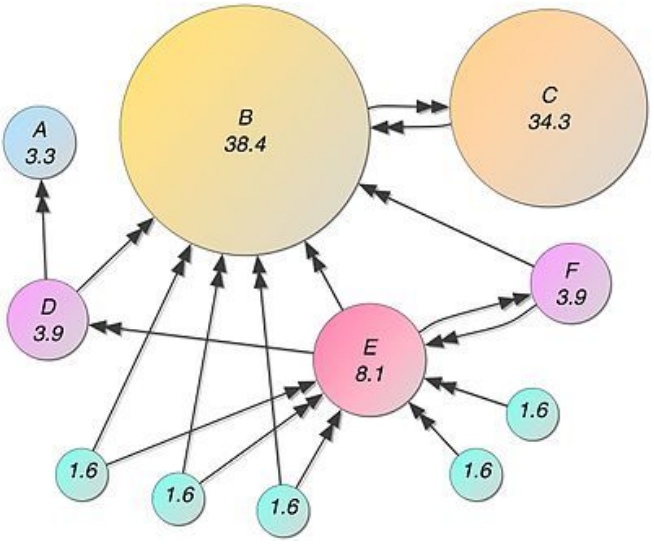

Figure 2: Mathematical Page Ranks for a simple network, expressed as percentages.

Now, we apply this Page Rank Algorithm for all the documents to get the individual page ranks as follows:

$$PR(d) = 0.15 + 0.85 \sum_i \frac{PR(i)}{ol(i)}$$

i is a document such that there is an outgoing link from i to d.

ol(i) = total number of outgoing links for document i

Now, we calculate the relevance score by the following formula:

$$RS(d) = 5 * (hf) + fhp + sdsh +$$

$$0.2 * (il) + 0.5 * (PR(d)) + 20 * (qw)$$

where,

hf = (query word) hit frequency in the document

fhp = first (query word) hit position in the document

sdsh = sum of difference of successive hit positions in the document

il = total number of incoming links for the given document

PR(d) = page rank for the given document

qw represents presence of query words in the given document url (mentioned above).

Once the relevance score for each document is calculated, top 10 documents with the highest relevance scores are returned as the query results.

3 CONCLUSION

We built a search engine which is based on modified Page Rank algorithm. We modified the original Page Rank Algorithm by introducing removal of Dangling Links from the corpus and then combining different criteria apart from the page ranks with weights and then summing them to get a relevance score. This captures much more useful information than just the page rank of the document (considers only the number

of outgoing links for each neighbor of the document [such that neighbor has an outgoing link to the given document]).

4 REFERENCES

[1] https://en.wikipedia.org/wiki/PageRank

[2]https://www.geeksforgeeks.org/page-rank-algorithm-implementation/

[3] https://en.wikipedia.org/wiki/Web_search_engine

[4]https://prchecker.net/what-are-the-effects-of-outbound-links-on-your-pagerank.html

[5] Programming Collective Intelligence by Toby Segaran