The Anatomy of a Large-Scale Hypertextual Web Search Engine: Summary

Raja Hasnain Anwar

This paper presents the prototype of Google, a large-scale and scalable search engine, and an *inspiration* for the researchers towards academic research on search engines.

1 Introduction

The web, in it's initial phase, had many challenges of data management and searching. The existing search engines relied on keywords matching which delivered poor results for user queries. The records were maintain manually which required time and effort, and lacked scalability. Google was designed to provide quality search results backed by *efficient record keeping*, *sustainable searching*, and *scalable storage management.*

2 System Features

Google works in two phases. First, it uses **crawlers** to gather information about the web pages and other documents. This information contains structure of a document, its **metadata**, and how it is linked with other documents. It then uses different algorithms to **index** this data structurally. In the second phase, it uses another set of algorithms to match user query with the indexed information about each document, and then delivers the results based on word *proximity*, its *presentation* in the document, and query *location.*

2.1 PageRank

Google uses the link structure of the web to ensure high quality search results. Google ranks pages based on their *approximate* quality which is calculated by the number of pages linking to a certain pages, and normalized by the number pages a certain page links to. This gives a numeric value on a page's quality called *PageRank.* PageRank is actually a probability distribution.

If a page **A** has pages $T_1...T_n$ linking to it, and $C(A)$ is the number of pages on it, then the PageRank of **A** can be calculated as:

$$PR(A) = (1-d) + d(\frac{PR(T_1)}{C(T_1)} + ... + \frac{PR(T_n)}{C(T_n)})$$

where d is a *hyper-parameter* between 0 and 1. Its optimal value given in original text is 0.85.

Google also considers the **anchor text** to rank the pages. This gives a relation among different pages which helps to provide more relevant search results.

3 Architecture

Google uses crawlers to download pages from the web using lists of URLs stored in **URLServer**. Crawlers run in *parallel* using their own list of URLs and DNS cache to speed up the process. Pages are first stored in **storesever** from where they are compressed and placed in a repository using their **docID**s. Google keeps a lexicon to match word occurrences and keep them in Hit Lists. A parser runs over all the stored documents and places words based their *hits* into **barrels** along with their **wordID**s. This *indexing* operation updates a link database storing all parsed link data; individual word data is used to generate an inverted index mapping words to documents those words come from. All this is implemented in C/C++ and runs on UNIX family OS. Crawlers and server are implemented in Python.
Data structures used in crawling, searching, and indexing are: *BigFiles*, *Repository*, *Document Index*, *Lexicon*, *Hit Lists*, *Forward Index*, and *Inverted Index.*

4 Indexing

The real Google magic happens in indexing the data into barrels. After the pages have been crawled, compressed, and parsed, Google uses its lexicon to give each word its wordID. Hit Lists are created which keep record or occurrences of wordIDs across documents with context of their font, style and position in the document. These lists take up most of the storage space. Further, by calculating the dot product of type-count and weight-count of hits, Google puts documents in Forward and Inverted Index which are used to calculate page ranks.

5 Searching

Against each search query, Google calculates the ranks of documents which match the *queried* terms. This gets complicated when a query is more than a word long. In that case, Google has to involve other factors like **proximity** for calculating ranks. Apart from that, searching process is a pre-set **8 step algorithm** which crawls through docIDs to match terms and calculates ranks according to that.

6 Conclusion

Google started as a mere research project but it had the potential to handle *real world* web data. It has evolved into an *internet giant* as of today. The key is to scale the system to cater for what is in the future. Sergey and Larry had tried to reach perfection, it worked out to be excellence.