Google - A case study

Overview:

"Google" derived from the word "googol" which refers to 10^100. Google is a web search engine and for a web search engine; few must have features are:

- Fast crawling technology

- Indexing must be efficient enough

- Searching & Query response time must be very faster.

Google caters to all the above features (major methods of a search engine). In addition, they have good back-end above good coding and data structure choice: they use distributed computing systems around the globe to ensure faster response times.

System Anatomy:



Understanding google architecture:

In Google, the web crawling (downloading of web pages) is done by several distributed crawlers, which is a computer program that browses the World Wide Web by employing many Computers. URLserver sends lists of URLs(uniform resource locator) to be fetched to the crawlers. The web pages that are fetched are then sent to the store server. The store server then compresses and stores the web pages into a repository. Every web page has an associated ID number called a docID which is assigned whenever a new URL is parsed out of a web page. The indexer and the sorter perform indexing functions and read the repository, uncompress the documents, and parse them. Each document is converted into a set of word occurrences called hits. The hits record the word, position in document, an approximation of font size, and capitalization. The indexer distributes these hits into a set of "barrels", creating a partially sorted forward index. The indexer performs another important function. It parses out all the links in every web page and stores important information about them in an anchors file. This file contains enough information to determine where each link points from and to, and the text of the link.

The URLresolver reads the anchors file and converts relative URLs into absolute URLs and in turn into docIDs. It puts the anchor text into the forward index, associated with the docID that the anchor points to. It also generates a database of links which are pairs of docIDs. The links database is used to compute PageRanks for all the documents. The sorter takes the barrels, which are sorted by docID and resorts them by wordID to generate the inverted index. This is done in place so that little temporary space is needed for this operation. The sorter also produces a list of wordIDs and offsets into the inverted index. A program called DumpLexicon takes this list together with the lexicon produced by the indexer and generates a new lexicon to be used by the searcher. The searcher is run by a web server and uses the lexicon built by DumpLexicon together with the inverted index and the Page Ranks to answer queries.

Most Appreciated Google’s System features:

* Page Rank Algorithm
  + PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))
* Anchor Text
* Other notable features:
  + It has location information for all hits, a set of all word occurrences so it makes extensive use of proximity or probability in searching.
  + Google keeps information about some visual presentation details such as font size of words (h1,h2,h3,b) – words in bolder/high font are weighted higher than other words.
  + Full raw HTML pages are available in the repository.
* Crawling : Spiders(bots) crawls complete web by visiting a link (and their links and their inner links) and so on! The bots continue to download all webpages (HTML) and keep them in the repository. Other tasks includes:
  + - Checking links
    - Validating html code
* Indexing: An index is prepared by google’s spider. When you query on the web then google simply look into their index of size equivalent to 10^7 GB’s (2017) and show you results based on their ranking algorithm.
* Ranking & Scoring: Relevant factors include:
  + Page Rank
  + Keyword Density, Presence of synonyms
  + Authority and trust of pages which refer to a page (backlinks)
  + Presence of Keyword in Meta information

All top Search engines and their market share:

Google: 81%

Yahoo: 12%

MSN: 3%

ASK: 1%