

The Anatomy of a Large-Scale Hypertextual Web Search Engine

Presented by: Nahi Ojeil

12/7/09

EECS 584, Fall 2009

1

Background

- Setting:
 - Year: 1997
 - Largest index: 100 million pages
 - Queries per day: 20 million
 - Precision: 1 out of 4 major search engines return themselves in top results

12/7/09

EECS 584, Fall 2009

2

More background

- Going back:
 - Year: 1994
 - Largest index: 110,000 pages
 - Queries per day: 1500

12/7/09

EECS 584, Fall 2009

3

Motivation

- The web is scaling! Up!
- Search engines need to follow that trend.
- Too much junk in query results.
- Precision at the expense of recall.
- Need for academic search engine research.

12/7/09

EECS 584, Fall 2009

4

System Features

- PageRank
- Anchor Text
- Other Features

12/7/09

EECS 584, Fall 2009

5

PageRank

- Definition:
 - An objective measure of a page's citation importance which corresponds well with people's subjective idea of importance.
- Properties:
 - Makes use of the link structure of the Web.
 - Improves search results.

12/7/09

EECS 584, Fall 2009

6

PageRank

■ Intuition:

- A "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back", but eventually gets bored and starts on another random page.
- The probability a random surfer visits a page is its PageRank.
- A damping factor is the probability a surfer gets bored.

12/7/09

EECS 584, Fall 2009

7

PageRank

■ Calculated using iterative algorithm.

$$PR(A) = (1 - d) + d(PR(T1)/C(T1) + PR(T2)/C(T2) + \dots + PR(Tn)/C(Tn))$$

■ Notation:

- PR(X): PageRank of page X.
- C(X): Number of links going out of X.
- T1...Tn: Pages that point to A.
- d: Damping factor (usually 0.85)

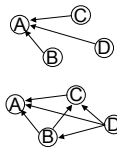
12/7/09

EECS 584, Fall 2009

8

PageRank

■ Examples:



$$PR(A) = PR(B) + PR(C) + PR(D).$$

$$PR(A) = \frac{PR(B)}{2} + \frac{PR(C)}{1} + \frac{PR(D)}{3}.$$

12/7/09

EECS 584, Fall 2009

9

Anchor Text

■ Anchor?

- Example:
 - `Yahoo!`
- Anchors provide more accurate description of web page being pointed to.
- Anchors exist for non-textual documents.

12/7/09

EECS 584, Fall 2009

10

Other Features

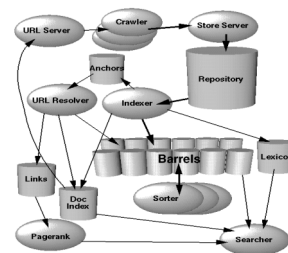
- It has location information for all hits.
- Google keeps track of some visual presentation details such as font size of words.
 - Words in a larger or bolder font are weighted higher than other words.
- Full raw HTML of pages is available in a repository

12/7/09

EECS 584, Fall 2009

11

Architecture Overview



12/7/09

EECS 584, Fall 2009

12

Major Data Structures

- **BigFiles:**
 - virtual files spanning multiple file systems
 - 64 bit addressable
- **Repository:**
 - contains full html for web pages
 - Compressed using zlib

12/7/09

EECS 584, Fall 2009

13

Major Data Structures

- **Document Index:**
 - keeps information about each document
 - Indexed using ISAM
- **Lexicon:**
 - in-memory mapping of word to WordID
 - stored as null delimited words and a hash table of pointers

12/7/09

EECS 584, Fall 2009

14

Major Data Structures

- **Hit Lists:**
 - a list of occurrences of a word in a particular document, including extra info
 - encoded using a hand optimized method
- **Forward Index:**
 - mapping from DocID to WordID
 - Partially sorted by WordID (due to barrels)

12/7/09

EECS 584, Fall 2009

15

Major Data Structures

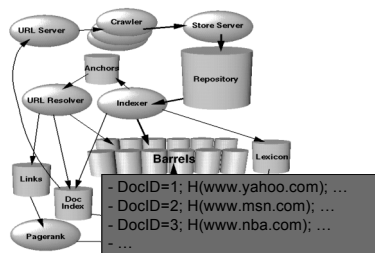
- **Inverted Index:**
 - the same barrels as forward index after being sorted
 - a mapping from WordID to DocID
 - two sets of barrels:
 - one for title and anchor hit lists
 - another for all hit lists

12/7/09

EECS 584, Fall 2009

16

Architecture Overview

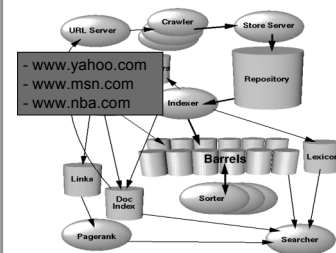


12/7/09

EECS 584, Fall 2009

17

Architecture Overview

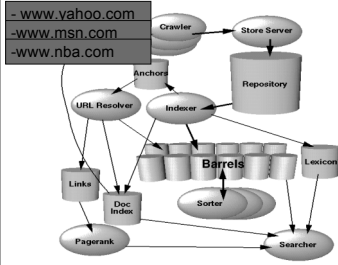


12/7/09

EECS 584, Fall 2009

18

Architecture Overview

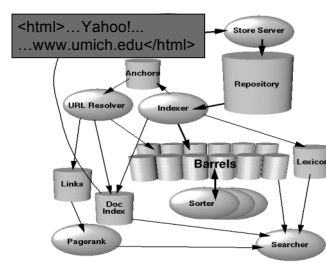


12/7/09

EECS 584, Fall 2009

19

Architecture Overview

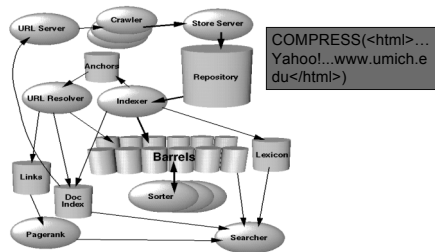


12/7/09

EECS 584, Fall 2009

20

Architecture Overview

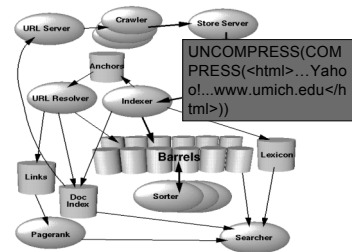


12/7/09

EECS 584, Fall 2009

21

Architecture Overview

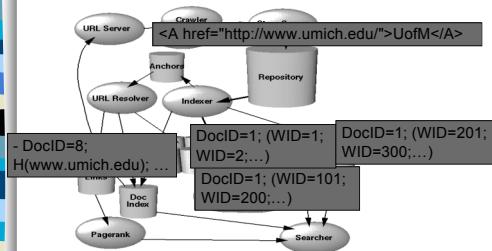


12/7/09

EECS 584, Fall 2009

22

Architecture Overview

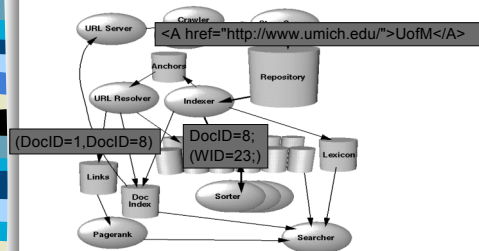


12/7/09

EECS 584, Fall 2009

23

Architecture Overview

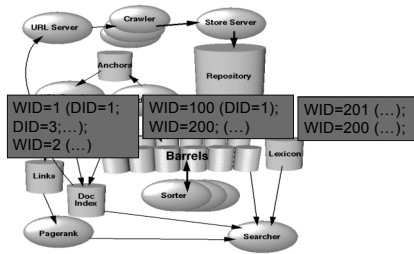


12/7/09

EECS 584, Fall 2009

24

Architecture Overview

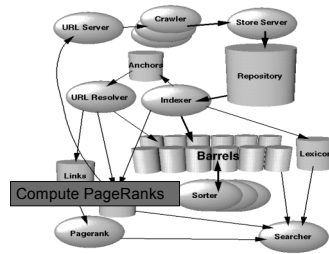


12/7/09

EECS 584, Fall 2009

25

Architecture Overview

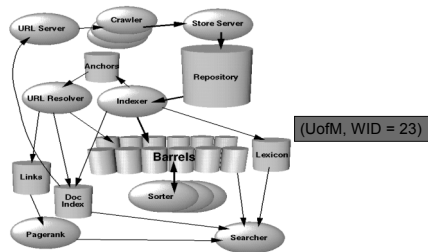


12/7/09

EECS 584, Fall 2009

26

Architecture Overview

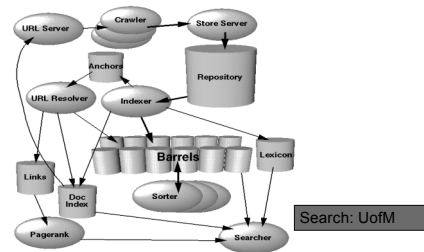


12/7/09

EECS 584, Fall 2009

27

Architecture Overview



12/7/09

EECS 584, Fall 2009

28

Crawling the Web

- Fast distributed crawling mechanism.
- URLserver provides crawlers the URLs.
- Implemented in Python.
- Have 300 open connections at a time.
- Keep a cache of DNS lookup.

12/7/09

EECS 584, Fall 2009

29

Indexing the Web

- Parsing:
 - custom written parser to handle errors
- Indexing:
 - word converted to WordID using Lexicon
 - occurrences translated into HitLists
 - shared main lexicon
 - centralized extra words lexicon

12/7/09

EECS 584, Fall 2009

30

Indexing the Web

- **Sorting:**
 - sorts barrels in place by WordID
 - happens one barrel at a time
- **Notes:**
 - crawlers will invariably cause problems
 - need significant resources to read email and answer phone calls

12/7/09

EECS 584, Fall 2009

31

Searching the Web

1. Parse the query
2. Convert words into wordIDs
3. Seek to the start of the doclist in the short barrel for every word
4. Scan through the doclists until there is a document that matches all the search terms
5. Compute the rank of that document for the query
6. If we are in the short barrels and at the end of any doclist, seek to the start of the doclist in the full barrel for every word and go to step 4.
7. If we are not at the end of any doclist go to step 4. Sort the documents that have matched by rank and return the top k.

Figure 4. Google Query Evaluation

12/7/09

EECS 584, Fall 2009

32

Results

- Query time: 1 to 10 seconds.
- Caching significantly improves results.

Query	Initial Query		Same Query Repeated (IO mostly cached)	
	CPU Time (s)	Total Time (s)	CPU Time (s)	Total Time (s)
al gore	0.09	2.13	0.06	0.06
vice president	1.77	3.84	1.66	1.80
hard disks	0.25	4.86	0.20	0.24
search engines	1.31	9.63	1.16	1.16

12/7/09

EECS 584, Fall 2009

33

Conclusion

- Google is designed to be a scalable search engine.
- The primary goal is to provide high quality search results over a rapidly growing World Wide Web.
- Google employs a number of techniques to improve search quality including PageRank, anchor text, and proximity information.
- Google is a complete architecture for gathering web pages, indexing them, and performing search queries over them.

12/7/09

EECS 584, Fall 2009

34