NJIT New Jersey's Science & Technology University

A. V. Gerbessiotis

CS 485-101

Fall 2011

Exam 1 Solutions (250 points)

120 minutes

Name: .....Solutions.....

ID Number: ..... **Exam Number:** .....

Grade: 1: ... 2: ... 3: ... 4: ... 5: ... 6: ... 7: ... 8: ... Total: .....

Solve **ALL the problems** in the space provided

Read the Problems CAREFULLY!

There are 7 (SEVEN) pages this page included

Read and Sign the statement below at the end of the exam

Unsigned exams will be marked with grade 0 (zero).

STATEMENT

On my honor, I pledge that I have not violated the provision of the NJIT Student Honor Code.

Sign below at the end of the exam

Signature .....

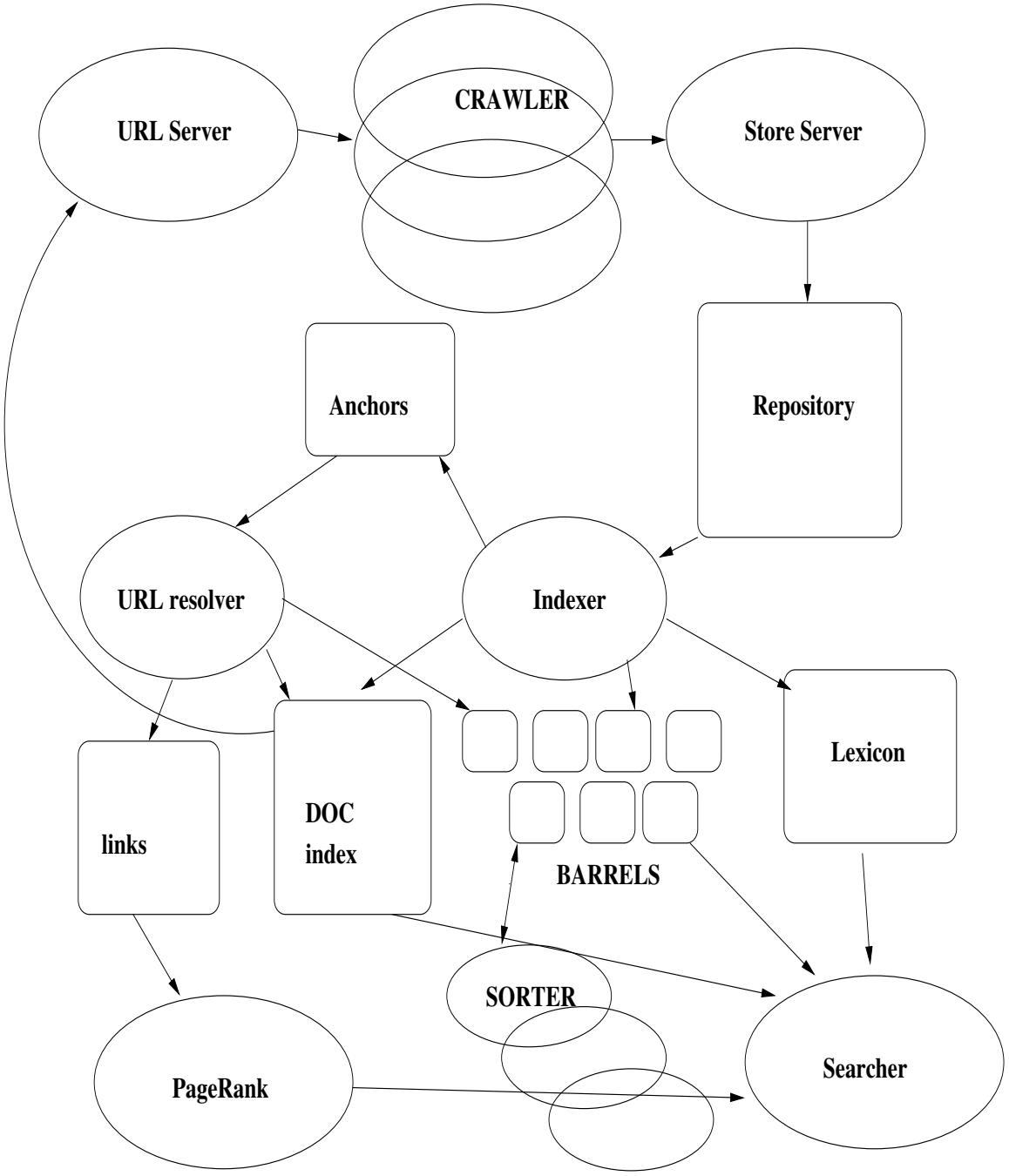

Figure 1: Google System Architecture.

Problem 1. (70 POINTS)
Give short answers to the following questions. (References related to Google are to the Google System Architecture of 1997/1998.)

(1) Does `Anchors` store information in the form of `URLs` or `docIDs`?
URLs

(2) Where are web-pages stored in Google?
Repository

(3) Does Google store web-pages in compressed or uncompressed format?
Compressed

(4) Name (do not further discuss) 5 data structures used by Google.
BigFiles, Repository, ForwardIndex, InvertedIndex, Document Index, Lexicon, HitLists.

(5) Give a data structure in Google that stores only `docIDs` and neither URLs nor URL checksums.
Links (forward/inverted index stored in the barrels is/are another one.)

(6) What is the input for the `PageRank` computations?
Links

(7) Why does the `Searcher` access the `DOCIndex`?
To convert docIDs to URLs and provide title info for them; to obtain handles to access the repository copy of a web-page.

(8) Where does Google store the forward index of `anchor text`?
Short Barrels

(9) What do we call the input provided to the `SORTER` and what do we call the output generated by the `SORTER`?
forward index and inverted index respectively; both are stored into the barrels.

(10) Name an operation that can map two different `keywords` into the same `index term`.
stemming

Problem 2. (20 POINTS)
(a) What is the difference between a `keyword` and an `index term`? Give an example, if it helps.

A keyword might or might not become an index-term itself. An index-term is a keyword (or the suffix of a keyword after stemming is performed), but a keyword itself might not become an index-term. For example stemming might reduce the number of index-terms or what keywords or portion of a keyword becomes an index term. So `algorithms, algorithmicality, algorithmic` are all keywords but the common index-term is `algorithm` after stemming. Also `car, automobile, sedan, coupe` might be keywords but the corresponding index-term is `vehicle`. (Case-folding, synonyms might have the same effect.)

(b) What is the difference between a `word` and a `keyword` in the context of information retrieval? Explain by providing an example.

A keyword is a `key` word, an important word, worth identifying. Stopwords or other common word might not become keywords; numbers and dates also fall into this group. Stopwords such as `a, or, an, of, the` might never considered to be keywords. Some numbers are keywords (eg 1776) though some other numbers (5123591) might not become keywords.

Turn page

Problem 3. (20 POINTS)

(a) Give an example of a bit-aligned code, and an example of a byte-aligned code.

Bit aligned: Elias codes , **Byte aligned: ASCII or UNICODE.**

(b) Give an example of a lossless compression method, and an example of a lossy one.

Lossless: GIF or Huffman and `Lossy: JPEG.`

Problem 4. (20 POINTS)

For a specific query involving term t, the corpus (collection of documents) includes the twenty documents $d_1, \ldots, d_{20}$. The first five $d_1, \ldots, d_5$ contain index term t and are thus relevant, whereas the remaining fifteen $d_6, \ldots, d_{20}$ do not contain t at all.

As a response to the query the search engine returned documents

$$d_1, d_3, d_8, d_4, d_7, d_9, d_{10}, d_{11}, d_{12}, d_{13}$$

in that specific order.

(a) What is the precision of the search query? Explain.

Only three of the ten documents are relevant that is d_1, d_3, d_4**. That is precision is** $3/10$ **i.e.** 30%**.**

(b) What is the recall of the search query? Explain.

Only three of the five relevant documents are retrieved, i.e. d_1, d_3, d_4 **of** $d_1, \ldots, d_5$**. Thus the recall is** $3/5$ **i.e.** 60%**.**

Problem 5. (32 POINTS)

Which one of the 4 marks `AND`, `OR` , `NOT`, and `-` are operators in (i) Google, and (ii) Bing?

In Google `AND, OR , -` are operators but `NOT` is NOT (sic).

In Bing, all four `AND,OR,NOT,-` are operators.

Problem 6. (28 POINTS)

Give the unary code for 0, and binary, Elias-γ, Elias-δ codes for 30.

Unary for 0 is 0.

Binary for 30 is `11110`.

Elias-γ for 30. $k = 30$ i.e. $k_d = \lg 30 = 4$ and thus $w_d = 11110$. Then $k_r = k - 2^{k_d} = 30 - 2^4 = 14$ and 14 in $k_d = 4$ bits binary is 1110 i.e. $w_r = 1110$. Thus the Elias-γ for 30 is `111101110`.

Elias-δ for 30. $k = 30$ and by the prior discussion $w_r = 1110$. Since $k_d = \lg 30 = 4$ we need to find the Elias-γ for $k_d + 1 = 5$. Then $k_{dd} = \lg 5 = 2$ i.e. $w_1 = w_{dd} = 110$ and $k_{dr} = 5 - 2^2 = 1$ i.e. $w_{dr} = 01$ in $k_{dd} = 2$ bits. Thus $w_1 w_2 = 11001$, and $w_1 w_2 w_r = 110011110$.

Problem 7. (30 POINTS)
You have a machine with 1000MBytes of free memory. We want to use all this memory to store the `docID`, `URLhash` and `URL`s of a search engine, similarly to the structure used by the lexicon to store `wordID`s and `words` in which the latter are stored in consecutive memory location of an array separated by NULLs; in our case, URLs will be separated by NULLs. The Operating System consumes 70MB and other user programs including the ones dealing with the tables another 30MB. The average length of a `URL` in the search engine is around 99 characters ASCII. The hash table will store a 16-byte `URLhash`, a 5-byte `docID`, and a pointer/index p that points to the location in the table where the strings for the URLs are stored delimited by ASCII NULLs. Organize the hash table and the table of URLs so that you can extract from them maximum performance. How many URLs can be accommodated? What is the length m of the hash table? How many bytes for p? How much space in MBytes will your scheme consume (inclusive of OS plus programs)? Explain.

1. Let n the number of URLs and m the size of the hash table.

2. The strings of URLs is $n \times (99+1) = 100n$ bytes in total since 99 is the average length and 1 accounts for the NULL separator. (1 ASCII char uses 1 byte.)

3. Fix $m = 2n$ for maximum efficiency. Every slot of the table contains a 16-byte URLhash and 5-byte docID plus b for a pointer p. Total is $21 + b$ per slot or $2n(21 + b)$ altogether.

4. Total space by 2 and 3 plus the quoted $70MB+30MB = 100MB$ is thus $100n+2n(21+b)+100$MB, i.e. $142n + 2nb + 100MB$. A typical value for a pointer uses $b = 4$. Thus $142n + 8b + 100MB$ if set to 1000MB gives $n = 6,000,000$. Thus $m = 12,000,000$ and since $b = 4$, we use up all available memory completely. Even if he had used $b = 3$ instead, the savings would not have been noteworthy (and note that pointers or other structures should respect word boundarie: this means they ought to be chosen to be multiples of 4 or in 64-bit addressing, multiples of 8). In other words, Google in 1998 could not maintain the info for 14,000,000 pages in the main memory of one machine!

Problem 8. (30 POINTS) We typed the following queries in Google and Bing and recorded the following number of hits in millions for the two search engines. Answer the following questions with respect to the Table below.

No	Query Term(s)	GOOGLE (number of hits in '000,000)	BING (number of hits in '000,000)
1	⟨`a b or c`⟩	2530	2680
2	⟨`a b OR c`⟩	22100	5150
3	⟨`A B OR C`⟩	22100	5150
4	⟨`a (b OR c)`⟩	25270	6870
5	⟨`(a b ) OR c)`⟩	22100	5150
6	⟨`a or b`⟩	3230	3090
7	⟨`a OR b`⟩	25270	11500
8	⟨`b or a`⟩	3270	73
9	⟨`b OR a`⟩	25270	11500

(a) Why the difference between query 1 and query 2 ?
OR is an operator in both search engines but `or` is a keyword.

(b) Why the similarity between query 2 and query 3 ?
Capitalization is not important in either search engine.

(c) Why the similarity between query 2, query 3, and query 5 ?
There is no difference between 1 and 2 by (b). The parenthesization which is explicit in 5 is implied in 2 and 3 since a conjunction has a higher priority over a disjunction.

(d) What's the difference between query 6 and query 7?
OR again is an operator but or is not

(e) Why the similarity between query 7, and query 9 ?
a OR b is the same as b OR a (commutativity).