

Name:

ID Number:

Exam Number:

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

SOLVE ALL the problems IN THE SPACE PROVIDED

Read the Problems CAREFULLY!

DURATION OF THE EXAM IS 105 minutes

THERE ARE 7(SEVEN) PAGES THIS PAGE INCLUDED

YOU MAY TEAR OFF CAREFULLY THE LAST PAGE; DO NOT TURN IT IN THEN

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

Problem 1. (100 POINTS)

Give short answers to the following questions. (References to Google are to the Google System Architecture of 1998 unless otherwise noted.) If a question has multiple answers write all that apply.

- (1) Which one(s) of **NOT**, **not**, **-** are operators in Google@2016?
- (2) Which one(s) of **NOT**, **not**, **-** are operators in Bing@2016?
- (3) What data structure(s) are written by the **URL resolver**?
- (4) Where is the input to the **Sorter** stored?
- (5) Does **LINKS** use **URL**, **URLsum** (hashes), or **docIDs** to represent information?
- (6) What is the input to the **Indexer**?
- (7) We have the **Web**, **Semantic Web**, and **Surface Web**. What else that ends with **Web**?
- (8) What program compresses web-pages in Google@1998?
- (9) What program decompresses web-pages in Google@1998?
- (10) List all data structures (of the Figure of Page 7) that store a **docID**.
- (11) Who writes into the short barrels in Google@1998?
- (12) Why does the **Searcher** access **DocIndex**?
- (13) What do we call the input provided to the **SORTER** and what do we call the output generated by the **SORTER**?
- (14) What data structure allows the conversion of **docIDs** into **URLs**?
- (15) Why does the **Searcher** access the **Lexicon**?
- (16) **Bot**, **robot**, **spider**, **harvester** all refer to the same thing. Give another name for it.
- (17) What do we call a word that we type in the search box of a web-search engine?
- (18) What is the difference between a keyword and an index-term. Explain and provide an example.
- (19) What is a frequently occurring token (word) that is not an index-term?
- (20) What does the acronym **URL** stand for?

Problem 2. (60 POINTS)

For this problem $1/7=15\%$, $1/3= 33\%$, $2/3 = 66\%$, $3/7 = 45\%$, $3/8 = 40\%$, $4/9 = 45\%$, $4/7=55\%$, $5/6 = 80\%$, $5/7 = 70\%$, $5/8= 60\%$, $5/9 =55\%$, $200/3 = 66$, $220/3= 76$, $260/3 = 86$. For a query Q involving term t_1 , the corpus includes the twenty documents $d_1, \dots, d_{20}$. Four of them, d_1, d_2, d_3, d_4 contain index term t_1 and are relevant, whereas the remaining ones do not contain the index term and are irrelevant. As a response to the query involving t_1 an experimental Search Engine called **Gooing** returned the following 10 documents; however there were three versions, V1, V2, and V3 of the engine and the documents that were returned are as follows for each version.

Gooing-V1	Gooing-V2	Gooing-V3
d5	d1	d1
d6	d5	d2
d7	d2	d5
d8	d6	d6
d9	d3	d3
d10	d7	d4
d1	d4	d7
d2	d8	d8
d3	d9	d9
d4	d10	d10

- (a) **(10 points)**. What is the precision for this query Q for each one of V1, V2, V3? Explain. Is there a winner and which version has better precision?
- (b) **(10 points)**. What is the recall rate for this query of the search query? Explain. Any winner?
- (c) **(40 points)**. What is the 3-point effectiveness using 25%, 50%, 75% recall levels? Explain. Is there a winner (better effectiveness)? Explain.

Problem 3. (40 POINTS)

Provide Binary, Elias- γ , Elias- δ codes for $k = 140$. Provide the v -byte code(s) for 140 (in hexadecimal).

Problem 4. (40 POINTS)

A corpus contains 32,000,000,000 documents UNICODE-encoded text documents, utilizing 32 different languages. The average byte length of a UNICODE-encoded text document is 96,000 bytes and the average length of a word in the corpus is 5 characters. Provide an estimate, as realistic as possible, of the (average) lexicon size (distinct words) of any given language of the corpus. (The intent is for you to end up with a nice round number. Do not mess it up! If you need a β choose a 0.5; if you need a k choose a 100.)

Problem 5. (45 POINTS)

The brackets below $\langle$ and $\rangle$ indicate the start and end of a query; they are not typed in the query. If the query system is accepting Boolean queries, answer the following questions. OR, NOT, AND, " (double quotation mark) are the operators for this imaginary engine; order of evaluation as in Boolean Algebra. Case of search terms is ignored. Note the double quotation marks in the last query.

No	Query Terms	Hits
Q1	$\langle \text{to} \rangle$	9000
Q2	$\langle \text{be} \rangle$	6000
Q3	$\langle \text{to be} \rangle$	3000
Q4	$\langle \text{to be or not} \rangle$	2000
Q6	$\langle \text{TO BE OR NOT TO BE} \rangle$	1000
Q7	$\langle \text{to be or not to be} \rangle$	1000
Q8	$\langle \text{to be OR not to be} \rangle$	3000
Q9	$\langle \text{toOR} - \text{to} \rangle$	12000

Table 1: Table for Problem 5

(a) Explain in English the (minimal) meaning of Query Q6. Is there any inconsistency between the results of Q6 and those of Q1-Q4? Explain. If thereis, correct the result for the query to be consistent with those of Q1-Q4.

(b) Explain in English the (minimal) meaning of Query Q7. Is there any inconsistency between the results of Q7 and those of Q1-Q4? Explain. If thereis, correct the result for the query to be consistent with those of Q1-Q4.

(c) Explain in English the (minimal) meaning of Query Q8. Is there any inconsistency between the results of Q8 and those of Q1-Q4? Explain. If thereis, correct the result for the query to be consistent with those of Q1-Q4.

Example. For say Q9 the (minimal) meaning of it is "all documents of the corpus" rather than "documents that contain to or documents that do not contain to".

Problem 6. (48 POINTS)

You have 2048MiB of main memory of which 548MiB are being used by the operating system plus related programs such as those manipulating tables A and T below. You are asked to organize the remaining space to support a hash table where URLs are stored in a contiguous table which is an array A of characters delimited by a null character `\0`. A hash table T will store a `docID` along with a reference (pointer or index) p to A . That way a `docID` will be associated with a specific URL. The average length of a URL is given as 63 UNICODE characters (2B encoding each). A `docID` and p can only be in multiples of a byte (i.e. 1B, 2B, 3B, 4B, thus 21bits won't be an option) for efficiency. Organize T and A for maximum efficiency. Round to nearest million for n, m, T, A but make sure your rounding doesn't exceed the amount of available memory (i.e. $A + T$ should be accommodated easily by the available memory). Make sure that you fill the following table with information. (**No calculator needed if you are careful.**)

<code>n</code>	=	(a) How many URLs can the scheme support,
<code>m</code>	=	(b) how big would the hash table size <code>m</code> be,
<code>docID</code>	=	(c) how many bits for a <code>docID</code> ,
<code>p</code>	=	(d) how many bits for pointer <code>p</code> ,
<code>T</code>	=	(e) how much space (bytes) will you scheme use for <code>T</code> ,
<code>A</code>	=	(f) how much space (bytes) will you scheme use for <code>A</code> , and
<code>A+T</code>	=	(g) what is the total space <code>\$A+T\$</code> used by your scheme?

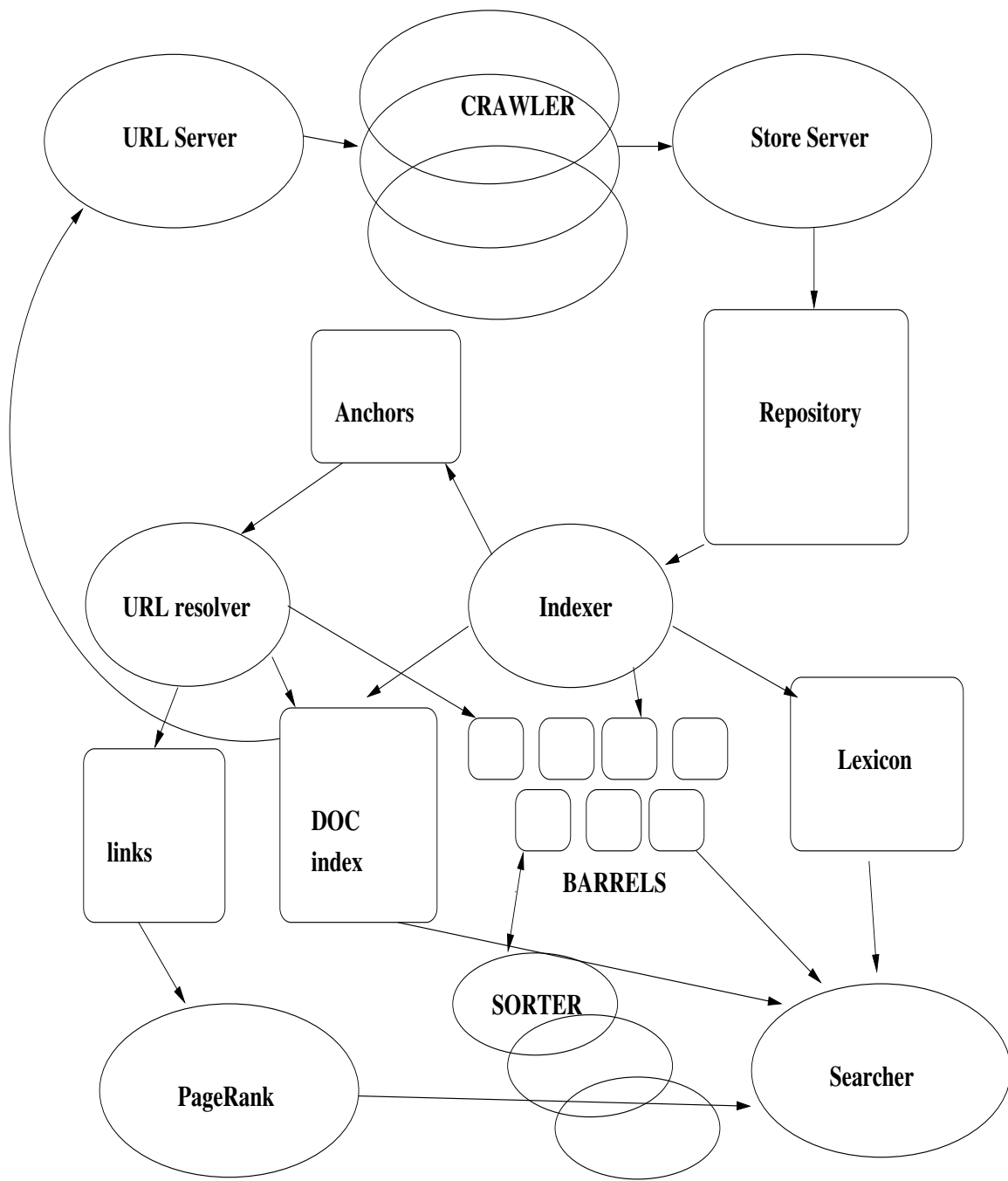


Figure 1: Google System Architecture.