

Name:

ID Number:

Exam Number:

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

SOLVE ALL the problems IN THE SPACE PROVIDED

Read the Problems CAREFULLY!

DURATION OF THE EXAM IS 90 (NINETY) 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. (65 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 **OR**, **or**, **Or** are operators in Bing?
- (2) Which one(s) of **NOT**, **not**, **-** are operators in Bing?
- (3) What is the input of the **URL resolver**?
- (4) What is the input to the **Sorter**?
- (5) Where (location) is the input to the **Sorter**?
- (6) What is the output of the **Sorter**?
- (7) Where (location) is the output of the **Sorter**?
- (8) List all data areas (rectangular shapes of the Figure) that store a **docID**.
- (9) Name/List exactly 5 data structures used by Google excluding the Barrels, for forward index and the (inverted) index.
- (10) Who writes into the short barrels in Google?
- (11) Why does the **Searcher** access **DOCIndex**?
- (12) Who generates a new Lexicon after a new indexing cycle?
- (13) What data structure allows the conversion of **docIDs** into URLs?

Problem 2. (50 POINTS)

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

- (1) Bot, robot, spider, harvester all refer to the same thing. Give another name for it.
- (2) What do we call a frequently occurring word in a text that is not a keyword?
- (3) What is the difference between a search term and an index-term. Explain and provide an example
- (4) Name an operation that can map two different **keywords** into the same **index term**.
- (5) What is the difference between a **word** and an **index-term** in the context of information retrieval? Give an example that highlights the difference.

Problem 3. (85 POINTS)

For this problem $1/3= 35\%$, $2/3 = 65\%$, $3/7 = 45\%$, $3/8 = 40\%$, $4/9 = 45\%$, $5/6 = 80\%$, $5/7 = 70\%$, $5/8= 60\%$, $5/9 =55\%$.

For a query involving term t_1 , the corpus includes the twenty documents $d_1, \dots, d_{25}$. Five of them, $d_1, d_2, d_3, d_{10}, d_{11}$ 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 a new Search Engine called **Gooing** returned the following 10 documents.

$d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8, d_{10}, d_{11}$

in that specific order (left to right).

- (a) **(10 points)**. What is the precision for this query of the search query? Explain.
- (b) **(10 points)**. What is the recall rate for this query of the search query? Explain.
- (c) **(30 points)**. What is the 3-point effectiveness using 40%, 60%, 80% recall levels? Explain. This means build as done in class and in HW2.
- (d) **(30 points)**. Another search engine **Bingle** returned the following 10 documents. What is its 3-point effectiveness.

$d_1, d_2, d_3, d_{10}, d_{11}, d_4, d_5, d_6, d_7, d_8$

- (e) **(5 points)**. Between Gooing (first engine) and Bingle (second engine) which one would you consider as more effective, using the same effectiveness measure? Justify your answer.

This table is given for your assistance

	Rlvnt	Recall	Prcsion	recall	prec	interprec
d1					0%	
d2					20%	
d3					40%	
d4					60%	
d5					80%	
d6					100%	
d7						
d8						
d10						
d11						

This table is given for your assistance

	Rlvnt	Recall	Prcsion	recall	prec	interprec
d1					0%	
d2					20%	
d3					40%	
d10					60%	
d11					80%	
d4					100%	
d5						
d6						
d7						
d8						

Problem 4. (25 POINTS)

What is the *v*-byte code in hexadecimal of 65 and 140?

Problem 5. (30 POINTS)

A corpus contains 25,000,000 documents. The average length of a document encoded in ASCII is 100,000 bytes and the average length of a word in the corpus is 9 characters. Provide an estimate, as realistic as possible, of the lexicon size in words. (The intent is for you to end up with a nice round number. Do not mess it up!)

Problem 6. (30 POINTS)

We typed the following queries in search engine Gooo and Beee and recorded the following number of hits (in millions) for the two search engines. The query system of the two engines is accepting Boolean queries and all of NOT , OR , AND are operators. Answer the following questions with respect to the Table below.

No	Query Terms	Gooo	Beee
Q1	$\langle \text{to} \rangle$	6000	10000
Q2	$\langle \text{be} \rangle$	2000	3000
Q3	$\langle \text{or} \rangle$	2100	3500
Q4	$\langle \text{not} \rangle$	1800	5000
Q5	$\langle \text{to be} \rangle$	1900	10000
Q6	$\langle \text{to be OR not} \rangle$	6000	15000

Table 1: Table for Problem 6

- (a) Is there any inconsistency in the output/operation of **Beee**? Explain.
- (b) For Q6 do the results for **Gooo** or **Beee** make sense? Explain inconsistencies that you might see.

Problem 7. (40 POINTS)

Assume $1\text{GB} = 10^9 B$, and $1\text{MB} = 10^6 B$. Round to the nearest million that won't exhaust your available memory; you don't need a calculator.

You have 512MB of main memory of which 204MB 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 along the lines of the paper (and the notes) where words are stored in a contiguous table which is an array A of characters delimited by a null character $\backslash 0$. A hash table T will store a **wordID** along with a reference (pointer or index) p to A . That way a **wordID** will be associated with a specific **word**.

The average length of a **word** is given as 14 UNICODE characters (2B encoding each). A **wordID** and p can only be in multiples of one byte (i.e. 1B, 2B, 3B but and thus 21bits won't be an option) for efficiency. Organize T and A for maximum efficiency.

In an efficient implementation

- (a) How many words n can the scheme support,
- (b) How big would the hash table size m (number of entries) be,
- (c) How many bits or bytes for a **wordID**,
- (d) How many bits or bytes for pointer p ,
- (e) How much space (bytes) will your scheme use for T ,
- (f) How much space (bytes) will your scheme use for A ,
- (g) What is the total space $A + T$ used by your scheme?

Justify your answers and choices. 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.**)

n	=
m	=
wordID	=
p	=
T	=
A	=
$A+T$	=

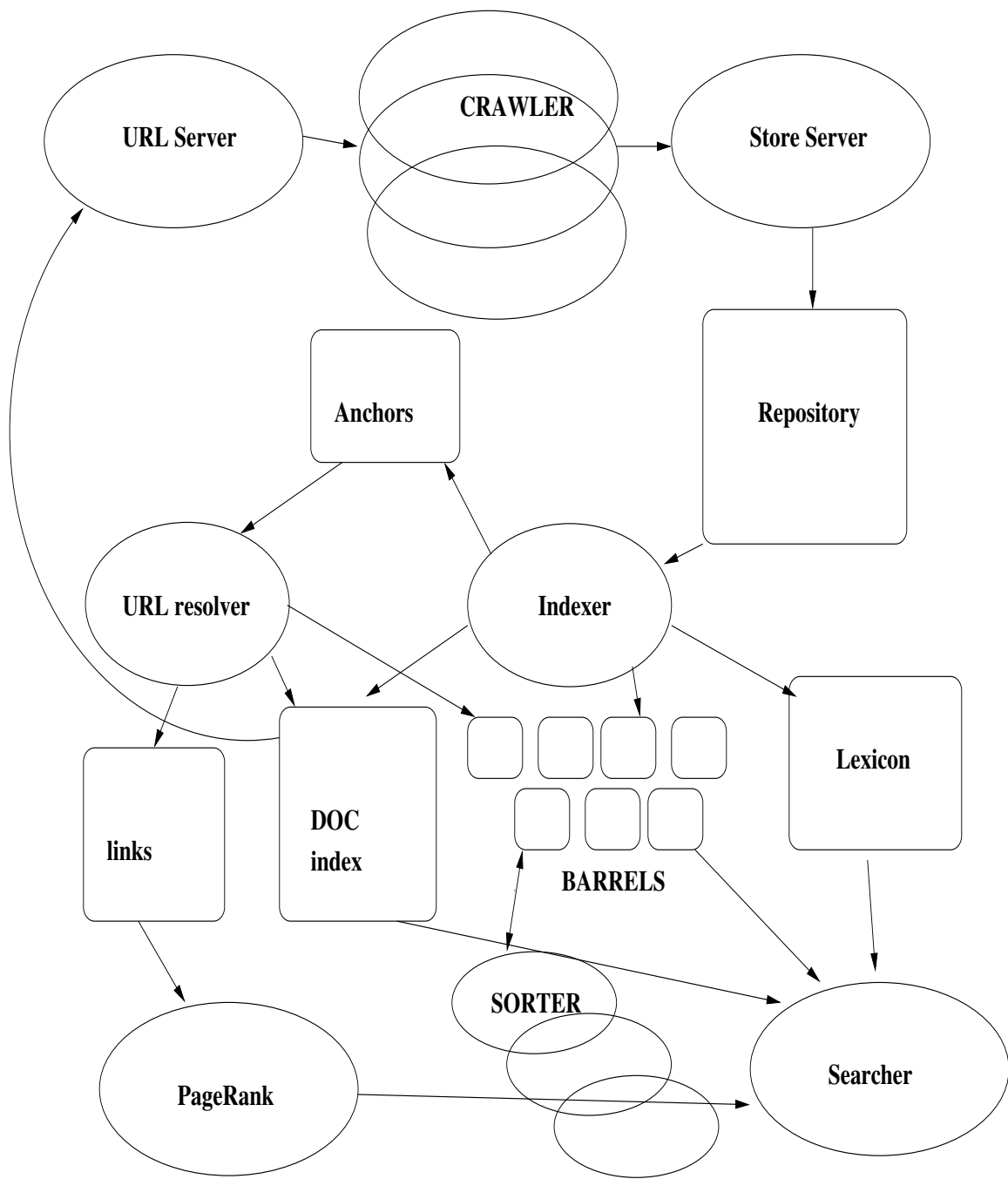


Figure 1: Google System Architecture.