CSC 483/583: Midterm (200 pts)

March 1st, 2017

Name (2 pts):

KEY

Instructions

Read each question carefully before determining the best answer. If you don't know how to answer a question, you might want to skip it until you've done the questions you're confident about. **Show all work**, and indicate if any work for credit is included somewhere other than directly below the question.

You are allowed one $8\frac{1}{2}'' \times 11''$ page (double-sided) of notes and a simple, self-contained hand-calculator. If you do not own a hand-calculator you may use your cell phone strictly as a calculator. No devices of any kind are allowed to access the Internet!

Violation of this policy or of the student code of academic integrity results in a penalty greater than the value of this exam up to and including failing the course.

1. **(28 pts) A few short questions**

-1 for small mistakes
-6 for completely wrong
-6 if solved using the k-gram index

a. (7 pts) If you wanted to search for `re*ri*v*l` in a Permuterm tree, what key would you do the lookup on in the tree? What would the post-processing do (if anything)?

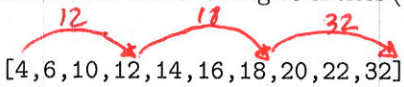
SEARCH for "l$re"
Postprocessing: make sure string matches *l*i*v*

b. (7 pts) A software program (a crawler) gathers a collection of documents in various formats (pdf, Word, etc.) and sends them to an indexer, which employs the following modules:
(A) a stemmer;
(B) a language detector to detect the language of each document;
(C) a stop-word eliminator;
(D) a filter that detects the format (pdf, Word, etc.) of the document.
(E) converters from one given format (e.g., pdf) to plain text.

Give the correct sequence in which the indexer should apply these modules to a document:

DEBAC or DEBCA

c. You are given a two-word query. For the first term the postings list consists of the following 10 entries (these are absolute doc ids, not gaps):



For the other it is the a postings list with a single entry: `[20]`. Work out how many comparisons would be done to intersect the two postings lists, assuming:

(i) (7 pts) standard postings lists (list all comparisons!)

-1 if they include 22

← 4, 6, 10, 12, 14, 16, 18, 20 => 8

(ii) (7 pts) postings lists stored with skip pointers, with a skip length of $\sqrt{length}$, as recommended in class. List all comparisons and briefly justify your answer, for example, by drawing the skip pointers in the first list.

$\sqrt{10} = 3$ or 4

4, 12, 18, 32, 20

-2 if they miss 32
-4 show work
0 if skip pointers end if diff. places (but be consistent!)
-2 otherwise

2. **(20 pts) Positional indexes with gaps** Consider the following positional index stored using an extension of the *gap* encoding covered in class. Here we use gaps to encode ***both*** docids and postings, in the form:

<word:
docid1: position, offset_from_previous_position, offset_from_previous_position ... ;
offset_from_docid1_to_docid2: position, offset_from_previous_position, offset_from_previous_position ... ;
offset_from_docid2_to_docid3: position, offset_from_previous_position, offset_from_previous_position ... ;
etc.
>

<information:	<retrieval:
11: 7, 18, 33;	**11:** 6, 20, 33, 72;
2: 3, 149;	**3:** 34, 19;
54: 17, 11, 291;	**53:** 107, 191, 22, 40, 434;

There are no skip pointers. We wish to process the phrase query "information retrieval" on this index.

(a) (10 pts) Fill in the positional index below using only absolute positions and docids (no offsets!).

-5 for incorrect doc ids

-5 for incorrect position ids

<information:	<retrieval:
11: 7, 25, 58	11: 6, 26, 59, 131
13: 3, 152	14: 34, 53
67: 17, 28, 319	67: 107, 298, 320, 360, 794

(b) (10 pts) What are all the documents and all the absolute positions at which the query phrase occurs?

-5

-~~5~~ for each missed result

11: 25, 58

67: 319

3. **(20 pts) Stemming and retrieval**

(a) Mark these statements **true/false**:

Statement	T/F
In a Boolean retrieval system, stemming generally increases recall (4 pts)	T
Stemming increases the size of the lexicon (3 pts)	F
Stemming should be invoked at indexing time but not while doing a query (3 pts)	F

(b) Mark these statements **true/false** (2 pts each):

During an ideal stemming procedure for a general-purpose search engine:

Statement	T/F
"abandon" should yield the same stem as "abandonment"	T
"university" should yield the same stem as "universe"	F
"marketing" should yield the same stem as "markets"	T
"organism" should yield the same stem as "organization"	F
"policy" should yield the same stem as "police"	F

+2 points to all grades!

4. **(20 pts) Compression**

Consider the postings list <4, 10, 11, 15, 265, 268> with a corresponding list of gaps <4, 6, 1, 4, 250, 3>. Assume that the length of the postings list is stored separately, so the system knows when a postings list is complete. For your information, these postings/gaps are represented using binary code as follows:

- 4: 100
- 6: 110
- 1: 1
- 250: 11111010
- 3: 11

Assume variable byte encoding for the encoding method referenced in the questions below:

(a) (5 pts) What is the largest gap in this list you can encode in 1 byte?

6

(b) (5 pts) What is the largest gap in this list you can encode in 2 bytes?

250

(c) (5 pts) How many bytes will the above postings list require under this encoding? (Count only space for encoding the sequence of numbers.)

-1 if encoding unique numbers

4, 6, 1, 4, 250, 3
1 1 1 1 2 1 => 7

(d) (5 pts) Write down the actual sequence of numbers encoded with variable byte encoding.

4: 1...100,
6: 1...110,
1: 1...1,
4 250: 0... 1|11111010
3: 1...11,

-3 if missing control bit
-2 for incorrect encoding of 250

5. **(20 pts) Vector space model**

Compute the vector space similarity between the query "digital cameras" and the document "digital cameras video cameras" by filling out the empty columns in the table below. Assume N = 10,000,000, logarithmic term weighting (wf columns) for query and document, idf weighting for the query only and cosine normalization for the document only. Use base 10 for all logarithms. What is the final cosine similarity score?

You might need this: $log_{10}(2) = 0.3$

	Query					Document			
word	tf	wf	df	idf	q_i = wf × idf	tf	wf	d_i = normalized wf	$q_i.d_i$
digital	1	1	10,000	3	3	1	1	0.52	1.56
video	0	0	100,000	2	0	1	1	0.52	0
cameras	1	1	50,000	2.3	2.3	2	1.3	0.67	1.54

3.1

$idf = \log_{10}\left(\frac{N}{df}\right)$

$\|d\|_2 = \sqrt{1+1+1.69} = \sqrt{3.69} = 1.92$

-2.5 points for each incorrect column, where the last column must show the sum

6. **(30 pts) Building a transplant donor search engine**

You are hired by a medical organization to build a search engine that identifies organ transplant donors for patients with end-stage renal disease. Both patients and donors are described by their blood type (A, B, AB, and O) and by a number of human leukocyte antigens (HLA), which are markers present in cells of the body that distinguish each individual as unique. A match between a patient and a donor is better when the overlap between blood type and HLAs between the two is maximized. Given this, answer the following issues:

(a) (10 pts) What is a "term", a "document", and what is the "query" for this search engine?

term: blood type or HLA
document: donor
query: patient

-3 if getting any wrong

(b) (10 pts) Describe a tiered index with at least two tiers that would serve this task well. What is contained in each tier? Briefly justify your decisions.

Tier 1: live donors
2: deceased

Tier 1: donors with type O blood
Tier 2: others

Tier 1: age < 60

-8 if tiers are not independent of the patient

(c) (10 pts) Describe a zone index that would serve this task well. What is contained in each zone? How is the search modified to take advantage of these zones?

zone 1: blood type
zone 2: HLAs
zone 3: location

search: Boolean search in zone 1.
Only if match move to zone 2.
↳ Vector space model here

-2 if each blood type in a different zone

7. **(20 pts) Edit distance**

Consider the standard edit distance algorithm below:

```
LEVENSHTEINDISTANCE(s1, s2)
 1  for i ← 0 to |s1|
 2  do m[i, 0] = i
 3  for j ← 0 to |s2|
 4  do m[0, j] = j
 5  for i ← 1 to |s1|
 6  do for j ← 1 to |s2|
 7      do m[i, j] = min{
 8              m[i − 1, j − 1] + if (s1[i] == s2[j]) then 0 else 1 fi,
 9              m[i − 1, j] + 1,
10              m[i, j − 1] + 1
11          }
12  return m[|s1|, |s2|]
```

Apple Inc., your new employer that recently poached you from the previous medical organization, hired you to change this algorithm so it works on the new iPhone. That is, the cost of replacing one character with another is no longer 1, but it is a function of the position of the two characters on the iPhone keyboard: if the two characters are adjacent, then the cost is 0.5; otherwise the cost is 1. Suppose you are given a procedure: *adjacent*(c_1, c_2), which returns True if the characters are adjacent on the keyboard, and False otherwise.

Which line(s) in the algorithm above need to change? → line 8

Write the new line(s) of pseudocode below.

then 0 else if adjacent (s1[i], s2[i])
then 0.5
else 1
fi
fi

−15 if they modify other lines in the algorithm

−10 if they change line 8, but also other lines

8. **(20 pts) Top K retrieval**

You are now employed by Twitter to revamp their search infrastructure for tweets. How would you design a "document-at-a-time" top K retrieval algorithm for tweets? Discuss all components that must be implemented, and your approach for all.

0. Doc == tweet

1. Sort tweets in descending order of a global score driven by number of retweets.

2. Adjust relevance_score to be:
   (cosine_sim + global_score
   → this allows early termination

-10 if each point is missed

-5 if "g" is not described

9. **(20 pts) Evaluation: recall, precision, accuracy**

a. (10 pts) An IR system returns 3 relevant documents, and 2 irrelevant documents. There are a total of 10 relevant documents in the collection. What are the precision and recall of this system on this search?

$TP = 3$, $FP = 2$, $FN = 7$ | $P = 3/5$, $R = 3/10$ — −5 each

b. Instead of using recall/precision for evaluating IR systems, we could use accuracy of classification. Consider a non-ranking IR system that classifies documents as being either relevant or non-relevant.

i. (5 pts) Why do the recall and precision measures reflect the utility (i.e., quality or usefulness) of this IR system better than accuracy does?

Because they ignore the many documents that are not relevant AND the system did not retrieve (ignore TN)

ii. (5 pts) Suppose that we have a collection of 10 documents, out of which 2 are relevant for query q, and two different boolean retrieval systems A and B. Give an example of two unranked result sets, A_q and B_q, assumed to have been returned by the systems in response to the query q, constructed such that A_q has clearly higher utility and a better score for precision than B_q, but such that A_q and B_q have the same scores on accuracy.

A_q : 2 correct + 2 incorrect $\Rightarrow$ $Acc = \frac{2+6}{10}$

B_q : returns nothing $\Rightarrow$ $Acc = \frac{8}{10}$

(there are other possible solutions)