COMP 336/533 Information Retrieval

FINAL EXAMINATION

December 19, 1996

Time allowed: 2 hours

1. **[5]** In the vector-space model, is it possible for users to search for documents which DON'T contain a certain keyword? Justify your answer.

   A negative weight can be assigned to the word that the user doesn't want.

2. **[10]** In the experiment that I performed for IDI, I have showed that the normalization factor $\sqrt{\text{number of terms in document}}$ is better than the full normalization factor as defined in the vector-space model.

   (i) Explain why it is expensive to compute the full normalization factor from an inverted file.

   If only the inverted file is available (i.e., document vector file is not kept), it is next to impossible to know what keywords does a document contain in order to calculate the vector length.

   (ii) Explain why it is NOT always a good idea to totally discard the document length when documents are ranked.

   With full normalization, two documents containing $< x, x, y, y, z, z >$ will have the same score as $< x, y, z >$ with respect to a query, say, $< x, y >$. In reality, people will favor documents which mention the query keywords more often than other documents do.

   Note that, the answer "people may favor long documents" is not a perfect answer. It is not convincing that long documents containing many irrelevant keywords are still more relevant than the short ones. In the example I give, both documents contain the same set of keywords, it is just that the term frequencies are different. Thus, it is arguable that the one containing more occurrences of the query keywords is more relevant.

3. **[10]** What is the definition of *fallout rate*? Explain why we need fallout rate, in addition to precision and recall, to evaluate the retrieval effectiveness of a retrieval system.

   $\text{Fallout} = \frac{\text{number of nonrelevant items retrieved}}{\text{total number of nonrelevant items in the collection}}$

   Precision and recall don't take into account of the number of nonrelevant documents in the collection.

4. **[5]** In the KMP pattern matching algorithm, the speed of pattern matching depends on the characters and their arrangements in the pattern. Give the most important factor that influence the speed of the algorithm.

   Repetition of characters in the pattern.

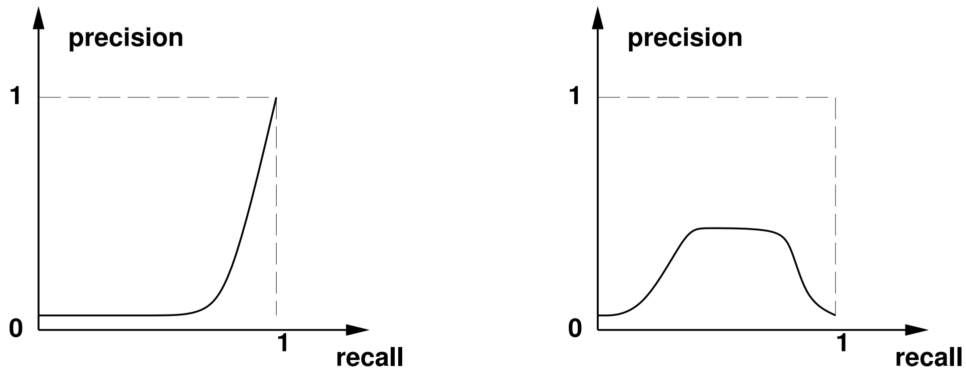
5. **[10]** Using the (improved) KMP method, fill in the following shift array, # is the end-of-string character.

pattern char	no. of shifts
a	1
a	2
a	3
a	4
a	5
b	1
#	6

What are the values for the next[] array?

0 0 0 0 0 5 1

6. **[10]** Is it possible for a precision/recall graph to look like (a) or (b). Explain why.



(a) (b)

(a) possible or impossible (circle your answer)? Justification:

It is impossible for the curve to touch the (1,1) point. At (1,1), it means both the recall and precision is one, meaning that all of the documents retrieved are relevant. However, this contradicts with the fact that precision is less than zero when recall is low, which means that some irrelevant documents are already retrieved.

(b) possible or impossible (circle your answer)? Justification:

It is possible. The curve means that there are relatively few relevant documents at the beginning and end of the retrieved documents but some relevant documents are concentrated in the middle of the retrieved documents.

7. **[10]** Imagine that you are the Chief System Analyst for an information provider. Your boss asks you to consider clustering as a mechanism to improve the speed of retrieval. You pull out all the notes that you kept for COMP 336/533 and study them overnight. What would be your recommendation and give at least two reasons to support your recommendation.

Don't recommend the use of clustering: (i) if the document is dynamic, re-clustering may have to be done periodically, which is an expensive process; (ii) if the user access pattern is not known, clustering may not give good precision and recall.

Recommend clustering: (i) speed can be improved; (ii) good clustering algorithms may improve the precision and recall as well. (Provided that document collection is rather static and user accesses are rather normal.)

8. [**10**] Fill in the following table with FAST, SLOW, MEDIUM to indicate the relative speeds of the three signature file methods in terms of search and insertion time, given that the signatures have the same width and the query signatures have the same number of ones in each case.

	sequential	Bit Slice	Frame Slice
search time	SLOW	MEDIUM*/FAST	FAST/MEDIUM*
insertion time	FAST	SLOW	MEDIUM

* It is hard to say which one is faster. I think Bit Slice should be slower because it requires one disk access for every bit set in the query signature, whereas Frame Slice typically *claims* to require one disk access. However, this claim may not be true, because a frame is larger than a slice and may require several disk pages to store.

9. [**10**] Given the following relations:

```
Document ( did integer, title text, body text );
```

containing the IDs, titles, and the body of the documents, and a function `score(text, text)`, which takes two text arguments as input and compute a similarity score s, $0 \leq s \leq 1$, write an Illustra SQL query to retrieve the IDs and scores of documents which have similarity greater than 0.5 to the string "java javascript plugin" and title containing the word "netscape", ordered by the scores.

```
select did
from   Document
where  title likes '%netscape%'
and    similarity(body, 'WWW network information' > 0.5
```

In performance evaluation, it is often necessary to generate a relation containing the query ID, the document ID and the document score for a collection of queries. Suppose the queries are stored in:

```
Query ( qid integer, body text );
```

Write an Illustra SQL query to generate the relation:

```
Result ( qid integer, did integer, score real );
```

```
insert into Result
select qid, did, similarity(Document.body, Query.body) score
from   Document, Query
order by qid, did
```

10. **[10]** Given three words and the random number generator generates the following sequences of integers, each between 0 and 15:

database	10, 6, 15, 10, 5, 7, ...
system	5, 9, 0, 11, 14, 8, ...
data	1, 12, 1, 9, 5, 8, ...

Given that the signature length is 16 and each word sets three (3) bits in the signature.

(a) Fill in the bits set in the following figure:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
database						1	1				1					1
system	1					1				1		1				
data		1				1				1			1			
superimposed	1	1				1	1			1	1	1	1			1

(b) Judged from this particular signature, is the resulting superimposed code optimal? Why?

It is optimal, because the number of 1's and 0's is about the same.

(c) What are the major differences between methods like BC (bit-block compression), CBS (compressed bit slices), etc., and superimposed coding in terms of the signature length and the number of bits set by each word?

Methods like BC and CBS typically uses a very long signature length but each word only sets one bit in the signature. (Thus, the signature is spare and requires compression.) Superimposed coding typically uses a shorter signature length and each word sets more than one bit in the signature.

11. **[10]** Given two retrieval systems s_1 and s_2, operating on two document collections c_1 and c_2, respectively. Suppose a query is submitted to both systems, and two rank lists of documents r_1 and r_2 are returned from s_1 and s_2, respectively.

(i) Explain why we cannot simply merge and sort r_1 and r_2 based on the document scores in r_1 and r_2 to obtain a global rank list for s_1 and s_2.

The score of a document is valid only within the collection it resides. It may be due to the different scales used in computing the document scores. However, even when the scales are the same, the first document in one server need not be better than the second (or the tenth) document in the other server.

The same document may get different scores in the two servers because of the differences in inverse document frequencies.

The same document may get different ranks in the two servers even when they have the same score.

(ii) What information has to be exchanged between s_1 and s_2 so that the rank lists returned are really the top documents globally?

The number of documents in each collection and the document frequencies of the query keywords.