Assignment 2 [I got 39 / 40]

Joey Tawadrous - 109528652

Problem 1 [10 points]

Consider these documents:

- Doc1: solution found for laziness
- Doc2: old laziness found
- Doc3: old approach for treatment of laziness
- Doc4: old hopes for laziness patients

A: Draw the term–document incidence matrix for this document collection.
B: Draw the inverted index representation for this collections.

A. Term-Document Incidence Matrix

	Doc1	Doc2	Doc3	Doc4
approach	0	0	1	0
for	1	0	1	1
found	1	1	0	0
hopes	0	0	0	1
laziness	1	1	1	1
of	0	0	1	0
old	0	1	1	1
patients	0	0	0	1
solution	1	0	0	0
treatment	0	0	1	0

B. Inverted Index (place a box around each term and each frequency)

approach	->	3				
for	->	1	3	4		
found	->	1	2			
hopes	->	4				
laziness	->	1	2	3	4	
of	->	3				
old	->	2	3	4		

patients -> 4
solution -> 1
treatment -> 3

Problem 2 [5 points]

Recommend a query processing order for the following Boolean query: (bush OR apricot) AND (pudding OR brown) AND (phones OR ears). Assume that the document frequencies of the terms in the above query are:

- ears 11331
- phones 9700
- pudding 10091
- brown 27160
- bush 4660
- apricot 21681

Explain the rationale of the order that you found.

Step 1
(bush OR apricot) (4660 + 21681)
AND
(pudding OR brown) (10091 + 27160)
AND
(phones OR ears) (9700 + 11331)

Step 2
(bush OR apricot) (26341)
AND
(pudding OR brown) (37251)
AND
(phones OR ears) (21031)

Step 3 (ordered)
(phones OR ears) (21031)
AND
(bush OR apricot) (26341)
AND
(pudding OR brown) (37251)

First we approximate the OR operator with the sum of the frequencies and then execute the query from lowest to highest frequency.

Problem 3 [5 points]
Why are skip pointers not useful for queries of the form x OR y?

- Skip pointers allow for faster postings merges.

In queries of the form “x OR y”, every docID in the posting lists of either terms is visited, thus removing the need for skip pointers.

Problem 4 [5 points]
Write down the entries in the permuterm index dictionary that are generated by the term **cork**

The entries are: cork$, ork$c, rkco, kcor, $cork

Problem 5 [5 points]
How do stopping and stemming reduce the size of an inverted index?

Stopping

- By eliminating the terms that have very long inverted lists. i.e. by removing very frequent words that appear in most of the documents and do not bear any meaningful content.

Stemming

- By reducing the number of inverted lists by joining the lists for two or more terms with the same stem. i.e. by reducing terms to their roots before indexing.

Problem 6 [10 points]
Consider the following collection:

- Doc1: new york times
- Doc2: new york post
- Doc3: los angeles times

Given the query **new times**, using tf/idf ranking documents.
Notice: tf–weight using: $1 + \log \mathrm{tf}_{t,d}$ and *log* is base 10.

Query = new times
D1 = new york times
D2 = new york post
D3 = los angeles times

D(number of documents) = 3
IDF = log(D/df)

W = ((1 + log(tf)) * idf)

Terms	Q	D1	D2	D3	df	D/df	IDF	Q	D1	D2	D3
angeles	0	0	0	1	1	3	0.4771	0	0	0	0.4771
los	0	0	0	1	1	3	0.4771	0	0	0	0.4771
new	1	1	1	0	2	1.5	0.1761	0.1761	0.1761	0.1761	0
post	0	0	1	0	1	3	0.4771	0	0	0.4771	0
times	1	1	0	1	2	1.5	0.1761	0.1761	0.1761	0	0.1761
york	0	1	1	0	2	1.5	0.1761	0	0.1761	0.1761	0

Q = how many times does the word appear in the query
Di = how many times does the word appear in the document
df (document frequency) = how many documents does the word appear in

Q, Di = ((1 + log(tf)) * idf)

Step 1	Step 2	Step 3	Step 4
|D1| = square root of ((0.1761)squared + (0.1761)squared + (0.1761)squared)	|D1| = square root of (0.03101121 + 0.03101121 + 0.03101121)	|D1| = square root of (0.09303363)	0.305014147 = 0.3050
|D2| = square root of ((0.1761)squared + (0.4771)squared + (0.1761)squared)	|D2| = square root of (0.03101121 + 0.22762441 + 0.03101121)	|D2| = square root of (0.28964683)	0.53818847 = 0.5382
|D3| = square root of ((0.4771)squared + (0.4771)squared + (0.1761)squared)	|D3| = square root of (0.22762441 + 0.22762441 + 0.03101121)	|D3| = square root of (0.48626003)	0.697323475 = 0.6973
|Q| = square root of ((0.1761)squared + (0.1761)squared)	|Q| = square root of (0.03101121 + 0.03101121)	|Q| = square root of (0.06202242)	0.249043008 = 0.2490

Q*Di below represents Q[dot]Di

Step 1	Step 2	Step 3
Q*D1 = 0.1761 * 0.1761 + 0.1761 * 0.1761	Q*D1 = 0.03101121 + 0.03101121	Q*D1 = 0.06202242 = 0.0620
Q*D2 = 0.1761 * 0.1761	Q*D2 = 0.03101121 = 0.0310	
Q*D3 = 0.1761 * 0.1761	Q*D3 = 0.03101121 = 0.0310	

Step 1	Step 2	Step 3	Step 4
Sim(Q, D1)	0.06202242 / 0.249043008 * 0.305014147	0.06202242 / 0.07596164	0.816496589 = 0.8165
Sim(Q, D2)	0.03101121 / 0.249043008 * 0.53818847	0.03101121 / 0.134032075	0.231371557 = 0.2314
Sim(Q, D3)	0.03101121 / 0.249043008 * 0.697323475	0.03101121 / 0.173663535	0.178570648 = 0.1786