CS433-CS533: Information Retrieval Systems, Midterm, Part 1: 55 points, March 28, 2021, 40 Min.
Note: You have to give the final value of arithmetic expressions such as write 0.5 instead of 1/2.
KEY

1. **(10 pts.) Query Processing**. When records/documents are grouped into fixed size blocks in secondary storage, it is frequently necessary to estimate the number of blocks, Nb, to be accessed for a given query. The following formula by Cardenas (CACM, 1975) assumes that there are n records divided into m blocks and the k records satisfying the query are distributed uniformly among m blocks.

$$N_b = m \left[1 - \left[1 - \frac{1}{m} \right]^k \right]$$

a. If there is only one record to retrieve then we have to access only one block. This is obvious. Does the formula developed by Cardenas imply the same? Explain your answer.

Nb: No. of blocks to be accessed
m: No. of blocks
k: No. of records to be accessed
With k= 1 the formula becomes m [1 - [1 - 1/m]] = m - m + 1= 1
Yes it follows the intuition as shown by the result.

b. Assume that for a particular query we want to access 5 records, and also assume that in the file there are 10 records and the block size is 2 (that is each block contains 2 records), find the value of Nb.

m= No. of records / Block size= 10 / 2= 5
k= No. of records to be accessed= 5
*Nb= 5 [1 - [1 - 1/5] **5]= 5 [1 - 0.8** 5]= approx 5 [1 - 0.33]= 5 - 1.65]= 3.35*

2. **(15 pts.) Inverted Files**. In this question consider an inverted file-based IR environment. The following observations are provided: (No. of Query Terms / Occurrence Probability): (1/0.20), (2/0.40), (3/0.20), (4/0.10), (5/0.10). For example, the first entry in the list indicates that 20% of the queries contain 1 term (i.e., their probability is 0.20), etc.. For the same environment, the posting lists have the following characteristics in terms of (Posting List Page Length / Occurrence Probability): (1 / 0.40), (2 / 0.30), (3 / 0.20), (4 / 0.10). For example, the first entry of the list shows that, 40% of the posting lists occupy 1 page, etc.

No. of Query Terms / Occurrence Probability: 1 / 0.20, 2 / 0.40, 3 / 0.20, 4 / 0.10, 5 / 0.10
Posting List Page Length / Occurrence Probability: 1 / 0.40, 2 / 0.30, 3 / 0.20, 4 / 0.10

a. Determine the expected number of terms in an average query.

A= Expected number of terms in an average query= 1 x 0.20 + 2 x 0.20 + 3 x 0.20 + 4 x).10 + 5 x 0.10= 2.50 terms

b. What is the expected number of posting pages to be accessed for an average query?

B= Expected posting list length= 1 x 0.40 + 2 x 0.30 + 3 x 0.20 + 4 x 0.10= 2.00 page

Expected number of posting pages to be accessed for an average query= A x B= 2.50 x 2.00= 5.00 pages

c. If page/block access time is 30 millisec what is the expected query processing time in seconds?

Expected query processing time= No. of pages to be accessed x Time needed for one page = 5 x 30= 150 millisec
In seconds = 0.150 sec.

3. (15 pts.) C^3M Clustering. Consider a binary document by term D matrix with 20,000 documents defined by 60,000 terms. Assume that 1% of the D matrix positions contain a non-zero value. For this D matrix please answer the following questions (for each case please give a number, i.e., do not leave it as an algebraic expressions). When necessary use the clustering indexing relationships implied by the concepts of C^3M.

m= 20,000
n= 60,000
D density= 1%= 0.01= 10^{-2}

a. What is the average depth of indexing (avg. no of terms/document)? x_d

t: total no. of non-zero entries in D
x_d= t / m= (m x n x D density) / m= n x D density= 60,000 x 10^{-2} = 600 terms / doc

b. What is the average term generality (avg. no of documents/term)? t_g

t_g= t / n= (m x n x D density) / n= m x D density= 20,000 x 10^{-2} = 200 documents / term

c. What is the expected number of clusters?

n_c= (m x n) / t= (m x n) / (m x n x D density)= 1 / 10^{-2}= 100

d. What is the expected number of C matrix positions that we need to calculate if we use C^3M for clustering? Briefly explain why?

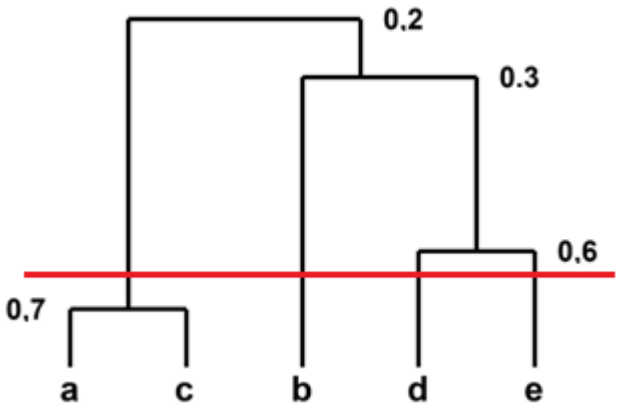
= m + (m - n_c) x n_c = 20,000 + (20,000 - 100) x 100= 20,000 + 19,900 x 100
= 20,000 + 1,990,000= 2,010,000

m + (m - nc) x n_c:
m: We need to calculate m no. of entries of C to find the no. of clusters
(m - nc) x n_c: We have to calculate the cover coefficient value for each non seed document (m - nc)
for all cluster seed n_c

4. **(15 pts.) Complete Link Clustering**. Consider the following document by document similarity matrix for five documents a to e.

$$\begin{bmatrix} 1 & 0.2 & 0.7 & 0.5 & 0.3 \\ X & 1 & 0.6 & 0.3 & 0.4 \\ X & X & 1 & 0.5 & 0.2 \\ X & X & X & 1 & 0.6 \\ X & X & X & X & 1 \end{bmatrix}$$

a. **(10 pts.)** Obtain the corresponding complete-link dendrogram.
b. (3 pts.) Give the similarity matrix implied by the dendrogram.
c. **(2.0 pts.)** If we cut the dendogram of at the similarity level 0.65 what kind of clustering structure do we obtain? What are the contents of these clusters?



a. Dendrogram construction Use similarity values in descending order.

ac: 07, bc: 0.6, de: 0.6, ad: 0.5, cd: 0.5, be: 0.4, ae: 0.3 bd: 0.3, ce: 0.2

b. The implied similarity matrix.

$$\begin{bmatrix} 1 & 0.2 & 0.7 & 0.2 & 0.2 \\ X & 1 & 0.6 & 0.3 & 0.3 \\ X & X & 1 & 0.2 & 0.2 \\ X & X & X & 1 & 0.6 \\ X & X & X & X & 1 \end{bmatrix}$$

c. Partition: {a, c }{b }{d }{e }

APPENDIX: Cover Coefficient Formulas (just in case provided).

$$c_{ij} = \alpha_i \times \sum_{k=1}^{n} (d_{ik} \times \beta_k \times d_{jk}) \qquad c'_{ij} = \beta_i \times \sum_{k=1}^{m} (d_{ki} \times \alpha_k \times d_{kj})$$

m: No. of documents, n: No. of terms

CS433-CS533: Information Retrieval

CS433-CS533: Information Retrieval Systems, Midterm, Part 2: 45 points, March 28, 2021, 40 Min.
Note: You have to give the final value of arithmetic expressions such as write 0.5 instead of 1/2.

KEY

1. **(15 pts.) C^3M**. Please provide a short, neat and easy to understand work, I may give no credit otherwise. According to the cover coefficient concept:

a. Show that average number of documents per cluster, d_c can be approximated as δ^{-1} where δ is the average decoupling coefficient of documents.

$d_c = m / n_c = m / (m \times \delta) = \delta^{-1}$

b. Show that the average document cluster size is in the following range.

$$\max(1, m/n) \leq d_c \leq m$$

Note that $n_c = n_c'$

$\max(d_c)$ can occur for the minimum number of clusters when all documents are identical $n_c = 1$, then $dc = m$

$\min(d_c)$ can occur for the max of (n_c). $\max(n_c) = \min(m, n) \Rightarrow d_c = m / \min(m, n) = \max(1, m/n)$

2. **(15 pts.) Rand's Coefficient.** Consider a set of objects a, b, c, d, e. These objects are grouped by an human expert and a gold standard is obtained

Using a clustering algorithm the following clusters are obtained.

{a, d, e} and {b, c}

For the Rand index calculation the following observations are given using the above clustering structure for the gold standard: The number of false negatives= 2 (for the object pairs: ab, ac), the number of false positives= 3 (for the object pairs: ad, ae, de). From the given information find all possible gold standard? If you cannot obtain any gold standard state your reason.

Clustering algorithm results {a, d, e }{b, c }
F N= 2 and they are ab, ac
which implies that a and b must be in the same cluster and a and c must be in the same cluster
therefore a, b, and c must be in the same cluster.

FP= 3 and they are given as ad, ae, de which means that a and d must be in separate clusters and the same is true for a and d; and furthermore for d and e. This implies that these three documents should be in different clusters.

These observations imply the following unique gold standard: {a, b, c}{d }{e }.

3. (15 pts.) Ranked Retrieval. Consider the posting lists given for two terms below.
term-1 ==> <14, 4> <32, 3> <45, 3> <107, 7>
term-2 ==> <14, 2> <21, 1> <32, 1> <45, 5> <85, 6> <107, 6>

a. Construct the posting list structure based on groupings used by the $f_{d,t}$ (number of occurrences of terms in documents) by remembering the purpose of such organizations. Show the structure in two ways: plain (no d-gap) and with d-gap.

b. Consider *ranked* query processing for the Google-like query Q: term-1 term-2. For ranking the documents use the summation of $f_{d,t}$ values of the query terms for the documents. Use the posting list structures grouped according to $f_{d,t}$ values. During query processing use interleaved processing. Give rankings for two cases: unlimited accumulators, and with (limited) three accumulators.

a. Posting list groupings using $f_{d,t:}$ < $f_{d,t}$, No. of occurrences, Associated doc. no.s>

term-1 ==> <7, 1, 107> <4, 1, 14> <3, 2, 32, 45>

term-2 ==> <6, 2, 85, 107> <5, 1, 45> <2, 1, 14> <1, 2, 21, 32>

Posting list structures using $f_{d,t}$ and d-gaps: < $f_{d,t}$, No. of occurrences, Associated oc. no.s listed using d-gaps>

term-1 ==> <7, 1, 107> <4, 1, 14> <3, 2, 32, 13>

term-2 ==> <6, 2, 85, 22> <5, 1, 45> <2, 1, 14> <1, 2, 21, 11>

b. Ranking without accumulator restriction: Document weights. Note that interleaved processing has no impact since it gives chance to all documents that appear in the posting lists.

Doc. No.	*14*	*21*	*32*	*45*	*85*	*107*
Total weight	*4+2= 6*	*1*	*3+1= 4*	*3+5= 8*	*6*	*7+3= 13*

Ranked Output:

Rank	*1*	*2*	*3*	*3*	*4*	*5*
Doc. No.	*107*	*45*	*14*	*85*	*32*	*21*
Total weight	*13*	*8*	*6*	*6*	*4*	*1*

Ranking with accumulator restriction and interleaved processing:
Highest weight is for term-1 <7, 1, 107>: 107/7 (Doc. no./accumulated total weight so far)
Highest weight is for term-2 <6, 2, 85, 107>: 107/13, 85/6
Highest weight is for term-2 <5, 1, 45>: 107/13, 85/6, 45/5
Highest weight is for term-1 <3, 2, 32, 45>: 107/13, 85/6, 45/8, no accumulator room for doc. no/ 32
No change can be observed for <2, 1, 14> <1, 2, 21, 32> since there is no room for the documents 14, 21, 32

Ranked Output:

Rank	*1*	*2*	*3*
Doc. No.	*107*	*45*	*85*
Total weight	*13*	*8*	*6*