CS533 Information Retrieval Systems

Assignment – 1

1. Consider the following search results for two queries Q1 and Q2. The documents are ranked in the given order, the relevant documents are shown in bold.

Q1: **D1**, D2, **D3**, **D4**, D5, **D6**, D7, D8, D9, **D10**.

Q2: **D1**, D2, **D3**, D4, D5, **D6**.

For Q1 and Q2 the total number of relevant documents is, respectively, 5 and 3 documents.

a. Using the TREC interpolation rule, in a table give the precision value for the 11 standard recall levels 0.0, 0.1, 0.2, … 1.0. Please also draw the corresponding recall-precision graph as shown in the first figure of TREC-6 Appendix A (its link is available on the course web site).

Please do this for each query separately and obtain one table for both queries using the average of two values at each recall point.

b. What is the intuition behind interpolation?

c. Find R-Precision (see TREC-6 Appendix A for definition) for Query1 and Query2.

d. Find MAP for these queries.

1) In this question, we have the following queries where the relevant documents are shown in bold:
Q1: **D1,** D2, **D3, D4,** D5, **D6**, D7, D8, D9, **D10** The number of relevant documents: 5
Q2: **D1,** D2, **D3,** D4, D5, **D6** The number of relevant documents: 3

"Interpolated" means that, for example, precision at recall 0.10 (i.e., after 10% of rel docs for a query have been retrieved) is taken to be MAXIMUM of precision at all recall points >= 0.10. Values are averaged over all queries (for each of the 11 recall levels). Based on this definition and using the TREC interpolation rule given in TREC-6 Appendix A, we have the following tables for the given queries Q1 and Q2.

a)

Doc. no.	1	2	3	4	5	6	7	8	9	10
Relevance	+	-	+	+	-	+	-	-	-	+
Precision	1/1	1/2	2/3	3/4	3/5	4/6	4/7	4/8	4/9	5/10
Recall	1/5	1/5	2/5	3/5	3/5	4/5	4/5	4/5	4/5	5/5

Table 1. Recall – precision table for Q1.

Precision	1	1	1	0.75	0.75	0.75	0.75	0.66	0.66	0.5	0.5
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Table 2. Interpolated recall – precision table for Q1.

Doc. no.	1	2	3	4	5	6
Relevance	+	-	+	-	-	+
Precision	1/1	1/2	2/3	2/4	2/5	3/6
Recall	1/3	1/3	2/3	2/3	2/3	3/3

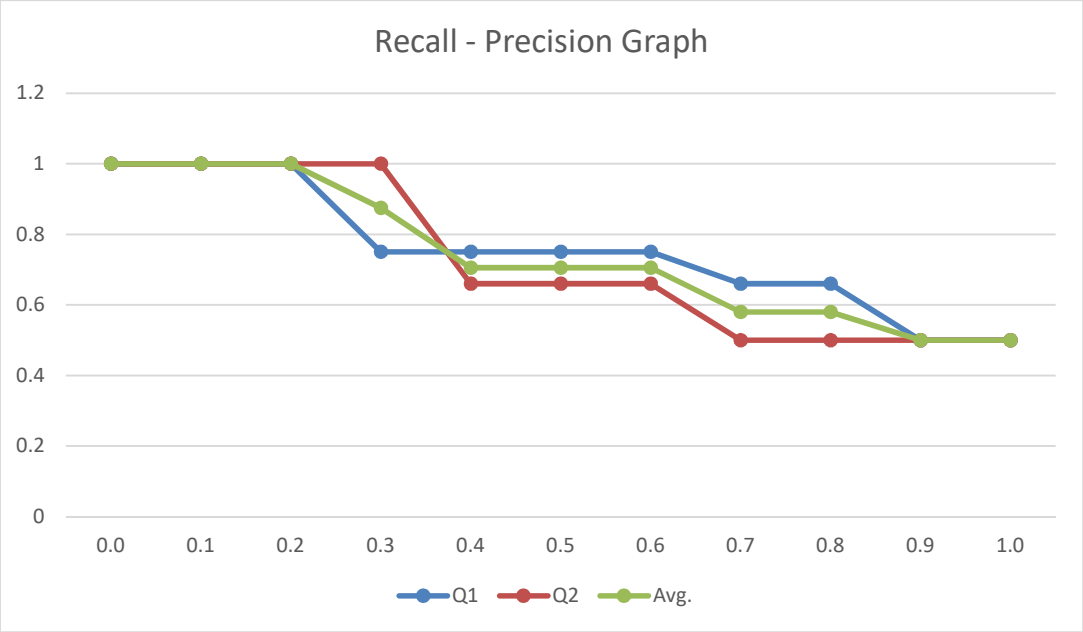
Table 3. Recall – precision table for Q2.

Precision	1	1	1	1	0.66	0.66	0.66	0.5	0.5	0.5	0.5
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Table 4. Interpolated recall – precison table for Q2.

Precision Average	1	1	1	0.875	0.705	0.705	0.705	0.58	0.58	0.5	0.5
Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0

Table 5. Average interpolated recall – precision table for both queries



b) Precision and recall are set-based measures to evaluate the quality of an unordered set of retrieved documents. In order to evaluate ranked lists, precision is plotted against recall. In case of set of queries (topics), each with a varying number of relevant documents, we need to interpolate individual query precision values to standard recall levels (0.0, 0.1, …, 1.0) so that we can observe the average performance of a system on precision recall graphs.

c) R-precision is the precision after R documents are retrieved where R is the number of relevant documents for the given query. There are 5 and 3 relevant documents for Q1 and Q2, respectively. Thus, R-Precision(Q1) = 0.6 and R-Precision(Q2) = 0.66.

d) Mean Average Precision (MAP) is given by: $MAP = \frac{\sum_{i=1}^{n} p(i) * rel(i)}{\#\ of\ relevant\ documents}$ where p(i) is precision at position i and rel(i) is relevancy of the k^{th} document that is either 0 or 1.
MAP(Q1) = (1 + 2/3 + 3/4 + 4/6 + 5/10)/5 = 0.714 and
MAP(Q2) = (1 + 2/3 + 3/6)/3 = 0.72

2. Consider the following document by term binary D matrix for m= 6 documents (rows), n= 6 terms (columns).

$$\begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Consider the problem of constructing a document by document similarity, S, matrix. How many similarity coefficients will be calculated using the following methods? For each case explain your answer briefly: give exact numbers for each document and explain how you came up with those numbers.

a. Straightforward approach (using document vectors) -the 1st method discussed in the class-.

b. Using term inverted indexes.

2) For this question we have the following D matrix: $\boldsymbol{D} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{0} \\ \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} \end{bmatrix}$

a) Similarity matrix, S is a symmetric matrix ($S_{ij} = S_{ji}$) with its all diagonal terms $S_{ii} = 1$ (i=1, …, m). Therefore, we only need to calculate elements in the upper triangle of S which has m*(m-1)/2 = 15 elements.

b) Using D matrix, we have term inverted indexes as follows:

$t_1 \rightarrow$<1,1>, <3,1> : $\{d_1, d_3\}$
$t_2 \rightarrow$<2,1>, <4,1> : $\{d_2, d_4\}$
$t_3 \rightarrow$<1,1>, <3,1> : $\{d_1, d_3\}$
$t_4 \rightarrow$<1,1>, <2,1>, <4,1> : $\{d_1, d_2, d_4\}$
$t_5 \rightarrow$<2,1>, <5,1>, <6,1> : $\{d_2, d_5, d_6\}$
$t_6 \rightarrow$<5,1>, <6,1> : $\{d_5, d_6\}$

Then, we need to consider each document and term sets that has the document at hand.

- Consider $d_1 \rightarrow t_1\ t_3\ t_4$: $\{d_1, d_3\} \cup \{d_1, d_3\} \cup \{d_1, d_2, d_4\} = \{d_1, d_2, d_3, d_4\}$: S_{12}, S_{13}, S_{14}
- Consider $d_2 \rightarrow t_2\ t_4\ t_5$: $\{d_2, d_4\} \cup \{d_1, d_2, d_4\} \cup \{d_2, d_5, d_6\} = \{d_1, d_2, d_4, d_5, d_6\}$: S_{24}, S_{25}, S_{26}
- Consider $d_3 \rightarrow t_1\ t_3$: $\{d_1, d_3\} \cup \{d_1, d_3\} = \{d_1, d_3\}$: No calculation needed.
- Consider $d_4 \rightarrow t_2\ t_4$: $\{d_2, d_4\} \cup \{d_1, d_2, d_4\} = \{d_1, d_2, d_4\}$: No calculation needed.
- Consider $d_5 \rightarrow t_5\ t_6$: $\{d_2, d_5, d_6\} \cup \{d_5, d_6\} = \{d_2, d_5, d_6\}$: S_{56}

So, we need to calclate only 7 terms.

3.	Obtain the similarity matrix S for the above D matrix (you don't need to show your intermediate steps). Use the Dice similarity coefficient. Use the S matrix to construct the dendrogram (cluster tree) structure corresponding to the single-link and complete-link clustering methodologies. Explain how to use the dendrogram structure to obtain a partitioning clustering structures.

3)

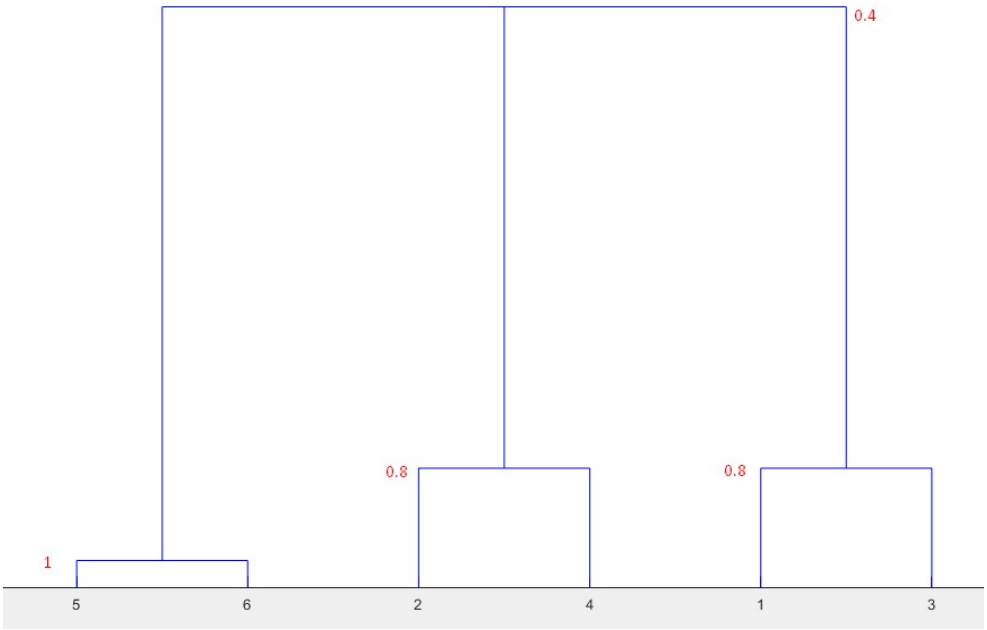
a) Dice coefficient is given by $\frac{2|X \cap Y|}{|X|+|Y|}$. After intermediate steps, we have the following similarity matrix, S.

$$S = \begin{bmatrix} 1 & 0.33 & 0.8 & 0.4 & 0 & 0 \\ x & 1 & 0 & 0.8 & 0.4 & 0.4 \\ x & x & 1 & 0 & 0 & 0 \\ x & x & x & 1 & 0 & 0 \\ x & x & x & x & 1 & 1 \\ x & x & x & x & x & 1 \end{bmatrix}$$

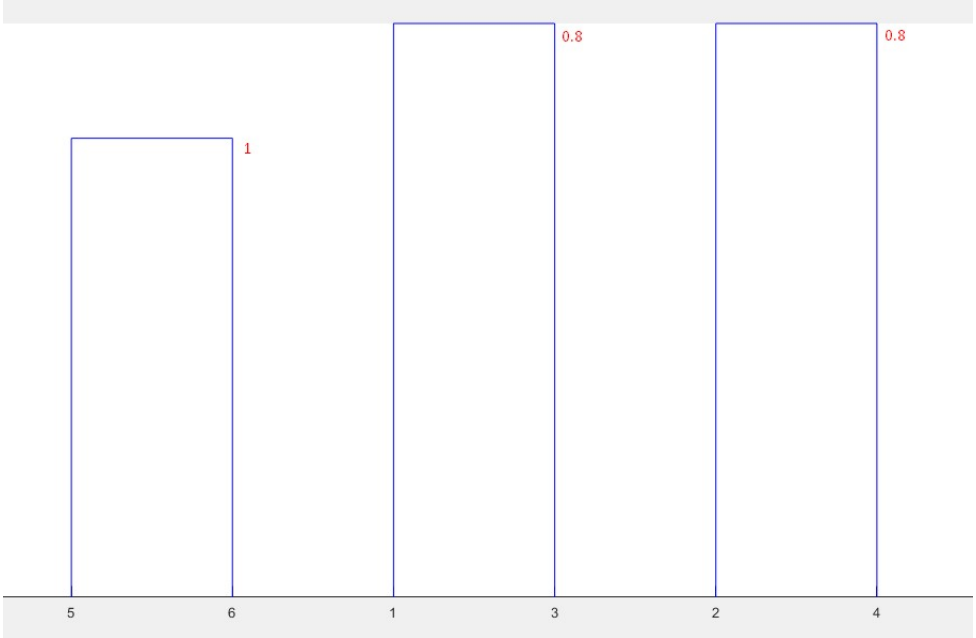
b) For the dendrogram structure, we need to sort document pair similarities first. The sorted list of document pairs in S matrix is given below.

Step	1	2	3	4	5	6	7
Pair	S_{56}	S_{24}	S_{13}	S_{14}	S_{25}	S_{26}	S_{12}
Similarity	1	0.8	0.8	0.4	0.4	0.4	0.33

So, the dendrogram using sindle-link methodology becomes:



So, the dendrogram using complete-link methodology becomes:



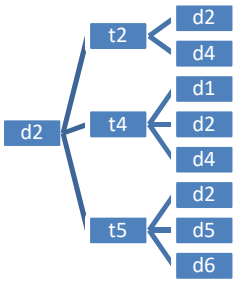
c) Documents can be distributed into clusters using a threshold in single-link and complete-link methodologies. For single-link case, the threshold value can be in the range (0.4, 0.8) so that we can distribute the documents in three clusters that are {1, 3}, {2, 4}, {5, 6}. As for the complete-link case, we do not need a threshold in this particular example since the dendrogram already indicates three different groups which are again {1, 3}, {2, 4}, {5, 6}. The complete-link did not give a "root" point to which all documents are connected as the corresponding similarity coefficients came later in the sorted list of pairs and some coefficients are zero.

d) Complete-link algorithm may generate different clusters in the case that similarity coefficients have ties and in which order these tied (equal) coeeficients are inserted to the graph. S_{24} and S_{13} are equal and also S_{14} , S_{25} , S_{26} are equal. In this case, the structure of the hierarchy of clusters depends on the order of inserting terms. A more detailed example of this problem can be found in *Clustering Methods and Algorithms, A. K. Jain, R. C. Dubes, pg. 77-78* (given on course webpage).

4. Consider the above D matrix. Cluster the documents using the cover coefficient-based clustering methodology (C^3M). Please a) Show the double-stage probability experiment tree for the second document, and show the calculation of c_{24} of the corresponding C matrix, b) obtain the C matrix (you do not need to show the intermediate steps), c) find the number of clusters implied by the C matrix – explain how-, d) find the cluster seeds, e) obtain the IISD (inverted index for seed documents), f) obtain the clusters and explain how you them.

4)

a) d_2 has t_2, t_4, t_5. The documents that have t_2 are $\{d_2, d_4\}$. The documents that have t_4 are $\{d_1, d_2, d_4\}$. The documents that have t_5 are $\{d_2, d_5, d_6\}$. Given this information, we can construct double-stage probability experiment tree as follows:



According to this tree let's calculate C_{24}:

C_{24} = 1/3*1/2 + 1/3*1/3 = 0.2778

b) Following the same approach we can obtain the C matrix as follows:

$$C = \begin{bmatrix} 0.444 & 0.111 & 0.334 & 0.111 & 0 & 0 \\ 0.111 & 0.389 & 0 & 0.278 & 0.111 & 0.111 \\ 0.5 & 0 & 0.5 & 0 & 0 & 0 \\ 0.166 & 0.417 & 0 & 0.417 & 0 & 0 \\ 0 & 0.166 & 0 & 0 & 0.417 & 0.417 \\ 0 & 0.166 & 0 & 0 & 0.417 & 0.417 \end{bmatrix}$$

c) Number of clusters, n_c is given by $n_c = \sum_{i=1}^{m} C_{ii}$ = 0.444 + 0.389 + 0.5 + 0.417 + 0.417 + 0.417 = 2.5840 ≈ 3.

d) Seed power of document d_i is given by $P_i = C_{ii}(1 - C_{ii})X_{di}$

$P_1 = C_{11}(1 - C_{11})X_{d1}$= 0.444*(1-0.444)*3 = 0.7406
$P_2 = C_{22}(1 - C_{22})X_{d2}$= 0.389*(1-0.389)*3 = 0.7103
$P_3 = C_{33}(1 - C_{33})X_{d3}$= 0.5*(1-0.5)*2 = 0.5
$P_4 = C_{44}(1 - C_{44})X_{d4}$= 0.417*(1-0.417)*2 = 0.4862
$P_5 = C_{55}(1 - C_{55})X_{d5}$= 0.417*(1-0.417)*2 = 0.4862
$P_6 = C_{66}(1 - C_{66})X_{d6}$= 0.417*(1-0.417)*2 = 0.4862

$C_{65} = C_{56} = C_{55} = C_{66}$ equality means d_5 and d_6 are identical. So, we can define d_1 and d_2 as cluster seeds since the first document covers the terms t_1, t_3, t_4 and the second document covers the terms t_2, t_4, t_5.

e) Inverted indexes for cluster seeds are given below:

t_1 →<1, 1>, <3, 1>: {d_1, d_3}
t_2 →<2, 1>, <4, 1>: {d_2, d_4}
t_3 →<1, 1>, <3, 1>: {d_1, d_3}
t_4 →<1, 1>, <2, 1>, <4, 1>: {d_1, d_2, d_4}
t_5 →<2, 1>, <5, 1>, <6, 1>: {d_2, d_5, d_6}

f) Cluster 1 → d_1, d_3 ($C_{31} > C_{32}$)
Cluster 2 →d_2, d_4 d_5, d_6 ($C_{42} > C_{41}$, $C_{52} > C_{51}$, $C_{62} > C_{61}$)

5.	Anomaly or outlier detection is a data mining problem. See the related 2009 paper in the *ACM Computing Surveys* with the title Anomaly Detection: A Survey. Skim the paper to understand the problem. Read the section on cluster-based anomaly detection. How can we use a clustering algorith to detect outliers within the context of a document collection. Define a method based on teh clustering algorithms we studied in the classroom. Assume that you have a static document collection.

5) Anomaly or outlier detection is an important data mining problem that has been confronted in various research areas. Anomalies, as the authors of this paper define, are patterns in data that do not follow the expected behavior. Anomaly detection is the problem of finding these patterns

in a given dataset. Anomalies may often indicate significant and critical information in many domains such as an anomalous traffic pattern in a network meaning a hacked computer, anomalies in MRI systems meaning false diagnoses. This paper discusses many anomaly detection techniques one of which is clustering-based anomaly detection. The clustering-based algorithms are divided into three categories. The first group has the assumption that normal data instances belong to a cluster whereas anomalies do not. In the context of document collection, we can implement this methodology by first clustering the retrieved document based on terms and define the documents that are non-seeds (small seed power) and do not have any common terms with seed documents. However, this approach might lead to mistake potential seeds as anomalies. The second group of detection algorithms assumes that normal data instances lie close to their closest cluster centroid, while anomalies fall far away from clusters. Following this approach, we can implement single-link clustering, K-means Clustering, Self-Organizing Maps (SOM) etc. to find clusters and calculate the distance between each distance and cluster centroids. Although this method may perform well for individual anomalies, it may mislead in the cases where anomalies form a clusters themselves. The last group has the assumption that normal data instances belong to large and dense clusters while anomalies belong to small or sparse clusters. Fixed width clustering algorithm follows this approach. An instance is assigned to a cluster whose centroid is within the predefined distance threshold to that instance. If no such cluster exists, then the instance initiates its own cluster. Afterwards, anomalies are detected based on their cluster density and sizes. This can be implemented through using similarity matrix coefficients as distances between seeds and non-seeds or we can use C matrix as distance measures.

6. In this part consider the paper J. Zobel, A. Moffat, "Inverted files for text search engines." *ACM Computing Surveys*, Vol. 38, No. 2, 2006.

a. Understand the skipping concept as applied to the inverted index construction.

Assume that we have the following posting list for term a: <1, 2> <3, 1> <9, 5> <10, 3> <12, 4> <17, 4> <18, 3>, <22, 2> <24, 2> <33, 4> <38, 5> <43, 5> <55, 3><64, 2> <68, 4> <72, 5> <75, 5> <88, 2>.. The posting list indicates that term-a appears in d1 twice and in d3 once, etc.

Assume that we have the following posting list for term-b: <1, 2> <12, 2> <66, 1>.

Consider the following conjunctive Boolean query: term-a **and** term-b. If no skipping is used how many comparisons do you have to find the intersection of these two lists?

Introduce a skip structure, draw the corresponding figure then give the number of comparisons involved to process the same query.

State the advantages and disadvantages of large and small skips in the posting lists. Note that in the paper it is assumed that compression will be used. The skip idea is applicable in an uncompressed environment too.

b. Give a posting list of of term-a (above it is given in standard sorted by document number order) in the following forms: 1), a) ordered by $f_{d,t}$, b) ordered by frequency information in prefix form. What are the advantages of the approaches a and b? Do they have any practical value?

6)

a) In this question, we are given the following posting lists for term-a and term-b:
term-a: <1, 2>, <3, 1>, <9, 5>, <10, 3>, <12, 4>, <17, 4> <18, 3>, <22, 2>, <24, 2>, <33,4>, <38, 5>, <43, 5>, <55, 3>, <64, 2>, <68, 4>, <72, 5>, <75,5>, <88, 2>.
term-b: <1, 2>, <12, 2>, <66, 1>.

If no skipping is used, we need to iterate through both lists comparing each element. So,

- For <1, 2> in term-b: compare <1, 2> → 1 comparison and we increment both term-a and term-b index (say x, y).
- For <12, 2> in term-b: compare <3, 1>, <9, 5>, <10, 3>, <12, 4> → 4 comparisons and we increment both x and y.
- For <66, 1> in term-b: compare <17, 4> <18, 3>, <22, 2>, <24, 2>, <33,4>, <38, 5>, <43, 5>, <55, 3>, <64, 2>, <68, 4> → 10 comparisons and we reach the end of the list.

So, we have to make 15 comparisons to find the intersection of these two lists without skipping.

Let's use a skipping structure where each chunk is size of 5 documents and the last chunk is of modulo 5 documents. The chunks are:

- Chunk 1: <1, 2>, <3, 1>, <9, 5>, <10, 3>, <12, 4>
- Chunk 2: <17, 4>, <18, 3>, <22, 2>, <24, 2>, <33,4>
- Chunk 3: <38, 5>, <43, 5>, <55, 3>, <64, 2>, <68, 4>
- Chunk 4: <72, 5>, <75,5>, <88, 2>

We continue with comparing the terms.

- For <1, 2> in term-b: Is it in Chunk 1? True → 1 comparison, 1 comparison to locate.
- For <12, 2> in term-b: Is it in Chunk 1? True → 1 comparison, 5 comparisons to locate.
- For <66, 1> in term-b: Is it in Chunk 1? False → 1 comparison. Chunk 2? False → 1 comparison. Chunk 3? True (in terms of range) → 1 comparison. 5 comparisons to iterate. In total we have 16 comparisons. 11 of them are term by term comparisons.

	Small Skips	Large Skips
Advantages	More chunks can be skipped Less comparisons within the chunk since number of docs in the chunk decreases	The total number of chunks decreases. The number of comparisons with chunk descriptors decreases
Disadvantages	The number of comparisons with the chunk descriptor increases.	The number of comparisons within the chunk increases.

b) Ordered by $f_{d,t}$:

term-a: <9, 5>, <38,5>, <43, 5>, <75, 5>, <12, 4>, <17, 4>, <68, 4>, <10, 3>, <18, 3>, <55, 3>, <1, 2>, <22, 2>, <24, 2>, <64, 2>, <88, 2>, <3, 1>.

Ordered by frequency information in prefix form:

term-a: <5: 3: 9, 38, 43>, <4: 3: 12, 17, 68>, <3: 3: 10, 18, 55>, <2: 5: 1, 22, 24, 64, 88>, <1: 1: 3>.

Ordering the terms by frequency increases performance when a frequency threshold is used as some portion of the documents can be disregarded (reduces disk I/O). The same advantage holds for the prefix form as well. The prefix form has room for improvement in

compression by taking differences between documents that have the same term frequency which leads us to further improvement in storage size of these lists. However, prefix construction is costly.

9. What are the components of an information retrieval test collection? Explain the pooling approach? Please read the paper by Zobel (How Reliable Are the Results of Large-Scale Information Retrieval Experiments?) and give some reflections of his criticism of this approach.

9)

The components of an information retrieval test collection are a set of documents, a set of queries and relevance information about each document with respect to each query. The pooling approach is an information retrieval system evaluation methodology where the top p documents are "pooled", i.e., gathered to eliminate duplicates and to get rid of any association between document and the IRS. "p" is defined as the pool depth.

In his paper, Zobel criticizes different aspects of the pooling approach. He states that it is necessary to use pooling with collections like TREC collection to identify documents for relativity assessment but pooling may introduce bias when the pool depth is fixed or recall estimates are unreliable. He continues that TREC results are reliable and pooling does not introduce a significant bias and the relevance judgements obtained provide a fair basis for new systems although it is quite probable that 50 – 70% of the relevant documents are discovered at best because of the queries with large amount of answers and measures on recall can be uncertain. He also points out that assuming unjudged documents to be irrelevant to the corresponding query is questionable. Zobel discusses the effect of choosing pool depth and mentions a possible disadvantage of having a measurement depth greater than the pool depth that might lead to similar systems reinforcing each other and misjudgment of novel retrieval systems. This difference between the measurement depth and the pool depth also introduces uncertainty.