Computer Engineering Department
Bilkent University

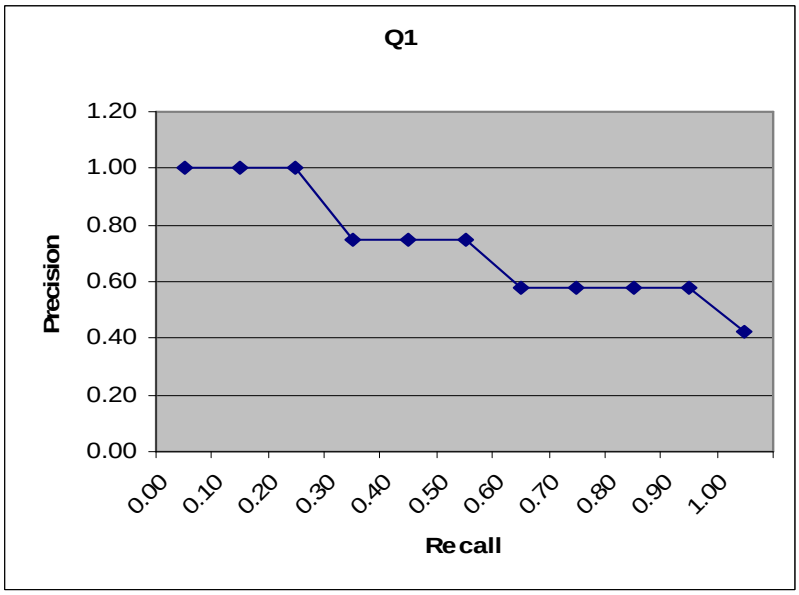
CS533: **Information Retrieval Systems HW # 1**

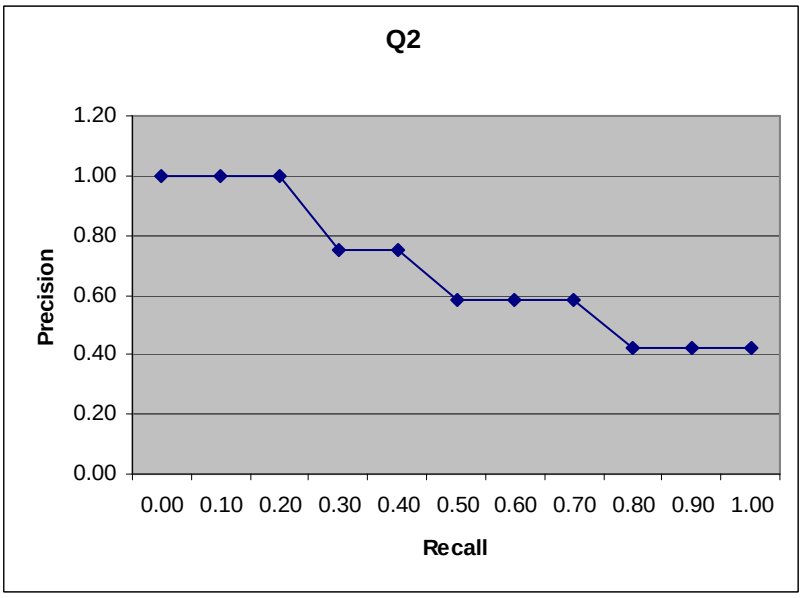
Hidayet AKSU
20007350

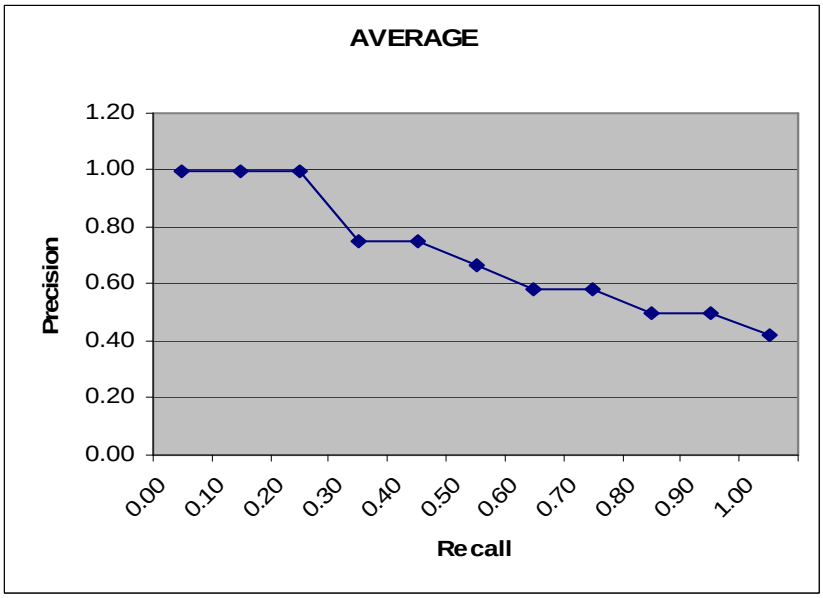
1. a.

		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	Relevant	Y	N	Y	N	Y	N	N	N	N	N
	Precision	1.00	0.50	0.67	0.50	0.60	0.50	0.43	0.38	0.33	0.30
Q1	**Recall**	0.33	0.33	0.67	0.67	1.00	1.00	1.00	1.00	1.00	1.00
	0.00	0.10	0.20	**0.30**	0.40	0.50	**0.60**	0.70	0.80	**0.90**	1.00
R-Q1	**1.00**	1.00	1.00	0.75	0.75	0.75	0.58	0.58	0.58	0.58	0.42

		D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
	Relevant	Y	N	Y	N	Y	N	N	N	N	N
	Precision	1.00	0.50	0.67	0.50	0.60	0.50	0.43	0.38	0.33	0.30
Q2	**Recall**	0.25	0.25	0.50	0.50	0.75	0.75	0.75	0.75	0.75	0.75
	0.00	0.10	0.20	**0.30**	0.40	0.50	**0.60**	0.70	0.80	**0.90**	1.00
R-Q2	**1.00**	1.00	1.00	0.75	0.75	0.58	0.58	0.58	0.42	0.42	0.42
	0.00	0.10	0.20	**0.30**	0.40	0.50	**0.60**	0.70	0.80	**0.90**	1.00
R-AVERAGE	**1.00**	**1.00**	**1.00**	**0.75**	**0.75**	**0.67**	**0.58**	**0.58**	**0.50**	**0.50**	**0.42**


Q1
Precision
1.20
1.00
0.80
0.60
0.40
0.20
0.00
0.00 0.10 0.20 0.30 0.40 0.50 0.60 0.70 0.80 0.90 1.00
Recall


Q2
Precision
1.20
1.00
0.80
0.60
0.40
0.20
0.00
0.00 0.10 0.20 0.30 0.40 0.50 0.60 0.70 0.80 0.90 1.00
Recall


AVERAGE
Precision
1.20
1.00
0.80
0.60
0.40
0.20
0.00
0.00 0.10 0.20 0.30 0.40 0.50 0.60 0.70 0.80 0.90 1.00
Recall

1-b
#relevant document for Q1 is 3 and for Q2 is 4 so,

	Q1	Q2
R-Precision	2/3	2/4

1-c
Q1- Average Precision = (1.00 + 0.67 + 0.60)/3= 0.76
Q2- Average Precision = (1.00 + 0.67 + 0.60 + 0)/4= 0.57
Average of these values is the MAP, which is (0.76 + 0.57)/2= 0.66

1-d

Binary preference (bpref)

$$bpref = \frac{1}{R}\sum_{r}\left(1-\frac{|n \text{ ranked higher than } r|}{\min(R,N)}\right)$$

R: # of judged relevant documents
N: # of judged irrelevant documents
r: relevant retrieved document
n: one of the first R irrelevant retrieved documents

For Query 1:

R = 3, N = 7,

Bpref =

$$\frac{1}{3}\sum_{r}\left(1-\frac{|n \text{ ranked higher than } r|}{3}\right)=\frac{2}{3}$$

For Query 2:

R = 3, N = 7,

Bpref =

$$\frac{1}{3}\sum_{r}\left(1-\frac{|n \text{ ranked higher than } r|}{3}\right)=\frac{2}{3}$$

Geometric mean (gm_ap)

$gm = \sqrt[n]{\prod_{i=1}^{n} a_i}$ Where, a is average precision.

2-a

a. Straightforward approach (using document vectors)

To calculate the similarity, we can use the follow algorithm:

```
for i=1 to m-1
        for j=i+1 to m
                calculate sij;
        end for
end for
```

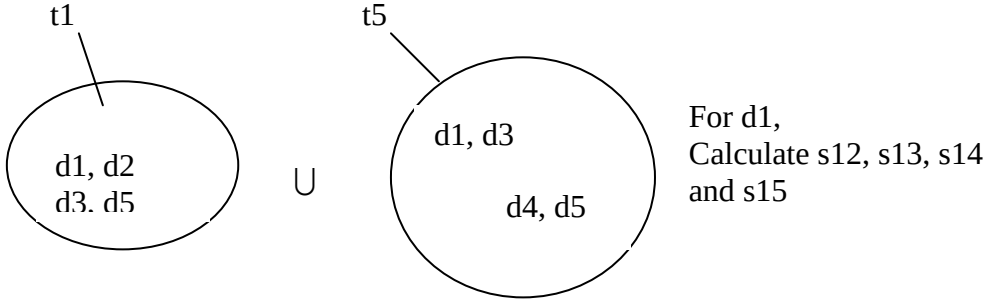
here we have m=7.
the total number of similarity values to be calculated is:
sum = $m(m-1)/2 = 7(7-1)/2 = 7*6/2=21$

b. Term inverted indexes:

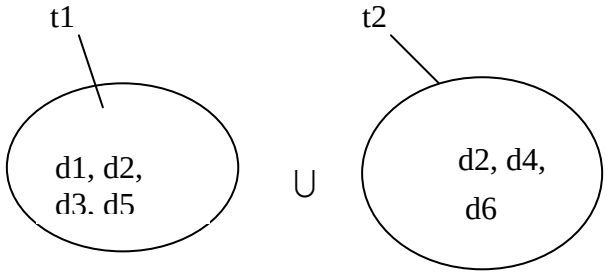
t(1) -> {<1,1>, <2,1>, <3,1>, <5,1>}
t(2) -> {<2,1>, <4,1>, <6,1>}
t(3) -> {<7,1>}
t(4) -> {<4,1>}
t(5) -> {<1,1>, <3,1>, <4,1>, <5,1>}
t(6) -> {<7,1>}
t(7) -> {<6,1>,<7,1>}

d1 -> {<1,1>, <5,1>}
d2 -> {<1,1>, <2,1>}
d3 -> {<1,1>, <5,1>}
d4 -> {<2,1>, <4,1>, <5,1>}
d5 -> {<1,1>, <5,1>}
d6 -> {<2,1>, <7,1>}
d7 -> {<1,1>, <6,1>, <7,1>}

Consider d1:
d1 contains t1 & t5.

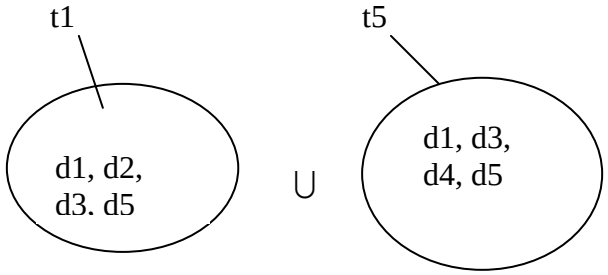


Consider d2:
It contains t1 and t2.



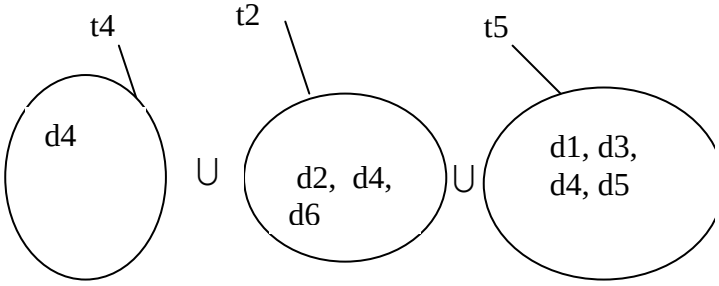
For d2,
Calculate s21, s23, s24, s25 and s26. But since s21 = s12, no need for recalculation

Consider d3:
It contains t1 and t5.



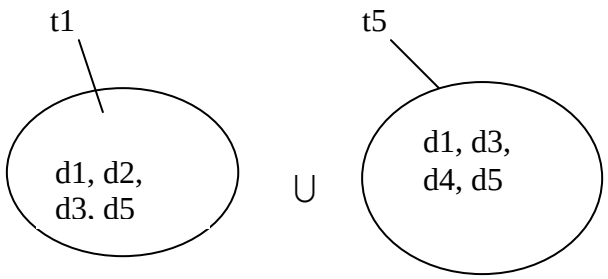
For d3,
Calculate s31, s32, s34, and s35. But since s31 = s13, and s32=s23 no need for recalculation

Consider d4:
It contains t2, t4 and t5.



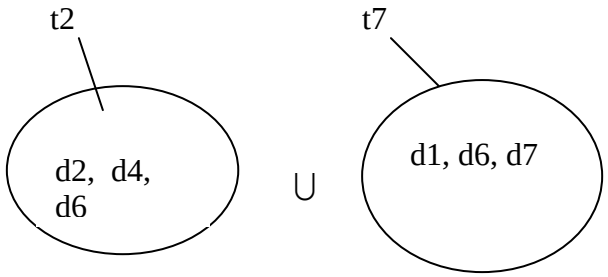
For d4,
Calculate s42, s43, s45, and s46. But since s42 = s24, and s43=s34 no need for recalculation

Consider d5:
It contains t1 and t5.



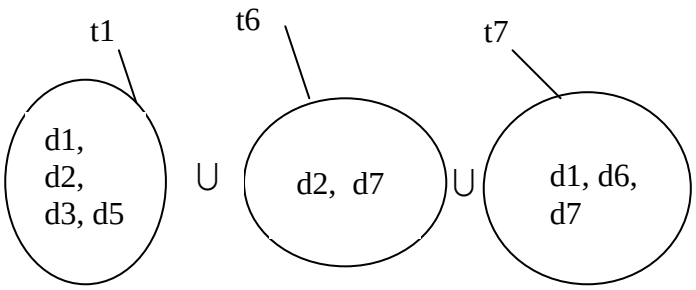
For d5,
Calculate s51, s52, s53, and s54. But since s51 = s15, s52 = s25, s53 = s35, and s54=s45 no need for recalculation

Consider d6:
It contains t2 and t7.



For d6,
Calculate s61, s62, s64, and s67. But since s61 = s16, s62 = s26, s64 = s46 no need for recalculation

Consider d7:
It contains t1, t6 and t7.



For d7,
no new s value, so, no need for recalculation

Then, we need to compute
Size of { s12, s13, s14, s15, s23, s24, s25, s26, s34, s35, s45, s46, s67 } = 13

c. First , we need the inverted index for the terms, which are:

t(1) -> {<1,1>, <2,1>, <3,1>, <5,1>}
t(2) -> {<2,1>, <4,1>, <6,1>}
t(3) -> {<7,1>}
t(4) -> {<4,1>}
t(5) -> {<1,1>, <3,1>, <4,1>, <5,1>}
t(6) -> {<7,1>}
t(7) -> {<6,1>,<7,1>}

Jaccard Similarity Coefficient: $|x \cap y| / (|x| + |y| - | x \cap y|)$

Analyze the similarity arrays one document at a time:

Consider d1:

X	0	0	0	0	0	0

d1 contains t1 and t5.

After process t1 and t5, use Jaccard Similarity Coefficient to calculate similarity values:

X	1/3	1	1/4	1	0	0

s_{12}=1/(2+2-1) =1/3 s_{13}=2/(2+2-2) =1 s_{14}=1/(2+3-1) =1/4 s_{15}=2/(2+2-2) =1

Consider d2:

X	X	0	0	0	0	0

d2 contains t1 and t2.

After process t1 and t2, use Jaccard Similarity Coefficient to calculate similarity values:

X	X	1/3	1/4	1/3	1/3	0

s_{23}=1/(2+2-1) =1/3 s_{24}=1/(2+3-1) =1/4 s_{25}=1/(2+2-1) =1/3 s_{26}=1/(2+2-1) =1/3

Consider d3:

X	X	X	0	0	0	0

d3 contains t1 and t5.

After process t1 and t5, use Jaccard Similarity Coefficient to calculate similarity values:

X	X	X	1/4	1	0	0

s_{34}=1/(2+3-1) =1/4 s_{35}=2/(2+2-2) =1

Consider d4:

X	X	X	X	0	0	0

d4 contains t2, t4 and t5.

After process t2, t4 and t5, use Jaccard Similarity Coefficient to calculate similarity values:

X	X	X	X	1/4	1/4	0

s_{45}=1/(2+3-1) =1/4 s_{46}=1/(2+3-1) =1/4

Consider d5:

X	X	X	X	X	0	0

d4 contains t1 and t5.

After process t1 and t5, use Jaccard Similarity Coefficient to calculate similarity values:

X	X	X	X	X	0	0

Consider d6:

X	X	X	X	X	X	0

D6 contains t2 and t7.

After process t2 and t7, use Jaccard Similarity Coefficient to calculate similarity values:

X	X	X	X	X	X	1/4

s_{67}=1/(2+3-1)
=1/4

No need to compute similarity coef. For d7.

3.

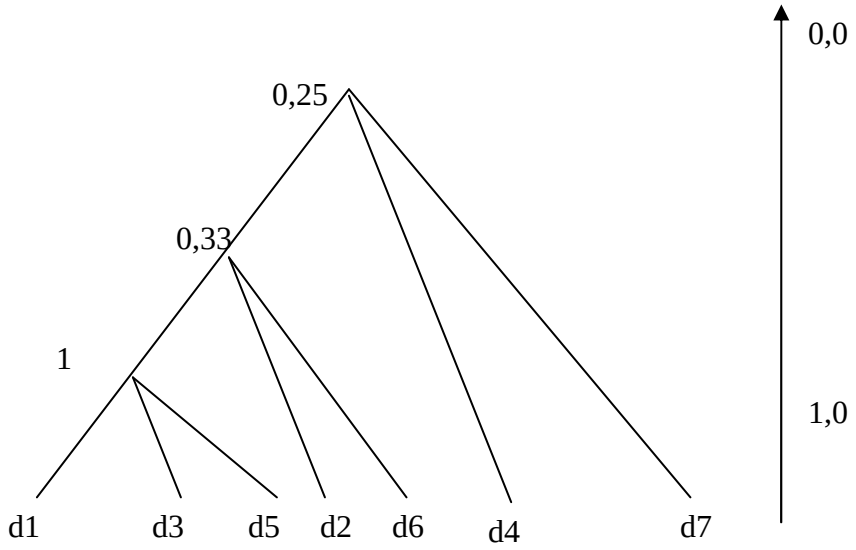
Using computed similarity coefficient from 2-c, we get the S matrix is as follows:

$$S = \begin{bmatrix} 1 & 0.33 & 1 & 0.25 & 1 & 0 & 0 \\ 0.33 & 1 & 0.33 & 0.25 & 0.33 & 0.33 & 0 \\ 1 & 0.33 & 1 & 0.25 & 1 & 0 & 0 \\ 0.25 & 0.25 & 0.25 & 1 & 0.25 & 0.25 & 0 \\ 1 & 0.33 & 1 & 0.25 & 1 & 0 & 0 \\ 0 & 0.33 & 0 & 0.25 & 0 & 1 & 0.25 \\ 0 & 0 & 0 & 0 & 0 & 0.25 & 1 \end{bmatrix}$$

single link:

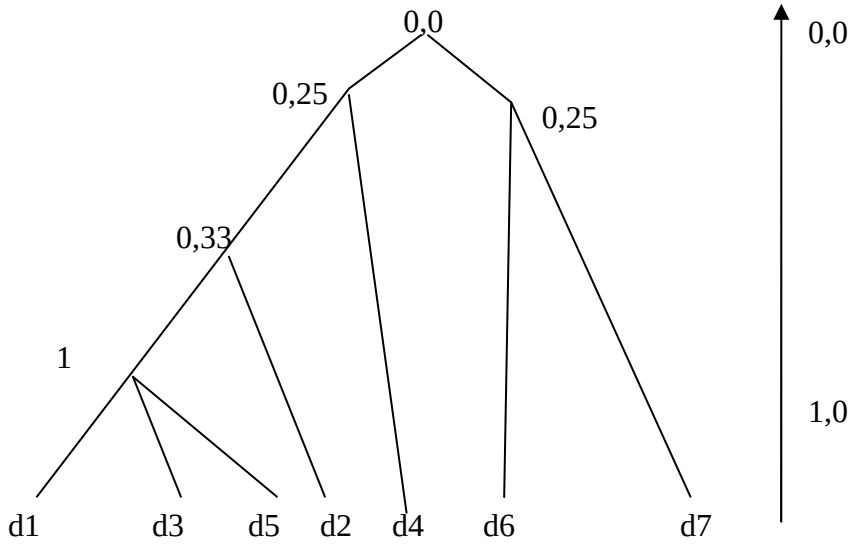
Step	Pair	Similarity
1	d1d3	1
2	d1d5	1
3	d3d5	1
4	d1d2	0,33
5	d2d3	0,33
6	d2d5	0,33
7	d2d6	0,33
8	d1d4	0,25
9	d2d4	0,25
10	d3d4	0,25
11	d4d5	0,25
12	d4d6	0,25
13	d6d7	0,25

dendrogram structure corresponding to the single-link



complete link:

Step	Pair	Similarity	
1	d1d3	1	Connect d1d3 at 1
2	d1d5	1	cannot connect yet, we do not know similarity between d3 and d5
3	d3d5	1	Connect d3d5 at 1
4	d1d2	0,33	cannot connect yet, we do not know s23 and s25
5	d2d3	0,33	cannot connect yet, we do not know s25
6	d2d5	0,33	Connect d2d5 at 0,33
7	d2d6	0,33	cannot connect yet, we do not know s61,s63,s65
8	d1d4	0,25	cannot connect yet, we do not know s42,s43,s45
9	d2d4	0,25	cannot connect yet, we do not know s43,s45
10	d3d4	0,25	cannot connect yet, we do not know s45
11	d4d5	0,25	Connect d4d5 at 0,25
12	d4d6	0,25	cannot connect yet, we do not know s61,s63,s65
13	d6d7	0,25	Connect d6d7 at 0,25



4)
wij = tfij × idfi

individual components:

tfij is the term frequency of occurrence inside that document

idfi is *inverse function of the number of documents in the collection to which it is assigned. The idf factor can be computed as 1 divided by the document frequency*

wij is the specificity of a given term as applied to a given document

The inverse document frequency of a term can be obtained in advance from a collection analysis. by use of posting list of the index, the document frequencies, and its inverse, can be computed. So, we can incorporate the idf component to indexing on the fly.

5 -a)
term a: <1, 2> <3, 1> <8, 3> <10, 2> <12, 3> <17, 4> <18, 3>, <22, 2> <24, 2> <33, 4> <38, 5> <43, 5> <55, 3>.

term-b: <25, 2> <57, 1>.

if no skip used: term-a **and** term-b = p.list (a) <intersect> p.list(b)
= {<1, 2> <3, 1> <8, 3> <10, 2> <12, 3> <17, 4> <18, 3>, <22, 2> <24, 2> <33, 4> <38, 5> <43, 5> <55, 3>} <intersect> {<25, 2> <57, 1>}
Then for each post in each list, we neeed to search other list, then 2x13=26 comparisons we have to do 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.

I introduce a skip structure such that I divide the positing list into chunks. Each chunk contains at most five posts. Chunks are ordered by its first posts document id in ascending order while Posts are ordered by documnt id in ascending order.

Then the search is given by:

```
Search post-x in posting-list(Y)
        For each chunk y of Y
        if x. document _id > y.fist_element. document_id  then
            skip chunk y
        else
                search only chunk y and then return
```

Then the term-a:

Chunk 1	<12, 3>	<10, 2>	<8, 3>	<3, 1>	<1, 2>

Chunk 2	<33, 4>	<24, 2>	<22, 2>	<18, 3>	<17, 4>
Chunk 3	<55, 3>	<43, 5>	<38, 5>		

Then the term-b:

Chunk 1	<57, 1>	<25, 2>			

Then let process the same query:
Query is term-a **and** term-b = p.list (a) <intersect> p.list(b)

Since p.list(b) contains less elements, we start with it.

For <57,1>:

1) compare with post in Chunk 1, 57 > 12, skip Chunk 1
2) compare with post in Chunk 2, 57 > 33, skip Chunk 2
3) compare with post in Chunk 3, 57 > 55, skip Chunk 3

For <25,2>:

1) compare with post in Chunk 1, 25 > 12, skip Chunk 1
2) compare with post in Chunk 2, 25 < 33
3) compare with next post in Chunk 2, 25 > 24, return not found.

Then 3+3=6 comparisons are performed in my skip structure while 26 comparisond are performed when no skip used.

if we use large skips then we will have less chunks. However, this time each chunk size will be longer, so searchs in chunks will cost more. Similarly, if we use small skips then we will have more chunks which are longer.

Searchs are done in two parts,

First-Find chunk: for this in worst case p.list size / chunk size comparisons are processsed.

Second-search chunk: for this in worst case chunk size comparisons are processsed.

Then totally [N = (p.list size / chunk size) + chunk size] comparisons are performed.
İn order to minimize comparisons, we can choose chunk size = sqrt(p.list size), then in worst case N = 2* sqrt(p.list size) comparisons are performed.

5 -b)

i) ordered by fd,t,

term a: <38, 5> <43, 5> <17, 4> <33, 4> <8, 3> <12, 3> <18, 3> <55, 3> <1, 2> <10, 2> <22, 2> <24, 2> <3, 1>.

i) ordered by frequency information in prefix form,

term a: <5:2: 38, 43> <4:2: 17, 33> <3:4: 8, 12, 18, 55> <2:4:1, 10, 22, 24> <1:1: 3>.

Repeated frequencies $f_{d,t}$ are not stored, giving a considerable saving. But within each equal-frequency segment of the list, the d-gaps are now on average larger when the document identifiers are sorted, and so the document number part of each pointer increases in cost. In combination, these two effects typically result in frequency-sorted indexes that are slightly smaller than document-sorted indexes.

Using a frequency-sorted index, a simple query evaluation algorithm is to fetch each list in turn, processing $<d, f_{d,t}>$ values only while $w_{q,t} \times w_{d,t} \geq S$, where S is the threshold. If disk reads are performed one disk block at a time rather than on a wholeof-inverted-list basis, this strategy significantly reduces the volume of index data to be fetched without degrading effectiveness.

A practical alternative is for the first disk block of each list to be used to hold the $<d,f_{d,t}>$ values with high $f_{d,t}$. These important blocks could then all be processed before the remainder of any lists, ensuring that all terms are given the opportunity to create accumulators. The remainder of the list is then stored in document order.

6

a. In supervised classification, we are provided with a collection of *labeled* (preclassified) patterns; the problem is to label a newly encountered, yet unlabeled, pattern. Typically, the given labeled (*training*) patterns are used to learn the descriptions of classes which in turn are used to label a new pattern. In the case of clustering, the problem is to group a given collection of unlabeled patterns into meaningful clusters. In a sense, labels are associated with clusters also, but these category labels are *data driven*; that is, they are obtained solely from the data.

I would you have The convergent k-means algorithm and its ANN equivalent in Information Filtering.

The reasons behind this choose are:

(1) Its time complexity is O(nkl), where n is the number of patterns, k is the number of clusters, and l is the number of iterations taken by the algorithm to converge. Typically, k and l are fixed in advance and so the algorithm has linear time complexity in the size of the data set .

(2) Its space complexity is O(k+n). It requires additional space to store the data matrix. It is possible to store the data matrix in a secondary memory and access each pattern based on need. However, this scheme requires a huge access time because of the iterative nature of the algorithm, and as a consequence processing time increases enormously.

(3) It is order-independent; for a given initial seed set of cluster centers, it generates the same partition of the data irrespective of the order in which the patterns are presented to the algorithm.

b.

Typical pattern clustering activity involves the following steps [Jain and Dubes 1988]:
(1) pattern representation (optionally including feature extraction and/or selection),
(2) definition of a pattern proximity measure appropriate to the data domain,
(3) clustering or grouping,
(4) data abstraction (if needed), and
(5) assessment of output (if needed).

Pattern representation refers to the number of classes, the number of available patterns, and the number, type, and scale of the features available to the clustering algorithm. Some of this information may not be controllable by the practitioner. *Feature selection* is the process of identifying the most effective subset of the original features to use in clustering. *Feature extraction* is the use of one or more transformations of the input features to produce new salient features. Either or both of these techniques can be used to obtain an appropriate set of features to use in clustering.

Pattern proximity is usually measured by a distance function defined on pairs of patterns. A variety of distance measures are in use in the various communities. A simple distance measure like Euclidean distance can often be used to reflect dissimilarity between two patterns, whereas other similarity measures can be used to characterize the conceptual similarity between patterns.

The *grouping* step can be performed in a number of ways. The output clustering (or clusterings) can be hard (a partition of the data into groups) or fuzzy (where each pattern has a

v ariabledegree of membership in each of the output clusters). Hierarchical clustering algorithms produce a nested series of partitions based on a criterion for merging or splitting clusters based on similarity. Partitional clustering algorithms identify the partition that optimizes (usually locally) a clustering c riterion.Additional techniques for the grouping operation include probabilistic and graph-theoretic clustering methods.
Data abstraction is the process of extracting a simple and compact representation of a data set. Here, simplicity is either from the perspective of automatic analysis (so that a machine can perform further processing efficiently) or it is human-oriented (so that the representation obtained is easy to comprehend and intuitively appealing). In the clustering context, a typical data abstraction is a compact description of each cluster, usually in terms of cluster prototypes or representative patterns such as the centroid.
Assessment of output: How is the output of a clustering algorithm evaluated? What characterizes a 'good' clustering result and a 'poor' one? All clustering algorithms will, when presented with data, produce clusters regardless of whether the data contain clusters or not.

c.
the purpose of "cluster tendency" analysis is to exemine the input data to see if there is any merit
to a cluster analysis prior to one being performed

d.
The k-means is the simplest and most commonly used algorithm employing a squared error criterion. It starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers until a convergence criterion is met (e.g., there is no reassignment of any pattern from one cluster to another, or the squared error ceases to decrease significantly after some number of iterations).

k-Means Clustering Algorithm:
(1) Choose k cluster centers to coincide with k randomly-chosen patterns or k randomly defined points inside the hypervolume containing the pattern set.
(2) Assign each pattern to the closest cluster center.
(3) Recompute the cluster centers using the current cluster memberships.
(4) If a convergence criterion is not met, go to step 2. Typical convergence criteria are: no (or minimal) reassignment of patterns to new cluster centers, or minimal decrease in squared error.

e.

An agglomerative approach begins with each pattern in a distinct (singleton) cluster, and successively merges clusters together until a stopping criterion is satisfied. A divisive method begins with all patterns in a single cluster and performs splitting until a stopping criterion is met.

What we do in the single-link and complete-link clustering methods we studied in class is to begins with each pattern in a distinct cluster, and successively merges clusters together until a stopping criterion is satisfied. So, the single-link and complete-link clustering methods we studied in class are agglomerative.

f.

What are the applications areas of clustering? List all of them. Explain two areas other than IR with
one or two paragraphs. (That is explain how clustering is being used in these areas.)

applications areas of clustering are:
(1) image segmentation

> The segmentation of the image(s)
> presented to an image analysis system
> is critically dependent on the scene to
> be sensed, the imaging geometry, configuration,
> and sensor used to transduce
> the scene into a digital image, and ultimately
> the desired output (goal) of the
> system.

(2) object and character recognition,

> The use of clustering to group views of 3D objects
> for the purposes of object recognition in
> range data was described in Dorai and
> Jain [1995]. The term *view* refers to a
> range image of an unoccluded object
> obtained from any arbitrary viewpoint.
> The system under consideration employed
> a *viewpoint dependent* (or viewcentered)
> approach to the object recognition
> problem; each object to be
> recognized was represented in terms of
> a library of range images of that object.

(3) document retrieval, and
(4) data mining.

7)
a) Cranfield methodology provides the researchers a laboratory enviroment for test of information retrival systems. It abstract the details and let to directly face with benchmark task called “test collection.”

b) The Text REtrieval Conference (TREC), co-sponsored by the National Institute of Standards and Technology (NIST) and U.S. Department of Defense. Its purpose was to support research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies. In particular, the TREC workshop series has the following goals:

- to encourage research in information retrieval based on large test collections;
- to increase communication among industry, academia, and government by creating an open forum for the exchange of research ideas;
- to speed the transfer of technology from research labs into commercial products by demonstrating substantial improvements in retrieval methodologies on real-world problems; and

- to increase the availability of appropriate evaluation techniques for use by industry and academia, including development of new evaluation techniques more applicable to current systems.

c)
An experiment conducted by the SMART retrieval group in TREC 1–8 demonstrated that retrieval effectiveness did indeed improve over that time. Developers of the SMART retrieval system kept a frozen copy of the system they used to participate in each of the eight TREC ad hoc tasks. They ran each system on each test collection. For each collection, the later versions of the SMART system were much more effective than the earlier versions, with the later scores approximately twice those of the earliest scores. While this experiment involved only the SMART system, **SMART results consistently tracked with the other systems' results in each TREC**. SMART results can therefore be considered representative of the field as a whole.

She mentions that "SMART results consistently tracked with the other systems' results in each TREC", if it is sure that "SMART results consistently tracked with **all** other systems' results in each TREC", than I may consider SMART results as representative of the IR filed.