SYLLABUS

Information Storage and Retrieval - (414441)

Credit Scheme :	Examination Scheme :
03 Credits	Mid_Semester : 30 Marks
	End_Semester : 70 Marks

Unit I Introduction to Information Retrieval

Basic Concepts of IR, Data Retrieval & Information Retrieval, Text mining and IR relation, IR system block diagram, **Automatic Text Analysis :** Luhn's ideas, Conflation Algorithm, Indexing and Index Term Weighting, Probabilistic Indexing, Automatic Classification. Measures of Association, Different Matching Coefficients, Cluster Hypothesis, **Clustering Techniques : Rocchio's** Algorithm, Single pass algorithm, Single Link algorithm. **(Chapter - 1)**

Unit II Indexing and Searching Techniques

Indexing : Inverted file, Suffix trees & suffix arrays, Signature Files, Scatter storage or hash addressing.

Searching Techniques : Boolean Search, sequential search, Serial search, cluster-based retrieval, Query languages, Types of queries, Patterns matching, structural queries.

IR Models : Basic concepts, Boolean Model, Vector Model, Probabilistic Model. **(Chapter - 2)**

Unit III Evaluation and Visualization of Information Retrieval System

Performance evaluation : Precision and recall, MRR, F-Score, NDCG, user-oriented measures.

Visualization in Information System : Starting points, Query Specification, document context, User relevance judgment, Interface support for search process. **(Chapter - 3)**

Unit IV Distributed and Multimedia IR

Distributed IR : Introduction, Collection Partitioning, Source Selection, Query Processing,

Multimedia IR : Introduction, Data Modeling, Query Language, Background-Spatial Access Method, A Generic Multimedia Indexing Approach, One Dimensional Time Series, Two-Dimensional color Images, Automatic Feature Extraction, Trends and Research Issue. **(Chapter - 4)**

Unit V Web Searching

Introduction, Challenges, Web Characteristics, **Search Engines :** Centralized Architecture, Distributed Architecture, User Interfaces, Ranking, Crawling the web, Indices, Browsing, Meta-searchers, Searching using Hyperlinks, Trends and Research Issues, **Introduction to Web Scraping :** Python for web scraping, Request, HTML parsing, Beautiful Soup. **(Chapter - 5)**

Unit VI Advanced Information Retrieval

XML Retrieval : Basic XML concepts, Challenges in XML retrieval, Vector space model for XML retrieval, Evaluation of XML retrieval, Text-Centric vs. Data-Centric XML retrieval.

Recommendation system : Collaborative Filtering and Content Based Recommendation of Documents and Products. Introduction to Semantic Web. **(Chapter - 6)**

TABLE OF CONTENTS

Unit I

Unit II

Unit III

Unit IV

Unit VI

Unit I

1 Introduction to Information Retrieval

Syllabus

Basic Concepts of IR, Data Retrieval and Information Retrieval, Text mining and IR relation, IR system block diagram, Automatic Text Analysis : Luhn's ideas, Conflation Algorithm, Indexing and Index Term Weighting, Probabilistic Indexing, Automatic Classification. Measures of Association, Different Matching Coefficients, Cluster Hypothesis, Clustering Techniques : Rocchio's Algorithm, Single pass algorithm, Single Link algorithm.

Contents

1.1 Basic Concept of IR

- Information Retrieval (IR) is about the process of providing answers to client's information needs. It is thus concerned with the collection, representation, storage, organization, accessing, manipulation and display, of the information items necessary to satisfying those needs.
- Definition : Information Retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections.
- There is a huge quantity of text, audio, video and other documents available on the Internet, on about any subject. Users need to be able to find relevant information to satisfy their particular information needs.
- There are two ways of searching for information : To use a search engines or to browse directories organized by categories.
- IR is the task of representing, storing, organizing, and offering access to information items. IR is different from data retrieval, which is about finding precise data in databases with a given structure.
- In IR systems, the information is not structured. It is contained in free form in text (web pages or other documents) or in multimedia content.
- The first IR systems implemented in 1970's were designed to work with small collections of text. Some of these techniques are now used in search engines.
- The information retrieval techniques focusing on the challenges faced by search engine.
  1. One particular challenge is the large scale, given by the huge number of web-pages available on the Internet.
  2. The ambiguity of the natural language (English or other languages) that makes it difficult to have perfect matches between documents and user queries.
- Information retrieval is never an easy task. The problem with IR is that document representation, either by index terms or texts cannot satisfy user need representation, which is dynamic and complicated.
- Traditional IR systems are designed to support only one type of information-seeking strategy that users engage in : Query formulation.

1.2 Data Retrieval and Information Retrieval

SPPU : June-19, March-20

- An information retrieval system is software that has the features and functions required to manipulate "information" items versus a DBMS that is optimized to handle "structured" data.

- Information retrieval and Data Retrieval (DR) are often viewed as two mutually exclusive means to perform different tasks, IR being used for finding relevant documents among a collection of unstructured/semi-structured documents.
- Data retrieval being used for finding exact matches using stringent queries on structured data, often in a Relational Database Management System (RDBMS).
- IR is used for assessing human interests, i.e., IR selects and ranks documents based on the likelihood of relevance to the user's needs. DR is different; answers to users' queries are exact matches which do not impose any ranking.
- Data retrieval involves the selection of a fixed set of data based on a well-defined query. Information retrieval involves the retrieval of documents of natural language.
- IR systems don't support transactional updates whereas database systems support structured data, with schemas that define the data organization. IR systems deal with some querying issues not generally addressed by database systems and approximate searching by keywords.

1.2.1 Difference between Data Retrieval and Information Retrieval

Parameters	Data retrieval	Information retrieval
Example	Data base query	WWW search
Matching	Exact	Partial match Best match
Inference	Deduction	Induction
Model	Deterministic	Probabilistic
Query language	Artificial	Natural
Query specification	Complete	Incomplete
Items wanted	Matching	Relevant
Error response	Sensitive	Insensitive
Classification	Monotonic	Polytechnic

University Question

1. *Differentiate between data retrieval and information retrieval.*

SPPU : June-19, End Sem, March-20, In Sem, Marks 6

1.3 Text Mining and IR Relation

- Text mining also is known as Text Data Mining (TDM) and Knowledge Discovery in Textual Database (KDT). A process of identifying novel information from a collection of texts.
- Text Mining is understood as a process of automatically extracting meaningful, useful, previously unknown and ultimately comprehensible information from textual document repositories.
- Text mining can be visualized as consisting of two phases: Text refining that transforms free-form text documents into a chosen intermediate form, and knowledge distillation that deduces patterns or knowledge from the intermediate form.

Text Mining Applications

1. Classification of news stories, web pages, ... , according to their content
2. Email and news filtering
3. Organize repositories of document-related meta-information for search and retrieval (search engines)
4. Clustering documents or web pages
5. Gain insights about trends, relations between people, places and/or organizations

Text Mining	Data Mining
Linguistic processing or natural language processing (NLP)	Process directly
Discover heretofore unknown information	Identify causal relationship
Applications deal with much more diverse and eclectic collections of systems and formats	Structured numeric transaction data residing in rational data warehouse

1.4 IR System Block Diagram

SPPU : March-19, 20, June-19

- An information retrieval system is an information system, which is used to store items of information that need to be processed, searched, retrieved, and disseminated to various user populations.
- Information retrieval is the process of searching some collection of documents, in order to identify those documents which deal with a particular subject. Any system that is designed to facilitate this literature searching may legitimately be called an information retrieval system.

- Conceptually, IR is the study of finding needed information. It helps users to find information that matches their information needs. Historically, IR is about document retrieval, emphasizing document as the basic unit.
- Information retrieval locates relevant documents, on the basis of user input such as keywords or example documents, for example : Find documents containing the words "database systems".
- Fig. 1.4.1 shows information retrieval system block diagram. It consists of three components : **Input, processor and output.**

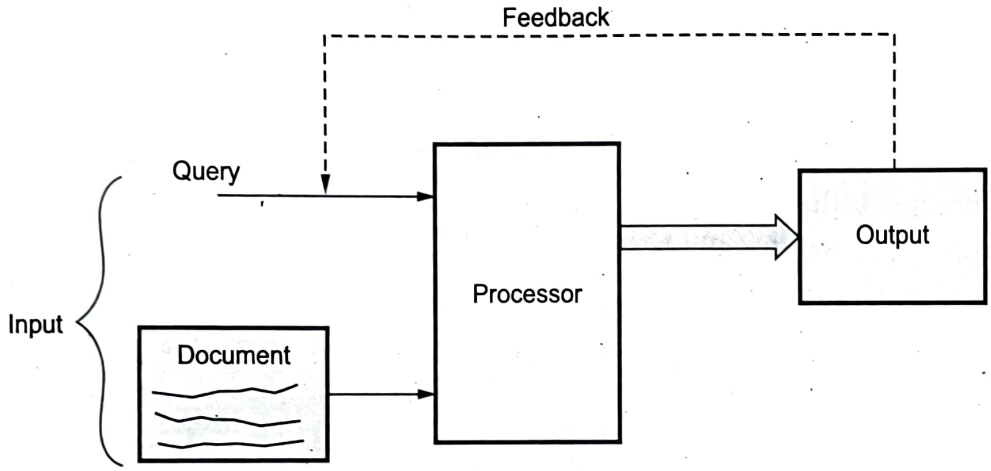


Fig. 1.4.1 IR block diagram

a) **Input :** Store only a representation of the document or query which means that the text of a document is lost once it has been processed for the purpose of generating its representation.

b) A **document representative** could be a list of extracted words considered to be significant.

c) **Processor :** Involve in performing actual retrieval function, executing the search strategy in response to a query.

d) **Feedback :** Improving the subsequent run after sample retrieval.

e) **Output :** A set of document numbers.

- Information retrieval locates relevant documents, on the basis of user input such as keywords or example documents.
- The computer-based retrieval systems store only a representation of the document or query which means that the text of a document is lost once it has been processed for the purpose of generating its representation.

- The process may involve structuring the information, such as classifying it. It will also involve performing the actual retrieval function that is executing the search strategy in response to a query.
- Text document is the output of information retrieval system. Web search engines are the most familiar example of IR systems.

University Question

1. *Draw and explain IR block diagram.*

SPPU : March-19, 20, In Sem, Marks 5, June-19, End Sem, Marks 6

1.5 Automatic Text Analysis

SPPU : March-19,20, June-19

- A computerized information retrieval system can actually operate to retrieve some information, that information must have already been stored inside the computer. Originally it will usually have been in the form of documents.
- Natural language is not used by computer for storing text. Document representative which may have been produced from the documents either manually or automatically.

1.5.1 Luhn's Ideas

- Luhn's determined significant index words by their frequency counts in the document text, as representation or characterization of a document. The result, a derived list of index terms, can be called keywords or terms for each document.
- Luhn's idea was that words with high and low frequency are too common and too rate to contribute significantly to the content of a document, and that only words with medium frequency are significant.
- The idea of using frequency to measure word significance is based on the fact that a writer normally repeats certain words when writing on a subject.
- The more a specific word is found in a document, the more significance may be assigned to these words. He also takes common words in consideration. Such words must be present to tie the words together, but the type of significance does not reside in such words.
- These common words must be segregated and excluded from being considered as significant words by some method. As frequency has been proposed as a criterion, this step can be seen as an upper boundary of significant words.

- A lower boundary is needed too. This is to bracket the most useful range of words. Fig. 1.5.1 shows Luhn's idea.

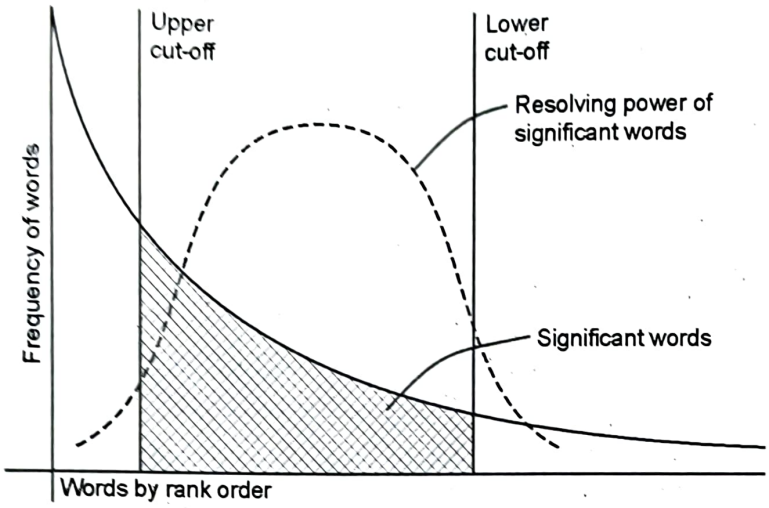


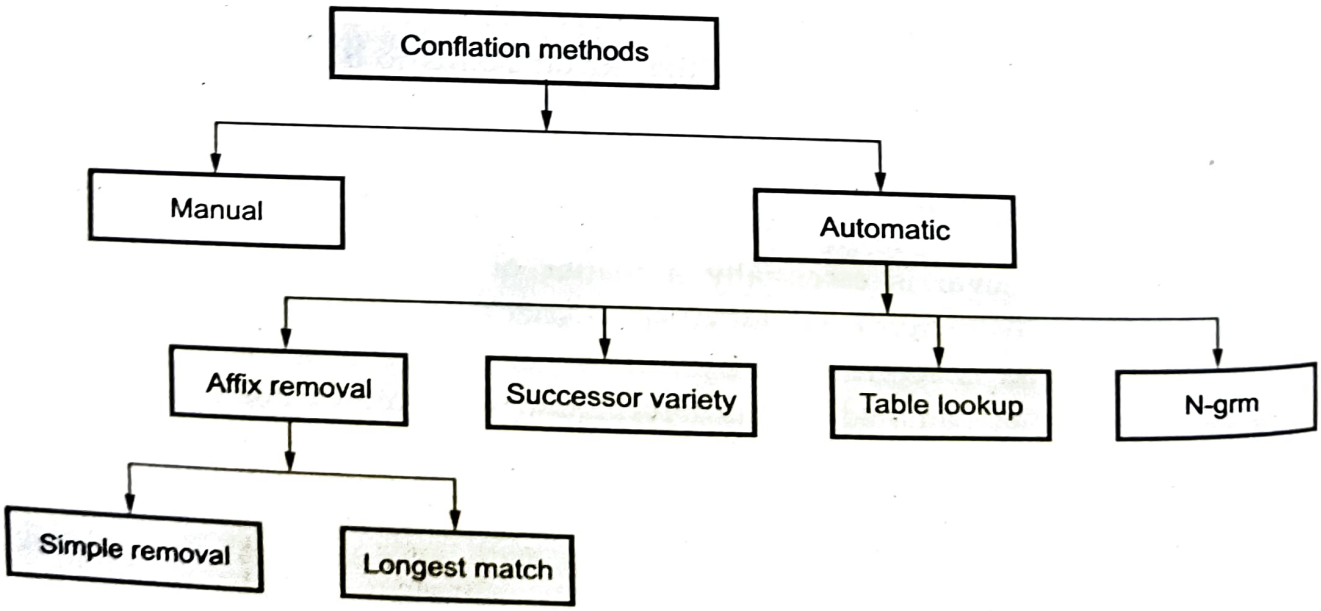
Fig. 1.5.1 Luhn's idea

- The frequency curve actually demonstrates Zipf's law. With this law Zipf stated that the product of the frequency of occurrences of words and the rank order is approximately constant.
- Luhn's method for each document :
  1. Filter terms in the document using a list of stopwords.
  2. Normalize terms by stemming like differentiate, different, differently.
  3. Calculate frequencies of normalized terms.
  4. Remove non-frequent terms.
- Problem with Luhn's model is that the cut off points to be determined empirically, but trial and error.

1.5.2 Conflation Algorithm

- Information retrieval is essentially a matter of deciding which documents in a collection should be retrieved to satisfy a user's need for information. The user's information need is represented by a query or profile, and contains one or more search terms, plus some additional information such importance weights.
- The retrieval decision is made by comparing the terms of the query with the index terms (important words or phrases) appearing in the document itself.
- The decision may be binary (retrieve/reject), or it may involve estimating the degree of relevance that the document has to the query.

- For example morphological variants (e.g., COMPUTATIONAL, COMPUTER, COMPUTERS, COMPUTING etc.) are generally the most common, with other sources including valid alternative spellings, mis-spellings, and variants arising from transliteration and abbreviation.
- A number of stemming algorithms, or stemmers, have been developed, which attempt to reduce a word to its stem or root form. Stemming is the process for reducing inflected words to their stem, base or root form, generally a written word form. The process of stemming is often called conflation. These programs are commonly referred to as stemming algorithms or stemmers.
- Word conflation is the process by which a group of words that have the same meaning are reduced to a single term. For example : COLLECT, COLLECTED, COLLECTING, COLLECTION, COLLECTIONS , can all be conflated to the common term COLLECT.
- The key terms of a query or document are represented by stems rather than by the original words. This not only means that different variants of a term can be conflated to a single representative form. It also reduces the dictionary size. The number of distinct terms needed for representing a set of documents. A smaller dictionary size results in a saving of storage space and processing time.
- Conflation algorithms are classified into two main classes : Stemming algorithms which are language dependent and string-similarity algorithms which are language independent.



- Stemming algorithms typically try to reduce related words to the same base (or stem).System will usually consist of three parts :
  1. Removal of high frequency words,
  2. Suffix stripping,
  3. Detecting equivalent stems.
- The removal of high frequency words, 'stop' words or 'fluff' words is one way of implementing Luhn's upper cut-off. This is normally done by comparing the input text with a 'stop list' of words which are to be removed.
- The advantages of the process are not only that non-significant words are removed and will therefore not interfere during retrieval, but also that the size of the total document file can be reduced by between 30 % to 50 %.
- The second stage, suffix stripping, is more complicated. A standard approach is to have a complete list of suffixes and to remove the longest possible one. For example, we may well want UAL removed from FACTUAL but not from EQUAL.
- Suffix stripping algorithms do not rely on a lookup table that consists of inflected forms and root form relations. Instead, a typically smaller list of "rules" is stored which provide a path for the algorithm, given an input word form, to find its root form. Some examples of the rules include :
  1. If the word ends in 'ed', remove the 'ed'
  2. If the word ends in 'ing', remove the 'ing'
  3. If the word ends in 'ly', remove the 'ly'
- Following are the simple steps of conflation algorithm :
  1. Open and read each input file and create a single index file.
  2. Remove or filer out all stop words.
  3. Remove all suffixes/affixes from each word if present.
  4. Count frequencies of occurrences for each root word from 3.
  5. Apply porters rules/algorithm for each root word from 3 and store in index file.

Affix removal stemmers :

a) If a word ends in "ies" but not "eies" or "aies" then replace "ies" with "y"

b) If a word ends in "es" but not "aes", "ees", or "oes" then replace "es" with "e"

c) If a word ends in "s", but not "us" or "ss" then replace "s" with NULL.

University Questions

1. *Explain steps in conflation algorithm using a suitable example.* **SPPU : March-19, In Sem, Marks 5**
2. *List and explain steps of conflation algorithm.* **SPPU : June-19, End Sem, Marks 6**
3. *Explain Luhn's idea in detail with diagram.* **SPPU : March-20, In Sem, Marks 4**

1.6 Indexing and Index Term Weighting

1.6.1 Index

- With the increasing large amount of information on the Internet, an apparent problem is to find the relevant information via the Internet.
- Indexing is the process of creating a representation of a document. The index language which comes out of the conflation algorithm and this index language used to describe documents and requests. The elements of the index language are index terms, which may be derived from the text of the document to be described.
- Indexing is a time saving mechanism. The process takes place off-line, that is, the end user of the information retrieval system is not directly involved. The indexing process results in a representation of the document.
- Traditional IR systems usually adopt index terms to index and retrieve documents. In a restricted sense, an index term is a keyword (or group of related words) which has some meaning of its own. In a more general form, an index term is simply any word which appears in the text of a document in a collection.
- Preprocessing of document i.e. raw documents must be converted into bag of terms representation. These expressions are sometimes called **document representatives.**
- The process of representing their information need is often referred to as the query formulation process. The resulting representation is the query.
- To make these representatives for each document, first collect the words and create the file containing words except the stop words, then stem the words with in a file thus we have all the terms important to our search in their root forms.
- Count the frequency of each word and the word having frequency above a threshold is selected as an index term. Collection of all such terms creates our index table (document representative) for that document.
- Automatic indexing is the process of representing the topical content of digitally stored documents or document surrogates, performed without, or with very modest, human intervention.

- Indexes are constructed, separately, on three distinct levels : Terms in a document such as a book; objects in a collection such as a library; and documents (such as books and articles) within a field of knowledge.
- Subject indexing systems have been classified broadly as pre-coordinate and post-coordinate systems. The major objective of any indexing system is to represent the contents of documents through keywords or descriptors.

1.6.2 Index Term Weighting

- An **exhaustive index** is one which lists all possible index terms. Greater **exh-austivity** gives a higher recall, or more likelihood of all the relevant articles being retrieved, however, this occurs at the expense of precision.
- This means that the user may retrieve a larger number of irrelevant documents or documents which only deal with the subject in little depth.
- In a manual system a greater level of exhaustivity brings with it a greater cost as more man hours are required.
- The **specificity** describes how closely the index terms match the topics they represent. An index is said to be specific if the indexer uses parallel descriptors to the concept of the document and reflects the concepts precisely.
- Human indexers are able to rank their indexing approximately in order of increasing exhaustivity or specificity. However, the same is not easily done for automatic indexing.
- **Exhaustivity** is a property of index descriptions and indicates the extent to which an index description covers the document content. **Specificity** is a property of an individual index term and indicates to what extent each index term is specific to a document.
- The effect of indexing exhaustivity and specificity on retrieval effectiveness can be explained by two parameters :

1. **Recall** is the ratio of the number of relevant documents retrieved to the total number of relevant documents in the collection.
2. **Precision** is the ratio of the number of relevant documents retrieved to the total number of documents retrieved.

$$\textbf{Recall} = \frac{\text{Number of relevant documents retrieved}}{\text{Number of relevant documents in the collection}}$$

$$\textbf{Precision} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of documents retrieved}}$$

- It is possible that a high level of exhaustivity of indexing leads to high recall and low precision. Conversely, a low level of exhaustivity leads to low recall and high

precision. The converse is true for levels of indexing specificity, high specificity leads to high precision and low recall, etc.

- Using single words as index terms generally has good exhaustivity, but poor specificity due to word ambiguity.
- In an information retrieval system using single words as index terms, a query including the word "bank" will retrieve all documents including "bank" irrespective of the meaning of "bank" in the query. One approach to remedy this ambiguity problem is to introduce index terms more complex than single words, such as phrases.
- **Good index** term is one which, when assigned as an index term to a collection of documents, renders the documents as dissimilar as possible, whereas a **bad term** is one which renders the documents more similar.

1.7 Probabilistic Indexing

- Probabilistic Indexing is probability model for information retrieval. It uses word concept. Whether the word is used for indexing or not ?
- Probabilistic Retrieval : Many documents - One query
- Probabilistic Indexing : One document - Many queries
- Information retrieval deals with uncertain information. Probability theory seems to be the most natural way to quantify uncertainty.
- The statistical behaviour of 'speciality' words was different from that of 'function' words. The function words were closely modeled by a Poisson distribution over all documents whereas speciality words did not follow a Poisson distribution.
- Distribution of a function word w over a set of texts is represented by probability f(n) is given as follows :

$$f(n) = \frac{e^{-x} X^n}{n!}$$

- According to the Bookstein-Swanson-Harter model, specialty words are 'content-bearing' whereas function words are not. It means that a word randomly distributed according to a Poisson distribution is not informative about the document in which it occurs.
- Document collection can be broken up into subsets. Each subset being made up of documents that are about a given word to the same degree.
- Harter has identified two assumptions :
  1. The probability that a document will be found relevant to a request for information on a subject is a function of the relative extent to which the topic is treated in the document.

2. The number of tokens in a document is a function of the extent to which the subject referred to by the word is treated in the document.

- To calculate the required probability of relevance for a content-bearing word we need to postulate what its distribution would look like.
- If there are only two such subsets differing in the extent to which they are about a word "w" then the distribution of "w" can be described by a mixture of two Poisson distributions.

$$f(n) = \frac{p_1^{e^{-x1}} x_1^n}{n!} + \frac{(1-p_1)^{e^{-x2}} x_2^n}{n!}$$

- Here p_1 is the probability of a random document belonging to one of the subsets and x_1 and x_2 are the mean occurrences in the two classes. This expression shows why the model is sometimes called the **2-Poisson model.**
- Calculate the probability of relevance for any document from one of these classes is given below. It is the ratio

$$\frac{p_1 e^{-x1} x_1^k}{p_1 e^{-x1} x_1^k + (1-p_1) e^{-x2} x_2^k}$$

Discrimination and/or Representation

- There are two conflicting ways of looking at the problem of characterizing documents for retrieval.
  1. To characterize a document through a representation of its contents, regardless of the way in which other documents may be described, this might be called **representation without discrimination.**
  2. To insist that in characterizing a document one is discriminating it from all, or potentially all, other documents in the collection, this we might call **discrimination without representation.**

1.8 Automatic Classification

- **Classification** is a technique used to predict group membership for data instances. For example : You may wish to use classification to predict the day. The weather on a particular day will be "sunny", "rainy" or "cloudy".
- **Clustering** is a process of partitioning a set of data in a set of meaningful subclasses. Every data in the subclass shares a common trait. It helps a user understand the natural grouping or structure in a data set. There are two main areas of application of classification methods in information retrieval :
  1. Document clustering (Relationship between document)
  2. Keyword clustering (Relationship between terms)

- A computer can process words. The principle on which automated classification system is based is statistics, whereas documents can be classified on the basis of the words they contain.
- **Automatic indexing** deals with the automation of the process whereby we assign index terms to documents, that is the terms which are used later to retrieve a subject of documents from a file. **Automatic classification** is concerned with procedures and systems which can make comparison between terms used to index documents.
- **Keyword classification** is the automatic construction of classification schemes through an examination of keywords used in indexing documents. Primary objective is to derive grouping of keywords.
- Keyword classification can be used to characterize the document in a collection as an aid to retrieval. It is required as a device for providing characterizations for all the documents in a collection.
- While describing the frequency of a word the first step is that the document is translated into machine-readable code. A computer program develops a machine listing of all words in the document, arranged by frequency of occurrence and alphabetically within frequency.

1.9 Measures of Association

- In order to cluster the items in a data set, some means of quantifying the degree of association between them is required. This may be a distance measure, or a measure of similarity or dissimilarity. Some clustering methods have a theoretical requirement for use of a specific measure.
- Binary relationship between objects is used for classification. On the basis of this relationship a classification method can construct a system of clusters. The relationship between the document is described by
  a. **Similarity :** These values indicate how much two documents or objects are near to each other.
  b. **Association :** It is same as similarity but difference is objects which are considered for comparison are object characterized by discrete state attributes.
  c. **Dissimilarity :** Dissimilarity value shows that how much far the objects are.
- The measure of similarity is designed to quantify the likeness between objects. It is possible to group objects in such a way that an object in a group is more like the other members of the group.
- A measure of association increases as the number or proportion of shared attribute states increases. In information retrieval system, two documents will be similar to

each other if they have more number of common index terms. If two documents are having less number of common index terms then obviously they will be semantically far from each other.

1.10 Different Matching Coefficient

1.10.1 Simple Matching Coefficient

- It is the number of shared index terms. This *coefficient does not* take into account the **sizes** of X and Y. An object is represented by a set of keywords and that the counting measure | . | gives the size of the set. The simplest of all association measures is :

  a. Simple matching coefficient :

  $$|X \cap Y| \qquad \sum_i x_i y_i$$

- The following coefficients which have been used in document retrieval *take into account* the information provided by the **sizes** of X and Y.

  b. **Dice's coefficient :** Normalises for length by dividing by the total number of non-zero entries. We multiply by 2 so that we get a measure that ranges from 0 to 1.0

  $$\frac{2|X \cap Y|}{|X| + |Y|}$$

  c. **Jaccard's coefficient :** Penalizes a small number of shared entries more than Dice Coefficient does.

  $$\frac{|X \cap Y|}{|X \cup Y|}$$

  d. **Cosine coefficient :** Magnitude of the vector is defined by the square root of the sum of the squares of the components of the vector. The length of the vector normalises for length by dividing by the total number of non-zero entries.

  $$\frac{|X \cap Y|}{|X| \cdot |Y|} \qquad \frac{|X \cap Y|}{|X|^{1/2} * |Y|^{1/2}}$$

2. Overlap coefficient :

$$\frac{|X \cap Y|}{\min(|X|, |Y|)}$$

- Example : Document 1 = (3,2,1,0,0,0,1,1) and Document2 = (1,1,1,0,0,1,0,0) then
  Dice = 2 x 6 / (8+4) = 1

Jaccard = 6 / 8 + 4 - 6 = 1

Cosine = 6 /sqrt(16x4) = 6/8 = 0.75

Let us consider following :

If S_1 and S_2 is as follows

$$S_1(X, Y) = |X \cap Y|$$

$$S_2(X, Y) = \frac{2|X \cap Y|}{|X| + |Y|}$$

Then :

$$|X_1| = 1, \quad |Y_1| = 1$$

$$|X_1 \cap Y_2| = 1 \Rightarrow S_1 = 1 \; S_2 = 1$$

$$|X_2| = 10 \quad |Y_2| = 10$$

$$|X_2| = 10 \quad |Y_2| = 10, \; |X_2 \cap Y_2| = 1 \Rightarrow S_1 = 1 \; S_2$$

$$= 1/10$$

1.10.2 Dissimilarity Coefficients

- Any dissimilarity function can be transformed into a similarity function by a simple transformation of the form $s = (1+d)^{-1}$ but the reverse is not always true.
- If P is the set of objects to be clustered, a pair-wise dissimilarity coefficient D is a function from P x P to the non-negative real numbers. Dissimilarity coefficient D in general, satisfies the following conditions :

  D1 $D(X, Y) \geq 0$ For all $X, Y \in P$

  D2 $D(X, X) = 0$ For all $X \in P$

  D3 $D(X, Y) = D(Y,X)$ For all $X, Y \in P$
- A dissimilarity coefficient is a kind of 'distance' function. Many of the dissimilarity coefficients satisfy the triangle inequality :

  D4 $D(X,Y) \leq D(X, Z) + D(Y,Z)$
- An example of a dissimilarity coefficient satisfying D1 - D4 is

$$\frac{|X \Delta Y|}{|X| + |Y|}$$

Where $(X \Delta Y) = (X \cup Y) - (X \cap Y)$ is the symmetric different of sets X and Y. It is simply related to Dice's coefficient by

$$1 - \frac{2|X \cap Y|}{|X| + |Y|} = \frac{|X \Delta Y|}{|X| + |Y|}$$

- Instead of representing each document by a set of keywords, we represent it by a binary string where the absence or presence of the i^{th} keyword is indicated by a zero or one in the i^{th} position respectively. In that case

$$\frac{\sum x_i (1-y_i) + \sum y_i (1-x_i)}{\sum x_i + \sum y_i}$$

where summation is over the total number of different keywords in the document collection.

1.11 Cluster Hypothesis

- Documents in the same cluster behave similarly with respect to relevance to information needs. Clustering is an unsupervised learning method. The result is based solely on the object representation, the similarity measure and the clustering algorithm.
- The relevant documents are more like one another than they are like non-relevant documents. We can compute the association between all pairs of documents as follows :
  1. Both of which are relevant to a request, and
  2. One of which is relevant and the other non-relevant
- **Summing** over a set of requests gives the relative distribution of Relevant-Relevant (R-R) and Relevant-Non-Relevant (R-N-R) associations of a collection. Fig. 1.11.1 shows relationship between them.

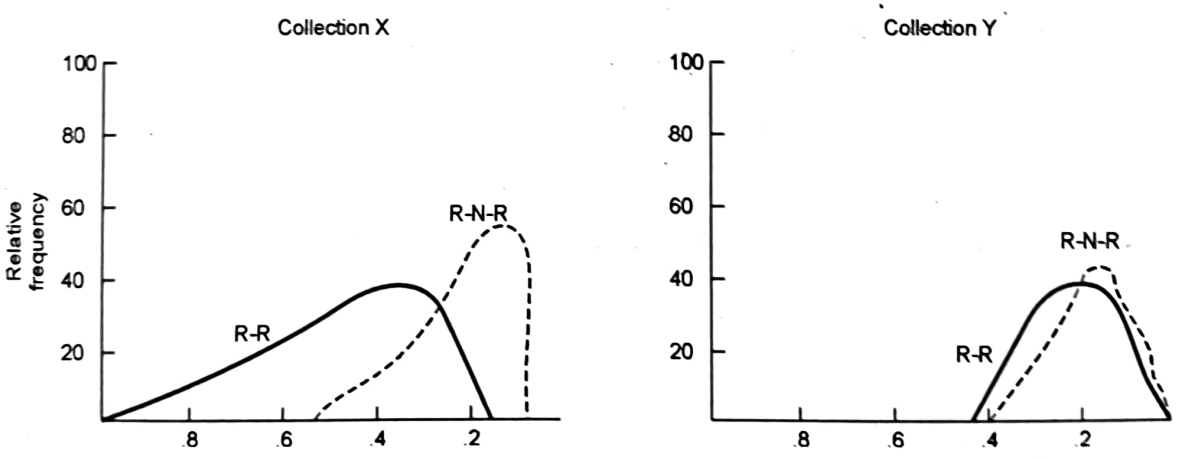


Fig. 1.11.1 R-R and R-N-R associations

- It shows that :
  1. Separation for collection X is good while for Y it is poor; and
  2. Strength of the association between relevant documents is greater for X than for Y.

- Cluster hypothesis refers to given document descriptions. The object of making permanent or temporary changes to a description by such techniques as keyword classifications can therefore be expressed as an attempt to increase the distance between the two distributions R-R and R-N-R.

1.11.1 Use of Clustering in IR

- Cluster analysis is a statistical technique used to generate a category structure which fits a set of observations. The groups which are formed should have a high degree of association between members of the same group and a low degree between members of different groups. While cluster analysis is sometimes referred to as automatic classification.
- Cluster analysis can be performed on documents in following ways :
  1. Documents may be clustered on the basis of the terms that they contain. The aim of this approach has usually been to provide more efficient or more effective retrieval, though it has also been used after retrieval to provide structure to large sets of retrieved documents.
  2. Documents may be clustered based on co-occurring citations in order to provide insights into the nature of the literature of a field.
  3. Terms may be clustered on the basis of the documents in which they co-occur, in order to aid in the construction of a thesaurus or in the enhancement of queries.
- Cluster analysis can be easily implemented with available software packages, it is not without problems. These include :
  1. Selecting the attributes on which items are to be clustered and their representation.
  2. Selecting an appropriate clustering method and similarity measure from those available.
  3. Creating the clusters or cluster hierarchies, which can be expensive in terms of computational resources.
  4. Assessing the validity of the result obtained.
  5. If the collection to be clustered is a dynamic one, the requirements for update must be considered.
  6. If the aim is to use the clustered collection as the basis for information retrieval, a method for searching the clusters or cluster hierarchy must be selected.

- In choosing a cluster method for use in experimental IR, two criteria are used :

1. **Theoretical soundness :** To list some of the more important of these :

   A. The method produces a clustering which is unlikely to be altered drastically when further objects are incorporated, i.e. it is stable under growth.

   B. The method is stable in the sense that small errors in the description of the objects lead to small changes in the clustering;

   C. The method is independent of the initial ordering of the objects.

2. **Efficiency** is really a property of the algorithm implementing the cluster method. It is sometimes useful to distinguish the cluster method from its algorithm, but in the context of IR this distinction becomes slightly less than useful since many cluster methods are defined by their algorithm, so no explicit mathematical formulation exists.

1.12 Clustering Algorithms

- A cluster is therefore a collection of objects which are "similar" between them and are "dissimilar" to the objects belonging to other clusters. Let us consider following Fig. 1.12.1

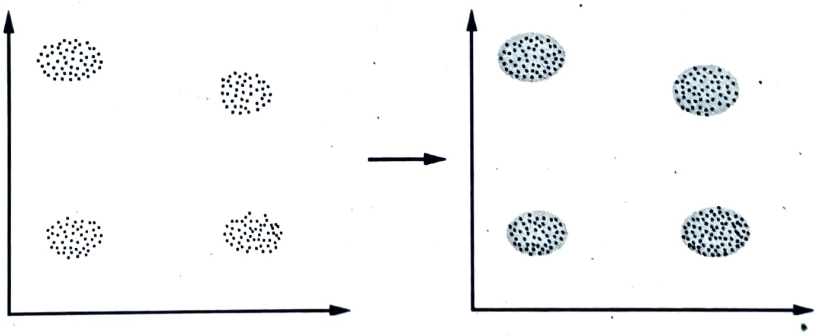


Fig. 1.12.1 Clustering

- In this case we easily identify the 4 clusters into which the data can be divided; the similarity criterion is distance : two or more objects belong to the same cluster if they are "close" according to a given distance (in this case geometrical distance). This is called **distance-based clustering.**
- **Conceptual clustering :** Two or more objects belong to the same cluster if this one defines a concept common to all that objects. In other words, objects are grouped according to their fit to descriptive concepts, not according to simple similarity measures.

- Clustering methods are usually categorized according to the type of cluster they produce. The clustering methods are categorized as :
  1. **Hierarchical methods :** These types cluster produces the output list of clusters. Small clusters of highly similar documents nested within larger clusters of less similar documents.
  2. **Non-hierarchical methods :** This method produced unordered lists.
- Other clustering methods are **exclusive cluster** and **overlapping cluster.** In the first case (exclusive cluster) data are grouped in an exclusive way, so that if a certain datum belongs to a definite cluster then it could not be included in another cluster. The **overlapping clustering** uses fuzzy sets to cluster data, so that each point may belong to two or more clusters with different degrees of membership. In this case, data will be associated to an appropriate membership value.
- Four of the most used clustering algorithms :
  a. K-means
  b. Fuzzy C-means
  c. Hierarchical clustering
  d. Mixture of Gaussians
- Each of these algorithms belongs to one of the clustering types listed above. So that, K-means is an exclusive clustering algorithm, Fuzzy C-means is an overlapping clustering algorithm, Hierarchical clustering is obvious and lastly mixture of Gaussian is a probabilistic clustering algorithm.

1.13 Rochhio's Algorithm

- Rocchio's clustering algorithm was developed on the SMART project. It operates in three stages.

1. In the first stage it selects a number of objects as cluster centers. The remaining objects are then assigned to the centers or to a 'rag-bag' cluster. On the basis of the initial assignment the cluster representatives are computed and all objects are once more assigned to the clusters. The assignment rules are explicitly defined in terms of thresholds on a matching function. The final clusters may overlap.
2. The second stage is essentially an iterative step to allow the various input parameters to be adjusted so that the resulting classification meets the prior specification of such things as cluster size, etc. more nearly.
3. The third stage is for 'tidying up'. Unassigned objects are forcibly assigned, and overlap between clusters is reduced.

1.14 Single Pass Algorithm

SPPU : March-20

- In this algorithm a document is joined to a cluster, if it is maximally covered by the corresponding cluster seed (i.e. document). We know that the matrix entry c_{ij} shows how strongly document "i" is covered by document "j". Therefore if j and k are two different cluster seeds and if $c_{ij} > cik$, then this means that document "i" will join the cluster initiated by documents "j" than that of document "k".
- The working of single pass algorithm is as follows :

1. Determine the cluster seeds, which are the documents determining the first nc highest cluster seed powers. i = 1;
2. Repeat;

   If document "i" is not a cluster seed.

   then

   do;

find the cluster seed which maximally covers document-i. If there is more than one cluster seed with this property the document might be assigned to all of the seeds if overlapping is allowed, otherwise it will be assigned to only one according to a predetermined decision;

   end;

   i = i + 1;

   Until i > m;

3. The document which are not covered any of the seeds, might form a separate cluster by themselves.
4. Stop.

Single-Pass algorithm operates as follows :

I. The object descriptions are processed serially;

II. The first object becomes the cluster representative of the first cluster;

III. Each subsequent object is matched against all cluster representatives existing at its processing time;

IV. A given object is assigned to one cluster according to some condition on the matching function;

V. When an object is assigned to a cluster the representative for that cluster is recomputed;

VI. If an object fails a certain test it becomes the cluster representative of a new cluster.

- Example : Consider the following object

  Object = {1, 2, 3, 4, 5, 6}

  Similarity matrix is as follows :

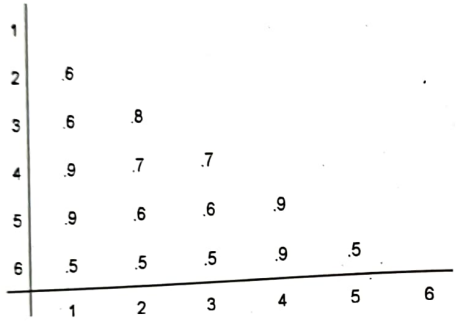


Fig. 1.14.1

Threshold : 0.89

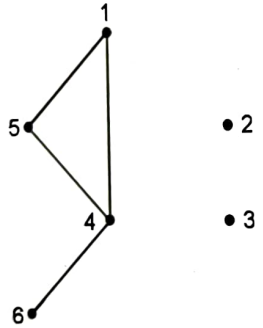


Fig. 1.14.2 A similarity coefficient for 6 objects and the graph

University Question

1. *Explain single pass algorithm with example.* **SPPU : March-20, In Sem, Marks 6**

1.15 Single Link Algorithm

SPPU : March-20

- Basic input to a single-link clustering algorithm is dissimilarity coefficient. The output is a hierarchy with associated numerical levels called a **dendrogram.**
- Frequently the hierarchy is represented by a tree structure such that each node represents a cluster. The two representations are shown side by side in Fig. 1.15.1 for the same set of objects {A,B,C,D,E}.

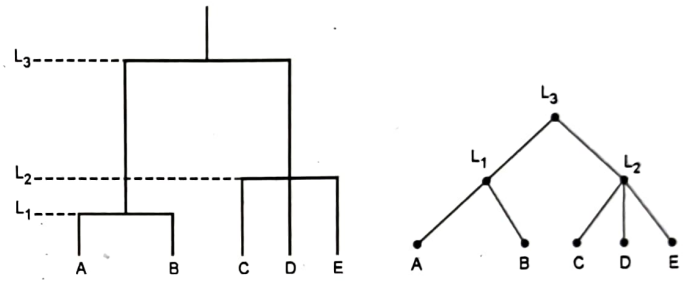


Fig. 1.15.1 A dendrogram with corresponding tree

- The clusters are :
  1. At level L1 : {A,B}, {C}, {D}, {E}
  2. At level L2 : {A,B}, {C,D,E}
  3. At level L3 : {A,B,C,D,E}
- At each level of the hierarchy one can identify a set of classes, and as one moves up the hierarchy the classes at the lower levels are nested in the classes at the higher levels.
- A Dissimilarity Coefficient (DC) can be characterized by a set of graphs, one for each value taken by the DC. The different values taken by the DC in the example are L = .1, .2, .3, .4. The graph at each level is given by a set of vertices corresponding to the objects to be clustered, and any two vertices are linked if their dissimilarity is at most equal to the value of the level L.
- Graphs at values other than those taken by the DC are simply the same as at the next smallest value actually taken by the DC, for example, compare the graphs at L = 0.15 and L = 0.1.

Dissimilarity matrix :

	1	2	3	4
2	.4			
3	.4	.2		
4	.3	.3	.3	
5	.1	.4	.4	.1

Binary matrices :

	1	2	3	4
2	0			
3	0	0		
4	0	0	0	
5	1	0	0	1

Threshold = .1

	1	2	3	4
2	0			
3	0	1		
4	0	0	0	
5	1	0	0	1

Threshold = .2

	1	2	3
2	0		
3	0	1	
4	1	1	1
5	1	0	0

Threshold = .3

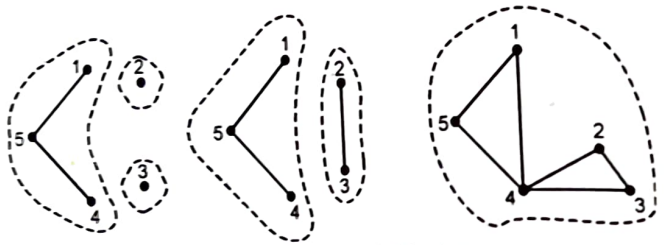
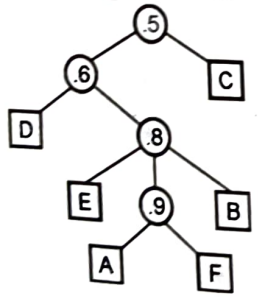


Fig. 1.15.2 Graph and Clusters

Example : Dissimilarity matrix is given below :

	A	B	C	D	E
B	.3				
C	.5	.4			
D	.6	.5	.3		
E	.8	.7	.5	.4	
F	.9	.8	.2	.1	.3

Graph and cluster for above method is given below :



Theoretical advantages

1. Only the rank-ordering of values of the similarity coefficients is relevant.
2. It is stable under small errors in similarity values.
3. It is stable under upgrade.
4. A completely linked set is never assigned to more than one cluster at a certain level in the hierarchy.

Problems

1. Chaining - long, loosely bound clusters with little internal cohesion
2. Aberration - relevant documents, isolated at high levels in the hierarchies (and retrieved by a search).

University Question

1. *Explain single link algorithm with example.* **SPPU : March-20, In Sem, Marks 6**

❑❑❑

Unit II

2 Indexing and Searching Techniques

Syllabus

Indexing : *Inverted file, Suffix trees & suffix arrays, Signature Files, Scatter storage or hash addressing.*

Searching Techniques : *Boolean Search, sequential search, Serial search, cluster-based retrieval, Query languages, Types of queries, Patterns matching, structural queries.*

IR Models : *Basic concepts, Boolean Model, Vector Model, Probabilistic Model.*

Contents

2.1 Indexing : Inverted File

- Each document is assigned a list of keywords or attributes. Each keyword (attribute) is associated with operational relevance weights.
- An inverted file is the sorted list of keywords (attributes), with each keyword having links to the documents containing that keyword.
- Penalty : The size of inverted files ranges from 10 % to 100 % of more of the size of the text itself. It need to update the index as the data set changes.
- Inverted file is composed of two elements :
  a. Vocabulary
  b. Occurrences
- **Vocabulary** is the set of all different words in the text. For each such word a list of all the text positions where the word appears is stored. The set of all those lists is called the occurrences .
- A controlled vocabulary which is the collection of keywords that will be indexed. Words in the text that are not in the vocabulary will not be indexed. A list of stop-words that for reasons of volume will not be included in the index.
- A set of rules that decide the beginning of a word or a piece of text that is indexable. A list of character sequences to be indexed (or not indexed).
- Fig. 2.1.1 shows a sample text with inverted index file. Each entry in the vocabulary has **the word**, a **pointer into the postings structure** and word **metadata.**
- Every query goes here first
  a. Keep in memory or a separate file b. Search should be fast
  c. Support prefix or pattern matching d. Support updating

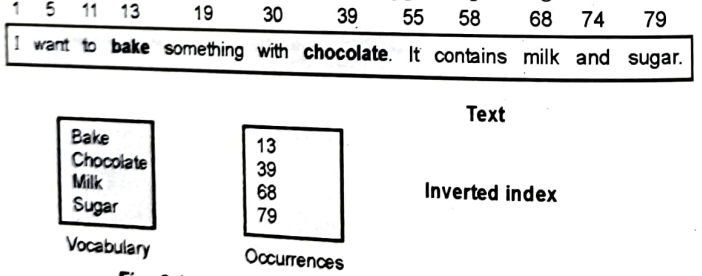


Fig. 2.1.1 Sample text with inverted index file

- The positions can refer to words or characters. The word positions simplify phrase and proximity queries, while character positions facilities direct access to the matching text positions. Time needed to access posting lists is a function of their length and their allocation.

- The space required for the **vocabulary** is rather small. According to Heaps law the vocabulary grows as O(nB), where B is a constant between 0 and 1 dependent on the text. The **occurrences** required much more space. Each word appearing in the text is referenced once in that structure, the extra space is O(n).
- Block addressing is used to reduce the space requirement. The text is divided in blocks and the occurrences point to the blocks where the word appears. The classical indices which point to the exact occurrences are called **full inverted indices.**
- The definition of inverted files does not require that the addresses in the directory are in any order. However, to facilitate operations such as conjunction ('and') and disjunction ('or') on any two inverted lists, the addresses are normally kept in record number order. This means that 'and' and 'or' operations can be performed with one pass through both lists. The penalty we pay is of course that the inverted file becomes slower to update.
- Fig. 2.1.2 shows the sample text split into blocks and an inverted index using block addressing built on it.

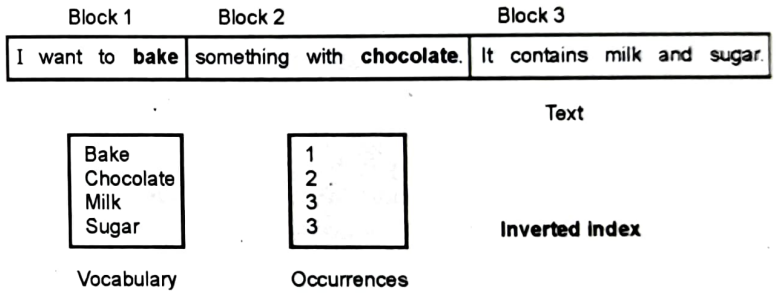


Fig. 2.1.2 Sample text split into blocks and an inverted index using block addressing

- The blocks can be of fixed size or they can be defined using the natural division of the text collection into files, documents, web pages etc. Concept of block improves the retrieval efficiency.

2.1.1 Searching

- The search algorithm on an inverted index follows three steps :
  1. **Vocabulary search :** The words and pattern presents in the query are isolated and searched in the vocabulary.
  2. **Retrieval of occurrences :** The list of the occurrences of all the words found is retrieved.

3. **Manipulation of occurrences :** The occurrences are processed to solve phrases, proximity or Boolean operations. If block addressing is used it may be necessary to directly search the text to find the information missing from the occurrences.

- Structures used in inverted files are sorted arrays, hashing structures, Tries (digital search trees) and combinations of these structures. Single word queries can be searched using any suitable data structure to speed up the search.
- If the index stores character positions the phrase query cannot allow the separators to be disregarded and the proximity has to be defined in terms of character distance.

2.1.2 Construction

- Building and maintaining an inverted index is a relatively low cost task. An inverted index on a text of "n" characters can be built in O(n) time. All the vocabulary is kept into the trie data structure, storing for each word a list of its occurrences.
- Each word of the text is read and searched in the trie. If it is found, it is added to the trie with an empty list of occurrences. Once it is in the trie, the new position is added to the end of its list of occurrences. Fig. 2.1.3 shows building an inverted index for the sample text.
- For simplicity, split the index into two files.
  1. First file contains list of occurrences that are stored contiguously. This file is typically called a **posting file.**
  2. Second file : The vocabulary is stored in lexicographical order and for each word; a pointer to its list in the first file is also included. This allows the vocabulary to be kept in memory at search time.

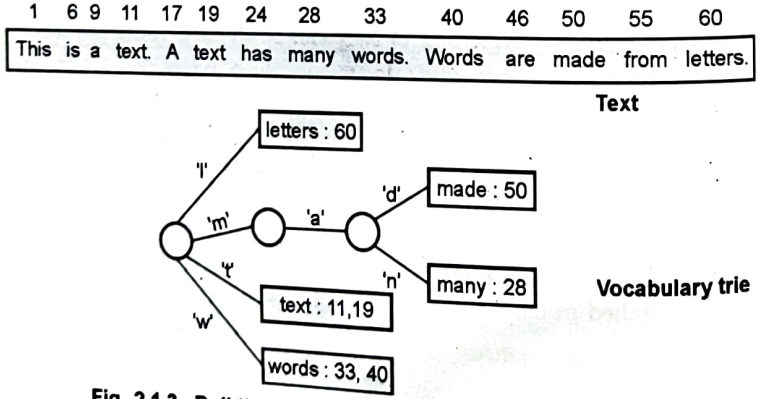


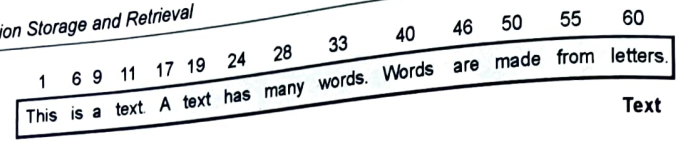
Fig. 2.1.3 Building an inverted index for the sample text

2.2 Suffix Trees and Suffix Arrays

- Suffix trees and suffix arrays are versatile data structures fundamental to string processing applications.
- 1973, Weiner introduced the concept of suffix tree (position tree). In 1990, Gene Myers and Udi Manber proposed suffix array. In 1992, Gonnet, Baeza-Yates and Snider independently discovered suffix array (called PAT array).
- A suffix tree stores all of the substrings of a given string in linear space in a way that the search for a substring is efficient. The time is proportional to the length of the query string. Suffix tree can be built in linear time.
- The suffix tree of "s", denoted ST(s) or simply ST, is a compacted trie of all suffixes of string s. Let |s | = n. It has the following properties :
  1. The tree has n leaves, labelled 1..... n, one corresponding to each suffix of s.
  2. Each internal node has at least 2 children.
  3. Each edge in the tree is labelled with a substring of s.
  4. The concatenation of edge labels from the root to the leaf labelled.
  5. The labels of the edges connecting a node with its children start with different characters.
- Suffix trie are a space-efficient data structure to store a string that allows many kinds of queries to be answered quickly. Suffix trees are hugely important for searching large sequences like genomes. The basis for a tool called "MUMMer" (developed by UMD faculty).
- **Suffix arrays** are a space efficient implementation of suffix trees. This type of index allows us to answer efficiently more complex queries.

2.2.1 Structure

- Suffix tree is a trie data structure built over all the suffixes of the text. The pointers to the suffixes are stored at the leaf nodes. Fig. 2.2.1 shows suffix trie and suffix tree for the sample text.
- Space requirement is main problem with suffix tree and suffix trie. Depending on the implementation, each node of the trie takes 12 to 24 bytes, and therefore even if only word beginnings are indexed, a space overhead of 120 % to 240 % over the text size is produced.
- Suffix arrays provide essentially the same functionality as suffix trees with much less space requirements. If the leaves of the suffix tree are traversed in left to right order, all the suffixes of the text are retrieved in lexicographical order.



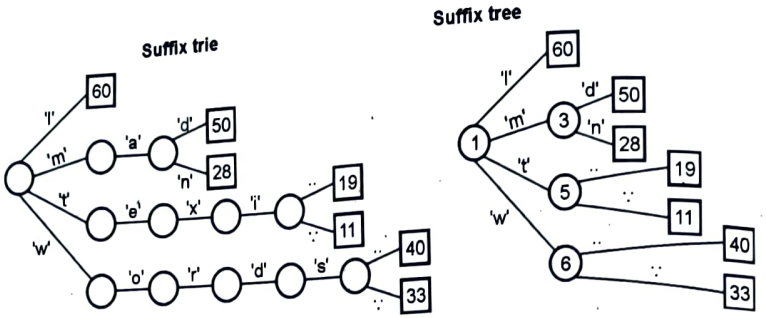
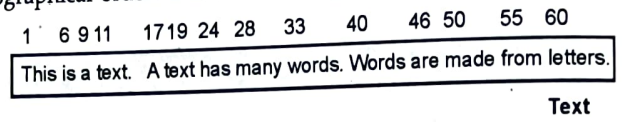


Fig. 2.2.1 Suffix trie and suffix tree for the sample text

- Suffix array is simply an array containing all the pointers to the text suffixes listed in lexicographical order. It is shown in Fig. 2.2.2.



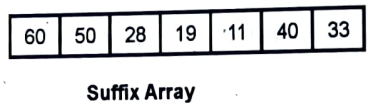
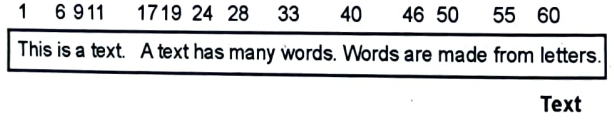


Fig. 2.2.2 Suffix array for simple text

- They store one pointer per indexed suffix, the space requirements are almost the same as those for inverted indices. Suffix arrays are designed to allow binary searches done by comparing the contents of each pointer. If the suffix array is large, this binary search can perform poorly because of the number of random disk accesses.



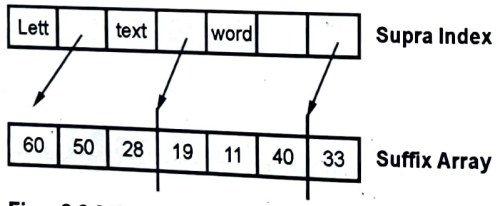


Fig. 2.2.3 Supra index over the suffix array

- To solve this problem, supra indices are used over suffix array. The simplest supra index is used as a first step of the search to reduce external accesses. Fig. 2.2.3 shows supra index over the suffix array.
- Supra index does not in fact need to take samples at fixed intervals, nor to take samples of the same length.

2.2.2 Searching

- Suffix array can perform the same search operations in O(log n) time by doing a binary search instead of a trie search.
- The search pattern originates two limiting patterns P_1 and P_2, so that we want any suffix S such that $P_1 <= S < P_2$. Binary search both limiting patterns in the suffix array. All the elements lying between both positions point to exactly those suffixes that start like the original pattern.
- All these queries retrieve a subtree of the suffix tree or an interval of the suffix array. The result has been collected later, which may imply sorting them in ascending text order. This is complication of suffix trees or arrays with respect to inverted indices.
- Binary search performed on suffix arrays. It is done on disk where the accesses to text positions force a seek operation which spans the disk tracks containing the text. Random seek is O(n) in size, this makes the search cost O(n log n) time.
- Supra indices are used as a first step in any binary search operation to solve this problem. To avoid performing O(n log n) random accesses to the text on disk, the search starts in the supra index, which usually fits in main memory.
- After this search is completed, the suffix array block which is between the two selected samples is brought into the memory and the binary search is completed.

Drawbacks

1. Suffix trees consume a lot of space.
2. It is O(n) but the constant is quite big.
3. Notice that if we indeed want to traverse an edge in O(1) time then we need an array of pointers of size |?| in each node.

2.3 Signature Files

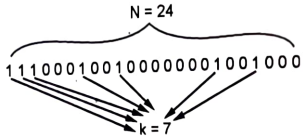
- A "signature" is created as an abstraction of a document. All the signatures that represent the documents in the collection are kept in a file called "signature file".
- Signature files are word oriented index structures based on hashing.

Characteristics

1. Low overhead (10 % ~ 20 % over the text size) at the cost of forcing a sequential search over the index.
2. Suitable for not very large texts.
3. Inverted files outperform signature files for most applications.

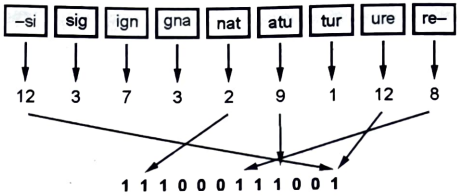
Structure

- A signature file uses a hash function that maps words to bit masks of B bits. It uses *superimposed coding* to create signature. Each text is divided into logical blocks. A block contains n distinct non-common words.
- Each word yields "word signature". A *word signature* is a B-bit pattern, with m 1-bit. Each word is divided into successive, overlapping triplets. e.g. *Free* --> **fr, fre, ree, ee **.
- Each such triplet is hashed to a bit position. The word signatures are OR'ed to form *block signature*. Block signatures are concatenated to form the *document signature*.
- A word signature is a fixed-length bit-string represents a word. It is described by the length (N) and number of bits set to 1(k).



Word Signature Generation

- Use a hash function to find the location of the bit(s) that will be set on. Using triplets of characters to generate word signature.
  a. Divide the word into overlapping triplets.
  b. For each triplet of characters :
     - Convert the characters to a numeric value (can be ASCII representation of the character).
     - Use the number as the input to the hash function.
     - The hash function will produce a number which represent the bit position of the triplet in the word signature.
- **Example :** A signature 111000111001 is generated for the word "signature".
- The position is read from left to right



Example : (n = 2, B = 12, m = 4)

Word	Signature			
free	001	000	110	010
text	000	010	101	001
block signature	001	010	111	001

- For searching it uses hash function to determine the m 1-bit positions. Examine each block signature for 1's bit positions that the signature of the search word has a1.
- **False alarm (false hit or false drop) F_d :** *The probability that a block signature seems to qualify, given that the block does not actually qualify.*

  F_d = **Prob{signature qualifies/block does not}**
- For a given value of B, the value of m that minimizes the false drop probability is such that each row of the matrix contains "1"s with probability 0.5.

  $F_d = 2^{-m}$ **m = B ln2/n**
- False drop occurs when a document's signature matches a query's signature but the query's word does not match any word in the document. It is possible because two distinct blocks may have the same signatures due to the **hashing algorithm and superimposed coding.**
- The rate of false drop depends on :
  1. The size of the signature (N bits)
  2. The size of bits set to 1(K bits)
  3. The number of words per-block

2.4 Scatter Storage or Hash Addressing

- Scatter storage techniques are used to minimize the time required to enter and retrieve information in tables. The technique by which the file structure is implemented is often called hash addressing.

- Suppose that each item consists of an identifying name or key, which may be regarded as an integer, and an associated value.
- If m keys $k_1, \ldots, k_m$ are stored at addresses $a(k_1), \ldots, a(k_m)$ in a table T of length $n \geq m$ and a key k is given, the problem is to determine efficiently whether k is in T, and if so, to find a(k).
- In order to compare the efficiency of different algorithms, we count the number of fetches of elements of T.
- A function that does so is called a hash function. The conversion process is called hashing. The storage structure is called a hash table or scatter-storage.
- The idea of a hash table is to allow many of the different possible keys that might occur to be mapped to the same location in an array under the action of the index function. Others have called scatter-storage or key-transformation.
- A hash function that takes a key and maps it to some index in the array. Often, two records may want to go to the same location. Therefore, a **collision** may occur and a collision procedure must be devised to handle this.
- **Clustered files :** If file processed by a clustering algorithm the it is referred as a clustered file.

2.5 Searching Techniques

- All search strategies are based on comparison between the query and the stored documents. Sometimes this comparison is only achieved indirectly when the query is compared with clusters.
- The distinctions made between different kinds of search strategies can sometimes be understood by looking at the query language that is the language in which the information need is expressed.
- The nature of the query language often dictates the nature of the search strategy. For example, a query language which allows search statements to be expressed in terms of logical combinations of keywords normally dictates a Boolean search. This is a search which achieves its results by logical comparisons of the query with the documents.

2.5.1 Boolean Search

- Boolean searches are named after the British born Mathematician George Boole. Boolean logic establishes the relationship between keywords in a search. Boolean logic has three operators : **AND, OR , NOT.**

- A Boolean search strategy retrieves those documents which are 'true' for the query. This formulation only makes sense if the queries are expressed in terms of index terms or keywords and combined by the usual logical connectives AND, OR and NOT.
- For example, if the query Q = (K_1 AND K_2) OR (K_3 AND (NOT K_4)) then the Boolean search will retrieve all documents indexed by K_1 and K_2, as well as all documents indexed by K_3 which are not indexed by K_4.

1. Search keywords Iceland AND Whales

This search will give you all pages with both the keyword Iceland and Whales contained in the page or metadata. This is the basic search that most people use on search engines like Google. This search will find any page that has the words Iceland and **Whales** in the page. The search engines usually look for words within the body of the page or in the Metadata in the head of the page.

2. Search keywords Iceland OR Whales

This search will give you all pages with either the keyword Iceland or Whales contained in the page or metadata. This search will produce many hits as it looks for pages with **either of the two keywords** within the body of the page or in the Metadata in the head. The results will include :

a. Pages with "Whales"

b. Pages with "Iceland"

c. Pages containing both "Whales" and "Iceland"

This style of Boolean search is useful when you have related topics or where there are several names for the same topics. The OR operator is used when the thing you are looking for has more than one legitimate name. For example : "Standards of Learning" is also called "SOL."

3. Search keywords Iceland NOT Whales

This search will give you all pages with the keyword Iceland but not the keyword Whales contained in the page or metadata. The Not search is a powerful filtering tool that allows the searcher to eliminate pages quickly and easily. It is particularly useful when combined with other search terms.

- On some systems, the Boolean search allow the user to narrow or broaden the search by giving the user access to a structured dictionary which, for any given keyword, stores related keywords which may be more general or more precise.
- Consider example of tree structure. Following Fig. 2.5.1 shows a set of hierarchically related keywords.

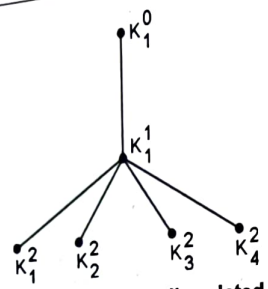


Fig. 2.5.1 Set of hierarchically related keywords

- The keyword K_1^1 is contained in the more general keyword K_1^0, but it can also be split up into the 4 more precise keywords K_1^2, K_2^2, K_3^2 and K_4^2. If one has an interactive system the search can easily be reformulated using some of these related terms.
- Boolean search is implemented using inverted file. An inverted file is a list of keywords and identifiers of the documents in which they occur. List for each keyword is stored in the vocabulary and in each list put the addresses or numbers of the documents containing that particular word.
- To satisfy a query, the set operations is performed on the K_i - lists corresponding to the logical connectives. For example :

  K_1 - List : D_1, D_2, D_3, D_4

  K_2 - List : D_1, D_2

  K_3 - List : D_1, D_2, D_3

  K_4 - List : D_1

  and $Q = (K_1 \text{ AND } K_2) \text{ OR } (K_3 \text{ AND } (\text{NOT } K_4))$

  then to satisfy the $(K_1 \text{ AND } K_2)$ part we intersect the K_1 and K_2 lists, to satisfy the $(K_3 \text{ AND } (\text{NOT } K_4))$ part we subtract the K_4 list from the K_3 list.
- The OR is satisfied by now taking the union of the two sets of documents obtained for the parts. The result is the set $\{D_1, D_2, D_3\}$ which satisfies the query and each document in it is 'true' for the query.
- Full Boolean search is allowed for slight modification by using only AND logic. It considers the ac actual number of terms the query has in common with a document. This number has become known as the **co-ordination level**. This is also called **simple matching**. The documents are partially ranked by the co-ordination levels.
- For example : Query $Q = K_1 \text{ AND } K_2 \text{ AND } K_3$ we obtain the following ranking :

Co-ordination level

3	D_1, D_2
2	D_3
1	D_4

- A general problem with Boolean search is that using AND operators tends to produce high precision but low recall searches, while using OR operators gives low precision but high recall searches, and it is difficult or impossible to find a satisfactory middle ground.

Advantages of Boolean Search

1. Simple queries are easy to understand.
2. Relatively easy to implement.

Disadvantages of Boolean Search

1. Difficult to specify what is wanted.
2. Too much returned, or too little.
3. Ordering not well determined.

2.5.2 Serial Search

- If suppose there are N documents D_i in the system, then the serial search calculates N values $M(Q, D_i)$ the set of documents to be retrieved is determined. Two methods used for this purposes.
  1. The matching function is given a suitable threshold, retrieving the documents above the threshold and discarding the ones below. If T is the threshold, then the retrieved set B is the set $\{D_i \mid M(Q, D_i) > T\}$.
  2. The documents are ranked in increasing order of matching function value. A rank position R is chosen as cut-off and all documents below the rank are retrieved so that $B = \{D_i \mid r(i) < R\}$ where r(i) is the rank position assigned to D_i. The hope in each case is that the relevant documents are contained in the retrieved set.

2.5.3 Cluster-based Retrieval

- **Clustering** is the process of grouping a set of objects into classes of similar objects. Documents within a cluster should be similar. Documents from different clusters should be dissimilar. Clustering picks out closely associated documents and groups them together into one cluster.

- **Cluster hypothesis :** Documents in the same cluster behave similarly with respect to relevance to information needs.
- Clustering is now considered as a useful method in information retrieval for not only documents categorization but also interactive retrieval. The use of clustering in information retrieval is based on the Clustering Hypothesis "closely associated documents tend to be relevant to the same requests".

Working :

- The search starts from root node 0. It then calculates the matching function at the nodes immediately descendant from node 0 i.e. nodes 1 and 2. Fig. 2.6.2 shows search tree and the appropriate values of a matching function.
- This pattern repeats itself down the tree. The search is directed by a decision rule, which on the basis of comparing the values of a matching function at each stage decides which node to expand further. Also, it is necessary to have a stopping rule which terminates the search and forces retrieval. This search strategy is of the **top-down** searches.

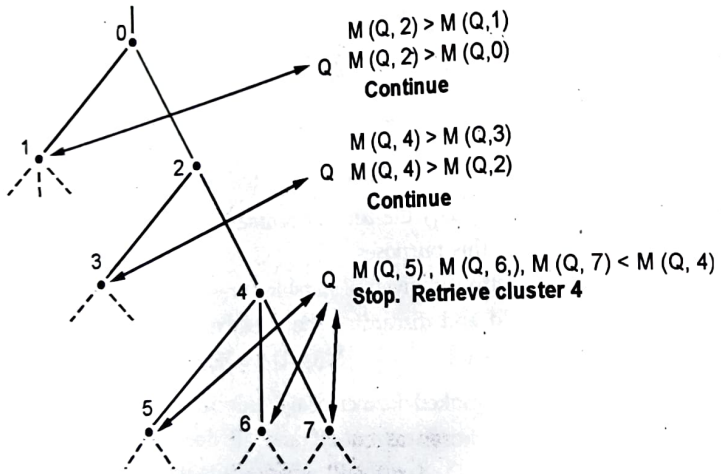


Fig. 2.5.2 Search tree and the appropriate values of a matching function

- In above figure, the **decision rule is :** Expand the node corresponding to the maximum value of the matching function achieved within a filial set.
- The **stopping rule is :** Stop if the current maximum is less than the previous maximum. A few remarks about this strategy are in order.
- Following points are consider for this search :
  1. It is assumed that effective retrieval can be achieved by finding just one cluster.

  2. Each cluster can be adequately represented by a cluster representative for the purpose of locating the cluster containing the relevant documents.
  3. If the maximum of the matching function is not unique some special action, such as a look-ahead, will need to be taken.
  4. The search always terminates and will retrieve at least one document.
- A **bottom-up search** is one which enters the tree at one of its terminal nodes, and proceeds in an upward direction towards the root of the tree. It will pass through a sequence of nested clusters of increasing size. A decision rule is not required; we only need a stopping rule which could be simply a cut-off.
- One of **the advantages** of cluster-based IR is that users can browse the documents of the best matching clusters. This may give them the opportunity of seeing additional relevant documents not ranked highly and that may or may not have common term with the submitted query.
- **Hard clustering :** Each document belongs to exactly one cluster. It is more common and easier to do clustering.
- **Soft clustering :** A document can belong to more than one cluster. It makes more sense for applications like creating browsable hierarchies. You may want to put a pair of sneakers in two clusters : i) Sports apparel and ii) Shoes. You can only do that with a soft clustering approach.

2.6 Sequential Search

- The sequential search is the simplest type of search, it is used when a list of integers is not in any order. It examines the first element in the list and then examines each "sequential" element in the list until a match is found.

2.6.1 Brute Force

- Brute force is a straightforward approach to solving a problem, usually directly based on the problem statement and definition.
- The brute force method moves the pattern only one character at each comparison. The method does not benefit from any information about which characters the pattern contains.
- It does not need any pattern preprocessing. Many algorithms use a modification of this scheme.
- Example :

  document S : abracabracadabra

  pattern P : abracadabra

- After applying Brute force method :

abracabracadabra

abracadabra

abracadabra

abracadabra

abracadabra

abracadabra

abracadabra

- Example :

Document S : TWO ROADS DIVERGED IN A YELLOW WOOD

Pattern P : ROADS

- The Brute Force algorithm compares the pattern to the text, one character at a time, until unmatching characters are found :

TWO ROADS DIVERGED IN A YELLOW WOOD *ROADS*
TWO ROADS DIVERGED IN A YELLOW WOOD *ROADS*
TWO ROADS DIVERGED IN A YELLOW WOOD *ROADS*
TWO ROADS DIVERGED IN A YELLOW WOOD *ROADS*
TWO ***ROADS*** DIVERGED IN A YELLOW WOOD ***ROADS***

Note : Compared characters are italicized. Correct matches are in boldface type.

2.6.2 Kunth-Morris Pratt

- Kunth-Morris Pratt (KMP) algorithm is one of the string matching algorithms used to find a pattern in a text.
- The basic idea behind this algorithm is that each time a mismatch is detected, the "false start" consists of characters that we have already examined. We can take advantage of this information instead of repeating comparisons with the known characters. Moreover, it is always possible to arrange the algorithm so that the pointer in the text is never decremented.
- **Example 1 :** The next table for the pattern abracadabra is,

	a	b	r	a	c	a	d	a	b	r	a	
Next[i]	0	1	1	0	2	0	2	0	1	1	0	5

- When the value in the next table is zero, we have to advance one position in the text and start comparing again from the beginning of the pattern. The last value of the next table (five) is used to continue the search after a match has been found.

2.6.3 Boyer-Moore Family

- The Boyer-Moore (BM) algorithm positions the pattern over the leftmost characters in the text and attempts to match it from right to left. If no mismatch occurs, then the pattern has been found. Otherwise, the algorithm computes a shift; that is, an amount by which the pattern is moved to the right before a new matching attempt is undertaken.
- Main idea is to search from right to left in the pattern. With this scheme, searching is faster than average.
- Two methods are used for shifting the pattern in relation to the text :

a) Matching shift

b) Occurrence shift

- The BM algorithm scans the characters of the pattern from right to left beginning with the rightmost one and performs the comparisons from right to left. In case of a mismatch, it uses two pre-computed functions to shift the window to the right.
- BM algorithm uses good-suffix function and bad-char function to calculate the new comparing position, shifting rightward P by taking maximum of these two values.
- BM algorithm is fast in the case of larger alphabet. It uses the information gained from that attempt to rule out as many positions of the text as possible where the string could not match.

2.7 Query Languages

- There are a number of techniques to enhance the usefulness of the queries. Some examples are the expansion of a word to the set of its synonyms or the use of a thesaurus. Some words which are very frequent and do not carry meaning may be removed. We refer to words that can be used to match query terms as keywords.
- Another issue is the subject of the retrieval unit the information retrieval system adopts. The retrieval unit is the basic element which can be retrieved as an answer to a query. We call the retrieval units simply documents, even if this reference can be used with different meanings.

2.7.1 Types of Queries

- **Keyword Based Querying :** A query is the formulation of a user information need. Keyword based queries are popular, since they are intuitive, easy to express,

and allow for fast ranking. However, a query can also be a more complex combination of operations involving several words.

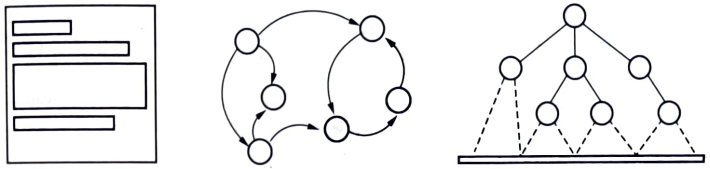
- **Word Queries :** The most elementary query that can be formulated in a text retrieval system is the word. Some models are also able to see the internal division of words into letters. The division of the text into words is not arbitrary, since words carry a lot of meaning in natural language. The result of word queries is the set of documents containing at least one of the words of the query. The resulting documents are ranked according to the degree of similarity with respect to the query. To support ranking, two common statistics on word occurrences inside texts are commonly used (term frequency and inverse document frequency).
- **Context Queries :** Many systems complement queries with the ability to search words in a given context. Words which appear near each other may signal higher likelihood of relevance than if they appear apart.
- **Boolean Queries :** The oldest way to combine keyword queries is to use boolean operators. A boolean query has a syntax composed of atoms and operators. Atoms are basic queries that retrieve documents and boolean operators work on their operands and deliver sets of documents.

2.7.2 Patterns Matching

- A pattern is a set of syntactic features that must be found in a text segment. Those segments satisfying the pattern specifications are said to match the pattern.
- We can search for documents containing segments which match a given search pattern. Each system allows specifying some types of patterns. The more powerful the set of patterns allowed the more involved queries can the user formulates.
- **The most used types of patterns are :**
  1. Words : A string which must be a word in the text.
  2. Prefixes : A string which must form the beginning of a text word.
  3. Suffixes : A string which must form the termination of a text word.
  4. Substrings : A string which can appear within a text word.
  5. Ranges : A pair of strings which matches any word which lexicographically lies between them.
  6. Allowing errors : A word together with an error threshold.
  7. Regular expressions : A rather general pattern built up by simple strings.
  8. Extended patterns : A more user-friendly query language to represent some common cases of regular expressions.

2.7.3 Structural Queries

- The text collections tend to have some structure built into them. The standardization of languages to represent structured texts has pushed forward in this direction.
- Mixing contents and structure in queries allows posing very powerful queries. Queries can be expressed using containment, proximity or other restrictions on the structural element.
- The three main types of structures :
  a) Form-like fixed structure b) Hypertext structure c) Hierarchical structure
- Fig. 2.7.1 shows types of structures.



(a) Form-like fixed structure (b) Hypertext structure (c) Hierarchical structure

Fig. 2.7.1

2.8 IR Models

SPPU : May-19, June-19, March-20

- An information retrieval system is a software program that stores and man-ages information on documents, often textual documents but possibly multimedia.
- Reason behind use of models in information retrieval is that models guide research and provide the means for academic discussion. Models can also serve as a blueprint to implement an actual retrieval system.
- A model of information retrieval predicts and explains what a user will find relevant given the user query. The correctness of the model's predictions can be tested in a controlled experiment.

2.8.1 Basic Concept

- Each document represented by a set of representative keywords or index terms. An index term is a document word useful for remembering the document main themes.
- Usually, index terms are nouns because nouns have meaning by themselves. However, search engines assume that all words are index terms.

- Not all terms are equally useful for representing the document contents: less frequent terms allow identifying a narrower set of documents. The importance of the index terms is represented by weights associated to them.
- Model is an idealization or abstraction of an actual process. Mathematical models are used to study the properties of the process, draw conclusions and make predictions. Conclusions derived from a model depend on whether the model is a good approximation of the actual situation. Statistical models represent repetitive processes; make predictions about frequencies of interesting events.

2.8.2 Boolean Model

- The Boolean model is the first model of information retrieval and probably also the most criticized model. It is based on set theory and Boolean algebra.
- It is based on a binary decision criterion without any notion of a grading scale. Boolean expressions have precise semantics. It is not simple to translate an information need into a Boolean expression. It can be represented as a disjunction of conjunction vectors(in disjunctive normal form-DNF).
  - **D :** Set of words (indexing terms) present in a document. Each term is either present (1) or absent (0).
  - **Q :** A Boolean expression. The terms are index terms and operators are AND, OR and NOT.
  - **F :** Boolean algebra over sets of terms and sets of documents.
  - **R :** A document is predicted as relevant to a query expression if and only if it satisfies the query expression

    $(($ *text* $\vee$ *information)* $\wedge$ *retrieval* $\wedge \neg$ *theory)*
- Each query term specifies a set of documents containing the term :
  a. AND ($\wedge$) : The intersection of two sets
  b. OR ($\vee$) : The union of two sets
  c. NOT ($\neg$) : Set inverse, or really set difference.

Boolean Relevance example :

$(($ *text* $\vee$ *information)* $\wedge$ *retrieval* $\wedge \neg$ *theory)*

It gives following list :

"Information Retrieval"

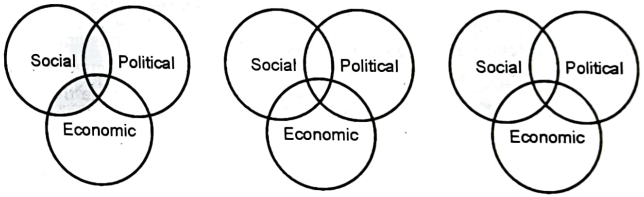
"Information Theory"

"Modern Information Retrieval: Theory and Practice"

"Text Compression"

Implementing the Boolean Model :

- First, consider purely conjunctive queries $(t_a \wedge t_b \wedge t_c)$.
- It only satisfied by a document containing all three terms.
- If $D(t_a) = \{d \mid t_a \in d\}$, then the *maximum possible* size of the retrieved set is the size of the smallest $D(t_a)$.
- $|D(t_a)|$ is the length of the inverted list for t_a.
- For instance, the query **social AND economic** will produce the set of documents that are indexed both with the term social and the term economic, i.e. the intersection of both sets. Combining terms with the OR operator will define a document set that is bigger than or equal to the document sets of any of the single terms. So, the query social OR political will produce the set of documents that are indexed with either the term social or the term political or both, i.e. the union of both sets. This is visualized in the Venn diagrams of Fig. 2.8.1 in which each set of documents is visualized by a disc.



Social AND Economic **Social OR Political** **(Social OR Political) AND NOT (Social AND Economic)**

Fig. 2.8.1 Boolean combinations of sets visualized as Venn diagrams

- The intersections of these discs and their complements divide the document collection into 8 non-overlapping regions, the unions of which give 256 different Boolean combinations of "social, political and economic documents". In Fig. 2.8.1 the retrieved sets are visualized by the shaded areas.

Advantages :

a. Clean formalism
b. Simplicity
c. It is very precise in nature. The user exactly gets what is specified.
d. Boolean model is still widely used in small scale searches like searching emails, files from local hard drives or in a mid-sized library.

Disadvantages :

a. It is not simple to translate an information need into a Boolean expression.

b. Exact matching may lead to retrieval of too few or too many documents.

c. The retrieved documents are not ranked.

d. The model does not use term weights.

2.8.3 Vector Model

- Assign non-binary weights to index terms in queries and in documents. Compute the similarity between documents and query. The index terms in the query are also weighted

$$\bar{d}_j = (w_{1,j}, w_{2,j}, \ldots w_{t,j})$$

$$\bar{q} = (w_{1,q}, w_{2,q}, \ldots w_{t,q})$$

- Term weights are used to compute the degree of similarity between documents and the user query. Then, retrieved documents are sorted in decreasing order.
- They considered the index representations and the query as vectors embedded in a high dimensional Euclidean space, where each term is assigned a separate dimension. The similarity measure is usually the cosine of the angle that separates the two vectors $\bar{d}$ and $\bar{q}$.
- The cosine of an angle is 0 if the vectors are orthogonal in the multidimensional space and 1 if the angle is 0 degrees. The cosine formula is given by :

$$\text{score}(\bar{d}, \bar{q}) = \frac{\sum_{k=1}^{m} d_k \cdot q_k}{\sqrt{\sum_{k=1}^{m} (d_k)^2} \cdot \sqrt{\sum_{k=1}^{m} (q_k)^2}}$$

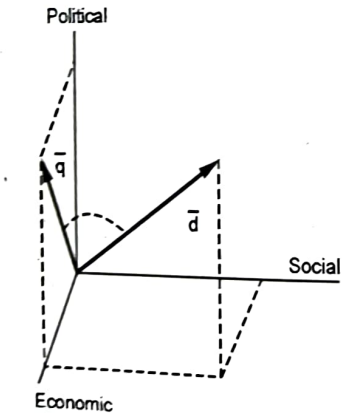


Fig. 2.8.2 Query and document representation in the vector space model

- The metaphor of angles between vectors in a multidimensional space makes it easy to explain the implications of the model to non-experts. Up to three dimensions, one can easily visualise the document and query vectors. Fig. 2.8.2 shows a query and document representation in the vector space model.
- Measuring the cosine of the angle between vectors is equivalent with normalizing the vectors to unit length and taking the vector inner product. If index representations and queries are properly normalised, then the vector product measure of equation 1 does have a strong theoretical motivation. The formula then becomes :

$$\text{score}(\bar{d}, \bar{q}) = \sum_{k=1}^{m} n(d_k) \cdot n(q_k)$$

where $n(v_k) = \frac{v_k}{\sqrt{\sum_{k=1}^{m} (v_k)^2}}$

- We think of the documents as a collection C of objects and think of the user query as a specification of a set A of objects. In this scenario, the IR problem can be reduced to the problem of determine which documents are in the set A and which ones are not (i.e. the IR problem can be viewed as a clustering problem).
  1. **Intra-cluster :** One needs to determine what are the features which better describe the objects in the set A.
  2. **Inter-cluster :** One needs to determine what are the features which better distinguish the objects in the set A.

tf : Inter-clustering similarity is quantified by measuring the raw frequency of a term k_i inside a document d_j, such term frequency is usually referred to as the tf factor and provides one measure of how well that term describes the document contents.

idf : Inter-clustering similarity is quantified by measuring the inverse of the frequency of a term k_i among the documents in the collection. This frequency is often referred to as the **inverse document frequency.**

Advantages :

1. Its term-weighting scheme improves retrieval performance.
2. Its partial matching strategy allows retrieval of documents that approximate the query conditions.
3. Its cosine ranking formula sorts the documents according to their degree of similarity to the query.

Disadvantages :

1. The assumption of mutual independence between index terms.
2. It cannot denote the "clear logic view" like Boolean model.

2.8.4 Probabilistic Model

- This model is introduced by Roberston and Sparck Jones in 1976. It is also called Binary Independence Retrieval (BIR) model.
- Idea : Given a user query q, and the ideal answer set R of the relevant documents, the problem is to specify the properties for this set.
- Assumption (probabilistic principle) : *The probability of relevance depends on the query and document representations only; ideal answer set R should maximize the overall probability of relevance.*
- The probabilistic model tries to estimate the probability that the user will find the document d_j relevant with ratio P(d_j relevant to q)/P(d_j nonrelevant to q)

Definition

- All index term weights are all binary i.e. $w_{i,j} \in \{0,1\}$
- Let R be the set of documents known to be relevant to query q
- Let $\bar{R}$ be the complement of R.
- Let $P(R|\vec{d}_j)$ be the probability that the document d_j is relevant to the query q
- Let $P(\bar{R}|\vec{d}_j)$ be the probability that the document d_j is nonelevant to query q
- The similarity sim(d_j, q) of the document d_j to the query q is defined as the ratio

$$sim(\vec{d}_j, q) = \frac{Pr(R|\vec{d}_j)}{Pr(\bar{R}|\vec{d}_j)}$$

- Using Bayes' rule

$$sim(\vec{d}_j, q) = \frac{Pr(\vec{d}_j|R) \times Pr(R)}{Pr(\vec{d}_j|\bar{R}) \times Pr(\bar{R})}$$

where :

a. P(R) stands for the probability that a document randomly selected from the entire collection is relevant.

b. $P(\vec{d}_j|R)$ stands for the probability of randomly selecting the document d_j from the set R of relevant documents.

$$sim(\vec{d}_j, q) = \frac{Pr(\vec{d}_j|R)}{Pr(\vec{d}_j|\bar{R})} + \log \frac{Pr(R)}{Pr(\bar{R})}$$

- Assuming independence of index terms and given $q = (d_1, d_2, \ldots, d_t)$,

$$Pr(\vec{d}_j|R) = \prod_{i=1}^{t} pr(k_i = d_i|R)$$

$$Pr(\vec{d}_j|\bar{R}) = \prod_{i=1}^{t} pr(k_i = d_i|\bar{R})$$

$$sim(\vec{d}_j, q) = \log \frac{\prod_{i=1}^{t} pr(k_i = d_i|R)}{\prod_{i=1}^{t} pr(k_i = d_i|\bar{R})}$$

- $Pr(k_i|R)$ stands for the probability that the index term k_i is present in a document randomly selected from the set R.
- $Pr(\bar{k}_i|R)$ stands for the probability that the index term k_i is not present in a document randomly selected from the set R

$$sim(\vec{d}_j, q) = \frac{\prod_{g_i(d_j=1)} Pr(k_i|R) \prod_{g_i(d_j=1)} Pr(\bar{k}_i|R)}{\prod_{g_i(d_j=1)} Pr(k_i|\bar{R}) \prod_{g_i(d_j=1)} Pr(\bar{k}_i|\bar{R})}$$

$\because \quad Pr(\vec{d}_j|R) + Pr(\vec{d}_j|\bar{R}) = 1$

$$sim(\vec{d}_j, q) = \sum_{i=1}^{t} \left(\log \frac{p(k_i|R)}{1-p(k_i|R)} + \log \frac{1-p(k_i|\bar{R})}{p(k_i|\bar{R})} \right)$$

$$sim(\vec{d}_j, q) = \sum_{i=1}^{t} w_{i,q} \times w_{i,j} \left(\log \frac{p(k_i|R)}{1-p(k_i|R)} + \log \frac{1-p(k_i|\bar{R})}{p(k_i|\bar{R})} \right)$$

Advantage : Documents are ranked in decreasing order of their probability of being relevant.

Disadvantages :

1. The need to guess the initial relevant and non-relevant sets.
2. Term frequency is not considered.
3. Independence assumption for index terms.

2.8.5 Comparison between Boolean Model and Vector Model

Sr. No.	Boolean model	Vector model
1.	Based on the notion of sets.	Based on geometry, the notion of vectors in high dimensional space.
2.	The retrieved documents are not ranked.	Documents are ranked based on their similarity to the query.
3.	Also known as 'exact-match' retrieval models.	Best/partial match.
4.	The user describes their information need using Boolean constraints (e.g., AND, OR, and AND NOT).	Formally, a vector space is defined by a set of linearly independent basis vectors.
5.	The user exactly gets what is specified.	It cannot denote the clear logic view like Boolean model.
6.	Boolean model cannot used to retrieve partial matching documents.	Vector model can be used to retrieve partial matching documents.

Example 2.8.1 *Find the similarity of following query with D_1, D_2, D_3 using vector model.*

Query	keywords	
q	ant, dog	
document	Text	Terms
D_1	ant ant bee	ant bee
D_2	dog bee dog hog dog ant dog	ant bee dog hog
D_3	cat gnu dog eel fox	cat dog eel fox gnu

SPPU : May-19, In Sem, Marks 5

Solution : The total number of documents is N = 3. Therefore, the idf values for the terms are :

ant $\rightarrow \log_2(3/3) = 0$

bee $\rightarrow \log_2(3/2) = 0.584$

cat $\rightarrow \log_2(3/1) = 1.584$

dog $\rightarrow \log_2(3/5) = -0.736$

eel $\rightarrow \log_2(3/1) = 1.584$

fox $\rightarrow \log_2(3/1) = 1.584$

gnu $\rightarrow \log_2(3/1) = 1.584$

hog $\rightarrow \log_2(3/1) = 1.584$

- For all the documents, we calculate the tf scores for all the terms in C. We assume the words in the vectors are ordered alphabetically.

	ant	bee	cat	dog	eel	fox	gnu	hog	length
D_1	1	1	0	0	0	0	0	0	$\sqrt{2}$
D_2	1	1	0	1	0	0	0	1	$\sqrt{4}$
D_3	0	0	1	1	1	1	1	0	$\sqrt{5}$

dot product

	d_1	d_2	d_3
d_1	2	**2**	0
d_2	2	5	2
d_3	0	**2**	5

cosin angle

	d_1	d_2	d_3
d_1	1	**0.63**	0
d_2	0.63	1	0.40
d_3	0	**0.40**	1

	ant	bee	cat	dog	eel	fox	gnu	hog
d_1	2	1						
d_2	1	1		4		1		1
d_3			1	1	1	1	1	

	ant	bee	cat	dog	eel	fox	gnu	hog	length
q	1			1					$\sqrt{2}$
D_1	2	1	0	0	0	0	0	0	$\sqrt{5}$
D_2	1	1	0	4	0	0	0	1	$\sqrt{19}$
D_3	0	0	1	1	1	1	1	0	$\sqrt{5}$

	d_1	d_2	d_3
q	$2/\sqrt{10}$	$5/\sqrt{38}$	$1/\sqrt{10}$
	0.63	0.81	0.32

University Questions

1. *Compare Boolean and vector model.* **SPPU : June-19, End Sem, Marks 6**
2. *Write a short note on probabilistic model, vector model.* **SPPU : March-20, In Sem, Marks 4**

❑❑❑