DIGITAL NOTES
ON
INFORMATION RETRIEVAL SYSTEMS
(R17A1209)

B.TECH IV YEAR - I SEM
(2020-2021)

DEPARTMENT OF INFORMATION TECHNOLOGY

MALLA REDDY COLLEGE OF ENGINEERING & TECHNOLOGY
(Autonomous Institution – UGC, Govt. of India)
(Affiliated to JNTUH, Hyderabad, Approved by AICTE - Accredited by NBA & NAAC – 'A' Grade - ISO 9001:2015 Certified)
Maisammaguda, Dhulapally (Post Via. Hakimpet), Secunderabad – 500100, Telangana State, INDIA.

IV Year B.Tech IT –I Sem **L T /P/D C**
3 -/-/- 3

(R17A1209)INFORMATION RETRIEVAL SYSTEMS
(Core Elective IV)

OBJECTIVES

- Study fundamentals of DBMS, Data warehouse and Digital libraries
- Learn various preprocessing techniques and indexing approaches in text mining
- Know various clustering approaches and study different similarity measures
- Study various search techniques in information retrieval systems
- Know different cognitive approaches used in text retrieval systems and evaluation approaches
- Study retrieval in multimedia systems and know various evaluation measures
- Know about query languages and online IRsystem

UNIT-I

Introduction: Definition, Objectives, Functional Overview, Relationship to DBMS, Digital libraries and Data Warehouses.
Information Retrieval System Capabilities: Search, Browse, Miscellaneous

UNIT-II

Cataloging and Indexing: Objectives, Indexing Process, Automatic Indexing, Information Extraction. **Data Structures:** Introduction, Stemming Algorithms, Inverted file structures, N-gram data structure, PAT data structure, Signature file structure, Hypertext data structure.

UNIT-III

Automatic Indexing: Classes of automatic indexing, Statistical indexing, Natural language, Concept indexing, Hypertext linkages
Document and Term Clustering: Introduction, Thesaurus generation, Item clustering, Hierarchy of clusters.

UNIT-IV

User Search Techniques: Search statements and binding, Similarity measures and ranking, Relevance feedback, Selective dissemination of information search, weighted searches of Boolean systems, Searching the Internet and hypertext.
Information Visualization: Introduction, Cognition and perception, Information visualization technologies.

UNIT-V

Text Search Algorithms: Introduction, Software text search algorithms, Hardware text search systems.
Information System Evaluation: Introduction, Measures used in system evaluation, Measurement example – TREC results.

TEXTBOOK:

1. Information Storage and Retrieval Systems: Theory and Implementation by Gerald J. Kowalski, Mark T. Maybury , Second Edition, Kluwer Academic Publishers.

REFERENCES:

1. Frakes, W.B., Ricardo Baeza-Yates: Information Retrieval Data Structures and Algorithms, Prentice Hall, 1992.
2. Modern Information Retrival By Yates Pearson Education.
3. Information Storage & Retieval By Robert Korfhage – John Wiley & Sons.

OUTCOMES:

Upon completion of the course, the students are expected to:

1. Recognize the Boolean Model, Vector Space Model, and Probabilistic Model.
2. Understand retrieval utilities.
3. Understand different formatting tags
4. Understand cross-language information retrieval
5. Understand the clustering techniques
6. Determine the efficiency.

INDEX

UNIT-I

Introduction: Definition, Objectives, Functional Overview, Relationship to DBMS, Digital libraries and Data Warehouses. **Information Retrieval System Capabilities:** Search, Browse, Miscellaneous

Introduction

There is a potential for confusion in the understanding of the differences between Database Management Systems (DBMS) and Information Retrieval Systems. It is easy to confuse the software that optimizes functional support of each type of system with actual information or structured data that is being stored and manipulated. The importance of the differences lies in the inability of a database management system to provide the functions needed to process "information." The opposite, an information system containing structured data, also suffers major functional deficiencies.

Definition of Information Retrieval System

An Information Retrieval System is a system that is capable of storage, retrieval, and maintenance of information6. Information in this context can be composed of text (including numeric and date data), images, audio, video and other multi-media objects. Techniques are beginning to emerge to search these other media types (e.g., EXCALIBUR's Visual RetrievalWare, VIRAGE video indexer).

The term "item" is used to represent the smallest complete unit that is processed and manipulated by the system. The definition of item varies by how a specific source treats information. A complete document, such as a book, newspaper or magazine could be an item. For example a video news program could be considered an item. It is composed of text in the form of closed captioning, audio text provided by the speakers, and the video images being displayed.

An Information Retrieval System consists of a software program that facilitates a user in finding the information the user needs. The system may use standard computer hardware or specialized hardware to support the search sub function and to convert non-textual sources to a searchable media (e.g., transcription of audio to text). Thus search composition, search execution, and reading non-relevant items are all

aspects of information retrieval overhead.

With the advent of inexpensive powerful personnel computer processing systems and high speed, large capacity secondary storage products, it has become commercially feasible to provide large textual information databases for the average user. The introduction and exponential growth of the Internet along with its initial WAIS (Wide Area Information Servers) capability and more recently advanced search servers (e.g., INFOSEEK, EXCITE) has provided a new avenue for access to terabytes of information (over 800 million indexable pages -Lawrence-99.) The algorithms and techniques to optimize the processing and access of large quantities of textual data were once the sole domain of segments of the Government, a few industries, and academics.

Images across the Internet are searchable from many web sites such as WEBSEEK, DITTO.COM, ALTAVISTA/IMAGES. News organizations such as the BBC are processing the audio news they have produced and are making historical audio news searchable via the audio transcribed versions of the news. Major video organizations such as Disney are using video indexing to assist in finding specific images in their previously produced videos to use in future videos or incorporate in advertising.

Objectives of Information Retrieval Systems

The general objective of an Information Retrieval System is to minimize the overhead of a user locating needed information. Overhead can be expressed as the time a user spends in all of the steps leading to reading an item containing the needed information (e.g., query generation, query execution, scanning results of query to select items to read, reading non-relevant items).

In information retrieval the term “relevant” item is used to represent an item containing the needed information. From a user’s perspective “relevant” and “needed” are synonymous.

The two major measures commonly associated with information systems are

1) Precision

2) Recall

When a user decides to issue a search looking for information on a topic, the total database is logically divided into four segments shown in Figure 1.1. Relevant items

are those documents that contain information that helps the searcher in answering his question. Non-relevant items are those items that do not provide any directly useful information. There are two possibilities with respect to each item: it can be retrieved or not retrieved by the user's query. Precision and recall are defined as:

Precision is directly affected by retrieval of non-relevant items and drops to a number close to zero. Recall is not effected by retrieval of non-relevant items and thus remains at 100 per cent once achieved.

Information Retrieval Systems such as RetrievalWare, TOPIC, AltaVista, Infoseek and INQUERY that the idea of accepting natural language queries is becoming a standard system feature. This allows users to state in natural language what they are interested in finding. But the completeness of the user specification is limited by the user's willingness to construct long natural language queries. Most users on the Internet enter one or two search terms.

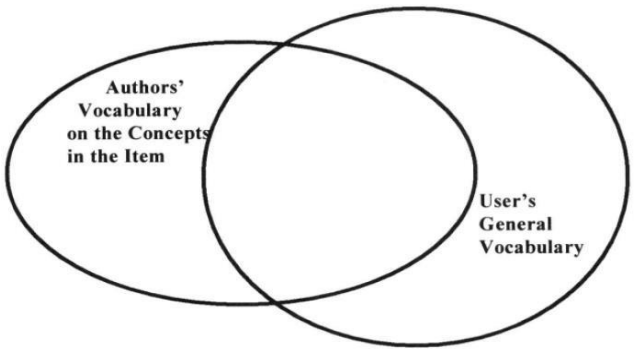


Figure 1.3 Vocabulary Domains

Functional Overview

A total Information Storage and Retrieval System is composed of four major functional processes:

1) Item Normalization

2) Selective Dissemination of Information (i.e., "Mail")

3) Archival Document Database Search, and an Index

4) Database Search along with the Automatic File Build process that supports Index Files.

1) Item Normalization:

The first step in any integrated system is to normalize the incoming items to a standard format. Item normalization provides logical restructuring of the item. Additional operations during item normalization are needed to create a searchable data structure: identification of processing tokens (e.g., words), characterization of the tokens, and stemming (e.g., removing word endings) of the tokens.

The processing tokens and their characterization are used to define the searchable text from the total received text. Figure 1.5 shows the normalization process. Standardizing the input takes the different external formats of input data and performs the translation to the formats acceptable to the system. A system may have a single format for all items or allow multiple formats. One example of standardization could be translation of foreign languages into Unicode. Every language has a different internal binary encoding for the characters in the language. One standard encoding that covers English, French, Spanish, etc. is ISO-Latin.

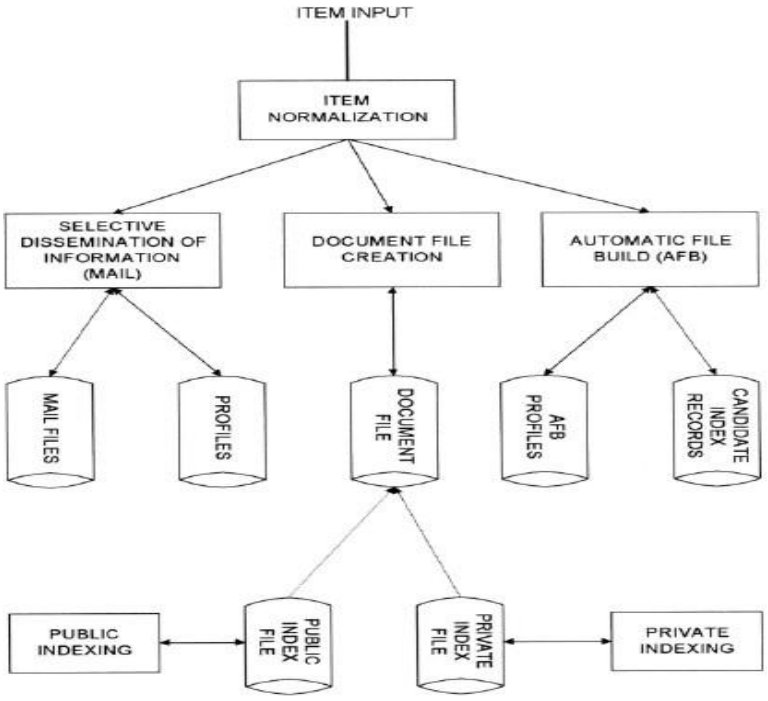


Figure 1.4 Total Information Retrieval System

To assist the users in generating indexes, especially the professional indexers, the system provides a process called *Automatic File Build(AFB)*.

Multi-media adds an extra dimension to the normalization process. In addition to normalizing the textual input, the multi-media input also needs to be standardized. There are a lot of options to the standards being applied to the normalization. If the input is video the likely digital standards will be either MPEG-2, MPEG-1, AVI or Real Media. MPEG (Motion Picture Expert Group) standards are the most universal standards for higher quality video where Real Media is the most common standard for lower quality video being used on the Internet. Audio standards are typically WAV or Real Media (Real Audio). Images vary from JPEG to BMP.

The next process is to parse the item into logical sub-divisions that have meaning to the user. This process, called "Zoning," is visible to the user and used to increase the precision of a search and optimize the display. A typical item is sub-divided into zones, which may overlap and can be hierarchical, such as Title, Author, Abstract, Main Text, Conclusion, and References. The zoning information is passed to the processing token identification operation to store the information, allowing searches to be restricted to a specific zone. For example, if the user is interested in articles discussing "Einstein" then the search should not include the Bibliography, which could include references to articles written by "Einstein."

Systems determine words by dividing input symbols into 3 classes:

1) Valid word symbols
2) Inter-word symbols
3) Special processing symbols.

A word is defined as a contiguous set of word symbols bounded by inter-word symbols. In many systems inter-word symbols are non-searchable and should be carefully selected. Examples of word symbols are alphabetic characters and numbers. Examples of possible inter-word symbols are blanks, periods and semicolons. The exact definition of an inter-word symbol is dependent upon the aspects of the language domain of the items to be processed by the system. For example, an apostrophe may be of little importance if only used for the possessive case in English,

but might be critical to represent foreign names in the database.

Next, a *Stop List/Algorithm* is applied to the list of potential processing tokens. The objective of the Stop function is to save system resources by eliminating from the set of searchable processing tokens those that have little value to the system. Given the significant increase in available cheap memory, storage and processing power, the need to apply the Stop function to processing tokens is decreasing.

Examples of Stop algorithms are: Stop all numbers greater than "999999" (this was selected to allow dates to be searchable) Stop any processing token that has numbers and characters intermixed

2) Selective Dissemination (Distribution, Spreading) of Information

The Selective Dissemination of Information (Mail) Process provides the capability to dynamically compare newly received items in the information system against standing statements of interest of users and deliver the item to those users whose statement of interest matches the contents of the item. The Mail process is composed of the search process, user statements of interest (Profiles) and user mail files. As each item is received, it is processed against every user's profile. A profile contains a typically broad search statement along with a list of user mail files that will receive the document if the search statement in the profile is satisfied. Selective Dissemination of Information has not yet been applied to multimedia sources.

3) Document Database Search

The Document Database Search Process provides the capability for a query to search against all items received by the system. The Document Database Search process is composed of the search process, user entered queries (typically ad hoc queries) and the document database which contains all items that have been received, processed and stored by the system. Typically items in the Document Database do not change (i.e., are not edited) once received.

Index Database Search

When an item is determined to be of interest, a user may want to save it for future reference. This is in effect filing it. In an information system this is accomplished via

the index process. In this process the user can logically store an item in a file along with additional index terms and descriptive text the user wants to associate with the item. The Index Database Search Process (see Figure 1.4) provides the capability to create indexes and search them.

There are 2 classes of index files:

1) Public Index files
2) Private Index files

Every user can have one or more Private Index files leading to a very large number of files. Each Private Index file references only a small subset of the total number of items in the Document Database. Public Index files are maintained by professional library services personnel and typically index every item in the Document Database. There is a small number of Public Index files. These files have access lists (i.e., lists of users and their privileges) that allow anyone to search or retrieve data. Private Index files typically have very limited access lists. To assist the users in generating indexes, especially the professional indexers, the system provides a process *called Automatic File Build* shown in Figure 1.4 (also called Information Extraction).

Multimedia Database Search

From a system perspective, the multi-media data is not logically its own data structure, but an augmentation to the existing structures in the Information Retrieval System.

Relationship to Database Management Systems

From a practical standpoint, the integration of DBMS's and Information Retrieval Systems is very important. Commercial database companies have already integrated the two types of systems. One of the first commercial databases to integrate the two systems into a single view is the INQUIRE DBMS. This has been available for over fifteen years. A more current example is the ORACLE DBMS that now offers an imbedded capability called CONVECTIS, which is an informational retrieval system that uses a comprehensive thesaurus which provides the basis to generate "themes" for a particular item. The INFORMIX DBMS has the ability to link to RetrievalWare to provide integration of structured data and information along with functions associated with Information Retrieval Systems.

Digital Libraries and Data Warehouses (DataMarts)

As the Internet continued its exponential growth and project funding became available, the topic of Digital Libraries has grown. By 1995 enough research and pilot efforts had started to support the 1ST ACM International Conference on Digital Libraries (Fox-96). Indexing is one of the critical disciplines in library science and significant effort has gone into the establishment of indexing and cataloging standards. Migration of many of the library products to a digital format introduces both opportunities and challenges. Information Storage and Retrieval technology has addressed a small subset of the issues associated with Digital Libraries.

Data warehouses are similar to information storage and retrieval systems in that they both have a need for search and retrieval of information. But a data warehouse is more focused on structured data and decision support technologies. In addition to the normal search process, a complete system provides a flexible set of analytical tools to "mine" the data. Data mining (originally called Knowledge Discovery in Databases - KDD) is a search process that automatically analyzes data and extract relationships and dependencies that were not part of the database design.

Information Retrieval System Capabilities

2.1 Search Capabilities

2.2 Browse Capabilities

2.3 Miscellaneous Capabilities

2.4 Standards

The search capabilities address both Boolean and Natural Language queries. The algorithms used for searching are called Boolean, natural language processing and probabilistic. Probabilistic algorithms use frequency of occurrence of processing tokens (words) in determining similarities between queries and items and also in predictors on the potential relevance of the found item to the searcher.

The newer systems such as TOPIC, RetrievalWare, and INQUERY all allow for natural language queries.

Browse functions to assist the user in filtering the search results to find relevant information are very important.

2.1 Search Capabilities

The objective of the search capability is to allow for a mapping between a user's specified need and the items in the information database that will answer that need. It can consist of natural language text in composition style and/or query terms (referred to as terms in this book) with Boolean logic indicators between them. One concept that has occasionally been implemented in commercial systems (e.g., RetrievalWare), and holds significant potential for assisting in the location and ranking of relevant items, is the "weighting" of search terms. This would allow a user to indicate the importance of search terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular search term is indicated by a value in parenthesis between 0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional parameters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

2.1.1 Boolean Logic

Boolean logic allows a user to logically relate multiple concepts together to define what information is needed. Typically the Boolean functions apply to processing tokens identified anywhere within an item. The typical Boolean operators are **AND, OR,** and **NOT**. These operations are implemented using set intersection, set union and set difference procedures. Asearch terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular search term is indicated by a value in parenthesis between 0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional paramesearch terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular

search term is indicated by a value in parenthesis between 0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional parameters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

ters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

few systems introduced the concept of "exclusive or" but it is equivalent to a slightly more complex query using the other operators and is not generally useful to users since most users do not understand it.

A special type of Boolean search is called "M of N" logic. The user lists a set of possible search terms and identifies, as acceptable, any item that contains a subset of the terms. For example, "Find any item containing any two of the following terms: "AA," "BB," "CC." This can be expanded into a Boolean search that performs an AND between all combinations of two terms and "OR"s the results together ((AA AND BB) or (AA AND CC) or (BB AND CC)).

2.1.2 Proximity

Proximity is used to restrict the distance allowed within an item between two search terms. The semantic concept is that the clossearch terms in either a Boolean or natural language interface. Given the following natural language query statement where the importance of a particular search term is indicated by a value in parenthesis between

0.0 and 1.0 with 1.0 being the most important.

The search statement may apply to the complete item or contain additional parameters limiting it to a logical division of the item (i.e., to a zone). Based upon the algorithms used in a system many different functions are associated with the system's understanding the search statement. The functions define the relationships between the terms in the search statement (e.g., Boolean, Natural Language, Proximity, Contiguous Word Phrases, and Fuzzy Searches) and the interpretation of a particular word (e.g., Term Masking, Numeric and Date Range, Contiguous Word Phrases, and Concept/Thesaurus expansion).

two terms are found in a text the more likely they are related in the description of a particular concept. Proximity is used to increase the precision of a search. If the terms COMPUTER and DESIGN are found within a few words of each other then the item is more likely to be discussing the design of computers than if the words are paragraphs apart. The typical format for proximity is:

TERM1 within "m" "units" of TERM2

The distance operator "m" is an integer number and units are in Characters, Words, Sentences, or Paragraphs.

SEARCH STATEMENT	SYSTEM OPERATION
COMPUTER OR PROCESSOR NOT MAINFRAME	Select all items discussing Computers and/or Processors that do not discuss Mainframes
COMPUTER OR (PROCESSOR NOT MAINFRAME)	Select all items discussing Computers and/or items that discuss Processors and do not discuss Mainframes
COMPUTER AND NOT PROCESSOR OR MAINFRAME	Select all items that discuss computers and not processors or mainframes in the item

Figure 2.1 Use of Boolean Operators

A special case of the Proximity operator is the Adjacent (ADJ) operator that normally has a distance operator of one and a forward only direction (i.e., in WAIS). Another special case is where the distance is set to zero meaning within the same semantic unit.

2.1.3 Contiguous Word Phrases

A Contiguous Word Phrase (CWP) is both a way of specifying a query term and a special search operator. A Contiguous Word Phrase is two or more words that are treated as a single semantic unit. An example of a CWP is "United States of America." It is four words that specify a search term representing a single specific semantic concept (a country) that can be used with any of the operators discussed above. Thus a query could specify "manufacturing" AND "United States of America" which returns any item that contains the word "manufacturing" and the contiguous words "United States of America."

A contiguous word phrase also acts like a special search operator that is similar to the proximity (Adjacency) operator but allows for additional specificity. If two terms are specified, the contiguous word phrase and the proximity operator using directional one word parameters or the Adjacent operator are identical. For contiguous word phrases of more than two terms the only way of creating an equivalent search statement using proximity and Boolean operators is via nested Adjacencies which are not found in most commercial systems. This is because Proximity and Boolean operators are binary operators but contiguous word phrases are an "N"ary operator where "N" is the number of words in the CWP.

Contiguous Word Phrases are called Literal Strings in WAIS and Exact Phrases in RetrievalWare. In WAIS multiple Adjacency (ADJ) operators are used to define a Literal String (e.g., "United" ADJ "States" ADJ "of" ADJ "America").

SEARCH STATEMENT	SYSTEM OPERATION
"Venetian" ADJ "Blind"	would find items that mention a Venetian Blind on a window but not items discussing a Blind Venetian
"United" within five words of "American"	would hit on "United States and American interests," "United Airlines and American Airlines" not on "United States of America and the American dream"
"Nuclear" within zero paragraphs of "clean-up"	would find items that have "nuclear" and "clean-up" in the same paragraph.

Figure 2.2 Use of Proximity

2.1.4 Fuzzy Searches

Fuzzy Searches provide the capability to locate spellings of words that are similar to the entered search term. This function is primarily used to compensate for errors in spelling of words. Fuzzy searching increases recall at the expense of decreasing precision (i.e., it can erroneously identify terms as the search term). In the process of expanding a query term fuzzy searching includes other terms that have similar spellings, giving more weight (in systems that rank output) to words in the database that have similar word lengths and position of the characters as the entered term. A Fuzzy Search on the term "computer" would automatically include the following

words from the information database: "computer," "compiter," "conputer," "computter," "compute."

2.1.5 Term Masking

Term masking is the ability to expand a query term by masking a portion of the term and accepting as valid any processing token that maps to the unmasked portion of the term. The value of term masking is much higher in systems that do not perform stemming or only provide a very simple stemming algorithm. There are two types of search term masking: fixed length and variable length. Sometimes they are called fixed and variable length "don't care" functions.

Fixed length masking is a single position mask. It masks out any symbol in a particular position or the lack of that position in a word. Variable length "don't cares" allows masking of any number of characters within a processing token. The masking may be in the front, at the end, at both front and end, or imbedded. The first three of these cases are called suffix search, prefix search and imbedded character string search, respectively. The use of an imbedded variable length don't care is seldom used. Figure 2.3 provides examples of the use of variable length term masking. If "*" represents a variable length don't care then the following are examples of its use:

"*COMPUTER" Suffix Search

"COMPUTER*" Prefix Search

"*COMPUTER*" Imbedded String Search

SEARCH STATEMENT	SYSTEM OPERATION
multi$national	Matches"multi-national," "multinational," "multinational" but does not match "multi national" since it is two processing tokens.
computer	Matches,"minicomputer" "microcomputer" or "computer"
comput*	Matches "computers," "computing," "computes"
comput	Matches "microcomputers" , "minicomputing," "compute"

Figure 2.3 Term Masking

2.1.6 Numeric and Date Ranges

Term masking is useful when applied to words, but does not work for finding ranges of numbers or numeric dates. To find numbers larger than "125," using a term "125*" will not find any number except those that begin with the digits "125."

2.1.7 Concept/Thesaurus Expansion

Associated with both Boolean and Natural Language Queries is the ability to expand the search terms via Thesaurus or Concept Class database reference tool. A Thesaurus is typically a one-level or two-level expansion of a term to other terms that are similar in meaning. A Concept Class is a tree structure that expands each meaning of a word into potential concepts that are related to the initial term (e.g., in the TOPIC system). Concept classes are sometimes implemented as a network structure that links word stems (e.g., in the RetrievalWare system). An example of Thesaurus and Concept Class structures are shown in Figure 2.4 (Thesaurus-93) and Figure 2.5.

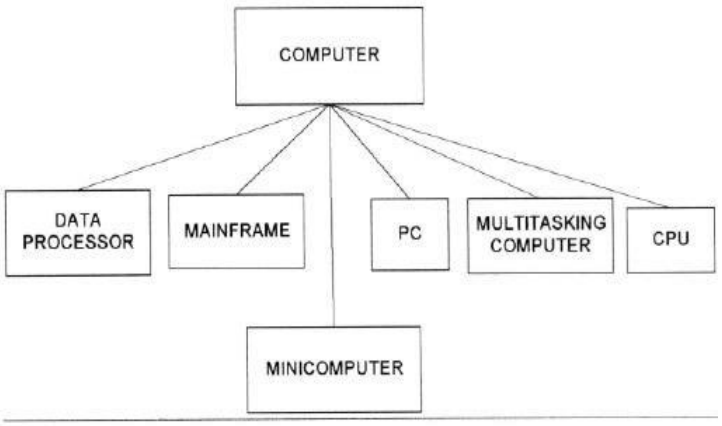


Figure 2.4 Thesaurus for term "computer"

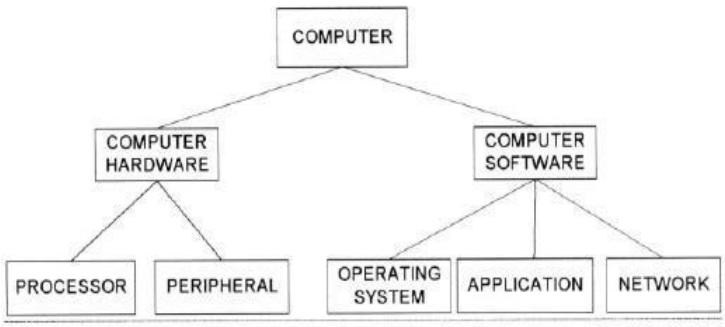


Figure 2.5 Hierarchical Concept Class Structure for "Computer"

Thesauri are either semantic or based upon statistics. A semantic thesaurus is a listing of words and then other words that are semantically similar.

The problem with thesauri is that they are generic to a language and can introduce many search terms that are not found in the document database. An alternative uses the database or a representative sample of it to create statistically related terms. It is conceptually a thesaurus in that words that are statistically related to other words by their frequently occurring together in the same items. This type of thesaurus is very dependent upon the database being searched and may not be portable to other databases.

2.1.8 Natural Language Queries

Natural language interfaces improve the recall of systems with a decrease in precision when negation is required.

2.2 Browse Capabilities

Once the search is complete, Browse capabilities provide the user with the capability to determine which items are of interest and select those to be displayed. There are two ways of displaying a summary of the items that are associated with a query: line item status and data visualization. From these summary displays, the user can select the specific items and zones within the items for display.

2.2.1 Ranking

Typically relevance scores are normalized to a value between 0.0 and 1.0. The highest value of 1.0 is interpreted that the system is sure that the item is relevant to the search statement. In addition to ranking based upon the characteristics of the item and the database, in many circumstances collaborative filtering is providing an option for selecting and ordering output.

Collaborative filtering has been very successful in sites such as AMAZON.COM MovieFinder.com, and CDNow.com in deciding what products to display to users based upon their queries.

Rather than limiting the number of items that can be assessed by the number of lines on a screen, other graphical visualization techniques showing the relevance

relationships of the hit items can be used. For example, a two or three dimensional graph can be displayed where points on the graph represent items and the location of the points represent their relative relationship between each other and the user's query. In some cases color is also used in this representation. This technique allows a user to see the clustering of items by topics and browse through a cluster or move to another topical cluster.

2.2.2 Zoning

Related to zoning for use in minimizing what an end user needs to review from a hit item is the idea of locality and passage based search and retrieval.

2.2.3 Highlighting

Most systems allow the display of an item to begin with the first highlight within the item and allow subsequent jumping to the next highlight. The DCARS system that acts as a user frontend to the Retrieval Ware search system allows the user to browse an item in the order of the paragraphs or individual words that contributed most to the rank value associated with the item. The highlighting may vary by introducing colors and intensities to indicate the relative importance of a particular word in the item in the decision to retrieve the item.

2.3 Miscellaneous Capabilities

2.3.1Vocabulary Browse

Vocabulary Browse provides the capability to display in alphabetical sorted order words from the document database. Logically, all unique words (processing tokens) in the database are kept in sorted order along with a count of the number of unique items in which the word is found. The user can enter a word or word fragment and the system will begin to display the dictionary around the entered text.

It helps the user determine the impact of using a fixed or variable length mask on a search term and potential mis-spellings. The user can determine that entering the search term "compul*" in effect is searching for "compulsion" or "compulsive" or "compulsory." It also shows that someone probably entered the word "computen" when they really meant "computer."

TERM	OCCURRENCES
compromise	53
comptroller	18
compulsion	5
compulsive	22
compulsory	4

2.3.2 Iterative Search and Search History Log

Frequently a search returns a Hit file containing many more items than the user wants to review. Rather than typing in a complete new query, the results of the previous search can be used as a constraining list to create a new query that is applied against it. This has the same effect as taking the original query and adding additional search statement against it in an AND condition. This process of refining the results of a previous search to focus on relevant items is called iterative search. This also applies when a user uses relevance feedback to enhance a previous search. The search history log is the capability to display all the previous searches that were executed during the current session.

2.3.3 Canned Query

The capability to name a query and store it to be retrieved and executed during a later user session is called canned or stored queries. A canned query allows a user to create and refine a search that focuses on the user's general area of interest one time and then retrieve it to add additional search criteria to retrieve data that is currently needed. Canned query features also allow for variables to be inserted into the query and bound to specific values at execution time.

2.4 Z39.50 and WAIS Standards

The Z39.50 standard does not specify an implementation, but the capabilities within an application (Application Service) and the protocol used to communicate between applications (Information Retrieval Application Protocol). It is a computer to computer communications standard for database searching and record retrieval. Its objective is to overcome different system incompatibilities associated with multiple database searching.

The first version of Z39.50 was approved in 1992. An international version of Z39.50, called the Search and Retrieve Standard (SR), was approved by the International Organization for Standardization (ISO) in 1991. Z39.50-1995, the latest version of Z39.50, replaces SR as the international information retrieval standard. The standard describes eight operation types: Init (initialization), Search, Present, Delete, Scan, Sort, Resource-report, and Extended Services. There are five types of queries (Types 0, 1, 2, 100, 101, and 102).

The client is identified as the "Origin" and performs the communications functions relating to initiating a search, translation of the query into a standardized format, sending a query, and requesting return records. The server is identified as the "Target" and interfaces to the database at the remote responding to requests from the Origin (e.g., pass query to database, return records in a standardized format and status). The end user does not have to be aware of the details of the standard since the Origin function performs the mapping from the user's query interface into Z39.50 format.

This makes the dissimilarities of different database systems transparent to the user and facilitates issuing one query against multiple databases at different sites returning to the user a single integrated Hit file. Wide Area Information Service (WAIS) is the de facto standard for many search environments on the INTERNET. WAIS was developed by a project started in 1989 by three commercial companies (Apple, Thinking Machines, and Dow Jones). The original idea was to create a program that would act as a personal librarian.

A free version of WAIS is still available via the Clearinghouse for Networked Information Discovery and Retrieval (CINDIR) called "FreeWAIS." The original development of WAIS started with the 1988 Z39.50 protocol as a base following the client/server architecture concept. The developers incorporated the information retrieval concepts that allow for ranking, relevance feedback and natural language processing functions that apply to full text searchable databases. Center for National Research Initiatives (CNRI) that is working with the Department of Defense and also the American Association of Publishers (AAP), focusing on an Internet implementation that allows for control of electronic published and copyright material. In addition to the Handle Server architecture, CNRI is also advocating a communications protocol to retrieve items from existing systems.This protocol call Repository Archive Protocol (RAP) defines the mechanisms for clients to use the handles to retrieve items. It also includes other administrative functions such as privilege validation. The Handle system is designed to meet the Internet Engineering Task Force (IETF) requirements for naming Internet objects via Uniform Resource Names to replace URLs as defined in the Internet's RFC-1737 (IETF- 96).

WAIS (Wide Area Information Servers)

WAIS (Wide Area Information Servers) is an Internet system in which specialized subject databases are created at multiple server locations, kept track of by a *directory of servers* at one location, and made accessible for searching by users with WAIS client programs. The user of WAIS is provided with or obtains a list of distributed database s. The user enters a search argument for a selected database and the client then accesses all the servers on which the database is distributed. The results provide a description of each text that meets the search requirements. The user can then retrieve the full text. **RetrievalWare** is an enterprise search engine emphasizing natural language processing and semantic networks.

UNIT-II

Cataloging and Indexing: Objectives, Indexing Process, Automatic Indexing, formation Extraction. **Data Structures:** Introduction, Stemming Algorithms, Inverted file structures, N-gram data structure, PAT data structure, Signature file structure, Hypertext data structure.

CATALOGING AND INDEXING

INDEXING:

The transformation from received item to searchable data structure is called indexing.

• Process can be manual or automatic.

• Creating a direct search in document data base or indirect search through index files.

• Concept based representation: instead of transforming the input into a searchable format some systems transform the input into different representation that is concept based .Search ? Search and return item as per the incoming items.

• History of indexing: shows the dependency of information processing capabilities on manual and then automatic processing systems .

• Indexing originally called cataloguing : oldest technique to identity the contents of items to assist in retrieval.

• Items overlap between full item indexing , public and private indexing of files

Objectives :

The public file indexer needs to consider the information needs of all users of library system . Items overlap between full item indexing , public and private indexing of files.

•Users may use public index files as part of search criteria to increase recall.

•They can constrain there search by private index files

•The primary objective of representing the concepts within an item to facilitate users finding relevant information .

•Users may use public index files as part of search criteria to increase recall.

•They can constrain there search by private index files

•The primary objective of representing the concepts within an item to facilitate users finding relevant information

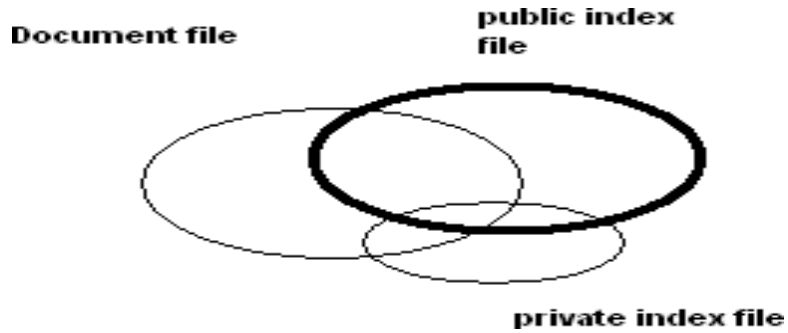


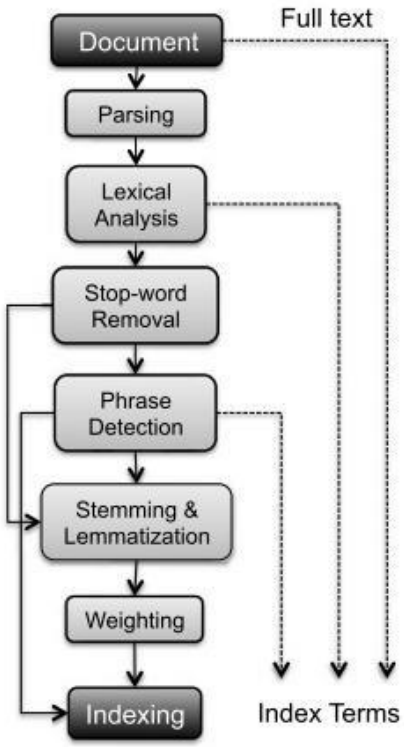
Fig : Indexing process

1. Decide the scope of indexing and the level of detail to be provided. Based on usage scenario of users.

2. Second decision is to link index terms together in a single index for a particular concept.

TEXT PROCESSING

Fig. 2.3 Text processing phases in an IR system



1. Document Parsing. Documents come in all sorts of languages, character sets, and formats; often, the same document may contain multiple languages or formats, e.g., a French email with Portuguese PDF attachments. Document parsing deals with the recognition and "breaking down" of the document structure into individual components. In this pre processing phase, unit documents are created; e.g., emails with attachments are split into one document representing the email and as many documents as there are attachments.

2. Lexical Analysis. After parsing, lexical analysis tokenizes a document, seen as an input stream, into words. Issues related to lexical analysis include the correct identification of accents, abbreviations, dates, and cases. The difficulty of this operation depends much on the language at hand: for example, the English language has neither diacritics nor cases, French has diacritics but no cases, German has both diacritics and cases. The recognition of abbreviations and, in particular, of time expressions would deserve a separate chapter due to its complexity and the extensive literature in the field For current approaches

3. Stop-Word Removal. A subsequent step optionally applied to the results of lexical analysis is stop-word removal, i.e., the removal of high-frequency words. For example, given the sentence "search engines are the most visible information retrieval applications" and a classic stop words set such as the one adopted by the Snowball stemmer,1 the effect of stop-word removal would be: "search engine most visible information retrieval applications".

4. Phrase Detection. This step captures text meaning beyond what is possible with pure bag- of-word approaches, thanks to the identification of noun groups and other phrases. Phrase detection may be approached in several ways, including rules (e.g., retaining terms that are not separated by punctuation marks), morphological analysis , syntactic analysis, and combinations thereof. For example, scanning our example sentence "search engines are the most visible information retrieval applications" for noun phrases would probably result in identifying "search engines" and "information retrieval".

5. Stemming and Lemmatization. Following phrase extraction, stemming and lemmatization aim at stripping down word suffixes in order to normalize the word. In particular, stemming is a heuristic process that "chops off" the ends of words in the hope of achieving the goal correctly most of the time; a classic rule based algorithm for this was devised by Porter [280]. According to the Porter stemmer, our example sentence "Search engines are the most visible information

retrieval applications" would result in: "Search engin are the most visibl inform retriev applic".

1. Lemmatization is a process that typically uses dictionaries and morphological analysis of words in order to return the base or dictionary form of a word, thereby collapsing its inflectional forms (see, e.g., [278]). For example, our sentence would result in "Search engine are the most visible information retrieval application" when lemmatized according to a WordNet-based lemmatizer
2. Weighting. The final phase of text pre processing deals with term weighting. As previously mentioned, words in a text have different descriptive power; hence, index terms can be weighted differently to account for their significance within a document and/or a document collection. Such a weighting can be binary, e.g., assigning 0 for term absence and 1 for presence.

SCOPE OF NDEXING

- When perform the indexing manually, problems arise from two sources the author and the indexer the author and the indexer .
- Vocabulary domain may be different the author and the indexer.
- This results in different quality levels of indexing.
- The indexer must determine when to stop the indexing process.
- Two factors to decide on level to index the concept in a item.
- The exhaustively and how specific indexing is desired.
- Exhaustively of index is the extent to which the different concepts in the item are indexed.
- For example, if two sentences of a 10-page item on microprocessors discusses on-board caches, should this concept be indexed
- Specific relates to preciseness of index terms used in indexing.
- For example, whether the term "processor" or the term "microcomputer" or the term "Pentium" should be used in the index of an item is based upon the specificity decision.

- Indexing an item only on the most important concept in it and using general index terms yields low exhaustively and specificity.
- Another decision on indexing is what portion of an item to be indexed Simplest case is to limit the indexing to title and

  abstract(conceptual) zone .
- General indexing leads to loss of precision and recall.

PREORDINATION AND LINKAGES

- Another decision on linkages process whether linkages are available between index terms for an item .
- Used to correlate attributes associated with concepts discussed in an item .'this process is called preordination .
- When index terms are not coordinated at index time the coordination occurs at search time. This is called post coordination , implementing by "AND" ing index terms .
- Factors that must be determined in linkage process are the number of terms that can be related.
- Ex., an item discusses 'the drilling of oil wells in Mexico by CITGO and the introduction of oil refineries in Peru by the U.S.'

DATA STRUCTURES

- Introduction to Data Structures
- Stemming Algorithms
- Inverted File Structure
- N-Gram Data Structure
- PAT Data Structure
- Signature File Structure
- Hypertext and XML Data Structures

Data structure :

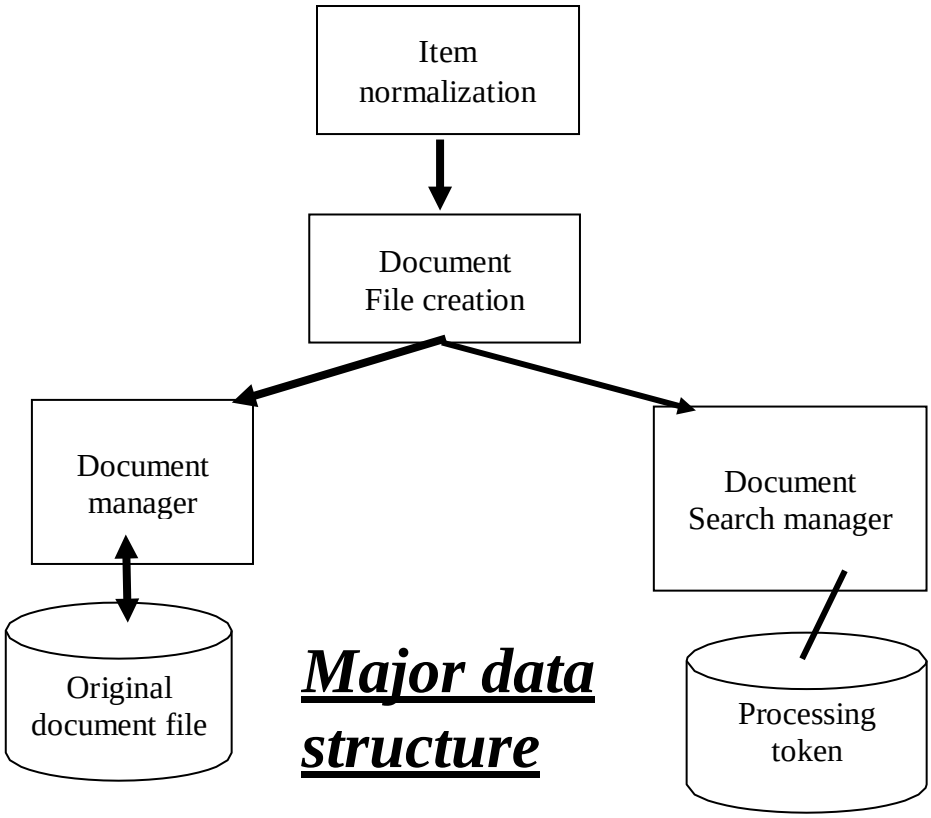
The knowledge of data structure gives an insight into the capabilities available to the system .

- Each data structure has a set of associated capabilities .
- Ability to represent the concept and their r/s.
- Supports location of those concepts Introduction

Two major data structures in any IRS:

1. One structure stores and manages received items in their normalized form is called document manger

2. The other data structure contains processing tokens and associated data to support search.



Result of a search are references to the items that satisfy the search statement which are passed to the document manager for retrieval.

Focus : on data structure that support search function

Stemming : is the transformation often applied to data before placing it in the searchable data structure

Stemming represents concept(word) to a canonical (authorized; recognized; accepted)morphological (the patterns of word formation in a particular

language) representation .Risk with stemming : concept discrimination information may be lost in the

process. Causing decrease in performance.

Advantage : has a potential to increase recall.

STEMMING ALGORITHMS

- Stemming algorithm is used to improve the efficiency of IRS and improve recall.
- Conflation(the process or result of fusing items into one entity; fusion; amalgamation)is a term that is used to refer mapping multiple morphological variants to single representation(stem).
- Stem carries the meaning of the concept associated with the word and the affixes(ending) introduce subtle(slight) modification of the concept.
- Terms with a common stem will usually have similar meanings,

  for example:
- Ex : Terms with a common stem will usually have similar meanings, for example:
- CONNECT
- CONNECTED
- CONNECTING
- CONNECTION
- CONNECTIONS
- Frequently, the performance of an IR system will be improved if term groups such as this are conflated into a single term. This may be done by removal of the various suffixes -ED, -ING, -ION, IONS to leave the single term CONNECT
- In addition, the suffix stripping process will reduce the total number of terms in the IR system, and hence reduce the size and complexity of the data in the system, which is always advantageous.

❖ Major usage of stemming is to improve recall.

➢ Important for a system to categories a word prior to making the decision to stem.

➢ Proper names and acronyms (A word formed from the initial letters of a name say IARE …) should not have stemming applied.

➢ Stemming can also cause problems for natural language processing NPL systems by causing loss of information .

PORTER STEMMING ALGORITHM

- Based on a set condition of the stem
- A consonant in a word is a letter other than A, E, I, O or U, some important stem conditions are

1. The measure m of a stem is a function of sequence of vowels (V) followed by a sequence of consonant (C) .
2. C (VC)mV. m is number VC repeats The case m = 0 covers the null word.
3. *<X> - stem ends with a letter X 3.*v* - stem contains a vowel
4. *d- stem ends in double consonant (e.g. -TT, -SS).
5. *o- stem ends in consonant vowel sequence where the final consonant is not w,x,y(e.g. -WIL, -HOP).

Suffix cond.s takes the form current _suffix = = pattern Actions are in the form old_suffix ->. New_suffix

Rules are divided into steps to define the order for applying the rule.

Examples of the rules

Step	Condition	Suffix	Replacement	Example
1a	Null	Sses	Ss	Stresses -> stress
1b	*v*	Ing	Null	Making -> mak
1b1	Null	At	Ate	Inflated-> inflate
1c	*v*	Y	I	Happy->happi
2	m>0	aliti	al	Formaliti-> formal
3	m>0	Icate	Ic	Duplicate->duplie
4	m>1	Able	Null	Adjustable -> adjust
5a	m>1	e	Null	Inflate-> inflat
5b	m>1 and *d	Null	Single letter	Control -> control

2. Dictionary look up stemmers

- Use of dictionary look up.
- The original term or stemmed version of the term is looked up in a dictionary and replaced by the stem that best represents it.
- This technique has been implemented in INQUERY and Retrieval ware systems-

INQUERY system uses the technique called Kstem.

- Kstem is a morphological analyzer that conflates words variants to a root form.
- It requires a word to be in the dictionary
- Kstem uses 6 major data files to control and limit the stemming process.

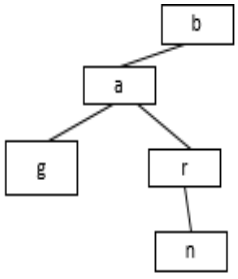
1. Dictionary of words (lexicon)
2. Supplemental list of words for dictionary
3. Exceptional list of words that should retain a 'e' at the end (e.g., "suites" to "suite" but "suited" to "suit").
4. Direct _conflation - word pairs that override stemming algorithm.
5. County_nationality _conflation (British maps to Britain)
6. Proper nouns -- that should not be stemmed

- New words that are not special forms (e.g., dates, phone numbers) are located in the dictionary to determine simpler forms by stripping off suffixes and respelling plurals as defined in the dictionary.

3. Successor stemmers:

- Based on length of prefixes .
- The smallest unit of speech that distinguishes on word from another
- The process uses successor varieties for a word .

Uses information to divide a word into segments and selects on of the segments to stem.



Symbol tree for term bag , barn

Successor variety of words are used to segment a word by applying one of the following four methods.

1. Cutoff method : a cut of value is selected to define the stem length.
2. Peak and plateau: a segment break is made after a character whose successor variety exceeds that of the character.
3. Complete word method: break on boundaries of complete words.
4. Entropy method:uses the distribution method of successor variety letters.

1. Let |Dak| be the number of words beginning with k length sequence of letters a.
2. Let |Dakj| be the number of words in Dak with successor j.
3. The probability that a member of Dak has the successor j

   is given as |Dakj| / |Dak| The entropy of |Dak| is

   26

   Hak = □ -(|Dakj| / |Dak|

)(log(|Dakj| /

   |Dak|)) p=1

After a word has been segmented the segment to be used as stem

must be selected. Hafer and Weiss selected the following rule

If (first segment occurs in <=12 words in database) First segment

is stem

Else (second segment is stem)

INVERTED FILE STRUCTURE

Inverted file structure

Most common data structure

Inverted file structures are composed of three files

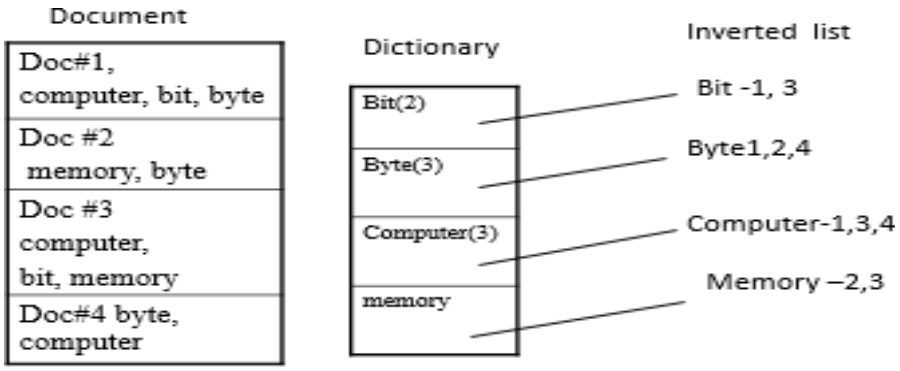
The document file

1. The inversion list (Posting List)
2. Dictionary
3. The inverted file : based on the methodology of storing an inversion of documents.

4. For each word a listof documents in which the word is found is stored(inversion of document
5. Each document is given a unique the numerical identifier that is stored in inversion list . Dictionary is used to located the inversion list for a particular word.

Which is a sorted list(processing tokens) in the system and a pointer to the location of its inversion list.

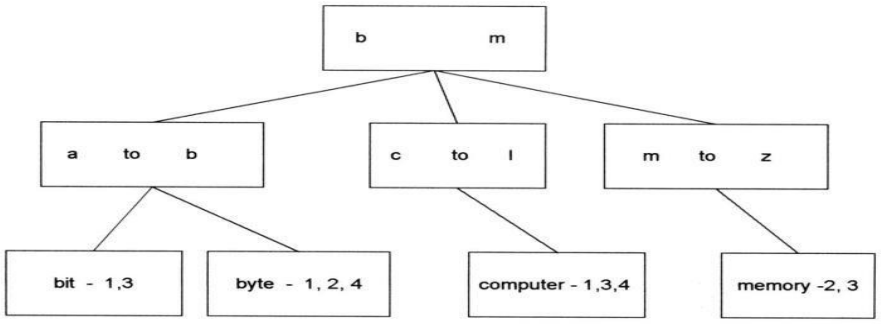
Dictionary can also store other information used in query optimization such as length of inversion lists to increase the precision.



- Use zoning to improve
- precision and Restrict entries.
- Inversion list consists of document identifier for each document in which the word is found.

Ex: bit 1(10),1(12) 1(18) is in 10,12, 18 position of the word bit in the document #1.

- When a search is performed, the inversion lists for the terms in the query are locate and appropriate logic is applied between inversion lists.
- Weights can also be stored in the inversion list.
- Inversion list are used to store concept and their relationship.
- Words with special characteristics can be stored in their own dictionary. Ex: Date… which require date ranging and numbers.
- Systems that support ranking are re-organized in ranked order.
- B trees can also be used for inversion instead of dictionary.
- The inversion lists may be at the leaf level or referenced in higher level pointers.
- A B-tree of order m is defined as:
- A root node with between 2 and 2m keys
- All other internal nodes have between m and 2m keys
- All keys are kept in order from smaller to larger.
- All leaves are at the same level or differ by at most one level.



N-GRAM DATA STRUCTURE

- N-Grams can be viewed as a special technique for conflation (stemming) and as a unique data structure in information systems.
- N-Grams are a fixed length consecutive series of "n" characters.
- Unlike stemming that generally tries to determine the stem of a word that represents the semantic meaning of the word, n-grams do not care about semantics.
- The searchable data structure is transformed into overlapping n-grams, which are then used to create the searchable database.
- Examples of bigrams, trigrams and pentagrams for the word phrase "sea colony." se ea co ol lo on ny Bigrams (no interword symbols)

sea col olo lon onyTrigrams (no interword symbols) #se sea ea# #co col olo lon ony ny# Trigrams (with interword symbol #)

#sea# #colo colon olony lony#
Pentagrams (with interword
symbol #)

The symbol # is used to represent the interword symbol which is anyone of a set of symbols (e.g., blank, period, semicolon, colon, etc.).

- The symbol # is used to represent the interword symbol which is anyone of a set of symbols (e.g., blank, period, semicolon, colon, etc.).
- Each of the n-grams created becomes a separate processing tokens and are searchable.
- It is possible that the same n-gram can be created multiple times from a single word.

Uses :

- Widely used as cryptography in world war II Spelling errors detection and correction

- Use bigrams for conflating terms.
- N-grams as a potential erroneous words.
- Damerau specified 4 categories of errors:

Error Category	Example
single char insertion	compuuter
single char deletion	compter
single char substitution	compiter
Transposition of 2 adjacent chars	comp*tu*er

- Zamora showed trigram analysis provided a viable data structure for identifying misspellings and transposed characters.
- This impacts information systems as a possible basis for identifying potential input errors for correction as a procedure within the normalization process.
- Frequency of occurrence of n-gram patterns can also be used for identifying the language of an item.
- Trigrams have been used for text compression and to manipulate the length of index terms.
- To encode profiles for the Selective Dissemination of Information.
- To store the searchable document file for retrospective search databases.

Advantage:

They place a finite limit on the number of searchable token

MaxSeg n=(□)n maximum number of unique n grams that can be generated. " n" is the length of n-grams

□ number of process able symbols

Disadvantage: longer the n gram the size of inversion list increase. Performance has 85 % precision .

PAT data structure (practical algorithm to retrieve information coded in alphanumeric)

- PAT structure or PAT tree or PAT array : continuous text input

  data structures(string like N- Gram data structure).
- The input stream is transformed into a searchable data structure consisting of substrings, all substrings are unique.
- Each position in a input string is a anchor point for a sub string.
- In creation of PAT trees each position in the input string is the anchor point for a sub-string that starts at that point and includes all new text up to the end of the input.
- Binary tree, most common class for prefix search,But Pat trees are sorted logically which facilitate range search, and more accurate then inversion file .
- PAT trees provide alternate structure if supporting strings search.

Text Economics for Warsaw is complex.

---

sistring 1 Economics for Warsaw is

complex. sistring 2 conomics for

Warsaw is complex. sistring 5 omics for

Warsaw is complex. sistring 10 for

Warsaw is complex.

sistring 20 w is

complex. sistring 30

ex.

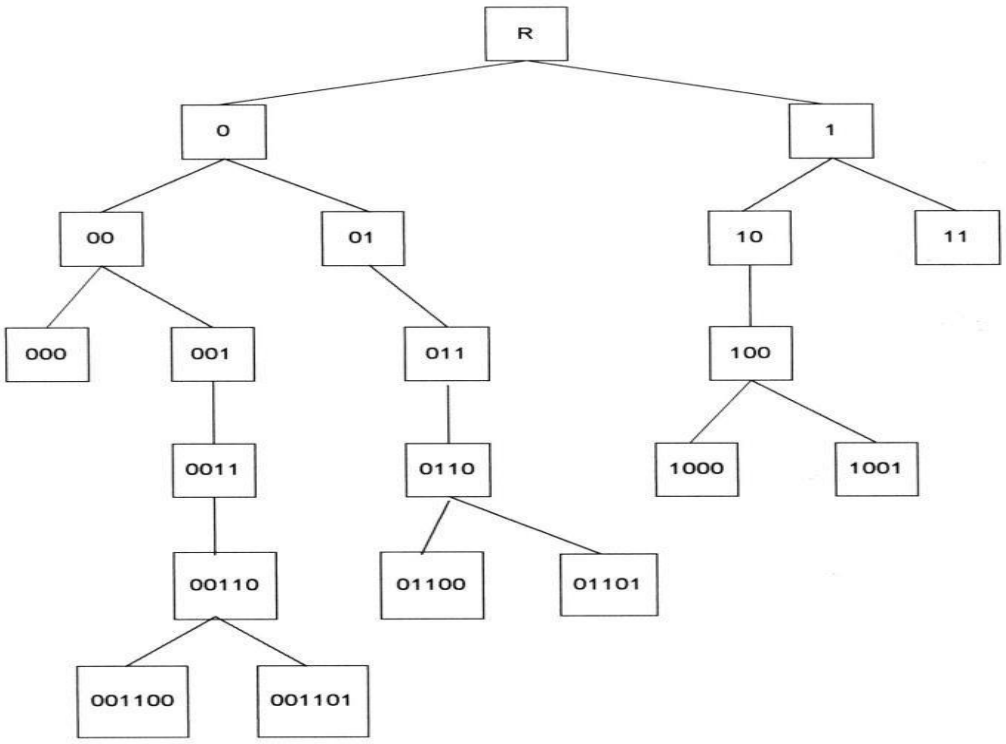
Examples of sistrings

- The key values are stored at the leaf nodes (bottom nodes) in the PAT Tree.
- For a text input of size "n" there are "n" leaf nodes and "n-1" at most higher level nodes.
- It is possible to place additional constraints on sistrings for the leaf nodes.
- If the binary representations of "h" is (100), "o" is (110), "m" is (001) and "e" is (101) then the word "home" produces the input

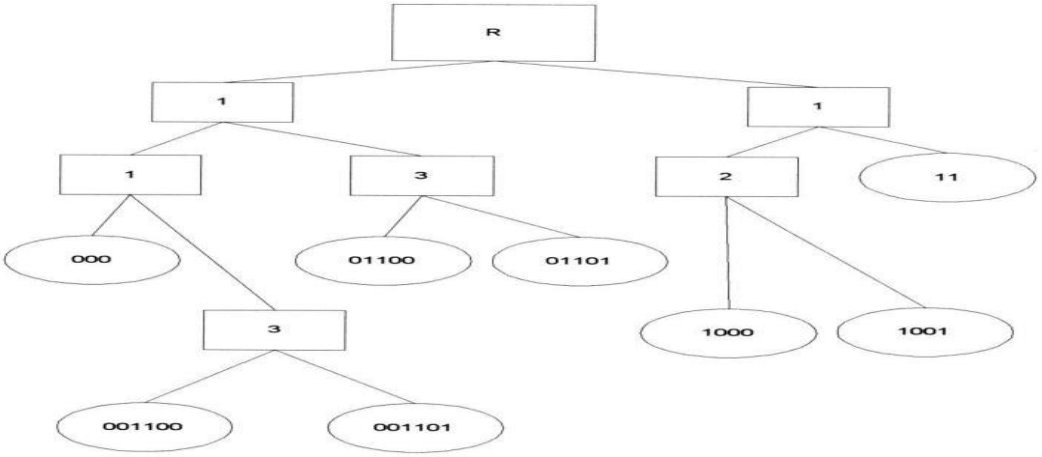
100110001101.....Using the sistrings.

INPUT	100110001101
sistring 1	1001....
sistring 2	001100...
sistring 3	01100....
sistring 4	11.....
sistring 5	1000...
sistring 6	000.....
sistring 7	001101
sistring 8	01101

The full PAT binary tree is



The value in the intermediate nodes (indicated by rectangles) is the number of bits to skip until the next bit to compare that causes differences between similar terms.



Skipped final version of PAT tree

Signature file structure

- The coding is based upon words in the code.
- The words are mapped into word signatures .
- A word signature is fixed length code with a fixed number of bits set to 1.
- The bit positions that are set to one are determined via a hash function of the word.
- The word signatures are Ored together to create signature of an item..
- Partitioning of words is done in block size ,Which is nothing but set of words, Code length is 16 bits .
- Search is accomplished by template matching on the bit position .
- provide a practical solution applied in parallel processing , distributed environment etc.

- To avoid signatures being too dense with "1"s, a maximum number of words is specified and an item is partitioned into blocks of that size.
- The block size is set at five words, the code length is 16 bits and the number of bits that are allowed to be "1" for each word is five.
- TEXT: Computer Science graduate students study (assume block size is five words)

WORD	Signature
computer	0001 0110 0000 0110
Science	1001 0000 1110 0000
graduate	1000 0101 0100 0010
students	0000 0111 1000 0100
study	0000 0110 0110 0100
Block Signature	1001 0111 1110 0110

Superimposed Coding

Application(s)/Advantage(s)

- Signature files provide a practical solution for storing and locating information in a number of different situations.
- Signature files have been applied as medium size databases, databases with low frequency of terms, WORM devices, parallel processing machines, and distributed environments

HYPERTEXT AND XML DATA STRUCTURES

- The advent of the Internet and its exponential growth and wide acceptance as a new global information network has introduced new mechanisms for representing information.
- This structure is called hypertext and differs from traditional information storage data structures in format and use.
- The hypertext is Hypertext is stored in HTML format and XML .
- Bot of these languages provide detailed descriptions for subsets of text similar to the zoning.
- Hypertext allows one item to reference another item via a embedded pointer .
- HTML defines internal structure for information exchange over WWW on the internet.
- XML: defined by DTD, DOM, XSL, etc.

Document and term clustering

Two types of clustering:

1) clustering index terms to create a statistical thesaurus and

2) clustering items to create document clusters. In the first case clustering is used to increase recall by expanding searches with related terms. In document clustering the search can retrieve items similar to an item of interest, even if the query would not have retrieved the item. The clustering process is not precise and care must be taken on use of clustering techniques to minimize the negative impact misuse can have.

UNIT-III

Automatic Indexing: Classes of automatic indexing, Statistical indexing, Natural language, Concept indexing, Hypertext linkages **Document and Term Clustering:** Introduction, Thesaurus generation, Item clustering, Hierarchy of clusters.

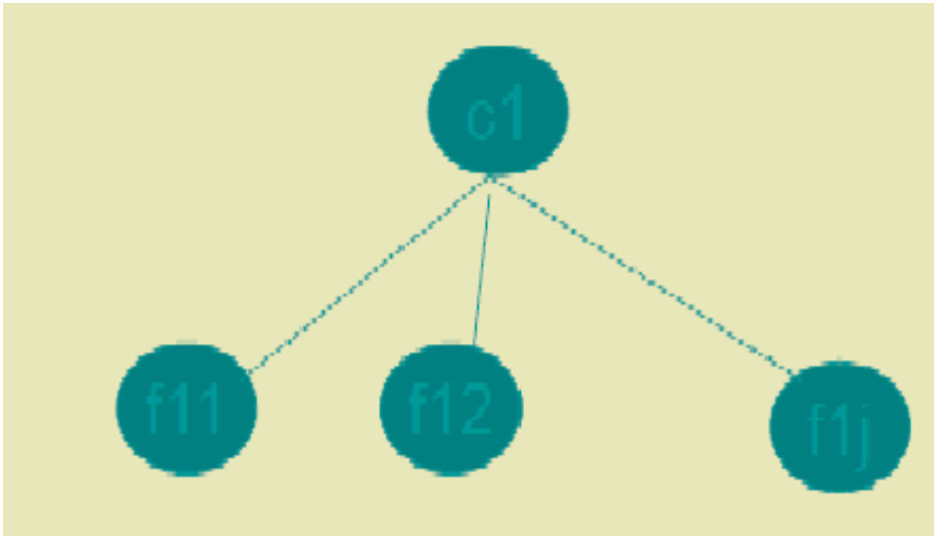
AUTOMATIC INDEXING

- Case: Total document indexing.
- Automatic indexing requires few seconds based on the processor and complexity of algorithms to generate indexes.'
- Adv. is consistency in index term selection process.
- Index resulting form automated indexing fall into two classes , weighted and un weighted .
- Un weighted indexing system : the existence of an index term in a document and some times its word location are kept as part of searchable data structure .
- Weighted indexing system: a attempt is made to place a value on the index term associated with concept in the document . Based on the frequency of occurrence of the term in the item .
- Values are normalized between 0 and 1.
- The results are presented to the user in order of rank value from highest number to lowest number .
- Indexing By term
- Terms (vocabulary) of the original item are used as basis of index process .
- There are two major techniques for creation of index statistical and natural language.
- Statistical can be based upon vector models and probabilistic models with a special case being Bayesian model(accounting for uncertainty inherent in the model selection process) .

- Called statistical because their calculation of weights use information such as frequency of occurrence of words .
- Natural language also use some statistical information , but perform more complex parsing to define the final set of index concept.
- Other weighted systems discussed as vectorised information system .
- The system emphasizes weights as a foundation for information detection and stores these weights in a vector form.
- Each vector represents a document. And each position in a vector represent a unique word(processing token) in a data base..
- The value assigned to each position is the weight of that term in the document.
- 0 indicates that the word was not in the document .
- Search is accomplished by calculating the distance between the query vector and document vector.
- Bayesian approach: based on evidence reasoning(drawing conclusion from evidence)
- Could be applied as part of index term weighing. But usually applied as part of retrieval process by calculating the relation ship

  between an item and specific query.
- Graphic representation each node represents a random variable arch between the nodes represent a probabilistic dependencies between the node and its parents .

  Two level Bayesian network
- " c"" represents concept in a query
- "f" representing concepts in an item



- Another approach is natural language processing.
- DR-LINK(document retrieval through linguistics knowledge)
- Indexing by concept
- Concept indexing determines a canonical set of concept based upon a test set of terms and uses them as base for indexing all items. Called latent semantics indexing .
- Ex: match plus system developed by HNC inc
- Uses neural NW strength of the system word relationship (synonyms) and uses the information in generating context vectors.
- Two neural networks are used one to generated stem context vectors and another one to perform query.
- Interpretation is same as the weights.
- Multimedia indexing:
- Indexing video or images can be accomplished at raw data level.

- Positional and temporal (time) search can be done.

INFORMATION EXTRACTION

There are two processes associated with information extraction:

1.determination of facts to go into structured fields in a database and

2. Extraction of text that can be used to summarize an item.

The process of extracting facts to go into indexes is called Automatic File Build.

In establishing metrics to compare information extraction, precision and recall are applied with slight modifications.

- Recall refers to how much information was extracted from an item versus how much should have been extracted from the item.
- It shows the amount of correct and relevant data extracted versus the correct and relevant data in the item.
- Precision refers to how much information was extracted accurately versus the total information extracted.
- Additional metrics used are over generation and fallout.
- Over generation measures the amount of irrelevant information that is extracted.
- This could be caused by templates filled on topics that are not intended to be extracted or slots that get filled with non-relevant data.
- Fallout measures how much a system assigns incorrect slot fillers as the number of
- These measures are applicable to both human and automated extraction processes.
- Another related information technology is document summarization.
- Rather than trying to determine specific facts, the goal of document summarization is to extract a summary of an item maintaining the most important ideas while significantly reducing the size.
- Examples of summaries that are often part of any item are titles, table of contents, and abstracts with the abstract being the closest.
- The abstract can be used to represent the item for search purposes or as a way for a user to determine the utility of an item without having to read the complete item.

6.1 Introduction to Clustering

The goal of the clustering was to assist in the location of information. Clustering of words originated with the generation of thesauri. Thesaurus, coming from the Latin word meaning "treasure," is similar to a dictionary in that it stores words. Instead of definitions, it provides the synonyms and antonyms for the words. Its primary purpose is to assist authors in selection of vocabulary. The goal of clustering is to provide a grouping of similar objects (e.g., terms or items) into a "class" under a more general title. Clustering also allows linkages between clusters to be specified. The term class is frequently used as a synonym for the term cluster.

The process of clustering follows the following steps:

- Define the domain for the clustering effort. Defining the domain for the clustering identifies those objects to be used in the clustering process. Ex: Medicine, Education, Finance etc.
- Once the domain is determined, determine the attributes of the objects to be clustered. (Ex: Title, Place, job etc zones)
- Determine the strength of the relationships between the attributes whose co-occurrence in objects suggest those objects should be in the same class.
- Apply some algorithm to determine the class(s) to which each item will be assigned.

Class rules:

- A well-defined semantic definition should exist for each class.
- The size of the classes should be less.
- Within a class, one object should not dominate the class. For example, assume a thesaurus class called "computer" exists and it contains the objects (words/word phrases) "microprocessor," "286-processor," "386-processor" and "pentium." If the term "microprocessor" is found 85 per cent of the time and the other terms are used 5 per cent each, there is a strong possibility that using "microprocessor" as a synonym for "286- processor" will introduce too many errors. It may be better to place "microprocessor" into its own class.
- Whether an object can be assigned to multiple classes or just one must be decided at creation time.

There are additional important decisions associated with the generation of thesauri that are not part of item clustering. They are

1) Word coordination approach: specifies if phrases as well as individual terms are to be clustered
2) Word relationships: Aitchison and Gilchrist specified three types of relationships: equivalence, hierarchical and nonhierarchical. Equivalence relationships are the most common and represent synonyms. Hierarchical relationships where the class name is a

general term and the entries are specific examples of the general term. The previous example of "computer" class name and "microprocessor," "pentium," etc Non-hierarchical relationships cover other types of relationships such as "object"-"attribute" that would contain "employee" and "job title."

3) Homograph resolution: a homograph is a word that has multiple, completely different meanings. For example, the term "field" could mean a electronic field, a field of grass, etc.
4) Vocabulary constraints: this includes guidelines on the normalization and specificity of the vocabulary. Normalization may constrain the thesaurus to stems versus complete words.

6.2 Thesaurus Generation

There are three basic methods for generation of a thesaurus; hand crafted, co- occurrence, and header-modifier based. In header-modifier based thesauri term relationships are found based upon linguistic relationships. Words appearing in similar grammatical contexts are assumed to be similar. The linguistic parsing of the document discovers the following syntactical structures: Subject-Verb, Verb- Object, Adjective-Noun, and Noun-Noun. Each noun has a set of verbs, adjectives and nouns that it co-occurs with, and a mutual information value is calculated for each using typically a log function.

6.2.1 Manual Clustering

The art of manual thesaurus construction resides in the selection of the set of words to be included. . Care is taken to not include words that are unrelated to the domain of the thesaurus. If a

concordance is used, other tools such as KWOC, KWIC or KWAC may help in determining useful words. A Key Word Out of Context (KWOC) is another name for a concordance. Key Word In Context (KWIC) displays a possible term in its phrase context. It is structured to identify easily the location of the term under consideration in the sentence. Key Word And Context (KWAC) displays the keywords followed by their context.

KWOC

TERM	FREQ	ITEM Ids
chips	2	doc2, doc4
computer	3	doc1, doc4, doc10
design	1	doc4
memory	3	doc3, doc4, doc8, doc12

KWIC

chips/	computer design contains memory
computer	design contains memory chips/
design	contains memory chips/ computer
memory	chips/ computer design contains

KWAC

chips	computer design contains memory chips
computer	computer design contains memory chips
design	computer design contains memory chips
memory	computer design contains memory chips

Figure 6.1 Example of KWOC, KWIC and KWAC

In the Figure 6.1 the character "/" is used in KWIC to indicate the end of the phrase. The KWIC and KWAC are useful in determining the meaning of homographs.

Once the terms are selected they are clustered based upon the word relationship guidelines and the interpretation of the strength of the relationship. This is also part of the art of manual creation of the thesaurus, using the judgment of the human analyst.

6.2.2 Automatic Term Clustering

There are many techniques for the automatic generation of term clusters to create statistical thesauri. When the number of clusters created is very large, the initial clusters may be used as a starting point to generate more abstract clusters creating a hierarchy. The basis for automatic generation of a thesaurus is a set of items that represents the vocabulary to be included in the thesaurus. Selection of this set of items is the first step of determining the domain for the thesaurus. The processing tokens (words) in the set of items are the attributes to be used to create the clusters.

Implementation of the other steps differs based upon the algorithms being applied. The automated method of clustering documents is based upon the polythetic clustering where each cluster is defined by a set of words and phrases. Inclusion of an item in a cluster is based upon the similarity of the item's words and phrases to those of other items in the cluster.

6.2.2.1 Complete Term Relation Method

In the complete term relation method, the similarity between every term pair is calculated as a basis for determining the clusters. The easiest way to understand this approach is to consider the vector model. The vector model is represented by a matrix where the rows are individual items and the

columns are the unique words (processing tokens) in the items. The values in the matrix represent how strongly that particular word represents concepts in the item.

Figure 6.2 provides an example of a database with 5 items and 8 terms. To determine the relationship between terms, a similarity measure is required. The measure calculates the similarity between two terms. In Chapter 7 a number of similarity measures are presented. The similarity measure is not critical

	Term1	Term2	Term3	Term4	Term5	Term6	Term7	Term8
Item 1	0	4	0	0	0	2	1	3
Item 2	3	1	4	3	1	2	0	1
Item 3	3	0	0	0	3	0	3	0
Item 4	0	1	0	3	0	0	2	0
Item 5	2	2	2	3	1	4	0	2

Figure 6.2 Vector Example

in understanding the methodology so the following simple measure is used:

$$SIM(Term_i, Term_j) = \Sigma (Term_{k,i}) (Term_{k,j})$$

where "k" is summed across the set of all items. In effect the formula takes the two columns of the two terms being analyzed, multiplying and accumulating the values in each row. The results can be paced in a resultant "m" by "m" matrix, called a Term-Term Matrix (Salton-83), where "m" is the number of columns (terms) in the original matrix. This simple formula is reflexive so that the matrix that is generated is symmetric. Other similarity formulas could produce a non- symmetric matrix.

Using the data in Figure 6.2, the Term-Term matrix produced is shown in Figure 6.3. There are no values on the diagonal since that represents the auto correlation of a word to itself. The next step is to select a threshold that determines if two terms are considered similar enough to each other to be in the same class. In this example, the threshold value of 10 is used. Thus two terms are considered similar if the similarity value between them is 10 or greater. This produces a new binary matrix called the Term Relationship matrix (Figure 6.4) that defines which terms are similar.

A one in the matrix indicates that the terms specified by the column and the row are similar enough to be in the same class. Term 7 demonstrates that a term may exist on its own with no other similar terms identified. In any of the clustering processes described below this term will always migrate to a class by itself.

The final step in creating clusters is to determine when two objects (words) are in the same cluster. There are many different algorithms available. The following algorithms are the most common: cliques, single link, stars and connected components.

	Term1	Term2	Term3	Term4	Term5	Term6	Term7	Term8
Term 1		7	16	15	14	14	9	7
Term 2	7		8	12	3	18	6	17
Term 3	16	8		18	6	16	0	8
Term 4	15	12	18		6	18	6	9
Term 5	14	3	6	6		6	9	3
Term 6	14	18	16	18	6		2	16
Term 7	9	6	0	6	9	2		3
Term 8	7	17	8	9	3	16	3	

Figure 6.3 Term-Term Matrix

	Term1	Term2	Term3	Term4	Term5	Term6	Term7	Term8
Term 1		0	1	1	1	1	0	0
Term 2	0		0	1	0	1	0	1
Term 3	1	0		1	0	1	0	0
Term 4	1	1	1		0	1	0	0
Term 5	1	0	0	0		0	0	0
Term 6	1	1	1	1	0		0	1
Term 7	0	0	0	0	0	0		0
Term 8	0	1	0	0	0	1	0	

Figure 6.4 Term Relationship Matrix

Applying the algorithm to Figure 6.4, the following classes are created: Class 1 (Term 1, Term 3, Term 4, Term 6)

Class 2 (Term 1, Term 5)

Class 3 (Term 2, Term 4, Term 6)

Class 4 (Term 2, Term 6, Term 8)

Class 5 (Term 7)

Notice that Term 1 and Term 6 are in more than one class. A characteristic of this approach is that terms can be found in multiple classes. In single link clustering the strong constraint that every term in a class is similar to every other term is relaxed. The rule to generate single link clusters is that any term that is similar to any term in the cluster can be added to the cluster. It is impossible for a term to be in two different clusters. This in effect partitions the set of terms into the clusters. The algorithm is:

1. Select a term that is not in a class and place it in a new class
2. Place in that class all other terms that are related to it
3. For each term entered into the class, perform step 2
4. When no new terms can be identified in step 2, go to step 1.

Applying the algorithm for creating clusters using single link to the Term Relationship Matrix, Figure 6.4, the following classes are created:

Class 1 (Term 1, Term 3, Term 4, Term 5, Term 6, Term 2, Term 8)

Class 2 (Term 7)

There are many other conditions that can be placed on the selection of terms to be clustered.

UNIT-IV

User Search Techniques: Search statements and binding, Similarity measures and ranking, Relevance feedback, Selective dissemination of information search, weighted searches of Boolean systems, Searching the Internet and hypertext. **Information Visualization:** Introduction, Cognition and perception, Information visualization technologies.

Search Statements and Binding

Search statements are the statements of an information need generated by users to specify the concepts they are trying to locate in items.
In generation of the search statement, the user may have the ability to weight (assign an importance) to different concepts in the statement. At this point the binding is to the vocabulary and past experiences of the user. Binding in this sense is when a more abstract form is redefined into a more specific form. The search statement is the user's attempt to specify the conditions needed to subset logically the total item space to that cluster of items that contains the information needed by the user.
The next level of binding comes when the search statement is parsed for use by a specific search system.
The final level of binding comes as the search is applied to a specific database. This binding is based upon the statistics of the processing tokens in the database and the semantics used in the database. This is especially true in statistical and concept indexing systems.
Figure 7.1 illustrates the three potential different levels of binding. Parenthesis are used in the second binding step to indicate expansion by a thesaurus.

INPUT	Binding
"Find me information on the impact of the oil spills in Alaska on the price of oil"	User search statement using vocabulary of user
impact, oil (petroleum), spills (accidents), Alaska, price (cost, value)	Statistical system binding extracts processing tokens
impact (.308), oil (.606), petroleum (.65), spills (.12), accidents (.23), Alaska (.45), price (.16), cost (.25), value (.10)	Weights assigned to search terms based upon inverse document frequency algorithm and database

Figure 7.1 Examples of Query Binding

Similarity Measures and Ranking
A variety of different similarity measures can be used to calculate the similarity between the item and the search statement. A characteristic of a similarity formula is that the results of the formula increase as the items become more similar. The value is zero if the items are totally dissimilar. An example of a simple "sum of the products" similarity measure from the examples in Chapter 6 to determine the similarity between documents for clustering purposes is:

$$SIM(Item_i, Item_j) = \Sigma (Term_{i,k}) (Term_{j,k})$$

This formula uses the summation of the product of the various terms of two items when treating the index as a vector. If is replaced with then the same formula generates the similarity between every Item and The problem with this simple measure is in the normalization needed to account for variances in the length of items. Additional normalization is also used to have the final results come between zero and +1 (some formulas use the range -1 to +1)

This assumption of the availability of relevance information in the weighting process was later relaxed

by Croft and Harper (Croft-79). Croft expanded this original concept, taking into account the frequency of occurrence of terms within an item producing the following similarity formula (Croft-83):

$$\text{SIM}(\text{DOC}_i, \text{QUERY}_j) = \sum_{t=1}^{Q} (C + IDF_i) * f_{i,j})$$

where C is a constant used in tuning, IDF_i is the inverse document frequency for term "i" in the collection and

$$f_{i,j} = \text{K} + (\text{K} - 1)\ \text{TF}_{i,j}/\text{maxfreq}_j$$

where K is a tuning constant, is the frequency of "i" and is the maximum frequency of any term in item "j." The best values for K seemed to range between 0.3 and 0.5. Another early similarity formula was used by Salton in the SMART system (Salton-83). Salton treated the index and the search query as ndimensional vectors (see Chapter 5). To determine the "weight" an item has with respect to the search statement, the Cosine formula is used to calculate the distance between the vector for the item and the vector for the query:

$$\text{SIM}(\text{DOC}_i, \text{QUERY}_j) = \frac{\sum_{k=1}^{n} (DOC_{i,k} * QTERM_{j,k})}{\sqrt{\sum_{k=1}^{n} (DOC_{i,k})^2 * \sum_{k=1}^{n} (QTERM_{j,k})^2}}$$

where is the kth term in the weighted vector for Item "i" and is the kth term in query "j." The Cosine formula calculates the Cosine of the angle between the two vectors. As the Cosine approaches "1," the two vectors become coincident (i.e., the term and the query represent the same concept). If the two are totally unrelated, then they will be orthogonal and the value of the Cosine is "0." What is not taken into account is the length of the vectors

For example, if the following vectors are in a three dimensional (three term) system:

Item = (4, 8, 0)

Query 1 = (1, 2, 0)

Query 2= (3, 6, 0)

$$\text{QTERM}_{i,k} = (0.5 + (0.5\ \text{TF}_{i,k}/\text{maxfreq}_k)) * \text{IDF}_i$$

$$SIM(DOC_i, QUERY_j) = \frac{\sum_{k=1}^{n} (DOC_{i,k} * QTERM_{j,k})}{\sum_{k=1}^{n} DOC_{i,k} + \sum_{k=1}^{n} QTERM_{j,k} - \sum_{k=1}^{n} (DOC_{i,k} * QTERM_{j,k})}$$

The Dice measure simplifies the denominator from the Jaccard measure and introduces a factor of 2 in the numerator. The normalization in the Dice formula is also invariant to the number of terms in common.

$$SIM(DOC_i, QUERY_j) = \frac{2 * \sum_{k=1}^{n} (DOC_{i,k} * QTERM_{j,k})}{\sum_{k=1}^{n} DOC_{i,k} + \sum_{k=1}^{n} QTERM_{j,k}}$$

QUERY = (2, 2, 0, 0, 4)
DOC1 = (0, 2, 6, 4, 0)
DOC2 = (2, 6, 0, 0, 4)

	Cosine	Jaccard	Dice
DOC1	36.66	16	20
DOC2	36.66	–12	20

Figure 7.2 Normalizing Factors for Similarity Measures

Figure 7.3 illustrates the threshold process. The simple "sum of the products" similarity formula is used to calculate similarity between the query and each document. If no threshold is specified, all three documents are considered hits. If a threshold of 4 is selected, then only DOC1 is returned.

Vector:	American, geography, lake, Mexico, painter, oil, reserve, subject
DOC1	geography *of* Mexico *suggests* oil reserves *are available* vector (0, 1, 0, 2, 0, 3, 1, 0)
DOC2	American geography *has* lakes *available everywhere* vector (1, 3, 2, 0, 0, 0, 0, 0)
DOC3	painters *suggest* Mexico lakes *as* subjects vector (0, 0, 1, 3, 3, 0, 0, 2)
QUERY	oil reserves *in* Mexico vector (0, 0, 0, 1, 0, 1, 1, 0)

SIM(Q, DOC1) = 6, SIM (Q, DOC2) = 0, SIM(Q, DOC3) = 3

Figure 7.3 Query Threshold Process

One special area of concern arises from search of clusters of terms that are stored in a hierarchical scheme (see Chapter 6). The items are stored in clusters that are represented by the centroid for each cluster. Figure 7.4 shows a cluster representation of an item space. In Figure 7.4, each letter at the leaf (bottom nodes) represent an item (i.e., K, L, M, N, D, E, F, G, H, P, Q, R, J). The letters at the higher nodes (A, C, B, I) represent the centroid of their immediate children nodes. The hierarchy is used in search by performing a top-down process. The query is compared to the centroids "A" and "B." If the results of the similarity measure are above the threshold, the query is then applied to the nodes' children. If not, then that part of the tree is pruned and not searched. This continues until

the actual leaf nodes that are not pruned are compared. The problem comes from the nature of a centroid which is an average of a collection of items (in Physics, the center of gravity). The risk is that the average may not be similar enough to the query for continued search, but specific items used to calculate the centroid may be close enough to satisfy the search. The risks of missing items and thus reducing recall increases as the standard deviation increases. Use of centroids reduces the similarity computations but could cause a decrease in recall. It should have no effect on precision since that is based upon the similarity calculations at the leaf (item) level.

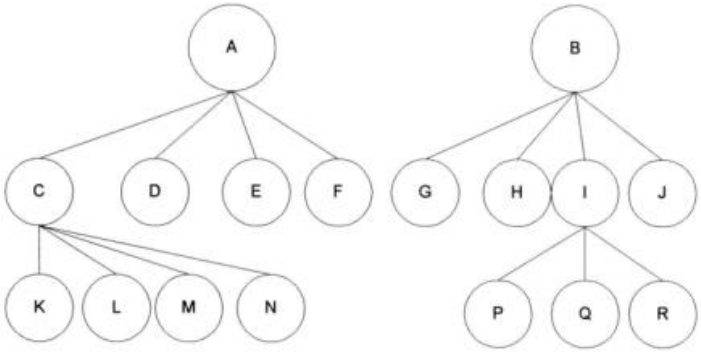


Figure 7.4 Item Cluster Hierarchy

Hidden Markov Models Techniques

Use of Hidden Markov Models for searching textual corpora has introduced a new paradigm for search. In most of the previous search techniques, the query is thought of as another "document" and the system tries to find other documents similar to it. In HMMs the documents are considered unknown statistical processes that can generate output that is equivalent to the set of queries that would consider the document relevant. Another way to look at it is by taking the general definition that a HMM is defined by output that is produced by passing some unknown key via state transitions through a noisy channel. The observed output is the query, and the unknown keys are the relevant documents. The noisy channel is the mismatch between the author's way of expressing ideas and the user's ability to specify his query. Leek, Miller and Schwartz (Leek-99) computed for each document the probability that D was the relevant document in the users mind given that Q was the query produced, i.e., P(D is R/Q). The development for a HMM approach begins with applying Bayes rule to the conditional probability

$$P(D \text{ is } R/Q) = P(Q/D \text{ is } R) * P(D \text{ is } R) / P(Q)$$

The biggest problem in using this approach is to estimate the transition probability matrix and the output (queries that could cause hits) for every document in the corpus.

Ranking Algorithms

A by-product of use of similarity measures for selecting Hit items is a value that can be used in ranking the output. Ranking the output implies ordering the output from most likely items that satisfy the query to least likely items. This reduces the user overhead by allowing the user to display the most likely relevant items first. The original Boolean systems returned items ordered by date of entry into the system versus by likelihood of relevance to the user's search statement. With the inclusion of statistical similarity techniques into commercial systems and the large number of hits that originate from searching diverse corpora, such as the Internet, ranking has become a common feature of modern systems. A summary of ranking algorithms from the research community is found in an article written by Belkin and Croft (Belkin-87).

Relevance Feedback

The first major work on relevance feedback was published in 1965 by Rocchio (republished in 1971: Rocchio-71). Rocchio was documenting experiments on reweighting query terms and query expansion based upon a vector representation of queries and items. The concepts are also found in the probabilistic model presented by Robertson and Sparck Jones (Robertson-76). The relevance feedback concept was that the new query should be based on the old query modified to increase the weight of

$$Q_n = Q_o + \frac{1}{r}\sum_{i=1}^{r} DR_i - \frac{1}{nr}\sum_{j=1}^{nr} DNR_j$$

where

Q_n	= the revised vector for the new query
Q_o	= the original query
r	= number of relevant items
DR_i	= the vectors for the relevant items
nr	= number of non-relevant items
DNR_j	= the vectors for the non-relevant items.

The factors r and nr were later modified to be constants that account for the number of items along with the importance of that particular factor in the equation. Additionally a constant was added to Q_o to allow adjustments to the importance of the weight assigned to the original query. This led to the revised version of the formula:

$$Q_n = \alpha Q_o + \beta\sum_{i=1}^{r} DR_i - \gamma\sum_{j=1}^{nr} DNR_j$$

Terms in relevant items and decrease the weight of terms that are in non-relevant items. This technique not only modified the terms in the original query but also allowed expansion of new terms from the relevant items. The formula used is:

Where and are the constants associated with each factor (usually 1/n or 1/nr times a constant). The factor is referred to as positive feedback because it is using the user judgments on relevant items to increase the values of terms for the next iteration of searching. The factor is referred to as negative

Figure 7.6 gives an example of the impacts of positive and negative feedback. The filled circles represent non-relevant items; the other circles represent relevant items. The oval represents the items that are returned from the query. The solid box is logically where the query is initially. The hollow box is the query modified by relevance feedback (positive only or negative only in the Figure).

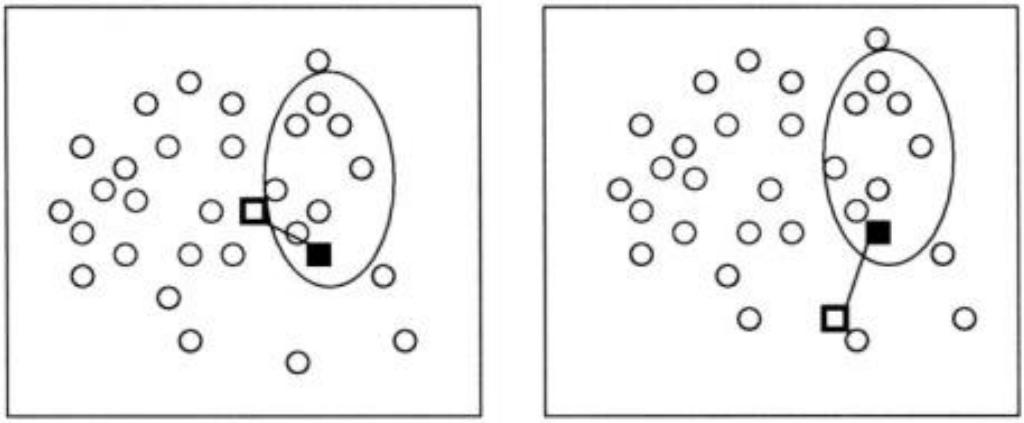


Figure 7.6 Impact of Relevance Feedback

Relevance feedback, in particular positive feedback, has been proven to be of significant value in producing better queries. Some of the early experiments on the SMART system (Ide-69, Ide-71, Salton-83) indicated the possible improvements that would be gained by the process. But the small collection sizes and evaluation techniques put into question the actual gains by using relevance feedback. One of the early problems addressed in relevance feedback is how to treat query terms that are not found in any retrieved relevant items. Just applying the algorithm would have the effect of reducing the relative weight of those terms with respect to other query terms. From the user's perspective, this may not be desired because the term may still have significant value to the user if found in the future iterations of the search process. Harper and van Rijisbergen addressed this issue in their proposed EMIM weighting scheme (Harper-78, Harper-80). Relevance feedback has become a common feature in most information systems. When the original query is modified based upon relevance feedback, the systems ensure that the original query terms are in the modified query, even if negative feedback would have eliminated them. In some systems the modified query is presented to the user to allow the user to readjust the weights and review the new terms added.

Selective Dissemination of Information Search

Selective Dissemination of Information, frequently called dissemination systems, are becoming more prevalent with the growth of the Internet. A dissemination system is sometimes labeled a "push" system while a search system is called a "pull" system. The differences are that in a search system the user proactively makes a decision that he needs information and directs the query to the information system to search. In a dissemination system, the user defines a profile (similar to a stored query) and as new information is added to the system it is automatically compared to the user's profile.

Weighted Searches of Boolean Systems

The two major approaches to generating queries are Boolean and natural language. Natural language queries are easily represented within statistical models and are usable by the similarity measures discussed. Issues arise when Boolean queries are associated with weighted index systems. Some of the issues are associated with how the logic (AND, OR, NOT) operators function with weighted values and how weights are associated with the query terms. If the operators are interpreted in their normal interpretation, thay act too restrictive or too general (i.e., AND and OR operators respectively). Salton, Fox and Wu showed that using the strict definition of the operators will suboptimize the retrieval expected by the user (Salton-83a). Closely related to the strict definition

problem is the lack of ranking that is missing from a pure Boolean process. Some of the early work addressing this problem recognized the fuzziness associated with mixing Boolean and weighted systems (Brookstein-78, Brookstein-80) To integrate the Boolean and weighted systems model, Fox and Sharat proposed a fuzzy set approach (Fox-86). Fuzzy sets introduce the concept of degree of membership to a set (Zadeh-65). The degree of membership for AND and OR operations are defined as:

$$DEG_{A\cap B} = \min(DEG_A, DEG_B)$$

$$DEG_{A\cup B} = \max(DEG_A, DEG_B)$$

$$SIM(QUERY_{OR}, DOC) = C_{OR\,1} * \max(DOC1_1, DOC_2, \ldots, DOC_n) + C_{OR2} * \min(DOC_1, DOC_2, \ldots, DOC_n)$$

$$SIM(QUERY_{AND}, DOC) = C_{AND1} * \min(DOC_1, DOC_2, \ldots, DOC_n) + C_{AND2} * \max(DOC1_1, DOC_2, \ldots, DOC_n)$$

The MMM technique was expanded by Paice (Paice-84) considering all item weights versus the maximum/minimum approach. The similarity measure is calculated as:

$$SIM(QUERY\ DOC) = \sum_{i=1}^{n} r^{i-1} d_i \ / \ \sum_{i=1}^{n} r^{i-1}$$

$$Q_{OR} = (A_1, a_1)\ OR\ (A_2, a_2)\ OR\ \ldots OR\ (A_n, a_n)$$

$$Q_{AND} = (A_1, a_1)\ AND\ (A_2, a_2)\ AND\ \ldots\ AND\ (A_n, a_n)$$

Searching the INTERNET and Hypertext

The Internet has multiple different mechanisms that are the basis for search of items. The primary techniques are associated with servers on the Internet that create indexes of items on the Internet and allow search of them. Some of the most commonly used nodes are YAHOO, AltaVista and Lycos. In all of these systems there are active processes that visit a large number of Internet sites and retrieve textual data which they index. The primary design decisions are on the level to which they retrieve data and their general philosophy on user access. LYCOS (http://www.lycos.com) and AltaVista automatically go out to other Internet sites and return the text at the sites for automatic indexing (http://www.altavista.digital.com). Lycos returns home pages from each site for automatic indexing while Altavista indexes all of the text at a site. The retrieved text is then used to create an index to the source items storing the Universal Resource Locator (URL) to provide to the user to retrieve an item. All of the systems use some form of ranking algorithm to assist in display of the retrieved items. The algorithm is kept relatively simple using statistical information on the occurrence of words within the retrieved text

Closely associated with the creation of the indexes is the technique for accessing nodes on the Internet to locate text to be indexed. This search process is also directly available to users via Intelligent Agents. Intelligent Agents provide the capability for a user to specify an information

need which will be used by the Intelligent Agent as it independently moves between Internet sites locating information of interest. There are six key characteristics of intelligent agents (Heilmann-96):

1. Autonomy - the search agent must be able to operate without interaction with a human agent. It must have control over its own internal states and make independent decisions. This implies a search capability to traverse information sites based upon pre-established criteria collecting potentially relevant information.

2. Communications Ability - the agent must be able to communicate with the information sites as it traverses them. This implies a universally accepted language defining the external interfaces (e.g., Z39.50).

3. Capacity for Cooperation - this concept suggests that intelligent agents need to cooperate to perform mutually beneficial tasks.

4. Capacity for Reasoning - There are three types of reasoning scenarios (Roseler-94): Rule-based - where user has defined a set of conditions and actions to be taken Knowledge-based - where the intelligent agents have stored previous conditions and actions taken which are used to deduce future actions Artificial evolution based - where intelligent agents spawn new agents with higher logic capability to perform its objectives.

5. Adaptive Behavior - closely tied to 1 and 4 , adaptive behavior permits the intelligent agent to assess its current state and make decisions on the actions it should take

6. Trustworthiness - the user must trust that the intelligent agent will act on the user's behalf to locate information that the user has access to and is relevant to the user.

Information Visualization

Functions that are available with electronic display and visualization of data that were not previously provided are:

- modify representations of data and information or the display condition (e.g., changing color scales)
- use the same representation while showing changes in data (e.g., moving between clusters of items showing new linkages)
- animate the display to show changes in space and time
- Create hyperlinks under user control to establish relationships between data

Information Visualization addresses how the results of a search may be optimally displayed to the users to facilitate their understanding of what the search has provided and their selection of most likely items of interest to read. Cognitive(the mental action or process of acquiring knowledge and understanding through thought, experience, and the senses) engineering derives design principles for visualization techniques from what we know about the neural processes involved

with attention, memory, imagery and information processing of the human visual system.

Cognitive engineering results can be applied to methods of reviewing the concepts contained in items selected by search of an information system. Visualization can be divided into two broad classes: link visualization and attribute (concept) visualization. Link visualization displays relationships among items. Attribute visualization reveals content relationships across large numbers of items.

There are many areas that information visualization and presentation can help the user:

a. reduce the amount of time to understand the results of a search and likely clusters of relevant information
b. yield information that comes from the relationships between items versus treating each item as independent
c. perform simple actions that produce sophisticated information search functions

Visualization is the transformation of information into a visual form which enables the user to observe and understand the information.

Cognition (the mental action or process of acquiring knowledge and understanding through thought, experience, and the senses)

Perception (the ability to see, hear, or become aware of something through the senses)

Proximity - nearby figures are grouped together

Similarity - similar figures are grouped together

Continuity - figures are interpreted as smooth continuous patterns rather than discontinuous concatenations of shapes (e.g., a circle with its diameter drawn is perceived as two continuous shapes, a circle and a line, versus two half circles concatenated together)

Closure - gaps within a figure are filled in to create a whole (e.g., using dashed lines to represent a square does not prevent understanding it as a square) Connectedness - uniform and linked spots, lines or areas are perceived as a single unit

Aspects of the Visualization Process

One of the first-level cognitive processes is pre attention, that is, taking the significant visual information from the photoreceptors and forming primitives. In Figure 8.1 the visual system detects the difference in orientations between the left and middle portion of the figure and determines the logical border between them. An example of using the conscious processing capabilities of the brain is the detection of the different shaped objects and the border between them shown between the left side and middle of the Figure 8.1. The reader can likely detect the differences in the time it takes to visualize the two different boundaries.

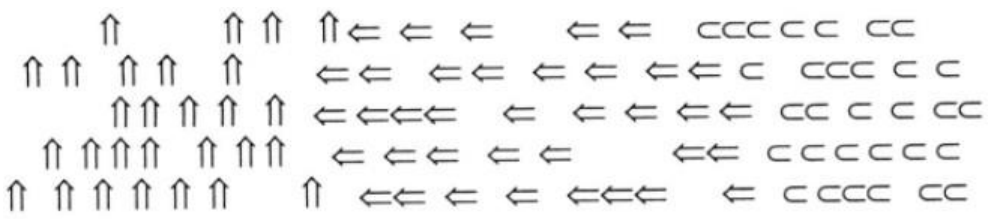


Figure 8.1 Preattentive Detection Mechanism

The preattentive process can detect the boundaries between orientation groups of the same object. A harder process is to identify the equivalence of rotated objects. For example, a rotated square requires more effort to recognize it as a square. As we migrate into characters, the problem of identification of the character is affected by rotating the character in a direction not normally encountered. It is easier to detect the symmetry when the axis is vertical. Figure 8.2 demonstrates these effects.

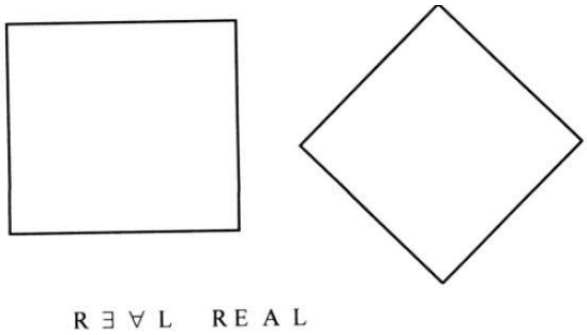


Figure 8.2 Rotating a Square and Reversing Letters in "REAL"

Color is one of the most frequently used visualization techniques to organize, classify, and enhance features.

The goals for displaying the result from searches fall into two major classes: document clustering and search statement analysis. The goal of document clustering is to present the user with a visual representation of the document space constrained by the search criteria. Within this constrained space there exist clusters of documents defined by the document content. Visualization tools in this area attempt to display the clusters, with an indication of their size and topic, as a basis for users to navigate to items of interest.

The second goal is to assist the user in understanding why items were retrieved, thereby providing information needed to refine the query. Visualization techniques approach this problem by displaying the total set of terms, including additional terms from relevance feedback or thesaurus expansion, along with documents retrieved and indicate the importance of the term to the retrieval and ranking process.

Link analysis is also important because it provides aggregate-level information within an information system. One way of organizing information is hierarchical. A two-dimensional representation becomes difficult for a user to understand as

The hierarchy becomes large. One of the earliest experiments in information visualization was the Information Visualizer developed by XEROX PARC. It incorporates various visualization formats such as DataMap,

InfoGrid, ConeTree, and the Perspective wall. The Cone-Tree is a 3-Dimensional representation of data, where one node of the tree is represented at the apex and ail the information subordinate to it is arranged in a circular structure at its base.

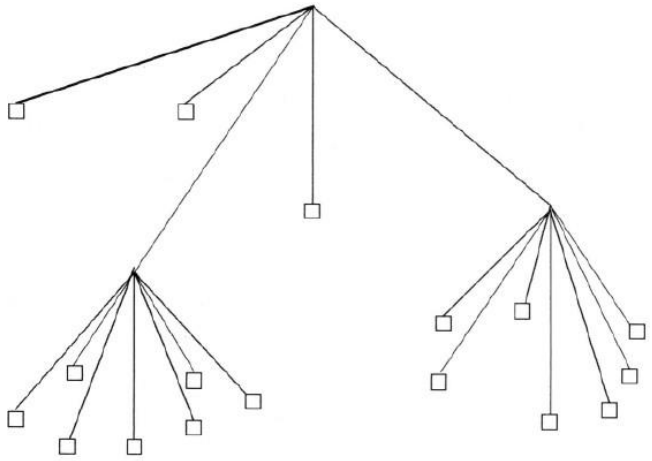


Figure 8.4 Cone Tree

Thus a six-dimensional coordinate space may have three of the coordinates defined as a subspace within the other three coordinate spaces. This has been called Feiner's "worlds within worlds"

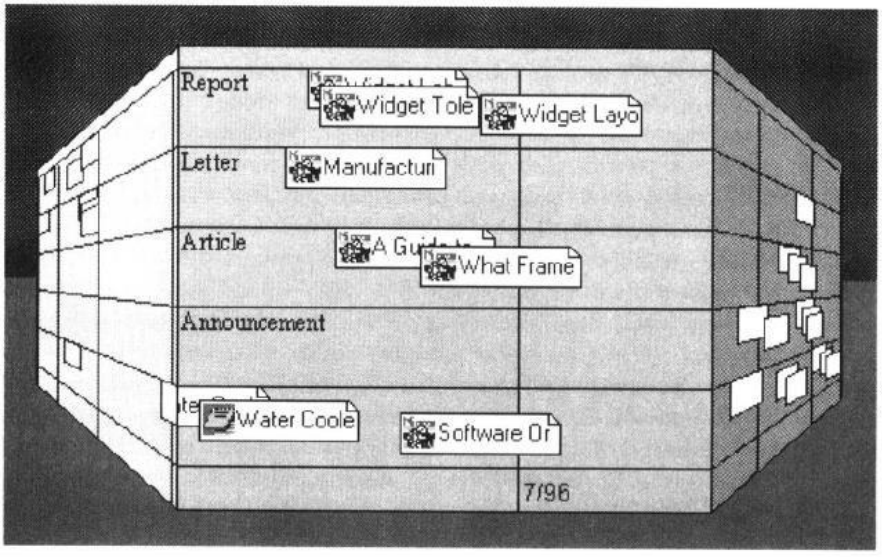


Figure 8.5 Perspective Wall
From inXight web site - www.inxight.com

approach.

Another clustering system that uses statistical information for a small number of items (50 - 120) to show term relationships via spatial positioning is the VIBE system . The VIBE system allows users to associate query terms with different locations in the visual display.

Lin has taken the self-organization concept further by using Kohonen's algorithm to automatically determine a table of contents (TOC) and display the results in a map display. Visualization tools need to assist the user in understanding the effects of his search statement even to the level of identifying important terms that are not contributing to the search process. One solution is a graphical display of the characteristics of the retrieved items which contributed to their selection. This is effected in the Envision system.

Figure 8.7 shows Envision's three interactive windows to display search results: Query window, Graphic View window, and Item Summary window. The Query window provides an editable version of the query. The Item Summary window provides bibliographic citation information on items selected in the Graphic View window. The Graphic View window is similar to scatterplot graphs. Each item in the Hit file is represented by an icon in the window. Selecting an item in the window provides bibliographic information on the same display. Circles represent single items with the relevance weights displayed below them.

Ellipses represent clusters of multiple items that are located at the same point in the scatterplot with the number of items in the ellipse and their weights below the ellipse. In this example, estimated relevance is on the X-axis and author's name is on the Y-axis. This type of interface provides a very user friendly environment but encounters problems when the number of relevant items and entries for an axis becomes very large. Envision plans to address this issue by a "zoom" feature that will allow seeing larger areas of the scatterplot at lesser detail.

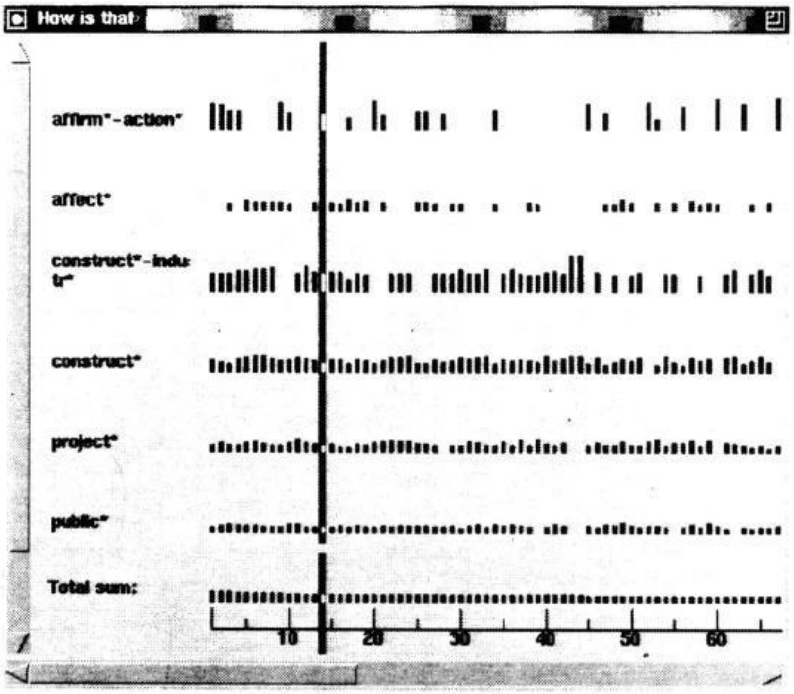


Figure 8.8 Visualization of Results
(from SIGIR 96, page 88)

A similar technique is used by Veerasamy and Belkin (Veerasamy-96). They use a series of vertical columns of bars. The columns of bars represent documents, and the rows represent index terms. The height of the bar corresponds to the weight of the corresponding term (row) in the corresponding item (column). In addition to the query terms, the system shows the additional words added to the system by relevance feedback. Figure 8.8 provides an example for a search statement of "How affirmative action affected the construction industry." This approach quickly allows a user to determine which terms had the most effect on retrieving a specific item (i.e. by scanning down the column). It also allows the user to determine how the various terms contributed to the retrieval process (i.e. by scanning a row).

This latter process is very important because it allows a user to determine if what he considers to be an important search term is not contributing strongly or not found at all in the items being retrieved. It also shows search terms that are causing items to be retrieved allowing their removal or reduction in query weight if they are causing false hits. In the Boolean environment this function was accomplished by vocabulary browsing (see Chapter 2) that allows for a user to see the number of items a particular term is in prior to including it in a search.

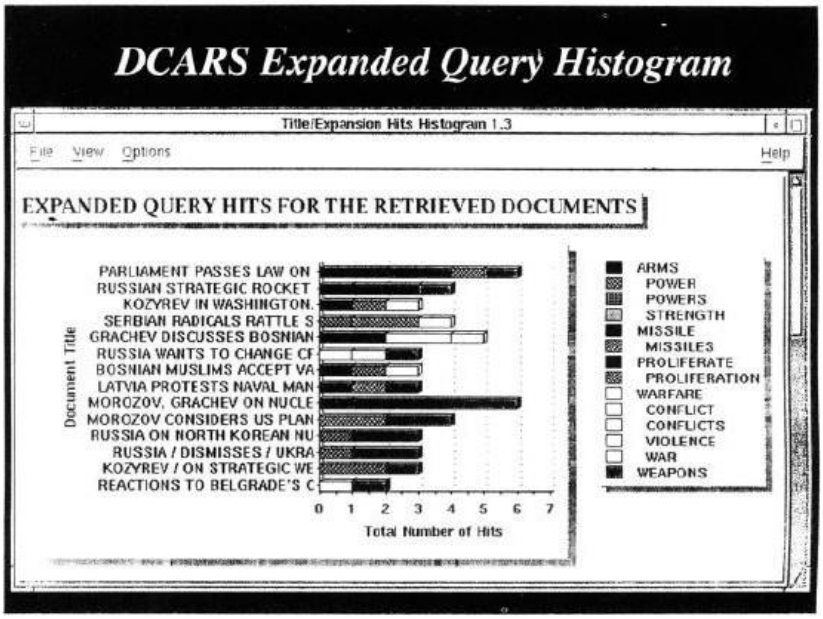


Figure 8.9 Example of DCARS Query Histogram
(from briefing by CALSPAN)

A slightly different commercial version having properties similar to the systems above is the Document Content Analysis and Retrieval System (DCARS) being developed by Calspan Advanced Technology Center. Their system is designed to augment the RetrievalWare search product. They display the query results as a histogram with the items as rows and each term's contribution to the selection indicated by the width of a tile bar on the row (see Figure 8.9). DCARS provides a friendly user interface that indicates why a particular item was found, but it is much harder to use the information in determining how to modify search statements to improve them.

Another representation that is widely used for both hierarchical and network related information is the "cityscape" which uses the metaphor of movement within a city. In lieu of using hills, as in the terrain approach, skyscrapers represent the theme (concept) area as shown in Figure 8.10. This is similar to extending bar charts to three dimensions. Buildings can be connected by lines which can vary in representation to describe interrelationships between themes. Colors or fill designs can be used for the visualization presenting another layer of information (e.g., the building having the same color may be members of a higher concept). Movement within the cityscape (or terrain) of the viewer perspective allows zooming in on specific information areas that will bring into view additional structures that might have been hidden by the previous viewpoint.

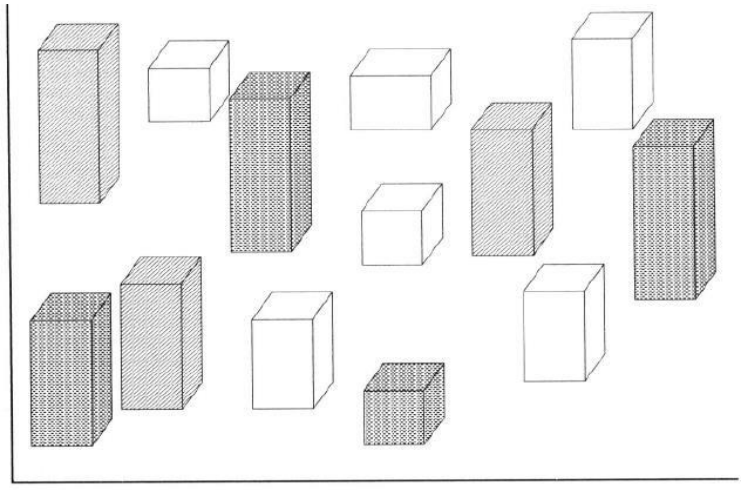


Figure 8.10 CityScape Example

Another task in information systems is the visualization of specific text within an item versus between items. In some situations, items are allowed to change over time via editing. Thus, there is both the static representation and a time varying representation. Text changing representations are very important when the text being represented is a software program of millions of lines of code.

AT&T Bell laboratories created the SeeSoft system which uses columns and color codes to show when different lines of code have been changed. This technique was used as a basis for a similar code visualization tool, DEC FUSE/SoftVis (Zaremba-95). They created small pictures of files that represent the code in the file with the size of the picture scaled to the number of lines of code in the file. Color coding indicates different characteristics of the code (e.g., green is comments). A user can quickly see the relative structure of all of the code files composing a system along with the complexity of each of the modules. The TileBars tool from Xerox PARC provides the user with a visualization of the distribution of query terms within each item in a Hit file. Using this tool, the user can quickly locate the section of the item that is most likely to be of interest.

The first visualization tool was the Query By Example user interface developed by IBM (Zloof-75). The interface presented the user with two dimensional tables on the display screen and based upon the user is defining values of interest on the tables, the system would complete the search. The Information Visualization and Exploration Environment (IVEE) makes use of the three dimensional representation of the structured database as constrained by

the user's search statement. MITRE Corporation has developed a tool used with web browsers that enables a user to see a tree structure visual representation of the information space they have navigated through.

Another area in information visualization is the representation of pattern and linkage analysis. A system that incorporates many information visualization techniques including those used to represent linkage analysis is the Pathfinder Project sponsored by the Army (Rose-96). It contains the Document Browser, CAMEO, Counts, CrossField Matrix, OILSTOCK and SPIRE tools. The Document Browser uses different colors and their density for words in the text of items to indicate the relative importance of the item to their profile of interest.

CAMEO models an analytic process by creating nodes and links to represent a problem. Queries are associated with the nodes. The color of the nodes change based on how well the found items satisfy the query. Counts uses statistical information on words and phrases and plots them over time. Time is used as a parameter to show trends in development of events. The display uses a three dimensional cityscape representation of the data. The Cross Field Matrix creates a two-dimensional matrix of two fields in a dataset.

The SPIRE tool is a type of scattergraph of information. Items are clustered and displayed in a star chart configuration. Distance between two points is indicative of their similarity based upon concurrence of terms.

UNIT- V

Text Search Algorithms: Introduction, Software text search algorithms, Hardware text search systems.

Information System Evaluation: Introduction, Measures used in system evaluation, Measurement example – TREC results.

Text Search Algorithms

Three classical text retrieval techniques have been defined for organizingitems in a textual database, for rapidly identifying the relevant items and foreliminating items that do not satisfy the search. The techniques are

1) Full text scanning (streaming)
2) Word inversion
3) Multiattribute retrieval

In addition to using the indexes as a mechanism for searching text ininformation systems, streaming of text was frequently found in the systems as anadditional search mechanism. The basic concept of a text scanning system is the ability for one or moreusers to enter queries, and the text to be searched is accessed and compared to thequery terms. When all of the text has been accessed, the query is complete.

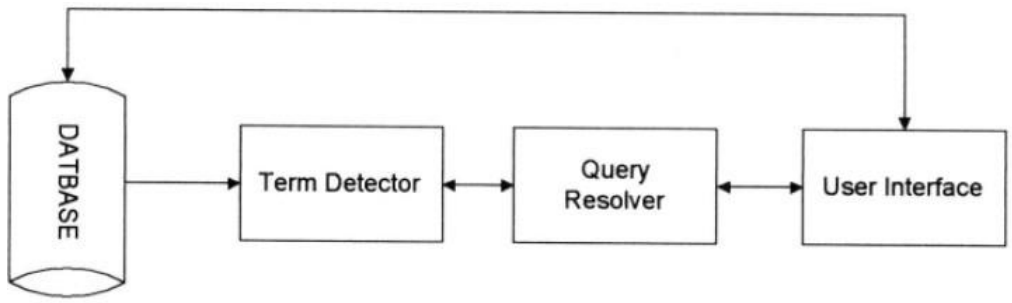


Figure 9.1 Text Streaming Architecture

The database contains thefull text of the items.The term detector is the special hardware/software that contains all of the terms being searched for and in some systems the logic betweenthe items. It will input the text and detect the existence of the search terms. It willoutput to the query resolver the detected terms to allow for final logical processingof a query against an item. The query resolver performs two functions.

It willaccept search statements from the users, extract the logic and search terms andpass the search terms to the detector. It also accepts results from the detector anddetermines which queries are satisfied by the item and possibility the weight associated with hit. The Query Resolver will

passinformation to the user interface that will be continually updating search status tothe user and on request retrieve any items that satisfy the user search statement. The worst casesearch for a pattern of *m* characters in a string of *n* characters is at least *n -m* + 1or a magnitude of O(*n*). Some of the original brute force methodscould require O(*n***m*) symbol comparisons. More recentimprovements have reduced the time to O(*n* + *m*).

In the case of hardware search machines, multiple parallel searchmachines (term detectors) may work against the same data stream allowing formore queries or against different data streams reducing the time to access thecomplete database. In software systems, multiple detectors may execute at thesame time.

There are two approaches to the data stream. In the first approach thecomplete database is being sent to the detector(s) functioning as a search of the
database. In the second approach random retrieved items are being passed to thedetectors. In this second case the idea is to perform an index search of the databaseand let the text streamer perform additional search logic that is not satisfied by theindex search.

Examples of limits of index searches are:

- Search for stop words
- Search for exact matches when stemming is performed
- Search for terms that contain both leading and trailing “don’t cares”
- Search for symbols that are on the inter-word symbol list (e.g., “ , ;)

The full text search function does not require anyadditional storage overhead. There is also the advantage where hits may bereturned to the user as soon as found. Typically in an index system, the completequery must be processed before any hits are determined or available. Streamingsystems also provide a very accurate estimate of current search status and time tocomplete the query. It is difficult to locate all the possible index valuesshort of searching the complete dictionary of possible terms.

Many of the hardware and software text searchers use finite stateautomata as a basis for their algorithms. A finite state automata is a logicalmachine that is composed of five elements:
I - a set of input symbols from the alphabet supported by the automata

S - a set of possible states

P - a set of productions that define the next state based upon the currentstate andinput symbol

$\mathbf{S_0}$- a special state called the initial state

$\mathbf{S_F}$- a set of one or more final states from the set **S**

9.2 Software Text Search Algorithms

In software streaming techniques, the item to be searched is read intomemory and then the algorithm is applied.

There are four major algorithms associated with software text search:

1) the brute force approach
2) Knuth-Morris-Pratt
3) Boyer-Moore, Shift-OR algorithm
4) Rabin-Karp.

Of all of the algorithms,Boyer-Moore has been the fastest requiring at most O(*n* + *m*) comparisons , Knuth-Pratt-Morris and Boyer-Moore both require O(*n*) preprocessing of search stringsThe Brute force approach is the simplest string matching algorithm. Theidea is to try and match the search string against the input text. If as soon as a mismatchis detected in the comparison process, shift the input text one position andstart the comparison process over. The expectednumber of comparisons whensearching an input text string of *n* characters for a pattern of *m* characters is

$N_c=c/c-1(1-1/c^m)*(n-m+1)+O(1)$

Where$N_c$is the expected number of comparisons and c is the size of the alphabetfor the text.

Knuth-Pratt-Morris(KPM) algorithm

Pattern:

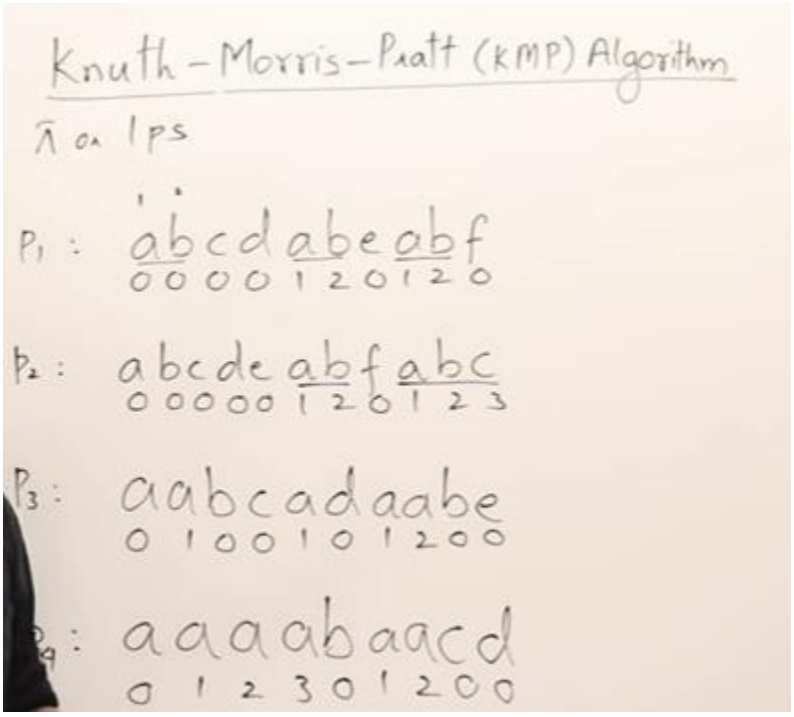
a	b	c	d	a	b	C
1	2	3	4	5	6	7

Now find out substrings as prefix, suffix by taking any number of characters from left to right and right to left.

Prefix: a, ab, abc, abcdetc

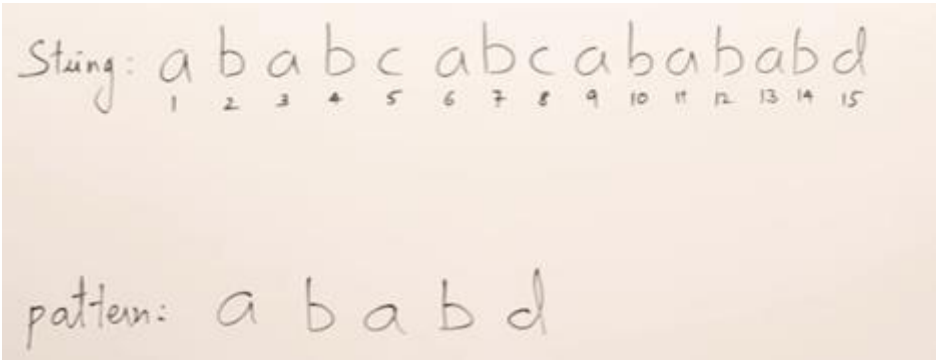
Suffix: c, bc, abc, dabcetc

From above prefix, suffix substrings we can observe a substring “abc” is there in both and also that is repeated twice in given pattern.

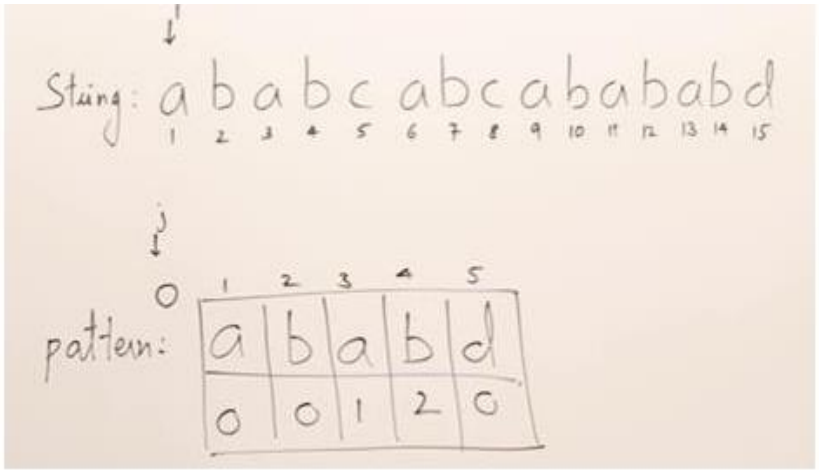


Example:

Given string and pattern is



Now construct table with repeated characters of pattern like following.



Here ‘a’ is repeated so keep index 1 below it, ‘b’ is repeated so keep index 2 below it and rest all ‘0’s. Now start search process.

Step-1: Initialize string index as I, pattern index as j. Start j from adding ‘0’ index.

Step-2: Compare string[i] and pattern [j+1]i,e, ‘a’ and ‘a’ both are matching so move both i and j to next position.

Step-3: Now compare string[i] and pattern [j+1] i.e, b and b matching so move both i, j to next position

When i=5 and j=4 string[i]=c and pattern[j+1]=d. here not matching then move j to its index location. i.e, 2. So now j position is pattern[2].

Now compare string[i]=c and pattern[j+1]=a. here not matching then move j to its index location i.e., 0. So now j position is pattern[0]. J is now on 0 position so we cannot move therefore move now I to next location i.e. 6.

Note: Here we can observe only j is moving back but not i. I is moving only in the forward direction.

Step-4: Repeat the process till find a match

https://www.youtube.com/watch?v=V5-7GzOfADQ **Boyer**

Moore Algorithm:

Step-1: Construct 'Bad Match Table'

Step-2: Compare right most character of pattern with given string based on the 'value' of bad match table

Step-3: If mismatch then shift the pattern to the right position corresponding to the 'value' of bad match table

While constructing bad match table use following formula for value. value=length of pattern-index-1 and last value=length of pattern

Constructing Bad Match Table

Example 1:

- Pattern – 'teammast'
- Text – 'welcometoteammast'

T E A M M A S T Length = 8

Index : 0 1 2 3 4 5 6 7

Letter	T	E	A	M	S	*
Value						

Constructing Bad Match Table

T E A M M A S T Length = 8

Index : 0 1 2 3 4 5 6 7

Letter	T	E	A	M	S	*
Value	7	6	5			

A = 8 - 2 - 1

Here the letter 'A' is occurring twice so replace the latest value by old one. In the same way for M also. T is the last character in pattern so its value=8(length of pattern)

Constructing Bad Match Table

T E A M M A S T Length = 8

Index : 0 1 2 3 4 5 6 7

Letter	T	E	A	M	S	*
Value	7	6	5	4		

M = 8 - 3 - 1

Constructing Bad Match Table

T E A M M A S T Length = 8

Index : 0 1 2 3 4 5 6 7

Letter	T	E	A	M	S	*
Value	7	6	5	3		

M = 8 - 4 - 1

Constructing Bad Match Table

T E A M M A S T Length = 8

Index : 0 1 2 3 4 5 6 7

Letter	T	E	A	M	S	*
Value	8	6	2	3	1	

T = 8 Last letter = length, if not already defined.

Boyer Moore Example

Letter	T	E	A	M	S	*
Value	8	6	2	3	1	8

Mismatch here so move 8 characters right hand side

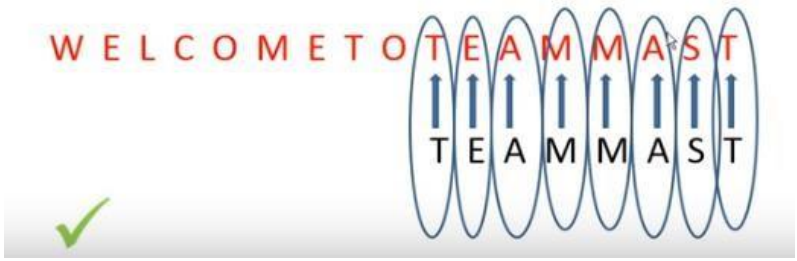
Boyer Moore Example

Letter	T	E	A	M	S	*
Value	8	6	2	3	1	8

Mismatch so move 1 character to the right hand side

Boyer Moore Example

Letter	T	E	A	M	S	*
Value	8	6	2	3	1	8



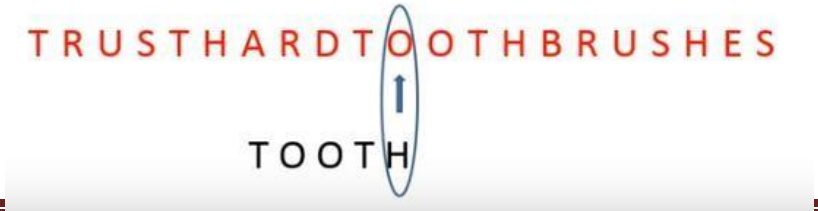
Letter	T	O	H	*
Value	1	2	5	5

Boyer Moore Example

Letter	T	O	H	*
Value	1	2	5	5

Boyer Moore Example

Letter	T	O	H	*
Value	1	2	5	5



9.3 Hardware Text Search Systems

Software text search is applicable to many circumstances but has encountered restrictions on the ability to handle many search terms simultaneously against the same text and limits due to I/O speeds. One approach that off loaded the resource intensive searching from the main processors was to have a specialized hardware machine to perform the searches and pass the results to the main computer which supported the user interface and retrieval of hits. Since the searcher is hardware based, scalability is achieved by increasing the number of hardware search devices.

Another major advantage of using a hardware text search unit is in the elimination of the index that represents the document database. Typically the indexes are 70% the size of the actual items. Other advantages are that new items can be searched as soon as received by the system rather than waiting for the index to be created and the search speed is deterministic.

Figure 9.1 represents hardware as well as software text search solutions.The arithmetic part of the system is focused on the term detector. There has been three approaches to implementing term detectors: parallel comparators or associative memory, a cellular structure, and a universal finite state automata.

When the term comparator is implemented with parallel comparators, each term in the query is assigned to an individual comparison element and input

data are serially streamed into the detector. When a match occurs, the term comparator informs the external query resolver (usually in the main computer) by setting status flags.

Specialized hardware that interfaces with computers and is used to search secondary storage devices was developed from the early 1970s with the most recent product being the **Parallel Searcher (previously the Fast Data Finder).** The typical hardware configuration is shown in Figure 9.9 in the dashed box. The speed of search is then based on the speed of the I/O.

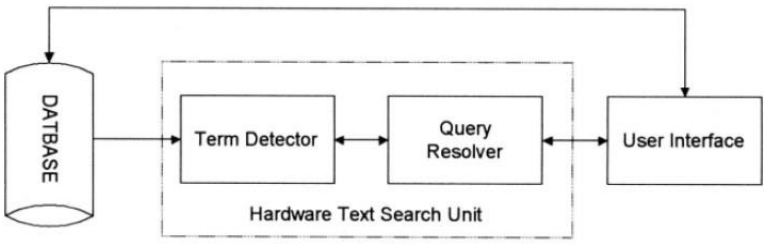


Figure 9.9 Hardware Text Search Unit

One of the earliest hardware text string search units was the ***Rapid Search*** Machine developed by General Electric. The machine consisted of a special purpose search unit where a single query was passed against a magnetic tape containing the documents. A more sophisticated search unit was developed by Operating Systems Inc. called the ***Associative File Processor (AFP).*** It is capable of searching against multiple queries at the same time. Following that initial development, OSI, using a different approach, developed the ***High SpeedText Search (HSTS) machine***. It uses an algorithm similar to the Aho- Corasick software finite state machine algorithm except that it runs three parallel state machines. One state machine is dedicated to contiguous word phrases, another for imbedded term match and the final for exact word match.

Inparallel with that development effort, GE redesigned their Rapid Search Machine into the ***GESCAN unit***. The GESCAN system uses a text array processor (TAP) that simultaneously matches many terms and conditions against a given text stream the TAP receives the query information from the user's computer and directly access the textual data from secondary storage. The TAP consists of a large cache memory and an array of four to 128 query processors. The text is loaded into the cache and searched by the query processors (Figure 9.10). Each query processor is independent and can be loaded at any time. A complete query is handled by each query processor.

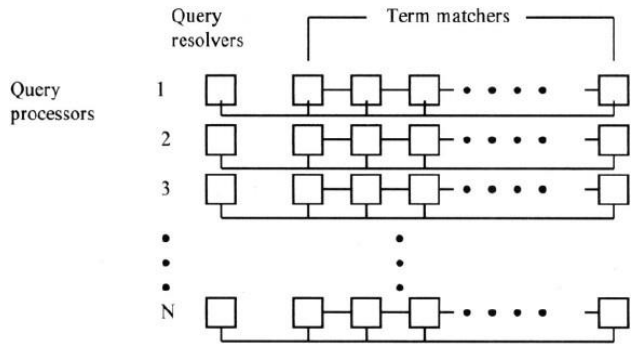


Figure 9.10 GESCAN Text Array Processor

A query processor works two operations in parallel; matching query terms to input text and Boolean logic resolution. Term matching is performed by a series of character cells each containing one character of the query. A string of character cells is implemented on the same LSI chip and the

chips can be connected in series for longer strings. When a word or phrase of the query is matched, a signal is sent to the resolution sub-process on the LSI chip. The resolution chip is responsible for resolving the Boolean logic between terms and proximity requirements. If the item satisfies the query, the information is transmitted to the users computer.

The text array processor uses these chips in a matrix arrangement as shown in Figure9.10. Each row of the matrix is a query processor in which the first chip performsthe query resolution while the remaining chips match query terms. The maximum number of characters in a query is restricted by the length of a row while the number of rows limit the number of simultaneous queries that can be processed.

Another approach for hardware searchers is to augment disc storage. The*augmentation is a generalized associative search* element placed between the read and write heads on the disk. The content addressable segment sequential memory (CASSM) system uses these search elements in parallel to obtain structured data from a database. The CASSM system was developed at the University of Florida as a general purpose search device. It can be used to perform string searching across the database. Another special search machine is the ***relational associative processor (RAP)*** developed at the University of Toronto. Like CASSM performs search across a secondary storage device using a series of cells comparing data in parallel.

The ***Fast Data Finder (FDF)*** is the most recent specialized hardware text search unit still in use in many organizations. It was developed to search text and has been used to search English and foreign languages. The early Fast Data Finders consisted of an array of programmable text processing cells connected in series forming a pipeline hardware search processor. The cells are implemented using a VSLI chip. In the TREC tests each chip contained 24processor cells with a typical system containing 3600 cells. Each cell will be a comparator for a single character limiting the total number of characters in a query to the number of cells.

The cells are interconnected with an 8-bit data path and approximately 20- bit control path. The text to be searched passes through each cell in a pipeline fashion until the complete database has been searched. As data is analyzed at each cell, the 20 control lines states are modified depending upon their current state and the results from the comparator. An example of a Fast Data Finder system is shown inFigure 9.11.

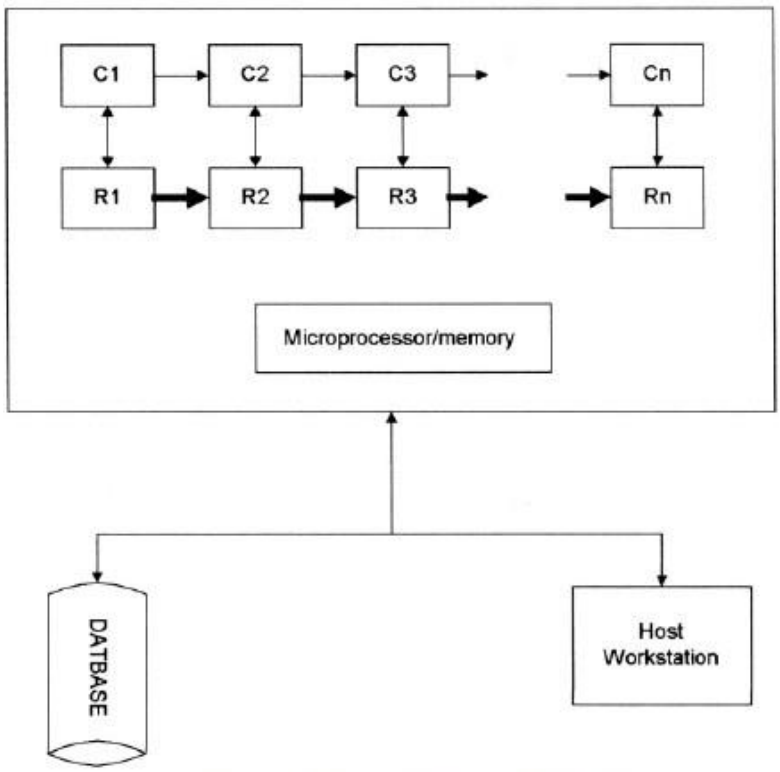


Figure 9.11 Fast Data Finder Architecture

A cell is composed of both a register cell (Rs) and a comparator (Cs).The input from the Document database is controlled and buffered by the micro process/memory and feed through the comparators. The search characters are stored in the registers. The connection between the registers reflect the control lines that are also passing state information.Groups of cells are used to detect query terms, along with logic between the terms, by appropriate programming of the control lines. When a pattern match is detected, a hit is passed to the internal microprocessor that passes it back to the host processor, allowing immediate access by the user to the Hit item.

The functions supported by the Fast data Finder are:

- Boolean Logic including negation
- Proximity on an arbitrary pattern
- Variable length "don't cares"
- Term counting and thresholds
- Fuzzy matching
- Term weights
- Numeric ranges

Information System Evaluation

The creation of the annual Text Retrieval Evaluation Conference (TREC) sponsored by the Defense Advanced Research Projects Agency (DARPA) and the National Institute of Standards and Technology (NIST) changed the standard process of evaluating information systems. The conference provides a standard database consisting of gigabytes of test data, search statements and the expected results from the searches to academic researchers and commercial companies for testing of their systems. This has placed a standard baseline into comparisons of algorithms.

In recent years the evaluation of Information Retrieval Systems and techniques for indexing, sorting, searching and retrieving information have become increasingly important.

There are many reasons to evaluate the effectiveness of an Information Retrieval System:

- To aid in the selection of a system to procure
- To monitor and evaluate system effectiveness
- To evaluate query generation process for improvements
- To provide inputs to cost-benefit analysis of an information system

- To determine the effects of changes made to an existing information system.

Measures Used in System Evaluations

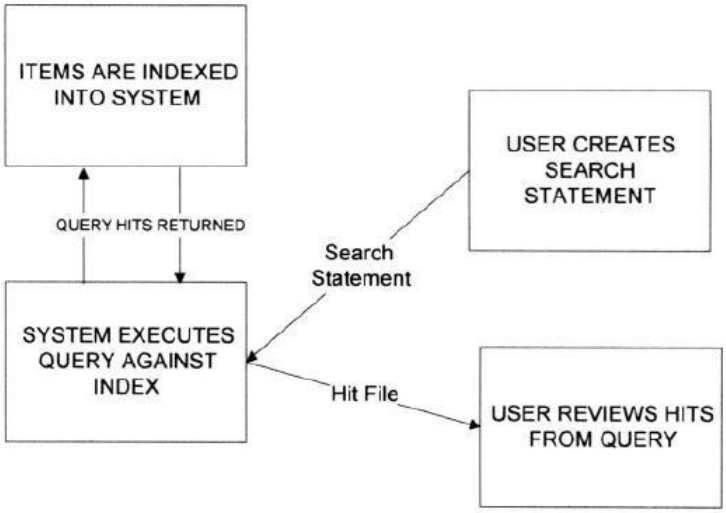


Figure 11.1 Identifying Relevant Items

Measurements can be made from two perspectives: user perspective and system perspective. Techniques for collecting measurements can also be objective or subjective. An objective measure is one that is well-defined and based upon numeric values derived from the system operation. A subjective measure can produce a number, but is based upon an individual users judgments.

Measurements with automatic indexing of items arriving at a system are derived from standard performance monitoring associated with any program in a computer (e.g., resources used such as memory and processing cycles) and time to process an item from arrival to availability to a search process. When manual indexing is required, the measures are then associated with the indexing process.

Response time is a metric frequently collected to determine the efficiency of the search execution. Response time is defined as the time it takes to execute the search. In addition to efficiency of the search process, the quality of the search results are also measured by precision and recall.

$$\text{Precision} = \frac{Number\_Retrieved\_Relevant}{Number\_Total\_Retrieved}$$

$$\text{Recall} = \frac{Number\_Retrieved\_Relevant}{Number\_Possible\_Relevant}$$

$$\text{Fallout} = \frac{Number\_Retrieved\_Nonrelevant}{Number\_Total\_Nonrelevant}$$

where *Number_Total_Nonrelevant* is the total number of non-relevant items in the database. Fallout can be viewed as the inverse of recall and will never encounter the situation of 0/0 unless all the items in the database are relevant to the search. It can

Another measure that is directly related to retrieving non-relevant items can be used in defining how effective an information system is operating. This measure is called Fallout and defined as:

There are other measures of search capabilities that have been proposed. A new measure that provides additional insight in comparing systems or algorithms is the "Unique Relevance Recall" (URR) metric. URR is used to compare more two or more algorithms or systems. It measures the number of relevant items that are retrieved by one algorithm that are not retrieved by the others:

$$\text{Unique\_Relevance\_Recall} = \frac{Number\_unique\_relevant}{Number\_relevant}$$

Number_unique_relevant is the number of relevant items retrieved that were not retrieved by other algorithms. When many algorithms are being compared, the definition of *uniquely* found items for a particular system can be modified, allowing a small number of other systems to also find the same item and still be considered unique. This is accomplished by defining a percentage **(P_u)** of the total number of systems that can find an item and still consider it unique. *Number_relevant* can take on two different values based upon the objective of the evaluation:

VALUE	INTERPRETATION
Total Number Retrieved Relevant (TNRR)	the total number of relevant items found by all algorithms
Total Unique Relevant Retrieved (TURR)	the total number of unique items found by all the algorithms

A	B	C	D	E	F	G	H	I	J	K	L	M
3	4	2	22	1	100	200	22	100	10	500	6	15

Figure 11.2a Number Relevant Items

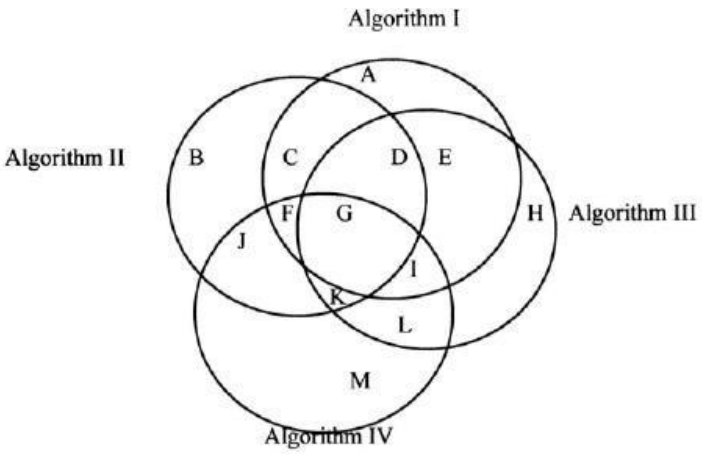


Figure 11.2b Four Algorithms With Overlap of Relevant Retrieved

actually found by the algorithm. Figure 11.2a and 11.2b provide an example of the overlap of relevant items assuming there are four different algorithms. Figure 11.2a gives the number of items in each area of the overlap diagram in Figure 11.2b. If a relevant item is found by only one or two techniques as a "unique item," then from the diagram the following values URR values can be produced:

Algorithm I - 6 unique items (areas A, C, E)
Algorithm II - 16 unique items (areas B, C, J)
Algorithm III - 29 unique items (areas E, H, L)
Algorithm IV - 31 unique items (areas J, L, M)

TNRR = A + B + C + • • • + M = 985
TURR = A + B + C + E + H + J + L + M = 61

Algorithm	URR_{TNRR}	URR_{TURR}
Algorithm I	6/985 = .0061	6/61= .098
Algorithm II	16/985 = .0162	16/61= .262
Algorithm III	29/985 = .0294	29/61 = .475
Algorithm IV	31/985 = .0315	31/61 = .508

Other measures have been proposed for judging the results of searches:

Novelty Ratio: ratio of relevant and not known to the user to total relevant retrieved

Coverage Ratio: ratio of relevant items retrieved to total relevant by the user before the search

Sought Recall: ratio of the total relevant reviewed by the user after the search to the total relevant the user would have liked to examine

In some systems, programs filter text streams, software categorizes data or intelligent agents alert users if important items are found. In these systems, the Information Retrieval System makes decisions without any human input and their decisions are binary in nature (an item is acted upon or ignored). These systems are called binary classification systems for which effectiveness measurements are created to determine how algorithms are working (Lewis-95). One measure is the utility measure that can be defined as (Cooper-73):

$$U = \alpha*(\text{Relevant\_Retrieved}) + \beta*(\text{Non-Relevant\_Not Retrieved}) - \delta*(\text{Non-Relevant\_Retrieved}) - \gamma*(\text{Relevant\_Not Retrieved})$$

where α and β are positive weighting factors the user places on retrieving relevant items and not retrieving non-relevant items while δ and γ are factors associated with the negative weight of not retrieving relevant items or retrieving non-relevant items. This formula can be simplified to account only for retrieved items with β and γ equal to zero (Lewis-96). Another family of effectiveness measures called the E-measure that combines recall and precision into a single score was proposed by Van Rijsbergen (Rijsbergen-79).

Measurement Example-TREC-Results

Until the creation of the Text Retrieval Conferences (TREC) by the Defense Advance Research Projects Agency (DARPA) and the National Institute of Standards and Technology (NIST), experimentation in the area of information retrieval was constrained by the researcher's ability to manually create a test database. One of the first test databases was associated with the Cranfield I and II tests (Cleverdon-62, Cleverdon-66). It contained 1400 documents and 225 queries.

It became one of the standard test sets and has been used by a large number of researchers. Other test collections have been created by Fox and Sparck Jones. There have been five TREC-conferences since 1992. TREC- provides a set of training documents and a set of test documents, each over 1 Gigabyte in size. It also provides a set of training search topics (along with relevance judgments from the database) and a set of test topics.

The researchers send to the TREC-sponsor the list of the top 200 items in ranked order that satisfy the search statements. These lists are used in determining the items to be manually reviewed for relevance and for calculating the results from each system. The search topics are "user need" statements rather than specific queries. This allows maximum flexibility for each researcher to translate the search statement to a query appropriate for their system and assists in the determination of whether an item is relevant.

The search Topics in the initial TREC-consisted of a Number, Domain (e.g., Science and Technology), Title, Description of what constituted a relevant item, Narrative natural language text for the search, and Concepts which were specific search terms.

In addition to the search measurements, other standard information on system performance such as system timing, storage, and specific descriptions on the tests are collected on each system. This data is useful because the TREC- objective is to support the migration of techniques developed in a research environment into operational systems. TREC-5 was held in November 1996. The results from each conference have varied based upon understanding from previous conferences and new objectives.

TREC-1 (1992) was constrained by researchers trying to get their systems to work with the very large test databases. TREC-2 in August 1993 was the first real test of the algorithms which provided insights for the researchers into areas in which their systems needed work. The search statements (user need statements) were very large and complex. They reflect long-standing information needs versus adhoc requests. By TREC-3, the participants were experimenting with techniques for query expansion and the importance of constraining searches to passages within items versus the total item.

TREC-4 introduced significantly shorter queries (average reduction from 119 terms in TREC-3 to 16 terms in TREC-4) and introduced five new areas of testing called “tracks” (Harman-96). The queries were shortened by dropping the title and a narrative field, which provided additional description of a relevant item. The multilingual track expanded TREC-4 to test a search in a Spanish test set of 200 Mbytes of articles from the “El Norte” newspaper.