Unit III

3 Evaluation and Visualization of Information Retrieval System

Syllabus

Performance evaluation : *Precision and recall, MRR, F-Score, NDCG, user-oriented measures.*
Visualization in Information System : *Starting points, Query Specification, document context, User relevance judgment, Interface support for search process.*

Contents

3.1 Performance Evaluation

SPPU : June-19, March-19, 20

- Performance evaluation of data retrieval system : The shorter the response tim the smaller the space used, the better the system is. It is a tradeoff between tim and space.
- Information Retrieval performance evaluation : Relevance of retrieved documen is important, besides time and space.
- In IR, since the user's query is inherently vague, the retrieved documents are n exact answers and have to be ranked according to their relevance to the query.
- Retrieval task could consist simply of a query processed in **batch mode** an **interactive mode.**

Batch mode (laboratory experiments)

1. The user submits a query and receives an answer back.
2. Measure : The quality of the generated answer set.
3. Still the dominant evaluation.
4. Main reasons : Repeatability and scalability.

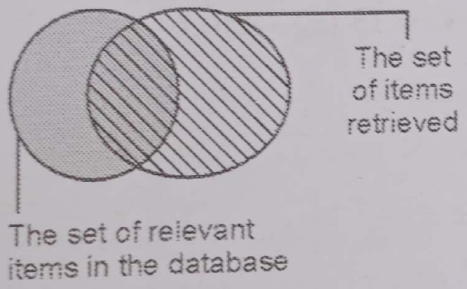
Interactive mode (real life situations)

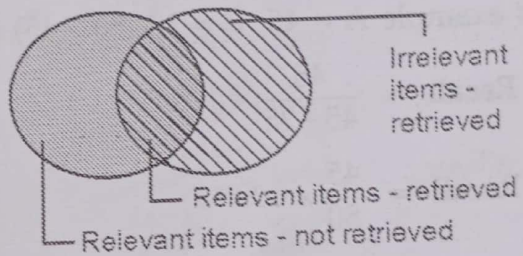
1. The user specifies his information need through a series of interactive steps with the system.
2. Measure : User effort, interface design, system's guidance, session duration.
3. Get a lot more attention since 1990.

- To measure ad hoc information retrieval effectiveness in the standard way, we need a test collection consisting of three things :
  1. A document collection.
  2. A test suite of information needs, expressible as queries.
  3. A set of relevance judgements, standard a binary assessment of either relevant or non-relevant for each query-document pair.
- The standard approach to information retrieval system evaluation revolves around the notion of relevant and non-relevant documents. With respect to a user information need, a document in the test collection is given a binary classification as either relevant or non-relevant. This decision is referred to as the gold standard or ground truth judgement of relevance.

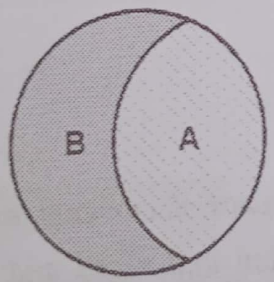
3.1.1 Precision and Recall

- **Relevance :** Relevance is a subjective notion. Different users may differ about the relevance or non-relevance of particular documents to given questions.
- In response to a query, an IR system searches its document collection and returns a ordered list of responses. It is called the retrieved set or ranked list. The system employs a search strategy or algorithm and measure the quality of a ranked list.
- A better search strategy yields a better ranked list and better ranked lists help the user fill their information need.
- Precision and recall are the basic measures used in evaluating search strategies. As shown in the first two figures, these measures assume :
  1. There is a set of records in the database which is relevant to the search topic
  2. Records are assumed to be either relevant or irrelevant.
  3. The actual retrieval set may not perfectly match the set of relevant records.





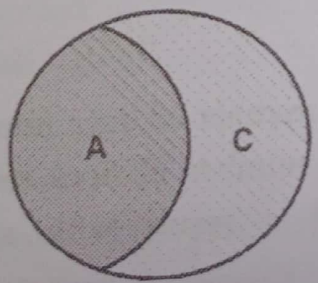
- **Recall** is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. It is usually expressed as a percentage.



A = Number of relevant records retrieved.
B = Number of relevant records not retrieved.

$$\text{Recall} = \frac{A}{A+B} \times 100\ \%$$

Precision is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. It is usually expressed as a percentage.



A = Number of relevant records retrieved.
C = Number of irrelevant records retrieved.

$$\text{Precison} = \frac{A}{A+C} \times 100\ \%$$

- As recall increases, the precision decreases and recall decreases the precisio increases.

Example 3.1.1 *Assume the following :*

A database contains 80 records on a particular topic.

A search was conducted on that topic and 60 records were retrieved.

Of the 60 records retrieved, 45 were relevant.

Calculate the precision and recall scores for the search.

Solution : Using the designations above :

A = The number of relevant records retrieved,

B = The number of relevant records not retrieved and

C = The number of irrelevant records retrieved.

In this example A = 45, B = 35 (80 - 45) and C = 15 (60 - 45).

$$\text{Recall} = \frac{45}{45+35} \times 100\ \%$$

$$\text{Recall} = \frac{45}{80} \times 100\ \%$$

Recall = 56.25 %

$$\text{Precision} = \frac{A}{A+C} \times 100\ \%$$

$$\text{Precision} = \frac{45}{45+15} \times 100\ \% = \frac{45}{60} \times 100$$

Precision = 75 %

Example 3.1.2 *20 found documents, 18 relevant, 3 relevant documents are not found, 27 irrelevant are as well not found. Calculate the precision and recall and fallout scores for the search.*

Solution :

Precision : 18/20 = 90 %

Recall : 18/21 = 85.7 %

Fall-out : 2/29 = 6.9 %

- Recall is a non-decreasing function of the number of docs retrieved. In a good system, precision decreases as either the number of docs retrieved or recall increases. This is not a theorem, but a result with strong empirical confirmation.

- The set of ordered pairs makes up the precision-recall graph. Geometrically when the points have been joined up in some way they make up the precision-recall curve. The performance of each request is usually given by a precision-recall curve. To measure the overall performance of a system, the set of curves, one for each request, is combined in some way to produce an average curve.
- Assume that set R_q containing the relevant document for q has been defined. Without loss of generality, assume further that the set R_q is composed of the following documents :

$$R_q = \{ d_3, d_5, d_9, d_{25}, d_{39}, d_{44}, d_{56}, d_{71}, d_{89}, d_{123} \}$$

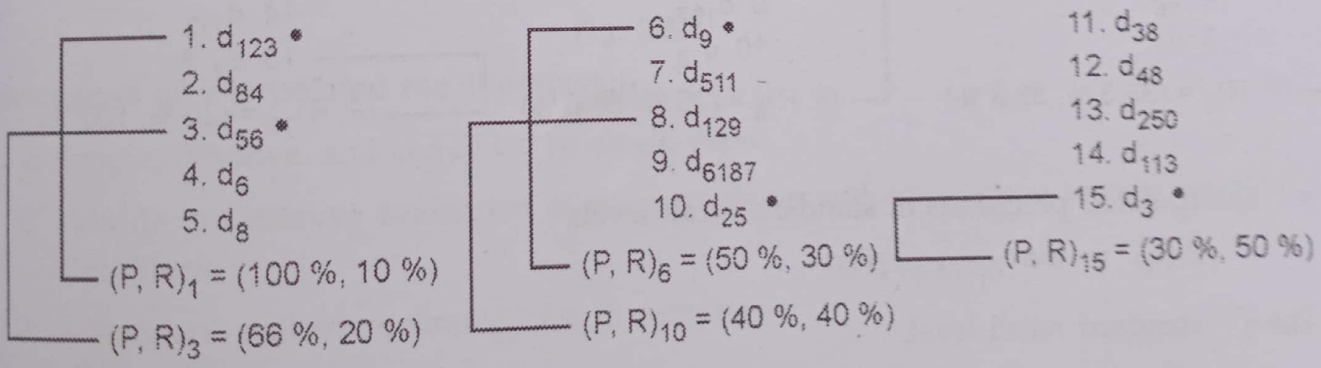
There are ten documents which are relevant to the query q.

- For the query q, a ranking of the documents in the answer set as follows.

Ranking for query q :

1. d_{123} *	6. d_9 *	11. d_{38}
2. d_{84}	7. d_{511}	12. d_{48}
3. d_{56} *	8. d_{129}	13. d_{250}
4. d_6	9. d_{187}	14. d_{113}
5. d_8	10. d_{25} *	15. d_3 *

- The documents that are relevant to the query q are marked with star after the document number. Ten relevant documents, five included in Top 15.



- Fig 3.1.1 shows the curve of precision versus recall. By taking various numbers of the top returned documents (levels of recall), the evaluator can produce a precision-recall curve.
- The precision versus recall curve is usually plotted based on 11 standard recall level: 0 %,10 %,....,100 %.
- In this example : The precisions for recall levels higher than 50 % drop to 0 because no relevant documents were retrieved. There was an interpolation for the recall level 0 %.

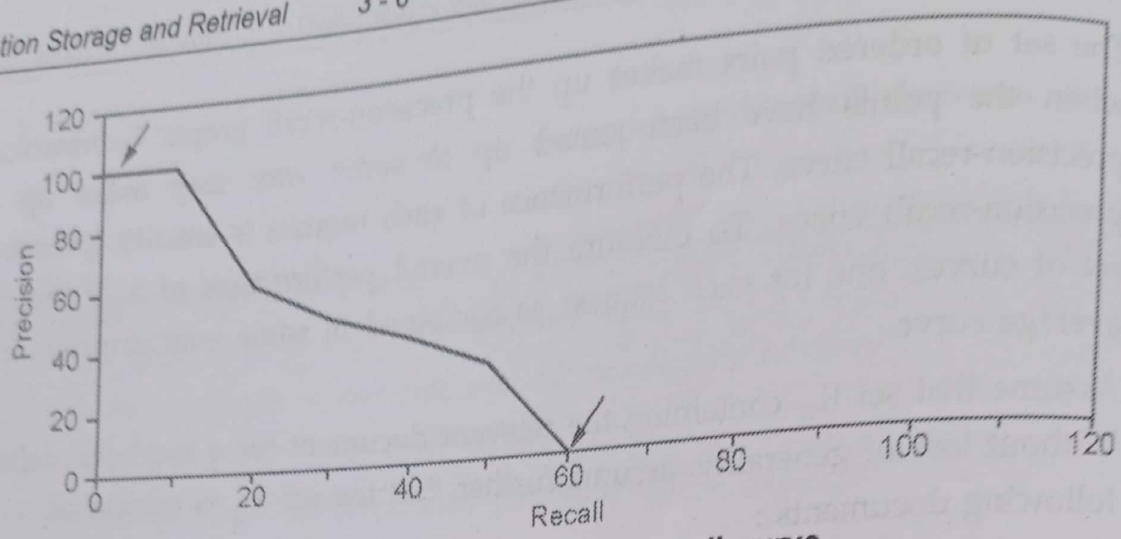


Fig. 3.1.1 Precision versus recall curve

- Since the recall levels for each query might be distinct from the 11 standard recall levels.

Interpolated recall-precision

- Idea : If locally precision increases with increasing recall, then you should get to count that. So you take the max of precisions to right of value.
- Consider again the set of 15 ranked documents. Assume that the set of relevant documents for the query q has changed and is now given by $R_q = \{d_3, d_{56}, d_{129}\}$
- The three relevant document :

1. d_{123}
2. d_{84}
3. d_{56} *
4. d_6
5. d_8

$(P, R)_3 = (33.3\ \%, 33.3\ \%)$

6. d_9
7. d_{511}
8. d_{129} *
9. d_{187}
10. d_{25}

$(P, R)_8 = (25\ \%, 66.6\ \%)$

11. d_{38}
12. d_{48}
13. d_{250}
14. d_{113}
15. d_3 *

$(P, R)_{15} = (20\ \%, 100\ \%)$

- Interpolated precisions at standard recall levels :

$$\overline{P}(r_j) = \max_{r_j \le r \le r_{j+1}} P(r)$$

The j^{th} standard recall level.

- Which means that the interpolated precision at the j^{th} standard recall level is the maximum known precision at any recall level between the j^{th} recall level and the $(j+1)^{th}$ recall level.
- At recall levels 0 %, 10 %, 20 % and 30 %, the interpolated precision is equal to 33.3 %.
- At recall levels 40 %, 50 %, 60 % the interpolated precision is equal to 25 %.

- At recall levels 70 %, 80 %, 90 % and 100 %, the interpolated precision is equal to 20 %.
- Fig. 3.1.2 shows the curve for interpolated precision and recall.

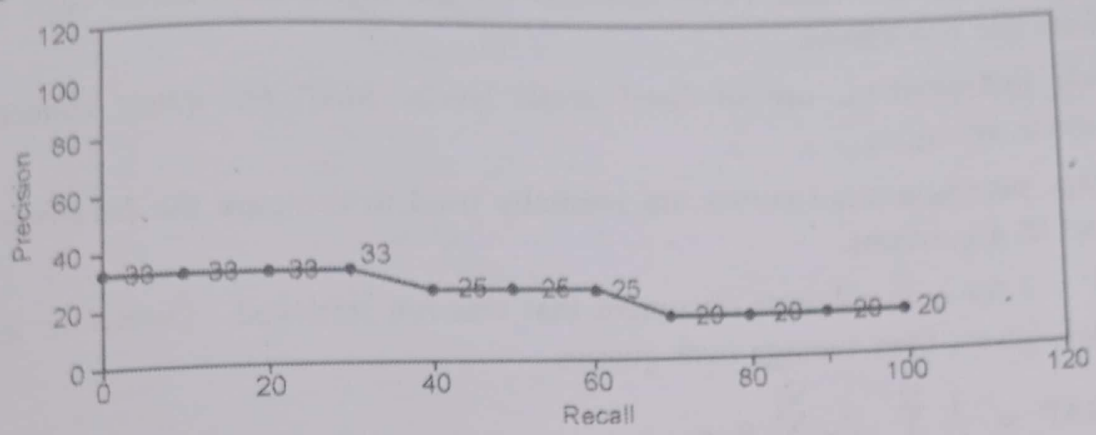


Fig. 3.1.2 Curve for interpolated precision and recall

- Following Fig. 3.1.3 shows the comparison between precision-recall curve and interpolated precision.

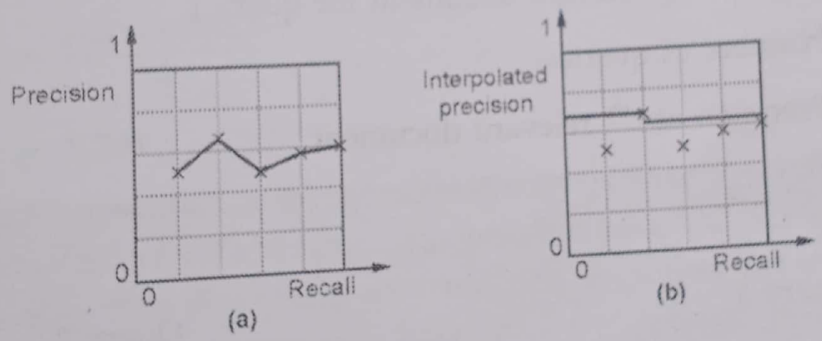


Fig. 3.1.3

Advantages of interpolated recall-precision

1. Simple, intuitive, and combined in single curve.
2. Provide quantitative evaluation of the answer set and comparison among retrieval algorithms.
3. A standard evaluation strategy for IR systems.

Disadvantages of interpolated recall-precision

1. Can not know true recall value except in small document collections.
2. Assume a strict document rank ordering.

- It is an experimental fact that average precision-recall graphs are monotonically decreasing. Consistent with this, a linear interpolation estimates the best possible performance between any two adjacent observed points. To avoid inflating the experimental results it is probably better to perform a more conservative interpolation.

Mean Average Precision (MAP)

- Also called average precision at seen relevant documents. It determine precision at each point when a new relevant document gets retrieved.
- Average of the precision value obtained for the top k documents, each time a relevant doc is retrieved.
- Avoids interpolation, use of fixed recall levels. MAP for query collection is arithmetic averaging.
- Average precision-recall curves are normally used to compare the performance of distinct IR algorithms.
- Use P = 0 for each relevant document that was not retrieved. Determine average for each query, then average over queries :

$$MAP = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{Q_j}\sum_{i=1}^{Q_j} P(doc_i)$$

where

Q_j = Number of relevant document for query j.

N = Number of queries.

$P(doc_i)$ = Precision at i^{th} relevant document

Example 3.1.3

Query 1		
Rank	**Relev.**	**$P(doc_i)$**
1	X	1.00
2		
3	X	0.67
4		
5		
6	X	0.50
7		
8		
9		
10	X	0.40

Query 2		
Rank	**Relev.**	**P(doc)**
1	X	1.00
2		
3	X	0.67
4		
5		
6		
7		
8		
9		
10		

11		
12		
13		
14		
15		
16		
17		
18		
19		
20	X	0.50
AVG :		0.564

11		
12		
13		
14		
15	X	0.2
AVG :		0.623

- *MAP favors systems which return relevant documents fast.*

Solution :

$$MAP = \frac{0.564 + 0.623}{2}$$

$$MAP = 0.594$$

- A necessary consequence of its monotonicity is that the average P-R curve will also be monotonically decreasing. It is possible to define the set of observed points in such a way that the interpolate function is not monotonically decreasing. In practice, even for this case, we have that the average precision-recall curve is monotonically decreasing.

Precision-recall appropriateness

- Precision and recall have been extensively used to evaluate the retrieval performance of IR algorithms. However, a more careful reflection reveals problems with these two measures :
- First, the proper estimation of maximum recall for a query requires detailed knowledge of all the documents in the collection.
- Second, in many situations the use of a single measure could be more appropriate.
- Third, recall and precision measure the effectiveness over a set of queries processed in batch mode.
- Fourth, for systems which require a weak ordering though, recall and precision might be inadequate.

Single value summaries

- Average precision-recall curves constitute standard evaluation metrics for information retrieval systems. However, there are situations in which we would like to evaluate retrieval performance over individual queries. The reasons are twofold :
  1. First, averaging precision over many queries might disguise important anomalies in the retrieval algorithms under study.
  2. Second, we might be interested in investigating whether a algorithm outperforms the other for each query.
- In these situations, a single precision value can be used.

R-Precision

- If we have a known set of relevant documents of size Rel, then calculate precision of the top Rel docs returned.
- Let R be the total number of relevant docs for a given query. The idea here is to compute the precision at the Rth position in the ranking.
- For the query q1, the R value is 10 and there are 4 relevant among the top 10 documents in the ranking. Thus, the R-Precision value for this query is 0.4.
- The R-precision measure is a useful for observing the behavior of an algorithm for individual queries. Additionally, one can also compute an average R-precision figure over a set of queries.
- However, using a single number to evaluate a algorithm over several queries might be quite imprecise.

Precision histograms

- The R-precision computed for several queries can be used to compare two algorithms as follows :
- Let,

  $RP_A(i)$: R-precision for algorithm A for the i-th query

  $RP_B(i)$: R-precision for algorithm B for the i-th query
- Define, for instance, the difference :

  $RP_{A/B}(i) = RP_A(i) - RP_B(i)$
- A positive value of $RP_{A/B}(i)$ indicates a better retrieval performance by algorithm A while a negative value indicates a better retrieval performance by algorithm B. Fig. 3.1.4 shows the $RP_{A/B}(i)$ values for two retrieval algorithms over 10 example queries.
- The algorithm A performs better for 8 of the queries, while the algorithm B performs better for the other 2 queries.

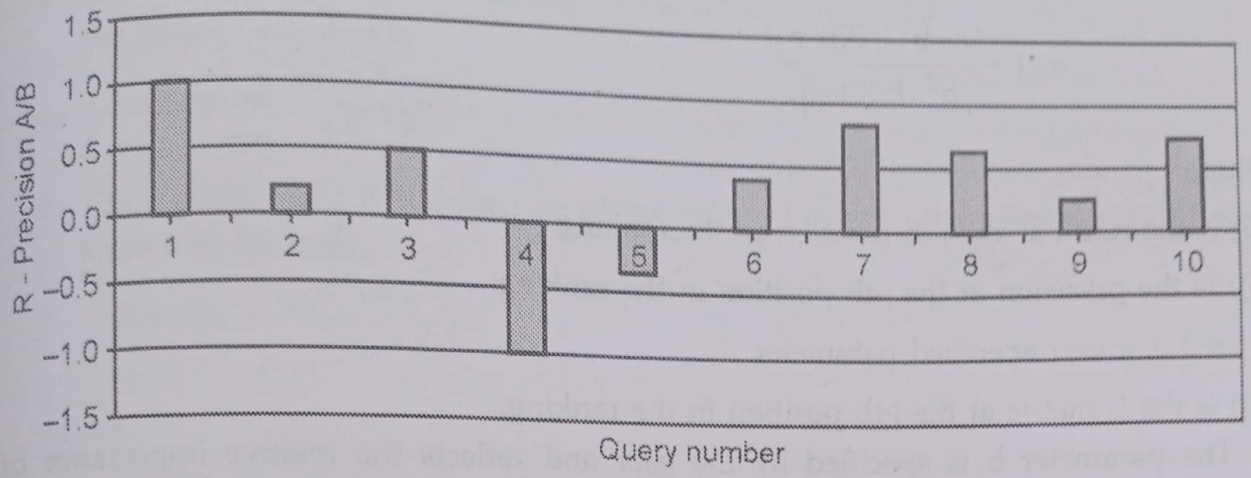


Fig. 3.1.4 Precision histograms

3.1.2 Alternative Measure

Method 1 : The harmonic mean (F Measure)

- The harmonic mean F of recall and precision is given by

$$F(j) = \frac{2}{\frac{1}{r(j)} + \frac{1}{P(j)}} = \frac{2 \cdot P(j) \cdot r(j)}{P(j) + r(j)}$$

where

r(j) : The recall for the j-th document in the ranking

P(j) : The precision for the j-th document in the ranking

Characteristics

- F = 0 : No relevant documents were retrieved.
- F = 1 : All ranked documents are relevant.
- A high F achieved only when both recall and precision are high.
- Determination of the maximal F can be interpreted as an attempt to find the best possible compromise between recall and precision.
- Harmonic mean emphasizes the importance of small values, whereas arithmetic mean is affected by large values.

Method 2 : The E measure

- A measure that combines recall and precision. The idea is to allow the user to specify whether he/she is more interested in recall or in precision.
- The E measure is defined as follows :

$$E(j) = 1 - \frac{1 + b^2}{\frac{b^2}{r(j)} + \frac{1}{P(j)}}$$

$$= 1 - \frac{(1+b^2)\cdot P(j)\cdot r(j)}{b^2 \cdot P(j) + r(j)}$$

where

r(j) is the recall at the j-th position in the ranking

P(j) is the precision at the j-th position in the ranking

b >= 0 is a user specified parameter

E(j) is the E metric at the j-th position in the ranking.

- The parameter b is specified by the user and reflects the relative importance of recall and precision.

Characteristics

- b = 1: Act as the complement of F measure
- b > 1: More interested in recall wrong statements
- b < 1: More interested in precision.

Method 3 : User-oriented measures

- Recall and precision assume that the set of relevant documents for a query is independent of the users. However, different users might have different relevance interpretations. To cope with this problem, user-oriented measures have been proposed.
- As before, consider a reference collection, an information request I and a retrieval algorithm to be evaluated with regard to I. Let R be the set of relevant documents and A be the set of answers retrieved.
- Fig. 3.1.5 shows the coverage ratio and novelty ratio for a given example information request.

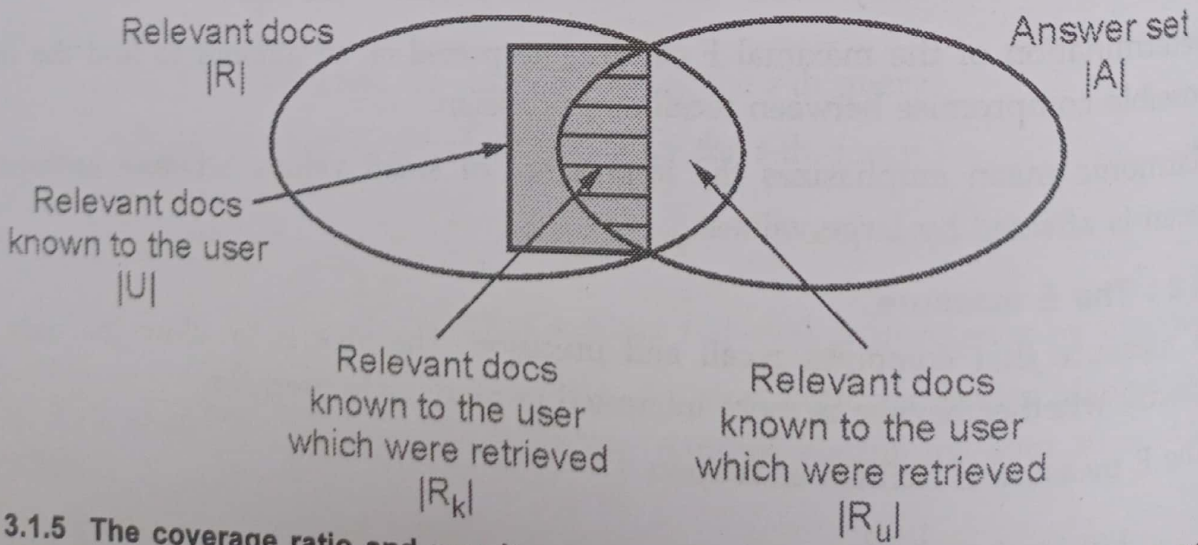


Fig. 3.1.5 The coverage ratio and novelty ratio for a given example information request

- The **coverage ratio** is the fraction of the documents known and relevant that are in the answer set, that is

$$\text{Coverage} = \frac{|K \cap R \cap A|}{|K \cap R|}$$

- The **novelty ratio** is the fraction of the relevant docs in the answer set that are not known to the user

$$\text{Novelty} = \frac{|(R \cap A) - K|}{|R \cap A|}$$

where

K : Set of documents known to the user

$|K \cap R \cap A|$: Set of relevant docs that have been retrieved and are known to the user

$|(R \cap A) - K|$: Set of relevant docs that have been retrieved but are not known to the user.

- A high coverage indicates that the system has found most of the relevant docs the user expected to see. A high novelty indicates that the system is revealing many new relevant docs which were unknown.
- Additionally, two other measures can be defined.
  1. Relative recall : Ratio between the number of relevant docs found and the number of relevant docs the user expected to find.

$$\frac{|R_k| + |R_u|}{|U|}$$

  2. Recall effort : Ratio between the number of relevant docs the user expected to find and the number of documents examined in an attempt to find the expected relevant documents.

$$\frac{|U|}{|A|}$$

University Questions

1. *List with definition different measures of association.* **SPPU : June-19, End Sem, Marks 4**
2. *Define and explain the following term : Precision and recall.* **SPPU : March-19, 20, In Sem, Marks 6**

2 Visualization in Information System

- Information visualization is the practice of representing data in a meaningful, visual way that users can interpret and easily comprehend. This includes data visualizations and dashboards.
- Information visualization is designed to assist users in understanding data.
- Information visualization is an art and therefore relies on the following aspects of design :
  a) The subject matter : Information or data being represented.
  b) The story : The concept being portrayed in the visualization.
  c) The goal.: Meeting the purpose with the right visualization.
  d) The visual : Using key elements of structure and design.

3.2.1 Starting Points

- Search interfaces must provide users with good ways to get started. An empty screen or a blank entry form does not provide clues to help a user decide how to start the search process. Users usually do not begin by creating a long, detailed expression of their information needs.
- Types of starting points are lists, overviews, examples and automated source selection.
  1. *Lists* : Frequent searches eventually learn a set of sources that are useful for their domains of interest, either through experience, formal training or recommendations from friends and others. Often used sources are stored on a favorites list also known as a bookmark list or hotlist on the web.
  2. Overviews : It can help users get started, directing them into general neighborhoods, after which they can navigate using more detailed descriptions.
  3. Example : Another way to help users get started is to start them off with an example of interaction with the system. This technique is also known as retrieval by reformulation.
  4. Automated source selection : Automatically selecting the best source for a query is to automatically send a query to multiple sources and then combine the result from the various systems in some way.

3.3 Query Specification

- To execute a query, users must select collections, metadata descriptions or information sets against which the query is to be matched and must specify words, phrases or other kinds of information that can be compared against the information in the collections.
- Shneiderman identifies five types of primary human computer interaction styles. These are command language, form filling, menu selection, direct mapping and natural language.

1. Command line interface : It provides a means of expressing instructions to the computer directly, using function keys, single characters, abbreviations or whole-word commands. In some systems it is the only way of communicating with the system, e.g. remote access using telnet.
2. Menus : The set of available options is displayed on the screen and selected using the mouse or numeric or alphabetic keys. These visible options rely on recognition rather than recall, but still need to be meaningful and logically grouped. Menus may be nested hierarchically, with the grouping and naming of menu options the only cue for finding the required option.
3. Natural language : Natural language is very difficult for a machine to understand. It is ambiguous, syntactically and semantically. It is difficult to provide the machine with context.
4. Question/answer, query dialogue : Question/answer dialogue is a simple mechanism for providing input to an application in a specific domain. The user is asked a series of questions and is led through the interaction step by step. Easy to learn and use, but limited in functionality and power.
5. Form-fills and spreadsheets : Used primarily for data entry but also useful in data retrieval. The display resembles a paper form, with slots to fill in. It may be based on an actual form with which the user is familiar.

3.3.1 Boolean Queries

- Many commercial full-text systems and bibliographic systems supported only Boolean queries. In information access systems, the matching process usually employs a statistical ranking algorithm.
- Boolean queries are queries using AND, OR and NOT to join query terms. It views each document as a set of terms.
- A query that matches documents matching boolean combinations of other queries.
- Boolean searches are named after the British born Mathematician George Boole.

Boolean logic establishes the relationship between keywords in a search. Boolean logic has three operators : AND, OR, NOT.

- A Boolean search strategy retrieves those documents which are 'true' for the query. This formulation only makes sense if the queries are expressed in terms of index terms or keywords and combined by the usual logical connectives AND, OR and NOT.
- For example, if the query Q = (K_1 AND K_2) OR (K_3 AND (NOT K_4)) then the Boolean search will retrieve all documents indexed by K_1 and K_2, as well as all documents indexed by K_3 which are not indexed by K_4.
- Problem with Boolean Queries
  a) Basic syntax is counter intuitive.
  b) Most query languages that incorporate Boolean operators also require the user to specify complex syntax for other kinds of connectors and for descriptive metadata.
  c) Boolean systems do not rank the retrieved documents according to their degree of match to the query.

3.3.2 Faceted Queries

- Faceted queries solve the problem of Boolean query by using command-line-based interfaces.
- Faceted search uses product or content features as criteria for a website visitor to refine their search results. User will get specific and relevant options to filter their result page. This makes faceted search an easy and practical way to search for products or pages.
- In DIALOG, each query produces a resulting set of documents that is assigned an identifying name. It shows the set number with a listing of the number and issuing a command to show the titles.
- Document sets that are not empty can be referred to by a set name and combined with AND operations to produce new sets. If this set in turn is too small, the user can back up and try a different combination of sets and this process is repeated in pursuit of producing a reasonably sized document set. This kind of query formulation is called a faceted query.

3.3.3 Graphical Approaches to Query Specification

- Direct manipulation is an interaction style in which the objects of interest in the UI are visible and can be acted upon via physical, reversible, incremental actions that receive immediate feedback.

- Here users act on displayed objects of interest using physical, incremental, reversible actions whose effects are immediately visible on the screen.
- The term direct manipulation was introduced by Ben Shneiderman in his keynote address at the NYU Symposium on User Interfaces
- Features of a direct manipulation interface :
  1. Visibility of the objects of interest.
  2. Incremental action at the interface with rapid feedback on all actions.
  3. Reversibility of all actions, so that users are encouraged to explore without severe penalties.
  4. Syntactic correctness of all actions, so that every user action is a legal operation.
  5. Replacement of complex command languages with actions to manipulate directly the visible objects.
- Direct manipulation examples :
  a) Drive a car
  b) If you want to turn left, what do you do ?
  c) What type of feedback do you get ?
  d) How does this help ?
  e) Think about turning left using a menu/text interface.
- Full-syntax Boolean query specification is not sufficiently usable for most searchers and thus is not widely used. Alternative way is to use Venn diagrams to improve Boolean query specification. Typically, a query term is associated with a circle or ring and intersection of rings indicates conjunction of terms. The number of documents that satisfy the various conjuncts are displayed within the appropriate segments of the diagram.
- In VQuery, a direct manipulation interface allowed users to assign any number of query terms to ovals. If two or more ovals were placed such that they overlap with one another and if the user selects the area of their intersection, an AND was implied among those terms.
- Fig. 3.3.1 (See Fig. 3.3.1 on next page) shows Venn diagram using VQuery.

3.3.4 Natural Language and Free Text Queries

- Statistical query algorithms have the advantages of allowing users to specify queries naturally, without having to think about Boolean or other operators.
- Drawback of statistical query : Gives the user less feedback about and control over the result.

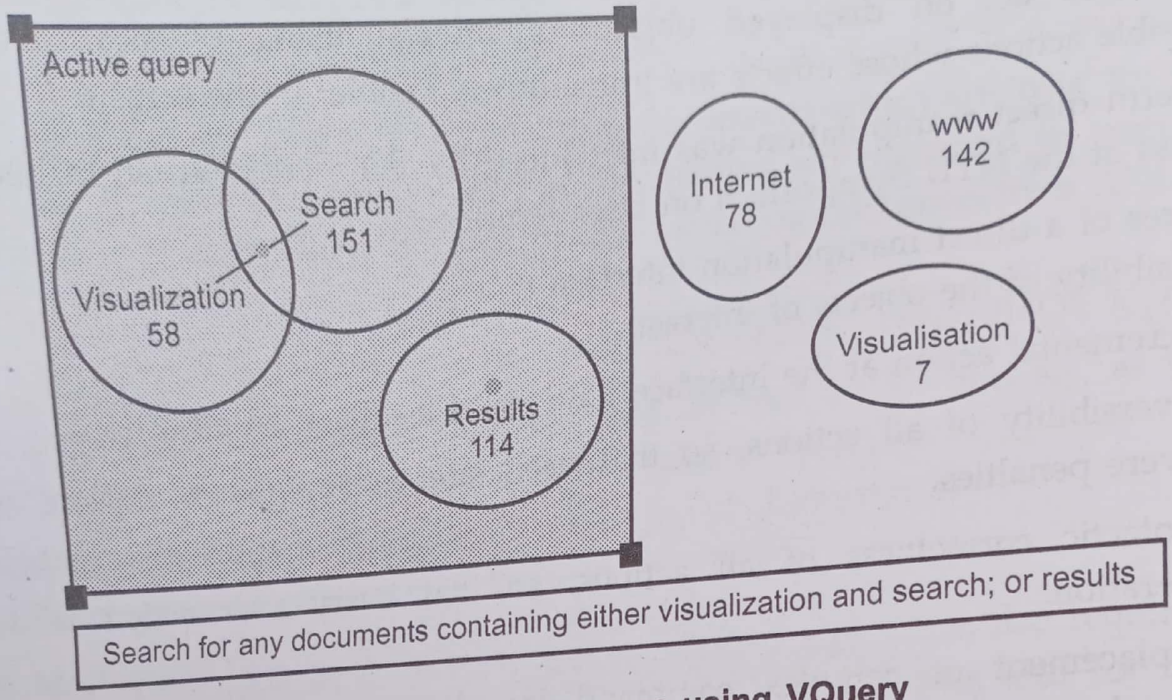


Fig. 3.3.1 Venn diagram using VQuery

- Natural language syntax of a question can be used to attempt to answer the question. The Murax system determines from the syntax of a question if the user is asking for a person, place or date.

3.4 Document Context

3.4.1 Document Surrogates

- A document surrogate is a limited representation of full documents.
- Surrogates example :
  a) Document identifiers : ISBN number of book.
  b) Titles and names : Author name or publisher name.
  c) Keyword/phrases : Introduction, summary, review.
- Some systems provide users with a choice between a short and detailed view Detailed view contains a summary or abstract.
- Use of surrogates has the risk of making firm decisions based on incomplet information.

3.4.2 Query Term Hits within Document Content

- In systems in which the user can view the full text of a retrieved document, it i often useful to highlight the occurrences of the terms or descriptors that matc those of the user's query.

1. KWIC

- The keyword-in-context (KWIC) feature allows users to search for any number of terms relevant to the analysis (the "keywords") and view them in a tabular overview along with the words that appear before and after (their respective contexts).
- The keywords are interactively linked to the original data, allowing you to jump directly to them in the original text.
- KWIC extract shows sentences that summarize the ways the query terms are used within the document. This display can show not only which subsets of query terms occur in the retrieved documents, but also the context they appear in with respect to one another.
- Fig. 3.4.1 shows KWIC.

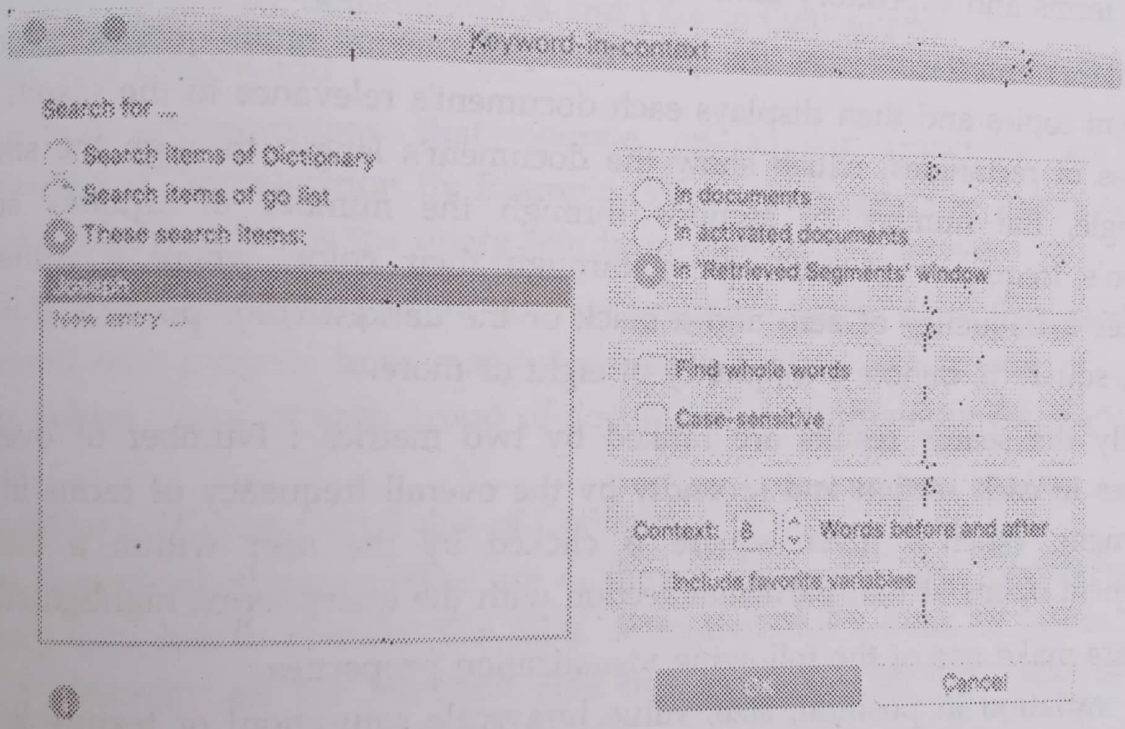


Fig. 3.4.1 KWIC

- In the left side window, select the keywords you would like to search for. Three options are available :
  1) Search items of the dictionary - All words in the currently selected dictionary will be searched for. The search options set in the dictionary are searched per keyword. The corresponding options in the dialog window will be ignored.
  2) Search items of go list - All words contained in the currently selected go list will be searched for.
  3) These search items - Enter the desired search terms in the list. The entries can contain spaces. Hit the enter key to move to the next line.

The following further options are available :

1) Find whole words - If the option is enabled, the search for "or" will not find, for example, "Inventor" or "Elevator". This option is available only if searching for words in the go list or if users enter specific search items.

2) Case sensitive - If this option is enabled, the search for "You" will not find the lowercase form "you". This option is available only if searching for words in the go list or if you enter specific search items.

3) Context x Words - Specify how many words to include before and after the keyword. The default setting is 5 words.

. TileBars

- TileBar is a data visualization interface used in conjunction with a search feature, query terms and expository text.
- The tileBar interface forces the user to input one or more queries, generally of different topics and then displays each document's relevance to the search through the use of rectangles, which show the document's length through the size of the rectangle, the number of sections through the number of squares and each section's frequency of each query through their color, where a white square denotes a frequency of zero and a black or the darkest color possible if not using black, square to denote a frequency of eight or more.
- Usually the search results are ranked by two metrics : Number of overlapping squares in each section and secondly by the overall frequency of terms in a given document. When a given square is clicked by the user within a tilebar, the document opens at that particular section with the query terms highlighted.
- TileBars make use of the following visualization properties :
  1. A variation in position, size, value [grayscale saturation] or texture is ordered [ordinal] that is, it imposes an order which is universal and immediately perceptible.
  2. If shading is used, make sure differences in shading line up with the values being represented. The lightest ("unfilled") regions represent "less", and darkest ("most filled") regions represent "more".
  3. Because they do have a natural visual hierarchy, varying shades of gray show varying quantities better than color.

3. SeeSoft

- The SeeSoft visualization represents text in a manner resembling columns of newspaper text, with one 'line' of text on each horizontal line of the strip. The representation is compact.

- Graphics are used to abstract away the details, providing an overview showing the amount and shape of the text. Color highlighting is used to pick out various attributes, such as where a particular word appears in the text.
- Details of a smaller portion of the display can be viewed via a pop-up window; the overview shows more of the text but in less detail.
- SeeSoft was originally designed for software development, in which a line of text is a meaningful unit of information.

3.4.3 Query Term Hits between Documents

1. InfoCrystal : InfoCrystal can be used as a visualization tool as well as a visual query language to help users search for information. The InfoCrystal shows how many documents contain each subset of query terms. This relieves the user from the need to specify Boolean ANDs and ORs in their query.
2. VIBE and Lyberworld :

- Graphical presentations that operate on similar principles are VIBE and Lyberworld. Visualization By Example (VIBE) plots results as points against query words which move as the query words are moved by the user.
- Query terms are placed in an abstract graphical space. After the search, icons are created that indicate how many documents contain each subset of query terms. The subset status of each group of documents is indicated by the placement of the icon.

3. SuperBook : Context via Table of Contents.

- The SuperBook system makes use of the structure of a large document to display query term hits in context. The table of contents for a book or manual are shown in a hierarchy on the left-hand side of the display and full text of a page or section is shown on the right-hand side.
- The user can manipulate the table of contents to expand or contract the view of sections and subsections. When the user moves the cursor to another part of the TOC, the display changes dynamically, making the new focus larger and shrinking down the previously observed sections.

3.4.4 Using Hyperlinks to Organize Retrieval Results

- A standard search engine retrieves web pages that fall within a widely diverse range of information contexts, but presents these results uniformly, in a ranked list. As an alternative, the Cha-Cha system organizes web search results in such a way as to react to the underlying structure of the intranet.

1. **Cha-Cha system**

- Cha-Cha demonstrates the application of the idea across a very large, heterogeneous web site that is an order of magnitude larger than those used by these other systems and is used operationally by thousands of users.
- A major problem with WebTOC, along with other attempts to provide categorical access to information (such as Yahoo), is that they do not couple navigation with ad hoc search.
- Fig. 3.4.2 shows outline view of the current implementation of Cha-Cha search on the query "earthquake".

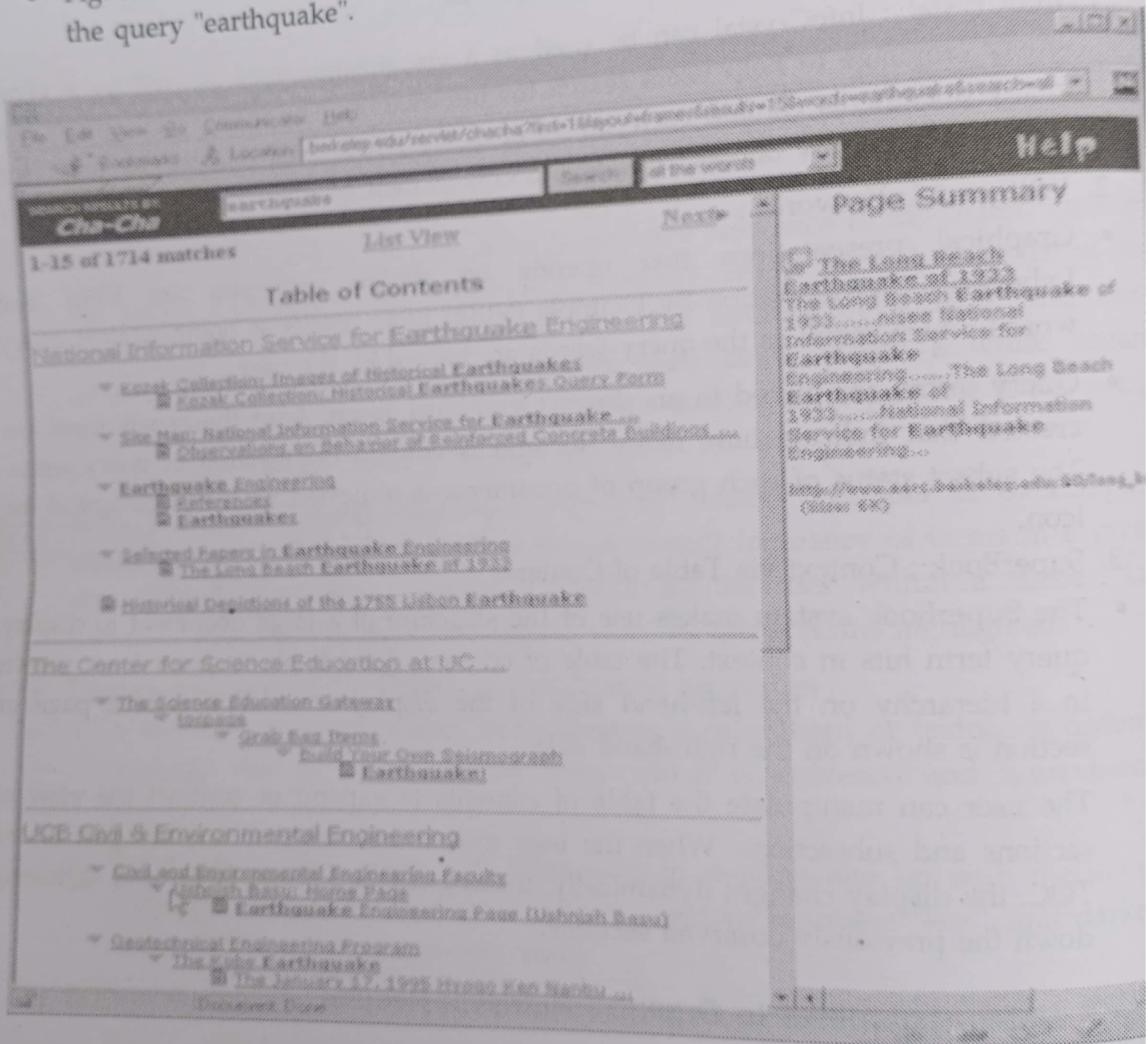


Fig. 3.4.2 Outline view of the current implementation of Cha-Cha search on the quer "earthquake"

Cha-Cha system

- Cha-Cha demonstrates the application of the idea across a very large, heterogeneous web site that is an order of magnitude larger than those used by these other systems and is used operationally by thousands of users.
- A major problem with WebTOC, along with other attempts to provide categorical access to information (such as Yahoo), is that they do not couple navigation with ad hoc search.
- Fig. 3.4.2 shows outline view of the current implementation of Cha-Cha search on the query "earthquake".

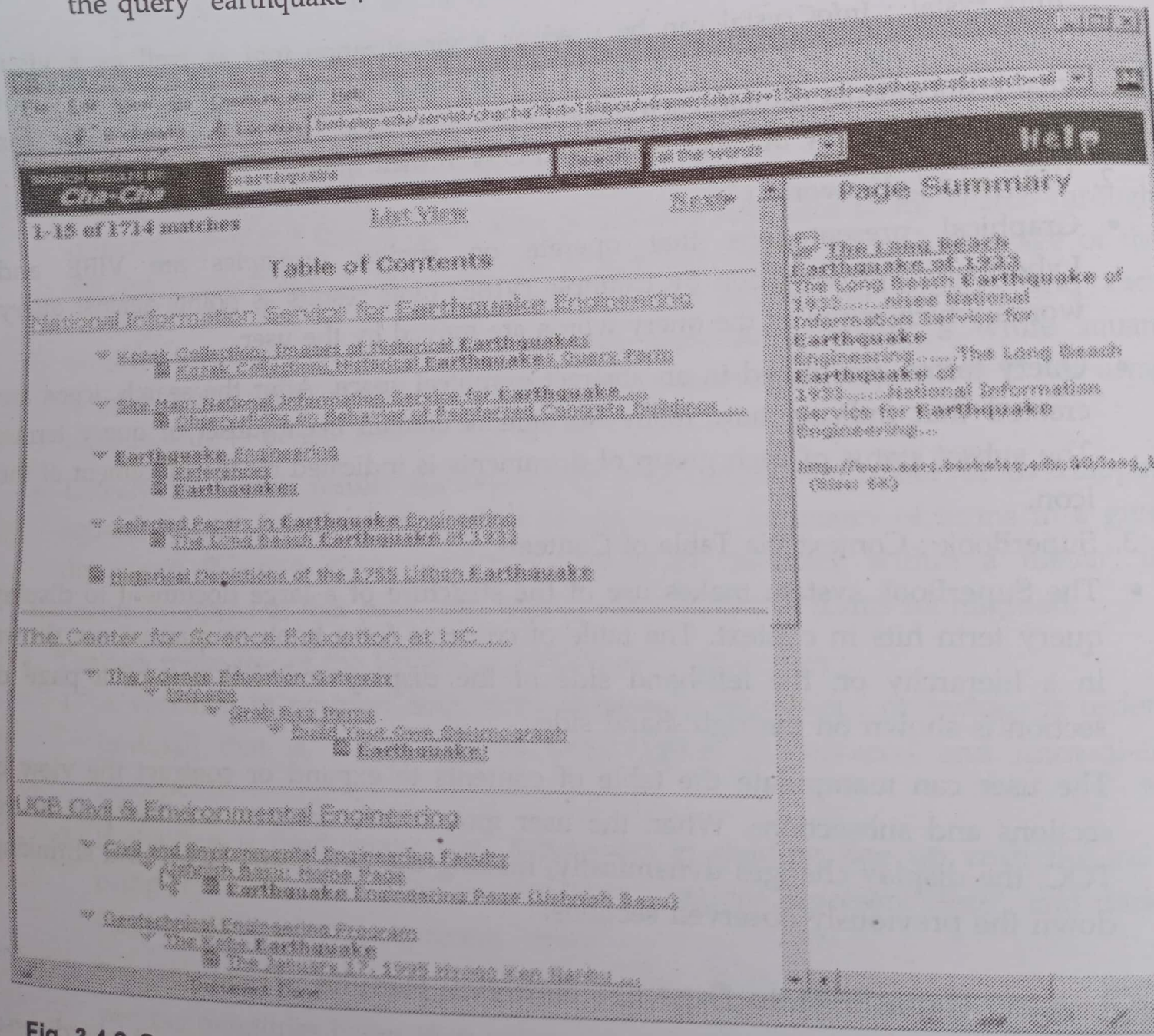


Fig. 3.4.2 Outline view of the current implementation of Cha-Cha search on the query "earthquake"

3.5 User Relevance Judgment

- Measuring relevance of documents with respect to a user's query is at the heart of Information Retrieval (IR), where the user's relevance judgment criteria have been recognized as multi-dimensional.
- An important part of the information access process is query reformulation and a proven effective technique for query reformulation is relevance feedback.
- Relevance feedback refers to an interaction cycle in which the user selects a small set of documents that appear to be relevant to the query and the system then uses features derived from these selected relevant documents to revise the original query. This revised query is then executed and a new set of documents is returned.
- Documents from the original set can appear in the new results list, although they are likely to appear in a different rank order. Relevance feedback in its original form has been shown to be an effective mechanism for improving retrieval results in a variety of studies and settings.
- A standard interface for relevance feedback consists of a list of titles with checkboxes beside the titles that allow the user to mark relevant documents. This can imply either that unmarked documents are not relevant or that no opinion has been made about unmarked documents, depending on the system.
- After the user has made a set of relevance judgments and issued a search command, the system can either automatically reweight the query and re-execute the search or generate a list of terms for the user to select from in order to augment the original query.
- An alternative approach, known either as **pseudo, blind or ad-hoc** Relevance Feedback (RF), employs RF techniques to automatically improve a ranking before any documents have been shown to the user.
- In this technique the system generates a document ranking from the initial query, selects a small number of documents from the top of the ranking, then initiates an iteration of RF by assuming these top-ranked documents are all relevant (the pseudo-relevant documents).
- The new query, generated by RF, is then used to produce a new document ranking which is shown to the user. The basis behind pseudo RF is that an iteration of feedback, based on the most similar documents to the user's initial query, will give a better initial ranking of documents.
- The pseudo RF technique then works well for 'good' initial queries - those that are good in retrieving relevant documents - and poorly for 'bad' initial queries - those that are bad at retrieving relevant documents.

- There are two possible solutions to this problem : Either improve the initial ranking, so that there is a greater likelihood of relevant documents being used to modify the query or improve the detection of relevant features, i.e. develop better RF techniques.

3.6 Interface Support for Search Process

- The user interface designer must make decisions about how to arrange various kinds of information on the computer screen and how to structure the possible sequences of interactions.
- Although the literature of information retrieval includes many studies of search interface design, many variables preclude the emergence of the right way to design search interfaces. Here are a few of the variables on the table :
  1. The level of searching expertise users have : Are they comfortable with Boolean operators or do they prefer natural language ? Do they need a simple or high-powered interface ? What about a help page ?
  2. The kind of information the user wants : Do they want just a taste or are they doing comprehensive research ? Should the results be brief or should they provide extensive detail for each document ?
  3. The type of information being searched : Is it made up of structured fields or full text ? Is it navigation pages, destination pages or both ? HTML or other formats ?
  4. How much information is being searched : Will users be overwhelmed by the number of documents retrieved ?

1. Window management :

- When arranging information within windows, the designer must choose between a monolithic display, in which all the windows are laid out in predefined positions and are all simultaneously viewable, tiled windows and overlapping windows.
- User studies have been conducted comparing these options when applied to various tasks. Usually the results of these studies depend on the domain in which the interface is used and no clear guidelines have yet emerged for information access interfaces.

3.6.1 Example Systems

1. The InfoGrid Layout

- The Information Grid (InfoGrid) is a framework for building information access applications that provides a user interface design and an interaction model.
- Fig. 3.6.1 shows InfoGrid layout. (See Fig. 3.6.1 on next page)

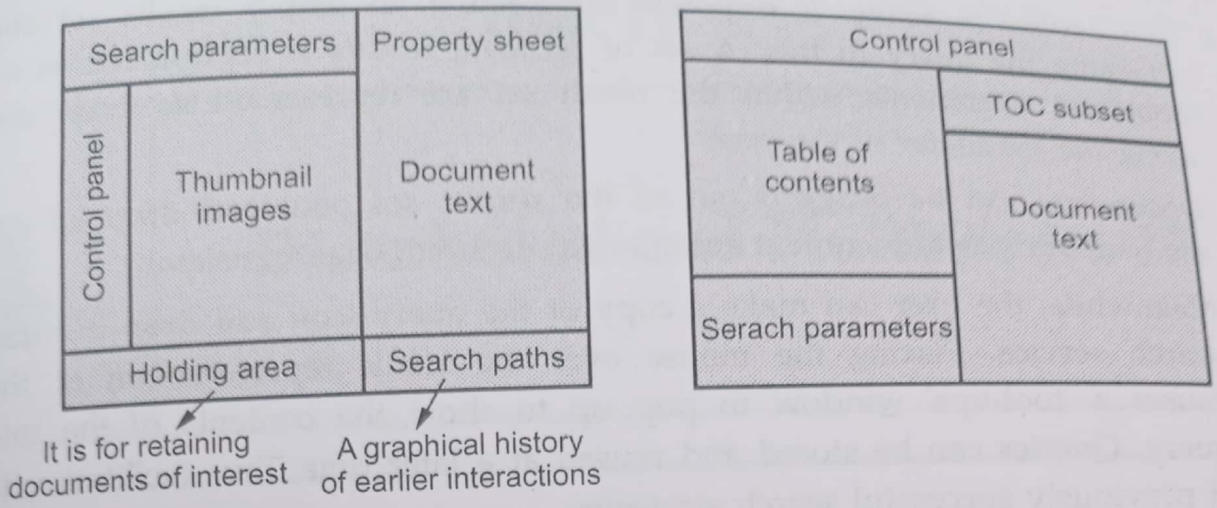


Fig. 3.6.1 InfoGrid layout

- InfoGrid system is a typical example of a monolithic layout for an information access interface. The layout assumes a large display is available and is divided into a left-hand and right-hand side. The left side of the screen is devoted to retrieving, selecting and operating on objects and the right side is devoted to browsing or editing a specific object.
- The main central area is used for the viewing of retrieval results, either as thumbnail representations of the original documents or derived organizations of the documents, such as Scatter/Gather-style cluster results.

2. The SuperBook Layout

- The layout of the InfoGrid is quite similar to that of SuperBook. The main difference is that SuperBook retains the table of contents-like display in the main left-hand pane, along with indicators of how many documents containing search hits occur in each level of the outline. Like InfoGrid, the main pane of the right-hand side is used to display selected documents.

3. The DLITE Interface

- The DLITE system makes a number of interesting design choices. It splits functionality into two parts : Control of the search process and display of results.
- The control portion is a graphical direct manipulation display with animation. Queries, sources, documents and groups of retrieved documents are represented as graphical objects. The user creates a query by filling out the editable fields within a query constructor object.
- The system manufactures a query object, which is represented by a small icon which can be dragged and dropped onto iconic representations of collections or search services.

- If a service is active, it responds by creating an empty results set object and attaching the query to this. A set of retrieval results is represented as a circular pool and documents within the result set are represented as icons distributed along the perimeter of the pool.
- Documents can be dragged out of the results set pool and dropped into other services, such as a document summarizer or a language translator.
- Meanwhile, the user can make a copy of the query icon and drop it onto another search service. Placing the mouse over the iconic representation of the query causes a 'tool-tips' window to pop up to show the contents of the underlying query. Queries can be stored and reused at a later time, thus facilitating retention of previously successful search strategies.

❑❑❑

Unit IV

4 Distributed and Multimedia IR

Syllabus

Distributed IR : *Introduction, Collection Partitioning, Source Selection, Query Processing,* ***Multimedia IR :*** *Introduction, Data Modeling, Query Language, Background-Spatial Access Method, A Generic Multimedia Indexing Approach, One Dimensional Time Series, Two-Dimensionalcolor Images, Automatic Feature Extraction, Trends and Research Issue.*

Contents

4.1 Distributed IR

Introduction

- A distributed computing system can be viewed as a MIMD parallel processor with relatively slow inter-processor communication channel and the freedom to employ a heterogeneous collection of processors in the system.
- Distributed model is very similar to the MIMD parallel processing model. The main difference here is that subtasks run on different computers and the communication between the subtasks is performed using network protocol such as TCP/IP.
- Distributed systems typically consist of a set of processes, each running on a separate processing node. A broker process is responsible for
  a) Accepting client requests,
  b) Distributing the requests to the servers,
  c) Collecting intermediate results from the servers, and
  d) Combining the intermediate results into a final result for the client.
- Distributed computing uses multiple computers connected by a network to solve a single problem. A distributed computing system can employ a heterogeneous collection of processors in the system. In fact, a single processing node in the distributed system could be a parallel computer in its own right.
- The cost of inter-processor communication is considerably higher in a distributed computing system. As such, distributed programs are usually coarse grained. Granularity refers to the amount of computation relative to the amount of communication performed by the program. The coarse grained programs perform large amounts of computation relative to the communication cost. An application may use different levels of granularity at different times to solve a given problem.

Algorithmic IR Issues

- How to distribute documents across the distributed search servers ?
  Collection partitioning
- How to select which servers should receive a particular search request ?
  Source selection
- How to combine the results from the different servers ?
  Merging the results
- A promising solution to distributed retrieval is meta-searching, which dispatches a user's query to multiple sources and gathers the results into a single result set. An important component of meta-searching is selecting the set of information sources most likely to provide relevant documents.

- The main challenges in implementing a successful metasearch engine can be categorized as :
  1. Information source selection : The process of determining which information sources are likely to contain relevant content.
  2. Query execution : The process of sending the user's query to each of the selected information source and
  3. Result merging : The process of aggregating the results from each of the selected source and presenting the final result to the user.

4.1.1 Collection Partitioning

Collection Partitioning in a Decentralized System :

- In a system comprising independently administered, heterogeneous search servers, the distributed document collections will be built and maintained independently. There is no central control of the document partitioning procedure. It may be that each search server is focused on a particular subject area.

Collection Partitioning in a Centralized System

- The collection can be replicated across all of the search servers. Appropriate when the collection is small enough to fit on a single search server, but high availability and query processing throughput are required.
- The parallelism in the system is being exploited via multitasking and the broker's job is to route queries to the search servers and balance the loads on the servers.
- When the distribute system is centrally administered, more options are available.
  1. The first option is simple replication of the collection across all of the search servers.
  2. The second option is random distribution of the documents.
  3. The final option is explicit semantic partitioning of the documents.
- **Indexing** the documents is handled in one of two ways.
  1. Each search server separately indexes its replica of the documents.
  2. Each server is assigned a mutually exclusive subset of documents to index and the index subsets are replicated across the search servers. A merge of the subsets is required at each server to create the final indexes.

Updates (in a centralized system)

- Document updates and deletions must be broadcast to all servers in the system.
- Document additions may be broadcast or they may be batched and partitioned depending on their frequency and how quickly updates must be reflected by the system.

The Second Option : Random Distribution of the Documents

- Appropriate when a large document collection must be distributed for performance reasons, but the documents will always be viewed and searched as if they are part of a single, logical collection. The broker broadcasts every query to all of the search servers and combines the results for the user.
- **The third option : Explicit semantic partitioning of the documents, which are either** already organized into semantically meaningful collections, such as by technical discipline or an automatic clustering or categorization procedure is used to partition the documents into subject-specific collections.

4.1.2 Source Selection

- Source selection is the process of determining which of the distributed document collections are most likely to contain relevant documents for the current query and therefore should receive the query for processing.
- Simple approach : Assume that every collection is equally likely to contain relevant document and always broadcast the query to all collections. This appropriate is used when documents are randomly partitioned or there is significant semantic overlap between the collections.
- The collections can also be ranked according to their likelihood of containing relevant documents. This is appropriate :
  a. If documents are partitioned into semantically meaningful collections or
  b. It is prohibitively expensive to search every collection every time
- The basic technique :
  a. Treat each collection as if it were a single large document
  b. Generate a collection vector for each collection
  c. Evaluate the query vector against each collection vector to produce a ranked listing of collections.
- If we apply a standard cosine similarity measure using a query vector and collection vectors. To calculate a term weight in the collection vector, using tf-idf style weighting, term frequency tf i,j is the total number of occurrences of term i in collection j and the inverse document frequency idfi for term i is log(N/ni) where N is the total number of collections and ni is the number of collections in which term i appears.
- A danger of this approach is that although a particular collection may receive a high query relevance score, there may not be individual documents within the collection that receive a high query relevance score. The problem can be avoided by indexing each collection as a series of blocks, where each block contains documents.

a) The query is evaluated against each block.

b) The score for a collection is computed from the scores of its blocks.

- Alternative approach to indexing collections : Training queries. A set of training queries are used to build a content model for each collection. When a new query is submitted to the system, its similarity to the training queries is computed and the content model is used to determine which collections should be searched and how many documents from each collection should be returned.

4.1.3 Query Processing

- Query processing in a distributed IR system :
  1. Select collections to search
  2. Distribute query to selected collections
  3. Evaluate query at distributed collections in parallel
  4. Combine results from distributed collections into final result.
- Step 1 may be eliminated if the query is always broadcast to every document collection in the system. Otherwise, one of the selection algorithms is used and the query is distributed to the selected collections.
- Each of the participating search servers then evaluates the query on the selected collections using its own local search algorithm. Finally, the results are merged.

Merging the Results

- There are a number of scenarios used for merging the result.
- If the query is Boolean and the search servers return Boolean result sets then the final result set equal to union of the result sets.
- If the query involves free-text ranking, a number of techniques are available ranging from simple to complex/accurate.
- Simplest approach: combine the ranked result lists using round robin interleaving

  $1 : 1^{st}$ document from the 1^{st} list,

  $2 : 1^{st}$ document from the 2^{nd} list,

  $N : 1^{st}$ document from the N^{th} list,

  N+1 : 2nd document from the 1^{st} list,...
- Likely to produce poor quality results, since hits from irrelevant collections are given status equal to that of hits from highly relevant collections.
- Improvement : Merge the result lists based on relevance score. Unless proper global term statistics are used to compute the document scores, we may get incorrect results. If documents are randomly distributed such that global term

statistics are consistent across all of the distributed collections, the merging based on relevance score is sufficient.

- If the document collections are semantically partitioned or maintained by independent parties, then reranking must be performed.
- **Reranking :** By weighting document scores based on their collection similarity computed during the source selection step. The weight for a collection can be computed as :

$$w = 1+|C|\cdot(s-\hat{s})/\hat{s}$$

where $|C|$ is the number of collections searched, s is the collection score and $\hat{s}$ is the mean of the collection scores.

- More accurate technique for merging ranked result lists is to use accurate global term statistics. If the collections have been indexed for source selection, that index will contain global term statistics across all of the distributed collections.
- The broker can include these statistics in the query when it distributes the query to the search servers. The servers can use these statistics in their processing and produce relevance scores that can be merged directly.
- If a collection index is unavailable, query distribution can proceed in two rounds of communication. In the first round, the broker distributes the query and gathers collection statistics from each server. These statistics are combined by the broker and distributed back to the servers in the second round.
- The search protocol can also require that the servers return global query term statistics and per-document query term statistics. The broker is then free to rerank every document using the query term statistics and a ranking algorithm of its choice.
- The end result is a list that contains documents from the distributed collections ranked in the same order as if all of the documents had been indexed in a single collection.

University Question

1. *Describe the architecture of distributed IR.* **SPPU : June-19, End Sem, Marks 9**

4.2 Multimedia IR

Multimedia data

- There are number of data types that can be characterized as multimedia data types. These are typically the elements for the building blocks of generalized multimedia environments. The basic types can be described as follows :

1. Text : The form in which the text can be stored can vary greatly. In addition to ASCII based files, text is typically stored in processor files, spreadsheets, databases.
2. Images : There is great variance in the quality and size of storage for still images. Digitalized images are sequence of pixels that represents a region in the user's graphical display. The space overhead for still images varies on the basis of resolution, size, complexity, and compression scheme used to store image. The popular image formats are jpg, png, bmp, tiff.
3. Audio : An increasingly popular data type being integrated in most of applications is Audio. Its quite space intensive.
4. Video : One on the most space consuming multimedia data type is digitalized video. The digitalized videos are stored as sequence of frames. Depending upon its resolution and size a single frame can consume upto 1 MB.
5. Graphic Objects : These consist of special data structures used to define 2D and 3D shapes through which we can define multimedia objects. These include various formats used by image, video editing applications. Examples are CAD / CAM objects.

- The information that can be perceived by the human senses is transported through different media, such as text or sound. Humans communicate with computers by means of various media, or use computers as tools for communication with each other. These observations led to the following definition of multimedia systems : ***"A multimedia system is characterized by the computer-controlled generation, manipulation, presentation, storage, and communication of independent discrete and continuous media."***
- Multimedia data is large and affects the storage, retrieval and transmission of multimedia data. For this reasons, the development of multimedia system is considerably more complex than a traditional information system. Conventional systems only deal with simple data types, such as strings or integers.
- Since information can be recorded on various media types, such as tables, images, text, audio and video, the retrieval system must be able to retrieve information from varying media representations, giving rise to the concept of Multimedia Information Retrieval Systems.

4.2.1 Comparison between Multimedia Information System and Traditional System

- Multimedia data is unstructured and rich in content. Conventional database systems, which are designed to handle structured data and support exact match retrieval, are inadequate for this type of data.

- Traditional IR system does not support metadata information such as that provided by database management system.
- The type of search that is most commonly associated with multimedia is content-based : the basic idea is that still images, music extracts, video clips themselves can be used as queries and that the retrieval system is expected to return 'similar' database entries. This technology differs most radically from the thousands-year-old library card paradigm in that there is no necessity for metadata at all.
- Metadata are pieces of information about a multimedia object that are not strictly necessary for working with it, but that are useful to :
  1. Describe resources so they can be indexed, classified, located, browsed and found.
  2. Store technical information, such as data formats and compression schemes.
  3. Manage resources such as their rights or where they are currently located.
  4. Record preservation actions.
  5. Create usage trails, e.g. which section of a video has been watched how many times.

Traditional IR	Multimedia IR
Does not support metadata information.	Require some forms of database schema.
Does not require metadata handling.	Require metadata handling.

- Multimedia IR systems require some form of database schema because sever multimedia applications need to structure their data at least partially. Th architecture of a Multimedia IR system depends on two main factors :
  1. The peculiar characteristics of multimedia data.
  2. The kinds of operations to be performed on such data .

Multimedia IR : Challenges

- Challenges in Multimedia IR are,
  1. Heterogeneity of data
  2. Fuzziness of information
  3. Loss of information in the creation of indexes
  4. Need of an interactive refinement of the query result.

Data modeling

- A data model should be defined by which the user can specify the data to stored into the system.

- A Multimedia IR system should be able to :
  1. Represent and store multimedia objects in a way that ensures fast retrieval
  2. Deal with different kinds of media
  3. Deal with semi-structured data
  4. Extract features from multimedia objects.
- It also provides a model for the internal representation of multimedia data.

4.2.2 Data Retrieval

Data retrieval relies on the following steps :

1. **Query specification :** User specifies the request. The query interface should allow the user to express fuzzy predicates for proximity searches.
   - Proximity predicates : ex. 'Find all images similar to a car'
   - Content-based predicates : ex. 'Find multimedia objects containing an apple'
   - Conventional predicates on the object attributes : ex. Conditions on the attribute 'color' of an image such as 'Find all red images'
   - Structural predicates : ex. 'Find all multimedia objects containing a video clip'
2. **Query processing and optimization :** The query is parsed and compiled into an internal form.
   - Traditional systems : Query is parsed, compiled into an internal form (may also be optimized)

- Query processing is a very complex activity in Multimedia IR system. Difficulties arise when deal with the presence of fuzzy terms, content-based predicates and structural predicates. The heterogeneity of data requires different query processing strategies.

3. **Query answer :** Returning the retrieved objects to the user.
   - The retrieved objects are returned to the user in decreasing order of relevance. Relevance is measured as a distance function from the query object to the stored ones.
4. **Query iteration :** Repeat until user is satisfied
   - Traditional DBMSs : Query process ends when the system returns the answer to the user.

- Multimedia IR : The user supplies additional information to the system to refine the result. This is due to the impreciseness of the answer.
- Multimedia IR system requires integration of traditional IR technology with the technology of multimedia database management systems to represent, manage, store multimedia objects. Object retrieval is based on a similarity approach.

- We should combine DBMS and IR technology :
  a. DBMS : Data modeling capabilities.
  b. IR system : Similarity-based query capabilities.

4.3 Data Modeling

- Traditional DBMSs are targeted to support conventional data.
- The multimedia data are not encoded into attributes provided by the data schema It requires large storage and the content is difficult to analyze and compare.
- Addressing data model in Multimedia IR systems consisting of two main tasks :
  1. A data model should be defined by which the user can specify the data to be stored into the system i.e. integrated support for both conventional and multimedia data types and provide methods to analyze, retrieve, and query.
  2. The system should provide a model for the internal representation of multimedia data.
- The performance of the OODBMS in terms of storage techniques, query processing and transaction management is not comparable to that of relational DBMS. OODBMS are highly non standard. Object database management group defined standard language but very few systems support this language.

Problems of data modeling

- The goal of the object relational technology is to extend the relational model with the ability of representing complex data types by maintaining the performance and the simplicity of relational DBMS and related query language.
- The possibility of defining abstract data types inside the relational model allows one to define ad hoc data types for multimedia data.
- How multimedia data are represented inside the system ?. Due to the nature of the multimedia data, it is not sufficient to describe it through a set of attributes as usually done with traditional data. Some information should be extracted from the objects and used for query processing. The extracted information is typically represented as a set of features.
- Features can be assigned to the multimedia object either manually or automatically by the system. Normally hybrid concept is used by which the system determines some of the values and the user corrects.
- Feature extraction cannot be precise; a weight is usually assigned to each feature value representing the uncertainty of assigning such a value to that feature.

4.3.1 Multimedia Data Support in Commercial DBMS

- To represent multimedia data, current relational DBMS support variable length data types. Data type supported by commercial DBMS is mostly non-standard and each DBMS vendor uses the different names for such data types and provides support for different operation on them.
- Oracle DBMS provides the VARCHAR2 data type to represent variable length character strings. The maximum length of VARCHAR2 data is 4000 bytes. The RAW and LONG RAW data types are used for data that is not to be interpreted by Oracle.
- BLOB : Binary LOB data
- CLOB : Character LOB data
- Sybase SQL server supports IMAGE and TEXT data types to store images and unstructured text respectively and provides a limited set of functions for their searching and manipulation. Most commercial relational DBMSs vendors are inventing a lot of effort in extending the relational model with the capability of modeling complex objects, typically for the object oriented context. SQL3 is an example of this type.
- In SQL3, each type specification consists of both attributes and function specifications. User defined functions can be either visible from any object or only visible in the object they refer to. Both single and multiple inheritances can be defined among user defined types and dynamic late binding is provided.
- SQL3 also provides three types of collection of data types: sets, multisets and lists. Several system defined operations are provided to deal with collections. SQL3 provides a restricted form of object identifier that supports sharing and avoids data duplication. But SQL3 has not yet been officially published most commercial products.
- Oracle provides data cartridges for text, spatial data, and image, audio and vide data. In addition to the efficient and secure management of data ordered unde the relational model, Oracle provides support for data organized under the objec model. Object types and other features such as large objects (LOBs), externa procedures, extensible indexing, and query optimization can be used to buil powerful, reusable server-based components called data cartridges.
- Content based queries on text documents can be combined with traditional queri in the same SQL statement and can be efficiently executed due to the use indexing techniques specific for texts.
- Illustra provides 3D and 2D spatial data blades for modeling spatial data. T supported data types include boxes, vectors, quadrangles etc and examples

supported operations are INTERSECT, CONTAINS, OVERLAPS, CENTER and so on.

- Spatial data blades also implements R-trees for performing efficient spatial queries. The text data blade provides data types for representing unstructured text and performing content based queries. Illustra supports a data blade which can be used to query images by content.
- The object relational technology and its extensive type system is now starting to be widely used both in industrial and research projects.

4.3.2 The MULTOS Data Model

- MULTimedia Office Server (MULTOS) is a multimedia document server with advanced document retrieval capabilities, developed in the context of an ESPRINT project in the area of office systems. It is based on the client/server architecture. Three different types of document servers are supported :
  1. Current servers
  2. Dynamic servers
  3. Archive servers.
- All these servers differ in storage capacity and document retrieval speed. These servers support filing and retrieval of multimedia objects based on document collections, document types, document attributes, document text and images.
- The multimedia filing system consists of a number of autonomous subsystems called document servers and a number of client subsystems.
- The MULTOS data model allows :
  a) The representation of high level concepts present in the documents contained in the database.
  b) The grouping of documents into classes of documents having similar content and structure.
  c) The expression of conditions on free text.
- Each document is described by a logical structure, a layout structure, and a conceptual structure. Documents having similar conceptual structures are grouped into conceptual types. Each document is described by :
  1. **Logical structure :** Determines arrangements of logical document components (titles, introduction, chapter, section, etc.)
  2. **Layout structure :** A deal with the layout of the document content and it contains components (pages, frames, etc.)
  3. **Conceptual structure :** Allows a semantic oriented description of the document content.

- In MULTOS the representation of documents and operations on them, are based on a formal model. At the beginning, a standardized document presentation was assumed, i.e. the ODA model. The Office Document Architecture (ODA) is a standard defined by ISO. ODA gives a formal description of the document composition (logical structure) and a formal device independent description of the document presentation/rendition (layout structure).
- The logical structure associates the content of the document with a hierarchy of logical objects, where as the layout structure associates the same content with a hierarchy of layout object.
- Documents having similar conceptual structures are grouped into conceptual types. Conceptual types are maintained in a hierarchy of generalization. The types can be strong (completely specifies the structure of its instance) or weak (partially specifies the structure of its instance). Component of unspecified types are called **spring component types.**
- The conceptual structure of the type **Generic_Letter** is shown in Fig. 4.3.1.

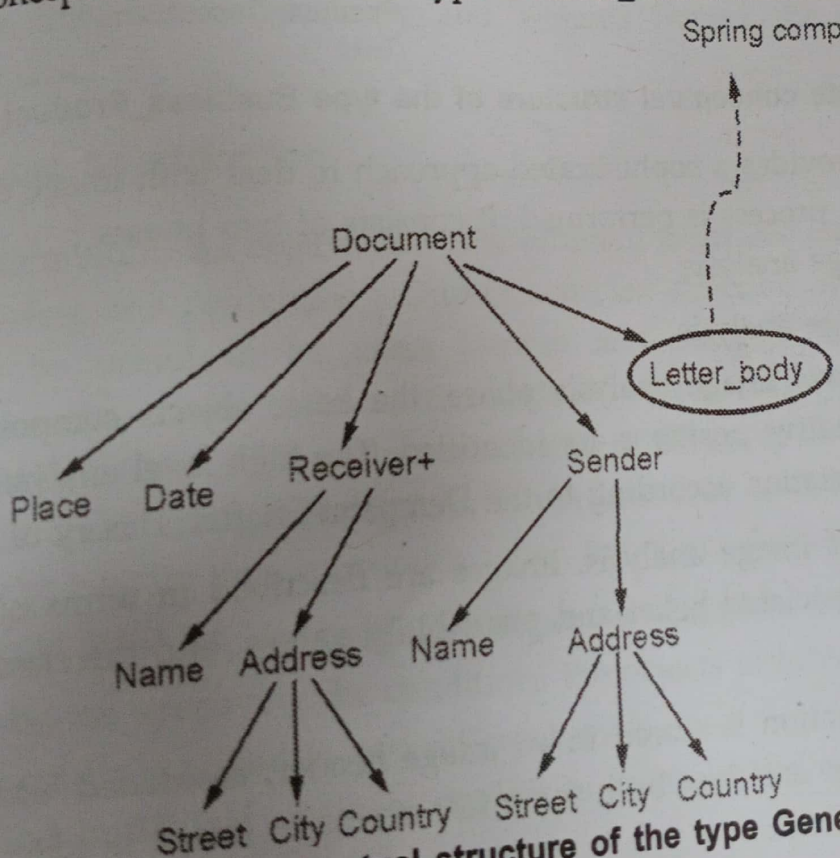


Fig. 4.3.1 Conceptual structure of the type Generic_Letter

- Fig. 4.3.2 shows complete conceptual structure of the type **Business_Product_Lette** This type has been obtained from Generic_Letter by specialization of Letter_Boc into a complex conceptual components, defined as an aggregation of fi conceptual components.

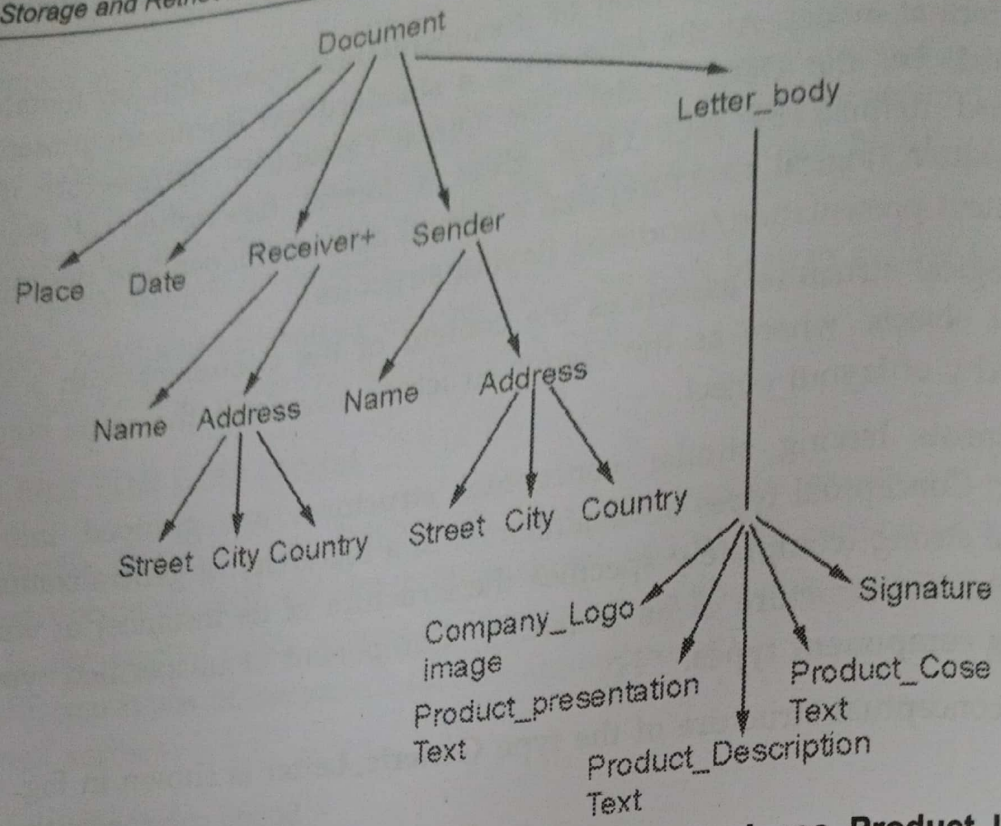


Fig. 4.3.2 Complete conceptual structure of the type Business_Product_Letter

- MULTOS also provides a sophisticated approach to deal with image data. Initially an image analysis process is performed. It consists of two phases :
  1. Low level image analysis
  2. High level image analysis
- During the low level image analysis phase, the basic objects composing a given image and their relative positions are identified. The high level analysis phase deal with image interpretation according to the Dempster-Shafter Theory of evidence.
- In the last phase of image analysis, images are described in terms of the objects recognized, with associated belief and plausibility values, and the classes to which they belong.
- Image access information is stored in an image header, associated with the image file. Access structures are then built for a fast access to image headers. Two types of index are constructed :
  1. **Object index :** For each object a list is maintained. Each elements of a list is a pair of (BI, IMH), where IMH is a pointer to the header of the image containing the object, and BI is the associated belief interval representing the probability that the image considered really contains the object.
  2. **Cluster index :** For each image class, a list of pairs (MF, IMH) is maintained. IMH is a pointer to an image header corresponding to an image with a

non-null degree of membership to the class, and MF is the value of the membership degree.

4.4 Query Languages

- Queries in relational or object-oriented database systems are based on an exact match mechanism, by which the system is able to return exactly those tuples satisfying some well specified criteria given in the query expression.
- When the query is submitted, the features of the query object are matched with respect to the features of the objects stored in the database and only the objects that are more similar to the query one are returned to the user.
- In designing a multimedia query language, following points are considered :
  1. How the user enters his/her request to the system, i.e. which interfaces are provided to the user for query information ?
  2. Which conditions on multimedia objects can be specified in the user request.
  3. How uncertainty, proximity and weights impact the design of the query language.

4.4.1 Request Specification

- Querying multimedia objects uses two different interfaces for the user.
  1. Browsing and navigation : Due to complex structure of multimedia objects, it may be useful to let users browse and navigate inside the structure of multimedia objects to locate the desired objects. This type of approach is used in CAD/CAM/CASE environments due to the complex structure of the objects under consideration. Navigation is not best way to find multimedia objects, it may be heavily time consuming when the object desired is deeply nested.
  2. Second approach for selecting objects is therefore based as traditionally in DBMSs, on specifying the conditions the objects of interest must satisfy.
- Queries can be specified in two different ways :
  1. Typical traditional database context is to enter the query by using a specific query language.
  2. Query by example approach is used sometimes. Here queries are specified b using actual data inside a visual environment; the user provides the syste with an object example that is then used to retrieve all the stored objec similar to the given one.

4.4.2 Conditions for Multimedia Data

- Multimedia query languages should provide predicates for expressing conditions on the attributes, the content and the structure of multimedia objects. Query predicates can be classified into three different groups :
  1. Attributes predicates concern the attributes of multimedia objects.
  2. Structural predicates concern the structure of the data being considered.
  3. Semantic predicates concern the semantic and unstructured content of the data involved.
- **Attributes predicates** supply the exact value for each object. Examples of attributes are the speaker of an audio object, the size of an object, or its type.
- **Structural predicates** concern the structure of multimedia objects. Such predicates can be answered by using some form of metadata and information about the database schema. An example of use of a structural predicate is the query: " *Find all multimedia objects containing at least one image and a video clip.*"
- **Semantic predicates** concern the semantic content of the queries data, depending on the features that have been extracted and stored for each multimedia object. An example of semantic query is : *"Find all the objects containing the word OFFICE"* Here the word OFFICE may appear either in a textual component of the object or as a text attributes of some image components.
- The query "Find all the red house" is a query on the image content. This query can executed only if color and shape are features that have been previously extracted from images. Current system support semantic predicates only with respects to specific features, such as the color, shape and texture.
- **Difference between attributes predicates and semantic predicates**
  A. Semantic predicates can apply the exact match. There is no guarantee that the objects retrieved by this type of predicate are 100 % correct or precise.
  B. The result of query involving semantic predicates is a set of objects, each which has an associated degree of relevance with respect to the query.
- Structural and semantic predicates can also refer to spatial or temporal properties of multimedia objects. Spatial semantic predicates specify conditions about the relative positions of a set of objects in an image or a video.
- Examples of spatial semantic predicates are : **contain, intersect, is contained in, adjacent to.**
- Temporal semantic predicates are mainly related to continuous media like audio and video. They allow one to express temporal relationships among the various frames of a single audio or video.

4.4.3 Uncertainty, Proximity and Weights in Query Expressions

- In designing multimedia query language, how it is possible to specify the degree of relevance of the retrieved objects. This can be done in following ways :

1. By using imprecise terms and predicates like normal, unacceptable , typical. Each of those terms does not represent a precise value but a set of possible acceptable values with respect to which the attribute or the feature has to be matched.
2. By specifying particular proximity predicates. The predicate does not represents a precise relationship between objects or between attributes and values. The relationship represented is based on the computation of a semantic distance between the query object and the stored ones, on the basis of the extracted features.
3. By assigning each condition or term a given weight, specifying the degree of precision by which a condition must be verified by an object. For example, the query "Find all the objects containing an image representing a screen (HIGH) and a keyboard (LOW)", can be used to retrieve all the objects containing an image representing a screen and a keyboard.

4.4.4 Query Languages Supporting Retrieval of Multimedia Object

- In this section we discuss how the standard language support the multimedia applications.

4.4.4.1 SQL3 Query Language

- Retrieving data from a database consists of three main processes :
  1. Formulation of an information/data request - the query,
  2. Query execution by the DBMS query processor and
  3. Result presentation.
- SQL3 was accepted as the new standard for SQL in 1999, after more than 7 years of debate. Basically, SQL3 includes data definition and management techniques from Object-Oriented DBMS, OO-DBMS, while maintaining the relational DBMS platform. Based on this merger of concepts and techniques, DBMSs that support SQL3 are called Object-Relational.
- SQL3 is a superset of SQL/92, in that it supports all of the constructs supported by that standard, as well as adding new ones of its own. Therefore, whatever worked in an implementation of SQL/92 should also work in an implementation of SQL3.

- It provides support for an extensible type system. Extensibility of the type system is achieved by providing constructs to define user-dependent abstract data types, in an object-oriented like manner. Each type specification consists of both attribute and function specifications.
- A strong form of encapsulation is provided; in that attribute values can only be accessed by using some system functions. User-defined functions can be either visible from any object or only visible in the object they refer to. Both single and multiple inheritances can be defined among user-defined types and dynamic late binding is provided.
- The most central data modeling notions included in SQL3 and support specification of :
  1. Classification hierarchies,
  2. Embedded structures that support composite attributes,
  3. Collection data-types (sets, lists/arrays, and multi-sets) that can be used for multi-valued attribute types,
  4. Large OBject types, LOBs, within the DB, as opposed to requiring external storage, and
  5. User defined data-types and functions (UDT/UDF) that can be used to define complex structures and derived attribute value calculations, among many other function extensions.
  6. Query formulation in SQL3 remains based in the structured, relational model, though several functional additions have been made to support access to the new structures and data types.
- SQL3 allows the user to integrate external functionalities with data manipulations. Functions of external library can be introduced into a database system as external functions.
- SQL3 support the active rules by which the database is able to react to some system or user dependent events by executing specific actions. Active rules or triggers are very useful to enforce integrity constraints.
- The ability to deal with external functions and user defined data types enables the language to deal with objects with a complex structure as multimedia objects.

Drawback of SQL3

- The ability to perform content based search is application dependent. Objects are not ranked and are therefore returned to the application as a unique set.
- Specialized indexing techniques can be used but they are not transparent to the user.
- No IR techniques are integrated into the SQL3 query processor

4.4.4.2 MULTOS Query Language

- In general, a MULTOS query has the form :

  FIND DOCUMENTS VERSION version-clause

  SCOPE scope-clause

  TYPE type-clause

  WHERE condition-clause

  WITH component

Where :

a) The **version-clause** specifies which versions of the documents should be considered by the query.

b) The **scope-clause** restricts the query to a particular set of documents. This set of documents is either user defined document collection or a set of documents retrieved by a previous query.

c) The **type-clause** allows the restriction of a query to document belonging to a pre-specified set of types. When no type is specified, the query is applied to all document types.

d) The **condition-clause** is a Boolean combination of simple conditions on document components.

e) The **with-clause** allows one to express structural predicates. Components are a path name and the clause looks for all documents structurally containing such a component.

- Different types of conditions can be specified in order to query different types of media. MULTOS supports three main classes of predicates :
  1. Predicates on data attributes : Exact match search is performed.
  2. Predicates on textual components : Determining all objects containing some specific strings.
  3. Predicates on images : Specifying conditions on the image content.
- Following example shows the basic features of the MULTOS query language :

```
FIND DOCUMENTS VERSION LAST WHERE
Document.Date > 29/11/2002 AND
(*Sender.Name = "Rakshita" OR
 *Product_Presentation CONTAINS "Rakshita ") AND
*Product_Description CONTAINS "Personal Computer" AND
(*Address.Country = "India" OR  TEXT CONTAINS "India") AND
WITH *Company_Logo
```

- According to the above query, the user looks for the last version of all documents dated after November 2002, containing a company logo, having the word 'Rakshita' either as sender name or in the product presentation, with the word 'Personal Computer' in the product description section and with the word 'India' either constituting the country in the address or contained in any part of the entire document.
- Example : This example shows the uncertainty is expressed by associating both a preference and an importance value with the attributes in query. Such values are then used for ranking the retrieved documents.

```
FIND DOCUMENTS VERSION LAST WHERE
(Document.Date BETWEEN (12/31/1998, 1/31/98) PREFERRED BETWEEN
   (2/1/1998, 2/15/98) ACCEPTABLE) HIGH AND
(*Sender.Name = "Olivetti" OR
*Product_Presentation CONTAINS "Olivetti") HIGH AND
(*Product_Description CONTAINS "Personal Computer") HIGH AND
(*Product_Description CONTAINS "good ergonomics") LOW AND
(*Address.Country = "Italy" OR TEXT CONTAINS "Italy") HIGH AND
WITH *Company_Logo HIGH
(IMAGE MATCHES
screen HIGH
Keyboard HIGH
AT LEAST 2 floppy_drives LOW) HIGH
```

4.5 Background - Spatial Access Method

- The main purpose of spatial access methods is to support efficient selection objects based on spatial properties.
- The idea is to map objects into points in "f-D" space and to use multi-attrib access methods to cluster them and to search for them. The prevailing meth form three classes :
  1) R*-trees and the rest of the R-tree family,
  2) Linear quad trees and
  3) Grid-files.
- The R-tree represents a spatial object by its Minimum Bounding Rectangle(MB Data rectangles are grouped to form parent nodes, which are recursively group to form grandparent nodes and eventually, a tree hierarchy.

4.6 Generic Multimedia Indexing Approach

- The GEMINI approach is based on the following two ideas. It is a 'quick-and-dirty' test to discard the vast majority of non-qualifying objects. The spatial access method is used to achieve faster searching than sequential searching. For such queries the problem is defined as follows :
  a. Collection of N objects : $O_1, O_2, \ldots, O_N$.
  b. The distance/dissimilarity between two objects (O_i, O_j) is given by the function $D(O_i, O_j)$, which can be implemented as a program.
  c. The user specifies a query object Q, and a tolerance ε .
- Design fast search algorithms that locate objects that match a query object, exactly or approximately. An obvious solution is to apply sequential scanning: for each and every object , we can compute its distance from Q and report the objects with distance $D(Q, O_i) <= \varepsilon$.
- However, sequential scanning may be slow for two reasons :
  1. The distance computation may be expensive.
  2. The database size N might be very large.
- GEMINI aims to provide a faster alternative, and is based on two ideas :
  1. A quick-and-dirty test, to discard quickly the vast majority of non-qualifying objects.
  2. The use of spatial access methods to achieve faster-than-sequential searching.
- The reason for the distance computation being expensive is that multimedia objects can have a very large dimensionality. The quick-and-dirty test is designed to reduce this dimensionality to more manageable proportions, often to only one or two dimensions.
- Effectively, each multimedia object is projected onto a lower dimensional space by extracting some important features. The distance between the query and collection objects is measured in this lower-dimension space, with little computation effort.
- Any object that is distant from the query object by more than is disqualified from further consideration. Finally, each collection object that was not disqualified is then fully compared to the query object in the original high dimensional space.
- Spatial access methods involve segmenting the multimedia objects into smaller, logically-cohesive components, and storing these in a data structure such that the original object can easily be reconstructed.
- For example, a video clip could be divided into scenes, while a still image could be stores as a fixed number of overlapping rectangular segments. When evaluating a query, objects can be compared on a component-by-component basis, possibly in short-circuit.

- Time series data have high complexity. This high complexity can be reduced by time series representation methods which aim at reducing the high complexity by transforming the time series into a lower dimensional space and performing the similarity search process at these lower dimensional spaces.

4.7.1 Distance Function

- According to GEMINI (algorithm 2), the first step is to determine the distance measure between two time series. A typical distance function is the Euclidean distance, which is routinely used in financial and forecasting applications.
- In a digital multimedia era, the research of Content Based Image Retrieval (CBIR) used to establish a database composed of images; each is represented as a vector of features derived from color, shape, and/or texture information. When the query is requested, a similarity measurement between a user-provided image and those pre-stored in the database is computed and compared to report a few of most similar images.
- The Euclidean distance between $x, y \in R^d$ is computed by

$$\delta_1 (x, y) = || x - y ||_2 - \sqrt{\sum_{j=1}^{d} (x_j - y_j)^2}$$

4.7.2 Feature Extraction and Lower Bounding

- Here we need some better features. Applying the second step of the GEMINI algorithm, we ask the feature-extracting question : 'If we are allowed to use only one feature from each sequence, what would this feature be ?' **A natural answer is the average.**
- The third step of the GEMINI methodology is to show that the distance in feature space lower-bounds the actual distance. The solution is provided by Parseval's theorem, which states that the DFT preserves the energy of a signal, as well as the distances between two signals :

$$D (\bar{x}, \bar{y}) = D (\bar{X}, \bar{Y})$$

where $\bar{X}$ and $\bar{Y}$ are Fourier transforms of $\bar{x}$ and $\bar{y}$, are respectively

- Thus, if we keep the first $f (f \leq n)$ coefficients of the DFT as the features, we lower-bound the actual distance :

$$D_{feature} (f(\bar{x}), F (\bar{y})) = \sum_{F=0}^{f-1} |X_F - Y_F|^2 \leq \sum_{F=0}^{n-1} |X_F - Y_F|^2 = \sum_{i=0}^{n-1} |x_i - y_i|^2$$

and finally $D_{feature} (F (\bar{x}), F (\bar{y}) \leq D (\bar{x}, \bar{y}))$

There will be no false dismissals. DFT concentrates the energy in the first few coefficients, for a large class of signals, the colored noises. These signals have a skewed energy spectrum ($O(F^{-b})$).

0. $b = 2$: It is called random walks or brown noises. Model stock movements and exchange rates.
1. $b > 2$: Black noises. It is model water level of rivers and rainfall patterns.
2. $b = 1$: Pink noise. 'Interesting signals': musical scores and other works of art.

- White noise is unpredictable, brown noise is too predictable. The 2D signals like photographs are far from white noise, exhibiting a few strong coefficients in the lower spatial frequencies. The JPEG image compression standard exploits this phenomenon, effectively ignoring the high frequency components of the discrete cosine transform, which is closely related to the Fourier transform.

4.7.3 Experiments

- Sequential scanning method is used for compression. The R*-tree was used for the spatial access method within GEMINI. Fig. 4.7.1 shows the break-up of the response time, as a function of the number f of DFT coefficient kept. The diamonds, triangles and squares indicates total time, post processing time and R*-tree time respectively.

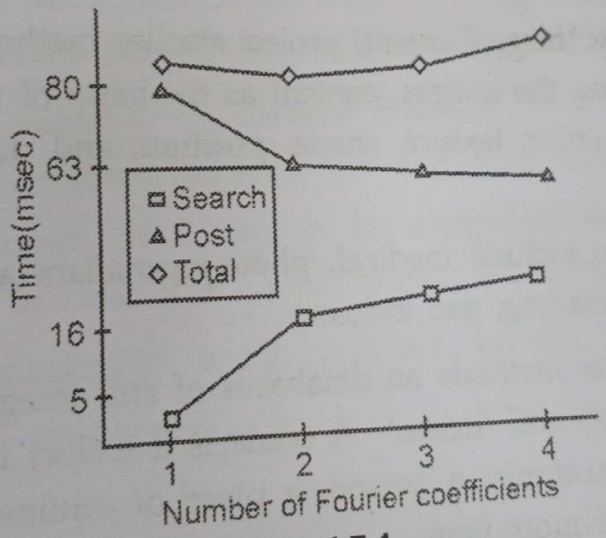


Fig. 4.7.1

- Artificially generated random walks with sequence length n = 1024 and database size N = 50 to 400.
- Fig. 4.7.2 shows response time for the two methods i.e. GEMINI and sequential scan, as a function of the number of sequences N.
- Conclusion from application of GEMINI on time series are the following :
  1. GEMINI can be successfully applied to time series, specifically to the ones that behave like colored noises.

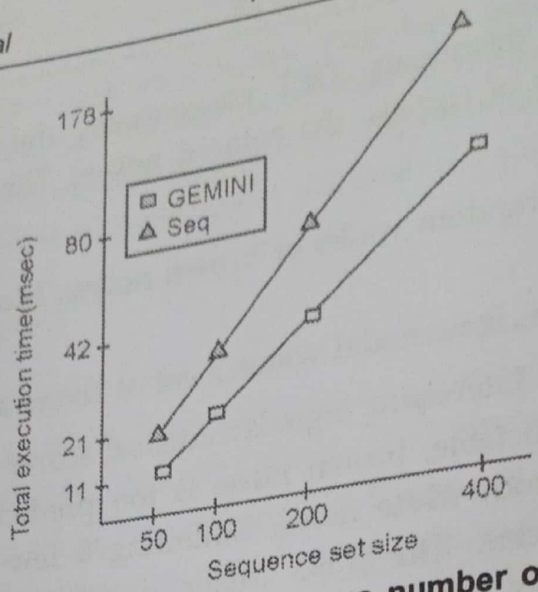


Fig. 4.7.2 response time per query versus number of N of sequences

2. For signals with skewed spectrum like the above ones, the minimum in the response time is achieved for a small number of Fourier coefficients. The minimum is rather flat, which implies that a suboptimal choice for f will give search time that is close to the minimum.
3. The success in 1D series suggests that GEMINI is promising for 2D or higher dimensionality signals, if those signals also have skewed spectrum.

4.8 Two Dimensional Color Images

- The QBIC (Query By Image Content) project studies methods to query large online image databases using the images, content as the basis of the queries. Examples of the content include color, texture, shape, position, and dominant edges of image items and regions.
- Potential applications include medical, photo-journalism and many others in art fashion, cataloging, retailing, and industry.
- In this we will discuss methods on databases of still images, with two main data types: **'images'**('scenes') and **'items.'** A scene is a (color) image, and an item is part of a scene, for example, a person, a piece of outlined texture, or an appl Each scene has zero or more item.

4.8.1 Image Features and Distance Functions

- Here we used color features because color presents an interesting problem, which can be resolved by the GEMINI approach.
- For color, we compute a k-element color histogram for each item and scene, where k = 256 or 64 colors. Each component in the color histogram is the percentage pixels that are most similar to that color. Fig. 4.8.1 shows an example of co

histogram of a fictitious photograph of a sunset. The following gives an example of such a histogram of a fictitious photograph of a sunset: there are many red, pink, orange, and purple pixels, but only a few white and green ones.

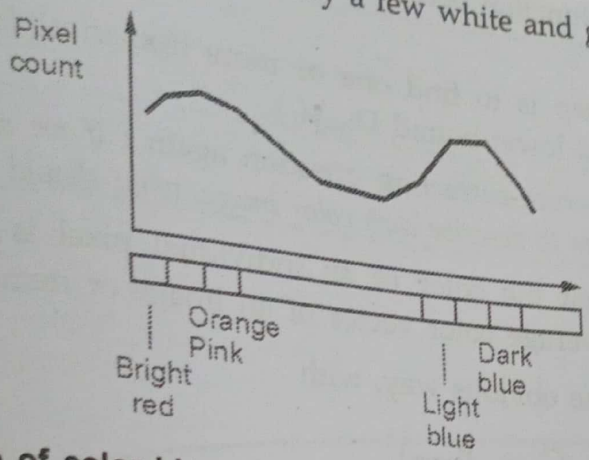


Fig. 4.8.1 Example of color histogram of a fictitious photograph of a sunset

- One method to measure the distance between two histograms (k×1 vectors) is given by

$$d^2_{hist}(\vec{x},\vec{y}) = (\vec{x},\vec{y})\, A\, (\vec{x},\vec{y}) = \sum_{i}^{k}\sum_{j}^{k} a_{ij}\,(x_i - y_i)\,(x_j - y_j)$$

8.2 Lower Bounding

- In applying the GEMINI method for color indexing, there are two obstacles :
  1. The 'dimensionality curse'(k may be large, e.g., 64 or 256 for color features)
  2. The quadratic nature of the distance function.
- To compute the distance between the two color histogram $\vec{x}$ and $\vec{q}$ then, e.g., bright-red component of $\vec{x}$ has to be compared not only to the bright-red component of $\vec{q}$, but also to the pink, orange, etc. components of $\vec{q}$.
- Fig. 4.8.2 shows illustration of the cross talk between two color histograms.

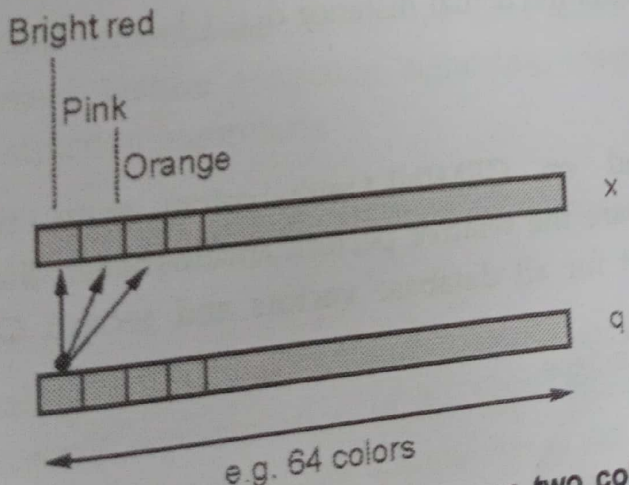


Fig. 4.8.2 Illustration of the cross talk between two color histograms

- To resolve the cross-talk problem, the GEMINI approach (algorithm 2) is used. The first step of the algorithm has been done :
  1. The distance function between two color images is given by equation, that is $D( ) = D_{hist} ( )$.
  2. The second step is to find one or more numerical features, whose Euclidean distance would lower-bound $D_{hist} ( )$.
  3. We ask the feature-extracting question again : *If we are allowed to use only one numerical feature to describe each color image, what should this feature be ?*
  4. This means that the color of an individual pixel is described by the triplet (R,G,B). The average color vector of an image or item $\bar{x} = (R_{avg}, G_{avg}, B_{avg})^t$ is defined in the obvious way, with

$$\bar{x} = (R_{avg}, G_{avg}, B_{avg})^t$$

$$R_{avg} = (1/P) \sum_{p=1}^{P} R(p)$$

$$G_{avg} = (1/P) \sum_{p=1}^{P} G(p)$$

$$B_{avg} = (1/P) \sum_{p=1}^{P} B(p)$$

Where P is the number of pixels in the item, and R(p), G(p), and B(p) are the red, green and blue components respectively of the p-th pixel.

- Given the average colors and $\bar{x}$ of two items, we define $d_{avg} ( )$ as the Euclidean distance between the three-dimensional average color vectors,

$$d^2_{avg}(\bar{x}, \bar{y}) = (\bar{x} - \bar{y})^t (\bar{x} - \bar{y}).$$

- The third step of the GEMINI algorithm is to prove that our simplified distance $d_{avg} ( )$ lower-bounds the actual distance $d_{hist} ( )$.

4.8.3 Experiment

- Results are based on GEMINI with color using the bounding theorem. Experiments compare the relative performance between the first, simple sequential evaluation of dhist for all database vectors and second GEMINI. The parameters used as follows :
  1. N = 924 color images
  2. K = 256 colors
  3. CPU time and disk accesses

- Fig. 4.8.3 shows the response time as a function of the selectivity. Figure also shows the CPU time for each method.

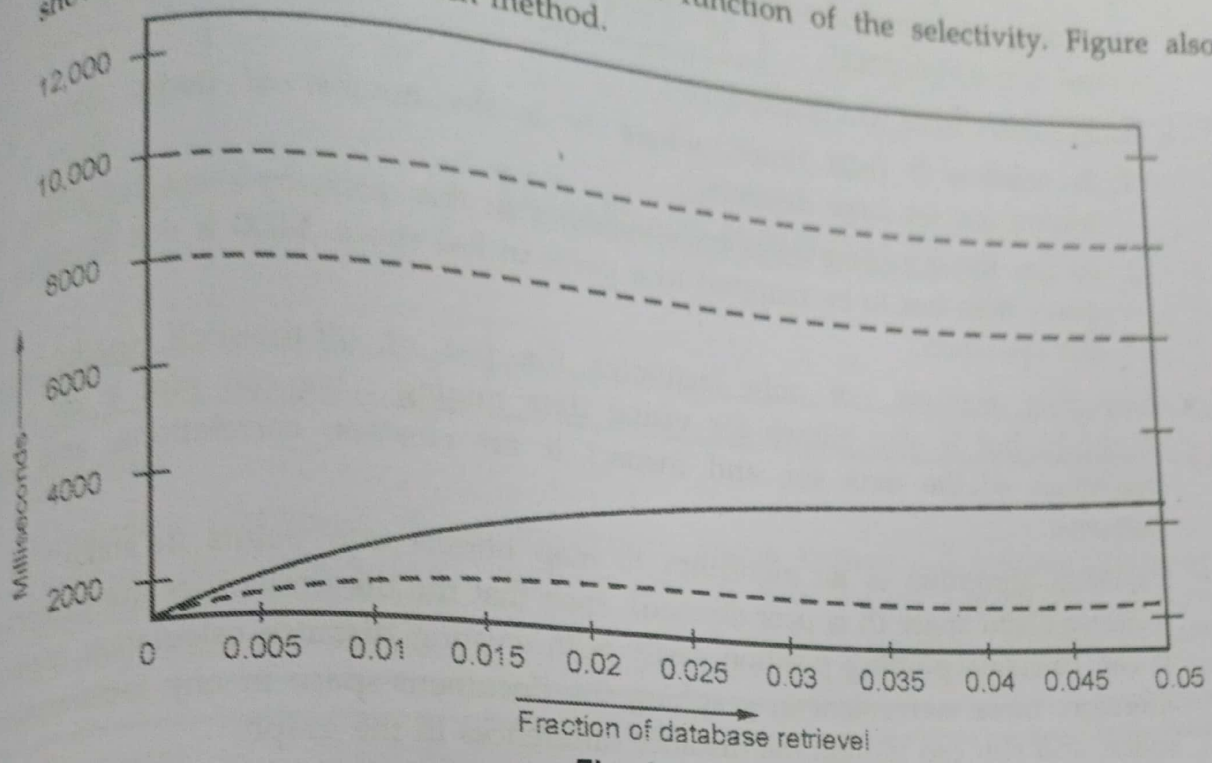


Fig. 4.8.3

- Conclusion :
  1. The GEMINI approach motivated a fast method , using the average RGB distance.
  2. It resolve the cross talk problem, GEMINI solved the dimensionality curve problem at no extra cost.

4.9 Automatic Feature Extraction

- GEMINI is useful for any setting that we can extract features from. Feature extraction algorithms can extract features that may be a combination of the original features.
- If the feature extraction methods do not use the class labels of every sample they are called unsupervised feature extraction algorithm, otherwise they are called supervised feature extraction algorithms.
- Automatic feature extraction methods are
  1. Multidimensional Scaling (MDS)
  2. FastMap.
- Multidimensional scaling (MDS) is a nonlinear feature extraction technique, which finds a lower dimensional representation of the high dimensional data, such that the distances between the samples in the original space are preserved, as much as possible in the reduced space. If the distance measurements are calculated

according to a metric measurement this is called metric MDS. Otherwise it is called non-metric MDS.

- MDS suffers from two drawbacks :
  1. It requires 0 (N2) times, where N is the number of items. Thus, it is impractical for large datasets.
  2. Its use for fast retrieval is questionable : In the 'query-by-example' setting, the query item has to be mapped to a point in k-d space. MDS is not prepared for this operation.
- Extracting features not only facilitates the use of off-the-shelf spatial access methods, but it also allows for visual data mining : we can plot a 2D or 3D projection of the data set, and inspect it for clusters, correlations, and other patterns.
- **FastMap** algorithm is an algorithm to map objects into points in some k-dimensional space (k is user-defined), such that the dis-similarities are preserved. It can calculate plotting position easily with existing distance calculation methods therefore more convenient in graphing the document space in any k-dimensional space, and still can remain the distance differences in the graph.
- The design of FastMap as a linear algorithm fulfills goals such as :
  - It solves the general problem.
  - It is linear on the database size, and therefore much faster than MDS.
  - At the same time, it leads to fast indexing, being able to map a new, arbitrary object into a k-d point in O(k) distance calculations regardless of the database size N.
- The simple step of FastMap algorithm works as follows :
  1. Find the objects which have most long distance each other among all documents.
  2. These objects are called the pivot objects.
  3. Calculate xi by cosine law to project the objects on a line.
  4. To map objects in k-d space, use Euclidean distance function D'().
  5. Do loop depend on the expected dimensional space.
  6. Eventually it will give 'images' of objects by which mapping in space is possible.

An example for GEMINI approach

- Let's consider a database of time series, such as yearly stock price movement, with one price per day. Distance function between two such series S and Q is the Euclidean distance.

$$D(S, Q) = \left(\sum_{i=1} S[i] - Q[i]^2 \right)^{\frac{1}{2}}$$

where S[i] stands for the value of stock S on i^{th} day.

- The idea behind the quick-and-dirty test is to characterize a sequence with a single number (feature), which help us discard many non-qualifying sequences. Average stock price over the year, standard deviation, some of the Discrete Fourier Transform (DFT) coefficients.

Mapping function

- Let F() be the mapping of objects to f-dimensional points, that is, F(O) will be the f-D point that corresponds to object O. Organize f-D points into a spatial access method, cluster them in a hierarchical structure, like the R*-trees. Upon a query, we can exploit the R*-tree, to prune out large portions of the database that are not promising.
- Fig. 4.6.1 shows basic idea of database sequence.

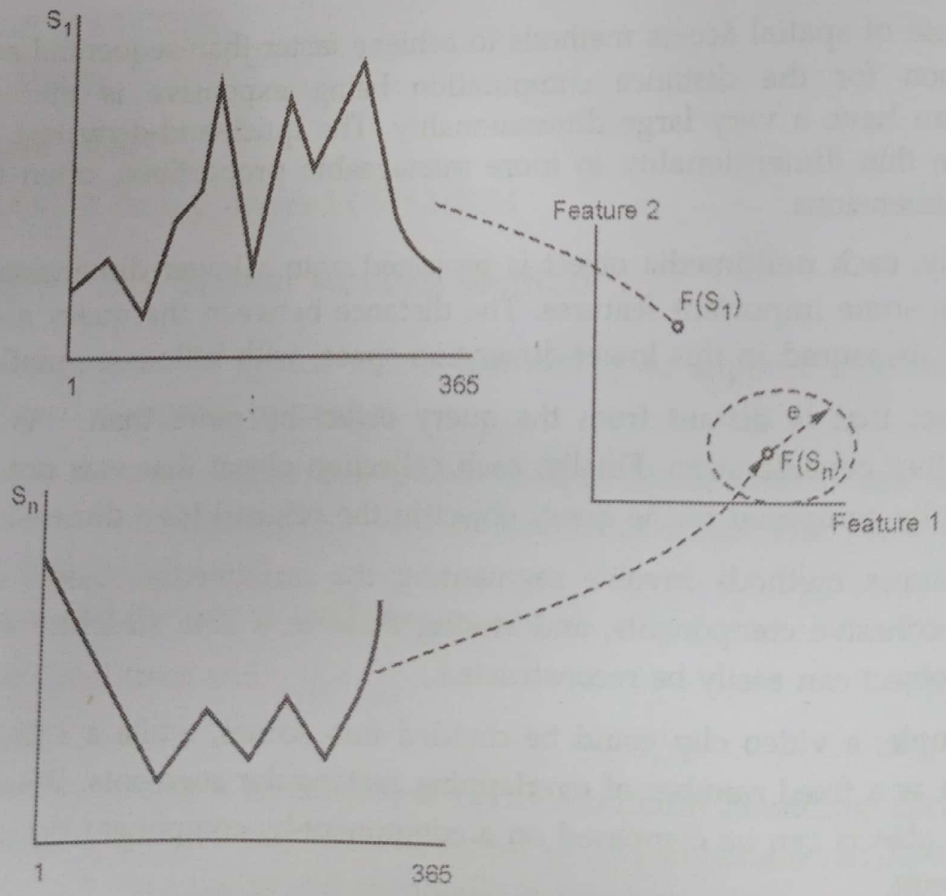


Fig. 4.6.1 Basic idea

- **Search algorithm (for whole match query)**
  1. Map the query object Q into a point F(Q) in feature space.
  2. Using a spatial access method, retrieve all points within the desired tolerance ? from F(Q).
  3. Retrieve the corresponding objects, compute their actual distance from Q and discard the false alarms.
- **Lower bounding lemma**
  1. To guarantee no false dismissals for whole-match queries, the feature extraction function F() should satisfy the following formula

     $D_{feature}(F(O_1), F(O_2)) \leq D(O_1, O_2)$

     Where $D_{feature}()$: Distance of two feature vectors

mapping F( ) from objects to points should make things look closer.

- **GEMINI algorithm**
  1. Determine the distance function D() between two objects.
  2. Find one or more numerical feature-extraction functions, to provide a 'quick-and-dirty' test.
  3. Prove that the distance in feature space lower-bounds the actual distance D( ), to guarantee correctness.
  4. Use a SAM (e.g., an R-tree), to store and retrieve the f-D feature vectors.
- **'Feature-extracting' question :** If we are allowed to use only one numerical feature to describe each data object, what should this feature be ?
- The successful answers to the question should meet two goals :
  a. They should facilitate step 3 (the distance lower-bounding)
  b. They should capture most of the characteristics of the objects.

4.7 One Dimensional Time Series

- Here the goal is to search a collection of (equal-length) time series, to find the ones that are similar to a desirable series. For example, 'in a collection of yearly stock price movements, find the ones that similar to IBM'.
- A time series is a sequence of real numbers, representing the measurements of a real variable at equal time intervals. For examples Stock prices, Volume of sales over time, Daily temperature readings and ECG data.
- A time series database is a large collection of time series.
- Similarity between two time series can be depicted using a similarity distance. Time series similarity search can be viewed as retrieving all the time series in the database that are "near" a given query according to a given similarity distance.

Unit V

5 Web Searching

Syllabus

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

Contents

Unit V

5 Web Searching

Syllabus

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

Contents

5.1 Introduction

- The World Wide Web is developed by Tim Berners-Lee in 1990 at CERN to organize research documents available on the Internet. It combined idea of documents available by FTP with the idea of hypertext to link documents.
- Ted Nelson developed idea of hypertext in 1965. Doug Engelbart invented the mouse and built the first implementation of hypertext in the late 1960's at SRI. ARPANET was developed in the early 1970's.
- The basic technology was in place in the 1970's; but it took the PC revolution and widespread networking to inspire the web and make it practical. Early browsers were developed in 1992.
- In 1993, Marc Andreessen and Eric Bina at UIUC NCSA developed the Mosaic browser and distributed it widely. Andreessen joined with James Clark to form Mosaic Communications Inc. in 1994. Microsoft licensed the original Mosaic from UIUC and used it to build Internet Explorer in 1995.
- In 1998, Larry Page and Sergey Brin, Ph.D. students at Stanford, started Google. Main advance is use of *link analysis* to rank results partially based on authority.
- Clients use browser application to send URIs via HTTP to servers requesting a Web page. Web pages constructed using HTML or other markup language and consist of text, graphics, sounds plus embedded files. Servers respond with requested Web page. Client's browser renders Web page returned by server. The entire system runs over standard networking protocols (TCP/IP, DNS,...).

5.1.1 Searching the Web

- Web documents are known as 'pages', each of which can be addressed by an identifier called a uniform resource locator. Web pages are usually grouped into 'sites', sets of pages published together. For example : *http://www.vtubooks.com*
- The ability to search and retrieve information from the Web efficiently and effectively is an enabling technology for realizing its full potential. With powerful workstations and parallel processing technology, efficiency is not a bottleneck.
- Current search tools retrieve too many documents, of which only a small fraction are relevant to the user query. Furthermore, the most relevant documents do not necessarily appear at the top of the query output order.
- Some pages provide an interactive service such as a search form. Some pages provide a commercial service, allowing users to shop for products online. Some pages are generated dynamically and intended to be used once, such as a search engine results list page.

- Web search engines discover pages by 'crawling' the Web, discovering new pages by following hyperlinks. Access to particular web pages may be restricted in various ways. The set of web pages which can not be included in search engine indexes is often called 'the hidden web', `the deep web', or `web dark matter'.

5.1.2 Web Challenges for IR

- WWW expanding faster than any current search engine can possibly index. Many web pages are updated frequently or are dynamically generated which forces search engines to repeatedly revisit them.
- Many dynamically generated sites are not indexable by search engines; this phenomenon is known as the invisible web. The ordering of results is not always solely by relevance, but some times influenced by monetary contributions. It is difficult for business model.
- Some sites use tricks to manipulate the search engine to improve their ranking for certain keywords; this is known as search engine spamming.

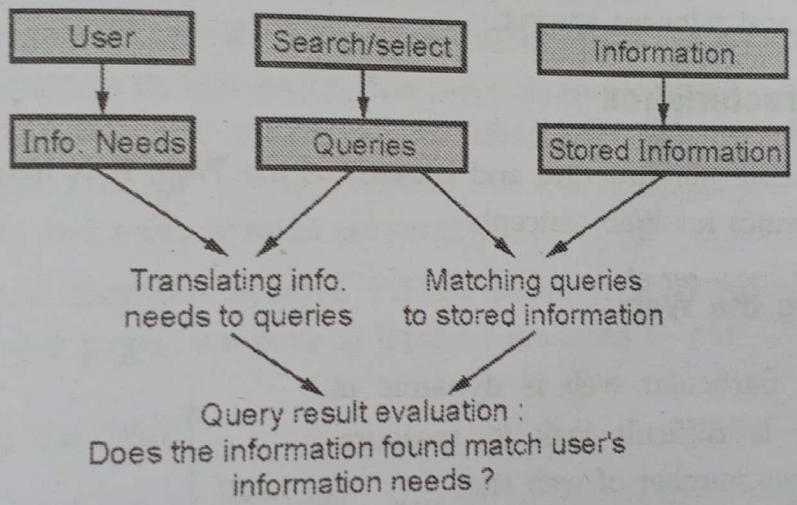


Fig. 5.1.1

- Web problems are divided into two classes : Problems with data itself and problems regarding the user and his/her interaction with the retrieval system.

Problems with data itself

1. Distributed data : Documents spread over millions of different web servers.
2. Volatile data : Many documents change or disappear rapidly (e.g. dead links).
3. Large volume : Billions of separate documents.
4. Unstructured and redundant data : No uniform structure, HTML errors, up to 30 % (near) duplicate documents.

5. Quality of data : No editorial control, false information, poor quality writing, typos, etc.
6. Heterogeneous data : Multiple media types (images, video, VRML), languages, character sets, etc.

- These entire above problem is not solved by improving software. Many of them will not changed because they are problems intrinsic to human nature.

Problems regarding the user

- A problem regarding the user is of two types.
  1. How to specify the query ?
  2. How to interpret the answer provided by the system ?
- A user that is seeking a phone number of their doctor will frequently be frustrated with the answers produced by the search engine. To cope with queries of this nature, search engines need to evolve further. They need to evolve to incorporate knowledge encoded in some form that it can be useful for ranking purposes.
- To get the proper result, submit a good query to the search system and obtain a manageable and relevant answer.

5.2 Web Characteristics

- In characterizing the structure and content of the Web, it is necessary to establish precise semantics for Web concepts.

5.2.1 Measuring the Web

- Internet and particular web is dynamic in nature so it is difficult task to measure. Fig. 5.2.1 shows number of web site.
- Web explosion is due in no small part to the extended application of an axiom known as Moore's Law. While ostensibly a prediction about semi-conductor innovation rates, this bit of prophecy from Intel co-founder Gordon Moore has come to represent the doubling not just of processing power, but of computing power in general.

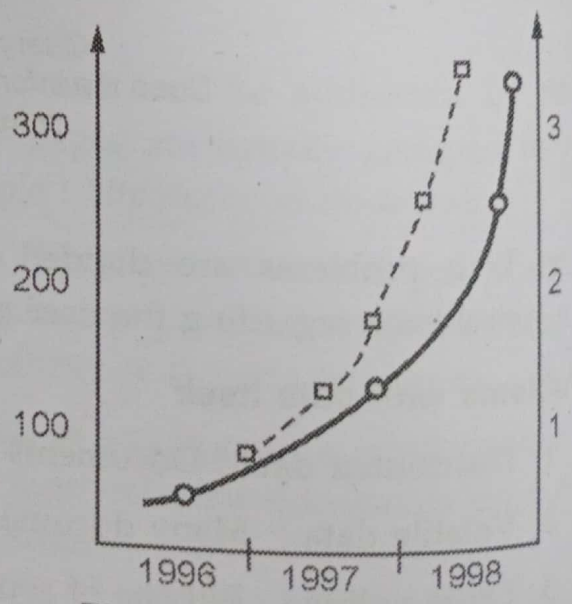


Fig. 5.2.1 Growth of web

- In the mid 1980s, you might spend all afternoon visiting your friends before dropping by the bank and grocery, and then go out to dinner and a show after.

Today, your banking, shopping and chat with your friends are all readily handled from your tablet or phone, and if you're not in the mood for a fancy outing, Netflix and a quick Internet-ordered pizza take care of the evening's entertainment needs and all of it can be accomplished in less time than it takes to say "The Internet made me a hermit."

- Of course, back then, it was something of a privilege to be online, but the net gained traction really quickly. While just 16 million people were using the web in 1995, this figure had leapt to 50 million three years later, one billion by 2009 and more than twice that last year.
- In 1993 there was an estimated 130 websites online, which jumped to ten thousand by 1996. Fast-forward to 2012, and there are 634 million websites competing for your attention.

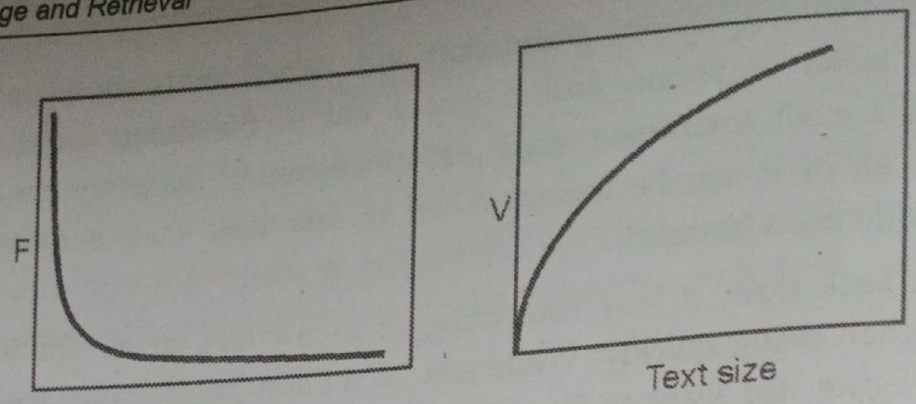
Web pages :

1. 634 million Number of websites (December-2012).
2. 51 million Number of websites added during the year.

- Popular formats for web documents are HTML, followed by GIF and JPG, ASCII text, and Postscript in that order. The most compression tools used are GNU zip, Zip and Compress. The top ten most referenced sites are Microsoft, Netscape, Yahoo, Google and top US universities. An average page has between five and 15 hyperlinks and most of them are local.
- If we assume that the average HTML pages have 5 kB and that there are 300 million web pages, we have at least 1.5 terabytes of text.

5.2.2 Modeling the Web

- Can we model the document characteristics of the whole web ? **Answer is "Yes".**
- The Heap's and Zipf's laws are also valid in the web. Normally the vocabulary grows faster and the word distribution should be more biased. But there are no such experiments on large Web collections to measure these parameters.
- **Heaps' law :** An empirical rule which describes the vocabulary growth as a function of the text size.
- It establishes that a text of n words has a vocabulary of size $O(n^{\beta})$ for $0<\beta<1$
- **Zipf's law :** An empirical rule that describes the frequency of the text words. It states that the i-th most frequent word appears as many times as the most frequent one divided by $i^{-\theta}$, for some $\theta>1$.



(a) Distribution of sorted word frequencies

(b) Size of the vocabulary

Fig. 5.2.2

- One more model is related to the distribution of document size. According to this model, the document sizes are self similar, they have a large variance. The probability of finding a document of size x bytes is given by :

$$p(x) = \frac{1}{x_\sigma \sqrt{2\pi}} \exp -(\ln x - \mu)^2 / 2\sigma^2$$

where the average (μ) and standard deviation (σ) are 9.357 and 1.318 respectively.

- The majority of the documents are small, but there is a non trivial number of large documents. This is intuitive for image or video files, but it is also true for HTML pages. A good fit is obtained with the Pareto distribution

$$p(x) = \frac{\alpha\ k^\alpha}{x^{1+\alpha}}$$

where x is measured in bytes and k and α are parameters of the distribution.

- So what languages dominate the Web ? It should come as no surprise that English still dominates the Web, with more than two-thirds of the Web's pages being in English. According to a study by, a Web site in the language Catalan, Japanese is the second most popular language of Web sites.

Web pages by language		
Language	**Web pages**	**Percent of total**
English	214,250,996	68.39
Japanese	18,335,739	5.85
German	18,069,744	5.77
Chinese	12,113,803	3.87

French	9,262,663	2.96
Spanish	7,573,064	2.42
Russian	5,900,956	1.88
Italian	4,883,497	1.56
Portuguese	4,291,237	1.37
Korean	4,046,530	1.29
Dutch	3,161,844	1.01
Sweden	2,929,241	0.93

- As you can see, the growth is impressive and unimpeded. Also :
  a. The total number of web sites seems to follow Moore's Law and double every 18-24 months.
  b. At the current rate, we will hit 1 billion sites in 2013 and 2 billion sites in 2015.
  c. Over the years, the number of web sites seems to be roughly equal to the number of people on the internet.
  d. If WordPress continues on its current trajectory, there will be 300-500 million WordPress sites by 2015.

5.3 Search Engines

SPPU : May-19

- Search engines are becoming the primary entry point for discovering web pages. Ranking of web pages influences which pages users will view. Exclusion of a site from search engines will cut off the site from its intended audience. The privacy policy of a search engine is important.
- In this section, we discuss the different architecture of retrieval system that model the web as a full text database.
- Search engines are among the most important applications or services on the web. Most existing successful search engines use a centralized architecture and global ranking algorithms to generate the ranking of documents crawled in their databases, for example, Google's PageRank.
- A search engine is a program designed to help find information stored on a computer system such as the World Wide Web or a personal computer. The search engine allows one to ask for content meeting specific criteria and retrieves a list of references that match those criteria.

5.3.1 Centralized Architecture

- Centralized crawler indexer architecture is used by most of the search engine. Using *crawlers*, information is gathered into a single site, where it is indexed; the site then processes all user queries.
- This system has its own data collecting mechanism, and all the data are stored and indexed in a conventional database system. Although many web search engines download web pages and provide service by thousands of servers, they all belong to this kind according to their basic architecture.
- Centralized architecture consists of following components :
  1. Crawlers 2. Index 3. Query engine 4. User interface.

1. Crawlers

- Crawlers are programs i.e. software agents that traverse the web sending new or updated pages to a main server where they are indexed. It sends requests to Web sites and downloads Web pages. The local system sends requests for Web pages to remote Web servers.
- In the downloaded documents it discovers new pages and new servers. The effect of such repeated requests is of a "robot" that moves from site to site, gathering information at every site it visits. In reality, this "crawler" never leaves the local system.
- Crawlers are also called robots, spiders, wanderers and walkers.
- Fig. 5.3.1 shows the software architecture of a search engine based on Altavista architecture.
- Figure consists of two parts : One deal with users, consisting of the user interface and the query engine and another that consists of the crawler and indexer modules.

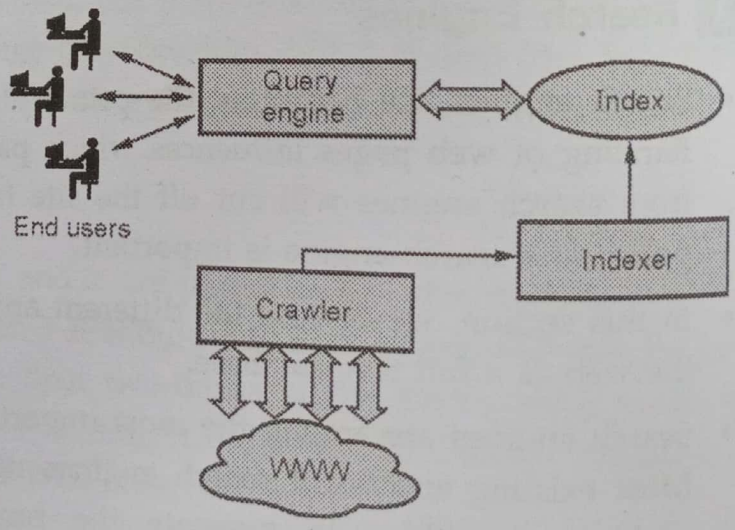


Fig. 5.3.1 Typical crawler indexer architecture

2. Indexer

- The index is used in a centralized fashion to answer queries submitted from different places in the web. Index the downloaded pages. Each downloaded page is processed locally.
- The indexing information is saved and the page is discarded.

- A module that takes a collection of documents or data and builds a searchable index from them. Common practices are inverted files, vector spaces, suffix structures and hybrids of these.
- *Exception :* Some search engines keep a local cache copy of many popular pages indexed in their database, to allow for faster access and in case the destination server is temporarily in-accessible.

3. **User interface :** Solicit queries and deliver answers. All requests are submitted to a single site.

4. **Query engine :** It process queries against the index. All processing is done locally. It requires a massive array of computers and storage.

- Following table gives idea about search engine with URL and web page indexed upto May 1998.

Search engine	URL	Web page indexed
AltaVista	www.altavista.com	140
AOL Netfind	www.aol.com/netfind/	-
Excite	www.excite.com	55
Google	google.stanford.edu	25
GoTo	goto.com	-
HotBot	www.hotbot.com	110
Infoseek	www.infoseek.com	30
Lycos	www.lycos.com	30
Magellan	www.mckinley.com	55
Microsoft	search.msn.com	-
northernLight	www.nlsearch.com	67
WebCrawler	www.webcrawler.com	2

Problems using this architecture :

1. Difficult to gathering the data because of dynamic nature of the Web.
2. The high load on Web servers.
3. Large volume of data : Could be difficult for crawler to cope with Web growth
4. Communication link problem.

- Most of the search engines are based in the United States and focuses on document in English. Some search engines specialized in different countries and different languages. Some of the search engines retrieve only specific web pages such as personal or institutional home pages or specific objects such as electronic mail address, images etc.

5.3.2 Distributed Architecture

- When the data source is large enough that even the metadata can't be efficiently managed in a database system, we can choose distributed system. Distributed information retrieval system has no its own actual record database. It just indexes the interface of sub database system.
- When receiving a query from a user, main system will instantly obtain the records from sub databases through their search interfaces. The limitation of this system is that the number of sub databases can't be many, otherwise the search speed can't be ensured. A famous system is InfoBus system in Stanford digital library project.
- Harvest is an example of distributed architecture.

Harvest

- Harvest is a distributed crawler-indexer architecture which addresses the main problems in crawling and indexing the Web :
  1. Web servers get requests from different crawlers of search engines which increase the server's load;
  2. Most of the entire objects retrieved by the crawlers are useless and discarded;
  3. No coordination exists among the crawlers.
- Harvest is designed to be a distributed search system where machines work together to handle the load which a single machine could not handle. Harvest also can be used to save bandwidth by deploying gatherers near the data source and exchanging the summarized data which usually is much smaller than the original data.
- But it seems most of further Harvest applications are in the field of caching Web objects instead of providing advanced internet search services. State of the art indexing techniques can reduce the size of an inverted file to about 30 % of the size of the text.
- Fig. 5.3.2 shows harvest architecture.

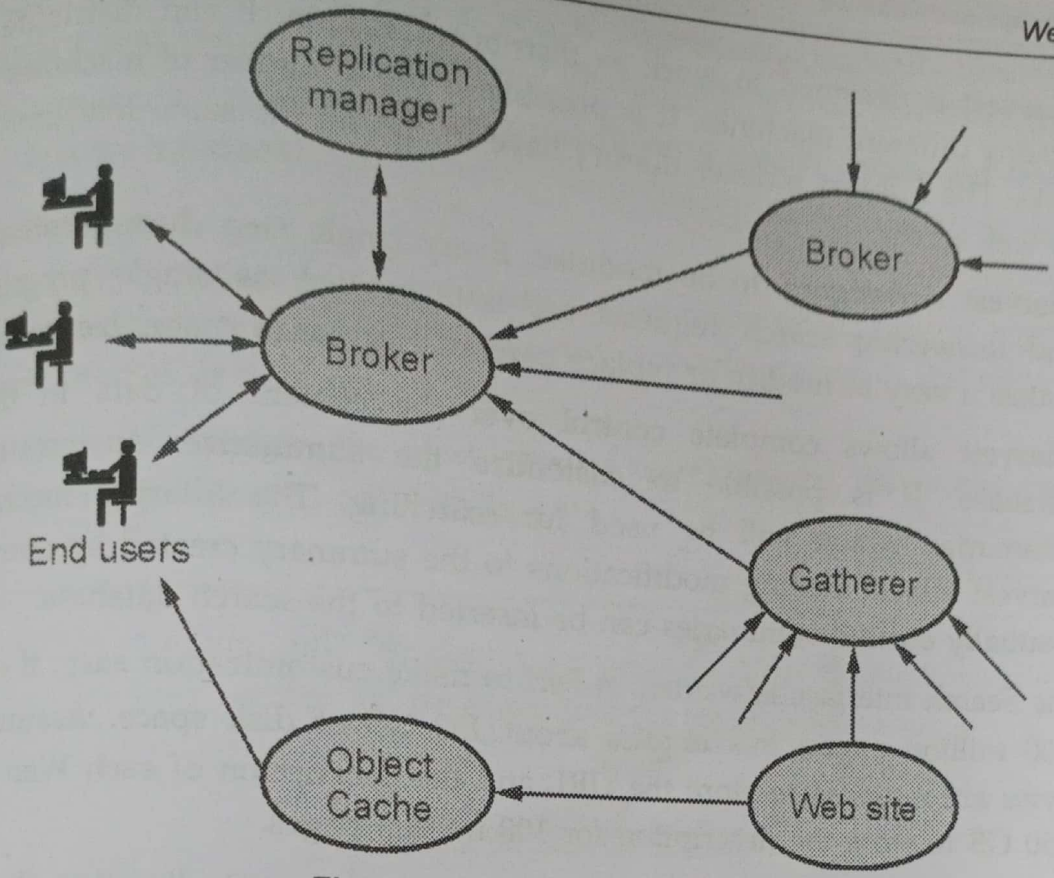


Fig. 5.3.2 Harvest architecture

Goals of harvest :

- One of the goals of the Harvest is to build topic specific brokers, focusing the index contents and avoiding many of the vocabulary and scaling problems of generic indices. It includes a distinguished broker that allows other brokers to register information about gathers and brokers.
- This is useful to search for an appropriate broker or gatherer when building a new system.

Components :

1. Gatherers : It extracts information from the documents stored on one or more Web servers. It can handle documents in many formats like HTML, PDF, Postscript, etc.
2. Broker : Provides the indexing mechanism and query interface.
3. Replicator : To replicate servers. It enhances user base scalability.
4. Object cache : Reduces network and server load. It also reduces response latency when accessing Web pages.

Features of harvest :

- Harvest is a modular, distributed search system framework with working set components to make it a complete search system. The default setup is to be a web search engine, but it is also much more and provides following features :

1. Harvest is designed to work as distributed system. It can distribute the load among different machines. It is possible to use a number of machines to gather data. The full-text indexer doesn't have to run on the same machine as broker or web server.
2. Harvest is designed to be modular. Every single step during collecting data and answering search requests are implemented as single programs. This makes it easy to modify or replace parts of Harvest to customize its behaviour.
3. Harvest allows complete control over the content of data in the search database. It is possible to customize the summarizer to create desired summaries which will be used for searching. The filtering mechanism of Harvest allows making modifications to the summary created by summarizers. Manually created summaries can be inserted to the search database.
4. The Search interface is written in Perl to make customization easy, if desired.

- For 100 million pages, this implies about 150 GB of disk space. Assuming that 500 bytes are required to store the URL and the description of each Web page, we need 50 GB to store the description for 100 million pages.
- The use of Meta search engines is justified by coverage studies that show that a small percentage of Web pages are in all search engines. Moreover, less than 1 % of the Web pages indexed by AltaVista, HotBot, Excite and Inforseek are in all of those search engines.

Advantages of distributed architecture :

1. *Server load reduced :* A gatherer running on a server reduces the external traffic (i.e., crawler requests) on that server.
2. *Network traffic reduced :* Crawlers retrieve entire documents, whereas Harvest moves only summaries.
3. *Redundant work avoided :* A gatherer sending information to multiple brokers reduces work repetition.

5.3.3 User Interface

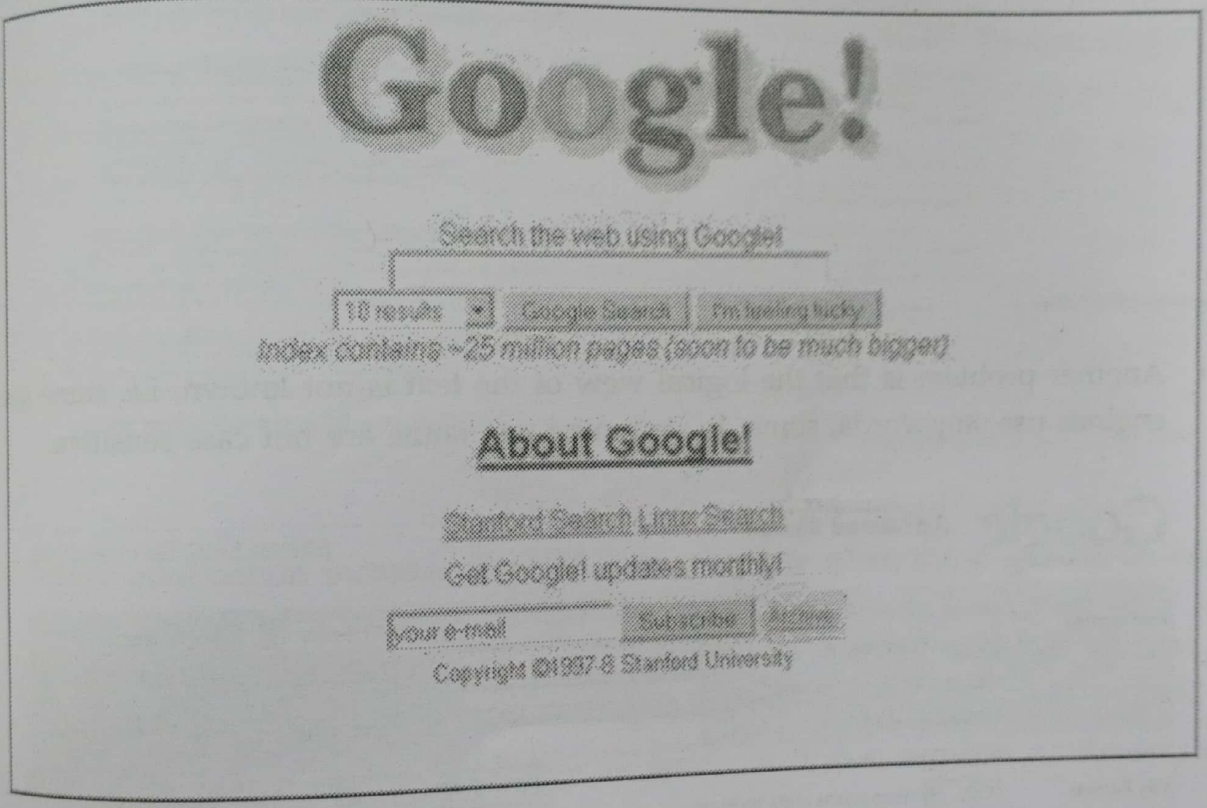
- The job of the search user interface is to aid users in the expression of their information needs, in the formulation of their queries, in the understanding of their search results, and in keeping track of the progress of their information seeking efforts.
- Search engines have a user interface for expressing the query and for displaying the answers. The interface is tightly coupled with the retrieval software and it varies from system to system. Given this scenario the user is not allowed to

choose a user interface and a search engine independently. In other words, searching and visualizing information are not independent components.

- Two important aspects of the user interfaces of search engine are query interface and **answer interface.**

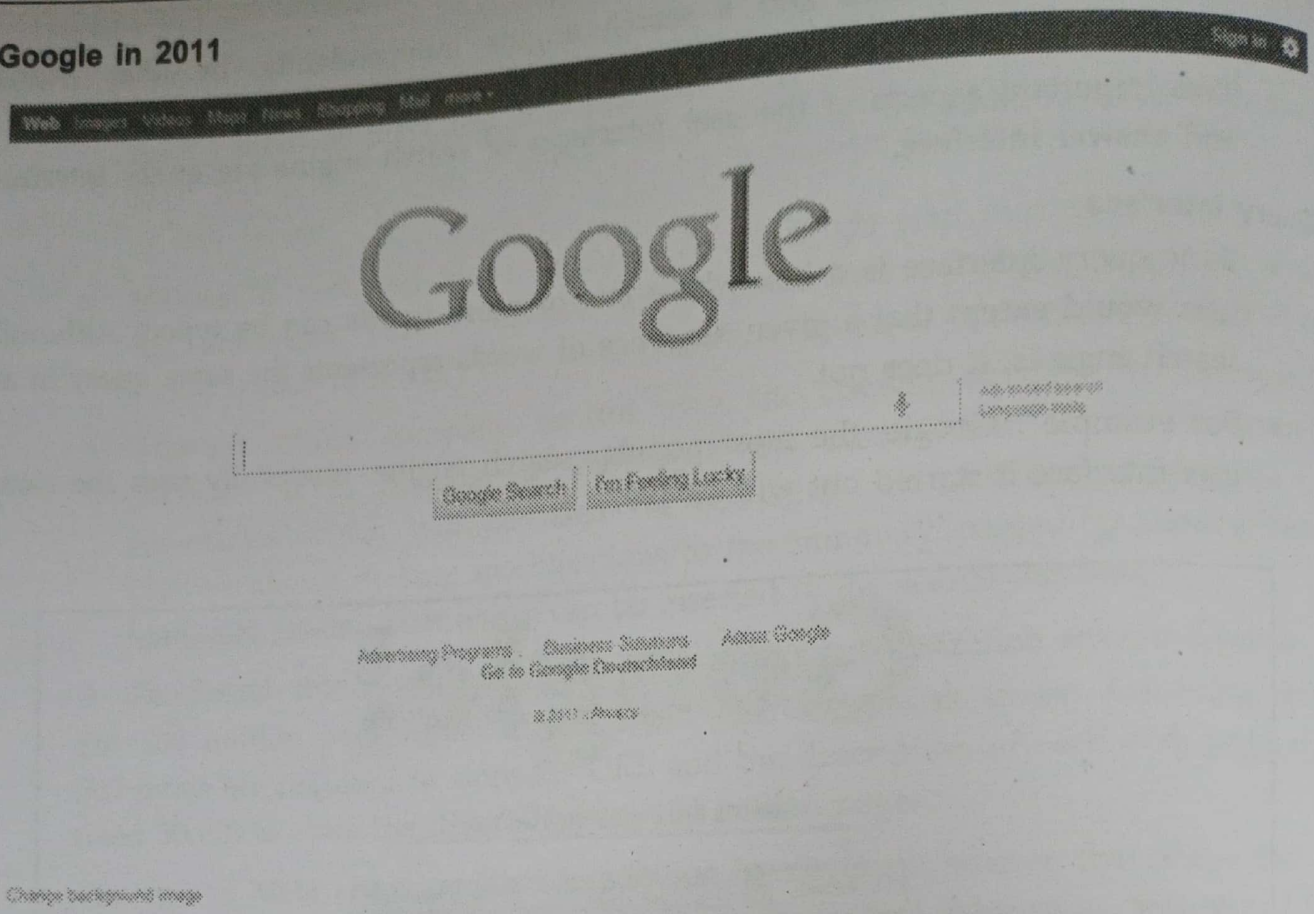
Query interface

- Basic query interface is a box where one or more words can be typed. Although user would except that a given sequence of words represents the same query in all search engines, it does not.
- For example : Google, the most popular search engine, essentially uses the same user-interface it started out with in 1997.

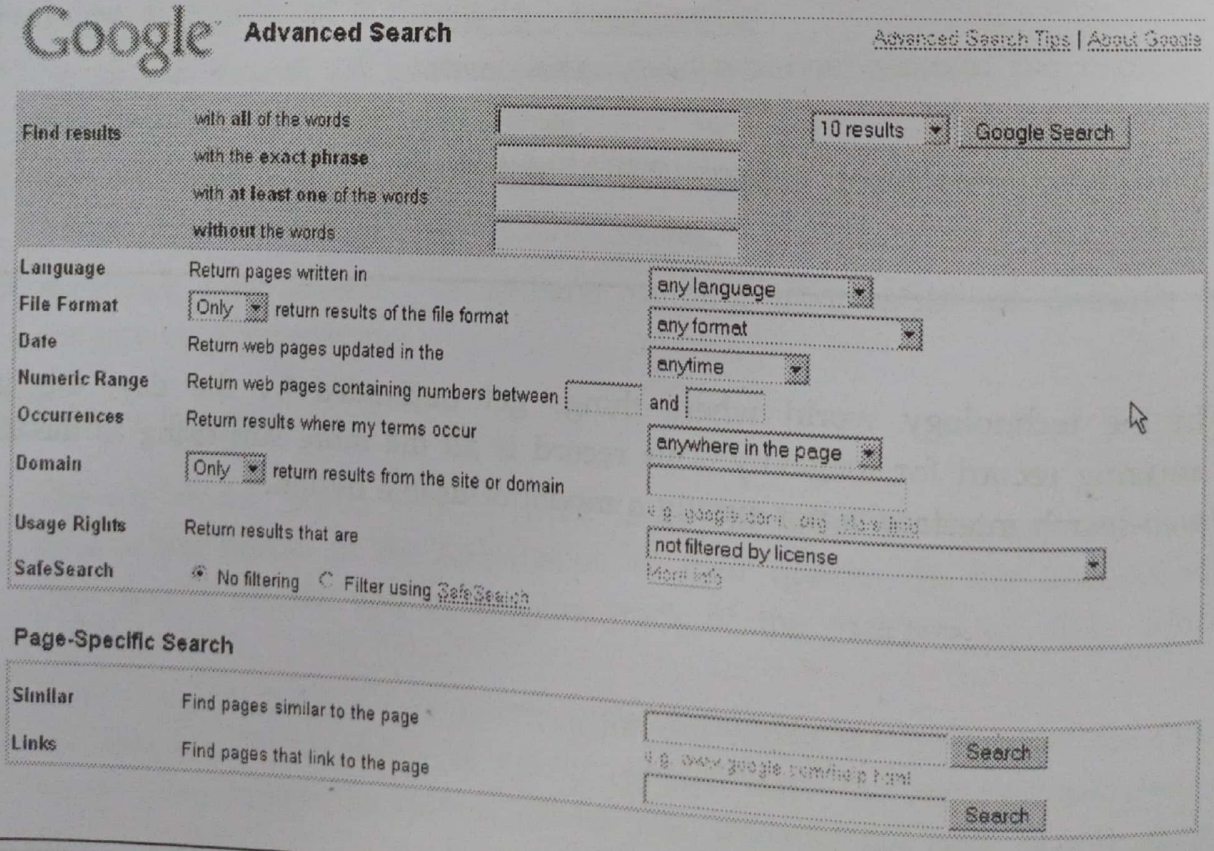


- In the technology world where things get outmoded by the day, this is an amazing record for longevity. This record is all the more surprising as the classic web-search interface is not exactly a model of usable design.

Google in 2011

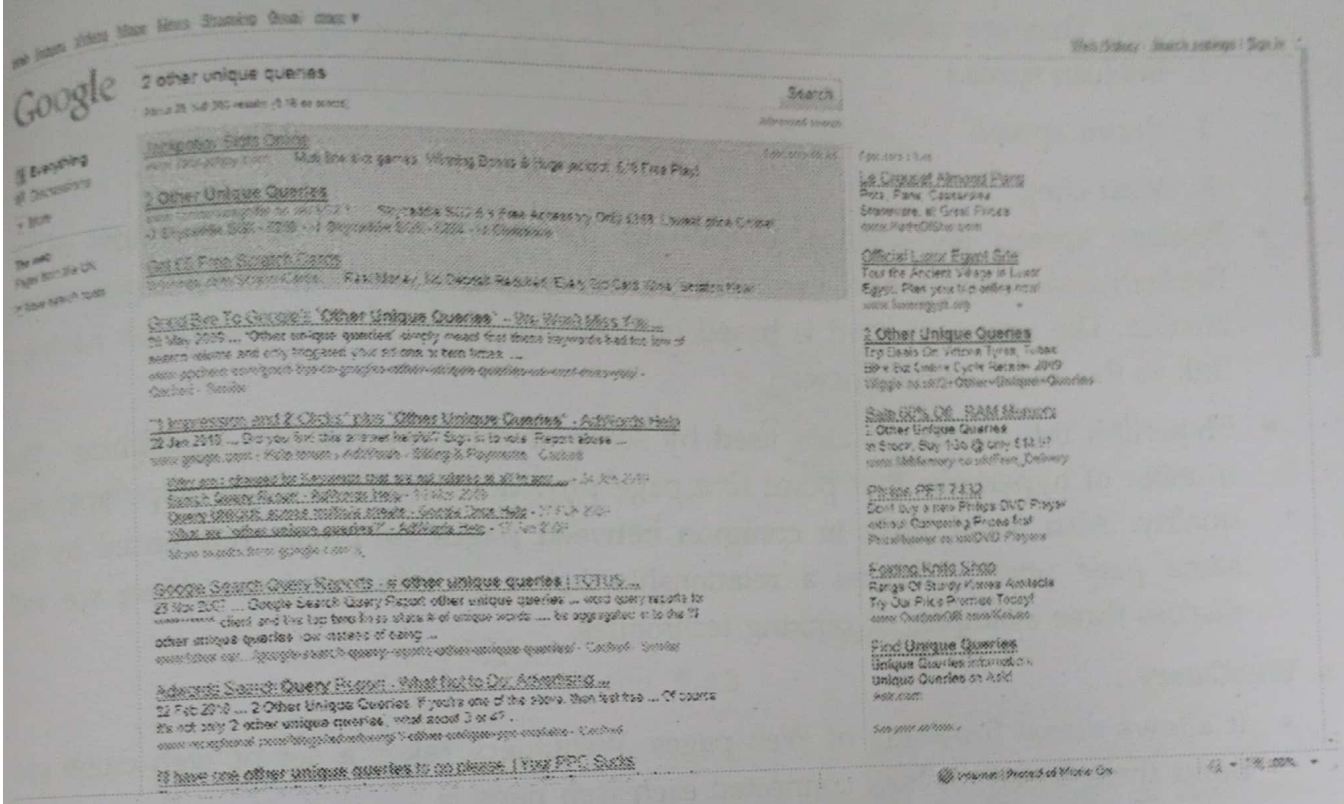


- Another problem is that the logical view of the text is not known, i.e. some search engines use stopwords, some do stemming and some are not case sensitive.



Answer interface

- Answer usually consists of a list of the ten top ranked web pages. Following figure shows Google search engine for the query searching with answer interface. Each entry in this list includes some information about the document it represents.



- Typical information includes the URL, size, the date when the page was indexed, and the couple of lines with its contents. Some of the search engines allow the user to change the number of pages returned in the list and the amount of information per page, but in most cases this is fixed or limited to a few choices.
- Most of the search engines also have an option to find documents similar to each web page in the answer.

5.3.4 Ranking

- The primary challenge of a search engine is to return results that match a user's needs. A word will potentially map to millions of documents. How to order them ?
- Most of the search engines use variations of the Boolean or vector model to do ranking. Every search engine strives to return relevant web pages that will satisfy your requests. Each search engine uses a proprietary 'ranking algorithm' that attempts to instantly build a list of highly appropriate responses to your query.

- Since each search engine applies its own formula to a unique database of information, results and relevancy rankings will always vary from search engine to search engine.
- Yuwono and Lee propose three ranking algorithms in addition to the classical scheme. They are called
  1. Boolean spread
  2. Vector spread
  3. Most cited
- Boolean spread and vector spread are the normal ranking algorithms of the Boolean and vector model extended to include pages pointed to by a page in the answer. The last most cited is based only on the terms included in pages having a link to the pages in the answer.
- Hyperlink information is also used by some of the new ranking algorithms. The number of hyperlinks that point to a page provides measures of its popularity and quality. Also many links in common between pages or pages referenced by the same page often indicates a relationship between those pages. Here we will discuss three examples of ranking techniques.

a. WebQuery :

- It allows visual browsing of Web pages. WebQuery takes a set of web pages and ranks them based on how connected each web page is.

b. HITS algorithm :

- Algorithm developed by Kleinberg in 1998.
- Second method is used in Hypertext Induced Topic Search (HITS). This ranking scheme depends on the query and considers the set of pages S that point to or are pointed by pages in the answer.
- Pages that have many links pointing to them in S are called authorities. Pages that have many outgoing links are called hubs. Gives each page a hub score and an authority score. A good authority is pointed to by many good hubs. A good hub points to many good authorities. Users want good authorities.
- Computes hubs and authorities for a particular topic specified by a normal query. First determines a set of relevant pages for the query called the base set S. Analyze the link structure of the web sub-graph defined by S to find authority and hub pages in this set.
- Hubs : It contains many outward links and lists of resources.
- Authorities : It contains many inward links and provides resources, content.

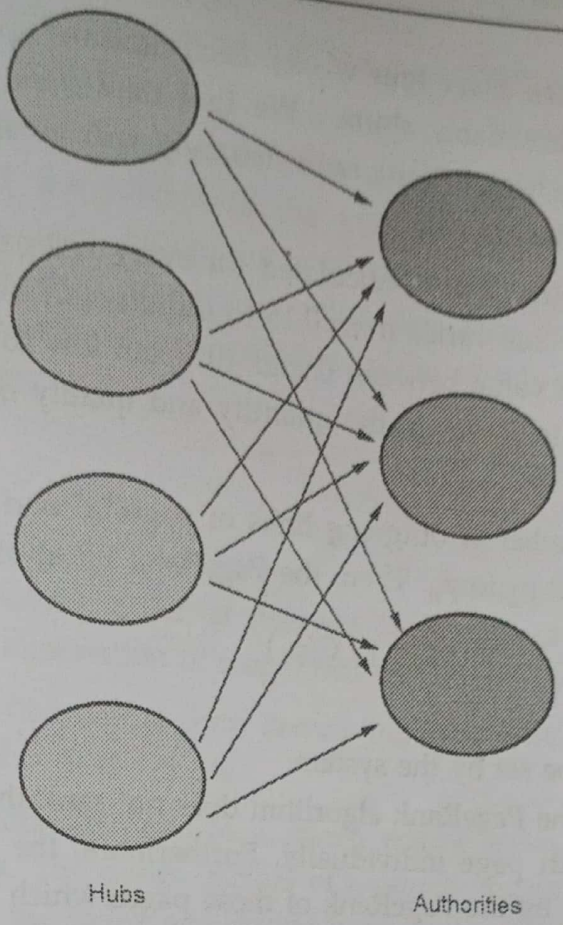


Fig. 5.3.3

- Let H(p) and A(p) be the hub and authority value of pages p. These values are defined such that the following equations are satisfied for all pages p. Authorities are pointed to by lots of good hubs :

$$a_p = \sum_{q:q \to p} h_q$$

- Hubs point to lots of good authorities:

$$h_p = \sum_{q:p \to q} a_q$$

C. PageRank

- Numeric value to measure how important a page is. PageRank (PR) is the actual ranking of a page, as determined by Google.
- A probability is expressed as a numeric value between 0 and 1. A 0.5 probability is commonly expressed as a "50 % chance" of something happening. Hence, a PageRank of 0.5 means there is a 50 % chance that a person clicking on a random link will be directed to the document with the 0.5 PageRank.
- Purpose : To increase the quality of search results and has left all of its competitors for dead.
- The Google Page Rank is based on how many links you have to and from your pages, and their respective page rank.

- We update our index every four weeks. Each time we update our database of web pages, our index invariably shifts : We find new sites, we lose some sites, and sites ranking may change. Your rank naturally will be affected by changes in the ranking of other sites.
- The Google PageRank (PR) is calculated for every webpage that exists in Google's database. It's real value varies from 0,15 to infinite, but for representation purposes it is converted to a value between 0 and 10 (from low to high). The calculation of the PR for a page is based on the quantity and quality of web pages that contain links to that page.
- Let C(a) be the number of outgoing links of page "a" and suppose that page "a" is pointed to by pages p_1 to p_n. Then, the PageRank PR(a) of "a" is defined as

$$PR(a) = 1+(1-q)\sum_{i=1}^{n} PR(p_i) / C(p_i)$$

Where q must be set by the system.

- It is obvious that the PageRank algorithm does not rank the whole website, but it's determined for each page individually. Furthermore, the PageRank of page A is recursively defined by the PageRank of those pages which link to page A.
- PageRank can be computed using an iterative algorithm. This means that each page is assigned an initial starting value and the PageRanks of all pages are then calculated in several computation circles based on the equations determined by the PageRank algorithm. The iterative calculation shall again be illustrated by the three-page example, whereby each page is assigned a starting PageRank value of 1.

5.3.5 Crawling the Web

- A web crawler (also known as a web spider) is a program which browses the World Wide Web in a methodical, automated manner. Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine that will index the downloaded pages to provide fast searches.
- It starts with a list of URLs to visit. As it visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, recursively browsing the Web according to a set of policies.
- Web crawlers are mainly used to create a copy of all the visited pages for later processing by a search engine, which will index the downloaded pages to provide fast searches. Crawlers can also be used for automating maintenance tasks on a Web site, such as checking links or validating HTML code. Also, crawlers can be

used to gather specific types of information from Web pages, such as harvesting e-mail addresses.

- When a search engine's web crawler visits a web page, it "reads" the visible text, the hyperlinks, and the content of the various tags used in the site, such as keyword rich Meta tags. Using the information gathered from the crawler, a search engine will then determine what the site is about and index the information. The website is then included in the search engine's database and its page ranking process.

5.3.6 Indices

- Most indices use variants of the inverted file. An inverted file is list of sorted words, each one having a set of pointers to the pages where it occurs. Some search engines use elimination of stopwords toreduce the size of the index.
- Index is complemented with a short description of each Web page. It includes title, creation date and size, first line etc.
- There are many different parts to a search engine index, such as design factors and data structures. The design factors of a search engine index design or outline the architecture of the index and decide how the index actually works. The parts all combine to create the working search engine index, and include: Merge factors, which decide how the information enters the index, deciding whether the data is new data or data that is being updated.
- Index size, which pertains to the amount of computer space necessary to support the index. Storage techniques, which is the decision of how the information should be stored. Larger files are compressed while smaller files are simply filtered.
- Inverted files can also point to the actual occurrences of a word within a document. However, it is too costly in space for the Web, because each pointer has to specify a page and a position inside the page. Currently some search engines are providing phrase searches, but the actual implementation is not known.
- Finding words which start with a given prefix requires two binary searches in the sorted list of words. Most complex searches, like words with errors, arbiratory wild cards or any regular expression on a word can be performed by doing a sequential scan over the vocabulary.
- Pointing to pages or to word positions is an indication of the granularity of the index. The index can be less dense if we point to logical blocks instead of pages. This reduces the variance of the different document sizes, by making all blocks roughly the same size. This not only reduces the size of the pointers but also reduces the numbers of pointers because words have locality of reference.

- When a search engine index is being built, there are also many different types of data structures to choose from. Choosing a particular data structure for a search engine index is like deciding on a particular form for a web page and depends on the factors that the search engine will serve. These data structures can be :
  a. Suffix tree : Supports linear time lookup and is structured like a Tree.
  b. Tree : An ordered tree data structure that stores an associative array where the keys are strings.
  c. Inverted index : Stores a list of occurrences in the form of a hash table or a binary tree.
  d. Citation index : Stores citations or hyperlinks between certain documents to support citation analysis.
  e. N-gram index : Stores sequences of length of data, which supports other types of retrieval. Sometimes supports text mini too.
  f. Term document matrix : This is used in latent semantic analysis. A term document matrix stores the occurrences of words in documents in a two-dimensional sparse matrix.
- These different types come together with architecture and build a search engine index that is ready to use and quickly returns the results that the visitor is looking for.
- For example, the word "civil" might occur in documents 3, 8, 22, 56, 68 and 92, while the word "war" might occur in documents 2, 8, 15, 22, 68 and 77.
- Suppose someone comes to Google and types in civil war. In order to present and score the results, we need to do two things :
  1. Find the set of pages that contain the user's query somewhere
  2. Rank the matching pages in order of relevance

Civil	3	8		22	56	68	92
War	2	8	15	22		68	77
Both words		8		22		68	

University Question

1. *What is web crawling ? Explain techniques used by web crawlers to crawl the web.*

SPPU : May-19, End Sem, Marks 8

5.4 Browsing

5.4.1 Web Directories

- Web directory is a classification of Web pages by subject. One of best and oldest Web directory is Yahoo, which is most used searching tool. eBLAST, LookSmart, Magellan and NewHoo are the example of Web directories. Some of them are hybrid in nature.
- Web Directories also called catalogs, yellow pages or subject directories. Following is some of the Web directories with their URL.

Search engine	URL	Web sites	Categories
eBLAST	www.eblast.com	125	-
LookSmart	www.looksmart.com	300	24
Lycos Subjects	a2z.lycos.com	50	-
Magellan	www.mckinley.com	60	-
NewHoo	www.newhoo.com	100	23
Netscape	www.netscape.com	-	-
Search.com	www.search.com	-	-
Snap	www.snap.com	-	-
Yahoo	www.yahoo.com	750	-

Principles :

- Classification is by a hierarchical taxonomy.
- Directory may be specific to a subject, a region, a language.
- Pages are submitted and reviewed before they are included.
- Automatic classification is not successful enough.
- Some subcategories are also available in the main page of Web directories.
- **Advantage :** If found, the answer will be useful in most cases;
- **Disadvantage :**
  1. Classification is not specialized enough;
  2. Not all Web pages are classified;

5.4.2 Combining Searching and Browsing

- The two paradigms of searching and browsing are currently almost always used separately. Software tool called WebGlimpse that combines the two paradigms.
- Browsing and searching are the two main paradigms for finding information on line. The search paradigm has a long history; search facilities of different kinds are available in all computing environments. The browsing paradigm is newer and less ubiquitous, but it is gaining enormous popularity through the World-Wide Web.
- Both paradigms have their limitations. Search is sometimes hard for users who do not know how to form a search query so that it is limited to relevant information. Browsing can make the content come alive, and it is therefore more satisfying to users who get positive reinforcement as they proceed. However, browsing is time-consuming and users tend to get disoriented and lose their train of thoughts and their original goals.

WebGlimpse works as follows :

- At indexing time, it analyzes a given WWW archive, computes neighborhoods, adds search boxes to selected pages, collects remote pages when relevant and caches those pages locally.
- Once indexing is done, users who browse that site can search from any of the added search boxes and limit their search to the neighborhood of that page.
- Glimpse's index consists essentially of two parts : The **word file** and the **pointers file.** The word file is simply a list of all the words in all documents, each followed by an offset into the pointers file. The pointers file contains for each word a list of pointers into the original text where that word appears.
- A search typically consists of two stages :
  1. First, the word file is searched and all relevant offsets into the pointers file are found. The relevant pointers in the pointers file are collected.
  2. The second stage is another search, using agrep, in the corresponding places in the original text. This is similar in principle to the usual inverted indexes, except that the word file, being one relatively small file, can be searched sequentially.
- This allows glimpse to support very flexible queries, including approximate matching, matching to parts of words and regular expressions. These flexible queries are implemented by running agrep directly on the word file. The fact that the files are searched directly allows the user to decide on-the-fly how much of the match to see.

- Glimpse's default is to show one line per match, but it can also show one paragraph or any user-defined record. This gives context to every match.

5.5 Meta-searchers

- Metasearcher : Web server that sends a given query to several search engines and Web directories and collects the answers and unifies them.
- Examples : Metacrawler, Savvysearch, MetaSearch, Mamma.
- **Advantages :**
  1. Combine the results of many sources.
  2. Save users from the need to pose queries to multiple searchers.
  3. Ability to sort the results by different attributes.
  4. Pages retrieved by multiple searchers are more relevant.
  5. Improve coverage : Individual searchers cover a small fraction of the Web.
- A metasearcher submits queries over multiple sources. But the interfaces and capabilities of these sources may vary dramatically. Even the basic query model that the sources support may vary. Some search engines only support the Boolean retrieval model.
- In this model, a query is a condition that documents either do or do not satisfy. The query result is then a set of documents. For example, a query distributed and a system returns all documents that contain both the words distributed and systems in them.
- Metasearch engines present the results of their searches in one of two ways :
  1. **Single list :** Most metasearcher display multiple-engine search results in a single merged list, from which duplicate entries have been removed.
  2. **Multiple lists :** Some metasearchers do not collate multiple-engine search results but display them instead in separate lists as they are received from each engine. Duplicate entries may appear.
- Following is the list of metasearcher with their URL.

Metasearcher	URL	Source Used
Cyber411	www.cyber411.com	14
Dogpile	www.dogpile.com	25
Highway61	www.highway61.com	5
Inference Find	www. infind.com	6

Mamma	www. mamma.com	7
MetaCrawler	www.metacrawler.com	7
MetaFind	www. metafind.com	7
MetaMiner	www. miner.uol.com.br	13
MetaSearch	www.metasearch.com	-
SavvySearch	www.savvy.cs.colostate.edu:2000	>13

- Metacrawler is a search engine which cluster retrieved web documents. A metasearch engine is a search tool that sends user requests to several other search engines and/or databases and aggregates the results into a single list.

Step 1 : When MetaCrawler receives a query, it posts the query to multiple search engines in parallel.

Step 2 : Performs sophisticated pruning on the responses returned.

- The search results you receive are a combination of the top commercial (sponsored advertising) and non-commercial (algorithmic) results from the most popular search engines on the Web.
- Meta Crawler's metasearch technology combines the top ranking search results from each of the separate search engines based on your specific query. The blend of sponsored and non-sponsored results for a given search term depends on the nature of the term, but sponsored results are always clearly identified with a "Sponsored" or similar designation like this".
- **Features :**
  1. Unifies the search Interface and provides a consistent user interface,
  2. Standardizes the query structure,
  3. May make use of an independent ranking method for the results {rank-merging},
  4. May have an independent ranking system for each search engine/database it searches,
  5. Meta search is not a search for meta data.
- User interface : Normally resemble search engine interfaces with options for types of search and search engines to use.
- Dispatcher : Generates actual queries to the search engines by using the user query. It may involve choosing/expanding search engines to use.
- Display : Generates results page from the replies received. It may involve ranking, parsing, clustering of the search results or just plain stitching.

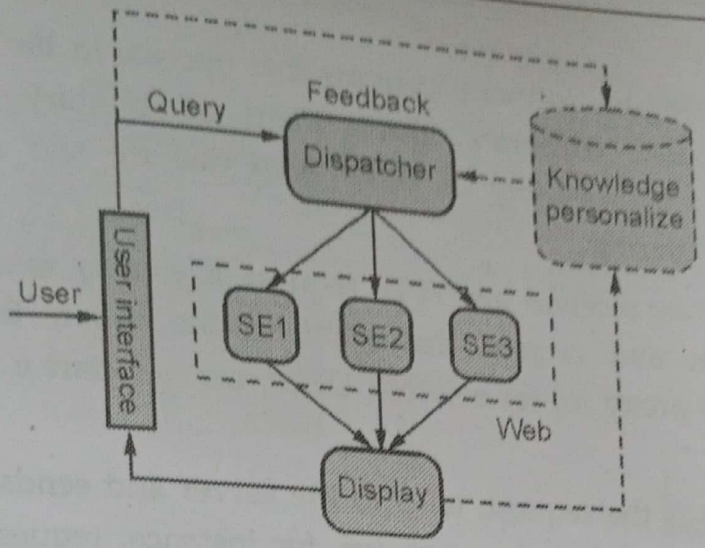


Fig. 5.5.1

- Personalization/Knowledge : It may contain either or both. Personalization may involve weighting of search results/query/engine for each user.

5.6 Searching using Hyperlinks

Web query languages

- Web query languages require knowledge of the Web sites and the language syntax. They are hard to use.
- Query is based on content of each page. The power of the Web resides in its capability of redirecting the information flow via hyperlinks, so it should appear natural that in order to evaluate the information content of a Web object, the Web structure has to be carefully analyzed.
- Recent experiments seem to confirm that hyperlinks can be very valuable in locating or organizing information. They have been used :
  a. To improve an initial ranking of documents
  b. To compute an estimate of a Web page's popularity
  c. To find the most important hubs and authorities for a given topic.
- It seems that each of these techniques is based on some underlying, sometimes partly implicit idea, on the type of knowledge which can be induced by the presence of a hyperlink between two pages.
- For example, when they are used to estimate the popularity of a Web page, hyperlinks can be interpreted in the following way : "if a document is cited by a popular document, then it is possibly popular itself". This type of knowledge can easily be captured by propositional logic, if some measure of uncertainty is associated.

- Companies use web harvesting tools to automate collecting the data needed to make business decisions. Using a web scraper can automate this entire process, freeing up time for your employees to spend on other things. It also ensures that the data is collected accurately and without human errors.
- By using web scrapers, companies can quickly collect information regarding competitors' prices to ensure their products are priced competitively. Businesses can also monitor brand sentiment by collecting all mentions of your brand and seeing what people are saying about you or your products.

5.8.1 Merits and Demerits

1. Merits

- Automates collecting data
- Can collect a vast amount of data
- It can run in the background without human interaction required

2. Demerits

- Can be misused
- Can be banned when not using a proxy

5.8.2 Challenges while Scraping at Scale

1. **Data warehousing :** Data extraction at a large scale generates vast volumes of information. Fault-tolerant, scalability, security, and high availability are the must-have features for a data warehouse.
2. **Honeypot traps :** Some websites have honeypot traps on the webpages for the detection of web crawlers. They are hard to detect as most of the links are blended with background color or the display property of CSS is set to none.
3. **Anti-scraping technologies :** Web scraping is a common thing these days, and every website host would want to prevent their data from being scraped. Anti-scraping technologies would help them in this.
4. **Pattern changes :** Scraping heavily relies on user interface and its structure, i.e., CSS and Xpath.
5. **Captchas :** Captchas is a good way of keeping crawlers away from a website and it is used by many website hosts.

5.8.3 Python for Web Scraping

- Python has become one of the most popular web scraping languages due in part to the various web libraries that have been created for it. One popular library,

Beautiful Soup, is designed to pull data out of HTML and XML files by allowing searching, navigating, and modifying tags.

- To scrape a website with python we're generally dealing with two types of problems : collecting the public data available online and then parsing this data for structured product information.
- Steps to perform web scraping :

1. **Step -1 :** Find the URL that we want to scrape

- Find the requirement of data according to project. A webpage or website contains a large amount of information. That's why scrap only relevant information.

2. **Step - 2 :** Inspecting the page

- The data is extracted in raw HTML format, which must be carefully parsed and reduce the noise from the raw data.

3. **Step - 3 :** Write the code

- Write a code to extract the information, provide relevant information, and run the code.

4. **Step - 4 :** Store the data in the file

5.8.4 Working with HTML

- HTML specifies the contents and layout of web pages. The content contains text, table, form, image, links, information for search engine, etc. The layout is in the form of text format, background and frame. HTML is also used to specify links and which resources are associated with them.
- Hyper Text Markup Language (HTML) is intended as a common medium for typing together information from widely different sources.
- HTML documents are the Standard Generalized Markup Language (SGML) documents with generic semantic that are appropriate for representing information from a wide range of applications.
- HTML documents are in plain text format that contain embedded HTML tags. Documents can be created in any text editor. There are also many other tools, including editors, designed specifically to assist in creating HTML documents. To view an HTML document, the user needs a browser.
- HTML defines the structural elements in a document such as headers, and addresses, layout information and the use of inline graphics together with the ability to provide hyper text links. Web pages were written in HTML level 0.

5.8.4.1 Parsing XML and HTML

- XML stands for eXtensible Markup Language. XML is a markup language for documents containing structured information. XML is a set of rules for structuring, storing and transferring information.
- Define a website with simple the HTML for a table. It is loaded into BeautifulSoup and parse it, returning a pandas data frame of the contents.

```
import pandas as pd
from bs4 import BeautifulSoup

html_string = '''
  <table>
     <tr>
       <td> Hello! </td>
       <td> Table </td>
     </tr>
   </table>
'''

soup = BeautifulSoup(html_string, 'lxml') # Parse the HTML as a string

table = soup.find_all('table')[0] # Grab the first table

new_table = pd.DataFrame(columns=range(0,2), index = [0]) # I know the size

row_marker = 0
for row in table.find_all('tr'):
   column_marker = 0
   columns = row.find_all('td')
   for column in columns:
      new_table.iat[row_marker,column_marker] = column.get_text()
      column_marker += 1

new_table
```

5.8.4.2 Using XPath for Data Extraction

- XPath specifies path expression that matches XML data by navigating down the tree. XPath is used as the embedded query language for both XQuery 1.0 and XSLT 2.0.
- XPath provides a common syntax and semantics for functionality shared between XSLT and XPointer. XPath is used in XSL transformations to find information in an XML document. It is used to navigate through elements and attributes in XML documents.

- Parsers are represented by parser objects. There is support for parsing both XML and (broken) HTML.

```
from pandas.io.parsers import TextParser
def parse_options_data(table):
rows = table.findall('.//tr')
header = _unpack(rows[0], kind='th')
data = [_unpack(r) for r in rows[1:]]
return TextParser(data, names=header).get_chunk()
```

3.8.4.3 Dealing with Unicode

- Unicode is not an encoding. Unicode is more like a big database of characters and properties and rules, and on its own says nothing about how to encode a sequence of characters into a sequence of bytes for actual use by a computer.
- Unicode doesn't work in terms of characters; it works in terms of things called code points.
- Python 2 comes with two different kinds of objects that can be used to represent strings, str and unicode.
- Since Python 3.0, all strings are stored as Unicode in an instance of the str type. Encoded strings on the other hand are represented as binary data in the form of instances of the bytes type.

5.8.4.4 Stemming and Removing Stop Words

- Stemming is the process of reducing words to their root word.
- Tokenization : The process of segmenting text into words, clauses or sentences.
- Stemming : Reducing related words to a common stem.
- Removal of stop words : Removal of commonly used words unlikely to be useful for learning.
- The act of stemming and removing stop words simplifies the text and reduces the number of textual elements so that just the essential elements remain.
- Stemming is as follows :

```
import nltk
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
#is based on The Porter Stemming Algorithm
stopword = stopwords.words('english')
snowball_stemmer = SnowballStemmer('english')
text = "This is a Demo Text for NLP using NLTK. Full form of NLTK is Natural Language
Toolkit"
word_tokens = nltk.word_tokenize(text)
stemmed_word = [snowball_stemmer.stem(word) for word in word_tokens]
```

```
print (stemmed_word)
[OUTPUT]: ['this', 'is', 'a', 'demo', 'text', 'for', 'nlp', 'use', 'nltk', '.', 'full', 'form', 'of', 'nltk', 'is',
'natur', 'languag', 'toolkit']
```

- Let's look at an example.

```
from nltk.stem import PorterStemmer
stemming = PorterStemmer()

my_list = ['frightening', 'frightened', 'frightens']

# Using a Python list comprehension method to apply to all words in my_list

print ([stemming.stem(word) for word in my_list])


Out: ['frighten', 'frighten', 'frighten']
```

Removing stop words

- 'Stop words' are commonly used words that are unlikely to have any benefit in natural language processing. These includes words such as 'a', 'the', 'is'.

```
from nltk.corpus import stopwords
stops = set(stopwords.words("english"))

def remove_stops(row):
    my_list = row['stemmed_words']
    meaningful_words = [w for w in my_list if not w in stops]
    return (meaningful_words)

imdb['stem_meaningful'] = imdb.apply(remove_stops, axis=1)
```

5.8.5 Beautiful Soup

- Beautiful Soup is a Python library that is used to pull data of HTML and XML files. It is mainly designed for web scrapping. It works with the parser to provide a natural way of navigating, searching, and modifying the parse tree. The latest version of Beautiful Soup is 4.8.1.
- We use the requests library to fetch an HTML page and then use the BeautifulSoup to parse that page.
- Beautiful Soup is a Python library for pulling data out of HTML and XML files. Beautiful Soup helps you pull particular content from a webpage, remove the HTML markup and save the information. It is a tool for web scraping that helps you clean up and parse the documents you have pulled down from the web.

- Importing the Beautiful Soup constructor function

```
from bs4 import BeautifulSoup
```

- The BeautifulSoup constructor function takes in two string arguments :
  a. The HTML string to be parsed.
  b. Optionally, the name of a parser.
- So, let's parse some HTML :

```
from bs4 import BeautifulSoup
htmltxt = "<p> Hello World </p>"
soup = BeautifulSoup(htmltxt, 'lxml')
```

- The BeautifulSoup object has a text attripute that returns the plain text of a HTML string. In the example of simple soup of Hello World , the text attribute returns :

```
soup.text
# 'Hello World'
```

❑❑❑

Unit VI

6 Advanced Information Retrieval

Syllabus

XML Retrieval : *Basic XML concepts, Challenges in XML retrieval, Vector space model for XML retrieval, Evaluation of XML retrieval, Text-Centric vs. Data-Centric XML retrieval.* ***Recommendation system :*** *Collaborative Filtering and Content Based Recommendation of Documents and Products. Introduction to Semantic Web*

Contents

6.1 Basic XML Concept

- XML stands for eXtensible Markup Language. It is emerging as a standard for exchanging data on the Web. It enables separation of content (XML) and presentation (XSL).
- The XML standard was created by W3C to provide an easy to use and standardized way to store self describing data.
- XML is a markup language in a standard plain text format. It contains structured or semi-structured data in verbose user-defined tags presented in a hierarchical way (tree-like structure).
- XML is not a replacement for HTM and traditional databases. XML documents are used either as a container to store semi-structured data or a media to exchange data between heterogeneous application.
- XML can be used to provide more information about the structure and meaning of the data in the Web pages rather than just specifying how the Web pages are formatted for display on the screen.
- XML provides the ability to structure, optionally validate and transform data, allowing it to be used across various applications in a platform independent manner.
- The term "Extensible" refers to the capability of being extended while the phase "Markup Language" refers to the set of conventions used for encoding textual information.

6.1.1 XML Document

- XML documents are text based. The basic object in XML is the XML document. XML documents, including XHTML ones, must be well-formed. These document is a labeled, unranked, ordered tree :
  1. Labeled means that some annotation, the label, is attached to each node.
  2. Unranked means that there is no a priori bound on the number of children of a node.
  3. Ordered means that there is an order between the children of each node.
- There are two main structuring concepts that are used to construct an XML document : **Elements and Attributes.**
  - Attributes in XML provide additional information that describes elements.
- Complex elements are constructed from other elements hierarchically, whereas simple elements contain data values. Within an XML document, each element

needs to be marked (or tagged) in some manner similar to HTML. XML elements are made up of a start tag, an end tag and information, or content between them.

- An XML document may also be valid. A valid document is checked against either a Document Type Definition (DTD) or a schema. The following is a very simple XML document.

```
<?xml version = "1.0" ?>
< address >
<name>Rakshita Dhotre</name>
<email> rdhotre@gmail.com<ma</email>
<phone>020-45-6789</phone>
<birthday>1993-07-13</birthday>
</address>
```

- The first line is a processing instruction. It begins with '<?' and indicates the version of xml that is used in the document.

Tree Structure :

- An XML document exhibits a tree structure. It has a single root node, <address> in the example above. The tree is a general ordered tree. There is a first child, a next sibling etc.
- Nodes have parents and children. There are leaf nodes at the bottom of the tree. The declaration at the top is not part of the tree, but the rest of the document is.
- In the tree representation, internal nodes represent complex elements, whereas leaf nodes represent simple elements. That is why the XML model is called a tree model or a **hierarchical** model.
- An XML document can be modeled as an ordered, labeled tree, with a document node serving as the root node.
- An XML document can be modeled as an ordered labeled tree. Each node in this tree corresponds to an element in the document and is labeled with the element tag name. Each edge in this tree represents inclusion of the element corresponding to the child node under the element correponding to the parent node in the XML file.
- Now <name> has two children and <birthday> has three. Most processing on the tree is done with a preorder traversal. One way to view the tree is shown in Fig. 6.1.1.

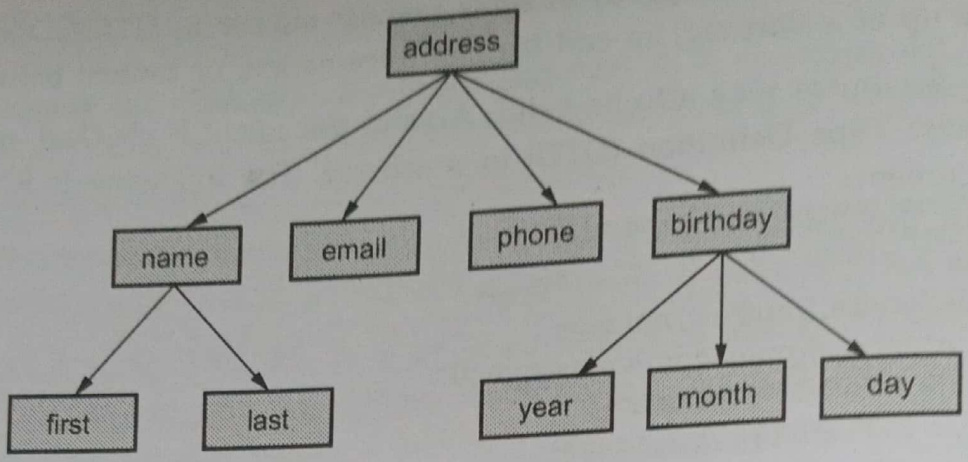


Fig. 6.1.1 XML document tree

```
<XML version = "1.0"? >
    <address>
    <name>
        <first>Rakshita</first>
        <last>Dhotre</last>
</name>
        <email>rdhotre>@gamil.com</email>
        <phone>020-45-6789</phone>
        <birthday>
            <year>2002</year>
            <month>09</ month>
            <day> 29 </day>
    </birthday>
</address>
```

- XML documents are self-describing, thus XML provides a platform independent means to describe data and therefore, can transport data from one platform to another. XML documents can be created and used by applications.

6.1.2 XML Syntax

- All XML elements must have a closing tag. XML tags are case sensitive. All XML elements must be properly nested.
- All XML documents must have a root tag and attribute values must always be quoted. With XML, white space is preserved.
- With XML, a new line is always stored as LF.
- Case sensitivity : The names of XML-elements are case-sensitive. That means the name of the start and the end elements need to be exactly in the same case.
- For example <Permanent _Address> is different from < permanent _address>.

XML Tags and Rules :

- All the markup languages use tags, names enclosed by angle brakets. Tages in XML tend to be spelled out, such as *<name> Rakshita</name >or< telephone>020-45-6789</telephone>.*
- The naming requirements are similar to those in many computer languages. Tags are case sensitive and along with letters, digits and underscores, names may include hyphens, periods and a single colon.

Some of the rules for XML tags are :

1. Opening tags all have matching closing tags. Empty tags such as *
* and *<input>* that have no closing tags are to be written as *
* and *<input/>*.
2. Attribute values (such as text or size) must be in quotes.
3. Tags must be nested. That means that ... is correct but ... is not.
4. Comments contain double hyphens (<!-- ....-->), and no double hyphens are allowed insided comments otherwise.
5. Values must be added to boolean attributes, e.g. multiple ="multiple".
6. The entities < and amp; must be used in place of <, less than and & ampersand.

- An XML element is made up of a start tag, an end tag and data in between. XML tags are case-sensitive.
- An attribute is a name-value pair separated by an equal sign (=). For example :

```
<City PIN = "411041" VadgaonBk </City>
```

- Attributes are used to attach additional, secondary information to an element.
- Data defined through XML documents can optionally use a component known as a Data Type Definition (DTD), Extensible - Data reduction or an XML Schema Definition (XSD) to describe the data and provide meaning by describing and enforcing the data tag structure for an XML document, laying the foundation for the concept of a document type within XML.

6.2 XML Retrieval

- Documents can be structured or unstructured. Unstructured documents have no fixed pre - defined format, whereas structured documents are usually organized according to a fixed pre - defined structure.
- An example of a structured document is a book organized into chapters, each with sections made of paragraphs and so on.

- Nowadays, the most common way to format structured content is with the W3C standard for information repositories and exchanges, the eXtensible Mark-up Language (XML).
- To identify the most useful XML elements to return as answers to given queries, XML information retrieval systems require :

1. Query languages that allow users to specify the nature of relevant components, in particular with respect to their structure.
2. Representation strategies providing a description not only of the content of XML documents, but also their structure.
3. Ranking strategies that determine the most relevant elements and rank these appropriately for a given query.

- XML documents are organized into a logical structure, as provided by the XML mark-up.

Query languages

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

6.2.1 Types of Queries

- **Keyword Based Querying :** A query is the formulation of a user information need. Keyword based queries are popular, since they are intuitive, easy to express, and allow for fast ranking. However, a query can also be a more complex combination of operations involving several words.
- **Word Queries :** The most elementary query that can be formulated in a text retrieval system is the word. Some models are also able to see the internal division of words into letters. The division of the text into words is not arbitrary, since words carry a lot of meaning in natural language. The result of word queries is the set of documents containing at least one of the words of the query. The resulting documents are ranked according to the degree of similarity with respect to the query. To support ranking, two common statistics on word occurrences inside texts are commonly used (term frequency and inverse document frequency).

- **Context Queries :** Many systems complement queries with the ability to search words in a given context. Words which appear near each other may signal higher likelihood of relevance than if they appear apart.
- **Boolean Queries :** The oldest way to combine keyword queries is to use boolean operators. A boolean query has a syntax composed of atoms and operators. Atoms are basic queries that retrieve documents and boolean operators work on their operands and deliver sets of documents.

6.2.2 Patterns Matching

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

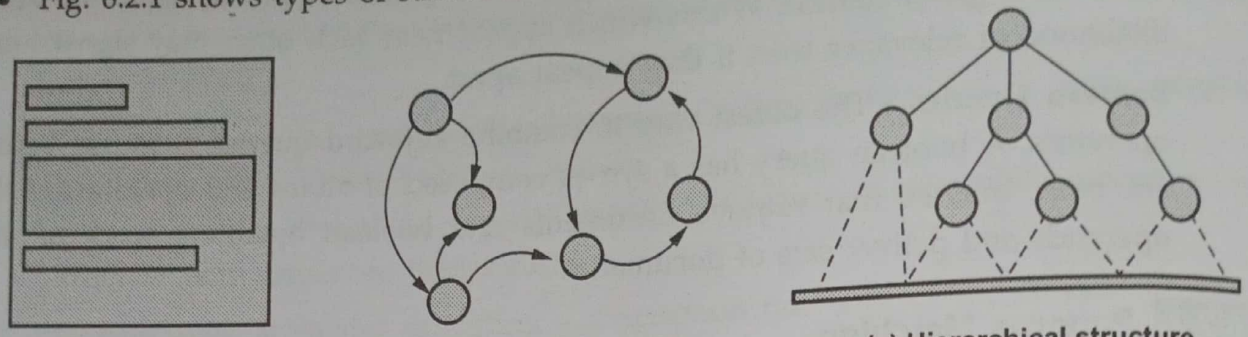
6.2.3 Structural Queries

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

  a) Form-like fixed structure b) Hypertext structure

  c) Hierarchical structure.

- Fig. 6.2.1 shows types of structures.



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

Fig. 6.2.1

6.3 Challenges in XML Retrieval

- The challenge in structured retrieval is that users want to return parts of documents (i.e., XML elements), not entire documents as IR systems usually do in unstructured retrieval.
- If we query Shakespeare's plays for Macbeth's castle, should we return the scene, the act or the entire play. In this case, the user is probably looking for the scene. On the other hand, an otherwise unspecified search for Macbeth should return the play of this name, not a subunit.
- A challenge in XML retrieval related to nesting is that we may need to distinguish different contexts of a term when we compute term statistics for ranking, in particular inverse document frequency (idf) statistics.
- **Second challenge :** Document parts to index
- Central notation for indexing and ranking in IR is documents unit or indexing unit.
  a) In unstructured retrieval, usually straightforward : Files on desktop, email messages, web pages on the web etc.
  b) In structured retrieval, there are four main different approaches to defining the indexing unit. They are non - overlapping pseudo documents, top down, bottom-up and all.
- **Third challenge : Nested elements**
- Because of the redundancy caused by the nested elements it is common to restrict the set of elements eligible for retrieval.
- Restriction strategies include :
  a) Discard all small elements
  b) Discard all element types that users do not look at working XML retrieval system logs.

c) Discard all element types that assessors generally do not judge to be relevant.

d) Only keep element types that a system designer or librarian has deemed to be useful search results.

- In most of these approaches, result sets will still contain nested elements.

6.3.1 Vector Space Model for XML Retrieval

- The dimensions of vector space in unstructured retrieval are vocabulary terms, whereas they are lexicalized subtrees in XML retrieval. There is a tradeoff between the dimensionality of the space and the accuracy of query results.
- If we trivially restrict dimensions to vocabulary terms, then we have a standard vector space retrieval system that will retrieve many documents that do not match the structure of the query. If we create a separate dimension for each lexicalized subtree occurring in the collection, the dimensionality of the space becomes too large.
- Aim : To have each dimension of the vector space encode a word together with its position within the XML tree.
- How : Map XML document to **lexicalized subtree**.
- Take each text node (leaf) and break it into multiple nodes, one for each word. E.g. split Bill Gates into Bill and Gates. Define the dimensions of the vector space to be lexicalized subtrees of documents, subtrees that contain at least one vocabulary term.
- Fig. 6.3.1 shows a mapping of an XML document (left) to a set of lexicalized subtrees (right).

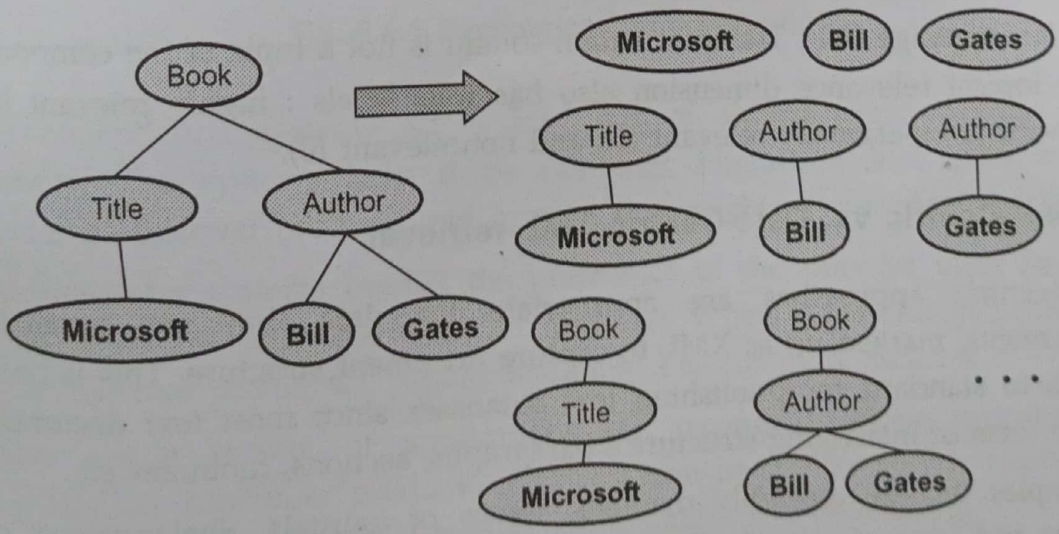


Fig. 6.3.1

- We can now represent queries and documents as vectors in this space of lexicalized subtrees and compute matches between them, e.g. using the vector space formalism.
- Vector space formalism in unstructured VS structured IR : The main difference is that the dimensions of vector space in unstructured retrieval are vocabulary terms whereas they are lexicalized sub-trees in XML retrieval.
- **Structural term :** There is a tradeoff between the dimensionality of the space and accuracy of query results. If we restrict dimensions to vocabulary terms, then we have a standard vector space retrieval system that will retrieve many documents that do not match the structure of the query.
- If we create a separate dimension for each lexicalized subtree occurring in the collection, the dimensionality of the space becomes too large.

6.3.2 Evaluation of XML Retrieval

- INEX 2002 defined component coverage and topical relevance as orthogonal dimensions of relevance. The component coverage dimension evaluates whether the element retrieved is "structurally" correct, that is, neither too low nor too high in the tree. It can be distinguished four cases :
  a) **Exact coverage (E) :** The information sought is the main topic of the component and the component is a meaningful unit of information.
  b) **Too small (S) :** The information sought is the main topic of the component, but the component is not a meaningful (self-contained) unit of information.
  c) **Too large (L) :** The information sought is present in the component but is not the main topic.
  d) **No coverage (N) :** The information sought is not a topic of the component.
- The topical relevance dimension also has four levels : highly relevant (3), fairly relevant (2), marginally relevant (1), and nonrelevant (0).

6.3.3 Text-Centric vs. Data-Centric XML retrieval

- Text-centric approaches are appropriate for data that are essentially text documents, marked up as XML to capture document structure. This is becoming a de facto standard for publishing text databases since most text documents have some form of interesting structure - paragraphs, sections, footnotes etc.
- Examples include assembly manuals, issues of journals, Shakespeare's collected works and newswire articles.
- Data-centric approaches are commonly used for data collections with complex structures that mainly contain non-text data. A text-centric retrieval engine will

have a hard time with proteomic data in bioinformatics or with the representation of a city map that forms a navigational database.

6.4 Recommendation System

- Recommender systems are a way of suggesting like or similar items and ideas to a users specific way of thinking. Recommender systems are widely used on the Web for recommending products and services to users
- Recommender systems try to automate aspects of a completely different information discovery model where people try to find other people with similar tastes and then ask them to suggest new things.
- The goal of a recommender system is to generate meaningful recommendations to a collection of users for items or products that might interest them. Suggestions for books on Amazon, or movies on Netflix, are real-world examples of the operation of industry strength recommender systems.
- Fig. 6.4.1 shows recommendation systems concept.

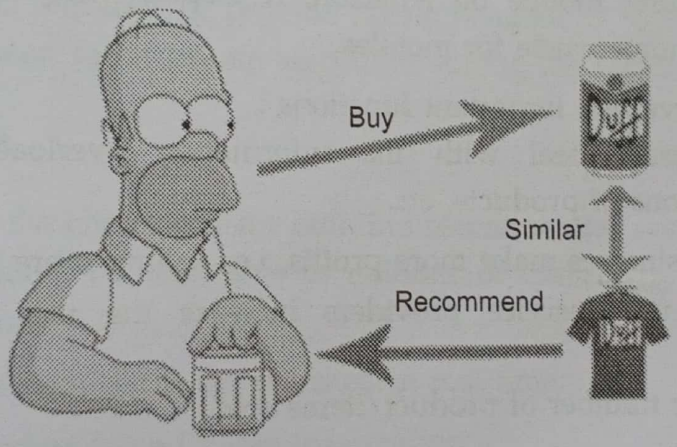


Fig. 6.4.1 Recommendation systems

- Recommendation systems are a key part of almost every modern consumer website. The systems help drive customer interaction and sales by helping customers discover products and services they might not ever find themselves.
- Recommender systems predict the preference of the user for these items, which could be in the form of a rating or response. When more data becomes available for a customer profile, the recommendations become more accurate.
- There are a variety of applications for recommendations including movies (e.g. Netflix), consumer products (e.g., Amazon or similar on-line retailers), music (e.g. Spotify), or news, social media, online dating and advertising.
- Fig. 6.4.2 shows how Amazon uses recommendation concept.

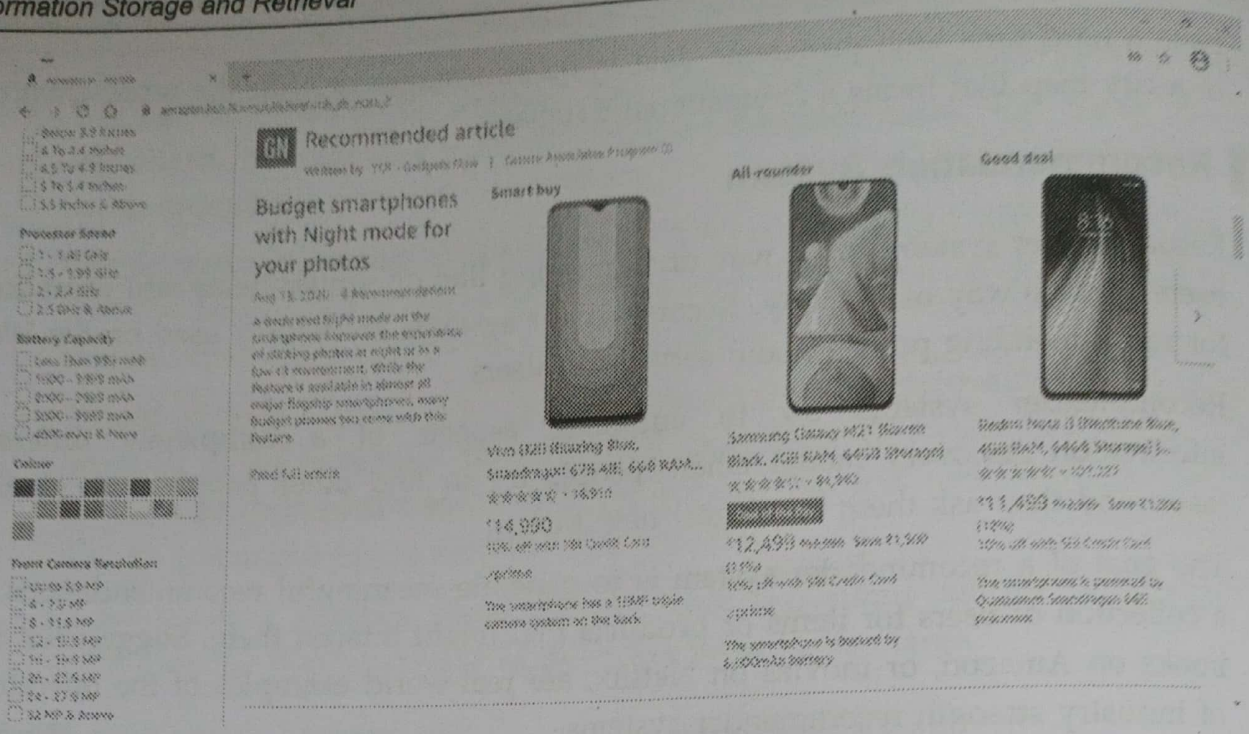


Fig. 6.4.2

- When you searching mobile on Amazon, it display various mobile and it also display recommended article for mobiles.
- These systems serve two important functions :
  1. They help users deal with the information overload by giving them recommendations of products, etc.
  2. They help businesses make more profits, i.e., selling more products
- Various reasons why service providers increase the use of recommendation systems :
  1. It increases the number of product/items sold.
  2. Sell more diverse products/items
  3. User satisfaction is increases
  4. Increase user fidelity
  5. Better understand what the user wants
- The most common scenario is the following :
  a) A set of users has initially rated some subset of movies (e.g., on the scale of 1 to 5) that they have already seen.
  b) These ratings serve as the input. The recommendation system uses these known ratings to predict the ratings that each user would give to those not rated movies by him/her.
  c) Recommendations of movies are then made to each user based on the predicted ratings.

Recommendation process :

- Every recommendation system follows a specific process in order to produce product recommendations.
- The recommendation approaches can be classified based on the information sources they use. Three possible sources of information can be identified as input for the recommendation process. The available sources are the user data (demographics), the item data (keywords, genres) and the user-item ratings.

1. **Collection :** Data collected can be explicit (ratings and comments on products) or implicit (page views, order history, etc.).
2. **Storing :** The type of data used to create recommendations can help user decide the kind of storage we should use- NoSQL database, object storage, or standard SQL database.
3. **Analyzing :** The recommender system finds items with similar user engagement data after analysis.
4. **Filtering :** This is the last step where data gets filtered to access the relevant information required to provide recommendations to the user. To enable this, user will need to choose an algorithm suiting the recommendation system.

6.4.1 Challenges

- Following are the challenges for building recommender systems

1. Huge amounts of data, tens of millions of customers and millions of distinct catalog items.
2. Results are required to be returned in real time.
3. New customers have limited information.
4. Old customers can have a glut of information.
5. Customer data is volatile.

6.4.2 Types of Recommendation Techniques

- In general, there are three types of recommender system:

1. Collaborative recommender system is a system that produces its result based on past ratings of users with similar preferences
2. Content based recommender system is a system that produces its result based on the similarity of the content of the documents or items.
3. Knowledge based recommender system is a system that produces its result based on additional and means-end knowledge.

4. Demographic based recommender system : This type of recommendation system categorizes users based on a set of demographic classes. This algorithm requires market research data to fully implement. The main benefit is that it doesn't need history of user ratings
5. Hybrid recommender systems combine various inputs and different recommendation strategies to take advantage of the synergy among them.

6.5 Collaborative Filtering

SPPU : May-19

- Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a single user by collecting preferences or taste information from many users (collaborating).
- Collaborative filtering (CF) uses given rating data by many users for many items as the basis for predicting missing ratings and/or for creating a top-N recommendation list for a given user, called the active user. Formally, we have a set of users $U = \{u_1, u_2, \ldots, u_m\}$ and a set of items $I = \{i_1, i_2, \ldots, i_n\}$. Ratings are stored in a $m \times n$ user-item rating matrix.
- The problem of collaborative filtering is to predict how well a user will like an item that he has not rated given a set of historical preference judgments for a community of users.

6.5.1 Type of CF

- There are two types of collaborative filtering algorithms : user based and item based.

1. User based

- User-based collaborative filtering algorithms work off the premise that if a user (A) has a similar profile to another user (B), then A is more likely to prefer things that B prefers when compared with a user chosen at random.
- The assumption is that users with similar preferences will rate items similarly. Thus missing ratings for a user can be predicted by first finding a neighborhood of similar users and then aggregate the ratings of these users to form a prediction.
- The neighborhood is defined in terms of similarity between users, either by taking a given number of most similar users (k nearest neighbors) or all users within a given similarity threshold. Popular similarity measures for CF are the Pearson correlation coefficient and the Cosine similarity.
- For example, a collaborative filtering recommendation system for television tastes could make predictions about which television show a user should like given a partial list of that user's tastes (likes or dislikes).

- Note that these predictions are specific to the user, but use information gleaned from many users. This differs from the simpler approach of giving an average score for each item of interest, for example based on its number of votes.
- User-based CF is a memory-based algorithm which tries to mimics word-of-mouth by analyzing rating data from many individuals.
- The two main problems of user-based CF are that the whole user database has to be kept in memory and that expensive similarity computation between the active user and all other users in the database has to be performed.

2. Item-based collaborative filtering

- Item-based CF is a model-based approach which produces recommendations based on the relationship between items inferred from the rating matrix. The assumption behind this approach is that users will prefer items that are similar to other items they like.
- The model-building step consists of calculating a similarity matrix containing all item-to-item similarities using a given similarity measure. Popular are again Pearson correlation and Cosine similarity. All pair-wise similarities are stored in n × n similarity matrix S.
- Item-based collaborative filtering has become popularized due to its use by YouTube and Amazon to provide recommendations to users. This algorithm works by building an item-to-item matrix which defines the relationship between pairs of items..
- When a user indicates a preference for a certain type of item, the matrix is used to identify other items with similar characteristics that can also be recommended.
- Item-based CF is more efficient than user-based CF since the model is relatively small (N × k) and can be fully pre-computed. Item-based CF is known to only produce slightly inferior results compared to user-based CF and higher order models which take the joint distribution of sets of items into account are possible. Furthermore, item-based CF is successfully applied in large scale recommender systems (e.g., by Amazon.com).

6.5.2 Collaborative Filtering Algorithms

1. Memory-based algorithms :

- Operate over the entire user-item database to make predictions.
- Statistical techniques are employed to find the neighbors of the active user and then combine their preferences to produce a prediction.

- Memory-based algorithms utilize the entire user-item database to generate a prediction. These systems employ statistical techniques to find a set of users, known as neighbors that have a history of agreeing with the target user.
- Once a neighborhood of users is formed, these systems use different algorithms to combine the preferences of neighbors to produce a prediction or top-N recommendation for the active user. The techniques, also known as nearest-neighbor or user-based collaborative filtering are more popular and widely used in practice.
- Dynamic structure. More popular and widely used in practice.

Advantages

1. The quality of predictions is rather good.
2. This is a relatively simple algorithm to implement for any situation.
3. It is very easy to update the database, since it uses the entire database every time it makes a prediction.

Disadvantages

1. It uses the entire database every time it makes a prediction, so it needs to be in memory it is very, very slow.
2. Even when in memory, it uses the entire database every time it makes a prediction, so it is very slow.
3. It can sometimes not make a prediction for certain active users/items. This can occur if the active user has no items in common with all people who have rated the target item.
4. Overfits the data. It takes all random variability in people's ratings as causation, which can be a real problem. In other words, memory-based algorithms do not generalize the data at all.

2. Model-based algorithms :

- Input the user database to estimate or learn a model of user ratings, then run new data through the model to get a predicted output.
- A prediction is computed through the expected value of a user rating, given his/her ratings on other items.
- Static structure. In dynamic domains the model could soon become inaccurate.
- Model-based collaborative filtering algorithms provide item recommendation by first developing a model of user ratings. Algorithms in this category take a probabilistic approach and envision the collaborative filtering process as computing the expected value of a user prediction, given his/her ratings on other items.

- The model building process is performed by different machine learning algorithms such as Bayesian network, clustering and rule-based approaches. The Bayesian network model formulates a probabilistic model for collaborative filtering problem.
- The clustering model treats collaborative filtering as a classification problem and works by clustering similar users in same class and estimating the probability that a particular user is in a particular class C and from there computes the conditional probability of ratings.
- The rule-based approach applies association rule discovery algorithms to find association between co-purchased items and then generates item recommendation based on the strength of the association between items

Advantages

1. **Scalability :** Most models resulting from model-based algorithms are much smaller than the actual dataset, so that even for very large datasets, the model ends up being small enough to be used efficiently. This imparts scalability to the overall system.
2. **Prediction speed :** Model-based systems are also likely to be faster, at least in comparison to memory-based systems because, the time required to query the model is usually much smaller than that required to query the whole dataset.
3. **Avoidance of over fitting :** If the dataset over which we build our model is representative enough of real-world data, it is easier to try to avoid over-fitting with model-based systems.

Disadvantages

1. **Inflexibility :** Because building a model is often a time- and resource-consuming process, it is usually more difficult to add data to model-based systems, making them inflexible.
2. **Quality of predictions :** The fact that we are not using all the information (the whole dataset) available to us, it is possible that with model-based systems, we don't get predictions as accurate as with model-based systems. It should be noted, however, that the quality of predictions depends on the way the model is built. In fact, as can be seen from the results page, a model-based system performed the best among all the algorithms we tried.

6.5.3 Advantages and Disadvantages

Advantages

1. Collaborative filtering application is to recommend interesting or popular information as judged by the community.

2. Collaborative filtering system can make more personalized recommendation by analyzing information from your past activity or the history of other users of similar taste.

Disadvantages

1. Many commercial recommender systems are based on large datasets. As a result, the user-item matrix used for collaborative filtering could be extremely large and sparse, which brings about the challenges in the performances of the recommendation.
2. As the numbers of users and items grow, traditional CF algorithms will suffer serious scalability problems.
3. Gray sheep refers to the users whose opinions do not consistently agree or disagree with any group of people and thus do not benefit from collaborative filtering.
4. A collaborative filtering system doesn't automatically match content to one's preferences

University Question

1. *Define Recommender system. Explain in brief Collaborative Filtering.*

SPPU : May-19, End Sem, Marks 8

6.6 Content based Recommendation of Documents and Products

- Content - based recommenders refer to such approaches, that provide recommendations by comparing representations of content describing an item to representations of content that interests the user. These approaches are sometimes also referred to as content - based filtering.
- Content - based recommendation systems try to recommend items similar to those a given user has liked in the past.
- In a movie recommendation application, a movie may be represented by such features as specific actors, director, genre, subject matter, etc.
- The user's interest or preference is also represented by the same set of features, called the user profile.
- Recommendations are made by comparing the user profile with candidate items expressed in the same set of features. The top-k best matched or most similar items are recommended to the user.

- The simplest approach to content - based recommendation is to compute the similarity of the user profile with each item.

6.6.1 High Level Architecture Content-based Recommender Systems

- Fig. 6.6.1 shows high level architecture content-based recommender systems.

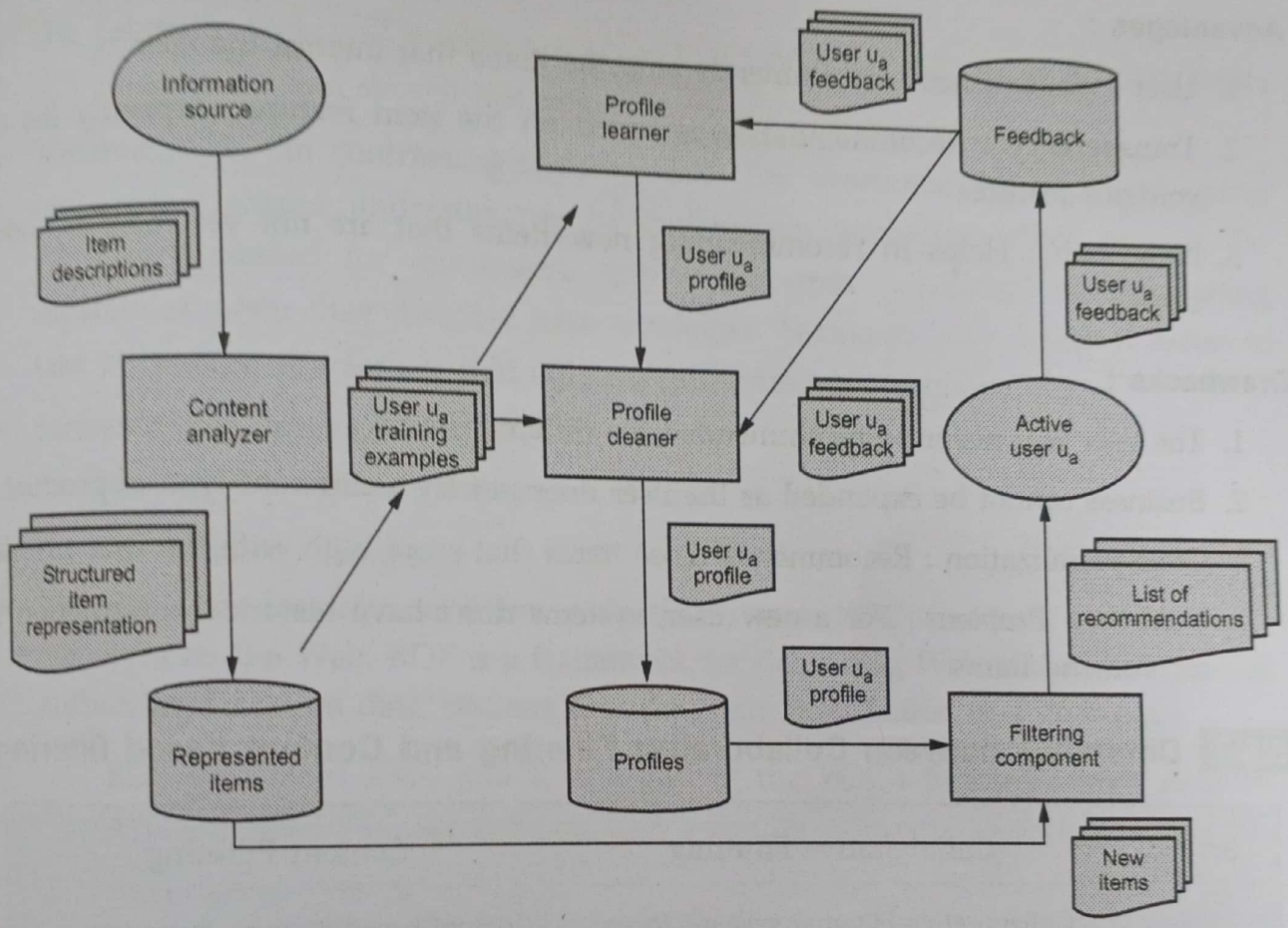


Fig. 6.6.1 High level architecture content-based recommender systems

1. Content Analyzer

- Extracts the features (keywords, n-grams) fromthe source
- Conversion from unstructured to structured item
- Data stored in the repository Represented Items

2. Profile Learner

- To build user profile
- Updates the profile using the data in Feedback repository

3. Filtering Component

- Matching the user profile with the actual item to be recommended

- Uses different strategies
- Users have no detailed knowledge of collection makeup and the retrieval environment. Most users often need to reformulate their queries to obtain the results of their interest.

6.6.2 Advantages and Drawbacks of Content-based Filtering

Advantages :

1. User Independence : Recommends only the items that interest the user
2. Transparency : Recommendation is based on the item features, explicitly list the contents features
3. New Item : Helps in recommending new items that are not yet rated by other users

Drawbacks :

1. The user will never be recommended for different items.
2. Business cannot be expanded as the user does not try a different type of product.
3. Overspecialization : Recommends those items that score high with the user profile
4. Cold Start Problem : For a new user, systems don't have historical information to recommend items

6.6.3 Difference between Collaborative Filtering and Content based filtering

Sr. No.	Collaborative Filtering	Content Filtering
1.	Collaborative-Filtering systems focus on the relationship between users and items.	Content-based systems focus on properties of items.
2.	Example : Netflix movie recommendations	Example : Pandora.com music recommendations
3.	**Pro :** Does not assume access to side information about items	**Con :** Assumes access to side information about items
4.	Cannot recommend new items	It can recommend new items
5.	Item features are inferred from ratings.	Match the item features with user preferences.
6.	**Con :** Does not work on new items that have no ratings	**Pro :** Got a new item to add ? No problem, just be sure to include the side information.

6.7 Introduction to Semantic Web

SPPU : May-19

- The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation.
- Semantic Web challenge : provide language that expresses both data and rules for reasoning about that data.
- The Semantic Web will enable machines to COMPREHEND semantic documents and data, not human speech and writings.
- Semantic Web, in contrast, is more flexible. The consumer and producer agents can reach a shared understanding by exchanging ontologies, which provide the vocabulary needed for discussion. Agents can even "bootstrap" new reasoning capabilities when they discover new ontologies. Semantics also makes it easier to take advantage of a service that only partially matches a request.
- Semantic Web languages today :
  1. Resource Description Framework (RDF)
  2. Web Ontology Language
- The Resource Description Framework (RDF) is a W3C standard for describing resources on the Web. RDF is a framework for describing Web resources, e.g., title, author, modification date, content, and copyright information of a Web page.
  - RDF written in XML and it is a part of the W3C's **Semantic Web** Activity. RDF provides a ***model*** for data, and a ***syntax*** so that independent parties can exchange and use it
  - RDF was designed to provide a common way to describe information so it can be read and understood by computer applications. RDF descriptions are not designed to be displayed on the web.

Why the Semantic Web ?

1. Syntax / semantics distinction : Long history in philosophy of language, linguistics, formal logic
2. Syntax concerned with arrangement of symbols
3. Semantics concerned with the relation between symbols strings and the world : what things actually *mean*.

- Ontologies are a key enabling technology for the Semantic Web. Semantic Web provides an infrastructure that enables not just Web pages, but databases, services, programs, sensors, personal devices and even household appliances to both consume and produce data on the Web.
- Semantic Web technology supports free co-mingling of vocabularies as well as the ad-hoc definition of new relationships to construct data descriptions.

University Question

1. *Explain semantic web in detail.* **SPPU : May-19, End Sem, Marks 8**

❑❑❑

SOLVED MODEL QUESTION PAPER (End Sem)

Information Storage and Retrieval

B.E. (IT) Semester - VII (As Per 2019 Pattern)

[Maximum Marks : 70

Time : $2\frac{1}{2}$ Hours]

N.B. :

1) *Attempt Q.1 or Q.2, Q.3 or Q.4, Q.5 or Q.6, Q.7 or Q.8.*
2) *Neat diagrams must be drawn wherever necessary.*
3) *Figures to the right side indicate full marks.*
4) *Assume suitable data if necessary.*

Q.1 a) *Explain the following : KWIC, TileBar.* **[Refer section 3.4.2]** [8]

b) *Define and explain following term : Precision and recall.* **[Refer section 3.1]** [5]

c) *What is user relevance judgment ? Explain.* **[Refer section 3.5]** [5]

OR

Q.2 a) *Describe briefly interface support for search process.* **[Refer section 3.6]** [8]

b) *What is Boolean queries ? List the problem with Boolean queries.* **[Refer section 3.3.1]** [5]

c) *What is information visualization ? Explain starting point.* **[Refer section 3.2]** [5]

Q.3 a) *What is query languages ? Explain query languages supporting retrieval of multimedia object.* **[Refer section 4.4]** [10]

b) *What is multimedia IR ? Comparison between multimedia information system and traditional system.* **[Refer section 4.2]** [7]

OR

Q.4 a) *Describe the architecture of distributed IR.* **[Refer section 4.1]** [9]

b) *What is data modeling ? Explain the MULTOS data model.* **[Refer section 4.3]** [8]

Q.5 a) *What is web crawling ? Explain techniques used by web crawlers to crawl the web.* **[Refer section 5.3]** [10]

b) *Explain searching using hyperlinks.* **[Refer section 5.6]** [8]

OR

Q.6 a) *What is web scraping ? Explain architecture of web scraping. Explain its merits and demerits.* **[Refer section 5.8]** [10]

b) *What is web directories ? How to combine searching and browsing ?* **[Refer section 5.4]** [8]

Q.7 a) *What is recommendation system ? Explain challenges of recommendation system.* **[Refer section 6.4]** [5]

b) *Compare Text-Centric vs. Data-Centric XML retrieval.* **[Refer section 6.3.3]** [4]

c) *Discuss content based recommendation of documents and products.* **[Refer section 6.6]** [8]

OR

Q.8 a) *What are the challenges in XML retrieval ?* **[Refer section 6.3]** [4]

b) *What is XML ? Explain XML syntax.* **[Refer section 6.1]** [5]

c) *What is collaborative filtering ? Explain collaborative filtering algorithms.* **[Refer section 6.5]** [8]

❑❑❑