



Modern Information Retrieval

Modern information retrieval (Rajiv Gandhi Proudyogiki Vishwavidyalaya)

Modern Information Retrieval Notes

Introduction:

- Information retrieval (IR) is the activity of obtaining information resources relevant to an information need from a collection of information resources.
- The user information need cannot be used directly to request information using the current interfaces of Web search engines. Instead, the user must first translate this information need into a *query* which can be processed by the search engine (or IR system).
- Information retrieval deals with the representation, storage, organization of, and access to information items such as documents, Web pages, online catalogs, structured and semi-structured records, multimedia objects. The representation and organization of the information items should be such as to provide the users with easy access to information of their interest.
- In terms of scope, the area has grown well beyond its early goals of indexing text and searching for useful documents in a collection. Nowadays, research in IR includes modelling, Web search, text classification, systems architecture, user interfaces, data visualization, filtering, languages.

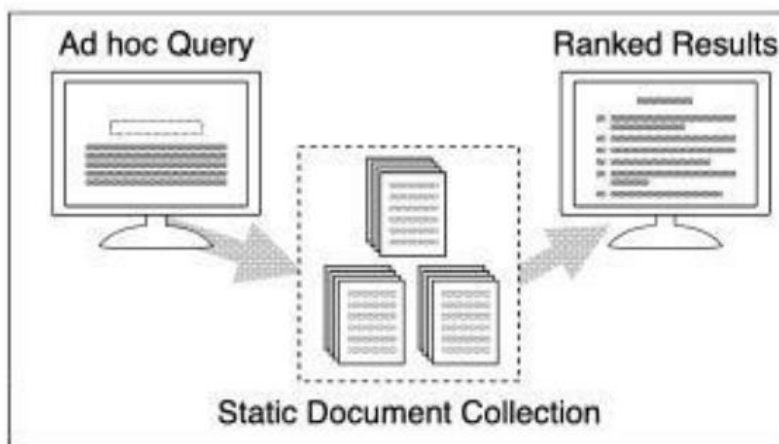


Fig shows basic information retrieval system

Information versus Data Retrieval

- Data retrieval, in the context of an IR system, consists mainly of determining which documents of a collection contain the keywords in the user query which, most frequently, is not enough to satisfy the user information need. In fact, the user of an IR system is concerned more with retrieving information about a subject than with retrieving data which satisfies a given query.
- In data retrieval a single erroneous object means total failure, while in information retrieval the retrieved objects might be inaccurate and small errors are ignored.
- The main reason for this difference is that information retrieval usually deals with natural language text which is not always well structured and could be semantically ambiguous. On the other hand, a data retrieval system (such as a relational database) deals with data that has a well-defined structure and semantics. Data retrieval, while providing a solution to the user of a database system, does not solve the problem of retrieving information about a subject or topic.
- Data retrieving systems can be found in the operating system search. Windows search is a best example for the data retrieval system.
- Search engines are the best example of search engines.

Basis	Data Retrieval	Information Retrieval
Matching	Exact match	Partial match, best match
Inference	Deduction	Induction
Model	Deterministic	Probabilistic
Classification	Monothetic	Polythetic
Query language	Artificial	Natural
Query specification	Complete	Incomplete
Items wanted	Matching	Relevant
Error response	Sensitive	Insensitive

The retrieval Process

Step 1: Before the retrieval process can even be initiated, it is necessary to define the text database. This is usually done by the manager of the database, which specifies the following:

- (a) the documents to be used
- (b) text operations
- (c) the text model

Step 2: Once the logical view of the documents is defined, the database manager builds an index of the text. An index is a critical data structure because it allows fast searching over large volumes of data(e.g. inverted file).

Step 3: Then, the user first specifies a user need which is then parsed and transformed by the same text operations applied to the text. Then, query operations are applied to the actual query which is then processed to obtain the retrieved documents. Fast query processing is made possible by the index structure previously built.

Step 4: Before being sent to the user, the retrieved documents are ranked according to a likelihood of relevance.

Step 5: The user then examines the set of ranked documents in the search for useful information. At this point, he might pinpoint a subset of the documents seen as definitely of interest and initiate a user feedback cycle.

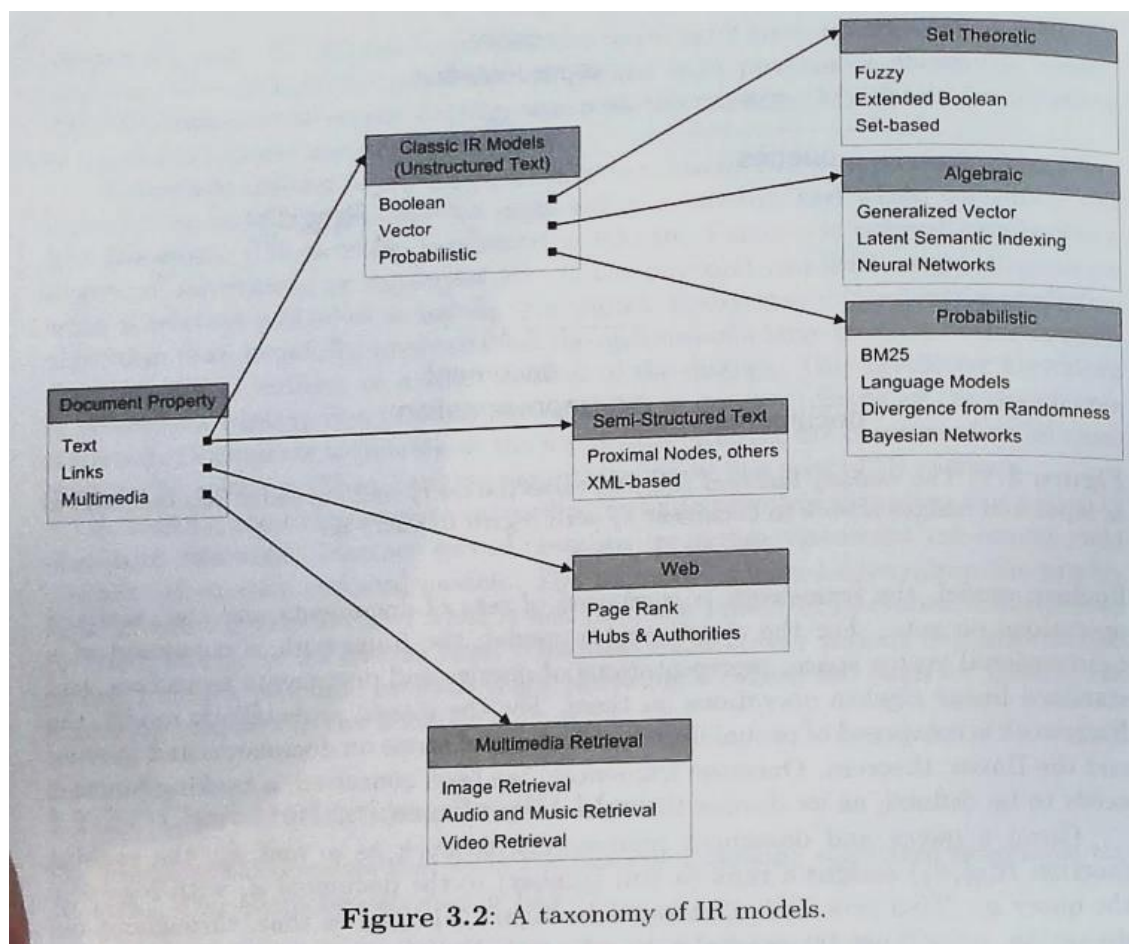
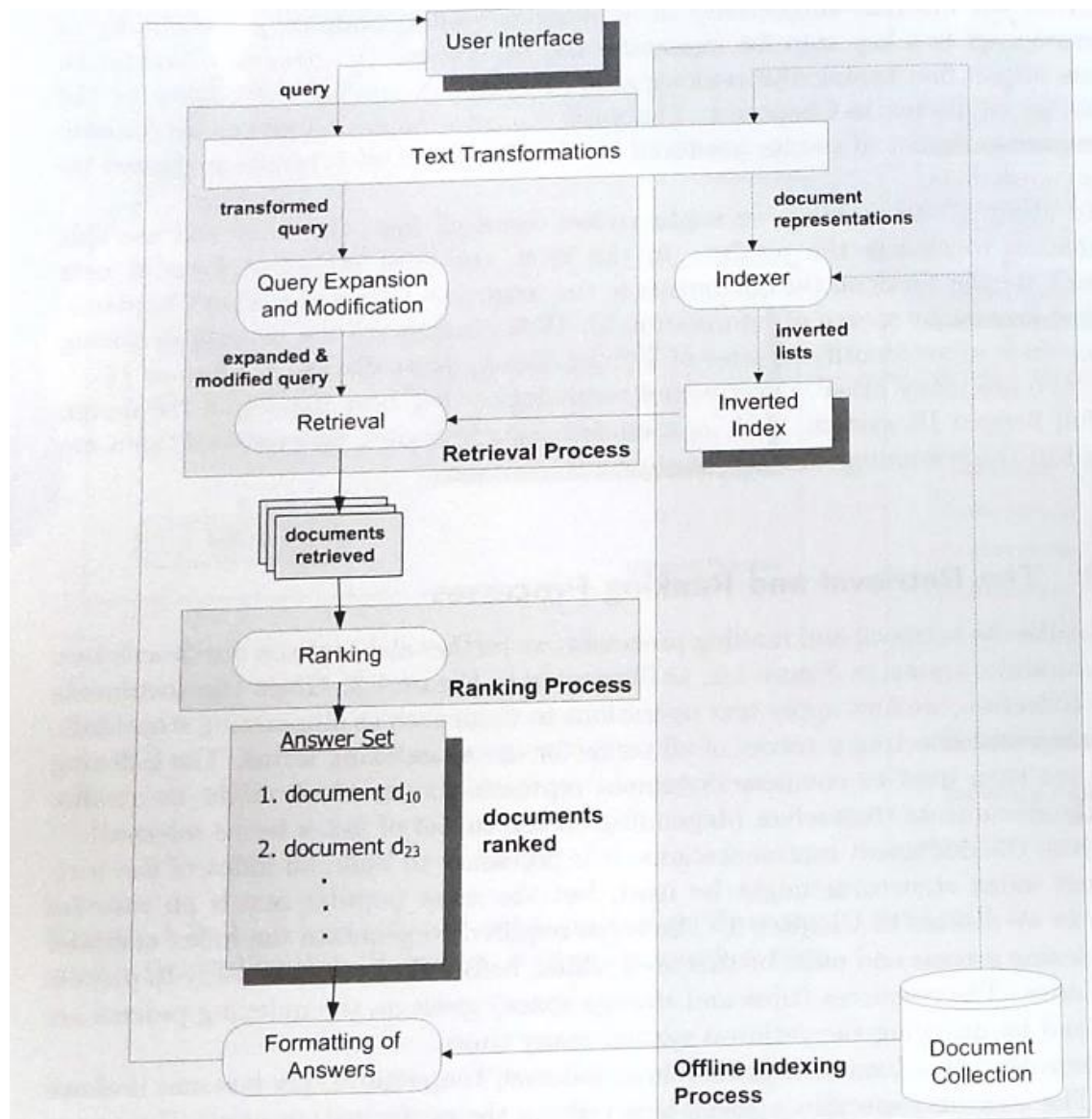


Figure 3.2: A taxonomy of IR models.

Taxonomy of IR Models

IR models are fundamentally based on text, i.e., they use the text of the documents to rank them with regard to a query. On the Web, however, it is also necessary to use information on the link structure of the Web to attain good ranking. Multimedia objects, on the other hand, are encoded rather differently than text documents. Images are encoded as bitmaps of pixels, video objects are encoded as temporal streams of images, and audio objects are encoded as discretized streams of sound. Because of these particularities in their representation, multimedia objects are ranked rather differently or are retrieved without a ranking. Given these characteristics, we distinguish three major types of IR models: those based on text, those based on links, and those based on multimedia objects.

Regarding text-based models, we distinguish among models for unstructured text and models that take into account the structure of the text. In the first category, text is modelled simply as a sequence of words. In the second category, structural components of the text such as title, sections, subsections, and paragraphs are an integral which is usually referred to as semistructured because it also includes unstructured text. Regarding unstructured text, the three classic models in IR are called Boolean, vector, and probabilistic.

In the Boolean model, documents and queries are represented as sets of index terms. We say that the model is set theoretic. In the vector model, documents and queries are represented as vectors in a t -dimensional space. Thus, we say that the model is algebraic. In the framework for modelling document and query representations is based on probability theory. Thus, as the name indicates, we say that the model is probabilistic.

Regarding alternative set theoretic models, we distinguish the fuzzy, the extended Boolean, and the set based models.

Regarding alternative algebraic models, we distinguish the generalized vector, the latent semantic indexing, and the neural network models. Regarding

Regarding alternative probabilistic models, we distinguish the BM25, the Bayesian network, the Divergence from Randomness, and the language models.

Regarding semi-structured text retrieval models i.e, models that deal with structure provided within the text we consider indexing approaches such as the proximal nodes and XML based indexing methods.

On the Web, due to the huge number of documents (or Web pages), text-based ranking by itself is not enough. It is also necessary to consider the links among Web pages as an integral part of the mode. These leads to the link-based retrieval models, particularly PageRank and Hubs & Authorities.

A distinct set of retrieval strategies are those related to the retrieval of multimedia data. Consider, for instance, the case of an image. It is represented as a describing the color and

luminosity of its pixels an information that is not directly related to any interpretation a user might have of the image. To retrieve images of interest to the user, it is necessary to take a series of intermediary steps that are not required for searching text collections. For instance, instead of writing a query, the user might specify the information need by pointing to a given image. That query image can then be compared to the images in the collection to retrieve related images. The approach is very noisy because commonality at pixel level might be very deceptive. The important point is that methods for multimedia retrieval are quite distinct from IR models for text because, for instance, many of them include no form of ranking. Because of that, they are represented separately in Figure 3.2 and are referred to as retrieval strategies.

The simplest form of multimedia retrieval is image retrieval, because an image is static. In the case of audio and video, the representation of the multimedia object has to also include the time dimension, which makes the files much larger and the problem more difficult. Image, audio, and video retrieval.

Boolean Model

- The Boolean model is a simple retrieval model based on set theory and Boolean algebra.
- The model is quite intuitive and has precise semantics.
- In the Boolean model, all elements of the term-document matrix are either 1, to indicate presence of the term in the document, or 0, to indicate absence of the term in the document.
- In Boolean model a query is specified as Boolean expressions with **AND, OR, NOT** operations (connectives) on index terms.
- A query can be expressed as a disjunctive normal form (DNF) composed of conjunctive components
 - \vec{q}_{dnf} : the DNF for a query q
 - \vec{q}_{cc} : conjunctive components (binary weighted vectors) of \vec{q}_{dnf}

Definition In the Boolean model, a query q is a conventional Boolean expression on index terms. Let $c(q)$ be any of the query conjunctive components. Given a document d_j , let $c(d_j)$ be the corresponding document conjunctive component. Then, the similarity of the document d_j to the query q is defined as

$$sim(d_j, q) = \begin{cases} 1 & \text{if } \exists c(q) \mid c(q) = c(d_j) \\ 0 & \text{otherwise} \end{cases} \quad (3.1)$$

If $sim(d_j, q) = 1$ then the Boolean model predicts that the document d_j is relevant to the query q (it might not be). Otherwise, the prediction is that the document is not relevant.

Disadvantages

- The Boolean model predicts that each document is either relevant or non-relevant. There is no notion of a partial match to the query conditions.
- This binary decision criterion without any notion of a grading scale prevents good retrieval quality.
- Boolean expressions have precise semantics; frequently it is not simple to translate information need into a Boolean expression. In fact, most users find it difficult and awkward to express their query requests in terms of Boolean expressions.

Advantage

The main advantage of the Boolean model are the clean formalism behind the model and its simplicity, with the adoption of binary index term weights.

The Vector Model

- The vector model recognizes that Boolean matching is too limiting and proposes a framework in which partial matching is possible. This is accomplished by assigning non-binary weights to index terms in queries and in documents, which are ultimately used to compute the degree of similarity between each document stored in the system and the user query.
- By sorting the retrieved documents in decreasing order of this degree of similarity, the vector model takes into consideration documents that match the query terms only partially.
- The main resultant effect is that the ranked documents provide a more precise answer (in the sense that it better matches the user information need) than the document set retrieved by the Boolean model.

Definition For the vector model, the weight $w_{i,j}$ associated with a pair (k_i, d_j) is positive and non-binary. Further, the index terms in the query are also weighted. Let $w_{i,q}$ be the weight associated with the pair $[k_i, q]$, where $w_{i,q} \geq 0$. Then, the query vector \vec{q} is defined as $\vec{q} = (w_{1,q}, w_{2,q}, \dots, w_{t,q})$ where t is the total number of index terms in the system. As before, the vector for a document d_j is represented by $\vec{d}_j = (w_{1,j}, w_{2,j}, \dots, w_{t,j})$.

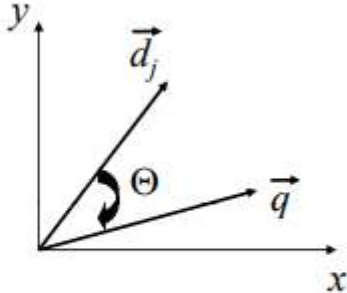
- The similarity of a document d_j to the query q

$$\begin{aligned}
 \text{sim}(d_j, q) &= \cos(\Theta) \\
 &= \frac{\vec{d}_j \cdot \vec{q}}{|\vec{d}_j| \times |\vec{q}|} \\
 &= \frac{\sum_{i=1}^t w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^t w_{i,j}^2} \times \sqrt{\sum_{i=1}^t w_{i,q}^2}}
 \end{aligned}$$

Document length normalization

The same for documents, can be discarded

Won't affect the final ranking



The main **advantages** of the vector model are:

- (1) its term-weighting scheme improves retrieval quality;
- (2) its partial matching strategy allows retrieval of documents that approximate the query conditions;
- (3) its cosine ranking formula sorts the documents according to their degree of similarity to the query;
- (4) document length normalization is naturally built-in into the ranking.

Theoretically, the vector model has the **disadvantage** that index terms are assumed to be mutually independent. However, in practice, leveraging term dependencies is challenging and might lead to poor results, if not done appropriately.

Despite its simplicity, the vector model is a resilient ranking strategy with general collections. It yields ranked answer sets which are difficult to improve upon without query expansion or relevance feedback.

The Probabilistic Model

- The classic probabilistic model introduced in 1976 by Roberston and Sparck Jones which later became known as the binary independence retrieval (BIR) model.
- The probabilistic model attempts to capture the IR problem within a probabilistic framework.
- The probabilistic model computes the similarity coefficient (SC) between a query and a document as the probability that the document will be relevant to the query. This reduces the relevance ranking problem to an application of probability theory. Probability theory can be used to compute a measure of relevance between a query and a document.
- This family of IR models is based on the general principle that documents in a collection should be ranked by decreasing probability of their relevance to a query. This is often called the probabilistic ranking principle (PRP). [20] Since true probabilities are not available to an IR system, probabilistic IR models estimate the probability of relevance of documents for a query. This estimation is the key part of the model, and this is where most probabilistic models differ from one another. The initial idea of probabilistic retrieval was proposed by Maron and Kuhns in a paper published in 1960. [18] Since then, many probabilistic models have been proposed, each based on a different probability estimation technique.
- The fundamental idea is as follows. Given a user query, there is a set of documents which contains exactly the relevant documents and no other. Let us refer to this set of documents as the ideal answer set. Given the description of this ideal answer set, we would have no problems in retrieving its documents. Thus, we can think of the querying process as a process of specifying the properties of an ideal answer set (which is analogous to interpreting the IR problem as a problem of clustering). The problem is that we do not know exactly what these properties are. All we know is that there are index

terms whose semantics should be used to characterize these properties. Since these properties are not known at query time, an effort has to be made at initially guessing what they could be. This initial guess allows us to generate a preliminary probabilistic description of the ideal answer set which is used to retrieve a first set of documents. An interaction with the user is then initiated with the purpose of improving the probabilistic description of the ideal answer set. Such interaction could proceed as follows.

The user takes a look at the retrieved documents and decides which ones are relevant and which ones are not (in truth, only the first top documents need to be examined). The system then uses this information to refine the description of the ideal answer set. By repeating this process many times, it is expected that such a description will evolve and become closer to the real description of the ideal answer set. Thus, one should always have in mind the need to guess at the beginning the description of the ideal answer set. Furthermore, a conscious effort is made to model this description in probabilistic terms.

Definition *For the probabilistic model, the index term weight variables are all binary i.e., $w_{i,j} \in \{0, 1\}$, $w_{i,q} \in \{0, 1\}$. A query q is a subset of index terms. Let R be the set of documents known (or initially guessed) to be relevant. Let \bar{R} be the complement of R (i.e., the set of non-relevant documents). Let $P(R|\vec{d}_j)$*

be the probability that the document d_j is relevant to the query q and $P(\bar{R}|\vec{d}_j)$ be the probability that d_j is non-relevant to q . The similarity $\text{sim}(d_j, q)$ of the document d_j to the query q is defined as the ratio

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

Advantages and Disadvantages of the Probabilistic Model

The main **advantage** of the probabilistic model, in theory, is its optimality, i.e., documents are ranked in decreasing order of their probability of being relevant, based on the information available to the system. In practice, however, this does not work so well because relevance of a document is affected by variables that are not in the system.

The disadvantages include: (1) the need to guess the initial separation of documents into relevant and non-relevant sets; (2) the fact that the method does not take into account the frequency with which an index term occurs inside a document i.e, all weights are binary); and (3) the lack of document length normalization. More advanced variations of the probabilistic model, such as the BM-25 model, correct these deficiencies to yield improved retrieval.

Comparison of Classic Models

BOOLEAN MODEL	VECTOR MODEL	PROBABILISTIC MODEL
It evaluates queries as evaluating Boolean expression.	It uses the concept of index weights and partial matching to match a document to a query.	It evaluates the queries by using the ideal set probabilistic index terms.
Weights are binary. The document is either relevant or irrelevant.	Index terms are weighted. So, there is a ranking created based on these weights(using similarity).	Weights are binary. Initially the document either belongs to the ideal set or is considered irrelevant.
It is simple to evaluate based on the query and the document.	It is more complex than binary as the index term weighting needs to be done.	This is the most complex model since neither the weights nor the ideal set is initially defined.
Performance is not that good.	Performance is considered to be optimal.	Performance is proved to be optimal. However, in practice it may become impractical.

In general, the Boolean model is considered to be the weakest classic method. Its main problem is the inability to recognize partial matches which frequently leads to poor performance. There is some controversy as to whether the probabilistic model outperforms the vector model. Croft performed some experiments and suggested that the probabilistic model provides a better retrieval performance.

However, experiments done afterwards by Salton and Buckley refute that claim. Through several different measures, Salton and Buckley showed that the vector model is expected to outperform the probabilistic model with general collections. This also seems to be the dominant thought among researchers, practitioners, and the Web community, where the popularity of the vector model runs high.

Introduction to alternative algebraic models

1. Generalized vector space model

GVSM is an expansion from VSM that represents the documents base on similarity value between query and minterm vector space of documents collection. Minterm vector is defined by the term in query. Therefore, in retrieving a document can be done base on word meaning inside the query.

Generalized Vector Space Models (GVSM) extend the standard Vector Space Model (VSM) by embedding additional types of information, besides terms, in the representation of

documents. An interesting type of information that can be used in such models is semantic information from word thesauri like WordNet.

The most challenging problem is to incorporate the semantic information in a theoretically sound and rigorous manner and to modify the standard interpretation of the VSM.

Definitions

GVSM introduces term to term correlations, which deprecate the pairwise orthogonality assumption. More specifically, the factor considered a new space, where each term vector t_i was expressed as a linear combination of 2^n vectors m_r where $r = 1 \dots 2^n$.

For a document d_k and a query q the similarity function now becomes

$$sim(d_k, q) = \frac{\sum_{j=1}^n \sum_{i=1}^n w_{i,k} * w_{i,q} * t_i \cdot t_j}{\sqrt{\sum_{i=1}^n w_{i,k}^2} * \sqrt{\sum_{i=1}^n w_{i,q}^2}}$$

where t_i and t_j are now vectors of a 2^n dimensional space.

Term correlation $t_i \cdot t_j$ can be implemented in several ways. For an example, Wong et al. uses the term occurrence frequency matrix obtained from automatic indexing as input to their algorithm. The term occurrence and the output is the term correlation between any pair of index terms.

2. Latent Semantic Indexing Model

Summarizing the contents of documents and queries through a set of index terms can lead to poor retrieval performance due to two effects.

1. Many unrelated documents might be included in the answer set.
2. Relevant documents which are not indexed by any of the query keywords are not retrieved.

The main reason for these two effects is the inherent vagueness associated with a retrieval process which is based on keyword sets. The ideas in a text are more related to the concepts described in it than to the index terms used in its description. Thus, the process of matching documents to a given query could be based on concept matching instead of index term matching.

This would allow the retrieval of documents even when they are not indexed by query index terms. For instance, a document could be retrieved because it shares concepts with another document which is relevant to the given query. Latent semantic indexing is an approach introduced in 1988 which addresses these issues.

Latent Semantic Indexing is a method implemented in IR system to retrieve document base on overall meaning of users' query input from a document, not based on each word

translation. LSI uses a matrix algebra technique namely Singular Value Decomposition (SVD).

The main idea in the latent semantic indexing model is to map each document and query vector into a lower dimensional space which is associated with concepts. This is accomplished by mapping the index term vectors into this lower dimensional space. The claim is that retrieval.

3. Neural Network Model

In an information retrieval system, document vectors are compared with query vectors for the computation of a ranking. Thus, index terms in documents and queries have to be matched and weighted for computing this ranking. Since neural networks are known to be good pattern matchers, it is natural to consider their usage as an alternative model for information retrieval.

Neural ranking models for information retrieval (IR) use shallow or deep neural networks to rank search results in response to a query. Traditional learning to rank models employ machine learning techniques over hand-crafted IR features. By contrast, neural models learn representations of language from raw text that can bridge the gap between query and document vocabulary.

There's no conclusive evidence that a neural network provides superior retrieval performance with general collections. In fact, the model has not been tested extensively with large document collections. However, a neural network does present an alternative modeling paradigm.

Keyword Based Querying

A query is the formulation of a user information need. In its simplest form, a query is composed of keywords and the documents containing such keywords are searched for. Keyword based queries are popular because they are intuitive, easy to express, and allow for fast ranking.

1. Single-Word Queries

- A query is formulated by a word
- A document is formulated by long sequences of words
- A word is a sequence of letters surrounded by separators.
- All documents that include this word are retrieved.
- Documents may be ranked by the *frequency* of this word in the document.

2. Context Queries

- Search words in a given context.
- Single-word queries with the ability to search words in a given context, that is, near other words.

➤ Types

1. Phrase

- A query is a sequence of words treated as a single unit.

- Also called “literal string” or “exact phrase” query.
- Phrase is usually surrounded by quotation marks.
- All documents that include this phrase are retrieved.
- Usually, separators (commas, colons, etc.) and “trivial words” (e.g., “a”, “the”, or “of”) in the phrase are ignored.

2. Proximity

- Restrict the distance within a document between two search terms.
- Important for large documents in which the two search words may appear in different contexts.

3. Boolean Queries

- Describe the information needed by relating multiple words with Boolean operators.
- Operators: **AND, OR, BUT**
- **OR** The query (e1 OR e2) selects all documents which satisfy e1 or e2. Duplicates are eliminated.
- **AND** The query (e1 AND e2) selects all documents which satisfy both e1 and e2.
- **BUT** The query (e1 BUT e2) selects all documents which satisfy e1 but not e2.

4. Natural Language

- Generalization of “fuzzy Boolean”
- A query is an enumeration of words and context queries
- All the documents matching a portion of the user query are retrieved
- Natural language queries are converted to a formal language for processing against a set of documents.

Pattern matching

- Pattern matching is the query formulations method based on the concept of a pattern which allow the retrieval of pieces of text that have some property.
- These data retrieval queries are useful for linguistics, text statistics, and data extraction. Their result can be fed into the composition mechanism to form phrases and proximity queries, comprising called basic queries. Basic queries can be combined using Boolean expressions.
- These data retrieval capabilities can be viewed as enhanced tools for information retrieval. However, it is more difficult to rank the result of a pattern matching expression.
- A pattern is a set of syntactic features that must occur in a text segment. Those segments satisfying the pattern specifications are said to 'match' the pattern. We are interested in documents containing segments which match a given search pattern. Each system allows the specification of some types of patterns, which range from very simple (for example, words) to rather complex (such as regular expressions). In general, as more powerful is the set of patterns allowed, more involved are the queries that the user can formulate and more complex is the implementation of the search.

The most used types of patterns are:

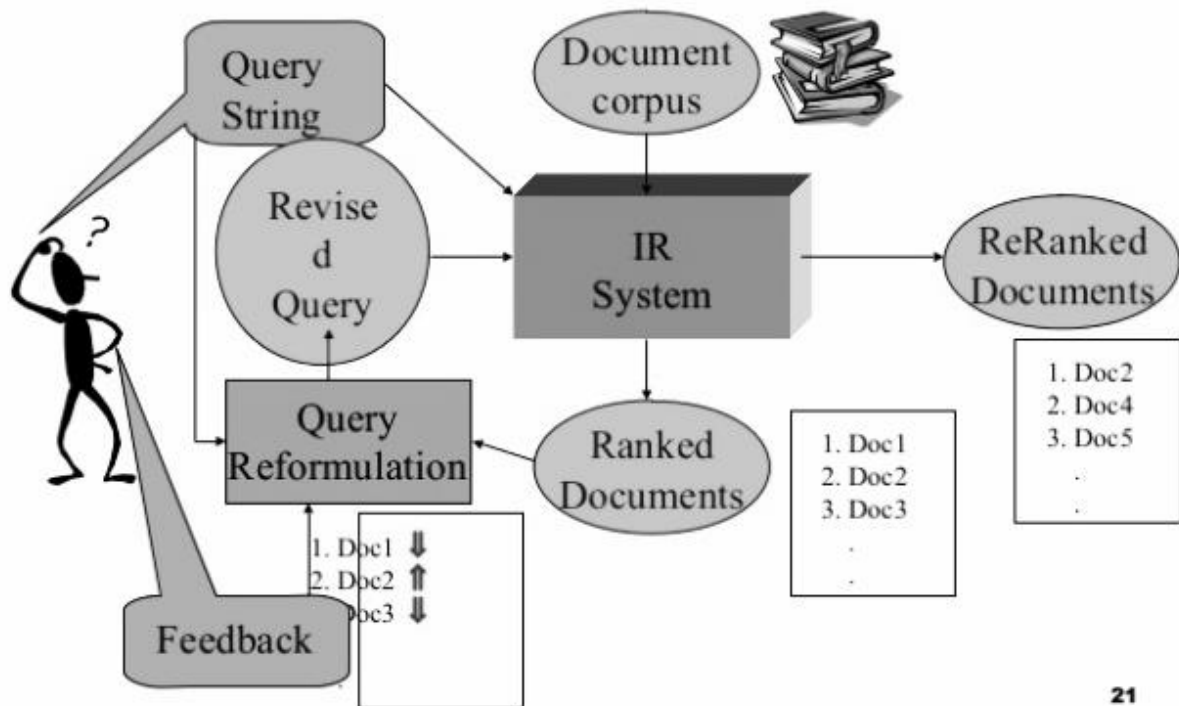
- **Words:** A string (sequence of characters) which must be a word in the text
- **Prefixes:** A string which must form the beginning of a text word.
Eg: 'comput'->'computer', 'computation', 'computing', etc
- **Suffixes:** A string which must form the termination of a text word.
Eg: 'ters'->'computers', 'testers', 'painters', etc
- **Substrings:** A string which can appear within a text word.
Eg: 'tal'->'coastal', 'talk', 'metallic', etc
- **Ranges:** A pair of strings which matches any word lying between them in lexicographical order. For instance, the range between words 'held' and 'hold' will retrieve strings such as 'hoax' and 'hissing'.
- **Allowing errors:** A word together with an error threshold. This search pattern retrieves all text words which are 'similar' to the given word.
- **Regular expressions:** Some text retrieval systems allow searching for regular expressions. A regular expression is a rather general pattern built up by simple strings (which are meant to be matched as substrings) and the following operators: union, concatenation, repetition.
- **Extended patterns:** It is normal to use a more user-friendly query language to represent some common cases of regular expressions. Extended patterns are subsets of the regular expressions which are expressed with a simpler syntax.

User Feedback Relevance

Relevance feedback is the most popular query reformulation strategy. In a relevance feedback cycle, the user is presented with a list of the retrieved documents and, examining them, marks those which are relevant. In practice, only the top 10 (or 20) ranked documents need to be examined. The main idea consists of selecting important terms, or expressions, attached to the documents that have been identified as relevant by the user, and of enhancing the importance of these terms in a new query formulation. The expected effect is that the new query will be moved towards the relevant documents and away from the non-relevant ones.

Early experiments using the Smart system [695] and later experiments using the probabilistic weighting model [677] have shown good improvements in precision for small test collections when relevance feedback is used. Such improvements come from the use of two basic techniques: query expansion (addition of new terms from relevant documents) and term reweighting (modification of term weights based on the user relevance judgement).

Relevance feedback presents the following main advantages over other query reformulation strategies: (a) it shields the user from the details of the query reformulation process because all the user has to provide is a relevance judgement on documents; (b) it breaks down the whole searching task into a sequence of small steps which are easier to grasp; and (c) it provides a controlled process designed to emphasize some terms (relevant ones) and de-emphasize others (non-relevant ones).



21

Query Expansion

In relevance feedback, users give additional input (relevant/non-relevant) on documents, which is used to reweight terms in the documents

In query expansion, users Add new terms to query from relevant documents.

Term Reweighting: In term reweighting we increase weight of terms in relevant documents and decrease weight of terms in irrelevant documents.

Document preprocessing is a procedure which can be divided mainly into five text operations (or transformations):

(1) Lexical analysis of the text with the objective of treating digits, hyphens, punctuation marks, and the case of letters.

Lexical analysis is the process of converting a stream of characters the documents) into a stream of words (the candidate words to be adopted as index terms). Thus, one of the major objectives of the lexical analysis phase is the identification of the words in the text.

(2) Elimination of stopwords with the objective of filtering out words with very low discrimination values for retrieval purposes.

Words which are too frequent among the documents in the collection are not good discriminators. In fact, a word which occurs in 80% of the documents in the collection is useless for purposes of retrieval. Such words are frequently referred to as stopwords and are normally filtered out as potential index terms. Articles, prepositions, and conjunctions are natural candidates for a list of stopwords.

Elimination of stopwords has an additional important benefit. It reduces the size of the indexing structure considerably.

(3) Stemming of the remaining words with the objective of removing affixes (i.e., prefixes and suffixes) and allowing the retrieval of documents containing syntactic variations of query terms (e.g., connect, connecting, connected, etc).

(4) Selection of index terms to determine which words/stems (or groups of words) will be used as an indexing elements. Usually, the decision on whether a particular word will be used as an index term is related to the syntactic nature of the word. In fact, noun words frequently carry more semantics than adjectives, adverbs, and verbs.

(5) Construction of term categorization structures such as a thesaurus, or extraction of structure directly represented in the text, for allowing the expansion of the original query with related terms (a usually useful procedure).

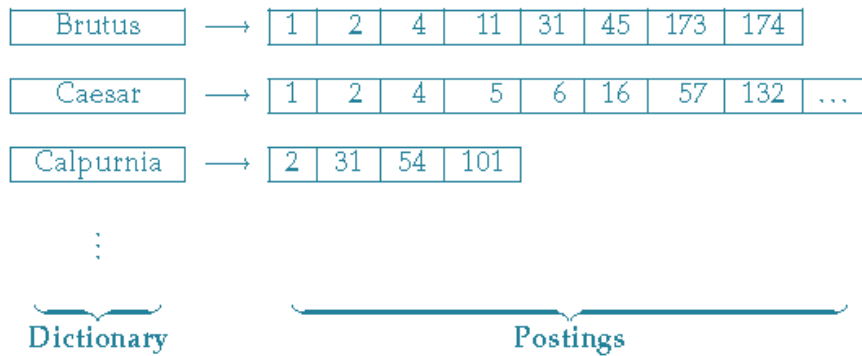
Document clustering is the operation of grouping together similar (or related) documents in classes. In this regard, document clustering is not really an operation on the text but an operation on the collection of documents.

The operation of clustering documents is usually of two types: global and local. In a global clustering strategy, the documents are grouped accordingly to their occurrence in the whole collection. In a local clustering strategy, the grouping of documents is affected by the context defined by the current query and its local set of retrieved documents.

Clustering methods are usually used in IR to transform the original query in an attempt to better represent the user information need. From this perspective, clustering is an operation which is more related to the transformation of the user query than to the transformation of the text of the documents.

INVERTED INDEX:

Each term in a document is called as index. **Inverted index**, or sometimes **inverted file**, has become the standard term in information retrieval. The basic idea of an inverted index is shown in Figure.



We keep a **dictionary** of terms (sometimes also referred to as a **vocabulary** or **lexicon**). Then for each term, we have a list that records which documents the term occurs in. Each item in the list – which records that a term appeared in a document (and, later, often, the positions in the document) – is conventionally called a **posting**. The list is then called a **postings list** (or **inverted list**), and all the postings lists taken together are referred to as the **postings**. The dictionary in above figure has been sorted alphabetically and each postings list is sorted by document ID.

Building an inverted index:

To gain the speed benefits of indexing at retrieval time, we have to build the index in advance. The major steps in this are:

1. Collect the documents to be indexed:

Friends, Romans, countrymen. So let it be with Caesar ...

2. Tokenize the text, turning each document into a list of tokens:

Friends Romans countrymen So ...

3. Do linguistic preprocessing, producing a list of normalized tokens, which are the indexing terms

friend roman countryman so ...

4. Index the documents that each term occurs in by creating an inverted index, consisting of a dictionary and postings.

Within a document collection, we assume that each document has a unique serial number, known as the **document identifier (docID)**. During index construction, we can simply assign successive integers to each new document when it is first encountered. The input to indexing is a list of normalized tokens for each document, which we can equally think of as a list of pairs of term and docID, as in following figure.

The core indexing step is **sorting** this list so that the terms are alphabetical, giving us the representation in the middle column of Figure. Multiple occurrences of the same term from the same document are then **merged**. Instances of the same term are then grouped, and the result is split into a dictionary and postings, as shown in the right column of Figure. The dictionary also records some statistics, such as the number of documents which contain

each term (the **Document Frequency**, which is here also the length of each postings list). The postings are secondarily sorted by docID.

In the resulting index, we pay for storage of both the dictionary and the postings lists. The latter are much larger, but **the dictionary is commonly kept in memory, while postings lists are normally kept on disk.**

What data structure should be used for a postings list? A fixed length array would be wasteful as some words occur in many documents, and others in very few. For an in-memory postings list, two good alternatives are **singly linked lists or variable length arrays**. Singly linked lists allow cheap insertion of documents into postings lists (following updates, such as when recrawling the web for updated documents), and naturally extend to more advanced indexing strategies such as skip lists, which require additional pointers. Variable length arrays win in space requirements by avoiding the overhead for pointers and in time requirements because their use of contiguous memory increases speed on modern processors with memory caches. Extra pointers can in practice be encoded into the lists as offsets. If updates are relatively infrequent, variable length arrays will be more compact and faster to traverse. **We can also use a hybrid scheme with a linked list of fixed length arrays for each term.**

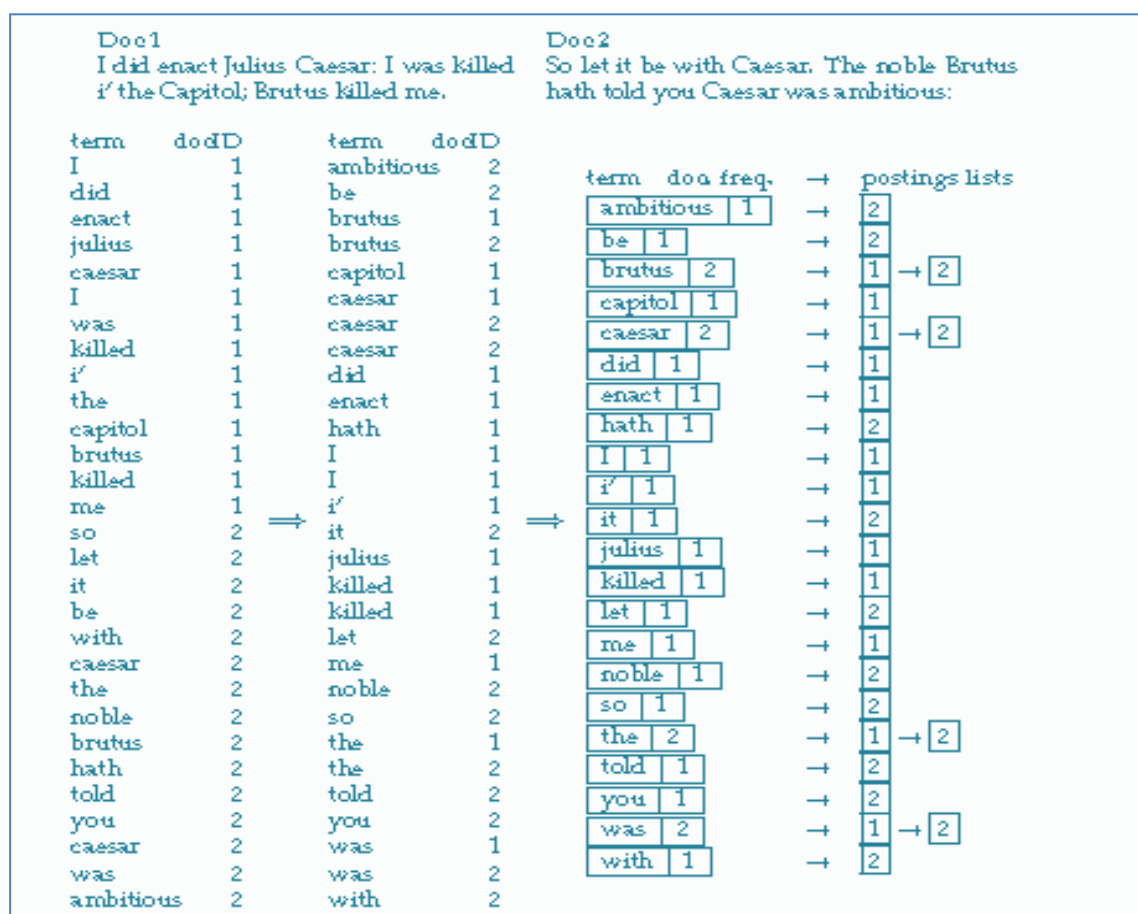


Figure: Building an index by sorting and grouping.

Web Crawling

Web crawling is the process by which we gather pages from the Web to index them and support a search engine. The objective of crawling is to quickly and efficiently gather as many useful web pages as possible, together with the link structure that interconnects them.

Features of Crawler:

Robustness: Ability to handle *spider traps*. The Web contains servers that create spider traps, which are generators of web pages that mislead crawlers into getting stuck fetching an infinite number of pages in a particular domain. Crawlers must be designed to be resilient to such traps.

Politeness: Web servers have both implicit and explicit policies regulating the rate at which a crawler can visit them. These politeness policies must be respected.

Distributed: The crawler should have the ability to execute in a distributed fashion across multiple machines.

Scalable: The crawler architecture should permit scaling up the crawl rate by adding extra machines and bandwidth.

Performance and efficiency: The crawl system should make efficient use of various system resources including processor, storage and network bandwidth.

Quality: The crawler should be biased towards fetching “useful” pages first.

Freshness: In many applications, the crawler should operate in continuous mode: it should obtain fresh copies of previously fetched pages.

Extensible: Crawlers should be designed to be extensible in many ways to cope with new data formats, new fetch protocols, and so on. This demands that the crawler architecture be modular.

Basic Operation:

The crawler begins with one or more URLs that constitute a **seed set**. It picks a **URL** from this seed set, and then fetches the web page at that URL. The fetched page is then **parsed**, to *extract both the text and the links from the page* (each of which points to another URL). The extracted text is fed to a **text indexer**. The **extracted links (URLs)** are then added to a **URL frontier**, which at all times consists of URLs whose corresponding pages have yet to be fetched by the crawler.

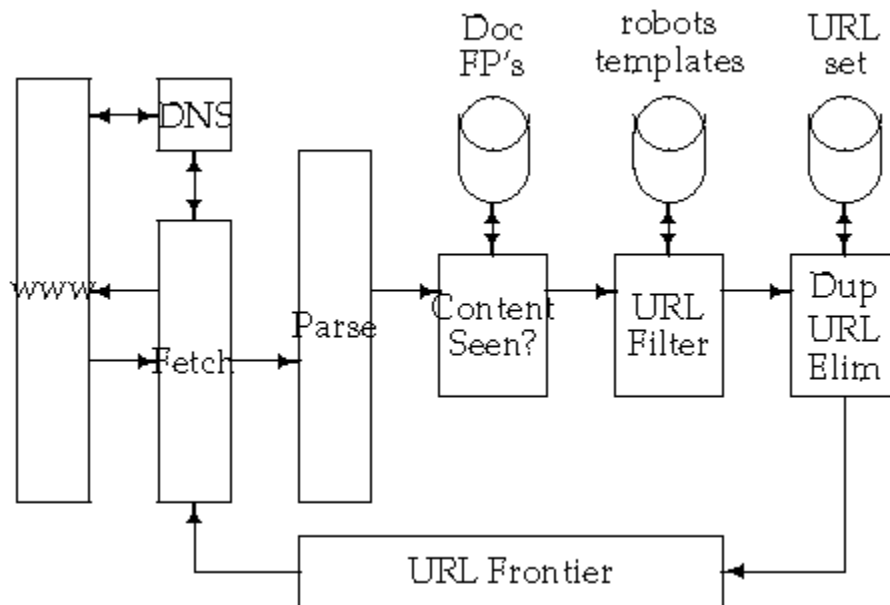
Initially, the URL frontier contains the seed set; as pages are fetched, the corresponding URLs are deleted from the URL frontier. The entire process may be viewed as traversing the web graph. In continuous crawling, the URL of a fetched page is added back to the frontier for fetching again in the future.

Web Crawler Architecture:

The simple scheme outlined above for crawling demands several modules that fit together as shown in Figure.

- 1) The **URL frontier**, containing URLs yet to be fetched in the current crawl (in the case of continuous crawling, a URL may have been fetched previously but is back in the frontier for re-fetching).

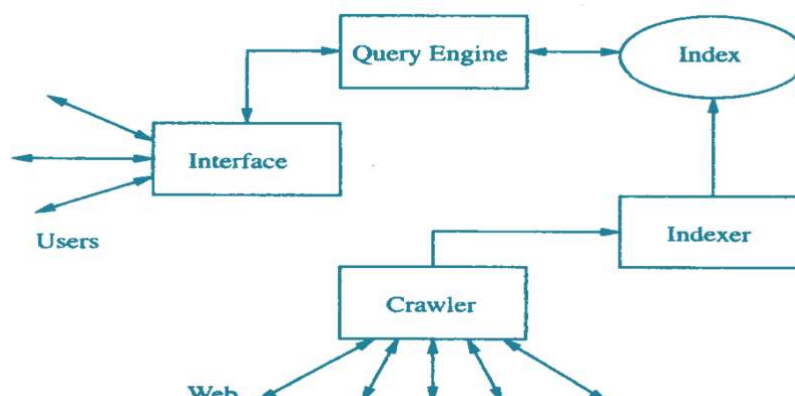
- 2) A **DNS resolution module** that determines the web server from which to fetch the page specified by a URL.
- 3) A **fetch module** that uses the http protocol to retrieve the web page at a URL.
- 4) A **parsing module** that extracts the text and set of links from a fetched web page.
- 5) A **duplicate elimination module** that determines whether an extracted link is already in the URL frontier or has recently been fetched.



WEB SEARCH ENGINE ARCHITECTURES:

a) Centralized Architecture

Most search engines use centralized crawler-indexer architecture. **Crawlers** are programs (software agents) that traverse the Web sending new or updated pages to a main server where they are indexed. **Crawlers are also called robots, spiders, wanderers, walkers, and know bots.** In spite of their name, a crawler does not actually move to and run on remote machines, rather the crawler runs on a local system and sends requests to remote Web servers. **The index** is used in a centralized fashion to answer queries submitted from different places in the Web. The following figure shows the software architecture of a search engine based on the Alta Vista architecture. It has two parts: one that deals with the users, consisting of the user interface and the query engine and another that consists of the crawler and indexer modules.



Problems:

- 1) The main problem faced by this architecture is the gathering of the data, because of the highly dynamic nature of the Web, the saturated communication links, and the high load at Web servers.
- 2) Another important problem is the volume of the data

b) Distributed Architecture:

There are several variants of the crawler-indexer architecture. Among them, the most important is **Harvest**. Harvest uses a distributed architecture to gather and distribute data, which is more efficient than the crawler architecture. **The main drawback is that Harvest requires the coordination of several web servers.**

The Harvest distributed approach addresses several of the problems of the crawler-indexer architecture, such as:

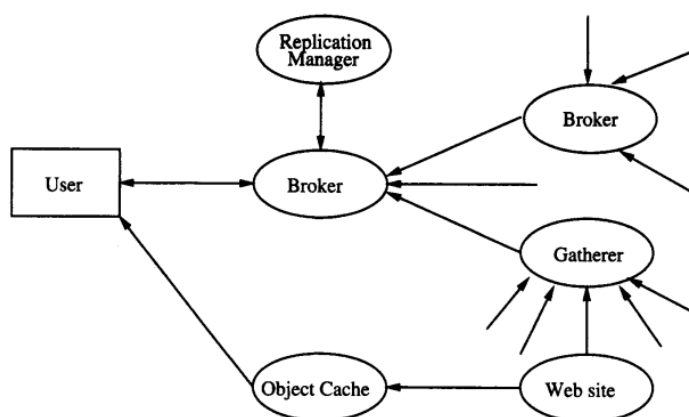
- (1) Web servers receive requests from different crawlers, increasing their load;
- (2) Web traffic increases because crawlers retrieve entire objects, but most of their content is discarded; and
- (3) information is gathered independently by each crawler, without coordination between all the search engines.

To solve these problems, Harvest introduces two main elements: gatherers and brokers.

A gatherer collects and extracts indexing information from one or more Web servers.

Gathering times are defined by the system and are periodic (i.e. there are harvesting times as the name of the system suggests). **A broker provides the indexing mechanism and the query interface to the data gathered.** Brokers retrieve information from one or more gatherers or other brokers, updating incrementally their indices. Depending on the configuration of gatherers and brokers, different improvements on server load and network traffic can be achieved.

A replicator can be used to replicate servers, enhancing user-base scalability. For example, the registration broker can be replicated in different geographic regions to allow faster access. *Replication can also be used to divide the gathering process between many Web servers.* Finally, the **object cache** reduces network and server load, as well as response latency when accessing Web pages.



Link Analysis:

- Link analysis is defined as using hyperlinks for ranking web search results
- Link analysis is only one of the factors used by search engines to compute a score on a given query.
- If a page B has a link to page A, then we say page A has a backlink from page B. We can view back-links as a type of endorsement. The more backlinks a page have, the more important the page is. While ranking if two pages have similar relevance to the query, the more important page should be ranked higher.
- Note that counting in-links is not enough (cf spam links)
- Link analysis is comparable to citation analysis (authority of a paper \equiv amount of citations)

Page Rank Algorithm

PageRank is a link analysis algorithm that estimates the importance of a document by analyzing the link structure of a hyperlinked set of documents. It is named after Larry Page (co-founder of Google).

The simple backlink count was sufficient for well controlled document collection of citation analysis. But in web, there is hardly any control. Millions of pages can be automatically created and linked with each other to manipulate the backlink count (Page et al., 1998). As web consists of conflicting profit making ventures, any evaluation strategy which counts replicable features of web pages is bound to be manipulated.

PageRank extends the idea of backlink by “not counting links from all pages equally, and by normalizing by the number of links on a page.” (Brin and Page, 1998). Here, we assume page A has pages $T_1 \dots T_n$ which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. Also $C(A)$ is defined as the number of links going out of page A (a normalizing factor). The PageRank of a page A is a value in the range 0 to 1 and is given by:

$$PR(A) = \frac{1-d}{N} + d \left(\frac{PR(T_1)}{C(T_1)} + \dots + \frac{PR(T_n)}{C(T_n)} \right)$$

Example

Here, page C has a higher PageRank than Page E, even though it has fewer links to it; the link it has is of a much higher value. A web surfer who chooses a random link on every page (but with 15% likelihood jumps to a random page on the whole web) is going to be on Page E for 8.1% of the time. (The 15% likelihood of jumping to an arbitrary page corresponds to a damping factor of 85%.) Without damping, all web surfers would eventually end up on Pages A, B, or C, and all other pages would have PageRank zero. Page A is assumed to link to all pages in the web, because it has no outgoing links.

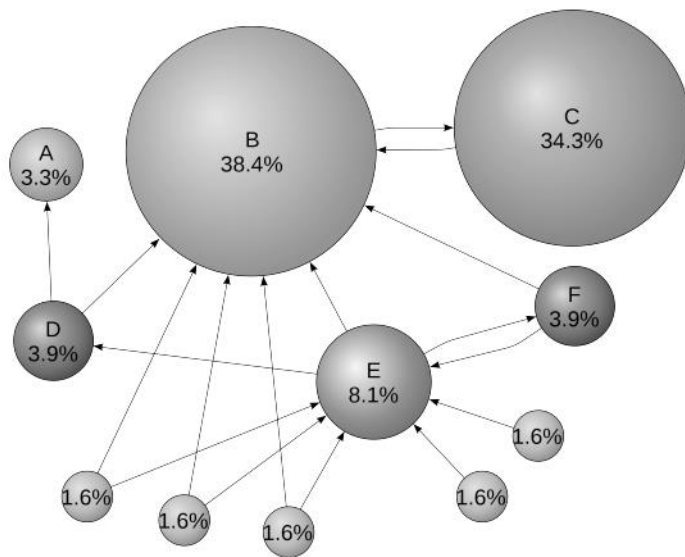


Figure 3.1: PageRank example from Wikipedia

Advantages of using pagerank algorithm

- Random Surfer Model is used as the intuitive justification of PageRank.
- Pages that are well cited from many places around the Web are worth looking at. Also, pages that have perhaps only one citation from a well known site are also generally worth looking at.

Disadvantage of PageRank

PageRank favors older pages than newer ones. The older pages are expected to have more number of citations from important page than a page just introduced. Therefore, page rank should not be used as a standalone metric. It should be used as a parameter only.

HITS algorithm

Hyperlink Induced Topic Search or HITS algorithm is a Link Analysis algorithm developed by Jon Kleinberg (Kleinberg, 1999). This algorithm computes two values for any page:

Hub Score The value of its links to other pages

Authority Score The value of the page's content.

Both the scores are calculated using each other.

A page with good authority score are authoritative sources of information in that topic. Pages with good hub score do not carry much information in themselves but contain links to good authoritative pages (Manning et al., 2009). Hub pages are generally manually compiled collection of links on some specific topic. A good hub page is one that has links to many good authoritative pages. A good authoritative page should have backlinks from many good hub pages.

Formally, the hub score is computed as the sum of the authority scores of all pages it links to. The authority score is the sum of hub scores of all the pages connects to it. So, the authority score increases if many hub pages links to that page. Mathematically,

$$h(v) \leftarrow \sum_{v \rightarrow y} a(y)$$

$$a(v) \leftarrow \sum_{y \rightarrow v} h(y)$$

where $h(v)$ is the hub score of page v , $a(v)$ is the authority score of page v , $v \rightarrow y$ denote the existence of a hyper link from v to y .

3.2.3.2.2 The HITS algorithm The HITS algorithm works in following way (Manning et al., 2009):

1. It assembles the root set of relevant documents by firing the query first.
2. It creates the base set of hubs and authorities from the root set through the procedure given in subsubsection 3.2.3.1.
3. It computes AA^T and $A^T A$ from the adjacency matrix of the base set of documents (a subset of web-graph).
4. It computes the value of \vec{h} and \vec{a} by computing the eigenvectors of AA^T and $A^T A$ respectively. Any algorithm for computing eigenvector like the power iteration method (Manning et al., 2009) can be used.
5. Output the top scoring hubs and authorities.

	HITS	PageRank
Scoring criteria	<i>Good authorities are pointed to by good hubs and good hubs point to good authorities.</i>	<i>A webpage is important if it is pointed to by other important pages.</i>
Number of scores	Dual Rankings a) one with the <i>most authoritative documents</i> related to the query b) other with the most "hubby" documents.	Page rank only presents one score.
Query indepedence	HITS score is calculated after getting the neighbourhood graph according to the query	PageRank score is query independent
Resilience to spamming	Susceptible to spamming since addition of pages slightly affects the ranking.	Since PageRank is able to isolate spam, it is risilient to spamming.

Performance evaluation of search engines using various measures

1. Recall

- Recall is a measure of completeness of the IR process.
- Recall is the ratio of the number of relevant records retrieved to the total number of relevant records in the database.
- Search Engines should have a high recall rate.

2. Precision

- Precision measures the exactness of the retrieval process.
- Precision is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved.
- The 100% precision is an obtainable goal, since the system could be programmed to return just one completely relevant document. However, returning just one document might not resolve the seeker's query. Therefore, a search system attempts to maximize both precision and recall simultaneously.

3. Fallout

- Fallout is all the junk that came up in search that was irrelevant.
- If one retrieves 100 documents and 20 are relevant, then fallout is 80%. Fallout becomes a bigger issue (and bigger problem) as the size of database grows and retrieval gets larger.

4. Precision at Rank k

In web search systems the set of retrieved document is usually huge. But it makes a lot of difference if the relevant document is retrieved early in the rank list that late. To take this into account, precision at a cut-off rank k is introduced. Here the list of relevant documents I is cut-off at rank k. Only documents up to rank k are considered to be retrieved set of documents.

5. F-Score

F-Score tries to combine the precision and Recall measure. It is the harmonic mean of the two. If P is the precision and R is the recall then the F-Score is given by:

$$F - Score = \frac{2 \cdot P \cdot R}{P + R}$$

6. Mean average precision

Mean average precision for a set of queries is the mean of the average precision scores for each query, The mean average precision is given by:

$$MAP = \frac{\sum_{q=1}^Q AveP(q)}{Q}$$

where Q is the number of queries.

7. Average Precision

Precision and recall are single-value metrics based on the whole list of documents returned by the system. For systems that return a ranked sequence of documents, it is desirable to

also consider the order in which the returned documents are presented. By computing a precision and recall at every position in the ranked sequence of documents, one can plot a precision-recall curve, plotting precision $P(r)$ as a function of recall r . Average precision computes the average value of $p(r)$ over the interval from $r=0$ to $r=1$

$$\text{AveP} = \int_0^1 p(r) dr$$

8. R-Precision

R-precision requires knowing all documents that are relevant to a query. The number of relevant documents, R , is used as the cutoff for calculation, and this varies from query to query. For example, if there are 15 documents relevant to "red" in a corpus ($R=15$), R-precision for "red" looks at the top 15 documents returned, counts the number that are relevant r turns that into a relevancy fraction: $r/R = r/15$.

Precision is equal to recall at the **R**-th position.

Empirically, this measure is often highly correlated to mean average precision.

Metasearchers

Metasearchers are Web servers that send a given query to several search engines, Web directories and other databases, collect the answers and unify them.

The main advantages of metasearchers are the ability to combine the results of many sources and the fact that the user can pose the same query to various sources through a single common interface. Metasearchers differ from each other in how ranking is performed in the unified result (in some cases no ranking is done), and how well they translate the user query to the specific query language of each search engine or Web directory.

The advantages of metasearchers are that the results can be sorted by different attributes such as host, keyword, date, etc; which can be more informative than the output of a single search engine. Therefore browsing the results should be simpler. On the other hand, the result is not necessarily all the Web pages matching the query, as the number of results per search engine retrieved by the metasearcher is limited (it can be changed by the user, but there is an upper limit). Nevertheless, pages returned by more than one search engine should be more relevant.

SEO(Search engine optimization) The process of maximizing the number of visitors to a particular website by ensuring that the site appears high on the list of results returned by a search engine.

SEO depends upon various factors like keyword density, website design, backlinks to a website, age of website etc.

Online Information retrieval system is one type of system or technique by which users can retrieve their desired information from various machine readable online databases. On the

other-word OIRS is a combination of computer and its various hardware such as networking terminal, communication layer and link, modem, disk driver and many computer software packages are used for retrieving information from various online database, this system is known as Online Information Retrieval System.

The terminology Online Information Retrieval System first introduced in 1970s. It is a global system for retrieving information from many online databases very rapidly.

Online Information Retrieval System is a method by which a user search information from machine readable database and retrieve their desired information very rapidly and easily.

Characteristic of Online Information Retrieval System:

Some of the characteristics of online information retrieval system are as:

1. **Direct access:** By using computer input / output device and communication channel user of information can directly access to their desired information storage.
2. **Conventional mode:** This system use conventional media channel like computer, software, telecommunication network, internet and other technologies.
3. **Two way communication:** It is a process, where a terminal and a server network communicate with each other.
4. **Centralized storage:** All the materials find on the searching process stored on a centralized point or database.
5. **Centralized control:** All the process of works is controlled by a centralized server.
6. **Rapid response:** This system response very rapidly when system receives a new search request from server.
7. **Real time communication:** This system has ability to response immediately when new search query found.
8. **Modern process:** OIRS is a modern way for information retrieval system.
9. **Effective communication:** Online Information Retrieval System is an effective way of communication between man and machine.

Advantages of Online Information Retrieval System:

Online Information retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources via the help of computer or other related technology. This system has many advantages. Some of them are as:

1. **Time saves:** OIRS save the time of the reader when search for new information.
2. **Easy to understand:** Searching process of OIRS is very easy to understand.
3. **Current information:** Information stored in a database is more current than the printed publication.
4. **Database:** All the information is stored on a database, so we can search information more effectively.
5. **Multi database search:** Users can search their desired information from multi-database at a same time.
6. **Multiple concepts:** At a same time they can use multiple keyword / concept for search.
7. **Multi user:** It has the ability to serve multi user at a same time.

8. **Geographical Barrier:** Geographical factors are not an obstacle for searching information from storage. Users may be able to search information from anywhere of the world.
9. **Preservation system:** We can easily store all of our search results on our computer.
10. **Various Formats:** We can retrieve information from our search as various formats, such as book, journal, PDF, document format, etc.
11. **Cost:** Searching cost is less than manual searching.
12. **Multi access point:** At a same time many users can be able to access its storage.
13. **Up-to-date information:** Most of the result retrieved by this system is up-to-date.
14. **Rapid access:** Users can be able to access very rapidly to the search result.
15. **Resource sharing:** It has the capability of resource sharing.
16. **Search logic:** Its search logic is user friendly.

These are some of the advantages of Online Information Retrieval System.

Disadvantages of Online Information Retrieval System:

Some of the common disadvantages of OIRS are as:

1. **Primary cost:** The primary establishment cost so high to bear for a small and medium library.
2. **Lack of Budget:** This is always a problem with lack of library budget, so it is not always possible to maintain such a costly system by a library.
3. **IT knowledge:** Most of the library user and staff do not have enough IT knowledge to run this system.
4. **Lack of training facility:** Bangladesh as a developing country, here is lack of proper IT training facility to create well trained manpower to run this system.
5. **Electric problem:** Load shedding is a common problem here. Frequency of power supply creates huge problem.
6. **Lack of Networking and internet facility:** Lack of network and internet facility creates many problems here.
7. **Internet speed:** Slow speeds of Internet delay the retrieval system.
8. **Mental dissatisfaction:** Online Information Retrieval system use computer technology for retrieving information which creates many physical and mental problems.
9. **Irrelevant information:** Sometime this system retrieve many irrelevant information, this may dissatisfy user.

Web Search Personalization

- Different users have different types of information need. Queries never fully represent the complete information need.
- For example, the query 'Matrix' can refer to the matrix concept of mathematics or the 1999 movie by Wachowski brothers.
- The idea behind is, as the information need varies from person to person, the search results should also be different for different people.

- Also, a single ranking of results is unlikely to be optimal for all users. Personalized web search ranking is the only way to bridge the gap.
- A search engine can predict user's information need by using user's past interactions with the search engine like which queries they fired earlier, which links they visited and so on.

User interactions are mainly collected in two ways:

1. The queries fired by users earlier are logged in a **Query Log**
2. The links visited by the user in the Search Results page is recorded (**Click-through data**)

General ways of personalizing search

1. Query Recommendation

When a user gives a query, we can suggest user more queries he might be interested in.

2. Query Expansion

Modify or augment user query. E.g., query term "IR" can be augmented with either "information retrieval" or "Ingersoll-Rand" depending on user interest.

3. ReRanking

- Issue the same query and fetch the same result but rerank the results based on a user profile.
- Allows both personalized and globally relevant results.

4. Relevance Feedback Relevance feedback is one of the classical ways of refining search engine rankings. It works in the following way: Search engine firsts generate an initial set of rankings. Users select the relevant documents within this ranking. Based on the information in these documents a more appropriate result and ranking is presented