

# Information Organisation: Information Retrieval

Jagdish S. Gangolly,  
Department of Informatics,  
SUNY Albany

November 1, 2009

Information retrieval is a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information.

Gerard Salton

*Automatic Information Organization and Retrieval*

McGraw Hill, 1968

# Table of Contents

- ▶ Information Retrieval v. Database Retrieval
- ▶ What are Documents
- ▶ Main Concepts
- ▶ Nature of Relevance
- ▶ Text Processing
- ▶ Word Frequencies & Zipf's Law
- ▶ Heap's Law and Vocabulary Growth
- ▶ Document Parsing
- ▶ Retrieval Models

# Information Retrieval v. Database Retrieval

- ▶ Data
  - ▶ Database contains structured data; in information retrieval the data is unstructured/semi-structured. (Semi-structured data is some times described as self-describing)
- ▶ Queries
  - ▶ Database queries are precise/exact and written in programming languages (such as SQL); in information retrieval the queries are imprecise/vague and written in natural language
  - ▶ Query processing
    - ▶ Database query processing involves exact matches; query processing in retrieval involves approximate matching
  - ▶ Nature of query results
    - ▶ In database queries the results are necessarily relevant; in information retrieval the query results may or may not be relevant
- ▶ User interaction
  - ▶ In database queries the interaction is minimal; in information retrieval the user interacts by providing feedback on the relevance of the results

# What are Documents

- ▶ Text
  - ▶ web pages, email, books, news stories, scholarly papers, text messages, Word, Powerpoint, PDF, forum postings, patents, IM sessions, etc.
- ▶ Images
- ▶ Audio
- ▶ Video

# Main Concepts

- ▶ **Relevance:** a document is relevant to a query if it contains the information that the user is looking for
- ▶ **Evaluation:** Measures to evaluate the performance of retrieval tasks
  - ▶ *Precision:* Proportion of retrieved documents that are relevant
  - ▶ *Recall:* Proportion of relevant documents that are retrieved
- ▶ *Vocabulary mismatch Problem:* Word use variability leads to poor recall in retrieval. There are many solutions such as:
  - ▶ Controlled vocabularies
  - ▶ Term expansion, by using thesauri
  - ▶ Document clustering to reflect semantics of the document
  - ▶ Latent Semantic Indexing/Singular Value Decomposition

# Nature of Relevance

- ▶ Relevance depends on the content of the document, the information needs of the user, the query posed
- ▶ Two approaches to the relevance issue are the use of test collections and use of clickthrough data
  - ▶ **Use of Test Collections:** Prepare a list of queries and a list of relevant documents for each query. Retrieval algorithms can be evaluated by the use of test collections. An example is TREC
  - ▶ **Use of Clickthrough data:** Data on web searches used to infer relevance

The goal of text processing is to index the text. Index is a concise representation of text for retrieval purposes. It facilitates efficient retrieval. The steps include

- ▶ Stripping punctuation
- ▶ Tokenisation
- ▶ Stopping: Removal of *function* or *stop* words  
Usually closed class words such as prepositions, determiners, conjunctions, etc.

- ▶ Stemming to allow words related by morphological rules to match each other in retrieval
  - ▶ Inflectional (walk: walking, walked; grounding the term) vs. Derivational (Walk: walker; changing the lexical category from verb to noun)
  - ▶ Algorithmic stemmers (eg., Porter stemmer) vs. Dictionary-based stemmers
  - ▶ Hybrid stemmers (eg., Krovetz stemmer)
- ▶ Phrases and N-grams
  - ▶ Identification by POS taggers
  - ▶ Identification of N-grams by language models, which can also identify idioms
  - ▶ N-grams of all lengths follow Zipf distribution

- ▶ Document structure and markup
  - ▶ Indexing might require detagging (using detaggers such as sgrep and fxgrep)
  - ▶ Some tags (such as anchors HREF are used by some search engines. For example Google's Page Rank algorithm uses such links)
  - ▶ XQuery language supports querying structure as well as content of xml pages, but if the content is largely text information retrieval models may be a better alternative
- ▶ Word frequencies and collocation statistics

# Word Frequencies & Zipf's Law

Distribution of words in text is very skewed, with a few words occurring most frequently

Table: Frequency of words in English

Words	Frequency (percentage)
Top 2 words (of and the)	10
Top 6 words	20
Top 50 words	50

**Zipf's Law** If word frequencies and word ranks are denoted by  $f$  and  $r$  respectively, then

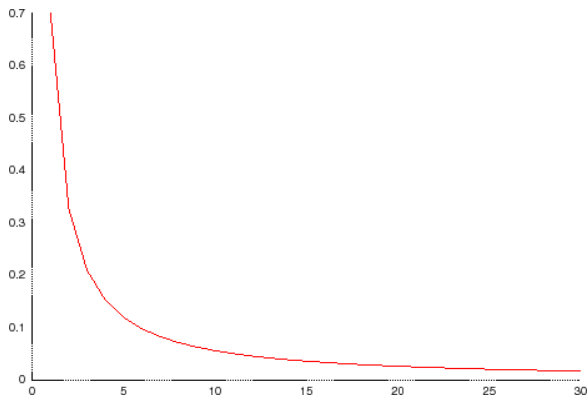
$$f \propto (1/r) \text{ or } f \cdot r = k$$

or, representing frequencies as probabilities  $P_r$ , we have

$$\log P_r = \log c - \log r$$

where  $c = r \cdot P_r$ .

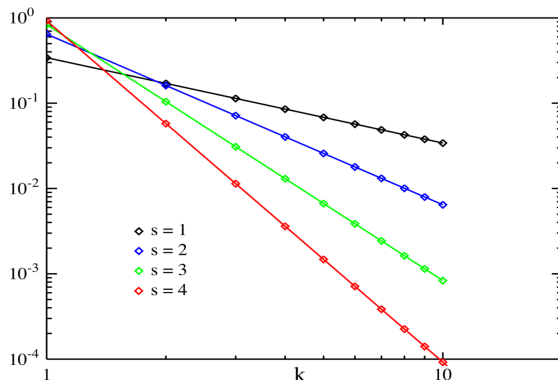
# Zipf's law



**Figure:** A typical Zipf-law rank distribution. The y-axis represents occurrence frequency, and the x-axis represents rank (highest at the left)

**Source:** <http://planetmath.org/encyclopedia/ZipfsLaw.html>

# Zipf's law



**Figure:** Zipf Probability Mass Function for  $N = 10$  on a log-log scale. The horizontal axis is the index  $k$

**Source:** [http://en.wikipedia.org/wiki/Zipf's\\_law](http://en.wikipedia.org/wiki/Zipf's_law)

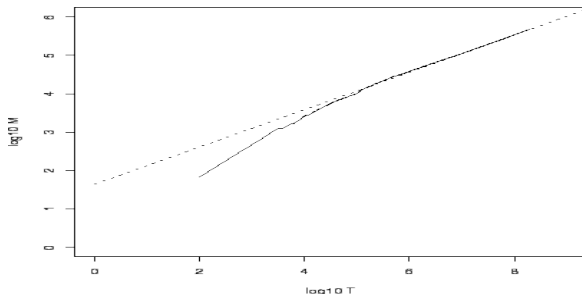
# Heap's Law and Vocabulary Growth

Relates the size of the vocabulary to the size of the corpus. For a corpus of size  $n$  tokens, the size of the vocabulary is  $v$  tokens where,

$$v = k \cdot n^{\beta}$$

Typical values for the parameters  $k$  and  $\beta$  are  $30 \leq k \leq 100$  and  $\beta \approx 0.5$ . The Heap's law predicts that the vocabulary grows rapidly at first, but then grows at a slower rate as the corpus grows large.

# Heap's Law and Vocabulary Growth



**Figure:** Heaps' law. Vocabulary size as a function of collection size (number of tokens) for Reuters-RCV1

## Source:

<http://nlp.stanford.edu/IR-book/html/htmledition/heaps-law-estimating-the-number-of-terms-1.html>

- ▶ Creation of a Document Object Model (DOM) representation using the tags in the document (if it is tagged)
- ▶ Tokernising:
  - ▶ small words and acronyms (xp, gm, ...)
  - ▶ hyphenated words (e-bay, active-x, cd-tom, ...)
  - ▶ special characters (in URLs, ...)
  - ▶ capitalised words (headings in news, ...)
  - ▶ apostrophes (can't, shriner's, ...)
  - ▶ numbers in words (windows 2000, nokia 2250, ...)
  - ▶ periods in abbreviations and numbers (Ph.D, version 2.1, ...)

# Retrieval Models

- ▶ **Boolean Retrieval:** Exact-match retrieval using Boolean operations and, or, not, and true, and false, with support for regular expressions and proximity operators. Very efficient, but the effectiveness depends on the user's skills and experience
- ▶ **Vector Space Model:** Each document is represented geometrically as a point in a term-space. A query is also represented in the same space. The retrieval task then reduces to one of finding a document that is as similar to the query as possible. The documents in the collection can be ranked by relevance in terms of similarity.
- ▶ **Probabilistic Models:** Computes the probability of a document's relevance to a user query, based on the information provided to the model. It then sorts the documents in descending order of relevance.

# Vector Space Model

Suppose there are  $t$  terms and  $n$  documents. The document frequencies can be represented by an  $n \times t$  document-terms matrix. This matrix can be represented in the following two ways:

- ▶ **Term Space:** each document can be represented as a point in a  $t$  dimensional vector (term) space, or
- ▶ **Document Space:** each term can be represented as a point in an  $n$  dimensional (document) vector space.

These two representations are duals of each other.

# Vector Space Model

A Query  $q = (q_1, q_2, \dots, q_t)$  can also be represented as vectors in the term space where the  $t$  components of  $q$  represent the weights.

The retrieval task here reduces to finding documents (points) which are, in some ways, closest to the query.

Usually, a similarity measure such as the cosine correlation measure that measures the cosine of the angle between the query and the document vectors is used.

# Vector Space Model: Weights

The greater the frequency of a term in a document, the greater the importance of the term in that document. On the other hand, the greater the number of documents in which a term appears, the lesser the discriminating power of the term for retrieval.

Suppose  $f_{ik}$  is the frequency of term  $t_k$  in document  $d_i$ . The normalised frequency of the term  $t_k$  in document  $d_i$  is given by

$$tf_{ik} = \frac{f_{ik}}{\sum_{j=1}^t f_{ij}}$$

And the weights representing such inverse document frequency is given by

$$idf_k = \log \frac{n}{n_k}$$

where  $idf_k$  is the weight for inverse document frequency for term  $t_k$ ,  $n$  is the number of documents, and  $n_k$  is the number of documents in which term  $t_k$  appears. This can be given an *information theoretic* explanation.

# Vector Space Model: Weights

The weights based on the term frequencies and inverse document frequencies can be combined to get the following weights:

$$d_{ik} = \frac{(\log(f_{ik})+1) \cdot \log(N/n_k)}{\sqrt{\sum_{k=1}^t [(\log(f_{ik})+1) \cdot \log(N/n_k)]^2}}$$

# Probabilistic Retrieval Models

- ▶ Probabilistic retrieval models are based on the idea that given a corpus and a query, the documents belong to two sets of documents: relevant and not-relevant. Given a query, retrieval task is one of classification of the documents into one of these two sets.
- ▶ Relevance is a subjective concept, and so estimation of the probability of a document being relevant to a query is difficult
- ▶ However, given a query and a set of terms that should occur frequently in a document if it were relevant, it is relatively easy to determine if a particular document in the corpus is relevant to the query. *Probabilistic models exploit this via the Bayes Theorem to estimate the probability of a document's relevance.*