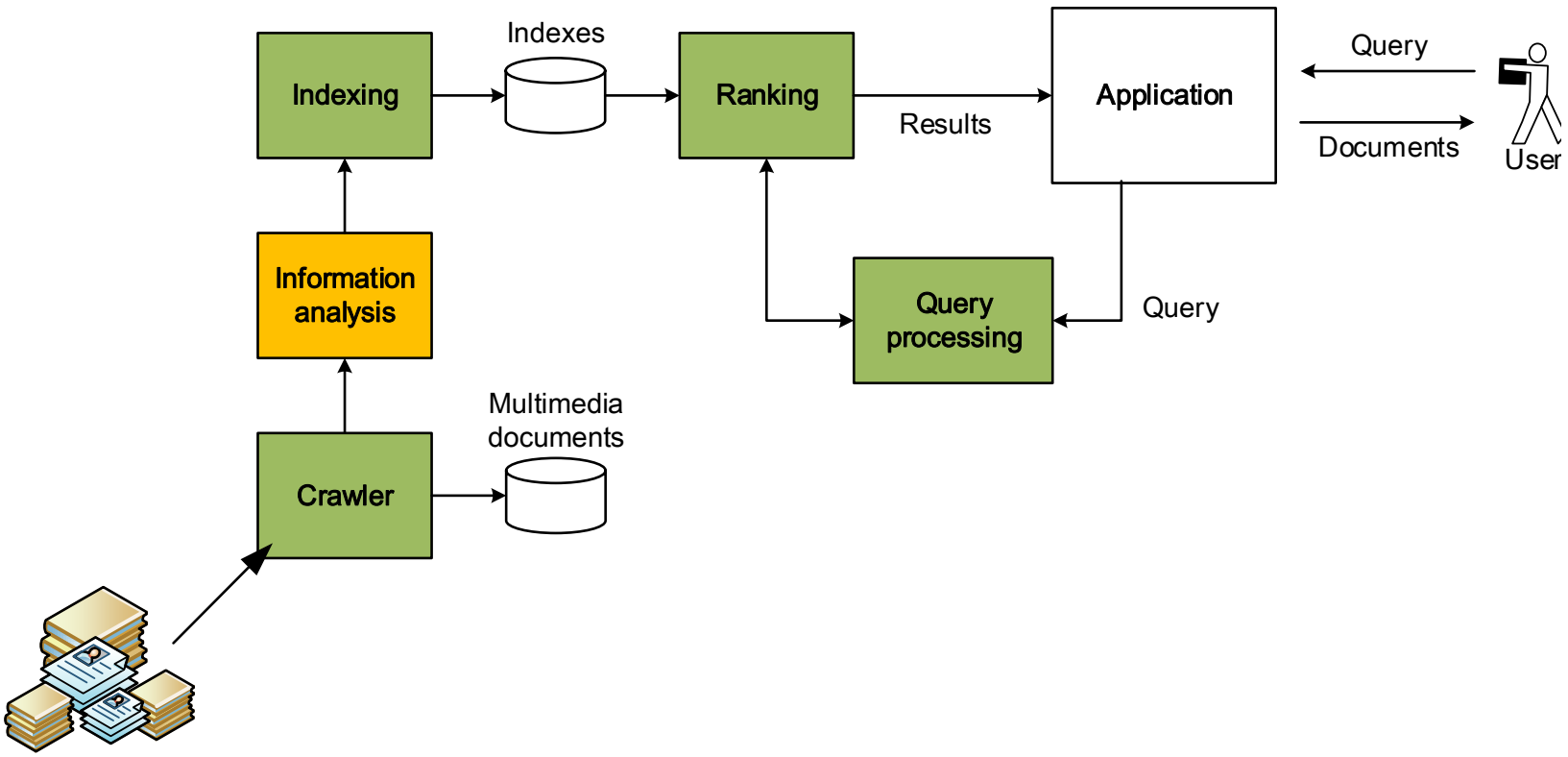
Basic techniques

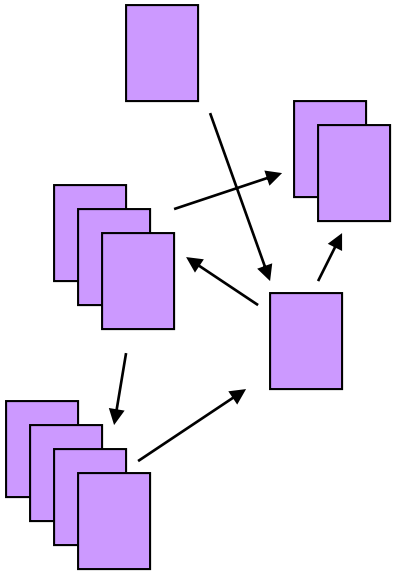
Text processing; term weighting; vector space model; inverted index;

Web Search

Overview


Indexing
Indexes
Ranking
Results
Application
Query
Documents
User
Information analysis
Query processing
Query
Multimedia documents
Crawler

The Web corpus

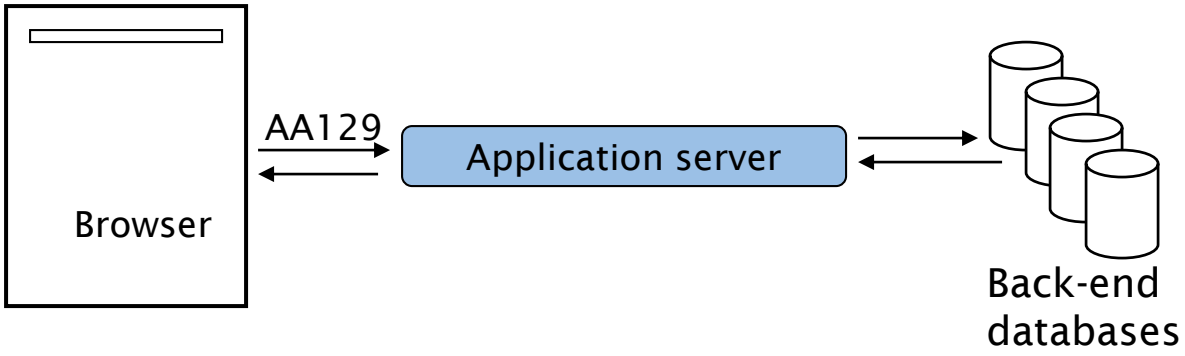


- No design/coordination
- Distributed content creation, linking, democratization of publishing
- Content includes truth, lies, obsolete information, contradictions …
- Unstructured (text, html, …), semi-structured (XML, annotated photos), structured (Databases)…
- Scale much larger than previous text corpora… but corporate records are catching up.
- Content can be dynamically generated

The Web: Dynamic content

- A page without a static html version
  - E.g., current status of flight AA129
  - Current availability of rooms at a hotel

- Usually, assembled at the time of a request from a browser
  - Typically, URL has a '?' character in it

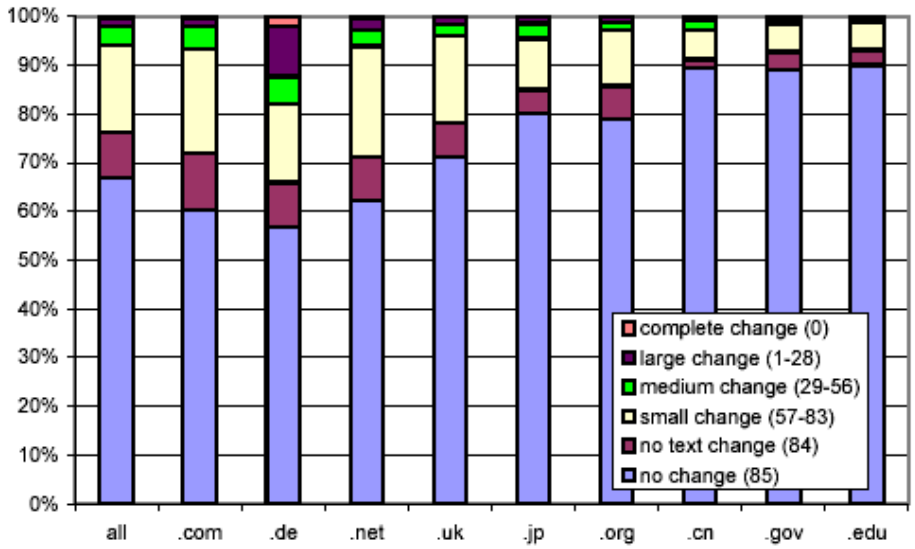


Dynamic content

- Most dynamic content is ignored by web spiders
  - Many reasons including malicious spider traps
- Some dynamic content (news stories from subscriptions) are sometimes delivered as dynamic content
  - Application-specific spidering
- Spiders commonly view web pages just as Lynx (a text browser) would
- Note: even “static” pages are typically assembled on the fly (e.g., headers are common)

Index updates: rate of change

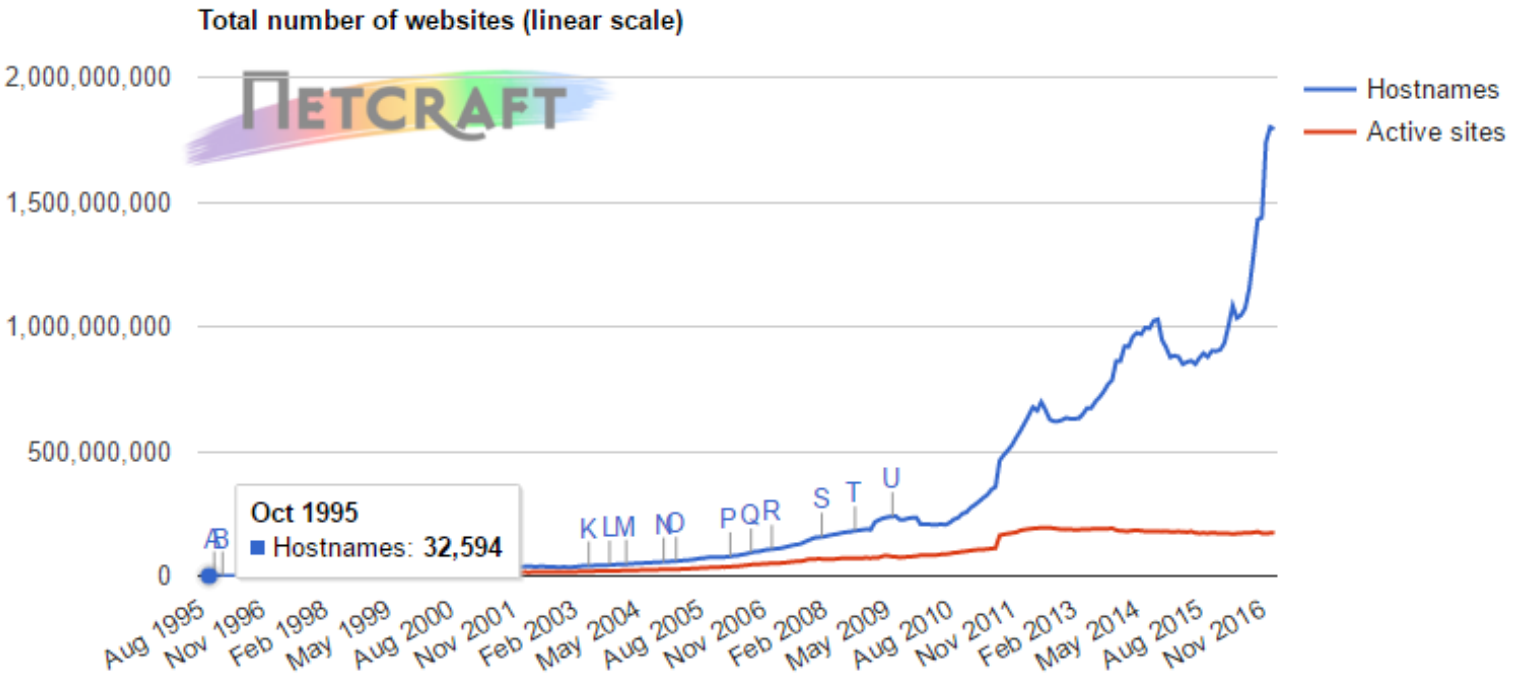
- Fetterly et al. study (2002): several views of data, 150 million pages over 11 weekly crawls
  - Bucketed into 85 groups by extent of change



What is the size of the web?

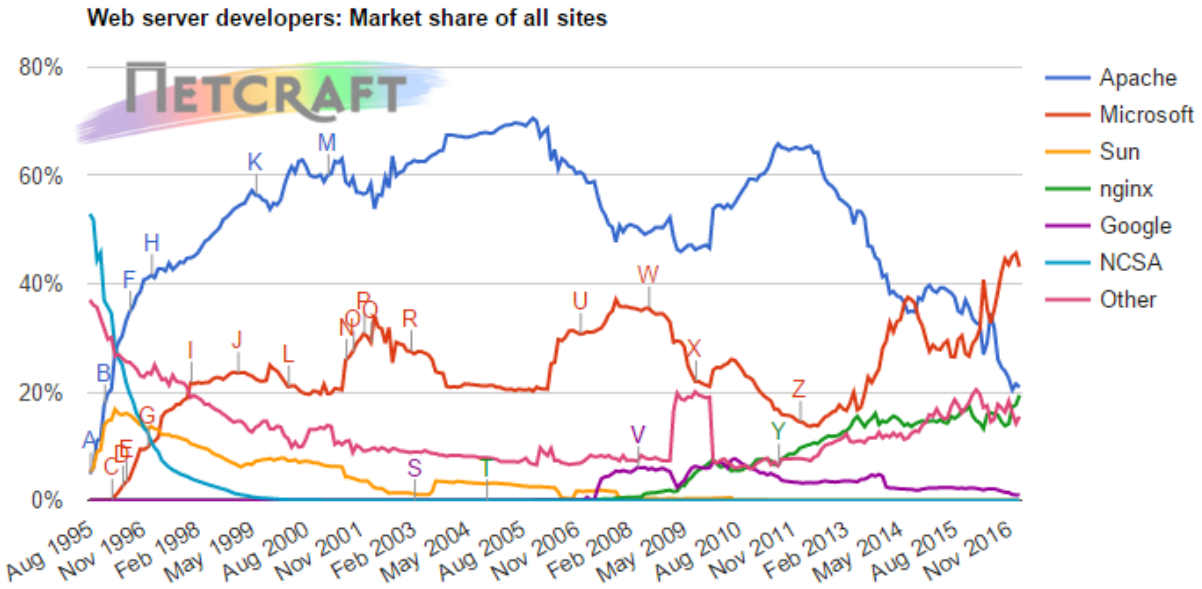
- What is being measured?
  - Number of hosts
  - Number of (static) html pages
    - Volume of data
- Number of hosts – netcraft survey
  - http://news.netcraft.com/archives/web_server_survey.html
  - Monthly report on how many web hosts & servers are out there
- Number of pages – numerous estimates

Total Web sites



http://news.netcraft.com/

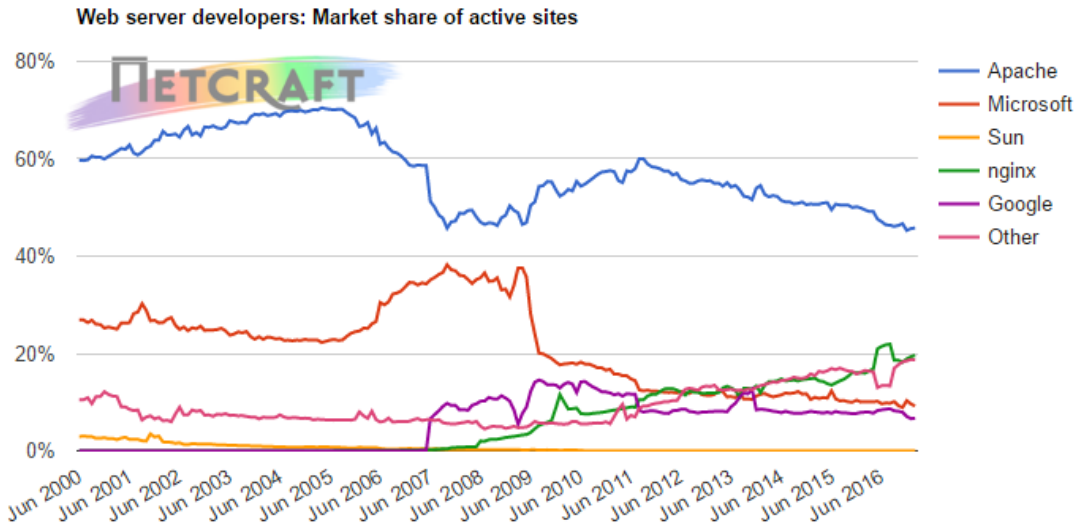
Market share for top servers



Developer	January 2017	Percent	February 2017	Percent	Change
Microsoft	821,905,283	45.66%	773,552,454	43.16%	-2.50
Apache	387,211,503	21.51%	374,297,080	20.89%	-0.63
nginx	317,398,317	17.63%	348,025,788	19.42%	1.79
Google	17,933,762	1.00%	18,438,702	1.03%	0.03

http://news.netcraft.com/

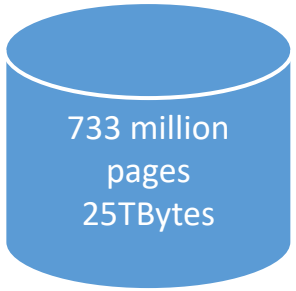
Market share for active sites



Developer	January 2017	Percent	February 2017	Percent	Change
Apache	78,707,037	45.67%	79,593,938	45.78%	0.11
nginx	33,331,358	19.34%	34,088,228	19.60%	0.27
Microsoft	16,601,302	9.63%	16,031,854	9.22%	-0.41
Google	11,372,796	6.60%	11,656,739	6.70%	0.11

Clue Web 2012

- Most of the web documents were collected by five instances of the Internet Archive's Heritrix web crawler at Carnegie Mellon University
  - Running on five Dell PowerEdge R410 machines with 64GB RAM.
  - Heritrix was configured to follow typical crawling guidelines.
- The crawl was initially seeded with 2,820,500 uniq URLs.
  - This is a small sample of the Web.
- Even at this “low-scale” there are many challenges:
  - How can we store it and make it searchable?
  - How can we refresh the search index?



Basic techniques

1. Text pre-processing
2. Terms weighting
3. Vector Space Model
4. Indexing
5. Web page segments

1. Text pre-processing

- Instead of aiming at fully understanding a text document, IR takes a pragmatic approach and looks at the most elementary textual patterns
  - e.g. a simple histogram of words, also known as “bag-of-words”.
- Heuristics capture specific text patterns to improve search effectiveness
  - Enhances the simplicity of word histograms
- The most simple heuristics are stop-words removal and stemming

Character processing and stop-words

- Term delimitation
- Punctuation removal
- Numbers/dates
- Stop-words: remove words that are present in all documents
  - *a, and, are, as, at, be, but, by, for, if, in, into, is, it, no, not, of, on, or, such, that, the, their, then, there, these, they, this, to, was, will…*

Stemming and lemmatization

- Stemming: Reduce terms to their "roots" before indexing
  - "Stemming" suggest crude affix chopping
  - **e.g., automate(s), automatic, automation all reduced to automat.**
    - http://tartarus.org/~martin/PorterStemmer/
    - http://snowball.tartarus.org/demo.php

- Lemmatization: Reduce inflectional/variant forms to base form, e.g.,
  - *am, are, is* $\rightarrow$ *be*
  - *car, cars, car's, cars'* $\rightarrow$ *car*

N-grams

- An n-gram is a sequence of items, e.g. characters, syllables or words.
- Can be applied to text spelling correction
  - *“interactive meida” >>>> “interactive media”*
- Can also be used as indexing tokens to improve Web page search
  - You can order the Google n-grams (6DVDs):
    - http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html
- N-grams were under some criticism in NLP because they can add noise to information extraction tasks
  - ...but are widely successful in IR to infer document topics.

Tolerant retrieval

- Wildcards
  - Enumerate all *k*-grams (sequence of *k* chars) occurring in any term
  - Maintain a *second* inverted index *from bigrams to dictionary terms* that match each bigram.
- Spelling corrections
  - Edit distances: Given two strings S_1 and S_2, the minimum number of operations to convert one to the other
- Phonetic corrections
  - Class of heuristics to expand a query into phonetic equivalents

Documents representation

- After the text analysis steps, a document (e.g. Web page) is represented as a vector of terms, n-grams, (PageRank if a Web page), etc. :
  - previou step document eg web page repres vector term ngram pagerank web page etc

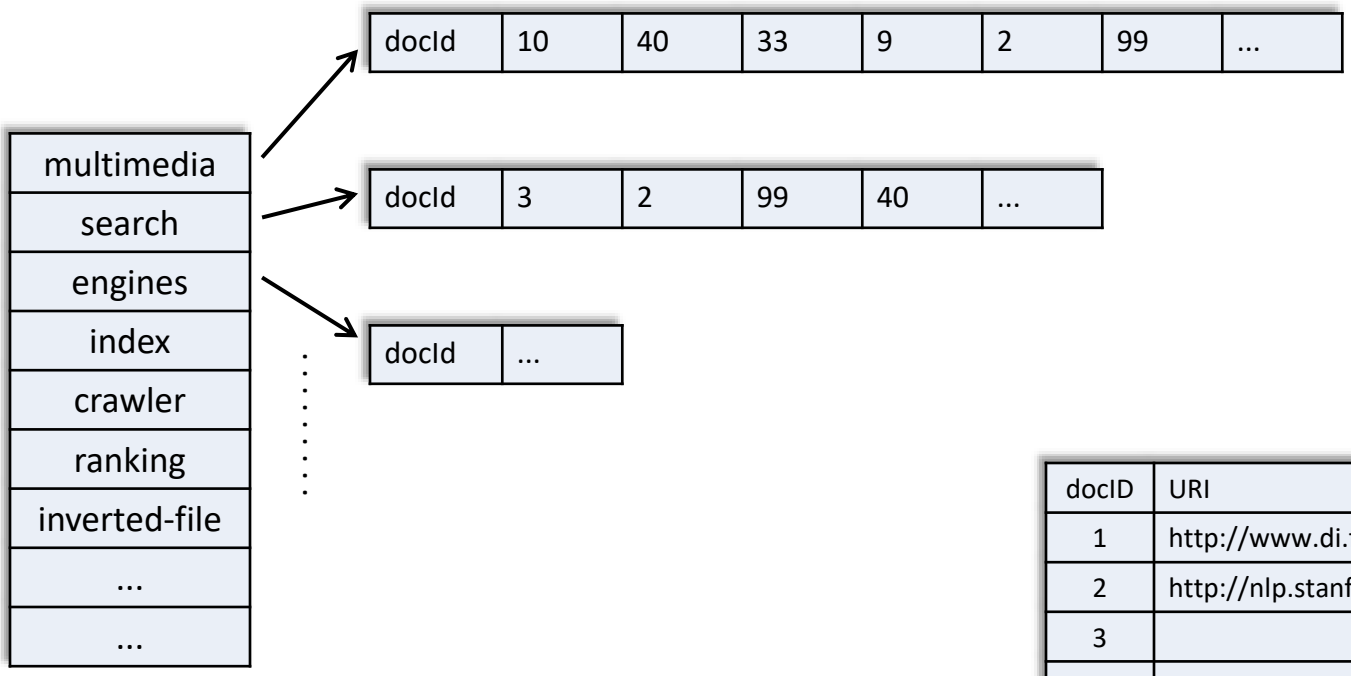
$$d = (w_1, \dots, w_L, ng_1, \dots, ng_M, PR, \dots)$$

Comparision of text parsing techniques

Table 3.3 Comparison of tokenization techniques for several European languages (based on McNamee, 2008). The table lists mean average precision values for tokenizations based on unstemmed words, the Snowball stemmer, character 4-grams, and character 5-grams.

Language	Words	Stemmer	4-grams	5-grams
Dutch	0.416	0.427	0.438	0.444
English	0.483	0.501	0.441	0.461
Finnish	0.319	0.417	0.483	0.496
French	0.427	0.456	0.444	0.440
German	0.349	0.384	0.428	0.432
Italian	0.395	0.435	0.393	0.422
Spanish	0.427	0.467	0.447	0.438
Swedish	0.339	0.376	0.424	0.427

Index for boolean retrieval



docID	URI
1	http://www.di.fct.unl...
2	http://nlp.stanford.edu/IR-book/...
3	...
4	...
...	...

2. Term weighting

- Boolean retrieval looks for terms overlap
- What's wrong with the overlap measure?
- It doesn't consider:
  - Term frequency in document
  - Term scarcity in collection (document mention frequency)
    - ***of*** is more common than ***ideas*** or ***march***
  - Length of documents
    - (And queries: score not normalized)

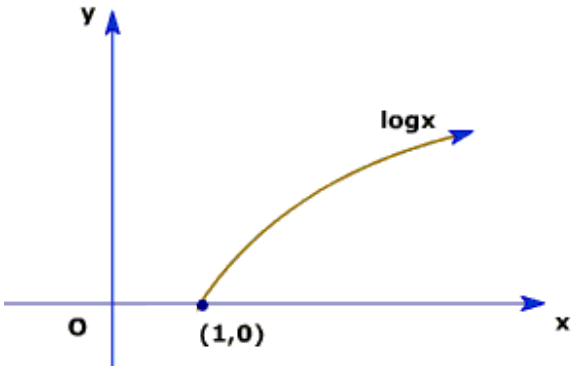
Term weighting

- Term weighting tries to reflect the importance of a document for a given query term.
- Term weighting must consider two aspects:
  - The frequency of a term in a document
  - Consider the rarity of a term in the repository
- Several weighting intuitions were evaluated throughout the years:
  - Salton, G. and Buckley, C. 1988. **Term-weighting approaches in automatic text retrieval**. *Inf. Process. Manage.* 24, 5 (Aug. 1988), 513-523.
  - Robertson, S. E. and Sparck Jones, K. 1988. **Relevance weighting of search terms**. In *Document Retrieval Systems*, P. Willett, Ed. Taylor Graham Series In Foundations Of Information Science, vol. 3.

TF-IDF: Term Freq.-Inverted Document Freq.

- Text terms should be weighted according to
  - their importance for a given document: $tf_i(d) = n_i(d)$
  - and how rare a word is: $idf_i = log\frac{|D|}{df(t_i)}$ $df(t_i) = |\{d: t_i \in d\}|$
- The final term weight is: $w_{i,j} = \mathrm{tf_i}(\mathrm{d_j}) \cdot idf_i$

Inverse document frequency



$$idf_i = log \frac{|D|}{df(t_i)} \qquad df(t_i) = |\{d: t_i \in d\}|$$

3. Vector Space Model

- Each doc d can now be viewed as a vector of tf $\times$ idf values, one component for each term

- So, we have a vector space where:
  - terms are axes
  - docs live in this space
  - even with stemming, it may have 50,000+ dimensions

- First application: Query-by-example
  - Given a doc d, find others “like” it.

- Now that d is a vector, find vectors (docs) “near” it.

Salton, G., Wong, A., and Yang, C. S. 1975. A vector space model for automatic indexing. Commun. ACM 18, 11 (Nov. 1975).

Documents and queries

- Documents are represented as an histogram of terms, n-grams and other indicators:

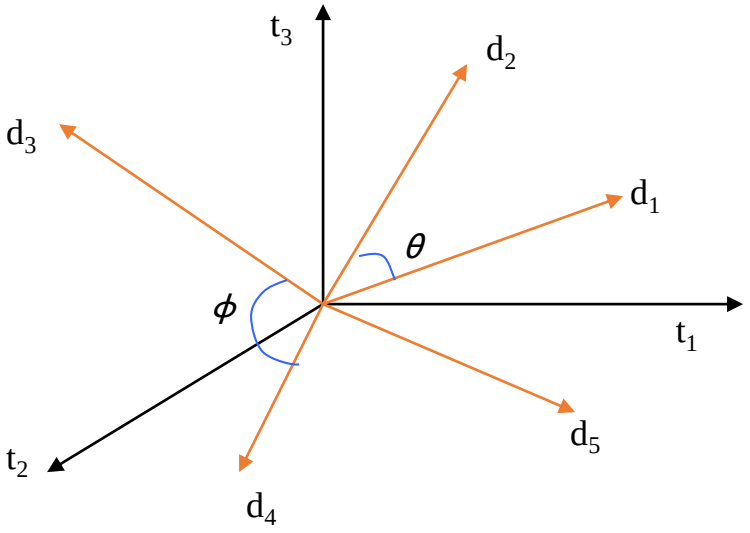
$$d = (w_1, \dots, w_L, ng_1, \dots, ng_M, PR, \dots)$$

- The text query is processed with the same text pre-processing techniques.
  - A query is then represented as a vector of text terms and n-grams (and possibly other indicators):

$$\mathrm{q} = (w_1, \dots, w_L, ng_1, \dots, ng_M)$$

Intuition

- If d_1 is near d_2, then d_2 is near d_1.
- If d_1 near d_2, and d_2 near d_3, then d_1 is not far from d_3.
- No doc is closer to d than d itself.
  - Postulate: Documents that are "close together" in the vector space talk about the same things.



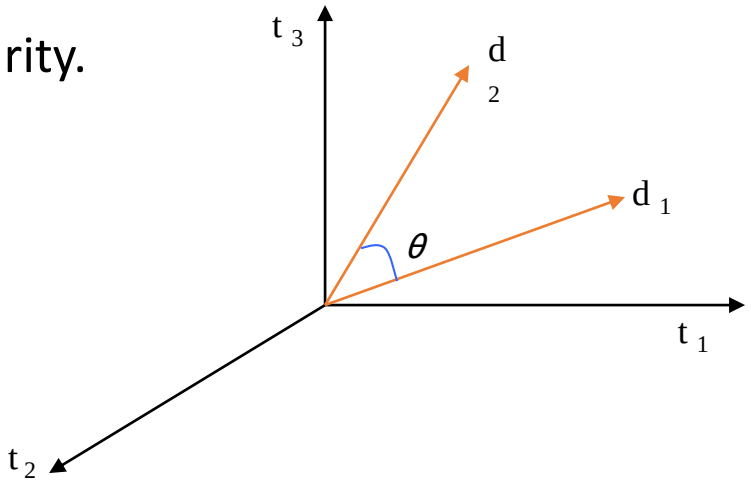
First cut

- Idea: Distance between d_1 and d_2 is the length of the vector $|d_1 - d_2|$.
  - Euclidean distance
- Why is this not a great idea?
- We still haven't dealt with the issue of length normalization
  - Short documents would be more similar to each other by virtue of length, not topic
- However, we can implicitly normalize by looking at angles instead

Angle as a similarity

- Distance between vectors d_1 and d_2 captured by the cosine of the angle x between them.
  - Vectors pointing in the same direction

- Note – this is similarity, not distance
  - No triangle inequality for similarity.



Vectors normalization

- A vector can be normalized (given a length of 1) by dividing each of its components by its length – here we use the L_2 norm

$$\|x\|_2 = \sqrt{\sum_i x_i^2}$$

- This maps vectors onto the unit sphere. Then, $|d_j| = \sqrt{\sum_{i=1}^{n} w_{i,j}} = 1$

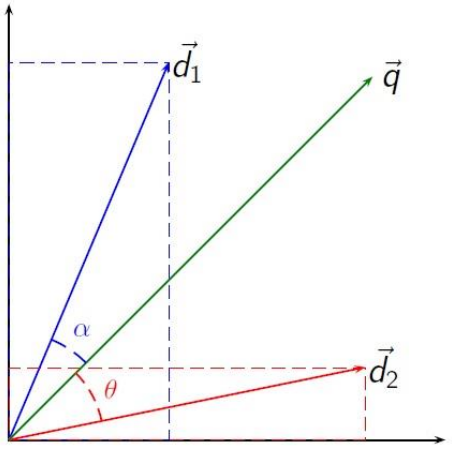
- Longer documents don't get more weight

Cosine similarity

- Cosine of angle between two vectors
- The denominator involves the lengths of the vectors.

$$sim(q, d_i) = \cos(q, d_i) = \frac{q \cdot d_i}{\|q\|\|d_i\|}$$

$$sim(q, d_i) = \cos(q, d_i) = \frac{\sum_t q_t \cdot d_{i,t}}{\sqrt{\sum_t q_t^2}\sqrt{\sum_t d_{i,t}^2}}$$



Improved ranking

- How to enhace/improve the previous model?
  - Positional indexing
    - Documents with distances between query terms greater than n are discarded
    - Distance between query terms in the documents affect rank score
  - Other ranking functions
    - BM-25
    - Bayesian networks
    - Learning to rank

BM-25 ranking function

- Proposed by Robertson et al. in 1994.

$$score(q, d_j) = \sum q_t \cdot \frac{f_{t,d}(k_1 + 1)}{k_1\left((1 - b) + b\left(\frac{l_{avg}}{l_d}\right)\right) + f_{t,d}} \cdot IDF_t$$

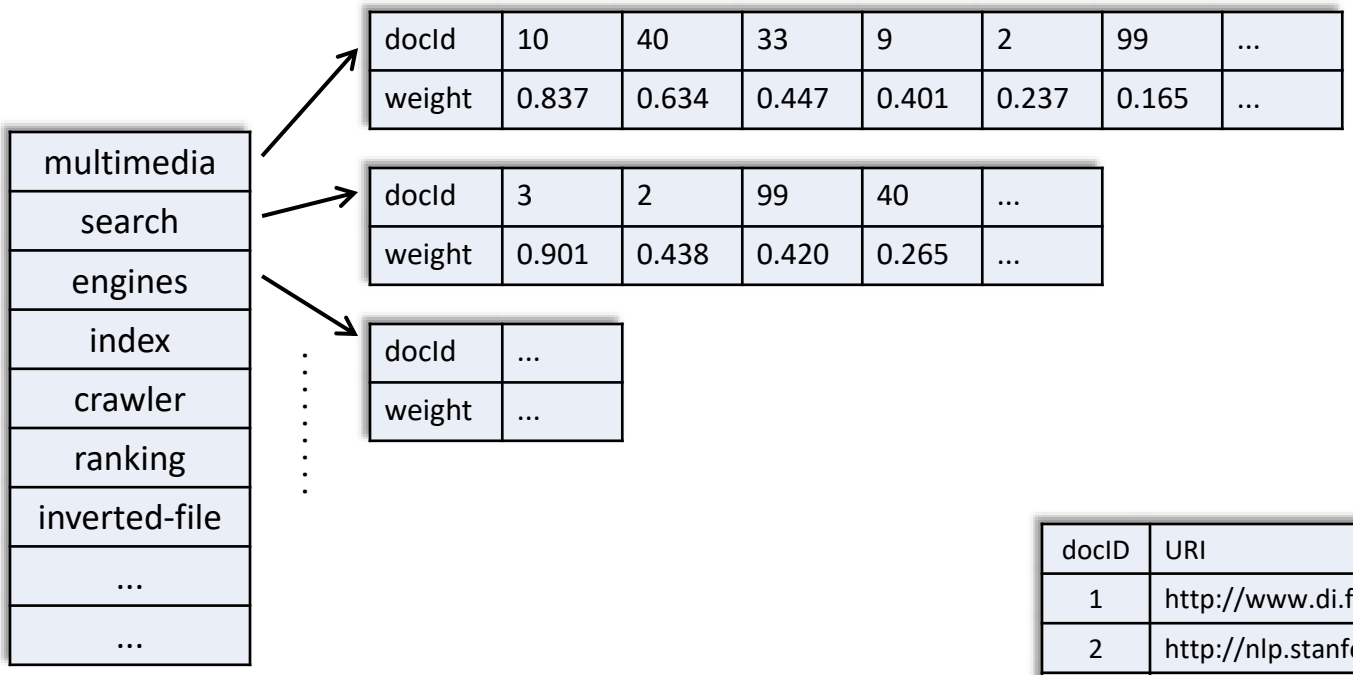
$$IDF(q_i) = \log \frac{N - nd(q_i) + 0.5}{nd(q_i) + 0.5}$$

Chapter 11: "**Introduction to Information Retrieval**", Cambridge University Press, 2008

4. Indexing

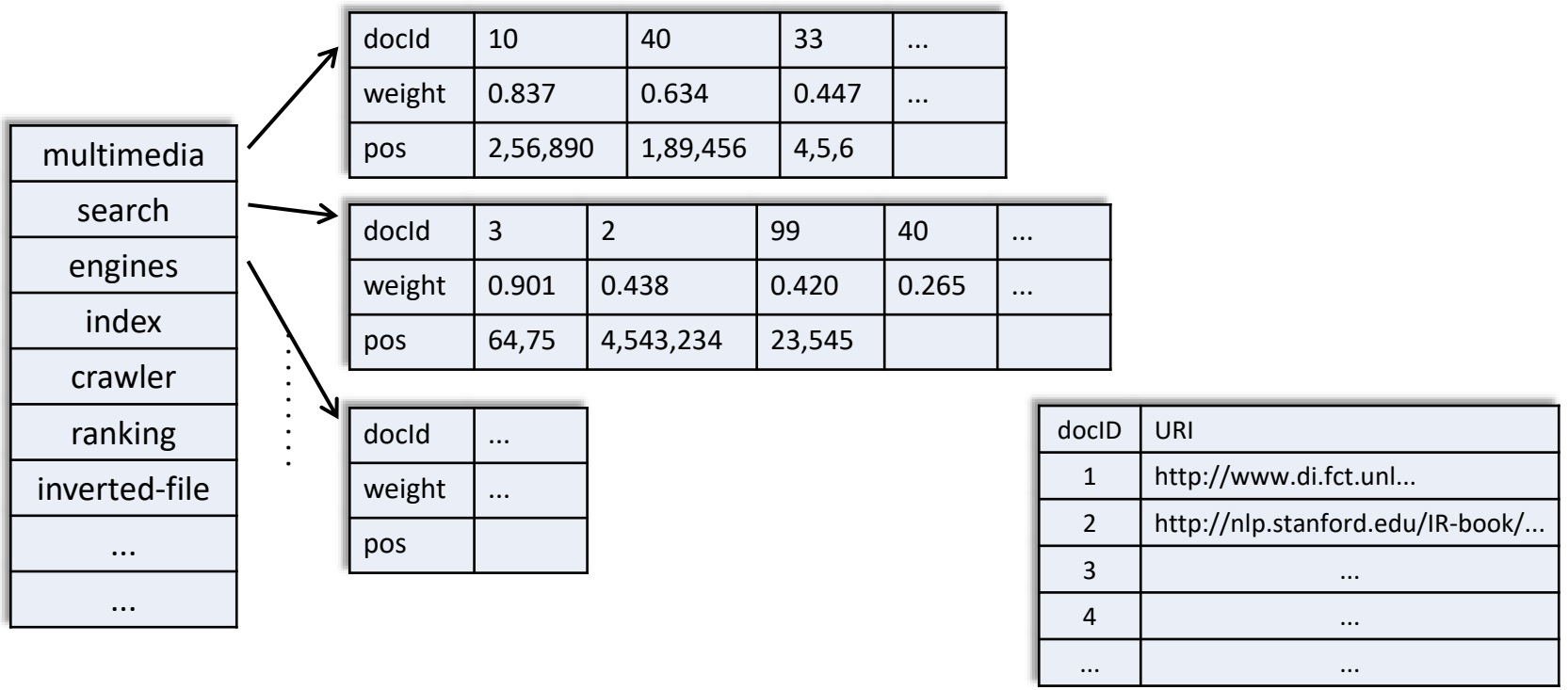
- This stage creates an index to quickly locate relevant documents
- An index is an aggregation of several data structures (e.g. B-trees or hash tables)
- Index compression is used to reduce the amount of space and the time needed to compute similarities
- The distribution of the index pages across a cluster improves the search engine responsiveness

Inverted file index v2



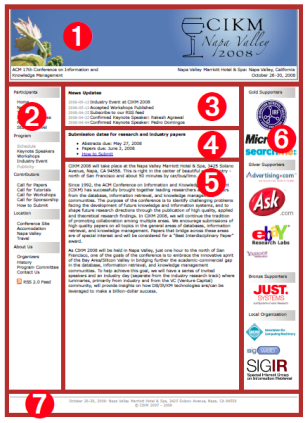
docID	URI
1	http://www.di.fct.unl...
2	http://nlp.stanford.edu/IR-book/...
3	...
4	...
...	...

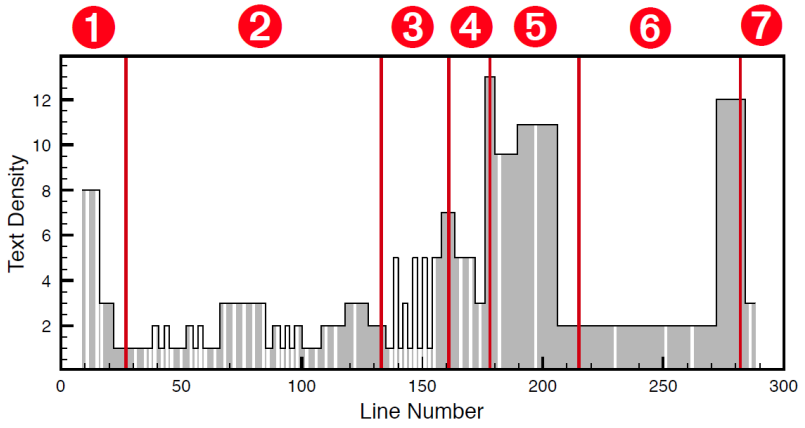
Inverted file index v3



5. Web page segments

- Web pages are divided into different parts (title, abstract, body, etc)
- Each part has a specific relevance to the main content
- A Web page can be divided by its HTML structure (e.g., tags) or by its visual aspect.





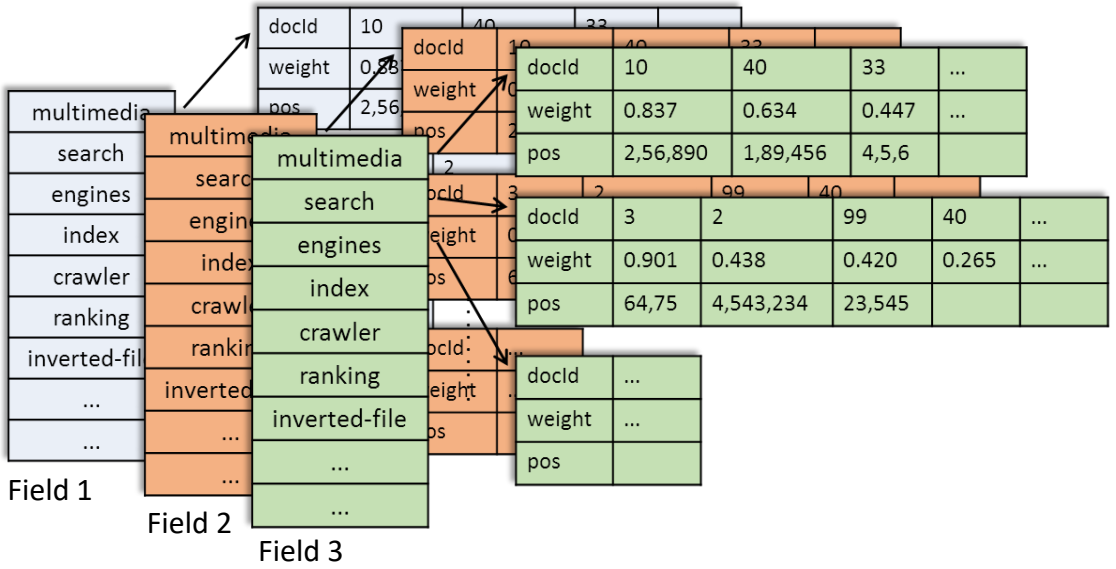
Web page segmentation methods

- Segmenting visually
  - Cai, D., Yu, S., Wen, J. R., & Ma, W. Y. (2003). VIPS: A vision-based page segmentation algorithm.
- Linguistic approach
  - Kohlschütter, C. , Fankhauser, P., and Nejdl, W. (2010). Boilerplate detection using shallow text features. ACM Web Search and Data Mining.
- Densitometric approach
  - Kohlschütter, C., and Nejdl, W., (2008). A densitometric approach to web page segmentation. ACM Conference on Information and Knowledge Management (CIKM '08).

https://boilerpipe-web.appspot.com/ https://github.com/kohlschutter/boilerpipe

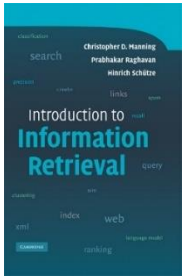
Multiple indexes

- Title, abstract, corpus, sections are all indexed separately
- Each zone index is inspected for the each query term
- The final rank is a linear combination of the multiple ranks
  - Discussed in class 10.



Summary of basic techniques

- Text processing
  - Text representation, stop words, stemming, lemmatization
- Term weighting
  - Term weighting
- Vector space model
  - TF-IDF and cosine distance
- Inverted index
- Web page segments



Chapter 2, 6