Introduction to Media Informatics: Information Retrieval

Dr. Mathias Lux
Associate Professor
Alpen-Adria Universität Klagenfurt

Exam

- Jan 22nd, 2016, 2pm
- 60 minutes
- multiple choice + a few open questions.

- pre-exam Q&A Jan 18th, 3-5pm on Hangout

Information Retrieval History

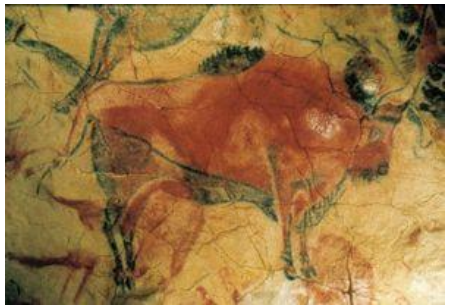
IR is the process of **searching** through a **document collection** based on a particular **information need**.

IR Key Concepts

- Searching
  - Indexing, Ranking
- Document Collection
  - Textual, Visual, Auditive
- Particular Needs
  - Query, User based

A History of Libraries

Libraries are perfect examples for document collections.



- Wall paintings in caves
  - e.g. Altamira, ~ 18,500 years old
- Writing in clay, stone, bones
  - e.g. Mesopotamian cuneiforms, ~ 4.000 BC
  - e.g. Chinese tortoise-shell carvings, ~ 6.000 BC
  - e.g. Hieroglyphic inscriptions, Narmer Palette ~ 3.200 BC

A History of Libraries (ctd.)

- Papyrus
  - Specific plant (subtropical)
  - Organized in rolls, e.g. in Alexandria
- Parchment
  - Independence from papyrus
  - Sewed together in books
- Paper
  - Invented in China (bones and bamboo too heavy, silk too expensive)
  - Invention spread -> in 1120 first paper mill in Europe

A History of Libraries (ctd.)

- Gutenberg's printing press (1454)
  - Inexpensive reproduction
  - e.g. "Gutenberg Bible"
- Organization & Storage
  - Dewey Decimal System (DDC, 1872)
  - Card Catalog (early 1900s)
  - Microfilm (1930s)
  - MARC (Machine Readable Cataloging, 1960s)
  - Digital computers (1940s+)

Library & Archives Today

- Partially converted to electronic catalogues
  - From a certain time point on (1992 - ...)
  - Often based on proprietary systems
  - Digitization happens slow
  - No full text search available
  - Problems with preservation
    - Storage devices & formats

History of Searching

- Browsing
  - Like "Finding information yourself"
- Catalogs
  - Organized in taxonomies, keywords, etc.
- Content Based Searching
  - `SELECT * FROM books WHERE title='%Search%'`
- Information Retrieval
  - Ranking, models, weighting
  - Link analysis, LSA, ...

History of IR

- Starts with development of computers
- Term "Information Retrieval" coined by Mooers in 1950
  - Mooers, C. (March 1950). "The theory of digital handling of non-numerical information and its implications to machine economics". *Proceedings of the meeting of the Association for Computing Machinery at Rutgers University*.

- Two main periods (Spark Jones u. Willett)
  - 1955 – 1975: Academic research
    - Models and Basics
    - Main Topics: Search & Indexing
  - 1975 – … : Commercial applications
    - Improvement of basic methods

A Challenge: The World Wide Web

- First actual implementation of **Hypertext**
  - Interconnected documents
  - Linked and referenced
- World Wide Web (1989, T. Berners-Lee)
  - Unidirectional links (target is not aware)
  - Links are not typed
  - Simple document format & communication protocol (HTML & HTTP)
  - Distributed and not controlled

Some IR History Milestones

- Book "Automatic Information Organization and Retrieval", *Gerard Salton* (1968)
  - Vector Space Model
- Paper "A statistical interpretation of term specificity and its application in retrieval", Karen Sparck Jones (1972)
  - IDF weighting
  - http://www.soi.city.ac.uk/~ser/idf.html
- Book "Information Retrieval" of *C.J. Rijsbergen* (1975)
  - Probabilistic Model
  - http://www.dcs.gla.ac.uk/Keith/Preface.html

Some IR History Milestones

- Paper "Indexing by Latent Semantic Analysis", S. Deerwester, Susan Dumais, G. W. Furnas, T. K. Landauer, R. Harshman (1990).
  - Latent Semantic Indexing
- Paper "Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval" Robertsen & Walker (1994)
  - BM25 weighting scheme
- Paper "The Anatomy of a Large-Scale Hypertextual Web Search Engine", *Sergey Brin & Larry Page* (1998)
  - World Wide Web Retrieval

Agenda

- Information Retrieval History
- **Information Retrieval & Data Retrieval**
- Searching & Browsing
- Information Retrieval Models
- Web Retrieval

Information Retrieval & Data Retrieval

Information Retrieval

- Information Level
- Search Engine
- Bing / Google

Data Retrieval

- Data Level
- Data Base
- Oracle / MySQL

Information Retrieval & Data Retrieval

Information Retrieval	Data Retrieval
Content Based Search	Search for Patterns and String
Query ambigous	Query formal & unambigous
Results ranked by relevance	Results not ranked
Error tolerant	Not error tolerant
Multiple iterations	Clearly defined result set
Examples	*Examples*
Search for synonyms	Search for patterns
Bag of Words	SQL Statement

- Retrieval is nearly always a combination of both.

Agenda

- Information Retrieval History
- Information Retrieval & Data Retrieval
- **Searching & Browsing**
- Information Retrieval Models
- Web Retrieval

Information Retrieval Basics: Searching

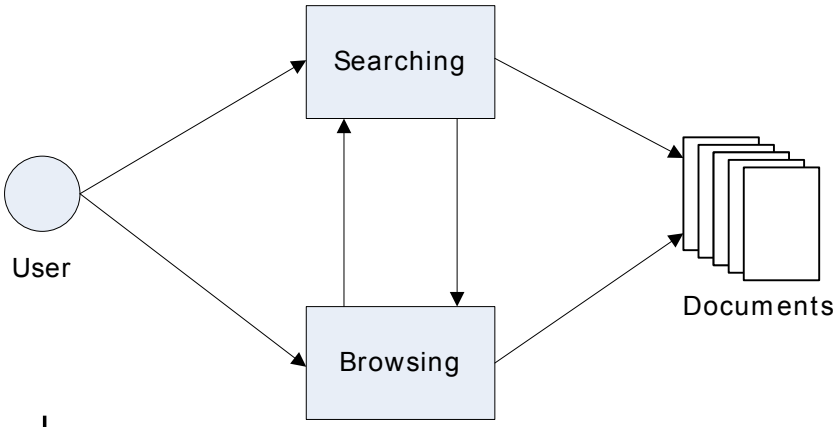
A **user** has an **information need**, which needs to be **satisfied**.

- Two different approaches:
  - Browsing
  - Searching

Searching & Browsing

Searching

- Explicit information need
- Definition through "query"
- Result lists
- e.g. Google



Browsing

- Not necessarily explicit need
- Navigation through repositories

Browsing

- Flat Browsing
  - User navigates through set of documents
  - No implied ordering, explicit ordering possible
  - Examples: One single directory, one single file
- Structure Guided Browsing
  - An explicit structure is available for navigation
  - Mostly hierarchical (file directories)
  - Can be generic digraph (WWW)
  - Examples: File systems, World Wide Web

Searching

- Query defines "Information Need"

- Ad Hoc Searching
  - Search when you need it
  - Query is created to fit the need
- Information Filtering
  - Make sets of documents smaller
  - Query is filter criterion
- Information Push
  - Same as filtering, delivery is different

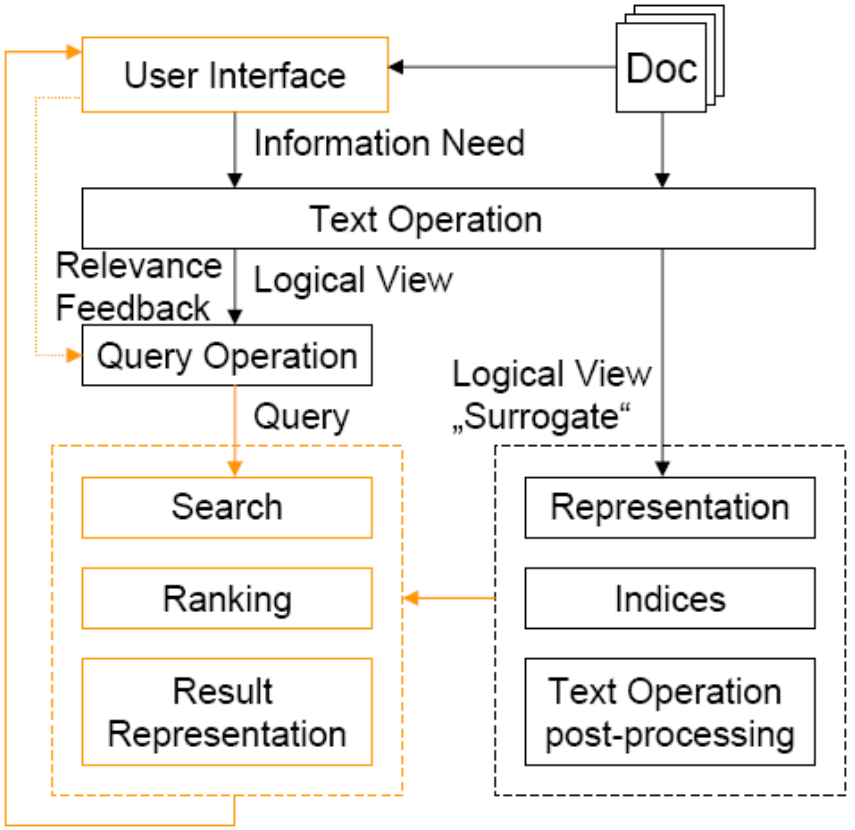
Agenda

- Information Retrieval History
- Information Retrieval & Data Retrieval
- Searching & Browsing
- **Information Retrieval Models**
- Web Retrieval

Information Retrieval System Architecture

Aspects

- Query & languages
- IR models
- Documents
- Internal representation
- Pre- and post-processing
- Relevance feedback
- HCI



Information Retrieval Models

- Boolean Model
  - Set theory & Boolean algebra
- Vector Model
  - Non binary weights on dimensions
  - Partial match
- Probabilistic Model
  - Modeling IR in a probabilistic framework

Formal Definition of Models

An information retrieval model is a quadruple $[D, Q, F, R(q_i, d_j)]$

- D is a set of logical views (or representations) for the **documents** in the collection.
- Q is a set of logical views (or representations) for the user needs or **queries**.
- F is a **framework** for modeling document representations, queries and their relationship.
- $R(q_i, d_j)$ is a **ranking function** which associates a real number with a query q_i of Q and a document d_j of D.

Definitions

in Context of Text Retrieval

- **index term** – word of a document expressing (part of) document semantics
- **weight $w_{i,j}$** – quantifies the importance of index term t_i for document d_j
- **index term vector** for document d_j (having t different terms in all documents):

$$\overrightarrow{d_j} = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$$

Boolean Model

- Based on set theory and Boolean algebra
  - Set of index terms
  - Query is Boolean expression
- Intuitive concept:
  - Wide usage in bibliographic system
  - Easy implementation and simple formalisms
- Drawbacks:
  - Binary decision components (true/false)
  - No relevance scale (relevant or not)

Boolean Model: Example

- Example queries
  - cat OR dog
  - cat AND dog
  - lecture AND (multimedia OR media AND informatics)

Boolean Model

- Advantages
  - Clean formalisms
  - Simplicity
- Disadvantages
  - Might lead to too few / many results
  - No notion of **partial match**
  - Sequential ordering of terms not taken into account.

Vector Model

- Integrates the notion of partial match
- Non-binary weights (terms & queries)
- Degree of similarity computed

$$\vec{d}_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$$

$$\vec{q} = (w_{1,q}, w_{2,q}, \ldots, w_{t,q})$$

Vector Model: Example

$$sim(d_j, q) = \frac{\vec{d}_j \bullet \vec{q}}{\left|\vec{d}_j\right| \times \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} w_{i,j} \cdot w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \cdot \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

$\vec{d} = (0.3, 0.4, 0, 0.1, 1)$

$\vec{q} = (1, 0, 0, 0.5, 0)$

$$sim(\vec{d}, \vec{q}) = \frac{1 \cdot 0.3 + 0.1 \cdot 0.5}{\sqrt{0.3^2 + 0.1^2 + 0.1^2 + 1^2} \cdot \sqrt{1 + 0.5^2}} \approx \frac{0.35}{2.24} \approx 0.17$$

Another Example

- Document & Query:
  - D = "The quick brown fox jumps over the lazy dog"
  - Q = "brown lazy fox"

$$sim(d_j, q) = \frac{\vec{d}_j \bullet \vec{q}}{\left|\vec{d}_j\right| \times \left|\vec{q}\right|} = \frac{\sum_{i=1}^{t} w_{i,j} \cdot w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \cdot \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}$$

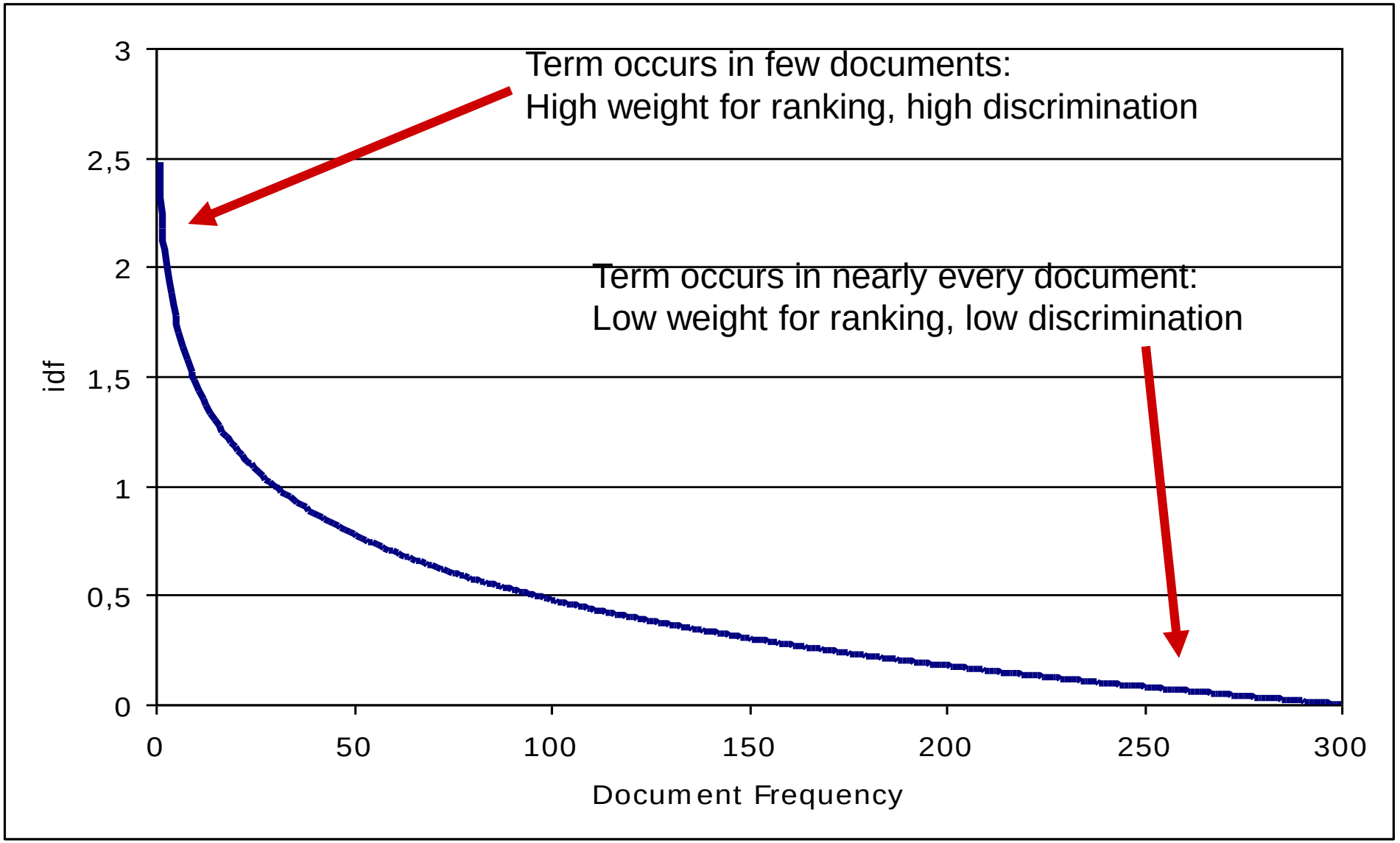
- Results:
  - $(1,1,1,1,1,1,1,2)^t * (1,1,1,0,0,0,0,0)^t = 3$
  - sqrt(11) * sqrt(3) = 5.04
  - Similarity = 3 / 6 = 0.595

Term weighting: TF*IDF

Term weighting increases retrieval performance

- Term frequency
  - How often does a term occur in a document?
  - Most intuitive approach
- Inverse Document Frequency
  - What is the information content of a term for a document collection?
  - Compare to *Information Theory* of Shannon

Example: IDF with 300 documents corpus



Definitions: Normalized Term Frequency

$$f_{i,j} = \frac{freq_{i,j}}{\max_l(freq_{l,j})}$$ ... normalized term frequency

$freq_{i,j}$ ... raw term frequency of term i in document j

- Maximum is computed over all terms in a document
- Terms which are not present in a document have a raw frequency of 0

Definitions: Inverse Document Frequency

$idf_i = \log \frac{N}{n_i}$ ... inverse document frequency for term i

N ... number of documents in the corpus

n_i ... number of document in the corpus which contain term i

- Note that idf_i is independent from the document.
- Note that the whole corpus has to be taken into account.

TF*IDF

- TF*IDF is a very prominent weighting scheme
  - Works fine, much better than TF or Boolean
  - Quite easy to implement

$$w_{i,j} = f_{i,j} \cdot \log \frac{N}{n_i}$$

Vector Model

- Advantages
  - Weighting schemes improve **retrieval performance**
  - Partial matching allows retrieving documents that **approximate query** conditions
  - Cosine coefficient allows **ranked list** output
- Disadvantages
  - Term are assumed to be mutually independent

Simple example (I)

- Scenario
  - Given a **document corpus on birds**: nearly each document (say 99%) contains the word bird
  - someone is searching for a document about sparrow nest construction with a query "**sparrow bird nest construction**"
  - Exactly the document which would satisfy the user needs **does not have the word "bird"** in it.

Simple example (II)

- TF*IDF weighting
  - knows upon the low discrimative power of the term bird
  - The weight of this term is near to zero
  - This term has virtually no influence on the result list.

Agenda

- Information Retrieval History
- Information Retrieval & Data Retrieval
- Searching & Browsing
- Information Retrieval Models
- **Web Retrieval**

Retrieval in the WWW

- General Retrieval is based on content
  - Represented e.g. by terms, keywords …
- What is different with the WWW?
  - Structured text (markup)
  - Hypermedia (links)
  - Heterogeneous formats (gif, pdf, flv, …)
  - Distributed content (access over network)

Web based Retrieval: Challenges

- Working with an enormous amount of data
  - 10 billion pages a 500kB estimated in 01-2004
    - 2 pages / person on the globe
  - 1 trillion unique URLs indexed by Google in 2008
  - 109.5 million top-level domains operated in 2009
- Furthermore there is a **Deep Web**
  - Including the usenet, tor, torrent, non-indexed WWW, ftp, ...

Web based Retrieval: Challenges

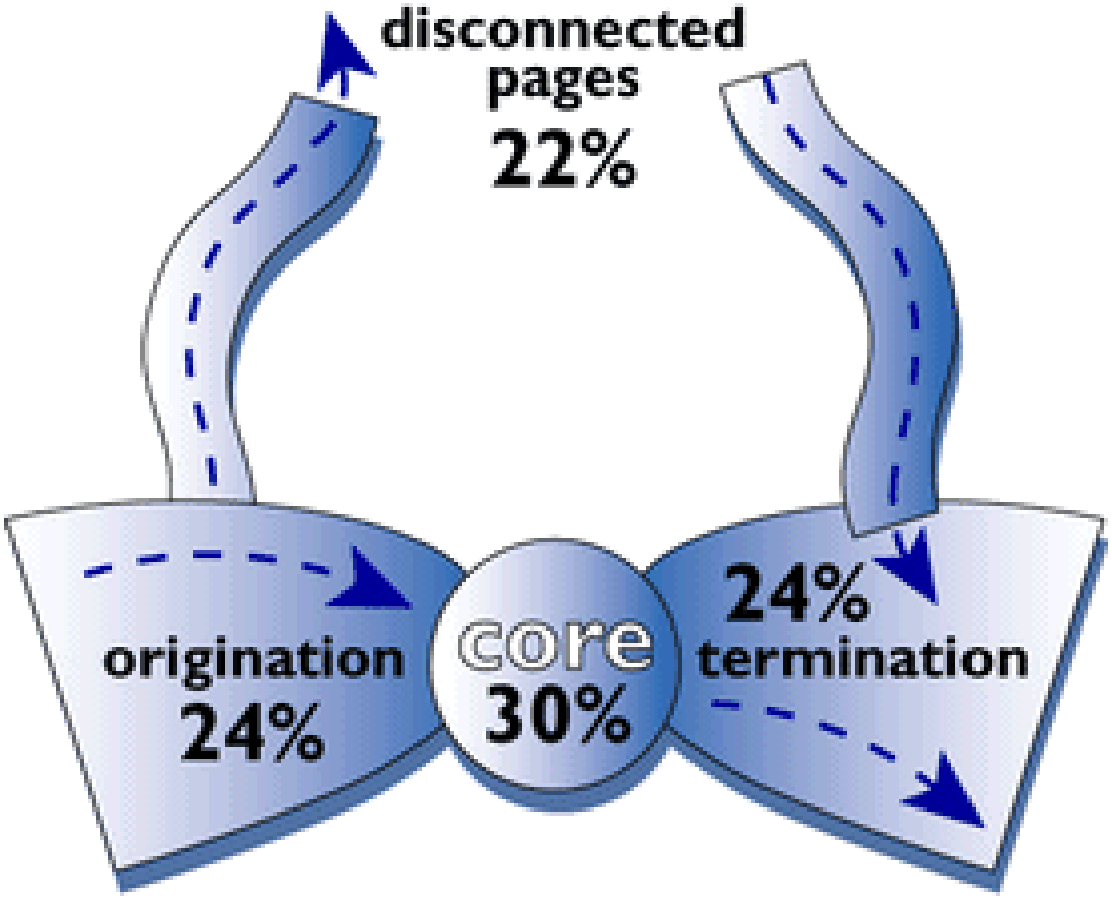
- Example for the amount of web pages:
  - Searching for 'Enterprise' yielded on Google ~ 435 millions of results
  - Users investigate up to 20 result list entries.
- What web page is the **most interesting**?
  - cp. Concept of relevance (IR)
- How to **index** this amount of pages?
  - eg. in an inverted list

Web based Retrieval: Challenges

The Web is self-organized

- No central authority / main index
  - For the WWW
- Everyone can add (or edit) pages
  - Cp. Personal homepages, blogs, wikis, …
- Pages disappear on regular basis
  - US study claimed that in 2 investigated tech. journals 50% of the cited links were inaccessible after four years.
- Lots of errors and falsehood, no quality control

WWW – Bow Tie Structure



Ranking by Popularity

- Problem with amount of data
  - Queries on popular terms yield many results
- Idea for selecting the most relevant …
  - Combine content with **popularity** of page
  - More popular pages are "authorities"
- How to define popularity?
  - Only hypertext documents are given …

Popularity Ranking

- 2 Algorithms developed independently
  - PageRank, Brin & Page
  - Hypertext Induced Topic Search (HITS), Kleinberg
- Basic idea of popularity
  - Someone likes a page
  - Gives a recommendation (on another page)
  - Using a hyperlink

Popularity Ranking: Basic Idea

- There are different types of people:
  - Regarding their idea of recommendation
    - People giving a lot of recommendations (links)
    - People giving few recommendations (links)
  - Regarding their state of recommendation
    - Recommended by a lot of people
    - Recommended by few people
- Combinations are possible:
  - Having no recommendation, but recommending a lot, ...

Popularity Ranking: Basic Idea

Think of ....

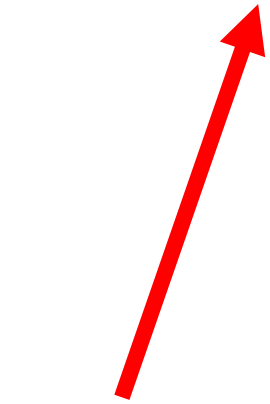
- people as pages
- recommendations as links

Therefore:

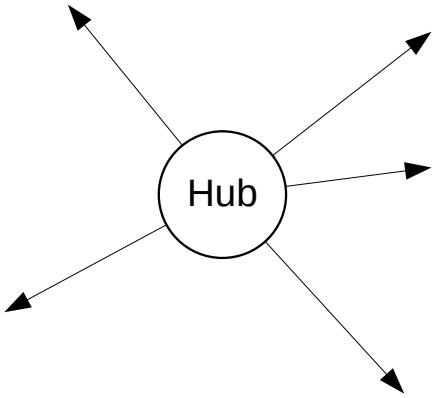
"Pages are popular, if popular pages link them"

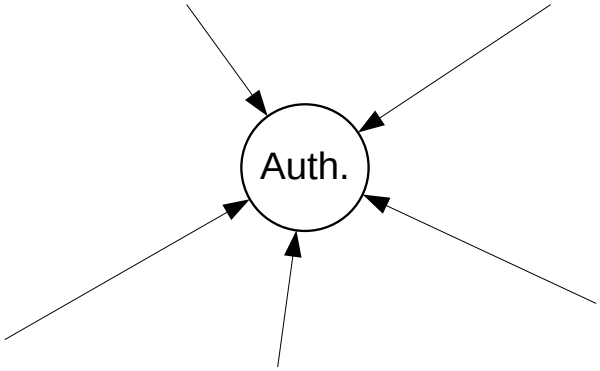
Popularity Ranking: Basic Idea

- Additional assumptions:
  - **Hubs** are pages that refer a lot
  - **Authorities** are pages, which are referred a lot



HITS





PageRank: Original Summation Formula

- Original summation formula
  - PageRank of page P_i is given by the summation of all pages that link to P_i given by Set B_{Pi}

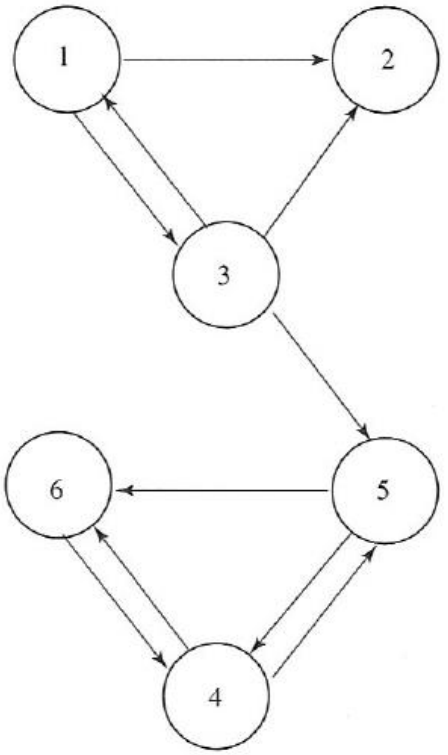
$$r(P_i) = \sum_{P_j \in B_{P_i}} \frac{r(P_j)}{|P_j|},$$

- Iterative formula, starting with rank 1/n for all n pages:

$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|}$$

PageRank: Original Summation Formula

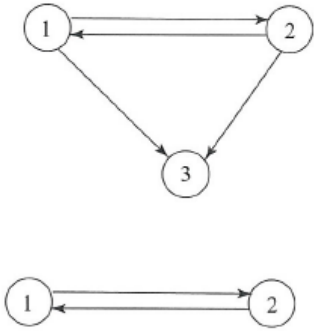
$$r_{k+1}(P_i) = \sum_{P_j \in B_{P_i}} \frac{r_k(P_j)}{|P_j|}$$



Iteration 0	Iteration 1	Iteration 2	Rank at Iter. 2
$r_0(P_1) = 1/6$	$r_1(P_1) = 1/18$	$r_2(P_1) = 1/36$	5
$r_0(P_2) = 1/6$	$r_1(P_2) = 5/36$	$r_2(P_2) = 1/18$	4
$r_0(P_3) = 1/6$	$r_1(P_3) = 1/12$	$r_2(P_3) = 1/36$	5
$r_0(P_4) = 1/6$	$r_1(P_4) = 1/4$	$r_2(P_4) = 17/72$	1
$r_0(P_5) = 1/6$	$r_1(P_5) = 5/36$	$r_2(P_5) = 11/72$	3
$r_0(P_6) = 1/6$	$r_1(P_6) = 1/6$	$r_2(P_6) = 14/72$	2

Initial Problems

- Rank sinks & cycles:
  - Some pages get all of the score, other pages none
  - Cycles just flip the rank
- How many iterations?
  - Will the process converge?
  - Will it converge to one single vector?



Approach of Brin & Page

- Notion of the random surfer
  - Someone navigates through the web using hyperlinks.
  - If there are 6 links, there is a probability of 1/6 that s/he takes a specific link
  - On dangling nodes (without out links) s/he can jump everywhere with equal chance
  - Furthermore s/he can leave the link path with a given probability every time

Features of PageRank

- Mathematical model
  - Created later on, based on Markov chains
- Can be handled in a distributed way
  - "Worlds biggest matrix multiplication"

HITS

- Every page i has a authority score x_i and a hub score y_i
- Successive refinement of scores:

$$x_i^{(k)} = \sum_{j:e_{ji} \in E} y_j^{(k-1)} \text{ and } y_i^{(k)} = \sum_{j:e_{ji} \in E} x_j^{(k)} \text{ for } k = 1, 2, 3, \ldots$$

Search Engine "Optimization"

- Business for "optimizing" rank in search listings (SEO)
- There are two ways:
  - Ethical: Good content and communication leads to extensive linking and a high content score as well as popularity
  - Unethical: Try to get a lot of links to the site of the customer or lay a *Google Bomb*.

Costs for Web Crawling

- How much does it cost to run a search engine?
  - Monthly amount of pages to crawl: 4 billion
  - 4.000.000.000 pages @ 200K = 80.000 GB per month.
- One connection:
  - 100mbs connection
    / 8 megabits per MB
    * 60 seconds in a minute
    * 60 minutes in an hour
    * 24 hours in a day
    * 30 days in a month=32.400 GB / month

Costs for Web Crawling

- Therefore at least 3 100 MBit connections are needed
  - Running at full capacity 24/7
  - Only with a simple calculation (w/o overhead)
- Also at least 3 servers are needed
- And a lot of storage
  - ~ 80.000 GB with caching

taken from http://www.mail-archive.com/nutch-user@lucene.apache.org/msg05577.html