Information retrieval and web search

Databases

Ira Assent

ira@cs.au.dk

Data-Intensive Systems group, Department of Computer Science, Aarhus University, DK

Intended learning outcomes

- Be able to
  - Describe and apply the main information retrieval and web search approaches
  - Evaluate retrieval results
  - Apply information retrieval and web search algorithms

Recap: NoSQL systems

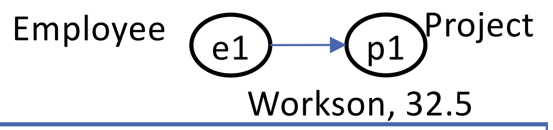
- Four major categories of systems
  - Document-based
    - Store data as documents, e.g. JSON
    - Access using document id, other indexes
  - Key-value stores
    - Simple data model of key-value, where value can be record, document, some other (complex) data structure
  - Column-based / wide-column e.g. MongoDB
    - Partition tables by column families (vertical partitioning)
    - Each column family stored in separate file
    - Typically versioning
  - Graph-based e.g. Neo4j
    - Graphs as data model
    - Access data by traversing edges using path expressions

mongoDB

Create:
```
db.createCollection(”worker”)
```
Read: `db.worker.find(_id:”W1”)`

Update:
```
db.project.update({_id:
”W2”}, {{$set:{Ename:”Jayce
Danish”}})
```
Delete:
```
db.project.remove(_id:”P42”)
```

neo4j



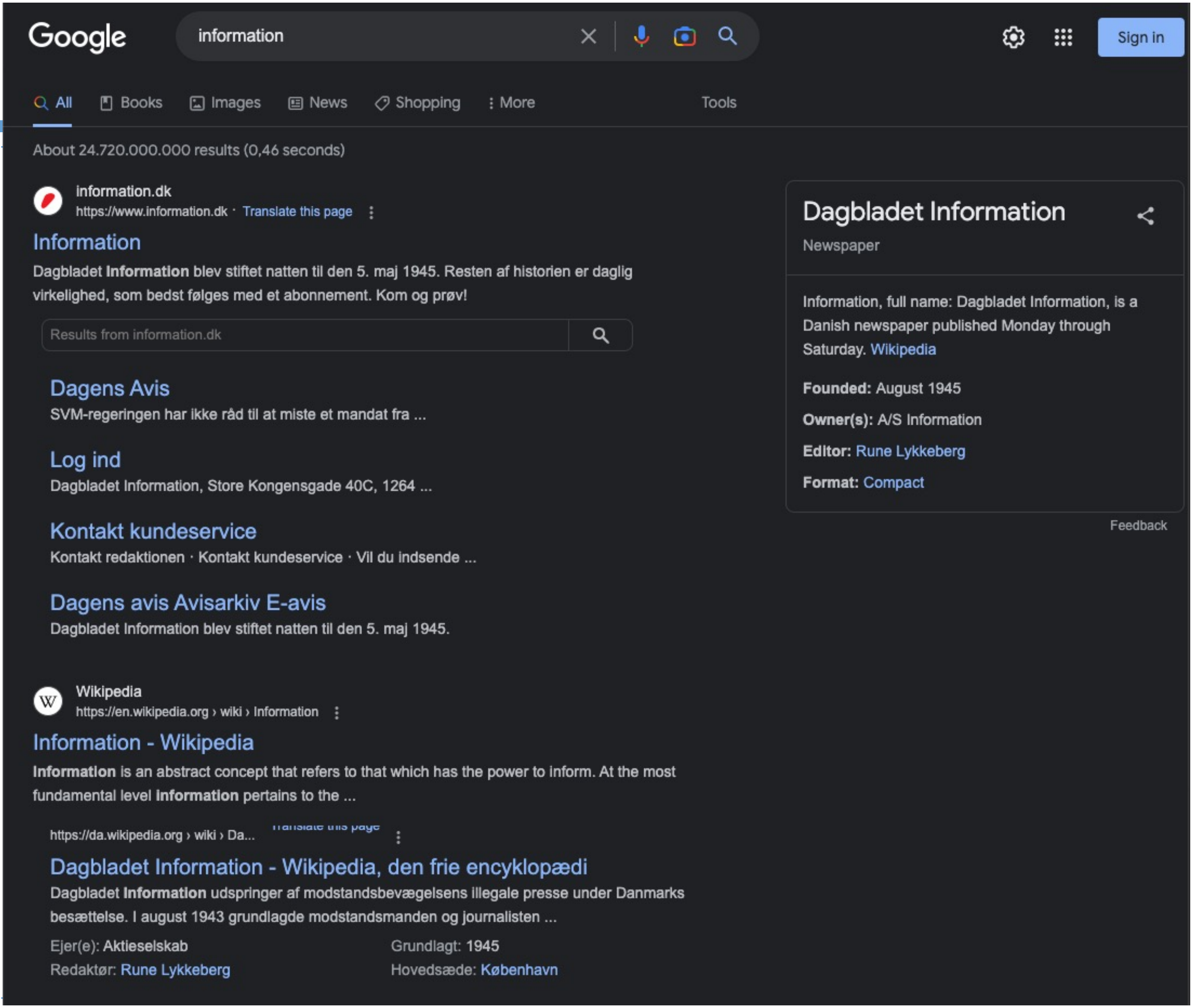
Create:
```
CREATE (e1:Employee,
{Empid:’1’,Name:’John’}),
CREATE (e1)-[:WorksOn]->(p1)
```
Read:
```
MATCH (e:Employee)-
[:Workson]-> (p:Project)
RETURN e, p
```
Update:
```
MATCH
(e:Employee{Empid:’1’}) SET
e.Name=‘Jean’
```
Delete:
```
MATCH
(p:Project{Pname:’ProductX’})
DELETE p
```

Information Retrieval (IR) Concepts

- **Information retrieval**
  - Process of retrieving documents from a collection in response to a query by a user
  - Distinction between structured and unstructured data
    - DBMS highly structured: e.g. relational schema
    - E.g. text documents unstructured: e.g. blog entry
- User's information need expressed as a **free-form search request** (unstructured), also called **keyword search query**
- High noise-to-signal ratio
  - Many irrelevant texts

Search

- Information
  - Several meanings
    - News-paper
    - Concept
    - Facebook page
    - …
- Unstructured
- We lack metadata
- We lack information on what is the intent of the query
- Huge data volumes



Google information

All Books Images News Shopping More Tools

About 24.720.000.000 results (0,46 seconds)

information.dk
https://www.information.dk · Translate this page

Information
Dagbladet **Information** blev stiftet natten til den 5. maj 1945. Resten af historien er daglig virkelighed, som bedst følges med et abonnement. Kom og prøv!

Results from information.dk

Dagens Avis
SVM-regeringen har ikke råd til at miste et mandat fra ...

Log ind
Dagbladet Information, Store Kongensgade 40C, 1264 ...

Kontakt kundeservice
Kontakt redaktionen · Kontakt kundeservice · Vil du indsende ...

Dagens avis Avisarkiv E-avis
Dagbladet Information blev stiftet natten til den 5. maj 1945.

Wikipedia
https://en.wikipedia.org › wiki › Information

Information - Wikipedia
Information is an abstract concept that refers to that which has the power to inform. At the most fundamental level **information** pertains to the ...

https://da.wikipedia.org › wiki › Da... Translate this page

Dagbladet Information - Wikipedia, den frie encyklopædi
Dagbladet **Information** udspringer af modstandsbevægelsens illegale presse under Danmarks besættelse. I august 1943 grundlagde modstandsmanden og journalisten ...

Ejer(e): Aktieselskab	Grundlagt: 1945
Redaktør: Rune Lykkeberg	Hovedsæde: København

Dagbladet Information
Newspaper

Information, full name: Dagbladet Information, is a Danish newspaper published Monday through Saturday. Wikipedia

Founded: August 1945
Owner(s): A/S Information
Editor: Rune Lykkeberg
Format: Compact

Feedback

Keyword Queries and Boolean Queries

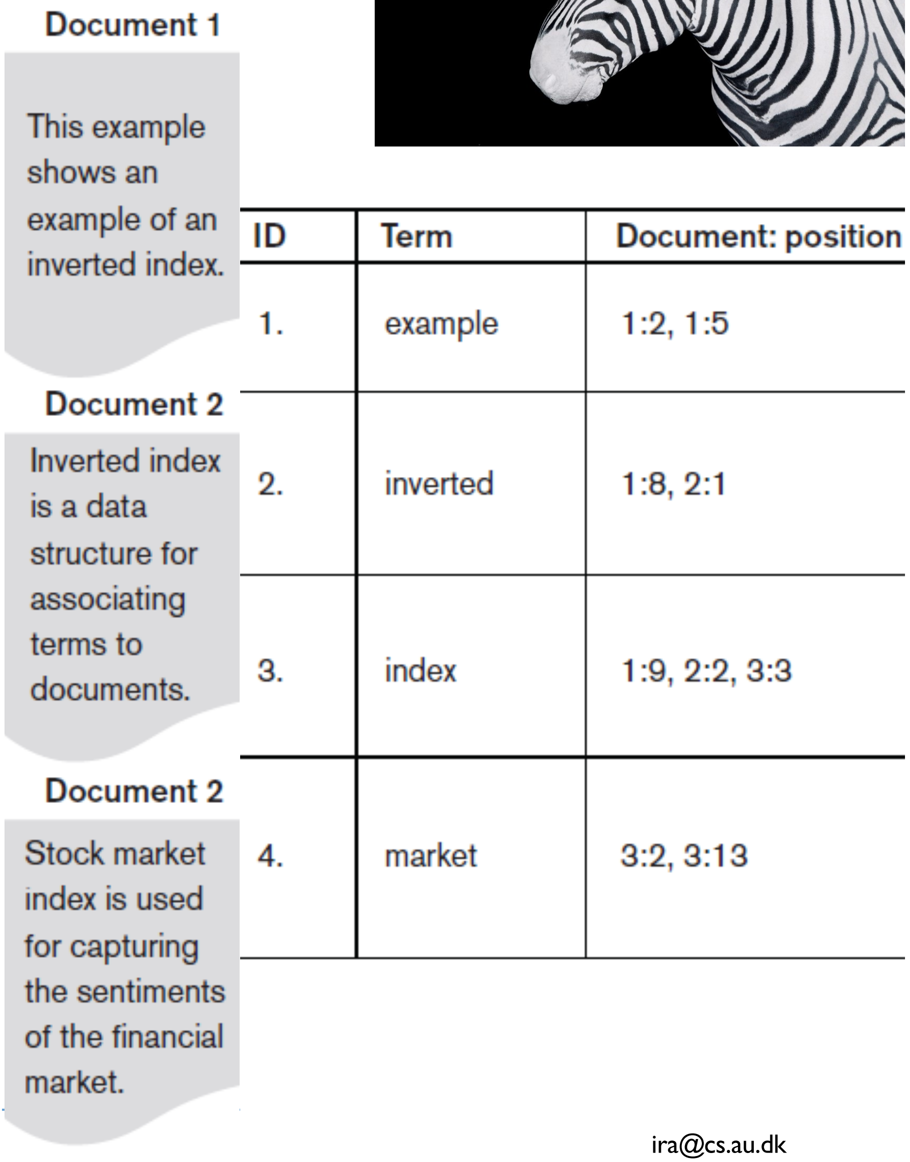
- Simplest and most commonly used forms of IR queries
- **Keywords** implicitly connected by a logical AND operator
- Remove **stopwords**
  - Most commonly occurring words such as the, of, to, a, and, …
    - Commonly used words in language (expected in >80% of documents)
  - Removal must be performed before indexing
- **Boolean queries**
  - AND: both terms must be found
  - OR: either term found
  - NOT: record containing keyword omitted
  - Document retrieved if query logically true as exact match in document
- Results as simple combination of keyword matches

Indexing for IR – what to do?

1. Build an index that for each document stores the keywords it contains
2. Build an index that for each keyword stores the documents that contain it
3. Both can be used to speed up the search query
4. Neither can be used to speed up search query

Indexing occurrences of terms

- Vocabulary
  - Set of distinct query terms in the document set
- Inverted index
  - attaches distinct terms with a list of all documents that contains term
- Inverted index construction
  1. break documents into vocabulary terms by tokenizing, cleaning, stopword removal, stemming, …
  2. derive frequency counts and other statistics
  3. turn document-term representation into a term-document representation



Document 1

This example shows an example of an inverted index.

Document 2

Inverted index is a data structure for associating terms to documents.

Document 2

Stock market index is used for capturing the sentiments of the financial market.

ID	Term	Document: position
1.	example	1:2, 1:5
2.	inverted	1:8, 2:1
3.	index	1:9, 2:2, 3:3
4.	market	3:2, 3:13

IR Process Pipeline

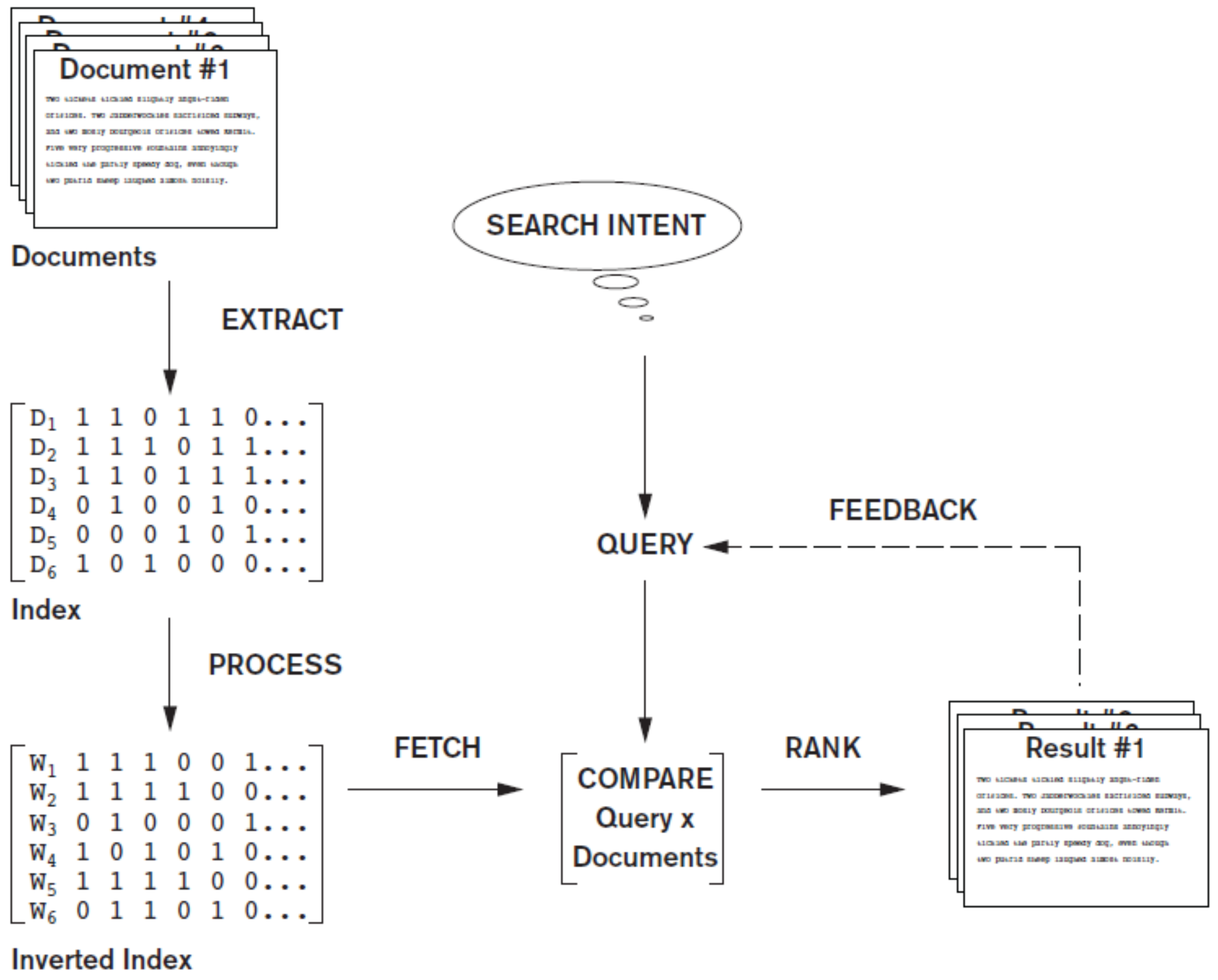


Figure 27.2
Simplified IR process pipeline.

Which document is the most relevant for the query ”data”?

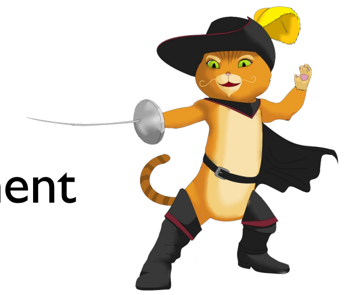
VOTE

1. A long one with one occurrence of ”data”
2. A long one with three occurrences ”databases”, ”datalogi”, ”dada”
3. A short one with one occurrence of ”data”
4. A short one with two occurrences ”databases”, ”datalogi”

Word Frequency

- A rare term like 'Rumplestiltskin' likely to be more indicative of document similarity than frequent term 'story'
  - consider **document frequency**
    - how frequently the word appears in document collection
  - If it is very frequent, it does not tell us much about the individual document
    - E.g. all documents talk about stories

- Document with 100 mentions of 'Cat' likely more relevant than one with a single mention when searching for cat content
  - consider **term frequency**
    - how many times word appears in a document



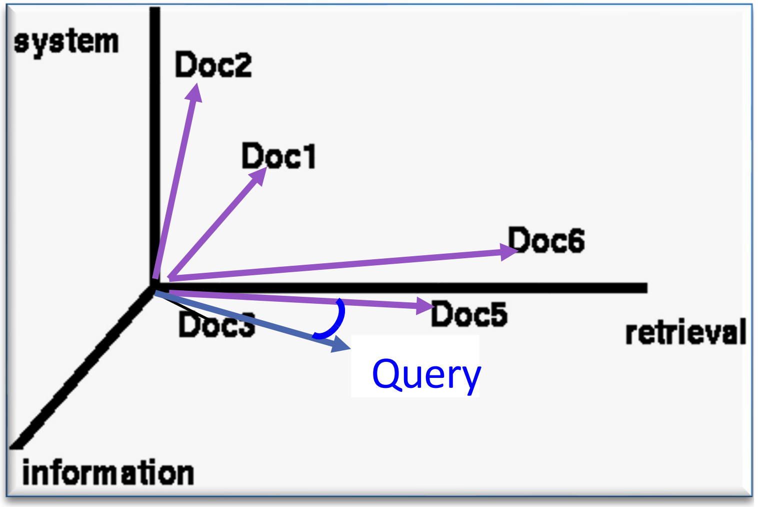
The vector space model

- Documents and queries represented as **vectors**
- Position 1 corresponds to term 1, …, position t to term t
- The **weight** of the term is stored in each position
- Easy to compute
- Vector distance measure used to rank retrieved documents
- Documents close to query measured using vector-space metric are returned first
- What happens beyond 2 or 3 dimensions? More terms → harder to understand which subsets of words are shared among similar documents

$$D_i = w_{d_{i1}}, w_{d_{i2}}, \ldots, w_{d_{it}}$$

$$Q = w_{q1}, w_{q2}, \ldots, w_{qt}$$

$$w = 0 \text{ if a term is absent}$$



TF-IDF

- TF assigns high weight to terms in a document that occur frequently
  - relative to length of document, so longer documents do not necessarily imply higher weight

$$TF_{ij} = \frac{f_{ij}}{\sum_i f_{ij}}$$

f

$\boldsymbol{f_{ij}}$ = frequency of term *i* in document *j*

- IDF assigns high weight to rare words found in few documents – "discriminative"; use logarithm to reduce impact of absolute values

$$IDF_i = log\frac{N}{n_i}$$

(To avoid division by zero add 1: $IDF_i = log\frac{N+1}{n_i+1}$, *different versions exist)*

$\boldsymbol{n_i}$ = number of docs that mention term ***i***

N = total number of docs

$$\log\left(\frac{10000}{10000}\right) = 0$$

$$\log\left(\frac{10000}{5000}\right) = 0.301$$

$$\log\left(\frac{10000}{20}\right) = 2.698$$

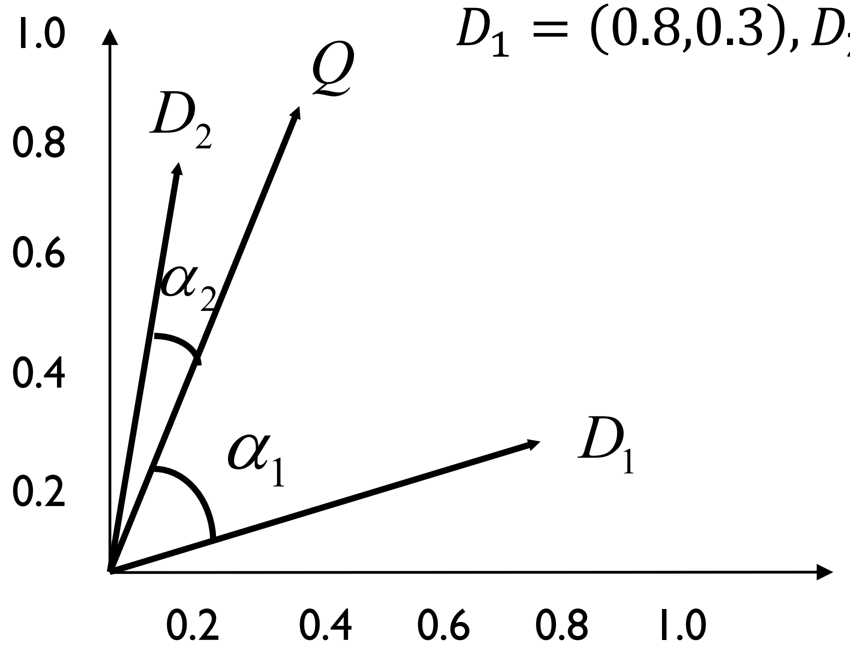
$$\log\left(\frac{10000}{1}\right) = 4$$

Term frequency - Inverse Document Frequency

TF-IDF score: $w_{ij} = TF_{ij} \times IDF_i$

Vector space similarity

- Use the weights to compare documents: cosine similarity



$D_1 = (0.8,0.3), D_2 = (0.2,0.7), Q = (0.4,0.8)$

$$\mathrm{CosSim}(\mathrm{D_j}, \mathrm{Q}) = \frac{\sum_i w_{ij} \times w_{iq}}{\sqrt{\sum_i w_{ij}^2} \times \sqrt{\sum_i w_{iq}^2}}$$

$cos\alpha_1 = 0.74, cos\alpha_2 = 0.98$

$$\mathrm{CosSim}(\mathrm{D_2}, \mathrm{Q}) = \frac{0.4 \times 0.2 + 0.8 \times 0.7}{\sqrt{(0.4)^2 + (0.8)^2} \times \sqrt{(0.2)^2 + (0.7)^2}}$$

$$= \frac{0.64}{\sqrt{0.42}} \quad = 0.98$$

Probabilistic Model

- Probability ranking principle
  - Decide whether the document belongs to the **relevant** set or the **nonrelevant** set for a query
- **BM25** (Best Match 25)
  - Popular probabilistic ranking algorithm
- **Okapi** system
  - Builds a score based on frequency of term in document, frequency of term in query, total number of documents, total number of documents that contain the term, document length, and average document length

BM25

- Popular and effective ranking algorithm
  - uses document and query term weights
- Basic idea: incorporate relevance information, use weights that work well in practice

$$\sum_{i\in Q} \log \frac{(r_i+0.5)/(R-r_i+0.5)}{(n_i-r_i+0.5)/(N-n_i-R+r_i+0.5)} \cdot \frac{(k_1+1)f_i}{K+f_i} \cdot \frac{(k_2+1)qf_i}{k_2+qf_i}$$

- relevance information r_i, R
- N number of documents, n_i number of documents that contain the term, f_i frequency of term
- k_1, k_2 and K are parameters whose values are set empirically
  - Means, try and see which values give good results
  - Typical value for k_1 is 1.2, k_2 varies from 0 to 1000, b = 0.75
  - dl document length, avdl average document length

$$K = k_1((1-b) + b \cdot \frac{dl}{avdl})$$

BM25 Example calculation

- Query with two terms "president lincoln"
  - qf = 1
- No relevance information
  - r and R are zero
- N = 500,000 documents
- "president" occurs in 40,000 documents
  - $n_1 = 40,000$
- "lincoln" occurs in 300 documents
  - $n_2 = 300$
- "president" occurs 15 times in document
  - $f_1 = 15$
- "lincoln" occurs 25 times in document
  - $f_2 = 25$
- document length is 90% of the average length
  - dl/avdl = .9
- $k_1 = 1.2$, b = 0.75, and $k_2 = 100$
- K = 1.2 · (0.25 + 0.75 · 0.9) = 1.11

$$\sum_{i \in Q} \log \frac{(r_i+0.5)/(R-r_i+0.5)}{(n_i-r_i+0.5)/(N-n_i-R+r_i+0.5)} \cdot \frac{(k_1+1)f_i}{K+f_i} \cdot \frac{(k_2+1)qf_i}{k_2+qf_i}$$

$$\log \frac{(0+0.5)/(0-0+0.5)}{(40000-0+0.5)/(500000-40000-0+0+0.5)} \times \frac{(1.2+1)15}{1.11+15} \times \frac{(100+1)1}{100+1}$$
$$+\log \frac{(0+0.5)/(0-0+0.5)}{(300-0+0.5)/(500000-300-0+0+0.5)} \times \frac{(1.2+1)25}{1.11+25} \times \frac{(100+1)1}{100+1}$$

$$= \log 460000.5/40000.5 \cdot 33/16.11 \cdot 101/101$$
$$+ \log 499700.5/300.5 \cdot 55/26.11 \cdot 101/101$$
$$= 2.44 \cdot 2.05 \cdot 1 + 7.42 \cdot 2.11 \cdot 1$$
$$= 5.00 + 15.66 = 20.66$$

BM25 example result

- Effect of term frequencies
- Query takes into account both query terms in "president lincoln"

Frequency of "president"	Frequency of "lincoln"	BM25 score
15	25	20.66
15	1	12.74
15	0	5.00
1	25	18.2
0	25	15.66

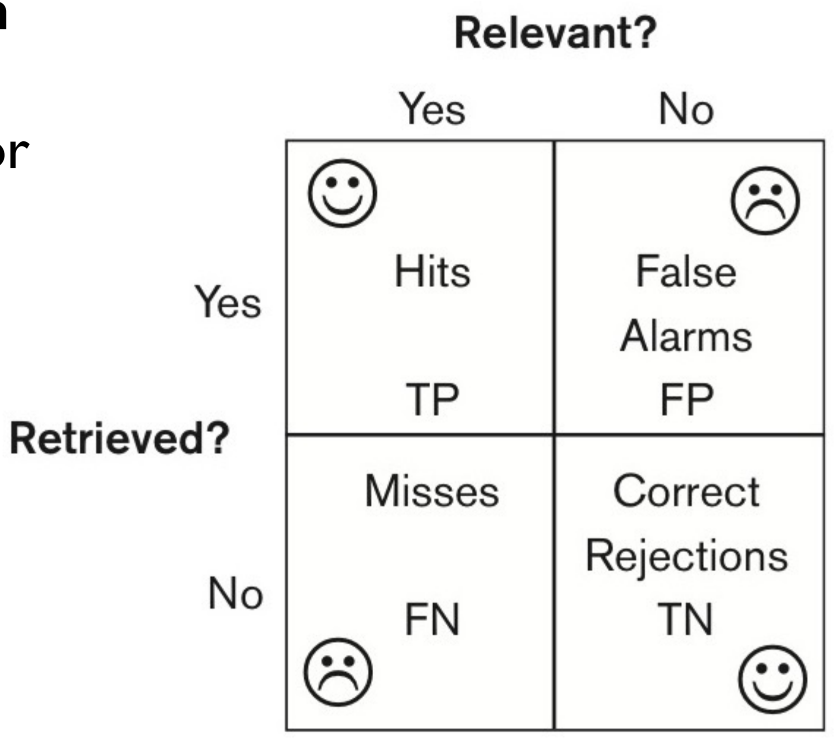
What is best?

A search engine that returns

A. 5 documents, 2 relevant
B. 15 documents, 3 relevant
C. 1 document, 1 relevant
D. 300 documents, 99 relevant

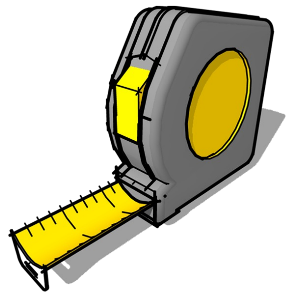
Evaluating information retrieval results

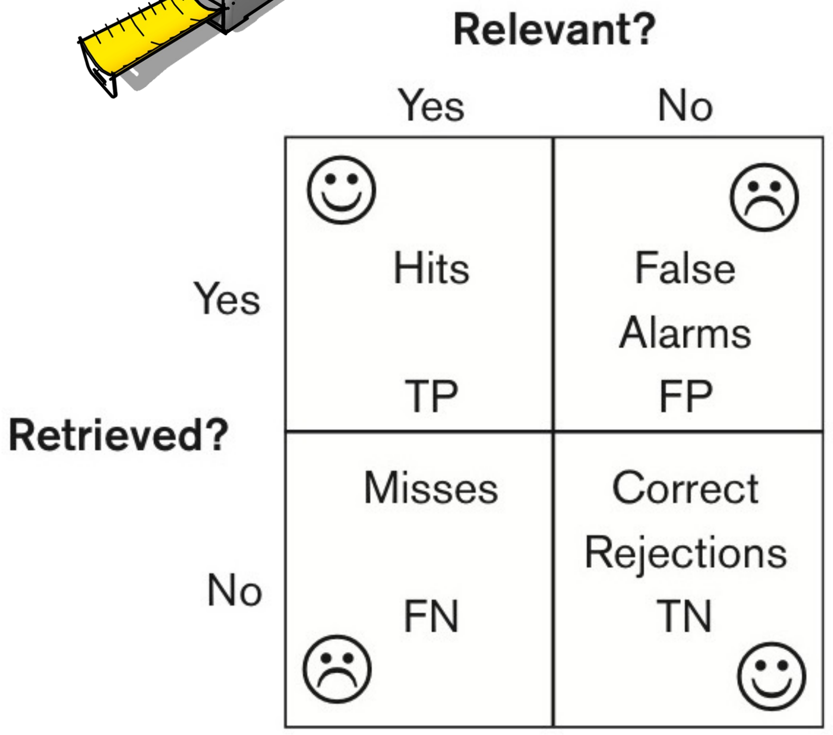
- As opposed to SQL queries, in information retrieval not all results necessarily correct
  - Need to measure the quality of the results
- Assuming we know some **ground truth** or **golden standard**
  - **i.e.**, whether a document is relevant or not
- check some results and see how the IR system fares
  - Hits / true positives
    - ground truth: relevant, in result
  - Misses / false negatives
    - ground truth: relevant, not in result
  - False alarms / false positives
    - ground truth: irrelevant, in result
  - Correct rejections / true negatives
    - ground truth: irrelevant, not in result



Popular measures

- **Recall**
  - Number of relevant documents retrieved by a search (Hits) / Total number of existing relevant documents (TP+FN)
    - How much of the good stuff?
- **Precision**
  - Number of relevant documents retrieved by a search / Total number of documents retrieved by that search (TP+FP)
    - How good is what I have?
- **F-score**
  - Single measure that combines precision and recall (harmonic mean) to compare different result sets: 2*precision*recall/(precision+recall)





		Relevant?	
		Yes	No
Retrieved?	Yes	☺ Hits TP	☹ False Alarms FP
	No	☹ Misses FN	Correct Rejections TN ☺

Databases and IR Systems: A Comparison

Table 27.1 A Comparison of Databases and IR Systems

Databases	**IR Systems**
■ Structured data	■ Unstructured data
■ Schema driven	■ No fixed schema; various data models (e.g., vector space model)
■ Relational (or object, hierarchical, and network) model is predominant	■ Free-form query models
■ Structured query model	■ Rich data operations
■ Rich metadata operations	■ Search request returns list or pointers to documents
■ Query returns data	■ Results are based on approximate matching and measures of effectiveness (may be imprecise and ranked)
■ Results are based on exact matching (always correct)	

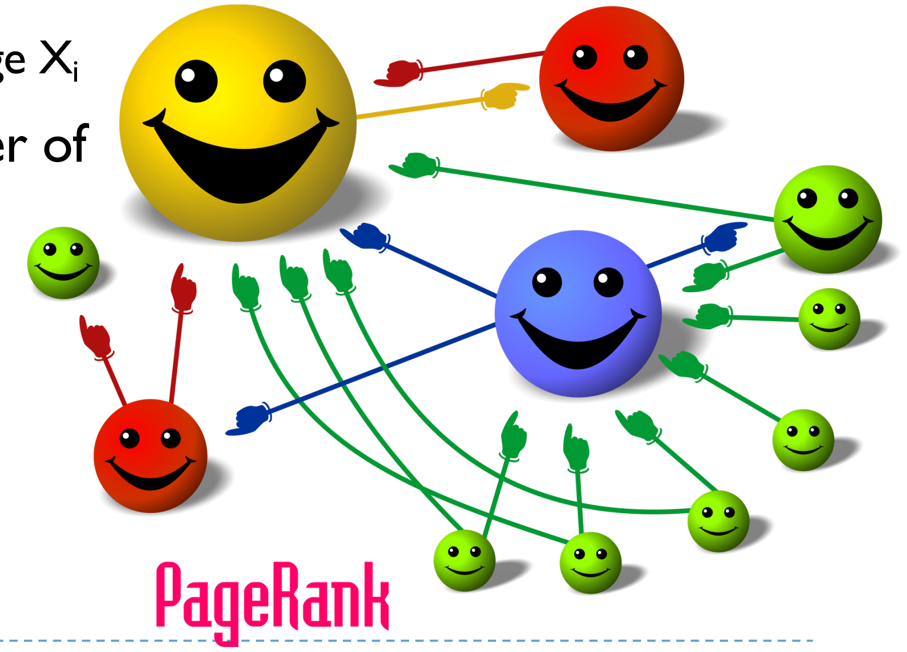
Web Search

- Deliver results with high precision even at the expense of recall
- Make system easy to use
  - No need to specify more than a few words
- Web pages are connected via **hyperlinks**
  - Destination page
  - Anchor text

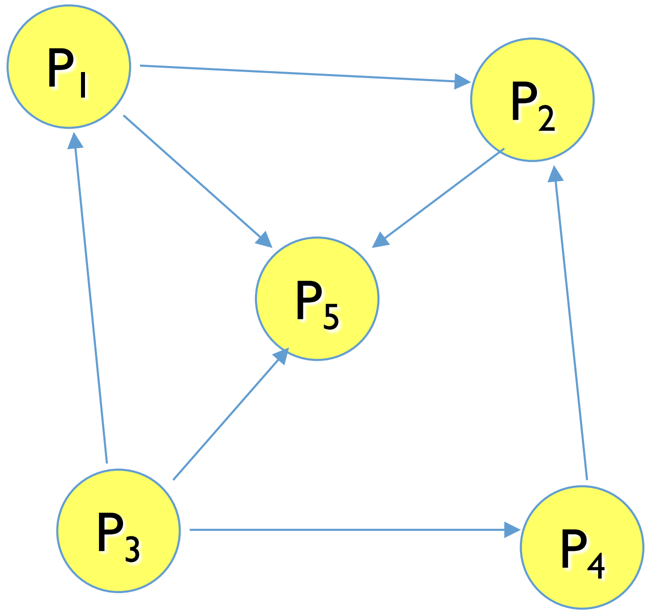
- The **PageRank** ranking algorithm
  - Used by Google
  - Highly linked pages are more important (have greater authority) than pages with fewer links
  - Measure of query-independent importance of a page/node
    - By Brin and Page (1998)
    - Also depend on the importance of the source webpage

PageRank

- Each webpage A has a score, the **pagerank** P(A) indicating its authority attributed by other webpages
  - P(A) is sum of normalized pageranks of all webpages pointing into A
  - $P(A)=\sum_i P(Xi)/O(Xi)$
    - X_i is the set of webpages pointing to A
    - $O(X_i)$ is the number of out-links from page X_i
- Normalized means relative to the number of outlinks of the referencing page $O(X_i)$
  - So being pointed to by a page with many outlinks counts less



Page rank example



Suppose the pagerank of P_1, P_2, P_3, and P_4 are known

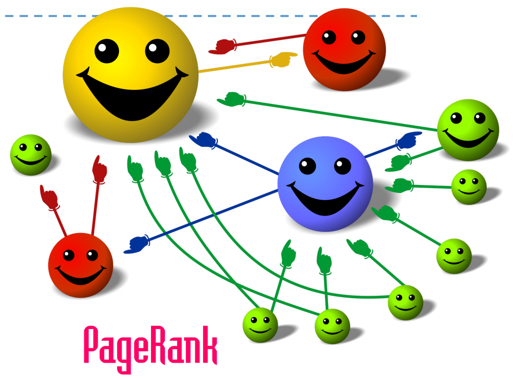
Pages	P_1	P_2	P_3	P_4
Pagerank	3.5	1.2	4.2	1.0

$$P(A) = \sum_i P(Xi)/O(Xi)$$

The pagerank of P_5 is computed as

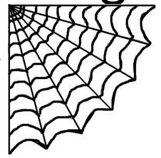
$$P(P_5) = 3.5/2 + 1.2/1 + 4.2/3 = 4.35$$

Computation of Pagerank



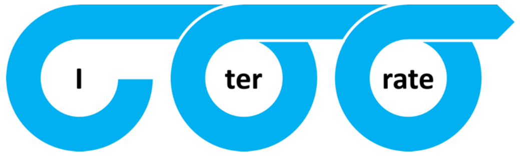
- Initially, pageranks unknown
  - How to determine the first pagerank value?
- Give an initial pagerank to every webpage
  - E.g., 1/n where n is the total number of webpages
- Perform the calculation of pagerank iteratively
  - Use the pagerank formula to update the pagerank of every webpage
  - Repeat the above step a number of times until the pagerank values are stable (converge)

- Imagine doing **random walk** on links in entire web for infinite number of steps
- Occasionally, get bored and instead of following a link, jump to another random page
- At some point, the percentage of time spent at each page will converge to fixed value: its pagerank



Iterative Procedure

- Let $P_k(X_i)$ PageRank of page X_i at iteration k
  - Starting with $P_0(X_i)$ = 1/n for all pages X_i
- At iteration k+1, the pagerank of every page X_i is updated using the pageranks at iteration k



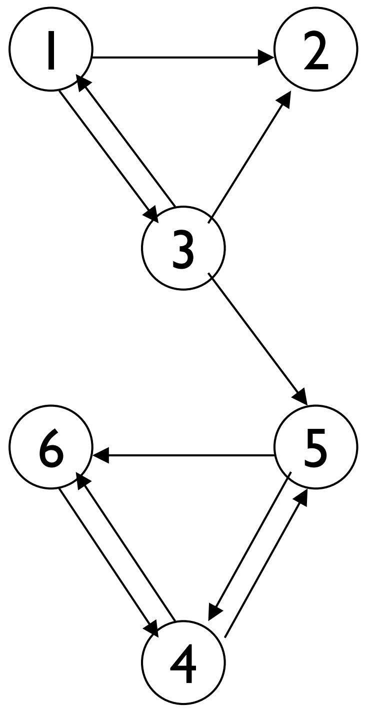
- Introduce a damping factor 0<d<1, usually 0.85 to encourage convergence
  - Otherwise, pagerank values would just keep growing
  - $P(A)=(1-d)+d(\sum_i P(Xi)/O(Xi))$

Iterative pagerank example

- Consider the graph on the right

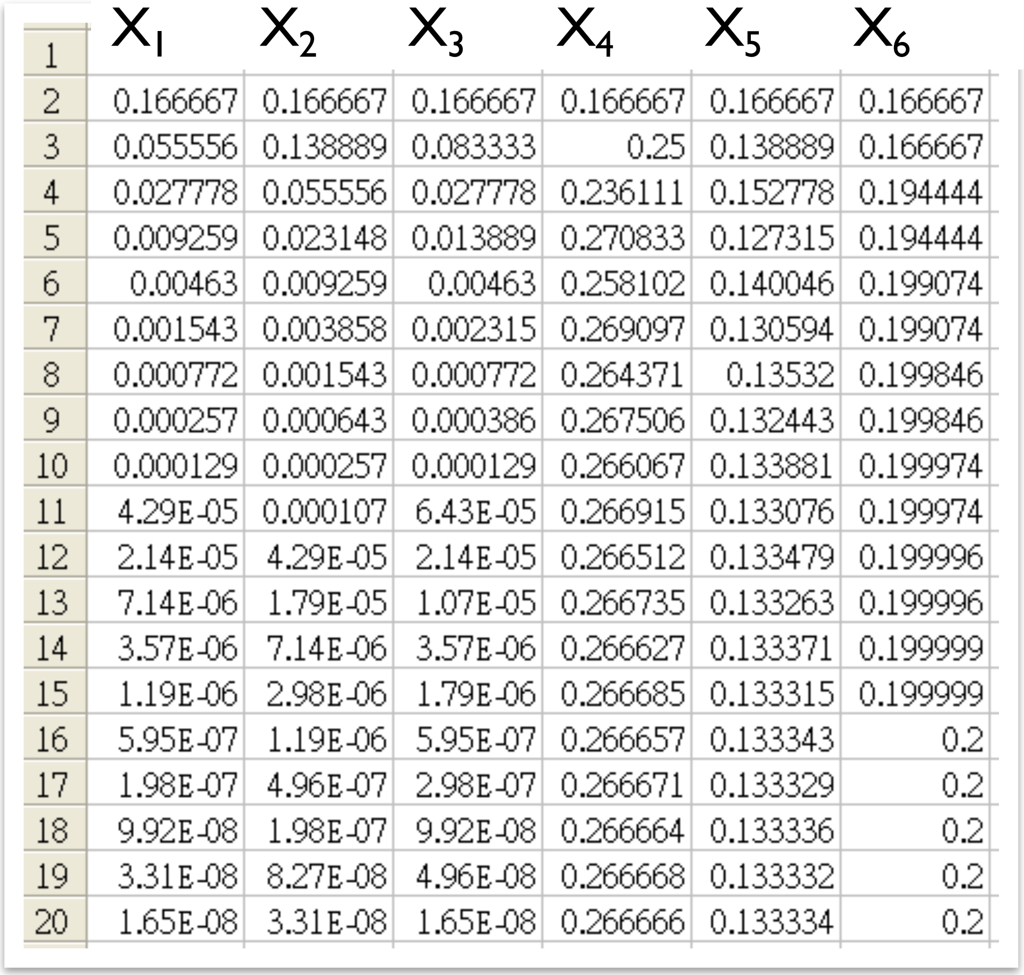
(for the sake of the example, no damping, i.e., d=1)

- Iteration 0
  - $P_0(X_i) = 1/6$ for $i=1, 2, ..., 6$
- Iteration 1
  - $P_1(X_1) = P_0(X_3) / 3 = 0.0556$
  - $P_1(X_2) = P_0(X_1) / 2 + P_0(X_3) / 3 = 0.1389$
  - $P_1(X_3) = P_0(X_1) / 2 = 0.0833$
  - $P_1(X_4) = P_0(X_5) / 2 + P_0(X_6) = 0.25$
  - $P_1(X_5) = P_0(X_3) / 3 + P_0(X_4) / 2 = 0.1389$
  - $P_1(X_6) = P_0(X_4) / 2 + P_0(X_5) / 2 = 0.1667$



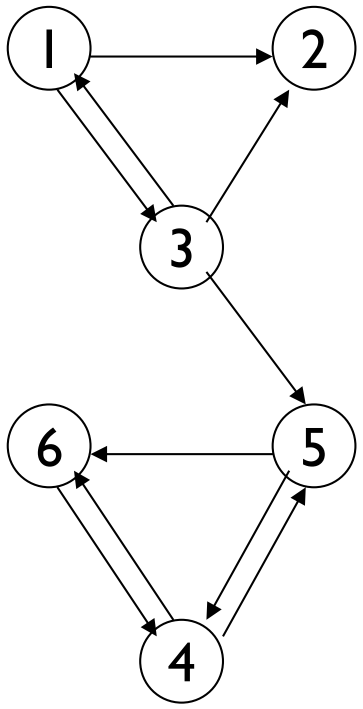
Iterative pagerank example continued

- After 20 iterations …



1	X_1	X_2	X_3	X_4	X_5	X_6
2	0.166667	0.166667	0.166667	0.166667	0.166667	0.166667
3	0.055556	0.138889	0.083333	0.25	0.138889	0.166667
4	0.027778	0.055556	0.027778	0.236111	0.152778	0.194444
5	0.009259	0.023148	0.013889	0.270833	0.127315	0.194444
6	0.00463	0.009259	0.00463	0.258102	0.140046	0.199074
7	0.001543	0.003858	0.002315	0.269097	0.130594	0.199074
8	0.000772	0.001543	0.000772	0.264371	0.13532	0.199846
9	0.000257	0.000643	0.000386	0.267506	0.132443	0.199846
10	0.000129	0.000257	0.000129	0.266067	0.133881	0.199974
11	4.29E-05	0.000107	6.43E-05	0.266915	0.133076	0.199974
12	2.14E-05	4.29E-05	2.14E-05	0.266512	0.133479	0.199996
13	7.14E-06	1.79E-05	1.07E-05	0.266735	0.133263	0.199996
14	3.57E-06	7.14E-06	3.57E-06	0.266627	0.133371	0.199999
15	1.19E-06	2.98E-06	1.79E-06	0.266685	0.133315	0.199999
16	5.95E-07	1.19E-06	5.95E-07	0.266657	0.133343	0.2
17	1.98E-07	4.96E-07	2.98E-07	0.266671	0.133329	0.2
18	9.92E-08	1.98E-07	9.92E-08	0.266664	0.133336	0.2
19	3.31E-08	8.27E-08	4.96E-08	0.266668	0.133332	0.2
20	1.65E-08	3.31E-08	1.65E-08	0.266666	0.133334	0.2

Rank	Page
1	X_4
2	X_6
3	X_5
4	X_2
5, 6	X_1, X_3



$$\begin{tikzcd}
1 \arrow[rr] \arrow[dr, shift left] & & 2 \\
& 3 \arrow[ul, shift left] \arrow[ur] \arrow[dr] & \\
6 \arrow[dr, shift left] & & 5 \arrow[ll] \arrow[dl, shift left] \\
& 4 \arrow[ul, shift left] \arrow[ur, shift left] &
\end{tikzcd}$$

What do you imagine is an issue for PageRank?

VOTE

1. Artificial booster pages with lots of links
2. Lots of artificial booster pages with the same link
3. Pages linking to the same document are hard to distinguish
4. Documents with few links are hard to distinguish

So, what’s ChatGPT then?

- Not classical information retrieval, but machine learning based NLP (natural language processing)
- Chatbot by OpenAI, based on GPT language model
  - Language model: learn to predict which text (piece) should be next
  - In simpler form similar to suggestions when typing message on mobile phone
    - Given what you typed so far, what are the most likely characters / words – probability distribution
- Based on some text input, learns advance statistics
  - Key ingredients: lots of text (Big Data), lots of hardware (training the model), and powerful machine learning models (Deep Learning, Reinforcement Learning)
  - Powerful, impressive examples for essay writing, coding, problem solving,…
  - Downside: “hallucinates”, no / makes up references
    - Lack of reliability, transparency, traceability
- In your studies: Machine Learning course introduces underlying methods
- We have more on the topic on the Master’s
  - We study models, efficient algorithms, scalability, use of modern hardware

Summary

- Intended learning outcomes
  - Describe and apply the main information retrieval and web search approaches
  - Evaluate retrieval results
  - Apply information retrieval and web search algorithms

Where to go from here?

Next lecture will provide a view on Transactions – the underlying logical model of the DBMS for handling database queries and statements

Acknowledgements: slide material from Raghavan, Schütze, Larson; from Croft, Metzler, Strohman; Colin Cooper

What was this all about?

Guidelines for your own review of today's session

- In information retrieval, our aim is…
  - The main difference between unstructured…
- IR systems generally work as follows…
  - The three main retrieval models…
  - TF-IDF is…
- Queries can be specified as…
- Information Extraction is about…
- We evaluate results using…
- Web analysis is related in that it…
- The idea underlying algorithms like PageRank is…