Search Engines
Chapter 2 – Architecture

28.4.2009
Felix Naumann
Overview

- Basic Building Blocks
  - Indexing
    - Text Acquisition
    - Text Transformation
    - Index Creation
  - Querying
    - User Interaction
    - Ranking
    - Evaluation
Software Architecture

- Software components
- Interfaces
- Relationships

- Example UIMA: Unstructured Information Management Architecture
  - www.research.ibm.com/UIMA
  - http://incubator.apache.org/uima/
Search Engine Architecture

- Determined by two main requirements
  - Effectiveness (quality of results)
    - As good as possible
  - Efficiency (response time and throughput)
    - As quickly as possible
- Other requirements fall into these categories
  - Changing documents -> Effectiveness and efficiency
  - Personalization: Effectiveness
  - Spam: Effectiveness and efficiency
  - ...

Felix Naumann | Search Engines | SoSe 2009
The Indexing Process

Text and metadata for all documents

Text Acquisition

identifies and stores documents for indexing

Document data store

Index Creation

Takes index terms and creates data structures (indexes) to support fast searching

Index (inverted index)

Text Transformation

Transforms documents into index terms or features
The Query Process

Document data store

- Supports creation and refinement of query, display of results

User Interaction

- Uses query and indexes to generate ranked list of documents

Index

- Uses query and indexes to generate ranked list of documents

Log data

- Monitors and measures effectiveness and efficiency (primarily offline)

Evaluation

- User Interaction
Overview

- Basic Building Blocks
- Indexing
  - Text Acquisition
  - Text Transformation
  - Index Creation
- Querying
  - User Interaction
  - Ranking
  - Evaluation
The Indexing Process

Document data store

Text Acquisition

Index Creation

Index

Crawler
Feeds
Conversion
Document data store

Text Transformation
Text Acquisition – Crawler

- Identifies and acquires documents for search engine
- Many types – web, enterprise, desktop
- Web crawlers follow *links* to find documents
  - Must efficiently find huge numbers of web pages (*coverage*) and keep them up-to-date (*freshness*)
  - Single site crawlers for *site search*
  - *Topical* or *focused* crawlers for vertical search
- *Document* crawlers for enterprise and desktop search
  - Follow links and scan directories
Real-time streams of documents
- e.g., web feeds for news, blogs, video, radio, tv

RSS is common standard
- Rich Site Summary (RSS-Versionen 0.9x)
- RDF Site Summary (RSS-Versionen 0.9 und 1.0)
- Really Simple Syndication (RSS 2.0)
- RSS “reader” can provide new XML documents to search engine
Text Acquisition – Conversion

- Convert variety of documents into a consistent text plus metadata format
  - e.g. HTML, XML, Word, PDF, etc. → XML
- Convert text encoding for different languages
  - Using a Unicode standard like UTF-8 or Unicode
  - Be consistent throughout application
- Non-content data (tags, metadata) is either removed or stored as metadata.
- First step towards text transformation

http://www.uni-mainz.de/Organisationen/TLA/dokumentation/sgml_eng.html

Felix Naumann | Search Engines | SoSe 2009
Text Acquisition – Document data store

- Two parts
  - Unstructured text (compressed)
  - Structured metadata

- Stores text, metadata, and other related content for documents
  - Metadata is information about document such as type and creation date
  - Other content includes links, anchor text

- Why store documents? They are available on the Web anyway...
  - Provide fast access to document contents for search engine components (e.g., result list generation, document summary)

- Could use relational database system
  - More typically, a simpler, more efficient storage system is used due to huge numbers of documents

More in Chapter 3
The Indexing Process

1. Text Acquisition
2. Text Transformation
3. Index Creation
4. Document data store
5. Index

- Parser
- Stopping
- Stemming
- Link analysis
- Information extraction
- Classifier
Text Transformation – Parser

- Processing the sequence of text tokens in the document to recognize structural elements
  - e.g., titles, links, headings, etc.
- **Tokenizer** recognizes “words” in the text
  - Must consider issues like capitalization, hyphens, apostrophes, non-alpha characters, separators
  - Many decisions up front:
    - apple vs. Apple
    - O’Conner vs. owner’s
    - Word separation in Chinese
- **Markup languages** such as HTML, XML often used to specify structure
  - **Tags** used to specify document elements
    - E.g., `<h2> Overview </h2>`
  - Document parser uses **syntax** of markup language (or other formatting) to identify structure
    - E.g. email format, MS Word metadata etc.
Text Transformation – Stopping

- Remove common words
  - e.g., “and”, “or”, “the”, “in”
- Some impact on efficiency and effectiveness
- Can be a problem for some queries
  - To be or not to be

I a about an are as at be by com de en for from how in is it la of on or that the this to was what when where who will with und the www

See also:
http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

http://www.ranks.nl/stopwords/german.html

Felix Naumann | Search Engines | SoSe 2009
Text Transformation – Stemming

- Group words derived from a common stem
  - “computer”, “computers”, “computing”, “compute”
  - Fish, fishing, fisherman

- Usually effective, but not for all queries
  - Aggressive vs. conservative vs. not at all

- Benefits vary for different languages
  - Arabic: Very complicated morphology
  - Chinese: Few word variations anyway
Text Transformation – Link Analysis

- Makes use of *links* and *anchor text* in web pages
  - Stored and indexed separately
  - `<a href = http://www.hpi.uni-potsdam.de/naumann/home.html> Information Systems Group</a>`
- Link analysis identifies *popularity* and *community* information
  - e.g., PageRank
- Anchor text can significantly enhance the representation of pages pointed to by links
- Significant impact on web search
  - Less importance in other applications
Identify classes of index terms that are important for some applications

- Simple: Bold-face, heading, title
- Part of speech tagging
- Named entity recognizers identify classes such as
  - People
  - Locations
  - Companies
  - Dates, etc.
Text Transformation – Classifier

- Identifies class-related metadata for documents
  - i.e., assigns labels to documents
  - e.g., topics, reading levels, sentiment, genre
  - Spam!
  - Advertisements in documents
- Use depends on application

More in Chapter 4

The Indexing Process

Text Acquisition → Index Creation → Index

Text Transformation

Document data store

Document statistics
Weighting
Inversion
Index distribution
Index Creation - Document Statistics

- Statistical information about words, features and documents
- Gathers counts and positions of words and other features
  - Within a document
  - Across groups of documents
  - Across all documents
- Used in ranking algorithm
Index Creation – Weighting

- Computes weights for index terms
  - Relative importance of words in documents
- Used in ranking algorithm
  - Global weight
  - Query-dependent weight
- e.g., tf.idf weight
  - Combination of term frequency in document
  - and inverse document frequency in the collection
Index Creation – Inversion

- Core of indexing process
- Converts document-term information to term-document for indexing
  - Difficult for very large numbers of documents
- Format of inverted file is designed for fast query processing
  - Must also handle updates
  - Compression used for efficiency
Index Creation – Index Distribution

- Distributes indexes
  - across multiple computers
  - and/or multiple sites
- Essential for fast query processing with large numbers of documents
- Many variations
  - Document distribution: Distribute index for subsets of documents
  - Term distribution: Distribute index for subset of terms
  - Replication
- P2P and distributed IR involve search across multiple sites

More in Chapter 5
Overview

- Basic Building Blocks
- Indexing
  - Text Acquisition
  - Text Transformation
  - Index Creation
- Querying
  - User Interaction
  - Ranking
  - Evaluation
The Query Process

Document data store

Query input
Query transformation
Result output

User Interaction

Ranking (retrieval model)

Index

Log data

Evaluation

Felix Naumann | Search Engines | SoSe 2009
User Interaction - Query input

- Provides interface and parser for *query language*
- Most web queries are very simple, other applications may use forms
- Query language used to describe more complex queries and results of query transformation
  - +, -, " ", ~, site:, AND, OR, ...
  - Similar to SQL language used in database applications
    - Not for “end users”
  - IR query languages also allow content and structure specifications, but focus on content
User Interaction - Query transformation

- Improves initial query
  - both before and after initial search
- Includes text transformation techniques used for documents
  - Tokenization, stemming, stopping
- Spell checking and query suggestion provide alternatives to original query
  - Based on query logs
- Query expansion and relevance feedback modify the original query with additional terms
User Interaction – Results output

- Constructs the display of ranked documents for a query
- Generates *snippets* to show how queries match documents
- *Highlights* important words and passages
- Retrieves appropriate *advertising* in many applications
- May provide *clustering* and other visualization tools
- May translate results from foreign languages
The Query Process

- Document data store
- User Interaction
- Ranking (retrieval model)
- Evaluation
- Log data
- Index

Scoring Optimization Distribution
Ranking – Scoring

- ≈ query processing
- Calculates scores for documents using a ranking algorithm
  - Based on retrieval model
- Core component of search engine
- Basic form of score is $\sum q_i \cdot d_i$
  - Summation over vocabulary of collection
  - $q_i$ and $d_i$ are query and document term weights for term $i$
- Many variations of ranking algorithms and retrieval models
- Key requirement: Fast execution!
Designing ranking algorithms for efficient processing

- Term-at-a time vs. document-at-a-time processing
- Safe vs. unsafe optimizations
  - Trade-off between speed and quality
Processing queries in a distributed environment

*Query broker* distributes queries and assembles results

*Caching* is a form of distributed searching

More in Chapter 7
The Query Process

- **Document data store**
- **User Interaction**
- **Ranking (retrieval model)**
- **Index**
- **Log data**
- **Evaluation**

Logging

Ranking Analysis

Performance Analysis

Felix Naumann | Search Engines | SoSe 2009
Logging user queries and interaction is crucial for improving search effectiveness and efficiency.

- Query logs and clickthrough data (& dwell time) used for
  - Query suggestion
  - Spell checking
  - Query caching
  - Ranking
  - Advertising search
  - ...

- Assumption: Pages clicked on a relevant to query.
Evaluation – Ranking and Performance Analysis

- Ranking analysis
  - Measuring and tuning ranking effectiveness
  - Variety of measures

- Performance analysis
  - Measuring and tuning system efficiency
  - Response time, throughput
  - Simulation

More in Chapter 8
This course explains these components of a search engine in more detail.

Often many possible approaches and techniques for a given component:
- Focus is on the most important alternatives.
- i.e., explain a small number of approaches in detail rather than many approaches.
- “Importance” based on research results and use in actual search engines.
- Alternatives described in references (see book).
Summary

- Indexing
  - Text Acquisition
  - Text Transformation
  - Index Creation

- Querying
  - User Interaction
  - Ranking
  - Evaluation