Apache Lucene
Searching the Web and Everything Else

Daniel Naber
Mindquarry GmbH
ID 380
AGENDA

> What's a search engine
> Lucene Java
  – Features
  – Code example
> Solr
  – Features
  – Integration
> Nutch
  – Features
  – Usage example
> Conclusion and alternative solutions
About the Speaker

> Studied computational linguistics
> Java developer
> Worked 3.5 years for an Enterprise Search company (using Lucene Java)
> Now at Mindquarry, creators on an Open Source Collaboration Software (Mindquarry uses Solr)
Question: What is a Search Engine?

> Answer: A software that
> - builds an index on text
> - answers queries using that index

“But we have a database already“

- A search engine offers
  - Scalability
  - Relevance Ranking
  - Integrates different data sources (email, web pages, files, database, ...)
What is a search engine? (cont.)

- Works on words, not on substrings
  auto != automatic, automobile

- Indexing process:
  - Convert document
  - Extract text and meta data
  - Normalize text
  - Write (inverted) index

- Example:
  Document 1: “Apache Lucene at Jazoon“
  Document 2: “Jazoon conference“

Index:
  apache -> 1
  conference -> 2
  jazoon -> 1, 2
  lucene -> 1
Apache Lucene Overview

> Lucene Java 2.2
  – Java library

> Solr 1.2
  – http-based index and search server

> Nutch 0.9
  – Internet search engine software

> http://lucene.apache.org
Lucene Java

> Java library for indexing and searching
> No dependencies (not even a logging framework)
> Works with Java 1.4 or later
> Input for indexing: Document objects
  – Each document: set of Fields, field name: field content (plain text)
> Input for searching: query strings or Query objects
> Stores its index as files on disk
> No document converters
> No web crawler
Lucene Java Users

> IBM OmniFind Yahoo! Edition
> technorati.com
> Eclipse
> Furl
> Nuxeo ECM
> Monster.com
> ...


Lucene Java Features

- Powerful query syntax
- Create queries from user input or programmatically
- Fast indexing
- Fast searching
- Sorting by relevance or other fields
- Large and active community
- Apache License 2.0
Lucene Query Syntax

- Query examples:
  - jazoon
  - jazoon AND java $\iff$ +jazoon +java
  - jazoon OR java
  - jazoon NOT php $\iff$ jazoon -php
  - conference AND (java OR j2ee)
  - “Java conference“
  - title:jazoon
  - j?zoon
  - jaz*
  - schmidt~ schmidt, schmit, schmitt
  - price:[000 TO 050]
  - + more
Lucene Code Example: Indexing

01 Analyzer analyzer = new StandardAnalyzer();
02 IndexWriter iw = new IndexWriter("/tmp/testindex", analyzer, true);
03
04 Document doc = new Document();
05 doc.add(new Field("body", "This is my TEST document", Field.Store.YES, Field.Index.TOKENIZED));
06 iw.addDocument(doc);
07
08 iw.optimize();
09 iw.close();

StandardAnalyzer: my, test, document
Lucene Code Example: Searching

01 Analyzer analyzer = new StandardAnalyzer();
02 IndexSearcher is = new IndexSearcher("/tmp/testindex");
03
04 QueryParser qp = new QueryParser("body", analyzer);
05 String userInput = "document AND test";
06 Query q = qp.parse(userInput);
07 Hits hits = is.search(q);
08 for (Iterator iter = hits.iterator(); iter.hasNext();) {
09     Hit hit = (Hit) iter.next();
10     System.out.println(hit.getScore() + " " + hit.get("body"));
11 }
12
13 is.close();
Lucene Hints

> Tools:
  – Lucli

> Common pitfalls and misconceptions
  – Limit to 10,000 tokens by default – see IndexWriter.setMaxFieldLength()
  – There’s no error if a field doesn’t exist
  – You cannot update single fields
  – You cannot “join” tables (Lucene is based on documents, not tables)
  – Lucene works on strings only -> 42 is between 1 and 9
    ➔ Use “0042“
  – Do not misuse Lucene as a database
Advanced Lucene Java

> Text normalization (Analyzer)
  – Tokenize foo-bar: text -> foo, bar, text
  – Lowercase
  – Linguistic normalization (children -> child)
  – Stopword removal (the, a, ...)
  ➔ You can create your own Analyzer (search + index)

> Ranking algorithm
  – TF-IDF (term frequency – inverse document frequency)
  – You can add your own algorithm
  – Difficult to evaluate
Lucene Java: How to get Started

> API docs

> FAQ
  - http://wiki.apache.org/lucene-java/LuceneFAQ
Lucene Java Summary

> Java Library for indexing and searching
> Lightweight / no dependencies
> Powerful and fast
> No document conversion
> No end-user front-end
Solr

- An index and search server (jetty)
- A web application
- Requires Java 5.0 or later
- Builds on Lucene Java
- Programming only to build and parse XML
  - No programming at all using Cocoon
- Communicates via HTTP
  - Index: use HTTP POST to index XML
  - Search: use GET request, Solr returns XML
    - Parameters e.g.
      - q = query
      - start
      - rows
  - Future versions will make use without HTTP easier (Java API)
Solr Indexing Example

> http POST to http://localhost:8983/solr/update

```xml
<add>
  <doc>
    <field name="url">http://www.myhost.org/solr-rocks.html</field>
    <field name="title">Solr is great</field>
    <field name="creationDate">2007-06-25T12:04:00.000Z</field>
    <field name="content">Solr is a great open source search server. It scales, it's easy to configure....</field>
  </doc>
</add>

> Delete a document: POST this XML:
  <delete><query>myID:12345</query></delete>
Solr Search Example

GET this URL: http://localhost:8983/solr/select/?indent=on&q=solr

Response (simplified!):

```xml
<response>
  <result name="response" numFound="1" start="0" maxScore="1.0">
    <doc>
      <float name="score">1.0</float>
      <str name="title">Solr is Great</str>
      <str name="url">http://www.myhost.org/solr-rocks.html</str>
    </doc>
  </result>
</response>
```
Solr Faceted Browsing

> Makes it easy to browse large search results

- **Handbook of genetic algorithms**

- **An introduction to genetic algorithms**
  by Mitchell, Melanie

- **COGANN-92, International Workshop on Combinations of Genetic Algorithms and Neural Networks, June 6, 1992, Baltimore, Maryland**
  by International Workshop on Combinations of Genetic Algorithms and Neural Networks (1992 : Baltimore, Md.)

**Refine your results**

- Genetic algorithms (6)
- Artificial intelligence (3)
- Combinatorial optimization (3)
- Neural networks (Computer science) (3)
- Algorithms (2)
- Data processing (2)
- Expert systems (Computer science) (2)
- Machine learning (2)
- Mathematical models (2)
- Biological applications (1)
Solr Faceted Browsing (cont.)

schema.xml:
<field name="topic" type="string" indexed="true" stored="true"/>

Query URL:
http://.../select?facet=true&facet.field=topic

Output from Solr:
<lst name="topic">
  <int name="Genetic algorithms">6</int>
  <int name="Artificial intelligence">3</int>
  ...

Solr: How to get Started

> Download Solr 1.2
> Install the WAR
> Use the post.jar from the exampledocs directory to index some documents
> Browse to the admin panel at http://localhost:8080/solr/admin/ and make some searches
> Configure schema.xml and solrconfig.xml in WEB-INF/classes

> Details at “Search smarter with Apache Solr“
> FAQ
Solr Summary

> A search server
> Access via XML sent over http
>   - Client doesn't need to be Java
> Web-based administration panel
> Like Lucene Java, it does no document conversion
> Security: make sure your Solr server cannot be accessed from outside!
Nutch

- Internet search engine software (software only, not the search service)
- Builds on Lucene Java for indexing and search
- Command line for indexing
- Web application for searching
- Contains a web crawler
- Adds document converters

Issues:
- Scalability
- Crawler Politeness
- Crawler Management
- Web Spam
Nutch Users

> Internet Archive
  – www.archive.org

> Krugle
  – krugle.com

> Several vertical search engines, see
  http://wiki.apache.org/nutch/PublicServers
Getting started with Nutch

> Download Nutch 0.9 (try SVN in case of problems)

> Indexing:
  - add start URLs to a text file
  - configure conf/crawl-urlfilter.txt
  - configure conf/nutch-site.xml
  - command line call
    \[ \texttt{bin/nutch crawl urls -dir crawl -depth 3 -topN 50} \]

> Searching:
  - install the WAR
  - search at e.g. \[ \texttt{http://localhost:8080/} \]
Getting started with Nutch (cont.)

![Nutch Interface](http://localhost:8080/nutch-1.0-dev/en/)

- [ ] Location: `http://localhost:8080/nutch-1.0-dev/en/`
Getting started with Nutch (cont.)

LanguageTool - style and **grammar** checker tree.pl - perl script...
http://localhost/homepage/ (cached) (explain) (anchors) (more from localhost)

**LanguageTool: an Open Source language checker for English, German, Polish, and more**
... It can also detect some **grammar** mistakes. It does not include...
http://localhost/homepage/languagetool/ (cached) (explain) (anchors) (more from localhost)

*show all hits*
Nutch Summary

- Powerful for vertical search engines
- Meant for indexing Intranet/Internet via http, indexing local files is possible with some configuration
- Not as mature as Lucene and Solr yet
- You will need to invest some time
Other Lucene Features

> „Did you mean...“
  – Spell checker based on the terms in the index
  – See contrib/spellchecker in Lucene Java

> Find similar documents
  – Selects documents similar to a given document, based on the document's significant terms
  – See contrib/queries MoreLikeThis.java in Lucene Java

> NON-features: security
  – Lucene doesn't care about security!
    ➔ You need to filter results yourself
    ➔ For Solr, you need to secure http access
Other Projects at Apache Lucene

- Hadoop - a distributed computing platform
  - Map/Reduce
  - Used by Nutch

- Lucene.Net - C# port of Lucene, compatible on any level (API, index, ...)
  - Used by Beagle, Wikipedia, ...
Lucene project – The big Picture

> Lucene: Java fulltext search library

> Solr = Lucene Java
- + Web administration frontend
- + HTTP frontend
- + Typed fields (schema)
- + Faceted Browsing
- + Configurable Caching
- + XML configuration, no Java needed
- + Document IDs
- + Replication

> Nutch = Lucene Java + Hadoop
- + Web crawler
- + Document converters
- + Web search frontend
- + Link analysis
- + Distributed search
Alternative Solutions for Search

> Commercial vendors (FAST, Autonomy, Google, ...)
  – Enterprise search

> Commercial search engines based on Lucene and Lucene support (see Wiki)
  – IBM OmniFind Yahoo! Edition

> RDBMS with integrated search features
  – Lucene has more powerful syntax and can be easily adapted and integrated

> Egothor
  – Lucene has a much bigger community
Conclusion

- no “Enterprise Search” (but: Intranet indexing using Nutch)

+ can be embedded or integrated in almost any situation
+ fast
+ powerful
+ large, helpful community
+ the quasi-standard in Open Source search