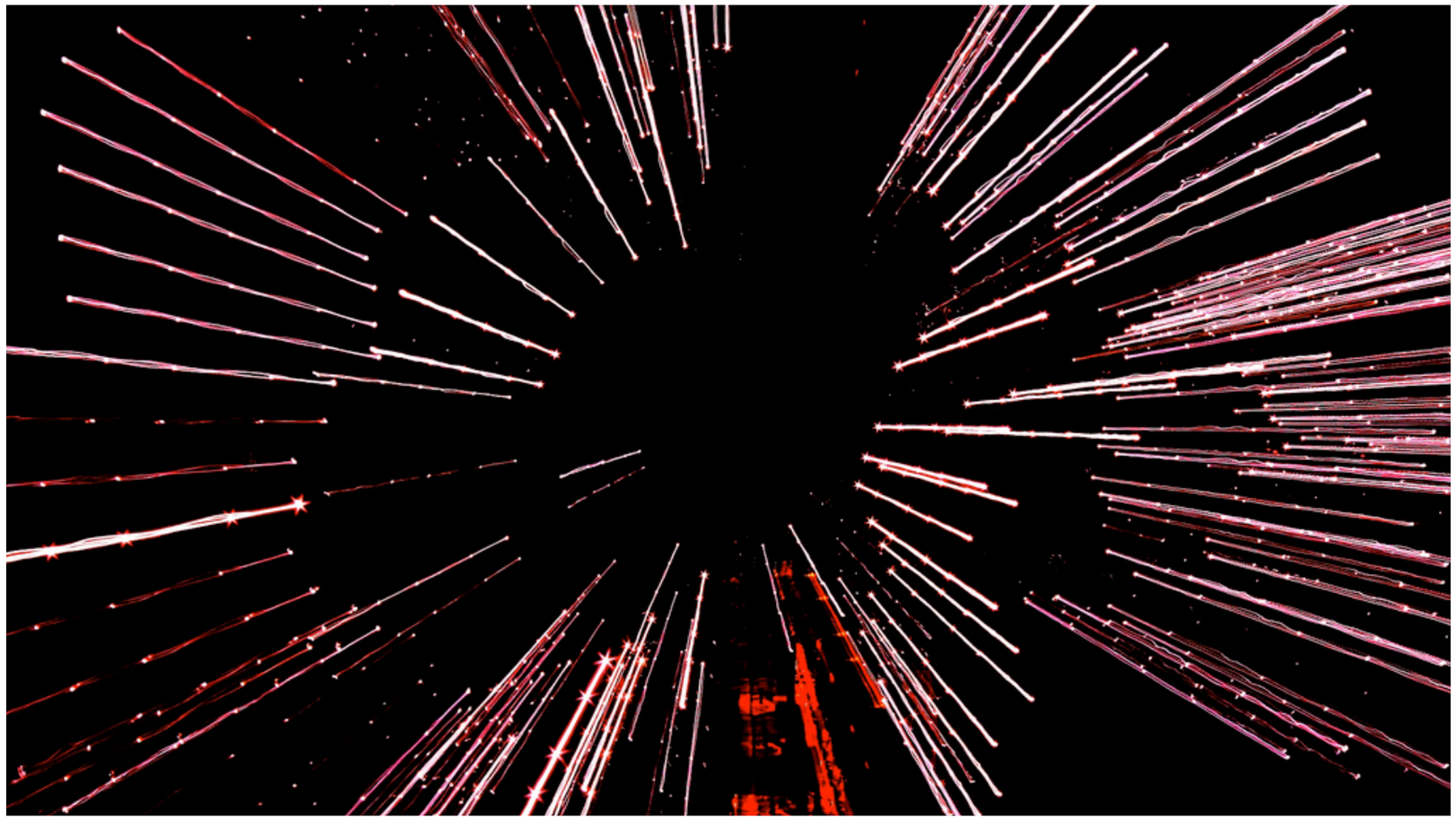
Solr Boot Camp

Instructor: Erik Hatcher
erik.hatcher@lucidimagination.com

Prerequisites

- Solr 1.3 (and Java5)
- These slides
- Web browser open to: http://wiki.apache.org/solr



Solr Story

Solr History

- Created by Yonik Seeley for CNET
- Contributed to Apache in January 2006
- December 2006: Version 1.1 released
- June 2007: Version 1.2 released
- September 2008: Version 1.3 released

Features

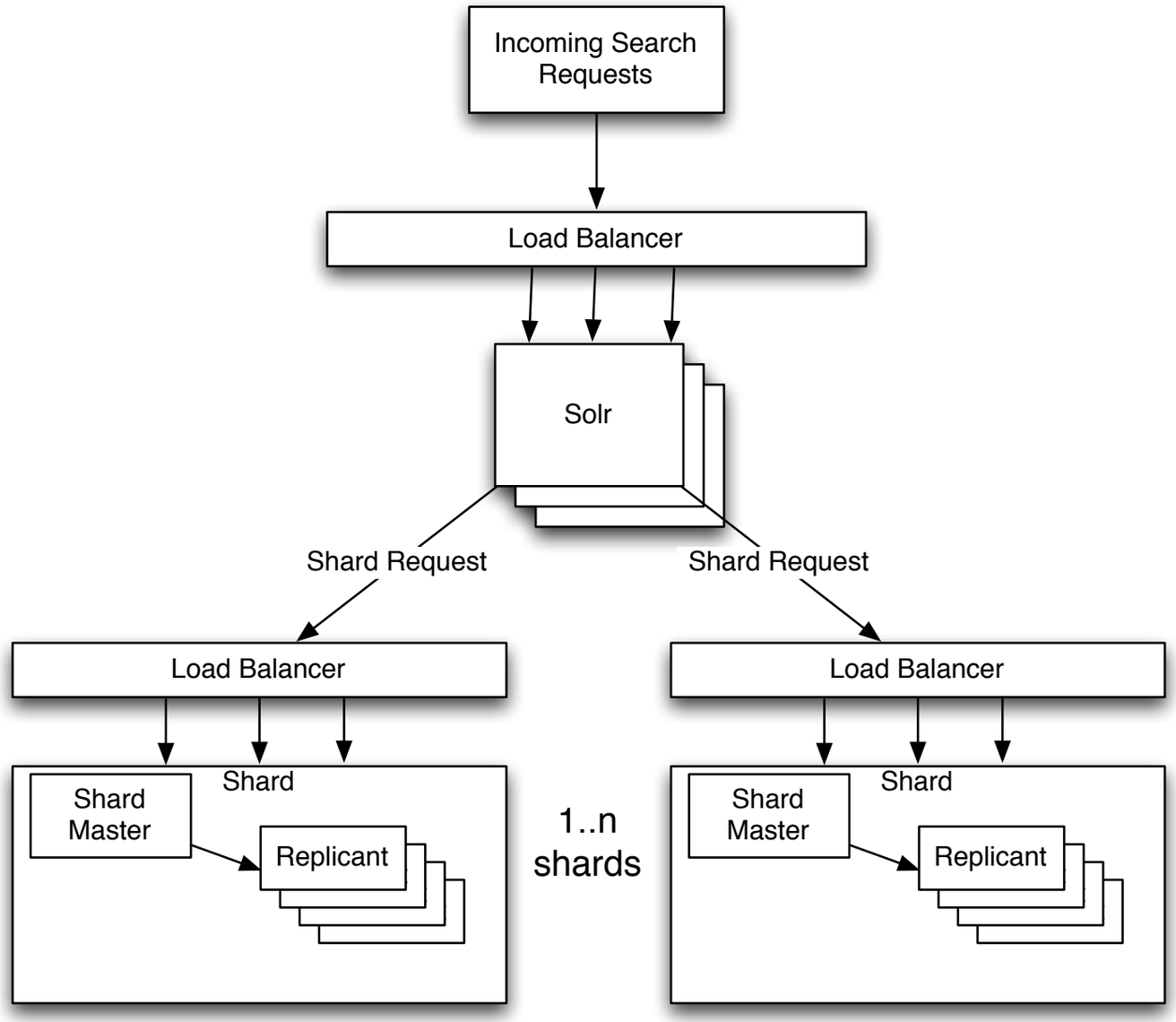
- Lucene power exposed over HTTP
- Caching
- Replication
- Faceting
- Distributed search
- etc, etc, etc

Solr APIs

- HTTP GET/POST (curl or any other HTTP client)
- SolrJ (embedded or HTTP)
- solr-ruby
- python, PHP, solrsharp, XSLT, JSON

Deployment Architecture

- Scales from:
  - single Solr server
  - master/ replicant(slave)
  - distributed shards
- Each server can also have multiple cores

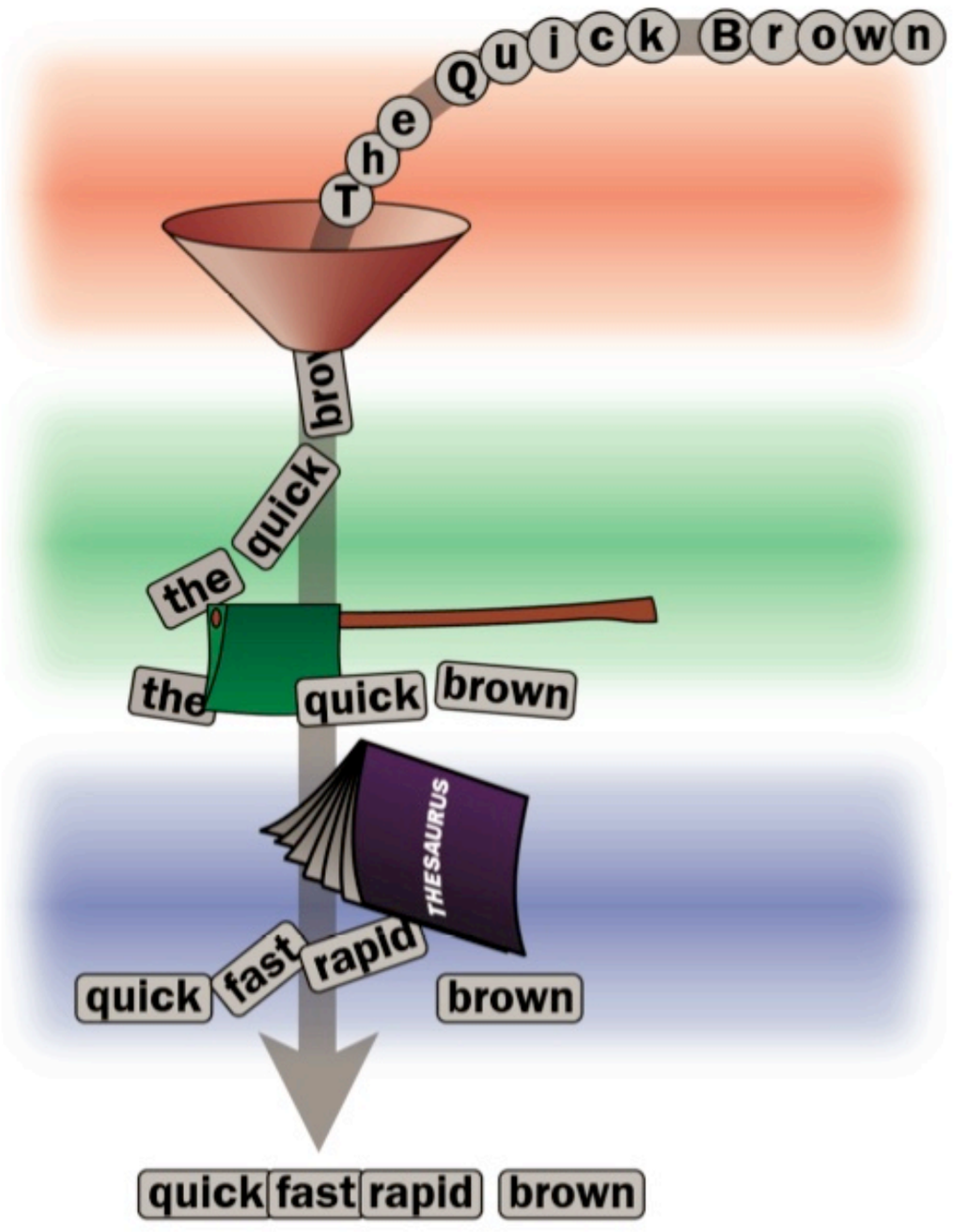


Lucene Fundamentals

Concepts

- Index
- Document
- Field
- Terms (aka Tokens)

Analysis Process



Analyzing the analyzer

Example phrase

```
The quick brown fox jumps over the lazy dog.
```

WhitespaceAnalyzer

Simplest built-in analyzer

```
The quick brown fox jumps over the lazy dog.
```

```
[The] [quick] [brown] [fox] [jumps] [over] [the]
[lazy] [dog.]
```

SimpleAnalyzer

Lowercases, splits at non-letter boundaries

the quick brown fox jumps over the lazy dog.

[the] [quick] [brown] [fox] [jumps] [over] [the] [lazy] [dog]

StopAnalyzer

Lowercases and removes *stop* words

The quick brown fox jumps over the lazy dog.

[quick] [brown] [fox] [jumps] [over] [lazy] [dog]

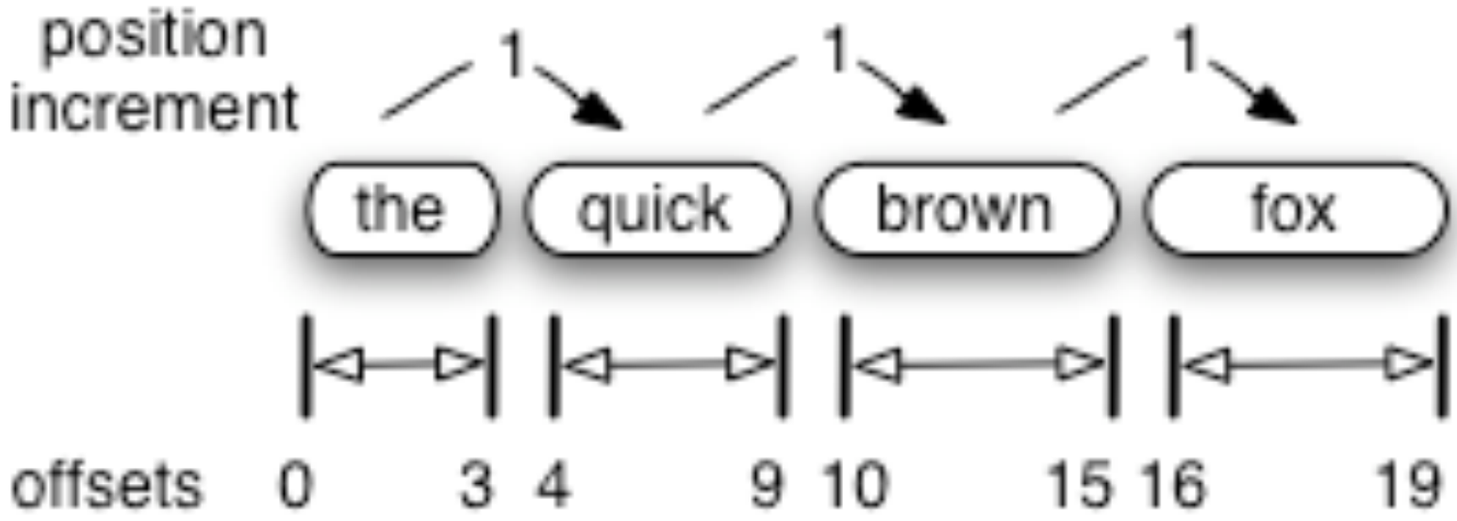
SnowballAnalyzer

Stemming algorithm

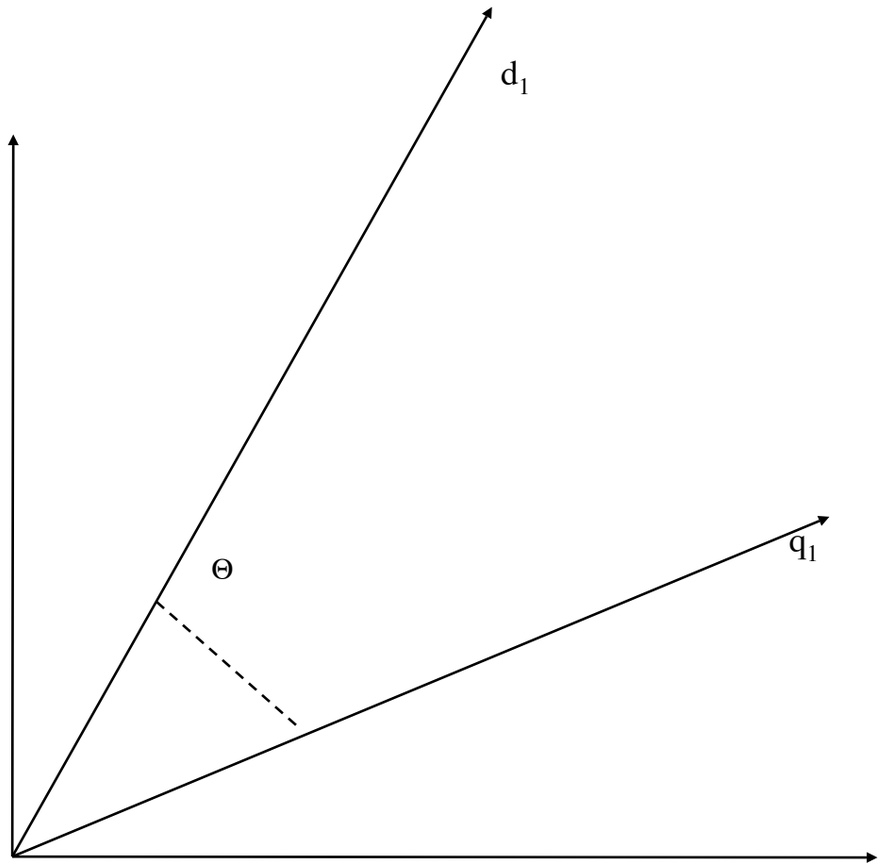
`The quick brown fox jumps over the lazi dog.`

[the] [quick] [brown] [fox] [jump] [over] [the] [lazi] [dog]

What's in a token?



Lucene Scoring



$$\text{score}(q,d) = \text{coord}(q,d) \cdot \text{queryNorm}(q) \cdot \sum_{t \text{ in } q} \left(\text{tf}(t \text{ in } d) \cdot \text{idf}(t)^2 \cdot \text{t.getBoost}() \cdot \text{norm}(t,d) \right)$$

Solr Quick Start

Solr Quick Start

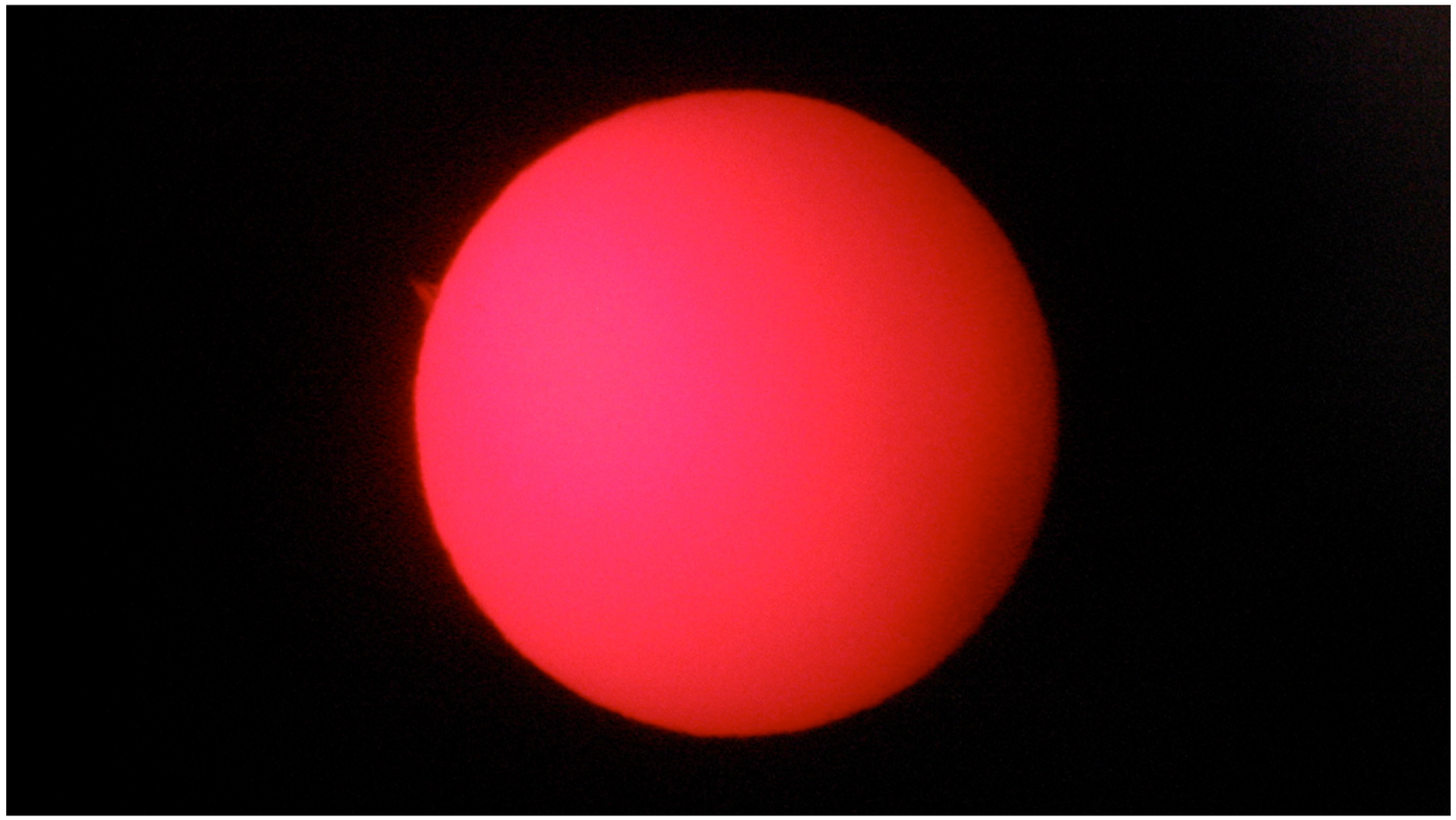
- Download Solr 1.3.0
- Follow Solr tutorial instructions
  - docs/tutorial.pdf or docs/tutorial.html
- cd example; java -jar start.jar

Solr Admin

- http://localhost:8983/solr/admin/
- Stats, schema browser, analysis analyzer, run-time logging configuration, and convenience links to standard request handlers

Exercise

Tutorial steps through section 6 (Querying Data). Poke around /admin



Configuration

Solr Core

- A single primary index
  [spell checking uses additional index(es) per core]
- schema.xml / solrconfig.xml
- Multiple cores per Solr instance, configured in solr.xml

schema.xml

- Field types
- Fields
- Unique key (optional)
- copy fields
- similarity and Solr query parser configuration

Schema Analysis

- http://localhost:8983/solr/admin/analysis.jsp
- Analysis request handler:

```
curl http://localhost:8983/
solr/analysis --data-binary
@ipod_video.xml -H 'Content-
type:text/xml; charset=utf-8'
```

Exercise

Tutorial section 7 (Text Analysis), experiment with different field types including tinkering with custom analysis configuration

solrconfig.xml

- Lucene indexing parameters
- Cache settings
- Request handler configuration
- HTTP cache settings
- Search components, response writers, query parsers

Request handlers

- mini-“servlets”
- SearchHandler extensions chain search components
- Flexible response formatting:
  - &wt=[json, ruby, xslt, php, phps, javabin, python]

Exercise

Create a custom request handler mapping, experiment with defaults, appends, and invariants

Updating

Solr XML

```
<add><doc>
  <field name="id">MA147LL/A</field>
  <field name="name">Apple 60 GB iPod with Video Playback Black</field>
  <field name="manu">Apple Computer Inc.</field>
  <field name="cat">electronics</field>
  <field name="cat">music</field>
  <field name="features">iTunes, Podcasts, Audiobooks</field>
  <field name="features">Stores up to 15,000 songs, 25,000 photos, or 150 hours of
              video</field>
  <field name="features">2.5-inch, 320x240 color TFT LCD display
                        with LED backlight</field>
  <field name="features">Up to 20 hours of battery life</field>
  <field name="features">Plays AAC, MP3, WAV, AIFF, Audible, Apple Lossless,
                        H.264 video</field>
  <field name="features">Notes, Calendar, Phone book, Hold button, Date display,
      Photo wallet, Built-in games, JPEG photo playback, Upgradeable firmware,
      USB 2.0 compatibility, Playback speed control, Rechargeable capability,
      Battery level indication</field>
  <field name="includes">earbud headphones, USB cable</field>
  <field name="weight">5.5</field>
  <field name="price">399.00</field>
  <field name="popularity">10</field>
  <field name="inStock">true</field>
</doc></add>
```

Indexing Solr XML

- Via curl:

```
curl 'http://localhost:8983/
solr/update?commit=true' --
data-binary @ipod_video.xml -
H 'Content-type:text/xml;
charset=utf-8'
```

- Via pure Java:
  java -jar post.jar ipod_video.xml

Indexing CSV

```
curl 'http://localhost:8983/solr/update/
csv?commit=true' --data-binary @books.csv -
H 'Content-type:text/plain; charset=utf-8'
```

Content Streams

- Allows Solr server to fetch local or remote data itself. Must enable remote streaming in solrconfig.xml
- http://localhost:8983/solr/update?stream.file=<local Solr path to exampledocs>/ipod_video.xml
- &stream.url=<url to content>
- Security warning: allows Solr to fetch arbitrary server-side file or network URL content

Indexing with SolrJ

```
SolrServer solr =
    new CommonsHttpSolrServer(new URL("http://localhost:8983/solr"));

SolrInputDocument doc = new SolrInputDocument();
doc.addField("id", "EXAMPLEDOC01");
doc.addField("title", "Solr Boot Camp SolrJ Example");
solr.add(doc);
solr.commit();     // after a batch, not per document
solr.optimize();   // periodically, when needed
```

Indexing with Ruby

```
solr = Connection.new(
  'http://localhost:8983/solr',
  :autocommit => :on)

solr.add(:id => 123,
         :title => 'Solr in Action')

solr.optimize  # periodically, as needed
```

delete, update, etc

- Delete:
  - <delete><id>05991</id></delete>
  - <delete>
    <query>category:Unused</query>
    </delete>
    - `java -Ddata=args -jar post.jar "<delete><query>*:*</query></delete>"`
- Update: simply <add> doc with same unique key
- Commit: <commit/>
- Optimize: <optimize/>

Data Import Handler

- Indexes relational database and XML data sources
- Supports full and incremental/delta indexing
- Highly extensible with custom data sources, transformers, etc
- http://wiki.apache.org/solr/DataImportHandler

DIH details

- Put JDBC driver JAR in <solr-home>/lib, configure dataimport request handler
- http://localhost:8983/solr/db/admin/dataimport.jsp - debugging console
- http://localhost:8983/solr/db/dataimport?command=full-import - removes all documents and imports from scratch

Exercise

Index some of your own data, by script and/or programmatically

Indexing Performance

- Where feasible:
  - add documents in batches
  - multithread / multiprocess
- Tune solrconfig.xml indexing settings
- Commit only when needed
- Optimize when you have enough time or are offline (on a master)

Basic Searching

Standard Search Request

- http://localhost:8983/solr/select?q=query

Debug Query

- &debugQuery=true is your friend
- Includes parsed query, explanations, and search component timings in response

Searching

- Send GET HTTP requests
  - http://localhost:8983/solr/select?q=solr&start=0&rows=10
- DisMaxRequestHandler
  - http://wiki.apache.org/solr/DisMaxRequestHandler
  - Simplified query parsing

Query Parser

- Controlled by defType parameter
  - &defType=lucene (actually a Solr extension of Lucene's QueryParser)
  - &defType=dismax

Solr Query Parser

- http://lucene.apache.org/java/2_4_0/queryparsersyntax.html + Solr extensions
- Kitchen sink parser, includes advanced user-unfriendly syntax
- Syntax errors throw parse exceptions back to client
- Example: title:ipod* AND price:[0 TO 100]

Dismax Query Parser

- Simplified syntax:
  loose text "quote phrases" -prohibited
  +required
- Spreads query terms across query fields (qf) with dynamic boosting per field, implicit phrase construction (pf), and boosting capability (bf)

Searching with SolrJ

```
SolrServer server = new CommonsHttpSolrServer("http://
  localhost:8983/solr");
SolrQuery params = new SolrQuery("author:John");
params.setFields("*,score");
params.setRows(3);
QueryResponse response = server.query(params);
for (SolrDocument document : response.getResults()) {
      System.out.println("Doc: " + document);
}
```

Searching with Ruby

```
conn = Connection.new(
  'http://localhost:8983/solr')

conn.query('my query') do |hit|
  puts hit.inspect
end
```

Searching Performance

- Optimized index
- Non-compound index format
- Cache tuning (solrconfig.xml)
- Searcher warming
- Query complexity: many terms, wildcard terms
- Faceting concerns
- Searcher replication

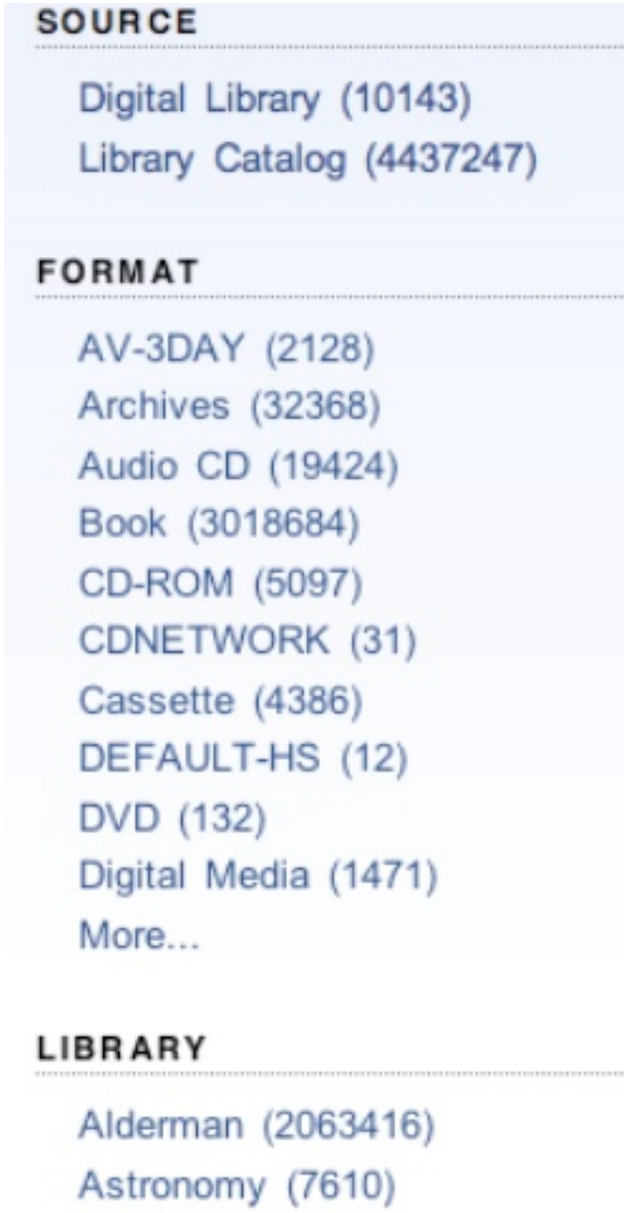
Exercise

Play around with searches using dismax and standard Lucene query parsing. Also, programmatically search Solr from your language of choice.

Search Components

Faceting

- Counts per subset within results
- Facet on: field terms, queries, date ranges
- &facet=on
  &facet.field=cat
  &facet.query=price:[0 TO 100]
- http://wiki.apache.org/solr/SimpleFacetParameters



Spell checking

- Not enabled by default, see example config to wire it in
- http://localhost:8983/solr/spellCheckCompRH?q=epod&spellcheck=on&spellcheck.build=true
- File or index-based dictionaries
- Supports pluggable distance algorithms: Levenstein and JaroWinkler
- http://wiki.apache.org/solr/SpellCheckComponent

Highlighting

- http://localhost:8983/solr/select?q=apple&hl=on&hl.fl=manu,name
- http://wiki.apache.org/solr/HighlightingParameters

More Like This

- http://localhost:8983/solr/select?q=apache&mlt=true&mlt.fl=manu,cat&mlt.mindf=1&mlt.mintf=1&fl=id,score
- http://wiki.apache.org/solr/MoreLikeThis

Query Elevation

- http://localhost:8983/solr/elevate?q=ipod&debugQuery=true&enableElevation=true
- Configure an “elevate.xml” to boost/exclude specific documents
- http://wiki.apache.org/solr/QueryElevationComponent

Exercise

Experiment with the various search components: faceting, spell checking, highlighting, more like this, and query elevation

Additional Request Handlers

stats.jsp

- Not technically a “request handler”, outputs only XML
- http://localhost:8983/solr/admin/stats.jsp
- Index stats such as number of documents, searcher open time
- Request handler details, number of requests and errors, average request time, average requests per second, number of pending docs, etc, etc

Dump

- http://localhost:8983/solr/debug/dump
- Echoes parameters, content streams, and Solr web context

Ping

- http://localhost:8983/solr/admin/ping
- If healthcheck configured and file not available, error is reported
- Executes single configured request and reports failure or OK

Luke

- http://localhost:8983/solr/admin/luke
- Introspects Lucene index structure and schema relationships
- See an individual document:
  - ?doc=<key> or ?docId=<lucene doc #>
- Schema details: ?show=schema
- Admin schema browser uses Luke request handler
- See also: original Luke tool - http://www.getopt.org/luke/

System

- http://localhost:8983/solr/admin/system
- core info, Lucene version, JVM details, uptime, operating system info

Plugins

- http://localhost:8983/solr/admin/plugins
- Configuration details of Solr core, available query and update handlers, cache settings

Threads

- http://localhost:8983/solr/admin/threads
- JVM thread details

Properties

- http://localhost:8983/solr/admin/properties
- All JVM system properties, or single property value (?name=os.arch)

File

- http://localhost:8983/solr/admin/file?file=/
  - See all fetchable files / directories
- http://localhost:8983/solr/admin/file?file=schema.xml&contentType=text/plain

Exercise

Play around with the various request handlers

Administration

Production

- Solr provides tools/support for:
  - caching, warming, replication
- Systems concerns:
  - CPU, Memory, Disk Space, Servlet Containers, Security, Backups, Fail over

Deployment Environment

- Usual answer for CPU/Memory/Disk Space: Bigger is better
- Servlet Containers
  - Use whatever you know how to maintain and make perform
  - Solr has been used in a wide variety of containers: Jetty, Tomcat, Resin...
- See "Installation and Configuration" on http://wiki.apache.org/solr

Security

- Solr doesn't secure either the documents or the communication protocols
- Use standard security practices:
  - Firewall
  - Secure ports
- Solr specifics:
  - Invariant parameters for RequestHandlers
  - Disable remoteStreaming (disabled by default)
  - Remove update handlers for read-only Solr instances?
  - Turn off "file" request handler, and perhaps other admin handlers

What's new since 1.3?

- Java-based replication
- VelocityResponseWriter
- Logging switched to SLF4J
- Rollback, since last commit
- StatsComponent
- TermVectorComponent

Resources

- http://wiki.apache.org/solr
- solr-user@lucene.apache.org

Lucid Imagination

- Support for all things lucene.apache.org, including Java Lucene and Solr
- Value-added components
- Training
- Services