

Searching large-text repositories:

Google in a box with LAMP tools

Shahzad Khan

# Shahzad Khan

- @ DISTIL
  - Senior Scientist
  - Automatic assessment, NLG, Analytics
  - [textai.blogspot.com](http://textai.blogspot.com) (DISTIL Blog)
  - [s.khan2@distilinteractive.com](mailto:s.khan2@distilinteractive.com)

# Virtual Persona

- Consultancy
  - [Whyztech.com](http://Whyztech.com)
- Twitter
  - fluxhawk
- LinkedIn
  - [www.linkedin.com/in/skhansj](http://www.linkedin.com/in/skhansj)

- Shahzad Khan
  - PhD
    - Computational Linguistics
    - Cambridge (UK)
  - 15 publications in information retrieval, machine translation, automatic text classification, summarization and word-sense-disambiguation

# Today..

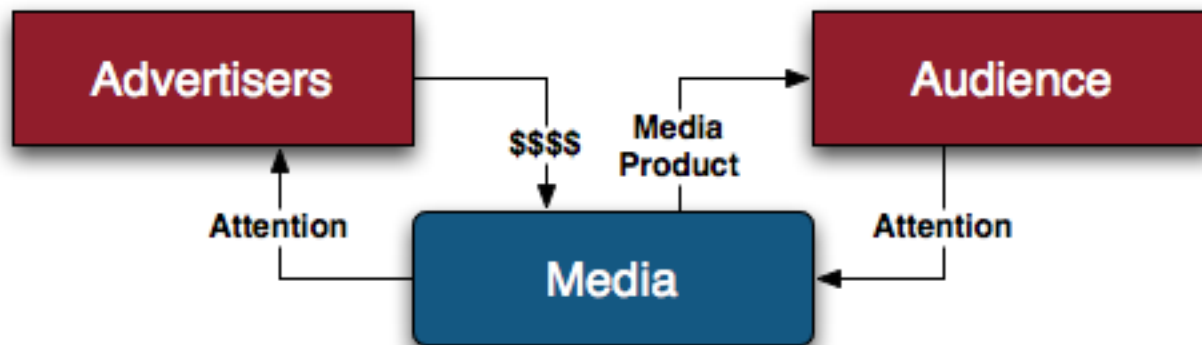
- Web Search
- Enterprise Search
- Free-Text Search
- Open Source Free-text Search Tools

# Question

- Which is your favorite search engine?
  - Google
  - Yahoo
  - Ask.com
  - Cuil.com
  - Microsoft Live Search
  - Others

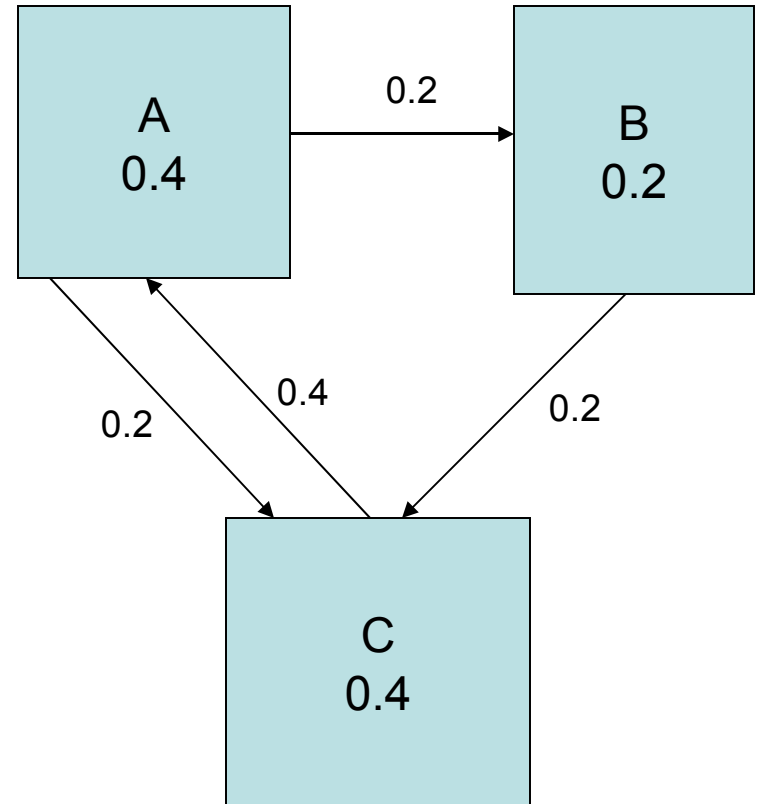
# Information Economics

- Direct Network Effect
- Value to me depends directly on number of adopters
  - Fax machine, telephone, email, IM
- Two Sided Market
  - Value depends on adoption of complementary products



# Search Engine: Technology

- Google PageRank
- Popularity Contest
  - Links = votes
  - Not democratic
  - Converges quickly
  - Works well



# Google PageRank



Credit:

<http://blaugh.com/2007/06/11/link-popularity-vs-pagerank-vs-yoda/>

# Enterprise Search

# Different Requirements

- No hyperlinks
  - Minimal cross-references
  - PageRank fails
- Applications driven
  - Articulating and fulfilling info needs
  - Customer segmentation
  - Knowledge management
  - e-discovery
  - Groupware

# Different Requirements

- *Full text* indexes and database fields
  - Linguistic processing
- Taxonomies and tagging of resources
  - ‘... we had the data, but we didn't have the information...’
  - Add metadata (entity recognition, etc)
- Register resources; retire resources
  - ‘.. digital landfills ..’
- Access control
  - ‘ .. we're on a need to know basis ..’

# More Requirements

- Administrators interfaces to update
  - Set access controls
  - Include/exclude target content in databases
  - Linguistic resource management
    - Taxonomies and thesauri
    - Gazetteers
  - Retrieval and results ranking algorithms
- Document adapters
  - Webpages are a small segment
  - Files, E-Mail, Databases ...
  - Content Management Systems
- Search engines...

# Search Technologies

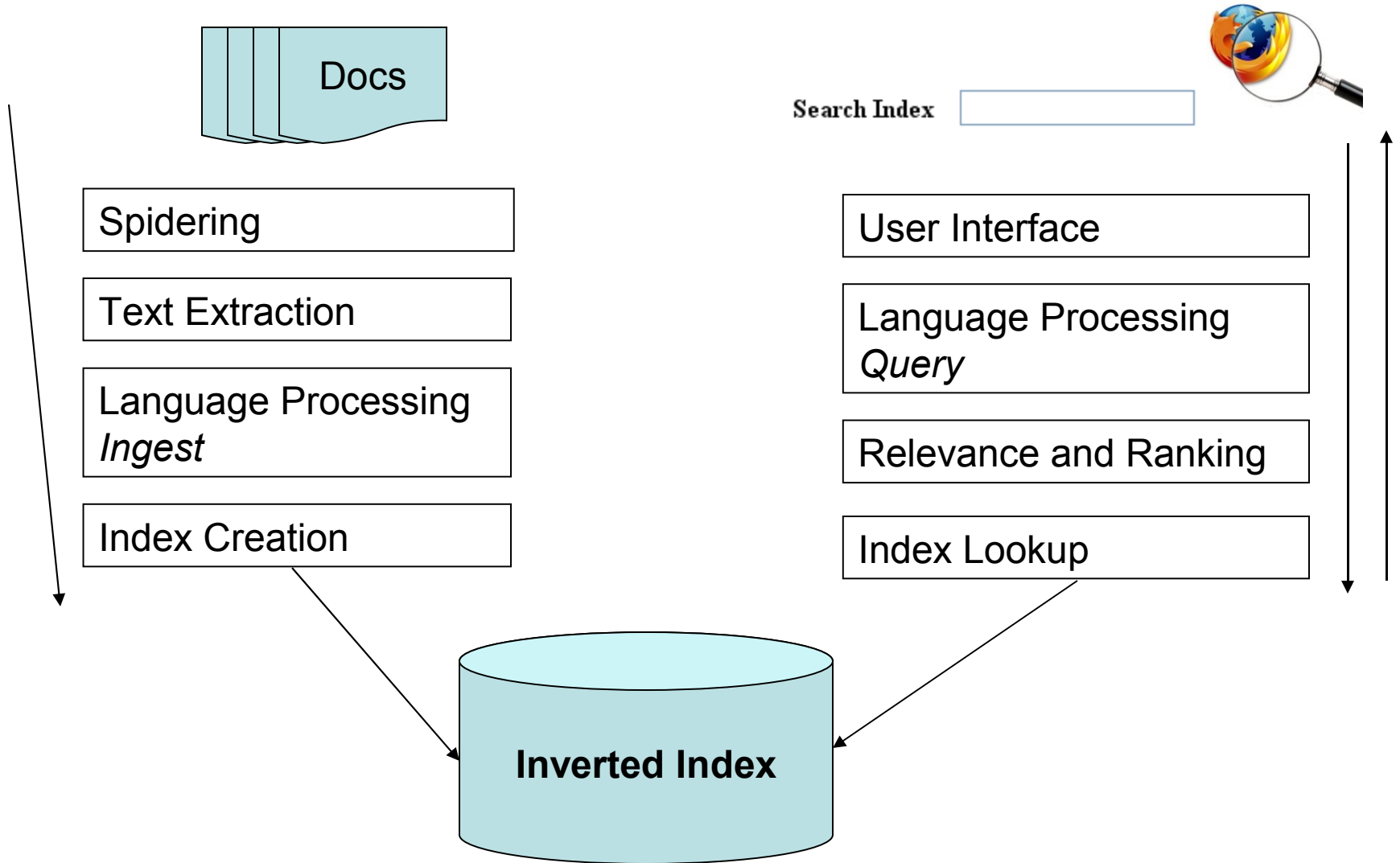
# Search Engine

- Builds an index on text
- Answers queries using that index

# Full Text Search

- Why not SQL?
  - Databases are structured
  - Text is not structured
  - Runtime full text searching is expensive!!
    - You really need a proper index
- A search engine offers
  - Scalability
  - Relevance Ranking
  - Integrates different data sources (email, web pages, files, database, ...)

# Full Text Search Architecture



# Inverted Index

- Doc Id
  - A unique “key” that identifies each document
- Documents are made up of **tokens**
- Term
  - Unique “key” that identifies each class of tokens
  - The token is the basic unit of indexing
  - Can be multiple instances of a given type in a document, each of which is a separate token.

# Inverted Index

- Traditional (back of book) Index:

Document\_ID  $\rightarrow$  {Term<sub>1</sub>, Term<sub>2</sub>.. Term<sub>N</sub>}

- Inverted Index:

Term\_ID  $\rightarrow$  {Document<sub>1</sub>, Document<sub>2</sub>.. Document<sub>N</sub>}

- *Reason:*

- Queries are represented as terms
- Terms should help to select candidate docs

# Boolean Search

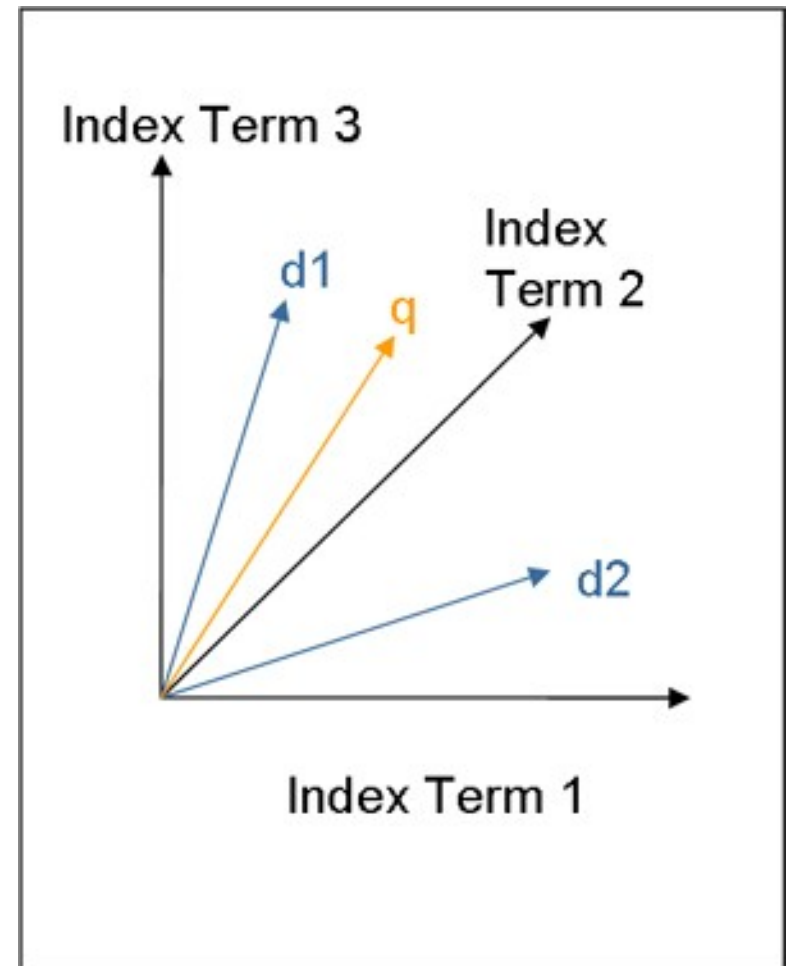
- Searches docs with at least 1 term from query
- Set operations (Venn Diagrams)
  - AND – Intersect
  - OR – Union
  - NOT – Complements
- Pro: Crisp Filtering
- Con: No ranking
- Expert Only Search Technique

# Vector Information Retrieval

- **Term Space**
  - each doc represented as vector
  - 30,000 vocabulary = 30,000 dimension
- Dimensions = unique words in entire collection
- Document's position based on contained words
  - Documents with common words closer

# Vector

- Step 1: Query represented as a document vector
- Step 2: Eliminate all documents that do not contain at least one keyword (Inverted Index op). Represent candidate docs as vectors.
- Step 3: Uses a cosine distance to determine how close candidate documents are to the query 'document' vector.
- Pro:
  - Superquick!
  - Supports ranking



# VSM Optimizations

- Reduce Dimensionality
  - Stop words
    - Function words
      - the, of, and, in...
  - Stemming (morphological normalization)
    - Computerization, computation, computer, compute
      - Reduced to: comput
- Increased Dimensionality
  - Expand with synonyms (query time)
    - Danger! Word-sense-disambiguation
- TF.IDF

# Open Source Implementations

# Apache Lucene



- A search API
  - Not a complete solution per-se
  - Good building block
  - Developed by Doug Cutting <Apache, Excite, Apple>
- Apache project
  - Active developers, books, etc.
  - Convince clients that platform is robust (no stranding)
  - High speed of engagement (as compared to commercial vendors)

# Apache Lucene

- Indexing: Document objects
  - Each document: set of
    - **field name: field content** (plain text)
- Searching: query strings or Query objects
- Limitations
  - Stores its index as files on disk
  - No document converters
  - No web crawler

# Lucene: Indexing

```
Analyzer analyzer = new StandardAnalyzer();
IndexWriter iw = new IndexWriter("/lucene/index", analyzer, true);
// .. Loop over documents
    Document doc = new Document();
    doc.add(new Field("body", "The fox jumped the dog",
        Field.Store.YES, Field.Index.TOKENIZED));
    iw.addDocument(doc);
// .. End loop
iw.optimize();
iw.close();
```

# Lucene: Query

```
Analyzer analyzer = new StandardAnalyzer();
IndexSearcher is = new IndexSearcher("/lucene/index");
QueryParser qp = new QueryParser("body", analyzer);
String userInput = "fox AND dog";
Query q = qp.parse(userInput);
Hits hits = is.search(q);
for (Iterator iter = hits.iterator(); iter.hasNext();) {
    Hit hit = (Hit) iter.next();
    System.out.println(hit.getScore() + " " + hit.get("body"));
}
is.close();
```

# Solr



- Developed and contributed by CNet
- Enterprise Search Server
- Supports multiple protocols (xml, json, ruby ...)
- Based on Lucene
- Programming only to build and parse XML
  - communicates via HTTP
    - index: use http POST to index XML
    - search: use GET request, Solr returns XML
      - Parameters e.g.
        - » q = query
        - » start
        - » rows

# Solr: Indexing

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

```
<add>
  <doc>
    <field name="url">http://www.myhost.org/solr-rocks.html</field>
    <field name="title">Solr rules!</field>
    <field name="creationDate">2009-02-08T12:04:00.000Z</field>
      <field name="content">Solr is a great search server.
    </field>
  </doc>
</add>
```

# Solr: Search

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

Response (simplified!):

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

# Solr: Faceted Browsing

- Navigational search: uses a hierarchy structure (taxonomy)
  - Yahoo! Directory, DMOZ, etc.
- Direct search: write queries as a bag of words in text box
  - Google
- Faceted search: navigate a multi-dimensional information space by combining text search with a progressive narrowing of choices in each dimension
  - You get ‘another chance’ at the search, and can browse as well
- Employs attributes/fields, and their values

# Nutch



- Web search application
  - Crawlers to gather pages
  - Link database, analysis and indexing
  - Command line for indexing
  - Web application for searching
  - Adds document converters

# Conclusion

- Web Search differs from Enterprise Search
- PageRank alternative;
  - classic VSM and Boolean search
- Open Source Tools Available
  - Lucene, Solr, Nutch

# Caveats

- Not all the pieces of enterprise search;
  - No silver bullet
  - Only indexing and query processing
  - Need know-how to translate data into information
  - Workflow requires significant investment
- Most commercial enterprise search project delayed/cancelled due to lack of expertise
- Open-source: Lucene, Solr, Nutch, Tika etc
- Proprietary Alternatives:
  - Autonomy, Endeca, FAST (now MSoft)
  - Google Search Appliance, MSoft Sharepoint

# Thank you

- Shahzad Khan
- shahzad@whyztech.com