Improved Search with Lucene 4.0

Robert Muir, Lucid Imagination
robert.muir@lucidimagination.com, 9/22/2011

Presented by

lucid IMAGINATION

Lucene

Apache Solr

Introductions

- What: Examine improvements in 4.0
- Who am I?
  - **Lucene Committer & PMC Member**
  - **Lucid Imagination Employee**
- Many big changes coming!
- Examine three general areas:
  - **Indexing Improvements**
  - **Search Improvements**
  - **Performance Improvements**
- Future improvements

Indexing Improvements

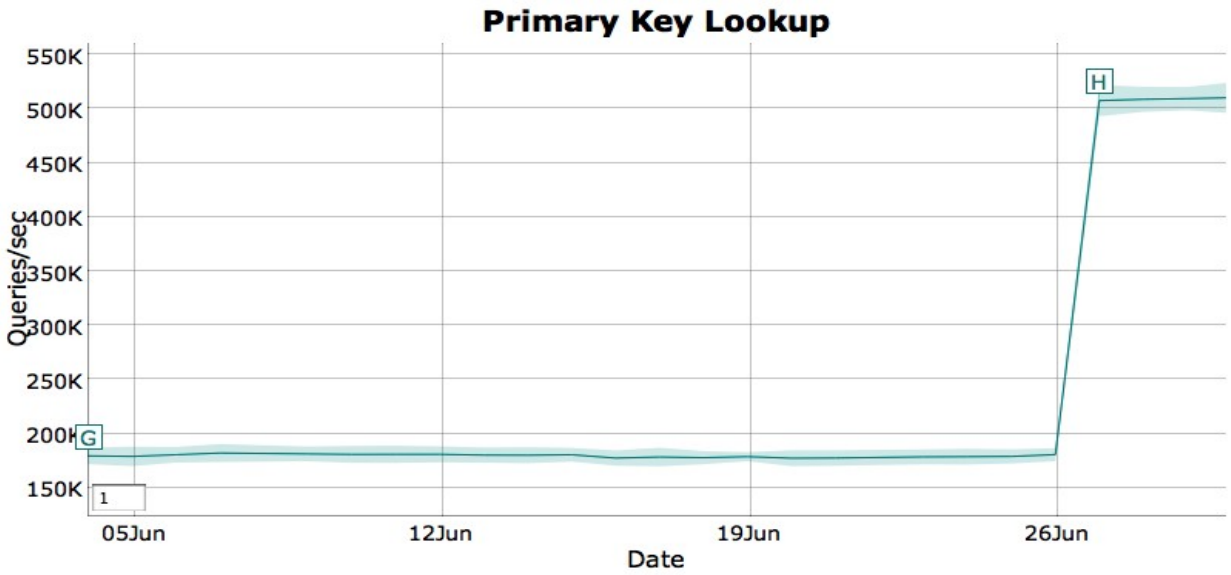
- Codecs
  - **Flexible index format**
- Index DocValues
  - **Efficient per-document lookup values**
- Miscellaneous Improvements
  - **Binary terms**
  - **Additional index statistics**

Codecs: Introduction

- Problem: “one size fits all” index format
- How to integrate future improvements?
  - **Example: more efficient index compression**
  - **But need to support old format, too.**
- How to optimize for your data?
  - **Without digging into the guts of indexer!**
- Idea: format should be pluggable

Codecs: Example use case

- Accelerate near-realtime reopen
- Speed up delete by term on ID field
  - **“Primary key lookup”**
- Idea: optimize ID field for this use case



Codecs: available formats

- Standard: Lucene 4.0 index format
- Pulsing: inline low frequency terms' postings
- Memory: loads field entirely into RAM
- Appending: supports append-only filesystem
- SimpleText: plaintext (not for production!!!!!)
- PreFlex: supports Lucene 3.x index format

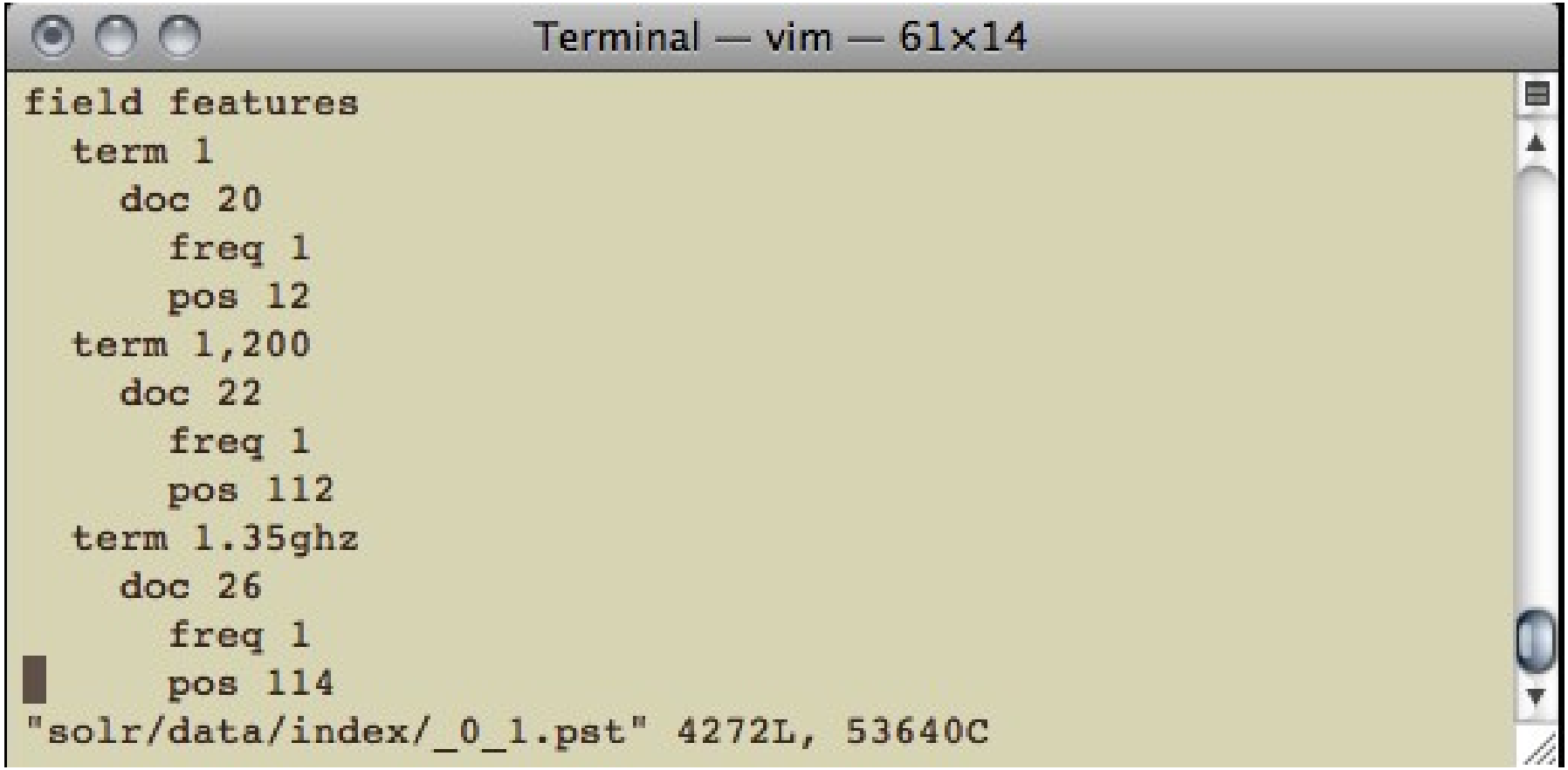
Codecs: Configuration

Lucene: use *CodecProvider* class

Solr: specify in schema.xml

```
<fieldType name="text_general" class="solr.TextField"
 codec="SimpleText">
```

Codecs: Configuration



Search Improvements

- Improved Scoring API
  - **Additional Scoring algorithms**
- Spellchecking improvements
- Miscellaneous
  - **Query parsing improvements**
  - **Deep paging support**

Scoring: Introduction

- Problem: “baked-in” vector space model
- How to integrate additional algorithms?
  - **Example: Language Models**
  - **Before 4.0: write custom Queries**
  - **Before 4.0: track certain statistics yourself**
- How to customize for your data?
  - **Without digging into the guts of postings lists!**
- Idea: separate “matching” from “scoring”

Scoring: additional algorithms

- BM25
- Language Models
- Divergence from Randomness
- Information-based Models

Scoring: Configuration

Lucene: use *Similarity* class

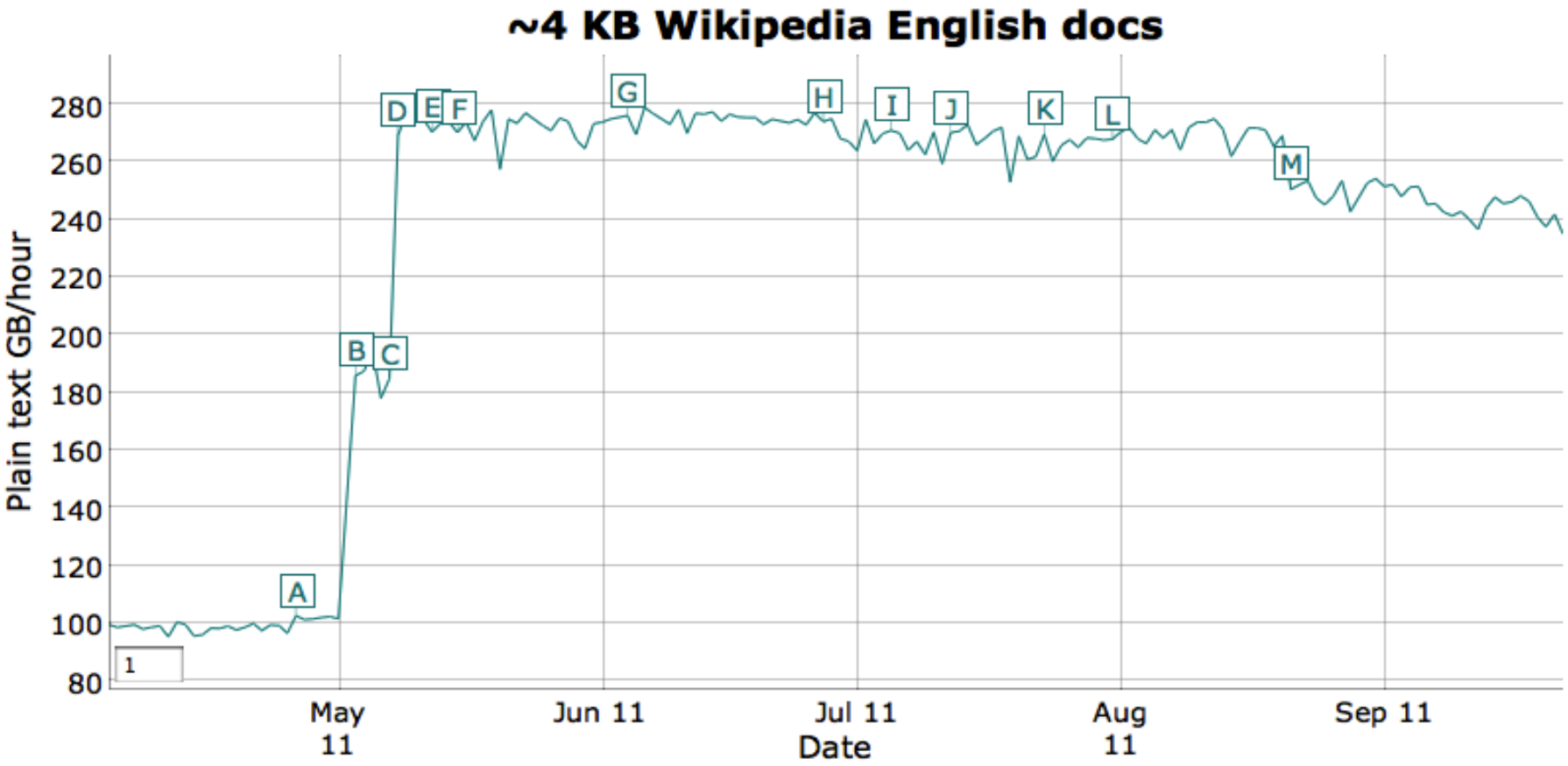
Solr: specify in schema.xml

```
<similarity class="solr.LMDirichletSimilarityFactory">
  <float name="mu">1000</float>
</similarity>
```

Performance Improvements

- Concurrent Flushing
- Fast Fuzzy Query
- Improved RAM efficiency

Concurrent Flushing



http://people.apache.org/~mikemccand/lucenebench/

Future Improvements

- Block Index Compression
  - **PFOR-delta, Simple8b, …**
- Positional iterators from Scorers
  - **Offsets in postings lists (fast highlighting)**
  - **Proximity Scoring**
- Structured/Section Scoring (e.g. BM25F)
- Improved flexibility through Codec
  - **stored fields, term vectors, ...**

Conclusion

- Lucene 4.0 will have many improvements, architectural, and API changes.
- A few of these were introduced here:
  - **Ability to customize the index format**
  - **Additional scoring algorithms**
  - **Performance Improvements**
- More are currently under development.
- For more information, look at CHANGES.txt in subversion, come to Barcelona, ...

More Information

- Lucene Website
  - **http://lucene.apache.org**
- Users List
  - **java-user@lucene.apache.org**
- Lucene Revolution presentations
  - **http://www.lucidimagination.com/devzone/events/conferences/revolution/2010**
  - **http://www.lucidimagination.com/devzone/events/conferences/revolution/2011**
- Contact Info
  - **robert.muir@lucidimagination.com**
  - **http://www.lucidimagination.com/blog/author/robert-muir**