HathiTrust Large Scale Search

www.hathitrust.org

Tom Burton-West
Information Retrieval Programmer
Digital Library Production Service
University of Michigan
www.hathitrust.org/blogs

October 7th 2010
HathiTrust

- HathiTrust is a shared digital repository
- 30+ member libraries
- Large Scale Search is one of many services built on top of the repository
- Currently about 6.5 million books
- 250 Terabytes
  - Preservation page images; jpeg 2000, tiff (244TB)
  - OCR and Metadata about (6TB)
Challenges

• Goal: Design a system for full-text search that will scale to 7 million -20 million volumes (at a reasonable cost.)

• Challenges:
  – Must scale to 20 million full-text volumes
  – Very long documents compared to most large-scale search applications
  – Multilingual collection (400+ languages)
  – OCR quality varies
Long Documents

- Average HathiTrust document is 700KB containing over 100,000 words.
  - Estimated size of 7 million Document collection is 4.5TB.
- Average HathiTrust document is about 38 times larger than the average document size of 18KB used in Large Research test collections.

<table>
<thead>
<tr>
<th>Collection</th>
<th>Size</th>
<th>Documents</th>
<th>Average Doc size</th>
</tr>
</thead>
<tbody>
<tr>
<td>HathiTrust</td>
<td>4.5 TB (projected)</td>
<td>7 million</td>
<td>700 KB</td>
</tr>
<tr>
<td>TREC GOV2</td>
<td>0.456 TB</td>
<td>25 million</td>
<td>18 KB</td>
</tr>
<tr>
<td>SPIRIT</td>
<td>1 TB</td>
<td>94 million</td>
<td>10 KB</td>
</tr>
<tr>
<td>NW1000G-04</td>
<td>1.3 TB*</td>
<td>100 million</td>
<td>16 KB</td>
</tr>
</tbody>
</table>
Index Size, Caching, and Memory

• Our 6 million document index is between 3 and 4 terabytes.
  – About 450 GB per million documents

• Large index means disk I/O is bottleneck

• Tradeoff JVM vs OS memory
  – Solr uses OS memory (disk I/O caching) for caching of postings
  – Memory available for disk I/O caching has most impact on response time (assuming adequate cache warming)
Response Time Varies with Query

Response time 1 Million Volumes
2 Shards 2 Machines 16GB Mem each
(log scale)

Average: 673
Median: 91
90th: 328
99th: 7,504
Slowest 5% of queries

Response Time 95th percentile (seconds)

Query number

Response Time (seconds)

0
1
10
100
1,000
940 950 960 970 980 990 1,000
# Standard index vs. CommonGrams

## Standard Index

<table>
<thead>
<tr>
<th>WORD</th>
<th>TOTAL OCCURRENCES IN CORPUS (MILLIONS)</th>
<th>NUMBER OF DOCS (THOUSANDS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>2,013</td>
<td>386</td>
</tr>
<tr>
<td>of</td>
<td>1,299</td>
<td>440</td>
</tr>
<tr>
<td>and</td>
<td>855</td>
<td>376</td>
</tr>
<tr>
<td>literature</td>
<td>4</td>
<td>210</td>
</tr>
<tr>
<td>lives</td>
<td>2</td>
<td>194</td>
</tr>
<tr>
<td>generation</td>
<td>2</td>
<td>199</td>
</tr>
<tr>
<td>beat</td>
<td>0.6</td>
<td>130</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>4,176</strong></td>
<td></td>
</tr>
</tbody>
</table>

## Common Grams

<table>
<thead>
<tr>
<th>TERM</th>
<th>TOTAL OCCURRENCES IN CORPUS (MILLIONS)</th>
<th>NUMBER OF DOCS (THOUSANDS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>of-the</td>
<td>446</td>
<td>396</td>
</tr>
<tr>
<td>generation</td>
<td>2.42</td>
<td>262</td>
</tr>
<tr>
<td>the-lives</td>
<td>0.36</td>
<td>128</td>
</tr>
<tr>
<td>literature-of</td>
<td>0.35</td>
<td>103</td>
</tr>
<tr>
<td>lives-and</td>
<td>0.25</td>
<td>115</td>
</tr>
<tr>
<td>and-literate</td>
<td>0.24</td>
<td>77</td>
</tr>
<tr>
<td>the-beat</td>
<td>0.06</td>
<td>26</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>450</strong></td>
<td></td>
</tr>
</tbody>
</table>
CommonGrams
Comparison of Response time (ms)

<table>
<thead>
<tr>
<th></th>
<th>average</th>
<th>median</th>
<th>90th</th>
<th>99th</th>
<th>slowest query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Index</td>
<td>459</td>
<td>32</td>
<td>146</td>
<td>6,784</td>
<td>120,595</td>
</tr>
<tr>
<td>Common Grams</td>
<td>68</td>
<td>3</td>
<td>71</td>
<td>2,226</td>
<td>7,800</td>
</tr>
</tbody>
</table>
Thank You!

www.hathitrust.org

Tom Burton-West
tburtonw@umich.edu

www.hathitrust.org/blogs/large-scale-search
How SLIP is Distributed Over Network & Hardware

**Key**
- producer script
- server/machine
- solr (servert)
- tomcat (servert container)

**LSS App**
- query goes to 1 solr servlet which then sends to all other solr servlets on each machine

**Production Index**
- shard 1
- shard 2
- shard 3
- shard 4
- shard 5
- shard 6
- shard 7
- shard 8
- shard 9
- shard 10

**Isilon Storage**

**Build Index**
- shard 1
- shard 2
- shard 3
- shard 4
- shard 5
- shard 6
- shard 7
- shard 8
- shard 9
- shard 10

**"Slurm-5"**
- build-1
- build-2
- build-3
- build-4
- build-5
- build-6
- build-7
- build-8
- build-9
- build-10

**Lucene Documents**
- each script picks a different solr instance each time it has a document to index via round robin

**"Bubble"**
- "Chamomile"
- "Kombucha"
- "Hibiscus"
- "Peppermint"