Offline Data Processing: Tasks and Infrastructure Support

T. Yang, UCSB 293S
Table of Content

• Offline incremental data processing: case study
• Example of content analysis
• System support
Offline Architecture for Ask.com Search

[Diagram showing content analysis, duplicate/spam detection, online data generation and update, old partitions, new partitions, crawling & data acquisition, update manager, online content controller, link/anchor manager database, link/anchor text database, search log, log analysis.]
Content Management

- Organize the vast amount of pages crawled to facilitate online search.
  - Data preprocessing
  - Inverted index
  - Compression
  - Classify and partition data
- Collect additional content and ranking signals.
  - Link, anchor text, log data
- Extract and structure content
- Duplicate detection
Classifying and Partitioning data

- **Classify**
  - Content quality. Language/country etc

- **Partition**
  - Based on languages and countries. Geographical distribution based on data center locations
  - Partition based on quality
    - First tier --- high chance that users will access
      - Quality indicator
      - Click feedback
    - Second tier – lower chance
Examples of Context Extraction/Analysis

- Identify key phrases that capture the meaning of this document.
  - For example, title, section title, highlighted words.
    - HTML vs PDF
- Identify parts of a document representing the meaning of this document.
  - Many web pages contain a side-menu, which is less relevant to the main content of the documents
- Capture page content through Javascript analysis.
  - Page rendering and Javascript evaluation within a page
Example of Content Analysis

- Detect content block related to the main content of a page
  - Non-content text/link material is de-prioritized during indexing process
Redundant Content Removal in Search Engines

- Over 1/3 of Web pages crawled are near duplicates
- When to remove near duplicates?
  - Offline removal
  - Online removal with query-based duplicate removal

![Diagram showing the process of redundant content removal in search engines.](image-url)
Why there are so many duplicates?

• Same content, different URLs, often with different session IDs.
• Crawling time difference
## Tradeoff of online vs. offline removal

<table>
<thead>
<tr>
<th>Impact to offline</th>
<th>Online-dominating approach</th>
<th>Offline-dominating approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>High precision</td>
<td>Remove fewer duplicates</td>
<td>Remove most of duplicates</td>
</tr>
<tr>
<td>Low recall</td>
<td></td>
<td>Higher offline burden</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Impact to online</th>
<th>More burden to online deduplication</th>
<th>Less burden to online deduplication</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Impact to overall cost</th>
<th>Higher serving cost</th>
<th>Lower serving cost</th>
</tr>
</thead>
</table>
Software Infrastructure Support at Ask.com

- Programming support (multi-threading/exception Handling, Hadoop MapReduce)
- Data stores for managing billions of objects
  - Distributed hash tables, queues etc
- Communication and data exchange among machines/services
- Execution environment
  - Controllable (stop, pause, restart).
  - Service registration and invocation
  - Service monitoring
  - Logging and test framework.
Requirements for Data Repository Support in Offline Systems

- **Update**
  - handling large volumes of modified documents
  - adding new content

- **Random access**
  - request the content of a document based on its URL

- **Compression and large files**
  - reducing storage requirements and efficient access

- **Scan**
  - Scan documents for text mining.
Options for Key-value Data Stores

- Support: append or put, get operations
- Bigtable at Google
- Dynamo at Amazon
- Open source software

<table>
<thead>
<tr>
<th>Technology</th>
<th>Language Platform</th>
<th>Users/sponsors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache Cassandra</td>
<td>Bigtable, Dynamo</td>
<td>Apache</td>
</tr>
<tr>
<td>Hypertable</td>
<td>Bigtable</td>
<td>Baidu</td>
</tr>
<tr>
<td>Hbase</td>
<td>Bigtable</td>
<td>Apache</td>
</tr>
<tr>
<td>LevelDB</td>
<td>Bigtable</td>
<td>Google</td>
</tr>
<tr>
<td>MongoDB</td>
<td>C++</td>
<td>Google</td>
</tr>
</tbody>
</table>
Sample Requirements for Applications: Data repository for crawling

• Common data operations
  - Update: Mainly append operations every day.
  - Content read:
    – Typically scan and then transfer data to another cluster
    – Sometime: random access individual pages for inspection

```
   (Diagram showing URL scheduling, URL download, and link extraction and filtering)
```

```
   (Diagram showing data flow and storage)
```
Sample Requirements for periodic data reclassification

- Data repository hosting a large page collection with periodical page re-classification
  - Update: Append only operations for raw data
    - Update → meta data modification periodically for selected pages (random access).
  - Read: Scan only operations for raw data processing.
    - Random read sometime for a small number of pages.

Data repository

MapReduce for classification