Querying
Introduction to Information Retrieval
INF 141
Donald J. Patterson

Content adapted from Hinrich Schütze
http://www.informationretrieval.org
Overview

- Boolean Retrieval
- Weighted Boolean Retrieval
- Zone Indices
- Term Frequency Metrics
- The full vector space model
Querying

From the bottom
From the bottom

- “Grep”
- Querying without an index or a crawl
- Whenever you want to find something you look through the entire document for it.

- Example:
  - You have the collected works of Shakespeare on disk
  - You want to know which play contains the words
  - “Brutus AND Caesar”
Querying
Querying

• “Grep”
  • “Brutus AND Caesar” is the query.
  • This is a boolean query. Why?
  • What other operators could be used?
• The grep solution:
  • Read all the files and all the text and output the intersection of the files
Querying

- “Grep”
  - Slow for large corpora
  - Calculating “NOT” requires exhaustive scanning
  - Some operations not feasible
    - Query: “Romans NEAR Countrymen”
  - Doesn’t support ranked retrieval
- Moving beyond grep is the motivation for the inverted index.
Our **inverted index** is a 2-D array or Matrix.

A Column For Each Document

<table>
<thead>
<tr>
<th></th>
<th>Anthony and Cleopatra</th>
<th>Julius Caesar</th>
<th>The Tempest</th>
<th>Hamlet</th>
<th>Othello</th>
<th>Macbeth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Brutus</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Caesar</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Calpurnia</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cleopatra</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mercy</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>worser</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• **Boolean Query**
  - Queries are boolean expressions
  - Search returns all documents which satisfy the expression
  - Does Google use the Boolean model?
- **Boolean Query**
  - Straightforward application of inverted index
  - where cells of inverted index are (0,1)
  - indicating presence or absence of a term

Document

<table>
<thead>
<tr>
<th>Term</th>
<th>Anthony and Cleopatra</th>
<th>Julius Caesar</th>
<th>The Tempest</th>
<th>Hamlet</th>
<th>Othello</th>
<th>Macbeth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Brutus</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Caesar</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Calpurnia</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cleopatra</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mercy</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>worser</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
**Boolean Query**

- 0/1 vector for each term
- "Brutus AND Caesar AND NOT Calpurnia =
- Perform bitwise Boolean operation on each row:
  - $110100 \text{ AND } 110111 \text{ AND } !(010000) = 100100$

<table>
<thead>
<tr>
<th>Term</th>
<th>Anthony and Cleopatra</th>
<th>Julius Caesar</th>
<th>The Tempest</th>
<th>Hamlet</th>
<th>Othello</th>
<th>Macbeth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anthony</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Brutus</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Caesar</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Calpurnia</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Cleopatra</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>mercy</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>worser</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Querying

- **Boolean Query**
- A big corpus means a sparse matrix
- A sparse matrix motivates the introduction of the posting
  - Much less space to store
  - Only recording the “1” positions
Querying

- **Boolean Query**
  - Query processing on postings
  - Brutus AND Caesar
    - Locate the postings for Brutus
    - Locate the postings for Caesar
    - Merge the postings

<table>
<thead>
<tr>
<th>Brutus</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caesar</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>13</td>
</tr>
</tbody>
</table>
• **Boolean Query**
  - Merging -> walk through the two posting simultaneously
  - postings sorted by doc ID
• **Boolean Query**

• An algorithm based on postings

• Linear in the size of the postings

\[
\text{INTERSECT}(p_1, p_2)
\]

1. \(answer \leftarrow <>\)

2. While \(p_1 \neq \text{nil} \) and \(p_2 \neq \text{nil}\)

3. Do if \(\text{docID}(p_1) = \text{docID}(p_2)\)

4. Then \(\text{ADD}(answer, \text{docID}(p_1))\)

5. \(p_1 \leftarrow \text{next}(p_1)\)

6. \(p_2 \leftarrow \text{next}(p_2)\)

7. Else if \(\text{docID}(p_1) < \text{docID}(p_2)\)

8. Then \(p_1 \leftarrow \text{next}(p_1)\)

9. Else \(p_2 \leftarrow \text{next}(p_2)\)

10. Return \(answer\)
**Querying**

- **Boolean Query**
  - Is the algorithmic complexity better than scanning?
  - Where would you put more complex formulae?

```plaintext
\text{INTERSECT}(p_1, p_2)

1 \quad answer \leftarrow \langle \rangle
2 \quad \text{while } p_1 \neq \text{nil} \text{ and } p_2 \neq \text{nil}
3 \quad \text{do if } \text{docID}(p_1) = \text{docID}(p_2)
4 \quad \quad \text{then ADD}(answer, \text{docID}(p_1))
5 \quad \quad p_1 \leftarrow \text{next}(p_1)
6 \quad \quad p_2 \leftarrow \text{next}(p_2)
7 \quad \text{else if } \text{docID}(p_1) < \text{docID}(p_2)
8 \quad \quad \text{then } p_1 \leftarrow \text{next}(p_1)
9 \quad \quad \text{else } p_2 \leftarrow \text{next}(p_2)
10 \quad \text{return } answer
```
Boolean Queries

- Exact match
- Views each document as a "bag of words"
- Precise: a document matches or it doesn’t
- Primary commercial retrieval tool for 3 decades
- Professional searchers (e.g., lawyers) still like Boolean queries
- No question about what you are getting
Building up our query technology

- Linear on-demand retrieval (aka grep)
- 0/1 Vector-Based Boolean Queries
- Posting-Based Boolean Queries
Querying

Building up our query technology

- Linear on-demand retrieval (aka grep)
- 0/1 Vector-Based Boolean Queries
- Posting-Based Boolean Queries

- How would it apply to
  - http://www.rhymezone.com/shakespeare/
Booolean Model vs. Ranked Retrieval Methods

* Only game for 30 years
* uses precise queries
* user decides relevance
* stayed current with proximity queries
* precise controlled queries
* transparent queries
* controlled queries

* Appeared with www
* uses “free-text” queries
* system decides relevance
* works with enormous corpora
* “no guarantees” in queries
Querying - Boolean Search Example

- **Westlaw**
  - Largest commercial (paying subscribers) legal search service (started in 1975, ranking added in 1992)
  - Tens of terabytes of data
  - 700,000 users
  - Majority of users still use boolean queries (default in 2005)
    - Example:
      - What is the status of limitations in cases involving federal tort claims act?
      - **LIMIT! /3 STATUTE ACTION /S FEDERAL /2 TORT /3 CLAIM**
      - /3 = within 3 words. /S same sentence
Westlaw

Example:

- Requirements for disabled people to be able to access a workplace
- disabl! /p access! /s work-site work-place employment /3 place
- space is a disjunction not a conjunction
- long precise queries, proximity operators, incrementally developed, not like web search
- preferred by professionals, but not necessarily better
Building up our query technology

- "Matching" search
- Linear on-demand retrieval (aka grep)
- 0/1 Vector-Based Boolean Queries
- Posting-Based Boolean Queries
- Ranked search
- Parametric Search
Ranked Search

• Rather than saying
  • (query, document) matches or not (0,1)
  • ("Capulet","Romeo and Juliet") = 1
• Now we are going to assign rankings
  • (query, document) in {0,1}
  • ("capulet","Romeo and Juliet") = 0.7
Querying

- **Metadata** = structured additional information about a document.

- **Examples:**
  - The author of a document
  - The creation date of a document
  - The title of a document
  - The location where a document was created

- author, creation date, title, location are *fields*

- searching for “William Shakespeare” in a doc differs from

- searching for “William Shakespeare” in the author of a doc
Querying

- **Parametric Search**
  - supports searching on meta-data explicitly
  - a parametric search interface allows a mix of full-text query and meta-data queries
- Example:
  - www.carfinder.com
Querying

- **Parametric Search**
  - **Example:**
    - Result is a large table
    - Columns are fields
    - Searching for “2005” only applied to year field
Querying

- **Parametric Search**
  - **Example:**
    - Result is a large table
    - Columns are fields
    - Searching for “2005” only applied to year field

<table>
<thead>
<tr>
<th>Save</th>
<th>Year</th>
<th>Make/Model</th>
<th>Miles</th>
<th>Price</th>
<th>Photos</th>
<th>Body Style</th>
<th>Color</th>
<th>Distance</th>
<th>Dealer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Berlinetta</td>
<td>1,030</td>
<td>$249,900</td>
<td>2 Door Coupe</td>
<td>CORSO RED</td>
<td>28 Miles</td>
<td>FleetRatescomNewUsed</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 575 Superamerica Co</td>
<td>4,200</td>
<td>$285,000</td>
<td>Convertible</td>
<td>Silver</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>3,500</td>
<td>$249,500</td>
<td>Convertible</td>
<td>Rosso Corsa</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>2,900</td>
<td>$249,000</td>
<td>Convertible</td>
<td>YELLOW</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Coupe</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 360 Spider F1 Converti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2005</td>
<td>Ferrari 430 F1 Coupe</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Parametric Search

Example:

- Result is a large table
- Columns are fields
- Searching for "2005" only applied to year field

<table>
<thead>
<tr>
<th>Save</th>
<th>Year</th>
<th>Make/Model</th>
<th>Miles</th>
<th>Price</th>
<th>Photos</th>
<th>Body Style</th>
<th>Color</th>
<th>Distance</th>
<th>Dealer</th>
</tr>
</thead>
<tbody>
<tr>
<td>2005</td>
<td>Ferrari 430 Berlinetta</td>
<td>1,030</td>
<td>$249,900</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>2 Door Coupe</td>
<td>CORSO RED</td>
<td>28 Miles</td>
<td>FleetRatescomNewUsed</td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 575 Superamerica Co</td>
<td>4,200</td>
<td>$285,000</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>Silver</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>3,500</td>
<td>$249,500</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>Rosso Corsa</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>2,900</td>
<td>$249,000</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>YELLOW</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>3,945</td>
<td>$239,500</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>BLACK</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Coupe</td>
<td>1,500</td>
<td>$219,500</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>2 Door Coupe</td>
<td>Grigio Alloy</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>4,500</td>
<td>$219,000</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>RED</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 360 Spider F1 Conve</td>
<td>4,000</td>
<td>$219,000</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>Black</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>10,317</td>
<td>$209,999</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>Red</td>
<td>28 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 Spider Converti</td>
<td>29,000</td>
<td>$205,000</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>Convertible</td>
<td>RED</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>Ferrari 430 F1 Coupe</td>
<td>5,300</td>
<td>$199,000</td>
<td><img src="image.jpg" alt="Photo" /></td>
<td>2 Door Coupe</td>
<td>BLACK</td>
<td>65 Miles</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Querying

- Parametric Search

- Example:
  - http://www.ocregister.com/realestate/
Querying

- Parametric Search

- Example:
  - http://www.ocregister.com/realestate/
Querying

- Parametric Search

- Example:
  - http://www.ocregister.com/realestate/
  - 92614: 77 results
Parametric Search

Example:

- http://www.ocregister.com/realestate/
- 92614: 77 results
Querying

- **Parametric Search**

- **Example:**
  - http://www.ocregister.com/realestate/
  - 92614: 77 results