CS222: Principles of Data Management

Lecture #15: Open Topics and Wrap up

Instructor: Chen Li
Outline

- Other indexing techniques: spatial, text
- DMBS architecture revisited
- Other DBMS capabilities
- What’s next?
Spatial Data

- Geographic Information Systems (GIS)
  - E.g., ESRI’s ArcInfo; OpenGIS Consortium
  - Geospatial information
  - All classes of spatial queries and data are common
- Computer-Aided Design/Manufacturing
  - Store spatial objects such as surface of airplane fuselage or the physical layout of an integrated circuit
  - Range queries and spatial join queries are common
- Multimedia Databases
  - Images, video, text, etc. stored and retrieved by content
  - First converted to feature vector form; high dimensionality
  - Nearest-neighbor queries are the most common
Types of Spatial Queries

- **Spatial Range Queries**
  - *Find all cities within 50 miles of Madison*
  - Query has associated region (location, boundary)
  - Answer includes overlapping or contained data regions

- **Nearest-Neighbor Queries**
  - *Find the 10 cities nearest to Madison*
  - Results must be ordered by proximity

- **Spatial Join Queries**
  - *Find all cities near a lake*
  - Expensive, join condition involves regions and proximity
Spatial indexing: R-Tree

- The R-tree is a tree-structured index that remains balanced on inserts and deletes.
- Each key stored in a leaf entry is intuitively a box (*bounding box*) or collection of *intervals*, with one interval per dimension.
- Example in 2-D:
Example of an R-Tree

Spatial object approximated by bounding box R8
Example R-Tree (Contd.)
Search for Objects Overlapping Box Q

Start at root.

1. If current node is non-leaf, for each entry \(<E, \text{ptr}>\), if box \(E\) overlaps \(Q\), search subtree identified by \(\text{ptr}\).

2. If current node is leaf, for each entry \(<E, \text{rid}>\), if \(E\) overlaps \(Q\), \(\text{rid}\) identifies an object that might overlap \(Q\). (Why?)

Note: May have to search several subtrees at each node! (In contrast, a B-tree equality search goes to just one leaf.)
R-Tree Search Examples (Cont.)
R-Tree Search Examples (Contd.)
Which plays of Shakespeare contain the words *Brutus AND Caesar* but *NOT Calpurnia*?

One could *grep* all of Shakespeare’s plays for *Brutus* and *Caesar*, then strip out lines containing *Calpurnia*?

- Slow (for large corpora)
- *NOT Calpurnia* is non-trivial
- Other operations (e.g., find the word *Romans* near *countrymen*) not feasible
Text indexing: Inverted index

- For each term \( T \), we must store a list of all documents that contain \( T \).
- Do we use an array or a list for this?

<table>
<thead>
<tr>
<th>Brutus</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calpurnia</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>21</td>
</tr>
<tr>
<td>Caesar</td>
<td>13</td>
<td>16</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Inverted index construction

Documents to be indexed.

Token stream.

More on these later.

Modified tokens.

Linguistic modules

Friends, Romans, countrymen.

Inverted index

Tokenizer

Indexer

friend

roman

countryman

2

4

1

2

13

16
Consider processing the query: 

*Brutus AND Caesar*

- Locate *Brutus* in the Dictionary;
  - Retrieve its postings.
- Locate *Caesar* in the Dictionary;
  - Retrieve its postings.
- “Merge” the two postings:
DBMS architecture revisited

SQL

Query Parser
Query Optimizer

Plan Executor
Relational Operators (+ Utilities)

Files of Records
Access Methods (Indices)
Buffer Manager

Disk Space and I/O Manager

Transaction Manager
Lock Manager
Log Manager

WAL

Data Files
Index Files
Catalog Files

API calls

Query plans

(CS 223)
Other Database Capabilities

- Other data types: multimedia, BLOB
- More operators: e.g., outer joins
- Other data models: XML, JSON, …
- User management, access control
- Views
- Stored procedures
- User defined functions
- Transaction processing
- Parallel DBMS
“Big Data” trend

- MapReduce, Hadoop
- NoSQL
- Column stores
- Document stores
- …
No Shortage of “NoSQL”
Big Data Analysis Platforms...
AsterixDB System
(asterix.ics.uci.edu)
So Where To From Here?

- CS 223: Transaction Processing and Distributed Data Management
- CS 224: Advances in Database Management System Technology
- More hands-on experiences....
SO MUCH TO DO
SO LITTLE TIME
GOOD LUCK ON THE FINAL
AND MAY THE ODDS BE EVER IN YOUR FAVOR