A Quick Introduction to Approximate Query Processing

Part-IV

CS286, Spring'2007

Minos Garofalakis
Logistics...

• Draft CS286 web site is finally up!
  - http://db.cs.berkeley.edu/cs286sp07/

• Project list and guidelines being worked on
  - Please email me & Raghu to discuss your own project ideas...
Approximate Query Processing using Data Synopses

- Decision Support Systems (DSS)
  - GB/TB
- Compact Data Synopses
  - KB/MB

SQL Query → Exact Answer → Long Response Times!

"Transformed" Query → Approximate Answer → FAST!!

- How to construct effective data synopses??
### Relations as Frequency Distributions

#### One-dimensional distribution

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Salary</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>MG</td>
<td>34</td>
<td>100K</td>
<td>25K</td>
</tr>
<tr>
<td>JG</td>
<td>33</td>
<td>90K</td>
<td>30K</td>
</tr>
<tr>
<td>RR</td>
<td>40</td>
<td>190K</td>
<td>55K</td>
</tr>
<tr>
<td>JH</td>
<td>36</td>
<td>110K</td>
<td>45K</td>
</tr>
<tr>
<td>MF</td>
<td>39</td>
<td>150K</td>
<td>50K</td>
</tr>
<tr>
<td>DD</td>
<td>45</td>
<td>150K</td>
<td>50K</td>
</tr>
<tr>
<td>JN</td>
<td>43</td>
<td>140K</td>
<td>45K</td>
</tr>
<tr>
<td>AP</td>
<td>32</td>
<td>70K</td>
<td>20K</td>
</tr>
<tr>
<td>EM</td>
<td>24</td>
<td>50K</td>
<td>18K</td>
</tr>
<tr>
<td>DW</td>
<td>24</td>
<td>50K</td>
<td>28K</td>
</tr>
</tbody>
</table>

#### Three-dimensional distribution

<table>
<thead>
<tr>
<th>Age</th>
<th>Salary</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>30</td>
<td>20</td>
<td>50</td>
</tr>
<tr>
<td>34</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>40</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>43</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>45</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>50</td>
<td>20</td>
<td>50</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>
Outline

• Intro & Approximate Query Answering Overview
  - Synopses, System architectures, Commercial offerings
• One-Dimensional Synopses
  - Histograms: Equi-depth, Compressed, V-optimal, Incremental maintenance, Self-tuning
  - Samples: Basics, Sampling from DBs, Reservoir Sampling
  - Wavelets: 1-D Haar-wavelet histogram construction & maintenance
• Multi-Dimensional Synopses and Joins
• Set-Valued Queries
• Discussion & Comparisons
• Advanced Techniques & Future Directions
Outline

• Intro & Approximate Query Answering Overview
  - Synopses, System architecture, Commercial offerings
• One-Dimensional Synopses
  - Histograms, Samples, Wavelets
• Multi-Dimensional Synopses and Joins
  - Multi-D Histograms, Join synopses, Wavelets
• Set-Valued Queries
  - Error metrics; Using Histograms, Samples, Wavelets
• Discussion & Comparisons
• Advanced Techniques & Future Directions
  - Dependency-based, Streaming data
Two-dimensional Haar Wavelets -- Non-standard decomposition

Wavelet Transform Array:

Averaging & Differencing

(a+b-c-d)/4  (a-b-c+d)/4

(a+b+c+d)/4  (a-b+c-d)/4

"Supports"

RECURSE
Multi-dimensional Haar Wavelets

- Haar decomposition in d dimensions = d-dimensional array of wavelet coefficients
  - Coefficient support region = d-dimensional rectangle of cells in the original data array
  - Sign of coefficient’s contribution can vary along the quadrants of its support

Support regions & signs for the 16 nonstandard 2-dimensional Haar coefficients of a 4x4 data array A
Range-sum Estimation Using Wavelet Synopses

- Coefficient thresholding
  - As in 1-d case, normalizing by appropriate constants and retaining the largest coefficients minimizes the overall L2 error

- Range-sums: selectivity estimation or OLAP-cube aggregates [VW99] ("measure attribute" as count)

- Only coefficients with support regions intersecting the query hyper-rectangle can contribute
  - Many contributions can cancel each other [CGR00, VW99]
Approximate Query Processing Using Wavelets [CGR00]

- Reduce relations into compact *wavelet-coefficient synopses*

*Entire query processing in the compressed (wavelet) domain*
Wavelet Query Processing

- Each operator (e.g., select, project, join, aggregates, etc.)
  - *input*: set of wavelet coefficients
  - *output*: set of wavelet coefficients

- Finally, rendering step
  - *input*: set of wavelet coefficients
  - *output*: (multi) set of tuples
Selection -- Relational Domain

Joint Data Distribution Array

<table>
<thead>
<tr>
<th>Dim. D1</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Relation

<table>
<thead>
<tr>
<th>Dim D1 (Attr1)</th>
<th>Dim D2 (Attr2)</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

- In relational domain, interested in only those cells inside query range
- In wavelet domain, interested in only the coefficients that contribute to those cells
Selection -- Wavelet Domain

Query Range

D1

D2

D1

D2
**Equi-join -- Relational Domain**

- **Relational domain:** Join count = 7*3 = (A1-A3)*(B2+B3)
- **Wavelet domain:** A1*B2 + A1*B3 - A3*B2 - A3*B3
- Consider all pairs of coefficients: (1) check joinability (overlap in join dimension(s)), (2) compute output coefficients
Equi-join -- Wavelet Domain

Join output coefficient:

\[ v = v_1 \ast v_2 \]
Wavelet Query Processing

- Each operator (e.g., select, project, join, aggregates, etc.)
  - \textit{input:} set of wavelet coefficients
  - \textit{output:} set of wavelet coefficients

- Finally, rendering step
  - \textit{input:} set of wavelet coefficients
  - \textit{output:} (multi)set of tuples
Outline

• Intro & Approximate Query Answering Overview
• One-Dimensional Synopses
• Multi-Dimensional Synopses and Joins
• Set-Valued Queries
• Discussion & Comparisons
• Advanced Techniques & Future Directions
• Conclusions
**Discussion & Comparisons (1)**

- **Histograms & Wavelets**: Limited by “curse of dimensionality”
  - Rely on data space partitioning in “regions”
  - Ineffective above 5-6 dimensions
    - Value/frequency uniformity assumptions within buckets break down in medium-to-high dimensionalities!!

- **Sampling**: No such limitations, BUT...
  - Ineffective for ad-hoc relational joins over arbitrary schemas
    - Uniformity property is lost
    - Quality guarantees degrade
  - Effectiveness for *set-valued* approximate queries is unclear
    - Only (very) small subsets of the answer set are returned (especially, when joins are present)
Discussion & Comparisons (2)

- **Histograms & Wavelets**: Compress data by accurately capturing rectangular “regions” in the data space
  - Advantage over sampling for typical, “range-based” relational DB queries
  - BUT, unclear how to effectively handle unordered/non-numeric data sets (no such issues with sampling...)

- **Sampling**: Provides strong probabilistic quality guarantees (unbiased answers) for individual aggregate queries
  - **Histograms & Wavelets**: Can guarantee a bound on the overall error (e.g., L2) for the approximation, BUT answers to individual queries can be heavily biased!!

**No clear winner exists!! (Hybrids??)**
Outline

• Intro & Approximate Query Answering Overview
• One-Dimensional Synopses
• Multi-Dimensional Synopses and Joins
• Set-Valued Queries
• Discussion & Comparisons
• Advanced Techniques & Future Directions
  - Dependency-based Synopses
  - Streaming Data
  - XML Synopses
• Conclusions
Dependency-based Histogram Synopses [DGR01]

- Extremes in terms of the underlying correlations!!
- **Dependency-Based Histograms**: explore space between extremes by explicitly identifying data correlations/independences
  - Build a *statistical interaction model* on data attributes
  - Based on the model, build a collection of low-dimensional histograms
  - Use this histogram collection to provide approximate answers

- **General methodology**, also applicable to other synopsis techniques (e.g., wavelets)
Dependency-based Histograms

- Identify (and exploit) attribute correlation and independence
  - Partial Independence:
    \[
    p(\text{salary, height, weight}) = p(\text{salary}) \times p(\text{height, weight})
    \]
  - Conditional Independence:
    \[
    p(\text{salary, age | YPE}) = p(\text{salary | YPE}) \times p(\text{age | YPE})
    \]
- Use forward selection to build a decomposable statistical model [BFH75], [Lau96] on the attributes
  - A,D are conditionally independent given B,C
    - \[
    p(\text{AD | BC}) = p(\text{A | BC}) \times p(\text{D | BC})
    \]
  - Joint distribution
    - \[
    p(\text{ABCD}) = p(\text{ABC}) \times p(\text{BCD}) / p(\text{BC})
    \]
  - Build histograms on model cliques

- Significant accuracy improvements (factor of 5) over pure MHIST
- New histogram construction & usage algorithms, etc.
Workload-tuned Biased Sampling -- Congressional Samples [AGPO00]

- Decision support queries routinely segment data into groups & then aggregate the information within each group
  - Each table has a set of “grouping columns”: queries can group by any subset of these columns

- **Goal**: Maximize the accuracy for all groups (large or small) in each Group-by query
  - E.g., census DB with state \( s \), gender \( g \), and income \( i \)
  - Q: Avg\( (i) \) group-by \( s \) : seek good accuracy for all 50 states
  - Q: Avg\( (i) \) group-by \( s,g \) : seek good accuracy for all 100 groups

- **Technique**: Congressional Samples
  - **House**: Uniform sample: good for when no group-by
  - **Senate**: Same size sample per group when use all grouping columns: good for queries with all columns
  - **Congress**: Combines House & Senate, but considers all subsets of grouping columns, and then scales down
Workload-tuned Biased Sampling -- ICICLES [GLR00]

- Biased sampling scheme that *dynamically adapts* to query workload
  - Exploit data locality -- more focus (i.e., #sample points) in frequently-queried regions
  - Let $Q = \{q_1, q_2, \ldots\}$ be a query workload, $R(q_i) = \text{subset of } R \text{ used in answering query } q_i$
  - $L(R, Q) = \text{Extension of } R \text{ wrt } Q = R \bigcup_{q_i \in Q} R(q_i)$ (multiset of tuples)

**Icicle:** Uniform random sample of $L(R, Q)$

- Incrementally maintained and adapt ("self-tune") to workload through *Reservoir Sampling* technique [Vit85]
- *Unbiased Icicle estimators:* New formulas to account for duplicates and bias in sample selection
- *Provably better* (smaller variance) than uniform for "focused" queries (that follow the workload model)
Workload-tuned Biased Sampling -- Lifted Workloads [CDN01]

- Formulate sample selection as an optimization problem
  - Minimize query-answering error for a given workload model
- Technique for "lifting a fixed workload W" to produce a probability distribution over all possible queries
  - Similar to kernel density estimation (queries in W = "sample points")

\[ W = \{ q_1, q_2 \} \]

\[ \text{prob}(q\mid W) = \text{parametric function of } q\text{'s overlap with queries in } W \]

"Fundamental regions" induced by W

CS286, Spring’07 – Minos Garofalakis  #25
Workload-tuned Biased Sampling --
Lifted Workloads

- **Problem:** Find sample of size $k$ that minimizes expected error for a given “lifted” workload

- **Solution:** *Stratified sampling* [Coc77]
  - Collection of uniform samples (of total size $k$) over disjoint subsets (“strata”) of the population
  - Much better estimates when variance within strata is small [Coc77]

- **Stratification:** Selecting appropriate partitioning of $R$
  - Using “fundamental regions” as strata is *optimal* for COUNT
  - For SUM, partition “fundamental regions” further to reduce variance of the aggregated attribute (Neymann technique [Coc77])

- **Allocation:** Dividing $k$ among strata
  - Closed form solutions (valid under certain simplifying assumptions)
Data Streams

• Data is continually arriving. Collect & maintain synopses on the data. Goal: Highly-accurate approximate answers
  - State-of-the-art: Good techniques for narrow classes of queries
  - E.g., Any one-pass algorithm for collecting & maintaining a synopsis can be used effectively for data streams

• Alternative scenario: A collection of data sets. Compute a compact sketch of each data set & then answer queries (approximately) comparing the data sets
  - E.g., detecting near-duplicates in a collection of web pages: Altavista
  - E.g., estimating join sizes among a collection of tables [AGM99]
Looking Forward...

- Optimizing queries for approximation
  - e.g., minimize length of confidence interval at the plan root

- Exploiting mining-based techniques (e.g., decision trees) for data reduction and approximate query processing
  - see, e.g., [BGR01], [GTK01], [JMN99]

- Dynamic maintenance of complex (e.g., dependency-based [DGR01] or mining-based [BGR01]) synopses

- Synopses and approximate query processing for richer data models and data streams
  - e.g., XPath/XQuery over XML databases
XML Data (Text)

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<booklist>
  <book genre="Science" format="Hardcover">
    <author>
      <firstname>Richard</firstname>
      <lastname>Feynman</lastname>
    </author>
    <title>The character of Physical Law</title>
  </book>
  <book genre="Fiction">
    <author>
      <firstname>R.K.</firstname>
      <lastname>Narayan</lastname>
    </author>
    <title>Waiting for the Mahatma</title>
    <published>1981</published>
  </book>
</booklist>
```
XML Data (Tree)

booklist

book

a

t

@f

@g

“Richard” “Feynman”

“The character of physical Law”

“Science” “Hardcover”

book

@g

atp

“…” “…” “…”

fl

“…” “…”
XML Basics

• Elements
  - Encode “concepts” in the XML database
  - Nesting denotes association/inclusion

• Attributes
  - Record information specific to an element (e.g., the genre of a book)

• References
  - Links between elements in different parts of the document
XML vs. Relational Data

<table>
<thead>
<tr>
<th>name</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>3634</td>
</tr>
<tr>
<td>Sue</td>
<td>6343</td>
</tr>
<tr>
<td>Dick</td>
<td>6363</td>
</tr>
</tbody>
</table>

Relation to XML:

- Row: "John" 3634 "Sue" 6343 "Dick" 6363
**XML vs. Relational Data**

- A relation instance is basically a tree with:
  - Unbounded fanout at level 1 (i.e., any # of rows)
  - Fixed fanout at level 2 (i.e., fixed # fields)

- XML data is essentially an arbitrary tree
  - Unbounded fanout at all nodes/levels
  - Any number of levels
  - Variable # of children at different nodes, variable path lengths
XPath Expressions

Examples:

• /booklist/book
• /booklist/book/author
• /booklist/book/author/lastname

Given an XML document, the value of a path expression 
$p$ is a set of elements (= XML subtrees)
Path Expressions

- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]

- Result is a set
**Path Expressions**

- **XPath expressions**
  - Simple: `/A/P/T`
  - Branching: `/A[B]/P/T`
  - Values: `/A/P/T[=v11]`

- Result is a **set**
Path Expressions

- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]

- Result is a set
Path Expressions

- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]

- Result is a set
Path Expressions

- XPath expressions
  - Simple: /A/P/T
  - Branching: /A[B]/P/T
  - Values: /A/P/T[=v11]

- Result is a set
XPath Syntax

- **Path wildcards**
  - `//` = descendant at any level (or self)
  - `*` = any (single) tag
  - Example: `/booklist//lastname`

- **Query attributes and attribute content**
  - Use “@”

- **Branching predicates: A[pred]**
  - Predicate on A’s subtree using *logical connectives* (and, or, etc.), *path expressions*, *built-in functions* (e.g., contains()), etc.
  - Example: `//author[contains(./lastname, “Fey”)]

CS286, Spring’07 – Minos Garofalakis  #40
Synopses for XML

- Summarize labeled tree/graph structure for approximate path navigation queries
  - Selectivity estimation: How many elements satisfy p?
  - Approximate answers: Return an approximate XML document as output of an XQuery fragment

- Key idea: Build a concise Graph Synopsis that captures the path/branching distribution in limited space
  - Use appropriate uniformity/independence assumptions to approximate path structure
  - Refine synopsis in parts of the XML document where assumptions fail
  - *XSketches* [SIGMOD'02,VLDB'02], *TreeSketches* [SIGMOD'04]
Conclusions

- Commercial data warehouses: approaching several 100’s TB and continuously growing
  - Demand for high-speed, interactive analysis (click-stream processing, IP traffic analysis) also increasing

- Approximate Query Processing
  - “Tame” these TeraBytes and satisfy the need for interactive processing and exploration
  - Great promise
  - Commercial acceptance still lagging, but will most probably grow in coming years
  - Still lots of interesting research to be done!!