Graph-Based Synopses for Relational Data

Alkis Polyzotis (UC Santa Cruz)
Data Synopses

Problem: exact answer may be too costly to compute
- Examples: massive data set exploration, selectivity estimation

Solution: run query on a synopsis and return an approximate answer
- Synopsis: lossy summary of data instance
Data Synopses and Query Optimization

- **Optimizer**
  - Plan Enumeration
  - Cost Model

- **Data Synopsis**

- **Physical Plan**

- **Query**

- **Plan A**
  - Z
  - W
  - R
  - T

- **Plan B**
  - Z
  - W
  - R
  - T

- **Data**
  - count(Q)
  - Selectivity
  - Efficient

- **Data Synopsis**
  - count(Q)
  - Selectivity
  - Estimate

- **Expensive**
Table-Level Synopses

- Examples: histograms, wavelets, table samples, sketches
- One synopsis per table
- The synopsis summarizes the frequency matrix
- Problem: ineffective for key/foreign-key joins
Schema-Level Synopses

- Examples: Join Synopses, Prob. Rel. Models
- One synopsis for the whole schema
- Problem: restricted to specific schemata
  - Many-to-many joins cannot be handled
Desiderata

- Schema-level synopsis
- Applicable to general schemata and queries
  - Many-to-many joins
  - Join graphs with cycles
- Affordable to construct
Intuition #1

Relational database $\leftrightarrow$ Semi-structured data graph

**Movie**

<table>
<thead>
<tr>
<th>mid</th>
<th>year</th>
<th>genre</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2005</td>
<td>“action”</td>
</tr>
<tr>
<td>2</td>
<td>2004</td>
<td>“drama”</td>
</tr>
<tr>
<td>3</td>
<td>2000</td>
<td>“drama”</td>
</tr>
</tbody>
</table>

**Actors**

<table>
<thead>
<tr>
<th>aid</th>
<th>sex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>male</td>
</tr>
<tr>
<td>2</td>
<td>female</td>
</tr>
<tr>
<td>3</td>
<td>male</td>
</tr>
</tbody>
</table>

**Cast**

<table>
<thead>
<tr>
<th>aid</th>
<th>mid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Diagram:

- m1
- m2
- m3
- c1
- c2
- c3
- c4
- c5

- Action
- Drama
- Male
- Female

Year:
- 2005
- 2004
- 2000
Intuition #2

- Join query <=> Sub-graph matching
- Selectivity <=> Count of matching sub-graphs

SELECT *
FROM M, C, A
WHERE M.mid=C.mid
AND C.aid=A.aid
AND a.sex=male
AND a.genre=drama
Tuple Graph Synopses (TuGs)

- Graph-based summaries for relational data
  - Key idea: summarize structure of data graph
  - Schema-level synopses
  - Support for a large class of schemata

- Joint work with Josh Spiegel (UCSC)

- Sponsors: NSF (CAREER Award), IBM (Faculty Development Award)
TuGs and XML

Why not use an existing XML technique?
- Relational data graph resembles XML data
- Relational queries resemble twig queries

The summarization problem is inherently different
- Relational data graph vs. XML tree
- Relational queries are fully specified (no // or *)
- Relational queries are undirected

Opportunities for an alternative approach!
Outline

- TuG Synopses
  - Synopsis Model
  - Estimation Framework
- TuG Construction
- Experimental Study
- Conclusions
TuG Synopsis: Joins

- **Node:** Set of tuples from same relation
- **Edge:** Join between tuple-sets
Values are represented as nodes + edges
A node aggregates information about its tuples.

Basic assumptions: independence and uniformity.

Correspondence to clustering:

- Each node has a representative “centroid” of ratios.
- Tight clusters $\iff$ Validity of independence.

Each actor has:
- $-\frac{4}{3}$ joining tuples in $C_1$
- $-\frac{1}{3}$ joining tuples in $C_2$
- $\text{Prob}[\text{sex=male}] = \frac{2}{3}$
- $\text{Prob}[\text{sex=female}] = \frac{1}{3}$
Example TuG Estimation

\[
est = 2 \times \left( \frac{1}{2} \right) \times \left( \frac{3}{2} \right) \times \left( \frac{4}{4} \right) \times \left( \frac{2}{3} \right)\]

SELECT *
FROM M, C, A
WHERE M.mid = C.mid
AND C.aid = A.aid
AND A.sex = male
AND M.genre = action
TuG Estimation Model

Two step process:
1. Identify query embeddings
2. Estimate selectivity of each embedding

Estimates are computed based on ratios

Closed expression for embedding estimates
Methodology extends to queries with cycles

Estimation uses independence => Accuracy depends on validity of independence

Intuition: centroid must be a good representative
Outline

- TuG Synopses
  - Synopsis Model
  - Estimation Framework
- TuG Construction
- Experimental Study
- Conclusions
TuG Construction: Outline

- Problem: Construct an accurate TuG for a specific storage budget
- Outline of construction algorithm:
  - Basic compression operation: node-merge
  - Stage 1: Apply lossless node-merge operations
  - Stage 2: Apply lossy node-merge operations
Node-Merge Operation

Collapse a set of nodes to one new node

- New node acquires aggregated characteristics
- Similar to merging clusters
Lossless Node-Merge

- Lossless merge => estimates remain unchanged
- Observation: A merge is lossless if the merged centroids are equal
- Definition used in XML summarization
- TuGs enable a relaxed condition => Opportunity for higher compression
Nodes $u$ and $v$ are $ab1$-similar $\leftrightarrow$ Equal join ratios to all schema neighbors except one

- Fully similar $\leftrightarrow$ Equal join ratios to all neighbors

Theorem: if $u$ and $v$ are $ab1$-similar then their merge is lossless
**All-but-1 vs Full Similarity**

Number of nodes in different synopses

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Data Graph</th>
<th>Full-Similarity Summary</th>
<th>Ab1-Similarity Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPC-H</td>
<td>8 million</td>
<td>4.4 million</td>
<td>33K</td>
</tr>
<tr>
<td>IMDB</td>
<td>4.7 million</td>
<td>4.5 million</td>
<td>65K</td>
</tr>
</tbody>
</table>
Lossy Merges

Question: when is a lossy merge good?

Intuition: Good merge $\iff$ Similar centroids

Measure quality through error of clustering

Radius, Diameter, Manhattan distance, ...
Construction Algorithm

Stage 1: Apply lossless node-merge ops on data graph to derive a smaller reference summary.

Stage 2: Compress reference summary with lossy node-merge ops.

Stage 3: Compress value distributions.
Construction: Stage 1

- Algorithm sketch:
  - do until no change
    - for each (R:table, N: all-but-one neighbors)
      - apply lossless node-merge

- Order of iteration is based on "clusterability"

- Intuition: select (R,N) with the most lossless node-merge operations
Construction: Stage 2

Algorithm sketch:

\[ r := \text{low} \]
\[ \text{while synopsis size} > \text{budget} \]
\[ \text{select } R \]
\[ \text{apply lossy node-merge on } R \text{ of radius } \leq r \]
\[ \text{if no such } R \text{ exists then increase } r \]

- \( r \): Threshold of quality
- Start with good mergers, deteriorate as needed
- Order of processing based on “clusterability”
- \( R \) has high priority if it can be clustered well
Identifying Merge Operations

- Discover node-mergers through clustering
- Variable r controls the radius of clusters
- Clustering is computed with variant of BIRCH
- Use of randomized sketches to approximate distances
- Typically single-pass processing
- Controllable memory overhead

Diagram:
- Nodes A1(5), A2(3), A3(14), B1(x), C1(y)
- Centroids with ratios to B1 and C1
Construction: Stage 3

Goal: substitute detailed value distributions with compressed value distributions

Key idea: use a single compressed distribution for multiple nodes
Construction Efficiency

- Processing based on disk-based structures
- Scalable clustering algorithm as the core module
- Result: increased efficiency for large data sets
  => affordable construction times
Outline

TuG Synopses
  Synopsis Model
  Estimation Framework

TuG Construction

Experimental Study

Conclusions
Techniques

- Baseline: 1-d histograms and indexes
  - Existing implementation in commercial system X
  - Size of histograms used as storage budget
- Multi-dimensional wavelets [Chakrabarti+00]
- Join Synopses [Acharya+99]
- TuGs
## Data Sets

<table>
<thead>
<tr>
<th></th>
<th>TPC-H</th>
<th>IMDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Relations</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>#Tuples in largest</td>
<td>6 million</td>
<td>2.7 million</td>
</tr>
<tr>
<td>relation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#Tuples in smallest</td>
<td>5</td>
<td>68K</td>
</tr>
<tr>
<td>relation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size of text files</td>
<td>1 GB</td>
<td>139 MB</td>
</tr>
</tbody>
</table>
## Workloads

<table>
<thead>
<tr>
<th></th>
<th>TPC-H</th>
<th>IMDB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. result size of positive queries</td>
<td>600K</td>
<td>50K</td>
</tr>
<tr>
<td>Number of join predicates</td>
<td>4–8</td>
<td>4–6</td>
</tr>
<tr>
<td>Number of selection predicates</td>
<td>1–7</td>
<td>1–5</td>
</tr>
</tbody>
</table>
Evaluation Metric

CFD of Average Relative Error

70% of the queries have ARE less than 20%

More accurate estimation
TuG Accuracy vs. Space

TPC-H
TuG vs. Join Synopses

TPC-H

![Graph showing TuG vs. Join Synopses with TPC-H metric. The graph compares query percentage against average relative error for TuG, Histograms, and Join Synopses.]
TuG vs. Wavelets

IMDB
TuGs vs. Histograms

IMDB

![Graph showing TuGs vs. Histograms for IMDB with Average Relative Error on the x-axis and Query Percentage on the y-axis. The graph compares TuG and Histograms with different error rates.]
Conclusions

Key idea: relational data is semi-structured

TuG Synopses
- Schema-level relational summaries
- Selectivity estimates for complex join queries
- Support for general schemata

Experimental results:
- Accurate selectivity estimates
- Affordable construction
- Benefits over existing techniques
Future Work

- Incremental synopsis maintenance
- Guarantees on estimation accuracy
- Transfer to XML domain
Links

- Google: alakis santa cruz
- DB Research at UCSC: http://db.cs.ucsc.edu