Bi-Level Online Aggregation On Raw Data

Yu Cheng\(^+\), Weijie Zhao\(^*\), Florin Rusu\(^*\)

\(^+\): Amobee. Inc.
\(^*\): University of California, Merced
Outline

- Background
- Problem
- OLA-RAW
- Evaluation
The Palomar Transient Factory (PTF) project aims to identify and automatically classify transient astrophysical objects such as variable stars and supernovae in realtime.
Illustrative Example

Supernova identification

- PTF Files

```
SELECT AGGREGATE(expression) AS agg
FROM candidate
WHERE predicate
HAVING agg < threshold
```
## Existing Solutions

<table>
<thead>
<tr>
<th>Time to query</th>
<th>Execution</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>instant</td>
<td>slow</td>
<td>zero</td>
</tr>
<tr>
<td>fast</td>
<td></td>
<td>full replication</td>
</tr>
<tr>
<td>instant</td>
<td>fast</td>
<td>adaptive</td>
</tr>
</tbody>
</table>

- **PTF Files**
  - External Table
  - SQL*Loader
  - SCANRAW

**UC Merced**
Illustrative Example

- Supernova identification

```
SELECT AGGREGATE(expression) AS agg
FROM candidate
WHERE predicate
HAVING agg > threshold
WITH ACCURACY \( \alpha \)
```
Existing Solutions

<table>
<thead>
<tr>
<th>Time to query</th>
<th>Execution</th>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>instant</td>
<td>slow</td>
<td>zero</td>
</tr>
<tr>
<td>loading</td>
<td>fast</td>
<td>full replication</td>
</tr>
<tr>
<td>instant</td>
<td>fast</td>
<td>adaptive</td>
</tr>
<tr>
<td>loading + shuffling</td>
<td>faster</td>
<td>full replication</td>
</tr>
</tbody>
</table>

PTF Files

External Table

SQL*Loader

SCANRAW

DB Online Aggregation
Research Problem

- Can we find a better solution to execute approximate queries in-situ over raw files?
  - Instant access to data
    - In-situ data processing
  - Generate results faster
    - Online aggregation (OLA)
  - Minimize used storage
    - In-memory synopsis
High Level Approach
Related Work

➢ Adaptive partial loading [Idreos et al., CIDR 2011]
  Only load necessary attributes before query starts
➢ NoDB [Alagianis et al., SIGMOD 2012]
  Instead of loading, build index and cache necessary attributes in memory
➢ Invisible loading [Abouzied et al., EDBT/ICDT 2013]
  Portion of necessary data is loaded into database for every query
➢ Data vaults [Ivanova et al., SSDBM 2012]
  Memory cache for complex data in scientific repositories
➢ SCANRAW [Cheng and Rusu, SIGMOD 2014]
  Load data using spare system resources without affecting query processing
OLÁ-RAW

- OnLine Aggregation for RAW data processing
  - How to generate random samples from raw files?
  - Design a feasible architecture to combine online aggregation with in-situ data processing
  - Find an efficient method to maintain extracted samples
OnLine Aggregation for RAW data processing

- How to generate random samples from raw files?
  Bi-Level Sampling
- Design a feasible architecture to combine online aggregation with in-situ data processing
- Find an efficient method to maintain extracted samples
## Sampling and Estimator

<table>
<thead>
<tr>
<th>Raw Data</th>
<th>1 1 1 1 2 28 6 9 9 12 8 25 3 3 3 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>9 12 8 25 1 1 1 1 1</td>
</tr>
<tr>
<td>Sampling</td>
<td></td>
</tr>
<tr>
<td>Bi-Level</td>
<td>2 28 6 9 9 12 8 25 1 1 1 1 1</td>
</tr>
<tr>
<td>Sampling</td>
<td></td>
</tr>
</tbody>
</table>

\[
\hat{T} = \frac{N}{n} \sum_{i=1}^{n} \hat{y}_i \\
\hat{y}_i = \frac{M_i}{m_i} \sum_{j=1}^{m_i} y_{ij}
\]
Sampling and Estimator

Raw Data: 1 1 1 1 2 28 6 9 9 12 8 25 3 3 3 3

Cluster Sampling: 9 12 8 25 1 1 1 1

Bi-Level Sampling: 2 28 6 9 9 12 8 25 1 1 1 1
2 4 3 1 3 4 2 1 1 4 3 2
28 9 6 8 25 12 1 1
Sampling and Estimator

\[ \text{Var}(\hat{\tau}) = N(N - n) \frac{s_{u}^2}{n} + \frac{N}{n} \sum_{i=1}^{n} M_i (M_i - m_i) \frac{s_i^2}{m_i} \]

\[ s_{u}^2 = \frac{1}{n-1} \sum_{i=1}^{n} (\hat{y}_i - \hat{\mu}_1)^2, \quad s_i^2 = \left( \frac{1}{m_i - 1} \right) \sum_{j=1}^{m_i} (y_{ij} - \bar{y}_i)^2 \]

where \( i = 1, \ldots, n \), and \( \hat{\mu}_1 = \left( \frac{1}{n} \right) \sum_{i=1}^{n} \hat{y}_i \).

- n : number of chunks
- m : number of processed tuples
OnLine Aggregation for RAW data processing

- How to generate random samples from raw files?
  Bi-Level Sampling
- Design a feasible architecture to combine online aggregation with in-situ data processing
  OLA-RAW
- Find an efficient method to maintain processed samples
Parallel super-scalar pipeline
Where Does the Time Go?

- I/O-bound
- permutation generation
- process more tuples

CPU-bound
- permutation generation
- flush samples

I/O
- CPU-bound
- permutation generation
- process more tuples

CPU
- flush samples
- process more tuples

Load
- CPU-bound
- permutation generation
- process more tuples

Time

UCMERCED
How many samples are enough?

- Make sure to generate good enough estimation by accessing raw data only once
- Generate accurate estimate for each chunk
Query Processing

CPU-bound process

Raw Data: 1 1 1 1 2 28 6 9 9 12 8 25 3 3 3 3

Bi-Level Sampling:
- Periodical check: Good enough?
  - 28 9 6
  - 8 25 12
  - 1 1

During query loading:
- 28 9 6
- 8 25 12
- 1 1

| : Thr<sub>local</sub> | : Thr<sub>balance</sub> | DB |
Query Processing

IO-bound process

- Raw Data
- Bi-Level Sampling
- After query loading

\[\text{Periodical check Good enough?}\]

<table>
<thead>
<tr>
<th>Thr_{local}</th>
<th>Thr_{balance}</th>
</tr>
</thead>
</table>
Sampling Strategy

- Parallel sampling procedure
  Result order ≠ Random chunk order → Inspection paradox
OLAL-RAW

- **Online Aggregation for RAW data processing**
  - How to generate random samples from raw files?
    - **Bi-Level Sampling**
  - Design a feasible architecture to combine online aggregation with in-situ data processing
    - **OLA-RAW**
  - Find an efficient method to maintain processed samples
    - **In-memory sample synopsis**
Sample Maintenance

- What kind of samples should be preserved?
  Variance-driven
- When to load the samples?
  During query or loading after query processing
- How to make sure the additional samples have not been selected before?
  Permutation seeds + offset
Sample Maintenance

- Variance-driven sample swap policy
Evaluation

Data: The PTF dataset with 1 billion transient detection tuples. Each tuple has 8 attributes, 6 of which are real numbers with 10 decimal digits

Query:

\[
\text{SELECT SUM} \left( \sum_{j=1}^{K} C_{i,j} \right) \quad \text{FROM FILE}
\]

System: 2 AMD 8-core processors, 40 GB of memory, 4 disks in RAID-0 with I/O throughput 450 MB/s

Illustration: 16 attributes, \(2^{26}\) lines, 20GB
Query Execution Time

- C-1
- C-4
- C-16
- BI-1
- BI-4
- BI-16
- EXT-1
- EXT-4
- EXT-16

Error ratio vs. Elapsed time [sec]
Sample Size
Parallel Sampling Comparison

- 1 thread
- 4 threads
- 16 threads

![Graphs showing error ratio over elapsed time for different thread counts.](image-url)
Sample Synopsis
Resource Utilization

![Graph showing resource utilization over OLA-RAW processing progress (in %)]
Conclusions

- OLA-RAW is a novel resource-aware bi-level sampling method for parallel on-line aggregation over raw data.
- OLA-RAW is an efficient scheme for data exploration that avoids unnecessary work.
Thank you!
Questions?