Database Support for Processing Complex Aggregate Queries over Data Streams

Yuanzhen Ji
SAP AG
yuanzhen.ji@sap.com

PhD supervisor: Christof Fetzer
Systems Engineering Group, TU Dresden
Forewords

• Context:
  – stream processing
  – in-memory systems

• Focus:
  – exact answers of window-based operations

• Goal: Investigate potential of combing stream processing engines (SPE) and DBMSs to minimize end-to-end latency.
Outline

- Motivation
  - SPE vs. DBMS
  - Empirical study
  - Roundup
- Solution Approach
  - Thesis prototype
  - Dynamic operation migration
- Conclusion
- Discussion
SPE (Stream Processing Engine)

- Requirement: process high velocity data with low latency
- Continuous queries over streaming data
- Push-based processing model
- Window-based and incremental processing
- Key idea: single pass over data

Limitation:
- Exact answers of windowed complex aggregates are not incrementally computable
- Require multiple passes over data

$\Rightarrow$ poor end-to-end latency for operations with big windows (e.g. a few million tuples)

$10,000 \text{ events/s} \times 5 \text{ minutes} = 3 \text{ million}$
Example: Correlated Aggregate

\[ \pi_{a, b, c} \]

\[ \text{SELECT } R.a, \text{ COUNT}(R.c) \text{ FROM } R \]
\[ \text{JOIN} \]
\[ (\text{SELECT } R.a, \text{ MAX}(R.b) \text{ MAX}_B \]
\[ \text{FROM } R \text{ GROUP BY } R.a) \text{ INNERAGGR} \]
\[ \text{ON } R.a=\text{INNERAGGR}.a \]
\[ \text{WHERE } R.b<0.5*\text{INNERAGGR}.MAX_B \]
\[ \text{GROUP BY } R.a \]
Correlated Aggregate Use Cases

- Use case 1: stock market analysis
  - How many companies are there in each business area (e.g., retail, banking, etc.) whose market capitalization is greater than 50% of the maximal market capitalization in the past $k$ time units?
  - 1\textsuperscript{st} level aggregate: MAX();
  - 2\textsuperscript{nd} level aggregate: COUNT()

- Use case 2: telephone fraud detection
  - For each telephone customer (a), what percentage of calls (c) longer than the average duration (b) are international calls in the past $k$ time units?
  - 1\textsuperscript{st} level aggregate: AVG();
  - 2\textsuperscript{nd} level aggregate: QUANTILE()
DBMS (Database Management Sys.)

- Focus on persistent data management
- One-shot queries over static data
- Pull-based query processing model

=> Not a good candidate for stream processing

- But...
  - Sophisticated data management and query optimization techniques
  - Promising performance for complex aggregate queries over large dataset with in-memory technology, vertical storage architecture, etc.

- Is it possible to compensate limitations of SPE with DBMS?
Empirical Study

\[ \pi_{a, a} \sigma_{b < 0.5 \cdot \text{MAX}(b)} \pi_{a, a} \text{G COUNT}(c) \]

\[ R(a, b, c) \]

Motivation – Empirical study
Further Investigation

Motivation – Empirical study
Further Investigation (cont.)

Motivation – Empirical study

- Query
- Compilation
- Optimization
- Execution
- Result

- Statement execution time of in-memory column-oriented DB
- Server processing time of in-memory column-oriented DB

correlated aggregate
Roundup

• Goal: minimize end-to-end latency
• Measurements indicate that SPE and DB systems can be combined so as to minimize end-to-end latency

• Possible approaches
  – Federated system
  – Integrated system
Federated vs. Integrated System

Federated System

- Research problem:
  - Dynamic operation migration
- Inter-system comm. overhead
- Fits well into real-world setup

Integrated System

- Research problem:
  - Efficient continuous query proc. (window & incremental proc.)
- Single system supports both continuous and one-shot queries
- Significant kernel extension
Federated vs. Integrated System

Federated System

- Related work:
  - Federated database (IBM DB2)
  - MaxStream (ETH Zurich)

Integrated System

- Related work:
  - DataCell/Monet DB (EPFL)
  - Extended PostgreSQL (HP Lab)

Chosen approach: Federated system
Outline

• Motivation
  – SPE vs. DBMS
  – Empirical study
  – Roundup

• Solution Approach
  – Thesis prototype
  – Dynamic operation migration

• Conclusion

• Discussion
Thesis Prototype

Solution Approach – Thesis prototype

1. Client queries
3. Query Rewriter
4. Modified queries
5. runtime statistics

Streaming data

Query results
Dynamic Operation Migration

- Operation migration criterion
  - Per-tuple processing time in SPE($\rho_{spe}$)
  - Per-tuple processing time in DB($\rho_{db}$)
  - data transfer latency ($l$)

  \[ \rho_{db} + l < \rho_{spe} \]

- Obtaining good estimation of $\rho_{db}$, $\rho_{spe}$, and $l$ is non-trivial
Dynamic Operation Migration

- Problem: varying system load and input data rate
- Two-step approach
  - Initial operation distribution
  - Runtime adaptation
- Challenges:
  - Avoid thrashing

Solution Approach – Dynamic operation migration
Outline

• Motivation
  – SPE vs. DBMS
  – Empirical study
  – Roundup

• Solution Approach
  – Thesis prototype
  – Dynamic operation migration

• Conclusion
• Discussion
Conclusion

• SPEs support real-time event handling by incremental operation evaluation.
• Exact answers of complex aggregates are not incrementally computable.
• Modern DBMSs show better performance for these operations.

⇒ Combine SPE and DBMS for stream processing to minimize end-to-end latency
⇒ Keep incrementally computable operations in SPE
⇒ Dynamically migrate complex aggregate operations from SPE to DBMS

Thank you!
References