An overview of Data Streaming

Vincenzo Gulisano
vinmas@chalmers.se (room 5119)
Agenda

- Motivation
- The data streaming philosophy
- System Model
- Sample Data Streaming application
- Evolution of Stream Processing Engines
- Challenges in the context of Smart Grids
Agenda

• Motivation
• The data streaming philosophy
• System Model
• Sample Data Streaming application
• Evolution of Stream Processing Engines
• Challenges in the context of Smart Grids
Motivation

• Applications such as:
  – Sensor networks
  – Network Traffic Analysis
  – Financial tickers
  – Transaction Log Analysis
  – Fraud Detection

• Require:
  – Continuous processing of data streams
  – Real Time Fashion
Motivation

• Store and process is not feasible
  – high-speed networks, nanoseconds to handle a packet
  – ISP router: gigabytes of headers every hour,…

• Data Streaming:
  – In memory
  – Bounded resources
  – Efficient one-pass analysis
Motivation

• DBMS vs. DSMS

1 Data

2 Query

Query Results

Main Memory

Query Processing

Disk

Continuous Query

Main Memory

Query Processing

Query Results
Agenda

• Motivation
• The data streaming philosophy
• System Model
• Sample Data Streaming application
• Evolution of Stream Processing Engines
• Challenges in the context of Smart Grids
Database vs. Data Streaming

• Problem:
  – James travels by car from A to B
  – His grandmother is worried, she wants to know if he exceeds the speed limit

• How will the “database” and the “data streaming” grandmothers do this?
Database vs. Data Streaming

\[ \text{distance}(A, B) = \frac{\text{End time} - \text{Start time}}{\text{Position B} - \text{Position A}} \]

Start time
Position A

End time
Position B

Database grandmother
Database vs. Data Streaming

1. First the data, then the query
2. Precise result
3. Need to store information
Database vs. Data Streaming

1. First the query, then the data
2. “Continuous” result
3. No need to store information

Data streaming grandmother
 Agenda

• Motivation
• The data streaming philosophy
• System Model
• Sample Data Streaming application
• Evolution of Stream Processing Engines
• Challenges in the context of Smart Grids
System Model

- Data Stream: unbounded sequence of tuples
  - Example: Call Description Record (CDR)

<table>
<thead>
<tr>
<th>Field</th>
<th>Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caller</td>
<td>text</td>
</tr>
<tr>
<td>Callee</td>
<td>text</td>
</tr>
<tr>
<td>Time (secs)</td>
<td>int</td>
</tr>
<tr>
<td>Price (€)</td>
<td>double</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>8:00</th>
<th>3</th>
<th>C</th>
<th>D</th>
<th>8:20</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>E</td>
<td>8:35</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2014-04-09
System Model

• Operators:

  Stateless
  1 input tuple
  1 output tuple

  Stateful
  1+ input tuple(s)
  1 output tuple

2014-04-09
Stateless Operators

**Map**: transform tuples schema
Example: convert price € $%

**Filter**: discard / route tuples
Example: route depending on price

**Union**: merge multiple streams (sharing the same schema)
Example: merge CDRs from different sources
System Model

Stateful Operators

**Aggregate**: compute aggregate functions (group-by)
Example: compute avg. call duration

**Equijoin**: match tuples from 2 streams (equality predicate)
Example: match CDRs with same price

**Cartesian Product**: merge tuples from 2 streams (arbitrary predicate)
Example: match CDRs with prices in the same range
System Model

- Infinite sequence of tuples / bounded memory
  → windows
- Example: 1 hour windows
System Model

- Infinite sequence of tuples / bounded memory
  → windows
- Example: count tuples - 1 hour windows

Output: 4
Agenda

• Motivation
• The data streaming philosophy
• System Model
• Sample Data Streaming application
• Evolution of Stream Processing Engines
• Challenges in the context of Smart Grids
Continuous Query Example

- Fraud detection, High Mobility
  - Spot mobile phone whose space and time distance between two consecutive calls is suspicious
High Mobility Continuous Query (1/2)

Field
- Caller
- Callee
- Time
- Duration
- Price
- Caller_Position
- Callee_Position

Input Stream

Map
Remove fields that are not needed

Map
Create separate tuple for caller

Map
Create separate tuple for callee

Field
- Phone number
- Start time
- End time
- Position

Union
Merge tuples
High Mobility Continuous Query (2/2)

For each consecutive pair of calls referring to the same number compute speed

Forward tuples with speed exceeding a given threshold

Window type: tuple based
Window size: 2
Window Advance: 1
Agenda

• Motivation
• The data streaming philosophy
• System Model
• Sample Data Streaming application
• Evolution of Stream Processing Engines
• Challenges in the context of Smart Grids
Centralized SPEs
Distributed SPEs

Inter-operator parallelism
Parallel SPEs

Intra-operator parallelism

Over-provisioning or under-provisioning?
Elastic SPEs

Scale up
Elastic SPEs

Scale down

2014-04-09
Agenda

• Motivation
• The data streaming philosophy
• System Model
• Sample Data Streaming application
• Evolution of Stream Processing Engines
• Challenges in the context of Smart Grids
Challenges in the context of Smart Grids

• Process energy consumption data
  – Build profiles and spot deviations
  – Predictions / forecasts about consumption
Challenges in the context of Smart Grids

• Process control events
  – Spot possible threats
  – Monitor the devices status
Challenges in the context of Smart Grids

How to process the information?

Centralized
Challenges in the context of Smart Grids

How to process the information?

Distributed
(In-network aggregation)
Challenges in the context of Smart Grids

How to deal with constrained/limited resources?

What if this device is running out of battery?
An overview of Data Streaming

Questions?
Bibliography


