Integrating Massive Data Streams

George Siachamis
Supervised by Geert-Jan Houben, Arie van Deursen and Asterios Katsifodimos

VLDB PhD Workshop
16/08/2021
Data Integration

• A long-standing & challenging problem.
• A traditionally manual task.
• An important, time-consuming preparatory task for any data scientist.

Typical Data Science Workflow
Data Integration

• Existing research focuses on:

  Automating the task

  Improving accuracy

  Improving efficiency
Streaming Data Integration

Traditionally

• Data stored on repositories.
• For months before processing.
• Offline analysis.

Modern Enterprises

• Data-driven real-time applications and analytics
• Fast decisions
• Online analysis.

In need for efficient data integration on streams to ensure quality!
Motivation

• Modern enterprises consist of multiple independent teams who manage their own data.
• Streaming data from those teams are “stored” in an internal streaming data lake.
  • Usually without valuable metadata.

• The absence of provided metadata render the ”stored” data unusable from other teams.
Motivating Example

• Motivating Use Case from ING

Alert 1: <server1234, CPU overload>
Alert 2: <VA@srv1234, Not responding>
Challenges

High velocity: Incoming records arrive in fast pace and they need to be processed immediately.

Concept drift: Statistical or other data properties change frequently.

Unboundedness: Streams can be infinite while our processing power is finite.
Streaming Data Lake

Three proposed operations:

• Stream profiling
• Stream discovery
• Stream integration
Stream Profiling

• Two categories of profiles:
  • Statistical: cardinalities, value distributions, data types etc.
  • Sketches & Summaries

• For a streaming data lake, profiles must be:
  • Computed in an online-fashion
  • Updated in a timely manner to capture the temporal properties of the streams
  • Incorporate time
  • Computed incrementally
Stream discovery

• Identify similar streams

• Provide temporal similarity queries.
  • Find streams that are similar in different timeframes.

• Existing solutions:
  • Can provide efficient parallel solutions
  • Are not designed for streams
  • Cannot handle the temporal needs.
Streaming Integration

• There are various ways of combining and integrating information
• However, joining the sources is one of the core tasks.

• We focus on streaming similarity joins.

What is a streaming similarity join?

Given two streams, join incoming records based on similar values of one or more target attributes.
Streaming Similarity Joins

Challenges in Streaming Similarity Joins
• Expensive similarity computations.
• Difficulty to reduced the number of similarity comparisons.
• Computation load balanced across multiple nodes.

Existing work:
• Non distributed solutions
• Application specific solutions
• Plenty of work in the MapReduce environment
The end

Thank you for your attention!

Contact details:

g.siachamis@tudelft.nl