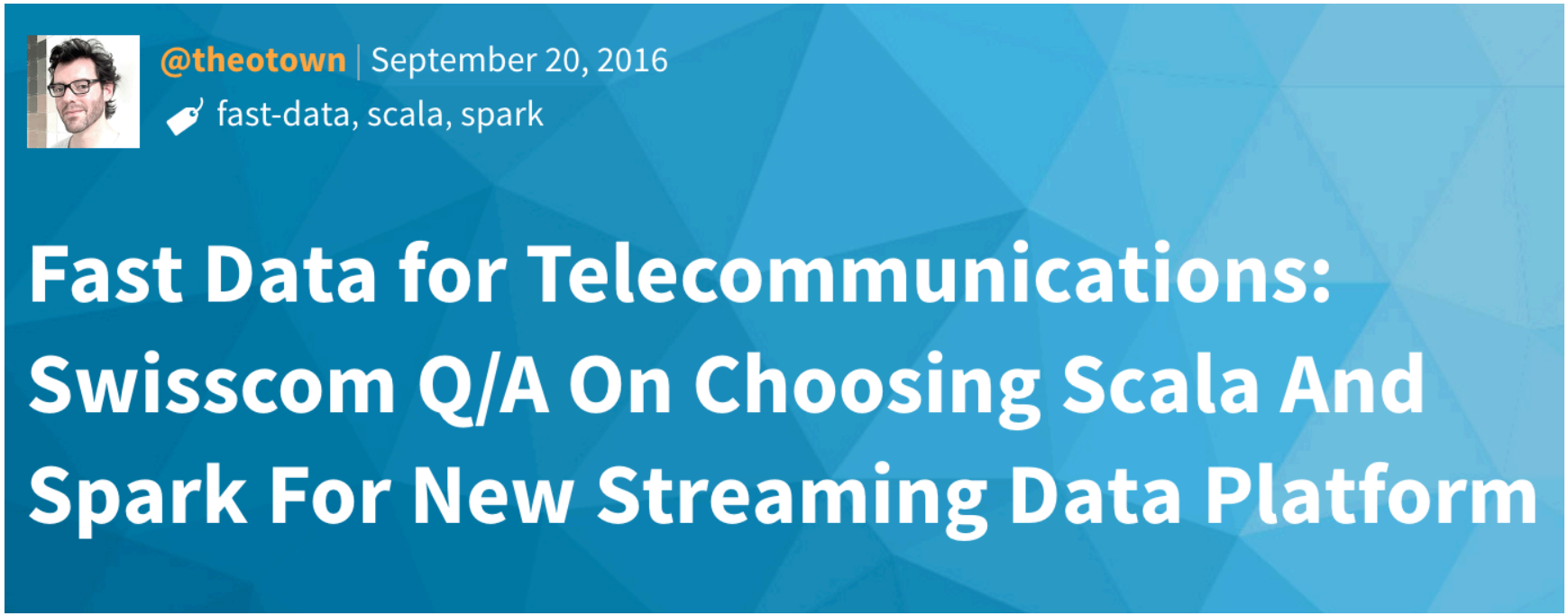
Stream processing

Vincent Leroy

Motivation



Analyze traffic in cities
using mobile phones

link

Batch VS Stream processing

- Batch
  - Process a full (large) dataset from scratch
  - Focus on throughput (time / size)
  - Takes a long time (minutes, hours …) to obtain results
  - Complex analysis requiring multiple pass over data (e.g. machine learning)

  → Good for analyzing a static dataset (post-mortem)

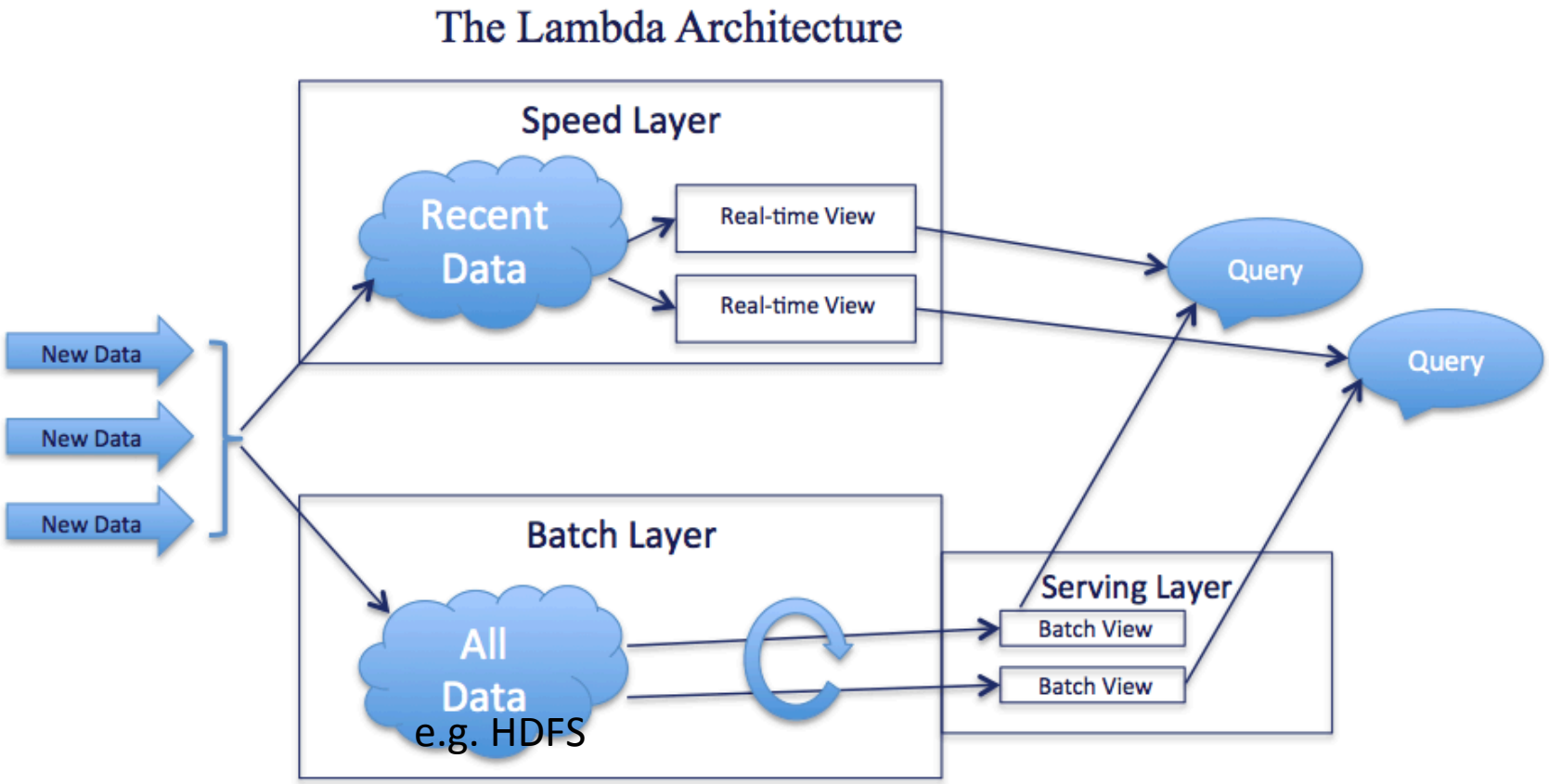
Batch VS Stream processing

- Stream
  - Process recent data (small window) to continuously update results
  - Focus on latency (time between data production and results update)
  - Near real-time
  - Incremental analysis, see data only once

  → Good to analyze live data (e.g. what is trending on Twitter?)

https://softwareengineeringdaily.com/2015/08/03/batch-vs-streaming-the-differences/

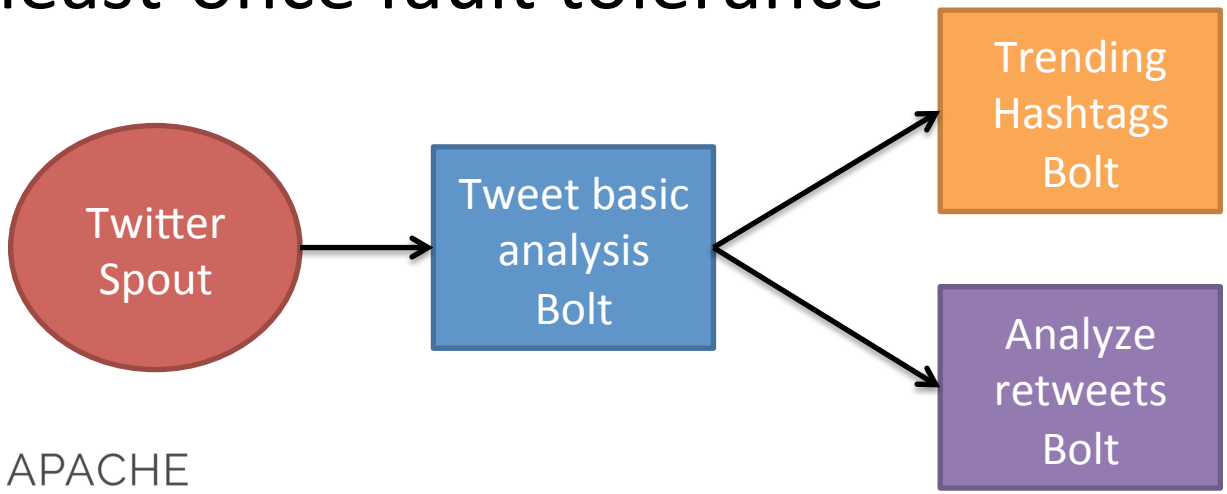
Lambda Architecture



https://www.voltdb.com/blog/simplifying-complex-lambda-architecture

Low latency stream processing: STORM

- Directed Acyclic Graph (DAG)
  - Spouts inject data tuples to create a stream
  - Bolts consume streams and emit new streams
- At-least-once fault tolerance



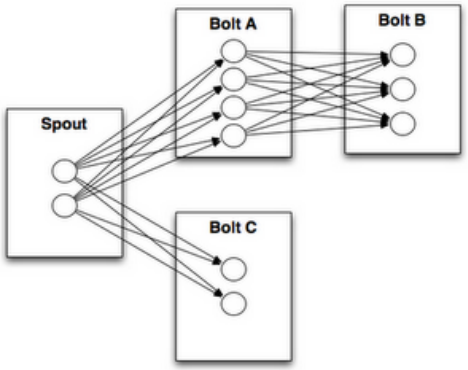
APACHE STORM™
Distributed • Resilient • Real-time

STORM scalability

- Each bolt defines operations on a stream
  - This is what developers implement

```
@Override
public void execute(Tuple tuple) {
    _collector.emit(tuple, new Values(tuple.getString(0) + "!!!"));
    _collector.ack(tuple);
}
```

- Scalability using parallelism (again!)
  - A bolt is deployed over multiple tasks

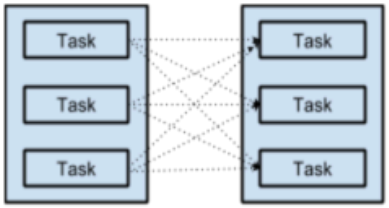


STORM stream routing

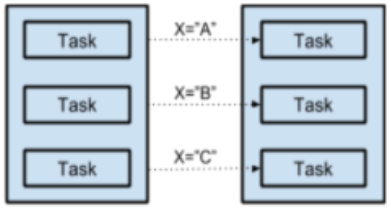
- When a bolt has multiple task, which specific task should get a data tuple?

Any: Shuffle grouping
- Stateless operator
- Like Map

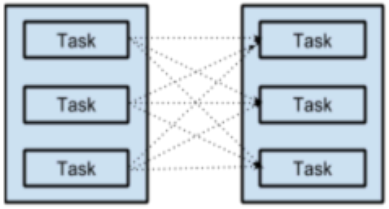
A specific task: Fields grouping
- Stateful operator
- Like Reduce



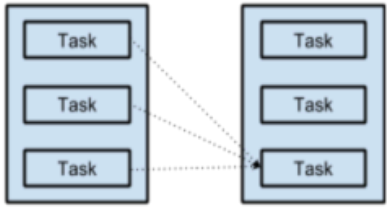
Shuffle Grouping



Fields Grouping



All Grouping



Global Grouping

Mini-Batch Stream Processing: Spark

- Launch many small jobs to simulate low latency processing