Streaming 101

Ilyas Toumlilt
i.toumlilt@criteo.com

Session 1 – 01/03/2023

Survey

Reviewing CS Basics

☕☕☕☕☕

Data-intensive streaming systems

Streaming Demo

What should I expect?

Subtitle

- 7 study days: Streaming, Kafka, Flink and company
- Strong foundation with (almost real) practical labs
- An email a day before class about what you will learn

Source:

What I will learn

Introduction to Streaming (101)	01 Feb (08:30 – 11:45)
Introduction to Messaging Systems	08 Mar (08:30 – 11:45)
Deep Architecture Kafka	15 Mar (08:30 – 11:45)
Kafka Lab Exam (Compulsory)	13 Apr (08:30 – 11:45)
Distributed Consensus Algorithms	18 Apr (08:30 – 11:45)
Introduction to Flink	23 May (08:30 – 11:45)
Flink High-Level Customisation	30 May (13:45 – 17:00)
Final Exam (Compulsory)	27 Jun (08:30 – 11:45) **TBC**

Source:

References

- Designing Data-Intensive Applications – Martin Kleppmann
- Kafka: The Definitive Guide – Gwen Shapira et al.
- Stream Processing with Apache Flink – Fabian Hueske

CRITEO

Let’s start with a survey

Reviewing CS Basics

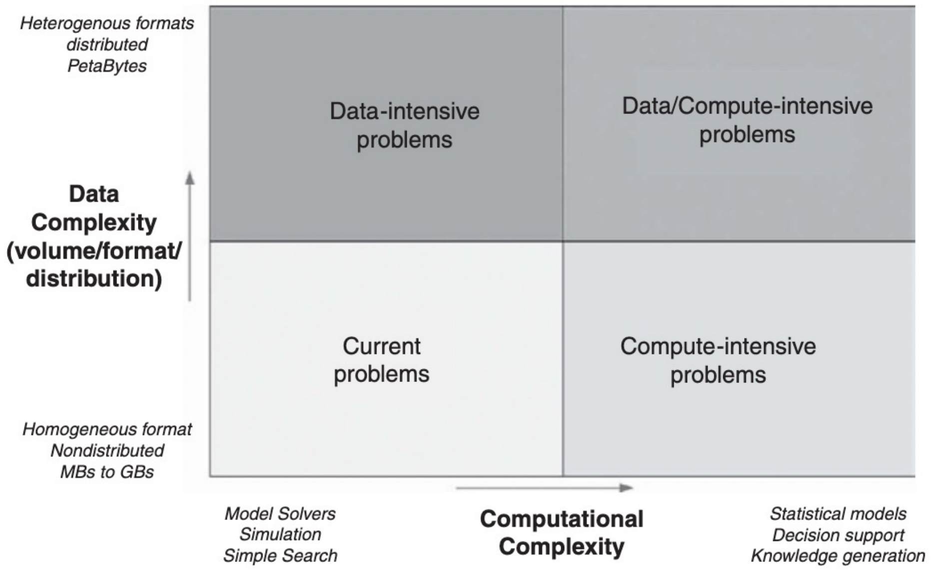
Computing in General

- Data-intensive computing

  e.g. HDFS, SPARK, KAFKA, PRESTO ,FLINK etc.

- Compute-intensive computing

  e.g.HPC, OPENMP, MPI



Gorton, Ian, and Deborah K. Gracio, eds. Data-intensive computing: architectures, algorithms, and applications. Cambridge University Press, 2012.

Data-driving decision making

- Information Theory: Data is a frozen information.
- Data drives decisions not every day, every sec!
- Decisions could be make by humans or by software

Real Numbers

- CERN LHC detectors generates 300 GBps [1]
- The New York Stock Exchange generates about 4-5 terabytes of data per day [2]
- The Internet Archive stores around 18.5 petabytes of data[3]

[1] https://wlcg.web.cern.ch/

[2] http://bit.ly/nyse_data_deluge

[3] https://archive.org/web/petabox.php

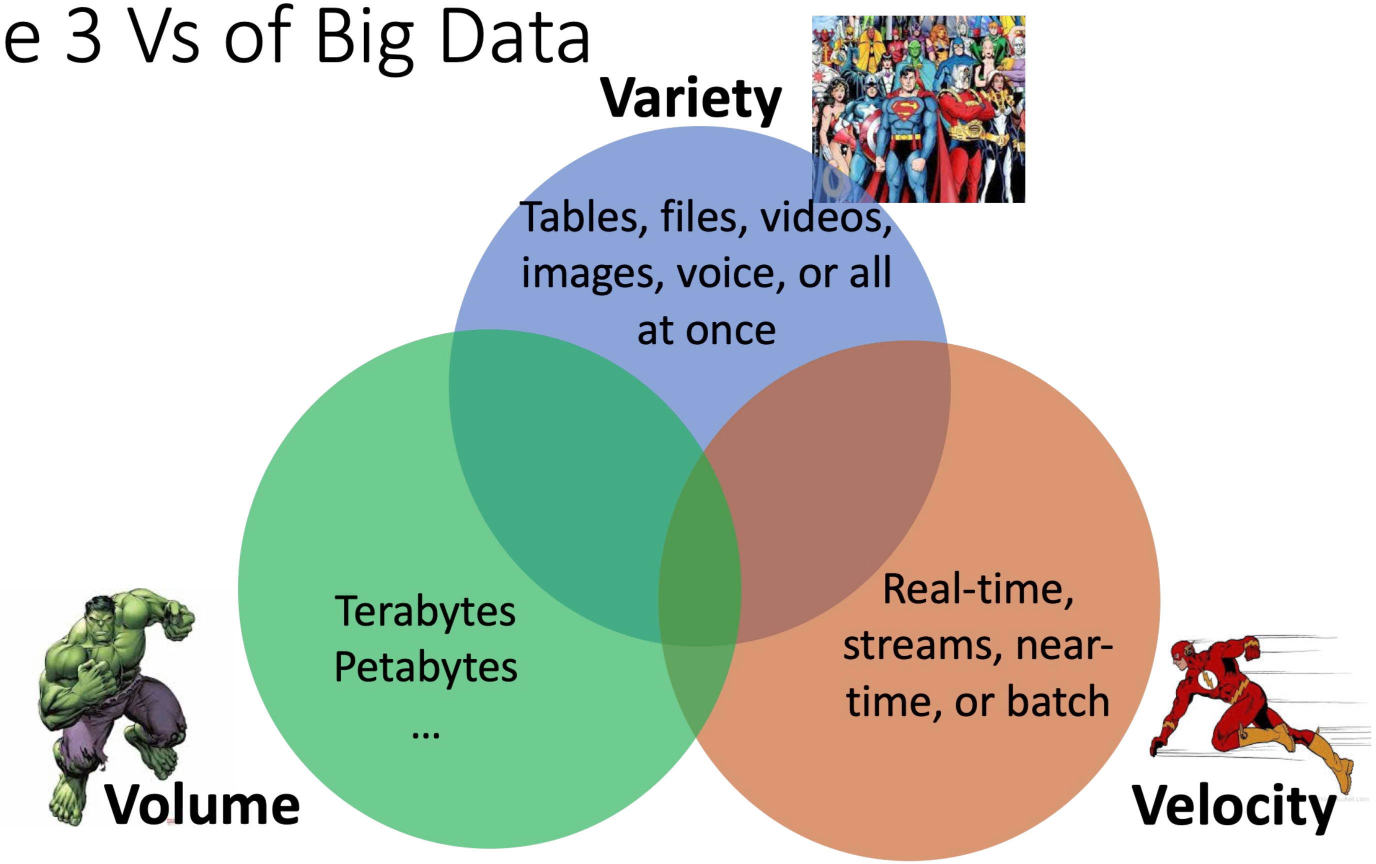
CRITEO

Problems of Data-intensive Distributed Computing

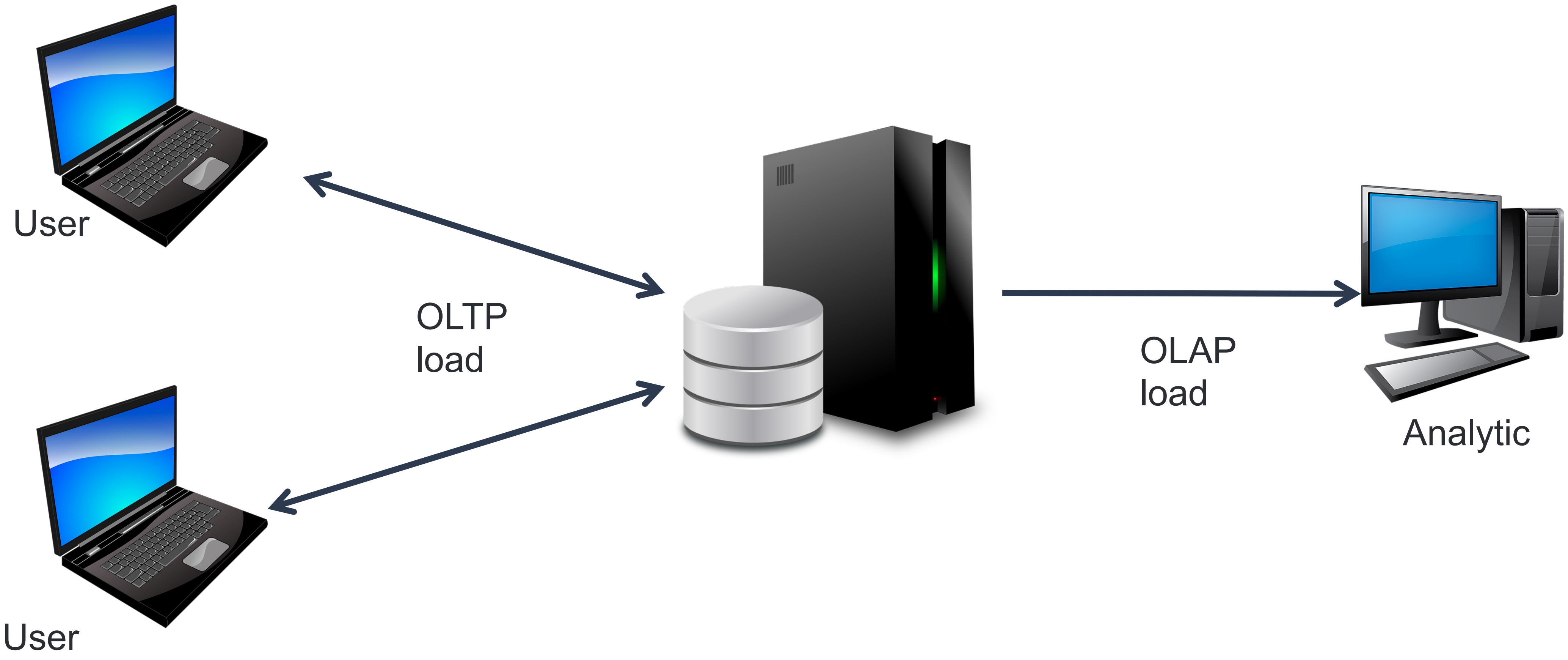
- The programming model: MapReduce
- Reliability and availability constraints
- Scalability
- Data locality

Dimensions of Data-intensive Computing

The 3 Vs of Big Data



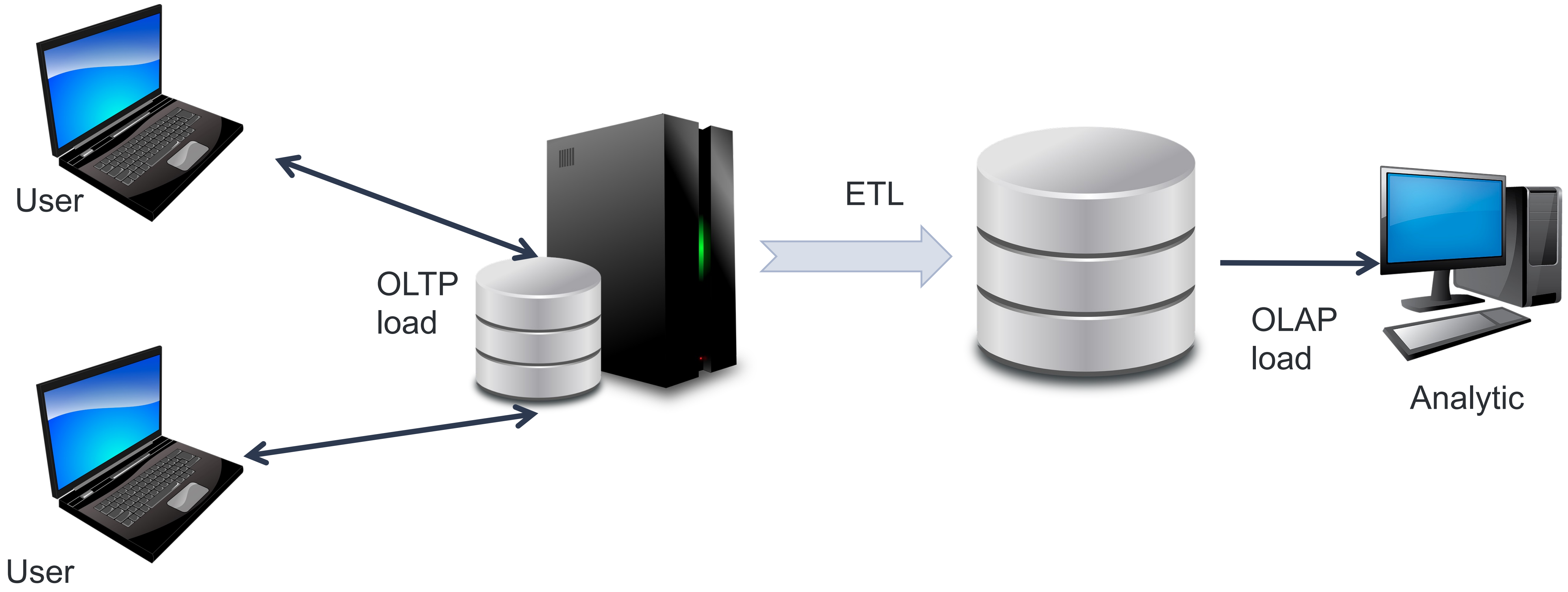
Data systems evolution



OLAP vs OLTP

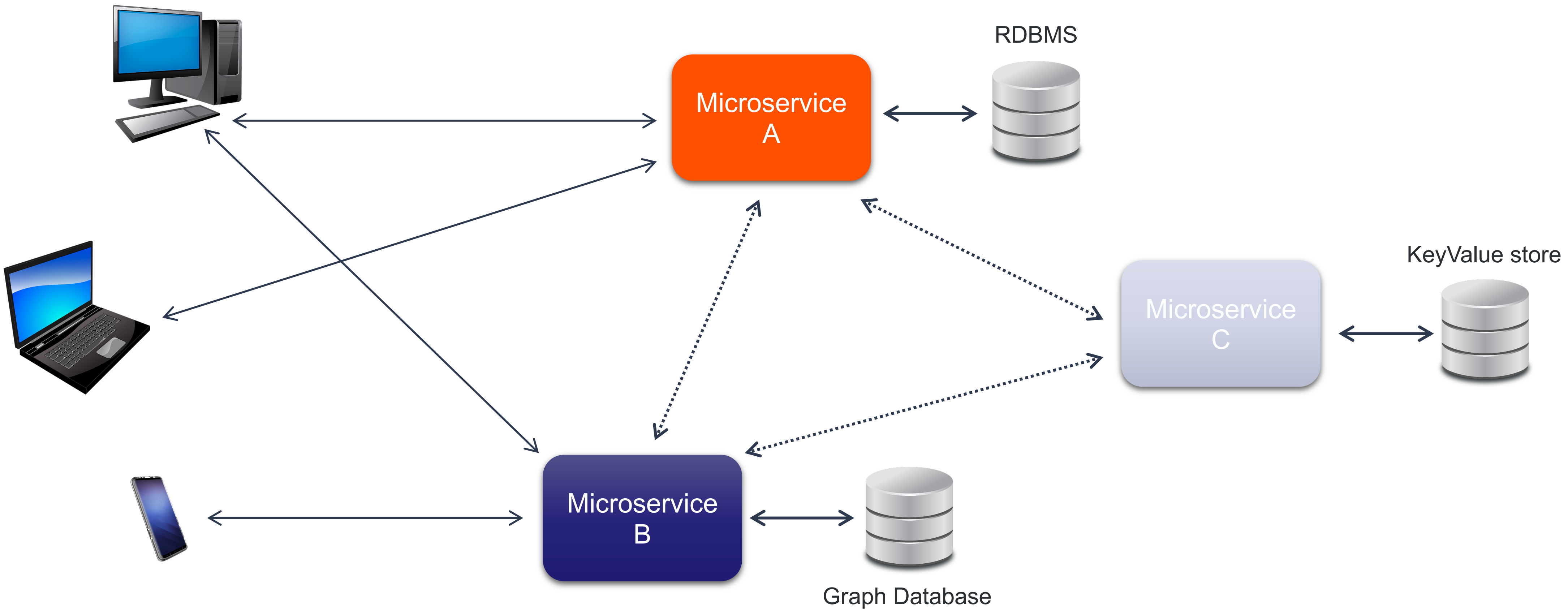
	OLAP	OLTP
Characteristics & Query Types	Complex involves more joins	Relatively simple
Response time	Minutes maybe even hours	Milliseconds level
Constraints	Relaxed integrity constraints (not normalized)	Database follows integrity constraint (normalized)

Data systems evolution

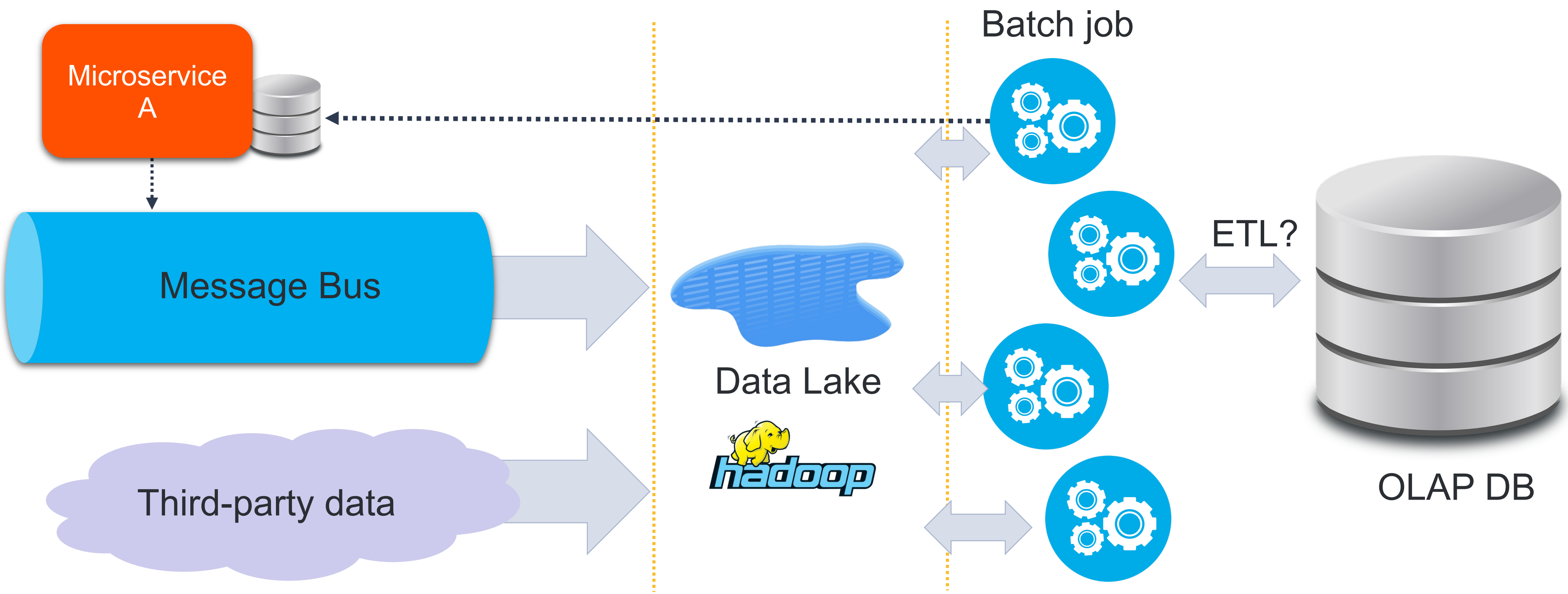


CRITEO

Microservices

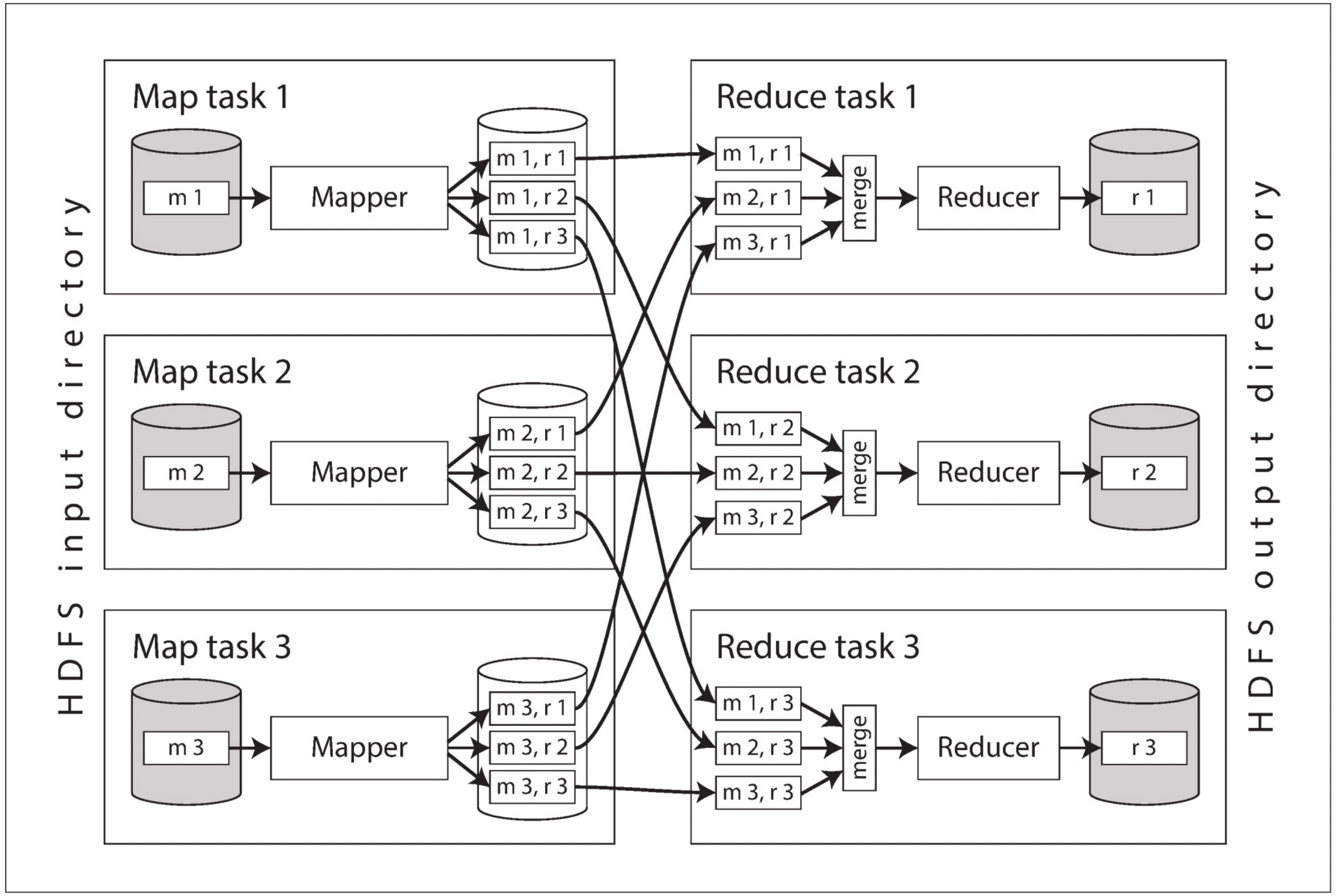


Batch Processing



CRITEO

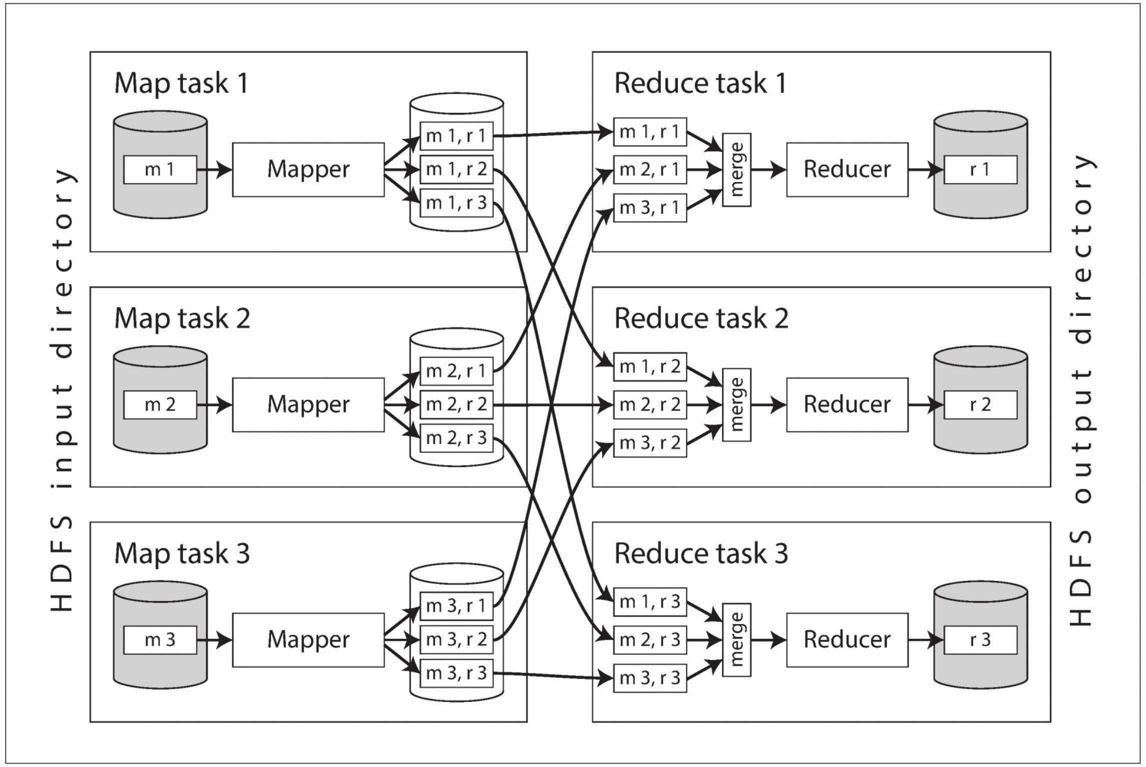
Batch Processing(MapReduce)



CRITEO

Well-known MR Problems

- Handling skew
- Map-side joins
- Broadcast hash joins



Real Use-case:

- Daily-batch, analysing production logs.
- Building search index
- Building machine learning algorithms
- Recommender systems algorithms

http://wiki.apache.org/hadoop/PoweredBy

Streaming

- "A type of data processing engine that is designed with infinite datasets in mind."
- Streaming systems may have lower latency than batch jobs, but this may be seen as a grateful side effect. In fact, streaming systems are made to deal with endless datasets by design.
- This means that even very small batches may be seen as a streaming system.

Streaming Formal Definition

$$stream(t) = \frac{\mathrm{d}\, state(t)}{\mathrm{d}t}$$

$$state(now) = \int_{t=0}^{now} stream(t)\ \mathrm{d}t$$

CRITEO

Unbounded data streams

- Unbounded data is an ever-growing, essentially infinite data set.
- It reflects the reality in a much natural way.
- In a batch world, the data must be finite, with a beginning and an end. Usually this is defined by partitions (by hour, by client etc).
- Unbounded data means there will always be new data arriving.
- The rate of new data usually is non-deterministic.
- We can think a bounded dataset as a subset of the unbounded dataset.

Let's talk about events!

- Events are stored in the message system one after the other.
- The stream of events can also be seen as a log of messages.
- Each message describe an event and contains the informations required to be processed:
  - User 1 added the product X to the shopping cart;
  - User 2 logged in;
  - Sensor Y recorded the temperature Z;
  - User 4 paused the video X at the position Y.

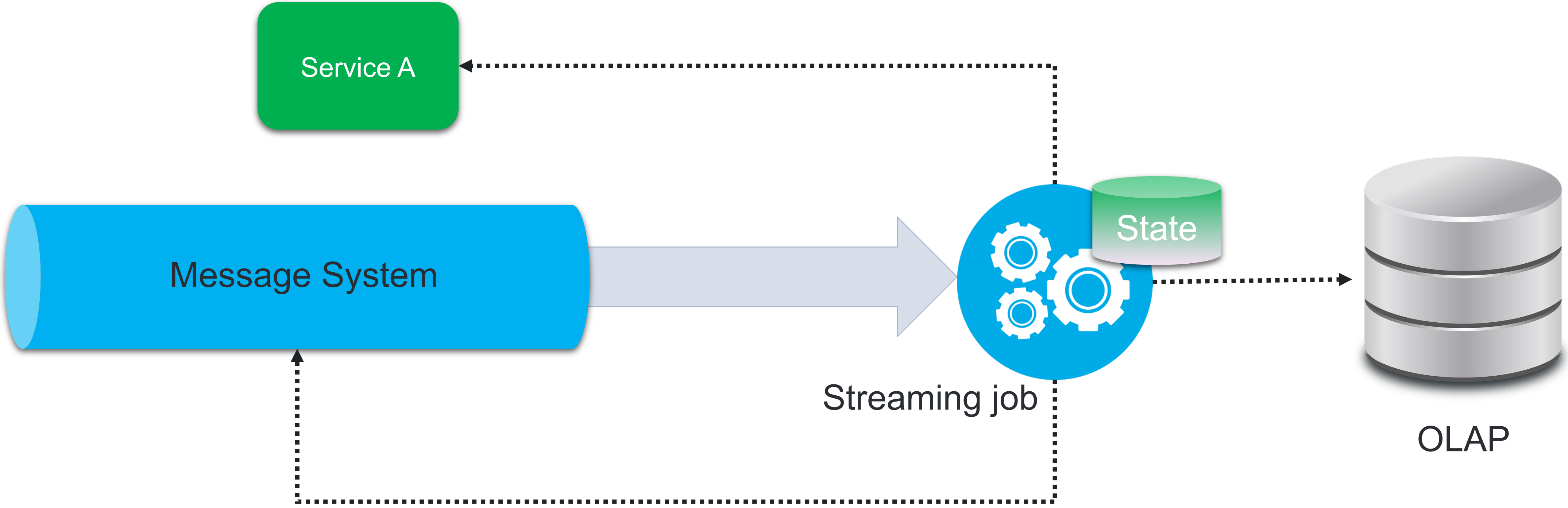
Log - Table Duality

a: +1	b: +2	c: +4	b: -1	d: 3	d: -2

$\equiv$

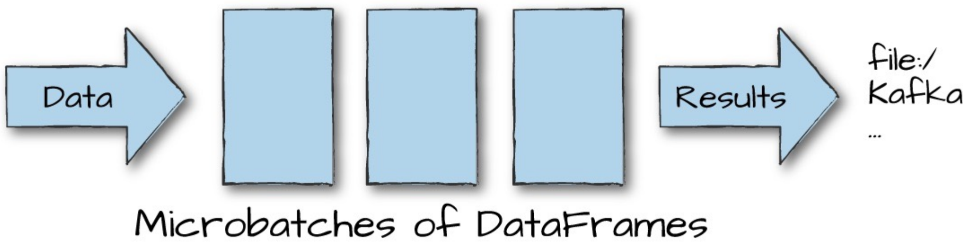
a	1	1
b	2-1	1
c	4	4
d	3-2	1

Stream Processing

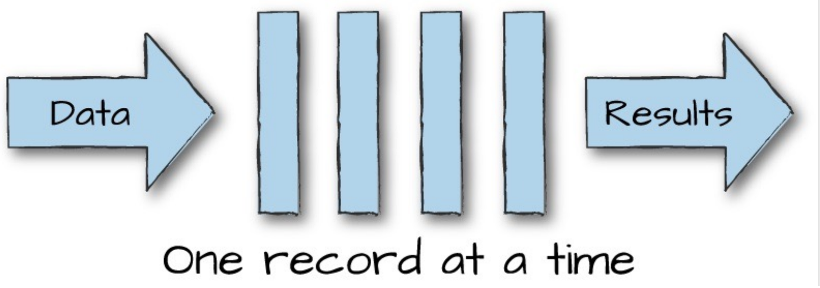


Stream Processing Patterns

- Micro-batch systems

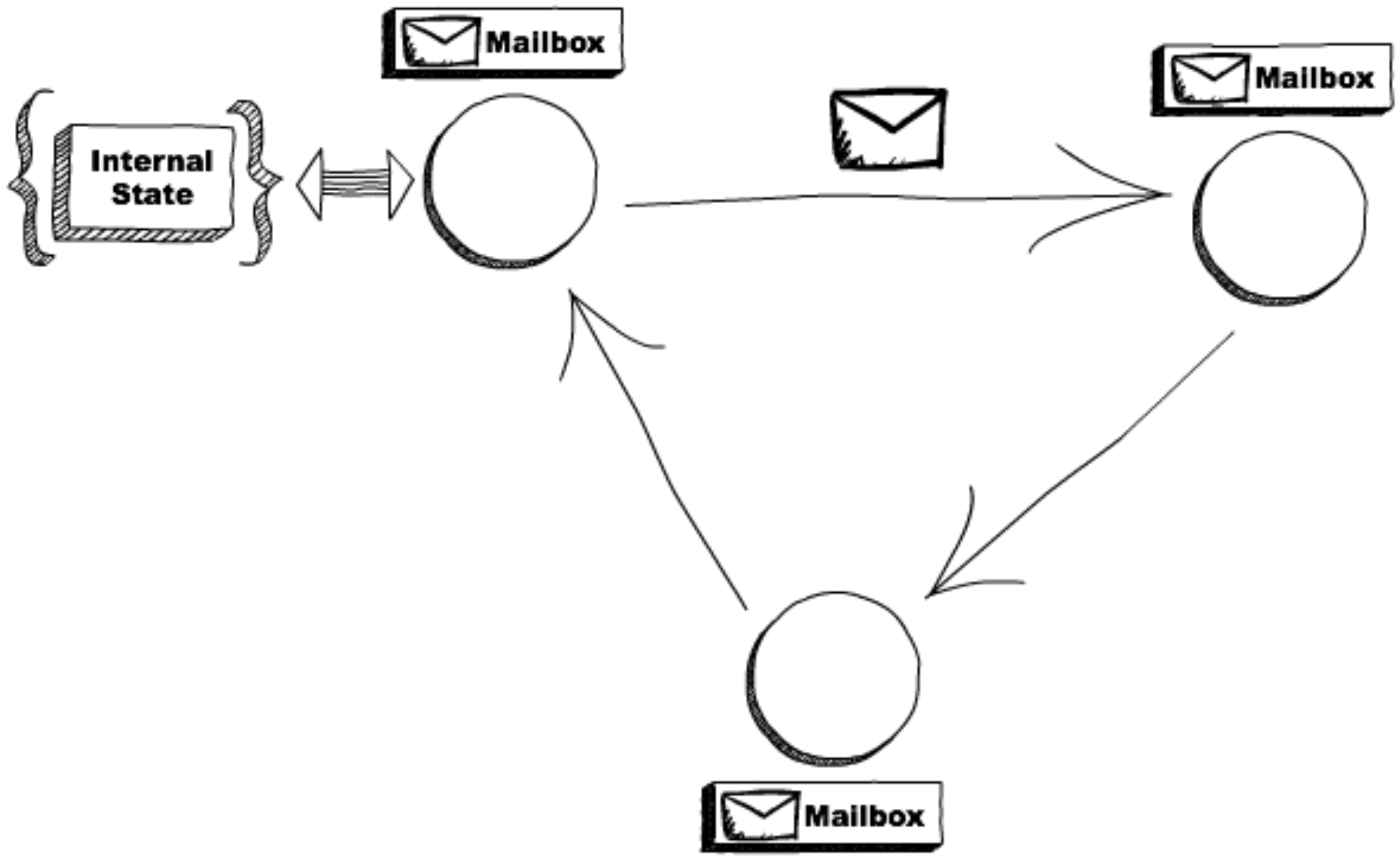


- Continuous processing-based systems



CRITEO

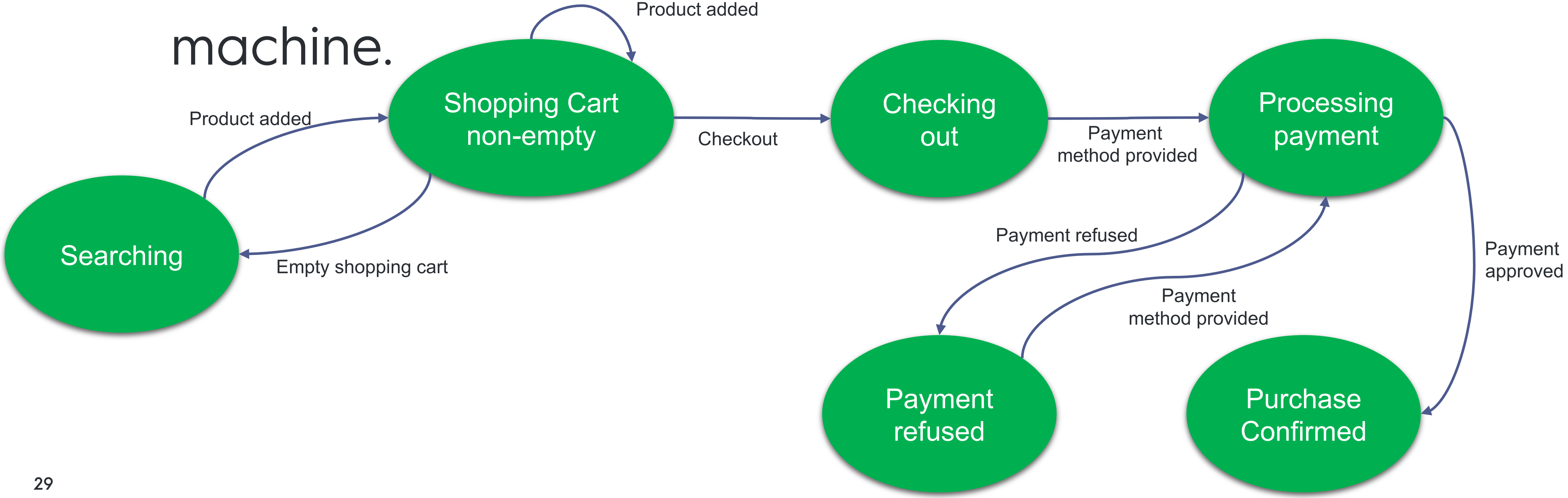
Streaming and Actor Models



CRITEO

Finite State Machine

- A stream of events can be used to feed a state machine.



CRITEO

Stream Processing vs Batch Processing

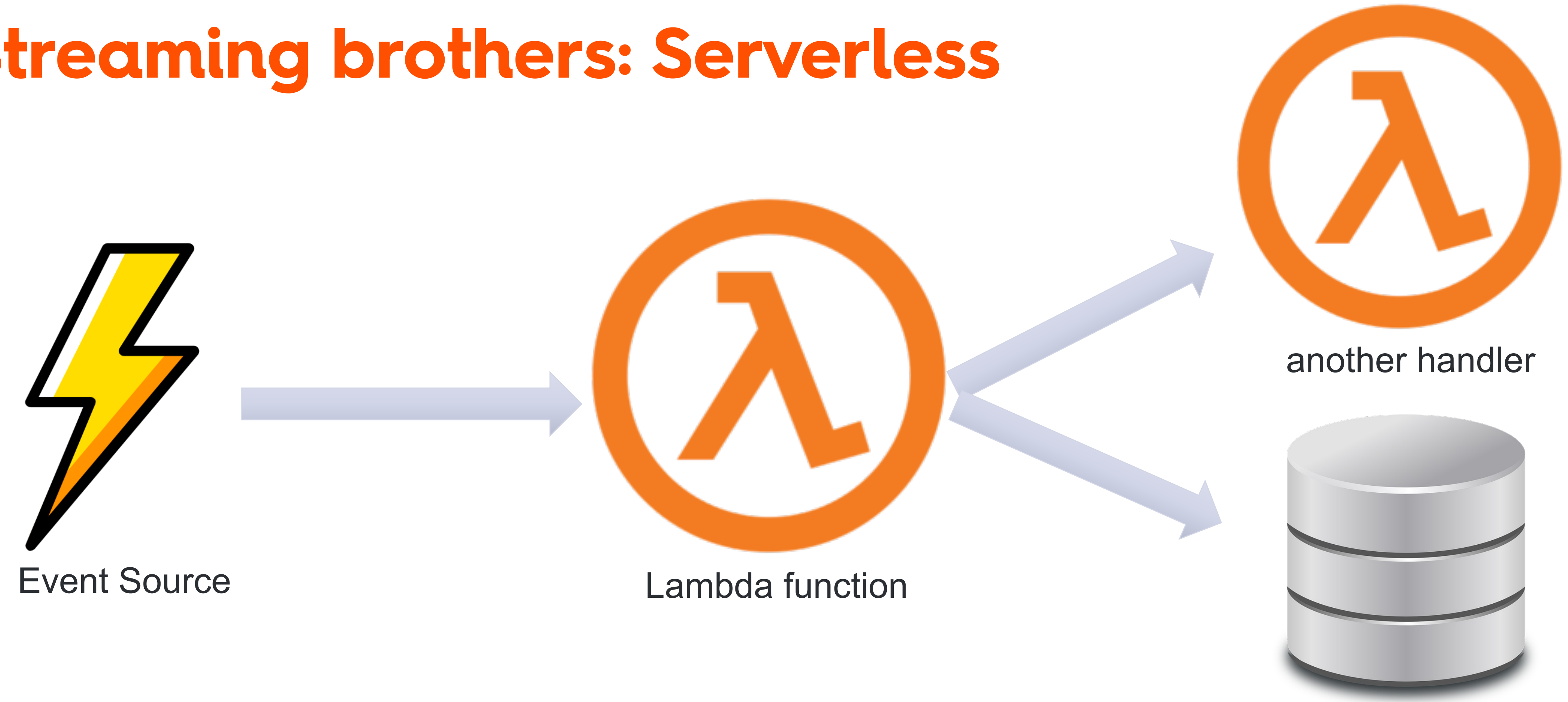
Batch

- Event time processing is for free
- Optimizations built on plan of executions
- Batches allows to catch up faster
- Scales

Streaming

- Less moving components
- Flexible data partitioning and windowing
- Less overhead -> reactivity
- Faster feedback loop during the development
- Time based joins are easy

Streaming brothers: Serverless



CRITEO

What about a break? ☕

CRITEO

Streaming Details

Distributed Log storage Zoo

- Apache Kafka
- Apache Pulsar
- Pravega
- Redpanda

Problems arise in Streaming

- Time
- Windows and late messages
- State

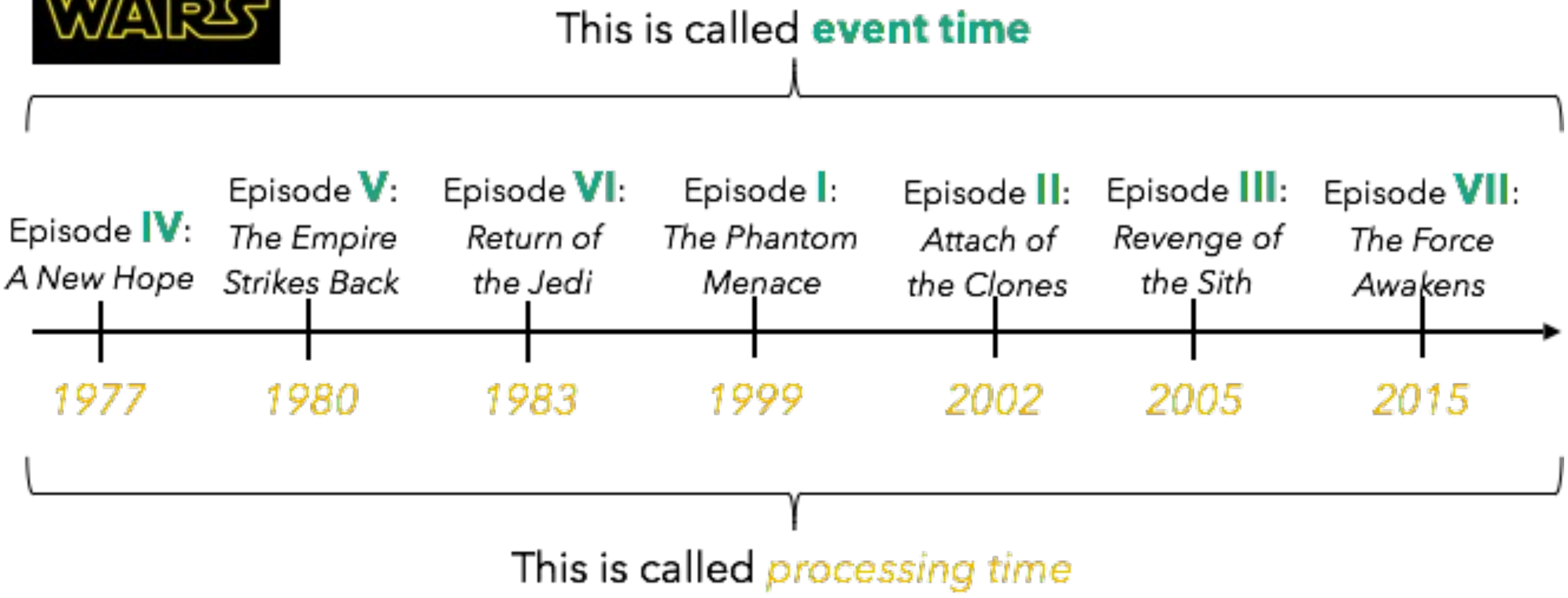
Time

- What is being computed?
- Where in event time?
- When in processing time?
- How do refinements relate?

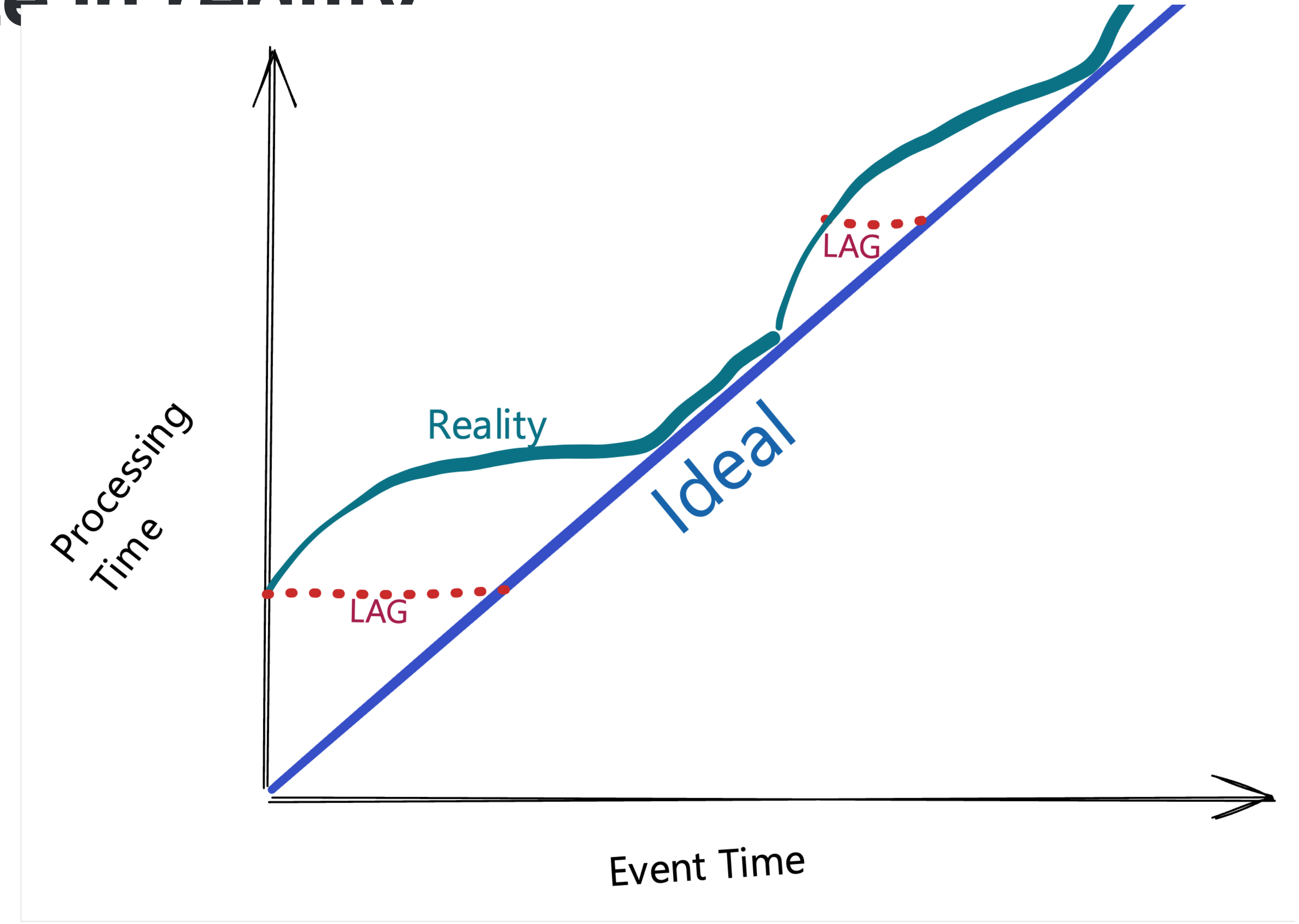
Time

- **Event time** - the time at which events actually occurred.
- **Processing time** - the time at which events are processed by the application.
- **Ingestion time** - the time at which events entered the message system.
- Not all use cases care about Event time.
- In an ideal world, Event time and Processing time would always be equal, with events being processed immediately as they occur.

Event time vs processing time

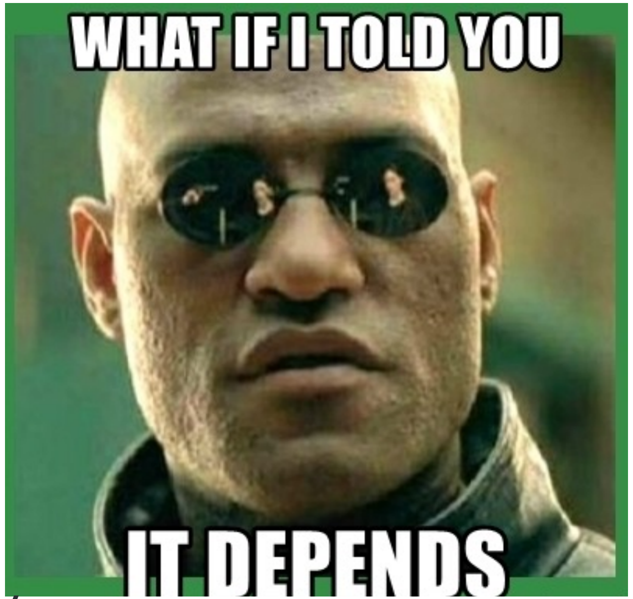


Time in reality

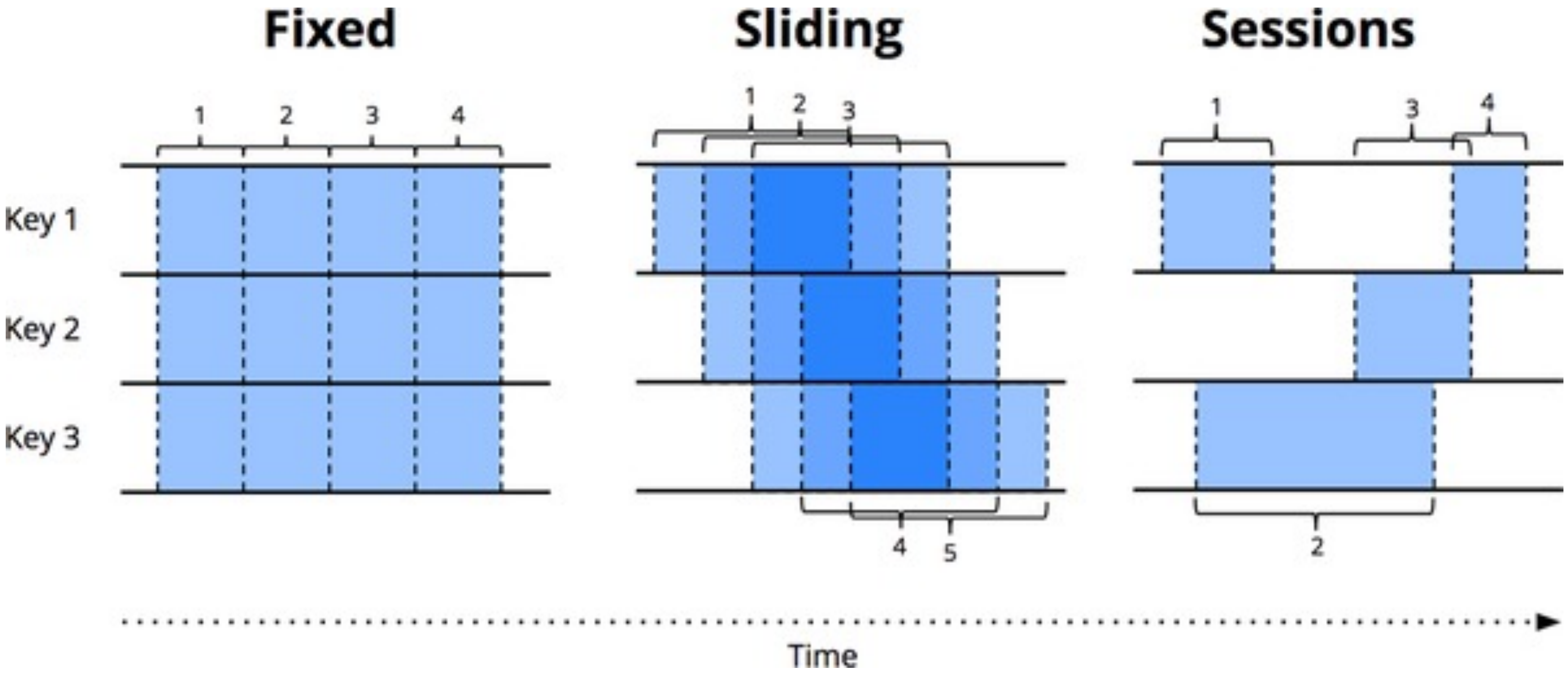


How much we should care about time ?

- Time-agnostic jobs:
  filtering, joins,map-only
- Approximation algorithms:
  top-N, streaming K means
- Windowing:
  fixed windows, sliding window, session window



Time Windows



Demo Time

CRITEO