Working with Windowing and Join Operations

Janani Ravi
CO-FOUNDER, LOONYCORN

www.loonycorn.com

Overview

Sliding, tumbling, session, and global windows

Event time vs. processing time

Watermarks and late arrivals

Performing join operations on streams

Using side inputs for processing

Apache Flink and Apache Spark 2 support for Beam

Stateless and Stateful Transformations

Transformations

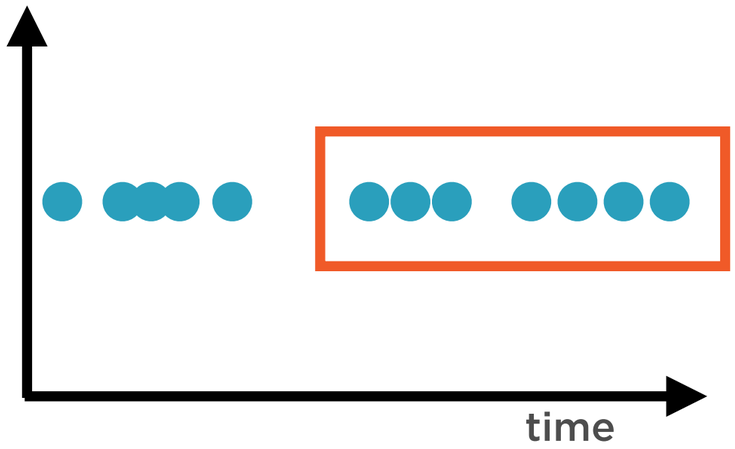
Stateless

Transformations which are applied on a single stream entity

Stateful

Transformations which accumulate across multiple stream entities

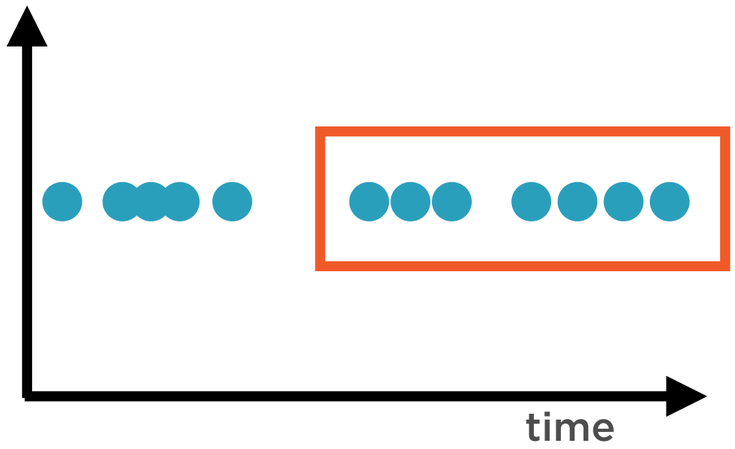
Streaming Data



A window is a subset of a stream based on

- Time interval
- Count of entities
- Interval between entities

Streaming Data



Transformations can be applied on all entities within a window

- sum, min, max, average

Types of Windows

Types of Windows

Tumbling Window

Sliding Window

Count Window

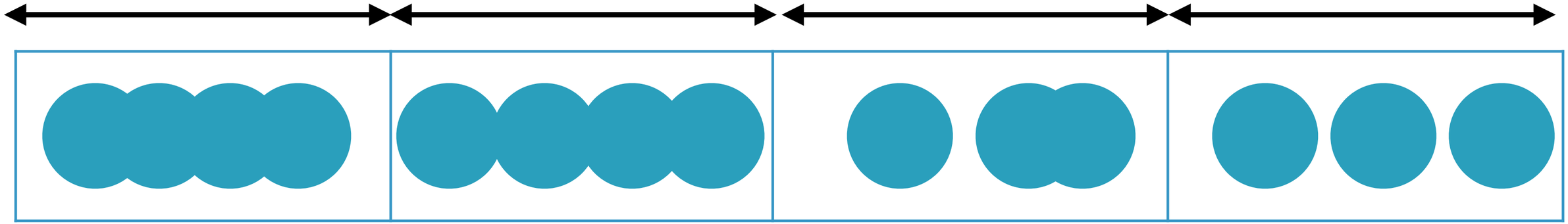
Session Window

Global Window

Types of Windows

A stream of data

Tumbling Window

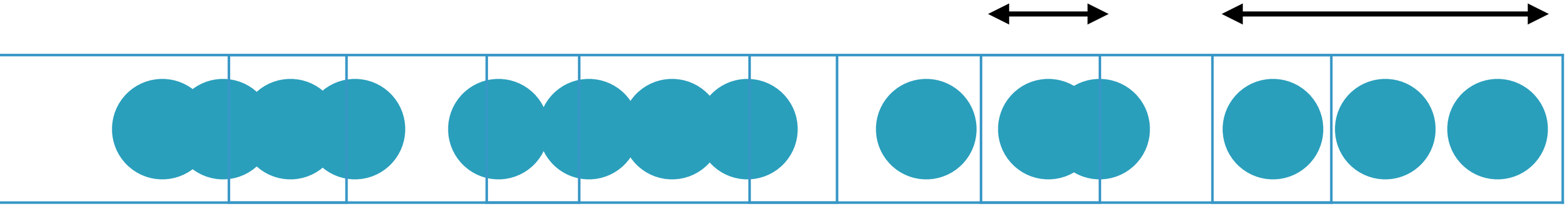


Fixed window size

Non-overlapping time

Number of entities differ within a window

Sliding Window

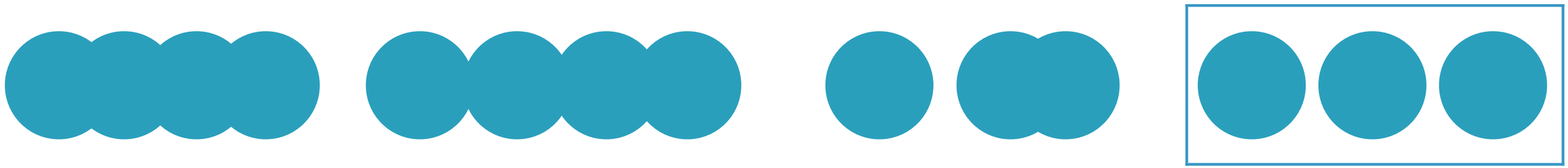


Fixed window size

Overlapping time - sliding interval

Number of entities differ within a window

Count Window

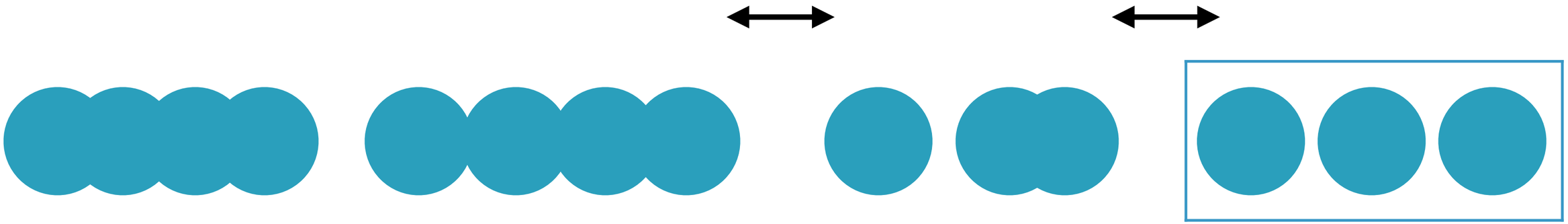


Changing window size

Can be overlapping or non-overlapping

Number of entities remain the same within a window

Session Window



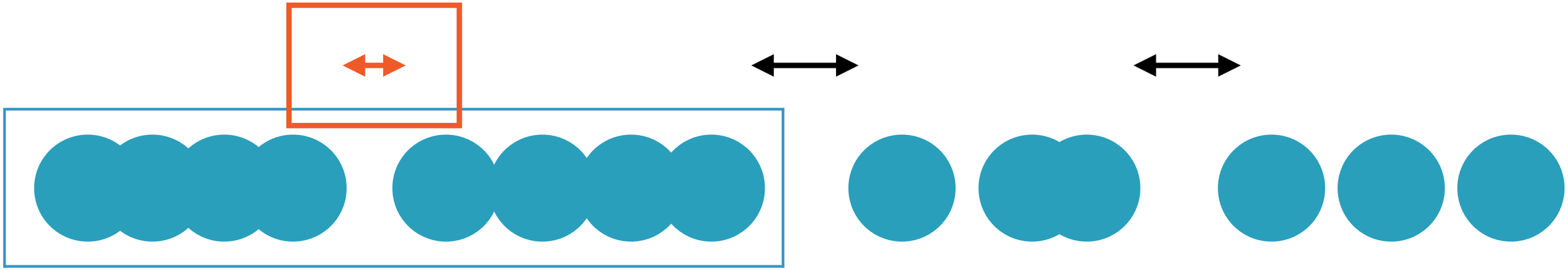
Changing window size based on session data

No overlapping time

Number of entities differ within a window

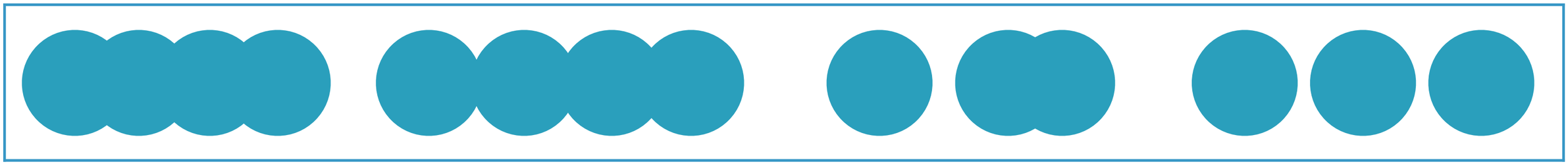
Session gap determines window size

Session Window



This gap is not large enough to start a new window

Global Window



All data in the stream in one window

Event Time and Processing Time

Time-based Windows

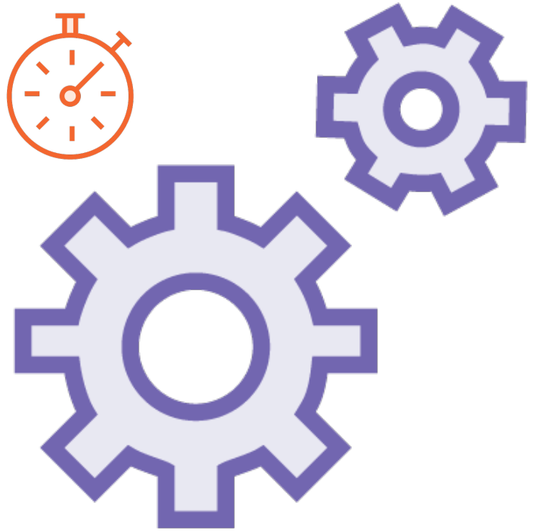
Tumbling and sliding windows consider entities in a fixed interval of time

There are different notions of time that can apply to entities in a stream

Time

Event Time

Ingestion Time



Processing Time

Event Time

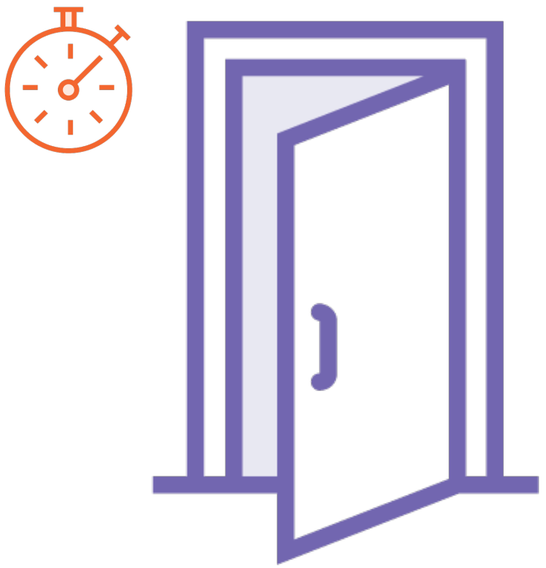
The time at which the event occurred at its original source

- Mobile phone, sensor, website

Usually embedded within records

Gives correct results in case of out of order or late events

Ingestion Time

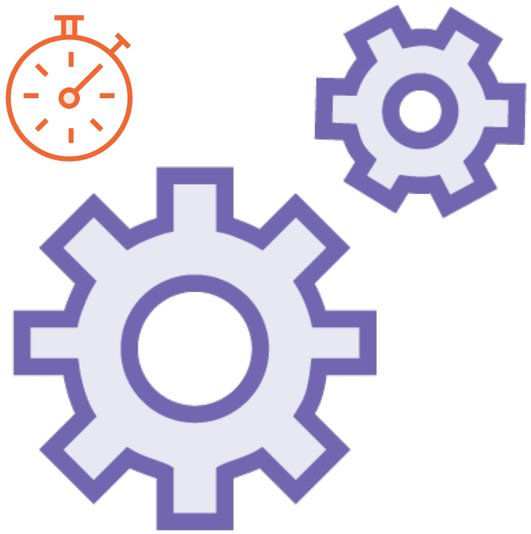


The time at which the event enters the system via a source

Timestamp given by system chronologically after the event time

Cannot handle out of order events

Processing Time



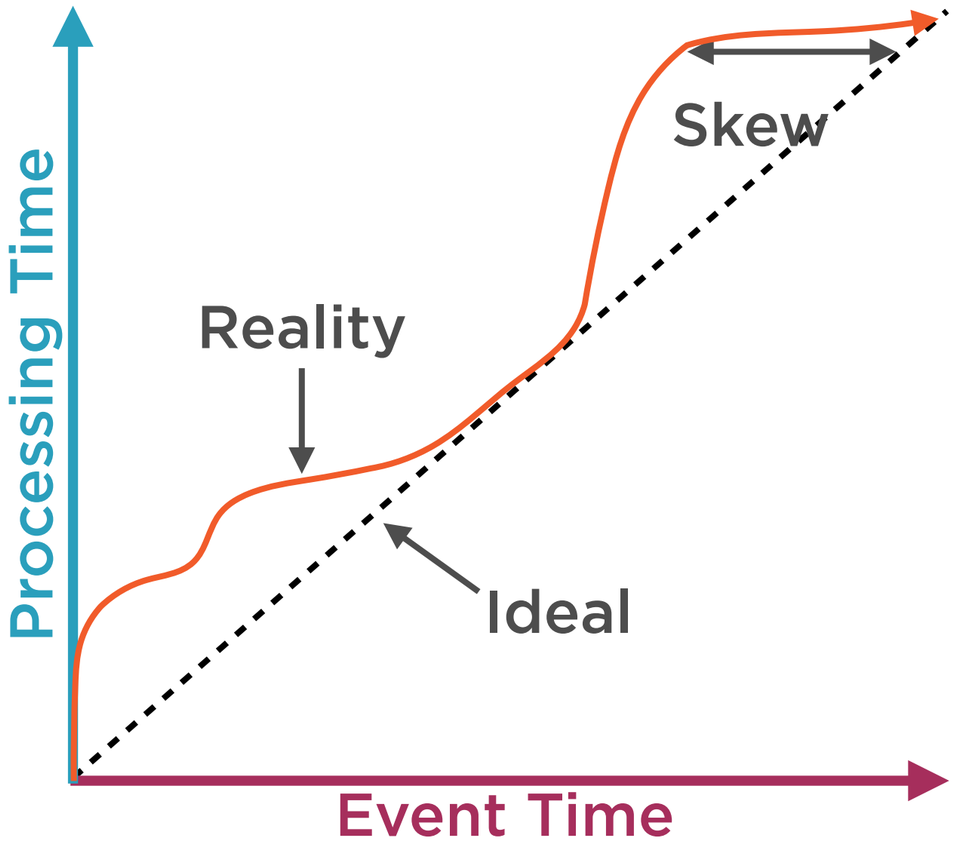
The system time of the machine processing entities

Chronologically after event time and ingestion time

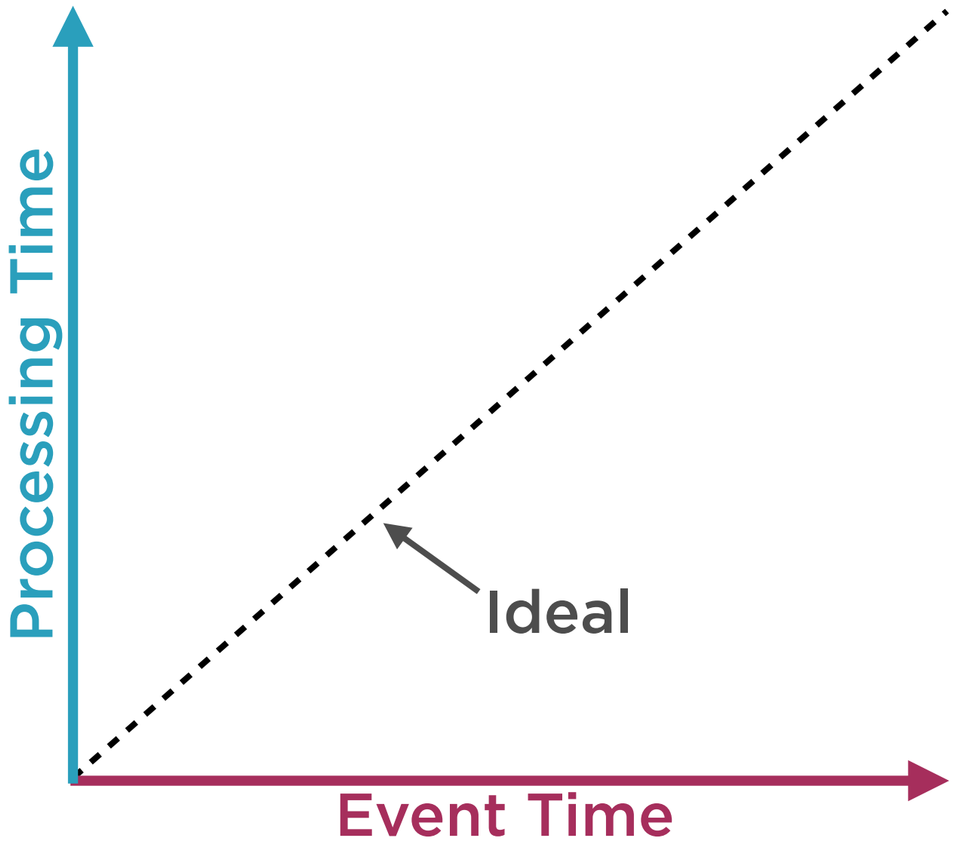
Non-deterministic, depends on when data arrives, how long operations take

Simple, no coordination between streams and processors

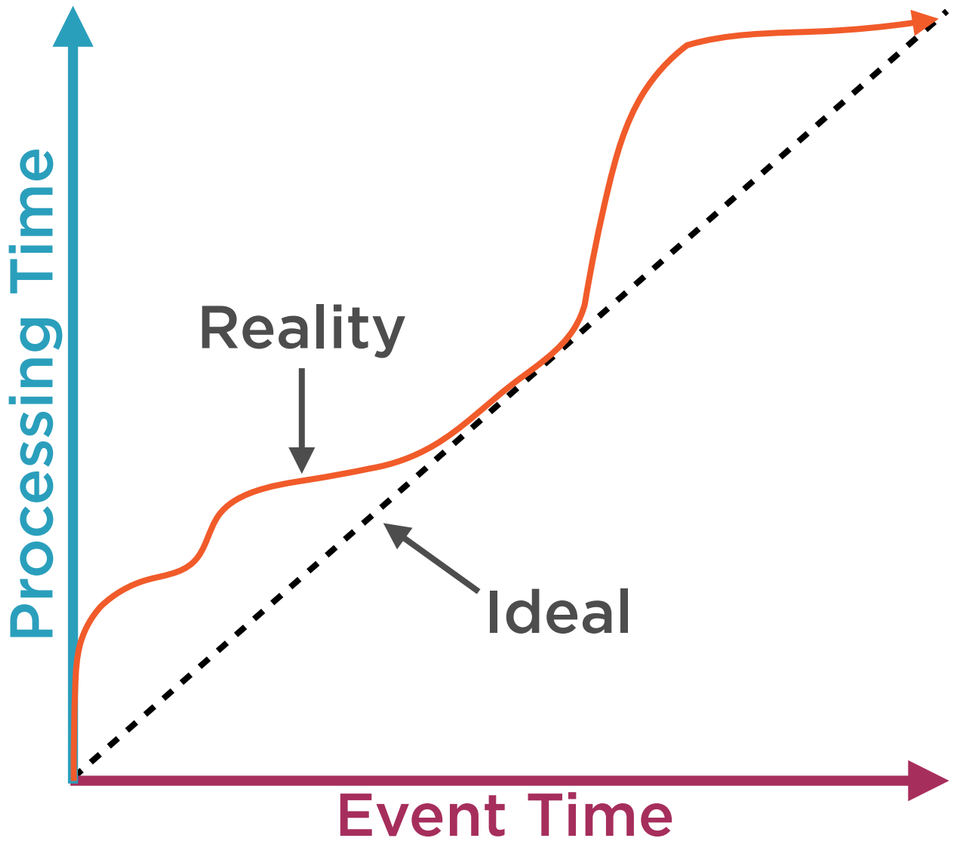
Event Time vs. Processing Time



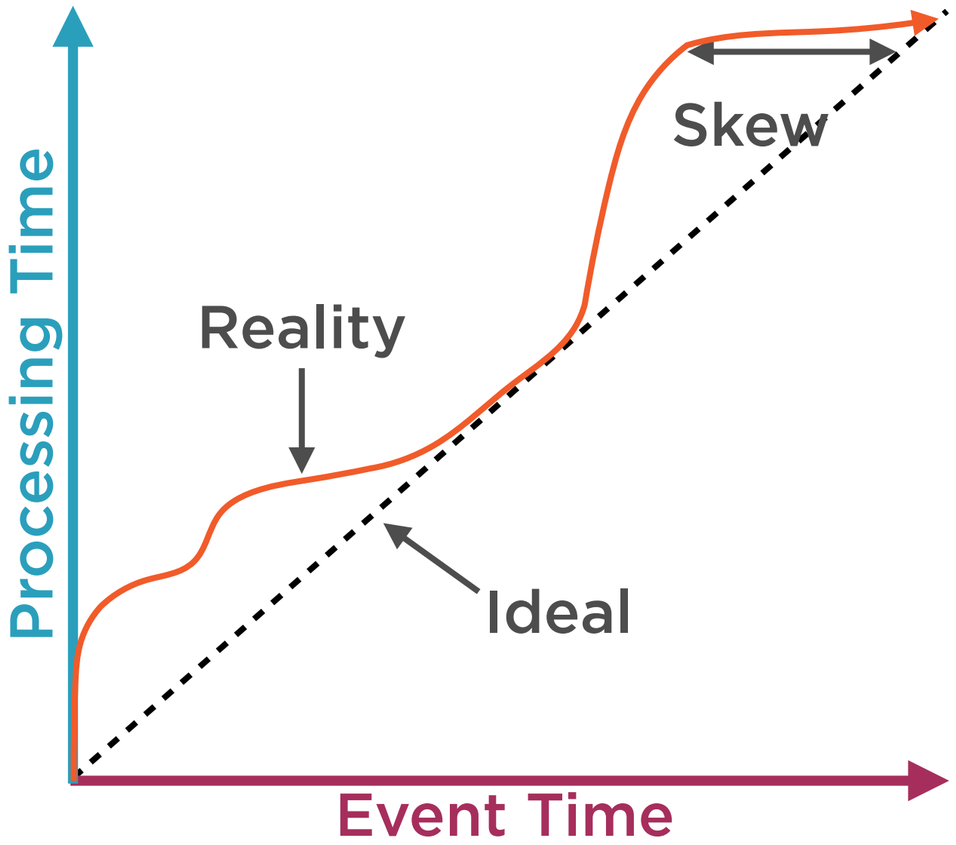
Event Time vs. Processing Time



Event Time vs. Processing Time

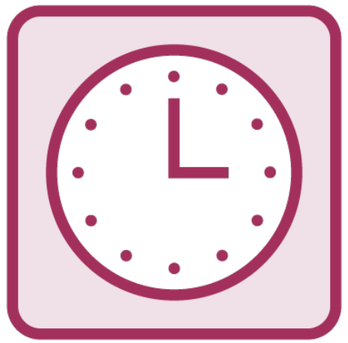


Event Time vs. Processing Time



Watermarks and Late Data

How Late Is Late?



Class At 9 am

Class starts when clock strikes 9

Is 9:01 Late?

Realistically, at least some folks are going to be a minute late

Is 10:10 late?

A student is an hour late - allow in or send back?

How Late Is Late?

The professor “knows” what lateness is reasonable

Students entering within this reasonable lateness are late but OK

Students entering after this reasonable lateness are too late

“Allowed Lateness”

How Late Is Late?

Dealing with excessive lateness

A student is too late

- Option 1: Send back home
- Option 2: Allow in, continue class
- Option 3: Allow in, restart class(!)

Watermarks and Late Data

The system “knows” what lateness is reasonable

Data entering within this reasonable lateness is late but OK

Data entering after this reasonable lateness is too late

Watermarks and Late Data

Watermark

Threshold of allowed lateness (event time)

Data entering within this reasonable lateness is late but OK

Data entering after this reasonable lateness is too late

Watermarks and Late Data

Watermark

Threshold of allowed lateness (event time)

Late Data

Data within watermark is aggregated

Data entering after this reasonable lateness is too late

Watermarks and Late Data

Watermark

Threshold of allowed lateness (event time)

Late Data

Data within watermark is aggregated

Dropped Data

Data outside watermark is dropped

Demo

Performing fixed (tumbling) window operations on input data

Demo

Performing sliding window operations on input data

Demo

Performing session window operations on input data

Demo

Performing global window operations on input data

Demo

Using side inputs in pipelines

Demo

Performing join operations using Beam's join extension library

Demo

Performing joins using side inputs

Demo

Performing joins using CoGroupByKey

Apache Beam Compatibility: Flink and Spark

Apache Flink

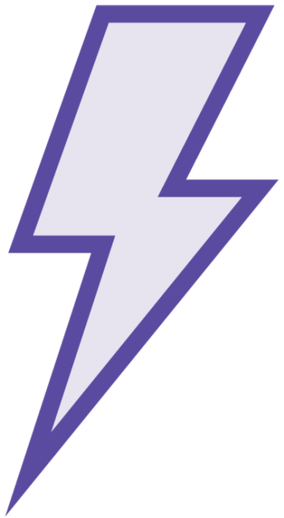
Open-source stream processing framework

Distributed engine written in Java and Scala

Computations and transformations over bounded and unbounded data

Can be used as a execution backend with Apache Beam

Apache Spark 2



Open-source distributed computing framework

Computation engine written in Scala

Leading platform for large-scale batch and stream processing

Can be used as a execution backend with Apache Beam

Different back-end runners have very different capabilities and manner of stream processing

Runner Capabilities

What

Where

When

How

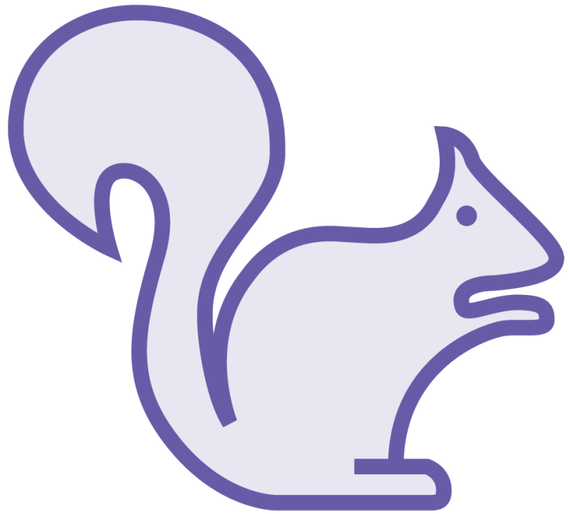
What

What is being computed?

Decides whether the result being computed

- Element-wise
- As an aggregate
- As a composite

Apache Flink

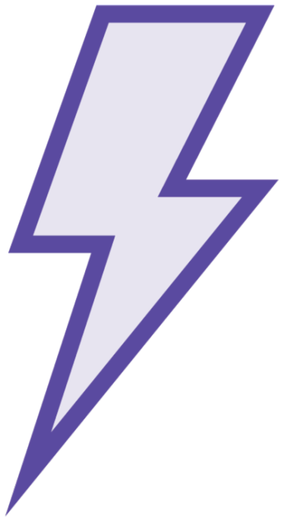


Almost all transforms supported

Partial support for composite transforms

Partial support for side inputs, metrics, and stateful processing

Apache Spark 2



Most Beam operations only partially supported for streaming data

All Beam operations supported for batch data

Where

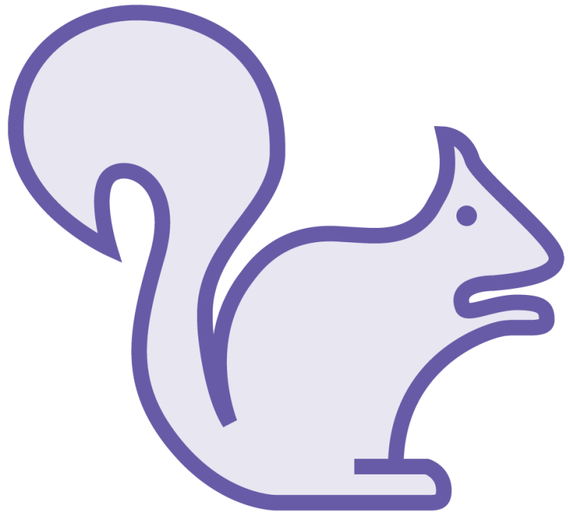
Where in event time is the result being computed?

Decides what type of windowing is being used

- Fixed
- Sliding
- Sessions

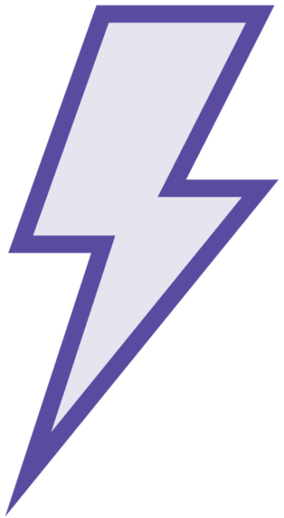
Most important for aggregation operations

Apache Flink



All Beam specified window types supported

Apache Spark 2



All Beam window types only partially supported for streaming data

All Beam window types supported for batch data

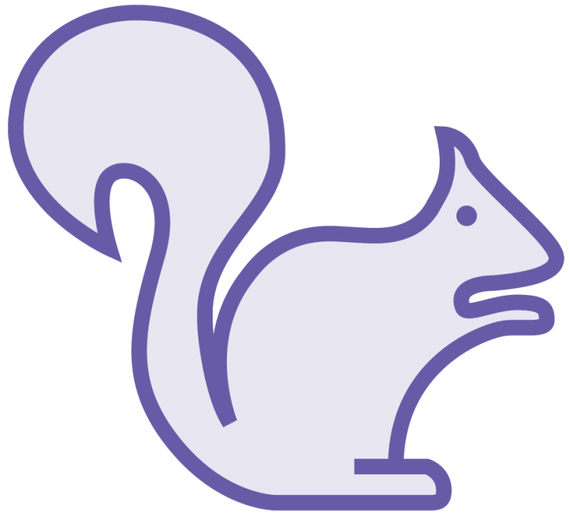
When

When in processing time is the result being computed?

Governs

- Type of Trigger
- Early and late firing

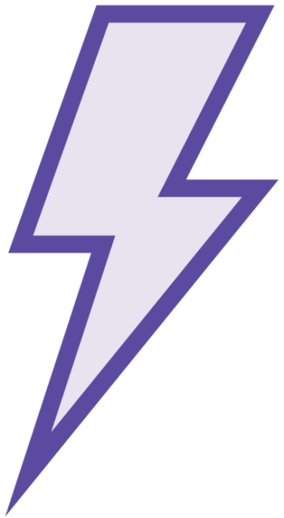
Apache Flink



All triggers, except metadata triggers, supported

Certain timer-related operations not supported

Apache Spark 2



All Beam triggers only partially supported for streaming data

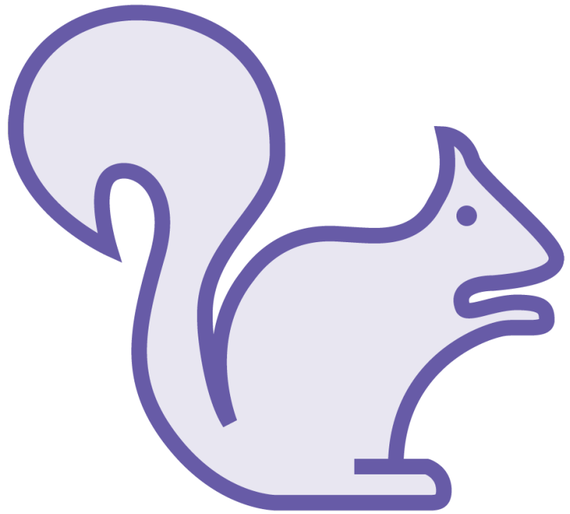
All Beam triggers supported for batch data

How

How do refinements relate?

- How should multiple outputs per window be reconciled?
- Accumulate
- Discard
- Accumulate and retract

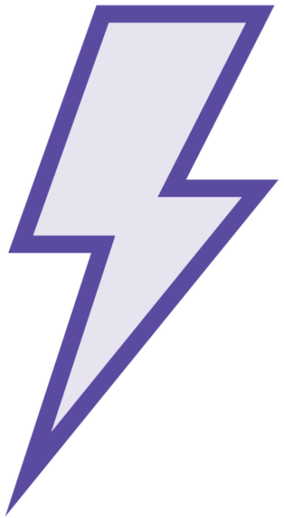
Apache Flink



Accumulate and discard refinements supported

Accumulate and retract refinement not supported

Apache Spark 2



Only discard refinement supported

Accumulate and accumulate and retract not supported

Summary

- Sliding, tumbling, session, and global windows
- Event time vs. processing time
- Watermarks and late arrivals
- Performing join operations on streams
- Using side inputs for processing
- Apache Flink and Apache Spark 2 support for Beam

Up Next:

Performing SQL Queries on Streaming Data