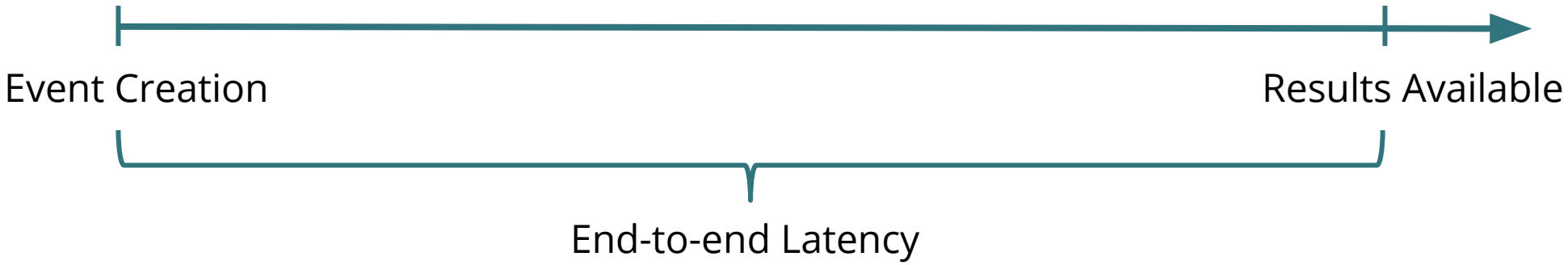
Latency

Alexander Fedulov, Solutions Architect

Defining Latency



- **Latency = Processing Time (of Stage) - Event Time (= Event-Time Lag)**
- Meaning of “Event Time” depends on logic of the application
  - timestamp of the event
  - end time of an event time-window
  - …

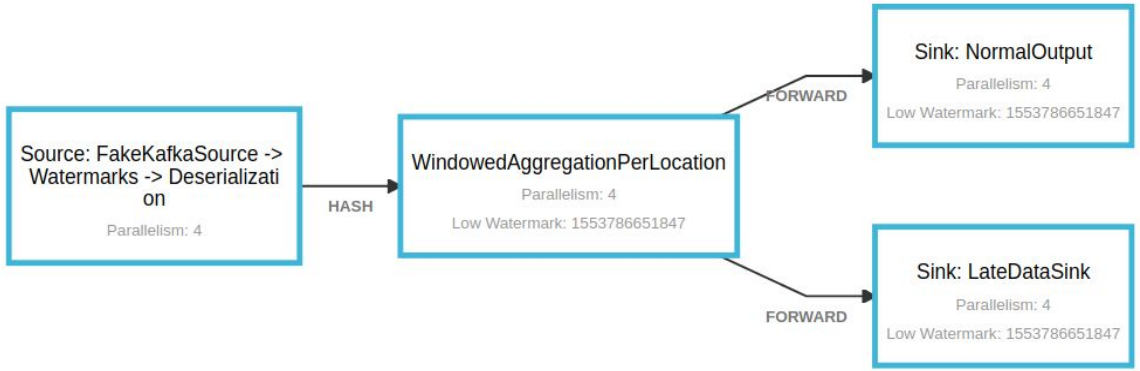
Reasons for Latency

 |

Reasons for Latency

Latency: Processing Time (Publish) - Event Time

Running Example



- Kafka Consumer
- Keyed Event Time Window
- *Exactly-Once* Kafka Producer

Reasons for Latency

Latency: Processing Time (Publish) - Event Time

- Latency accumulated **before Flink application**
  - time between event creation and storage of event in queue
  - time between storage in queue and consumption by application

Reasons for Latency

Latency: Processing Time (Publish) - Event Time

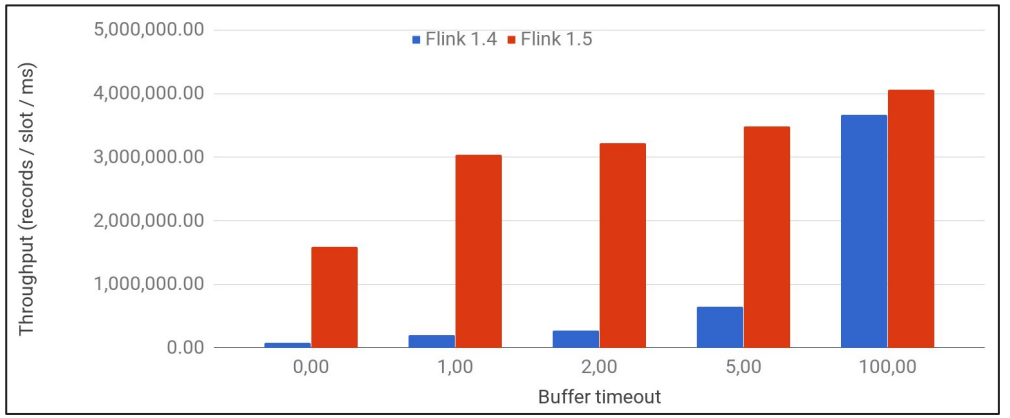
- Latency accumulated **inside Flink application**
  - latency due to event time processing
  - latency due to network (incl. network buffers)
  - latency due to processing delays
  - latency due to transactional sinks
  - latency due to checkpointing

Latency due to Event Time Processing

- Applies to anything based on (event time) timers:
  - windows
  - process functions
- Watermark progresses with `min(all input watermarks)`
  - influenced by any upstream operator
  - allowed out-of-orderness adds to latency
  - watermarking interval matters
- Window / timer fires when watermark exceeds window end / timer

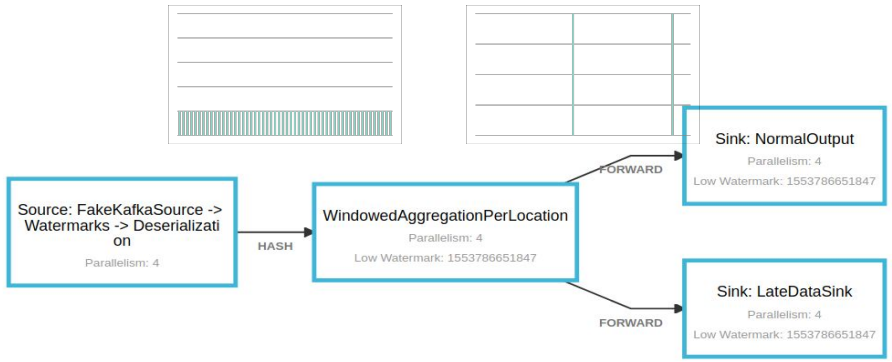
Latency due to Network Delays

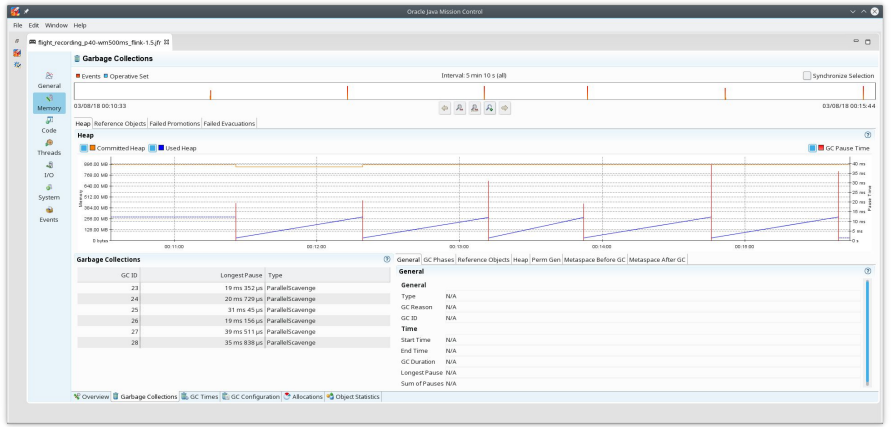
- Every transfer over the network (repartition/rebalance) adds latency
- Flink assembles (serialized) records in buffers for network/local transfer
- Buffers are sent once full or after buffer timeout
- Trade-off between throughput & latency
- `StreamExecutionEnvironment#setBufferTimeout(int)`



Latency due to Processing Delays

- Execution of user & framework code adds latency
- try to mitigate load spikes due to windowing, e.g. by pre-aggregating as much as possible
- garbage collections will cause temporary backpressure & small latency spikes





Latency due to Transactional Sinks

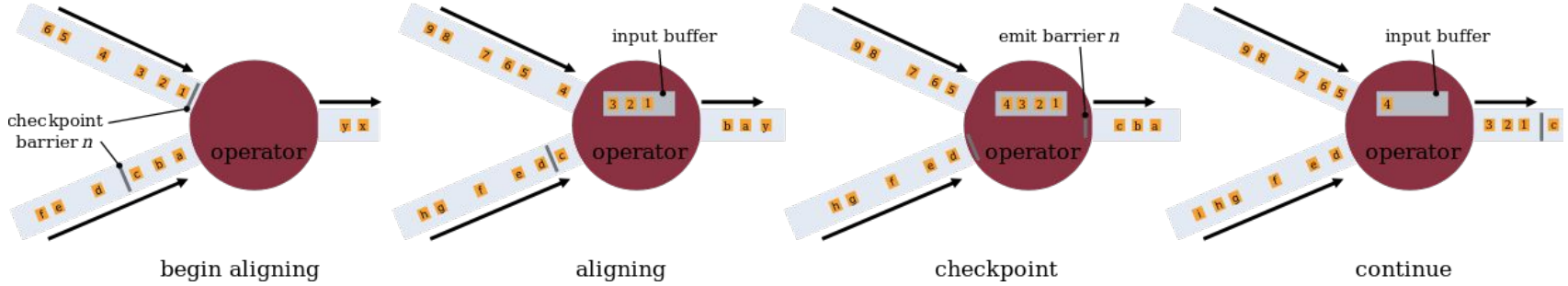
- Lifecycle of a typical Transactional Sink

Phase	**Actions**
onElement	• asynchronously send to sink system
onSnapshot	• flush all records & wait for acknowledgement • create a new transaction for next checkpointing epoch • store transaction metadata in Flink state
onCheckpointComplete	• commit pending transactions & publish data

- Transactional sink adds latency up to the checkpoint interval

Latency due to Checkpointing

- Checkpointing consists of three phases
  - Checkpoint Alignment (synchronous)
  - Synchronous Part (synchronous)
  - Asynchronous Part
- Checkpoint Alignment



- backpressure on blocked channels

Exercises

Troubleshooting Watermarks & Latency Tuning

Exercise 2
After the first exercise, the job is running stable, but no record is entering the sink. Investigate the issue and fix it.

(start from your code from the previous exercise or **`TroubledStreamingJobSolution1`**)

Exercise 3
In the deployment's Flink configuration, set the `state.backend` to `filesystem` and reduce the 99th percentile of the event time lag of the `WindowedAggregationPerLocation` operator. The `eventTimeLag` metric will show the current value.

(start from your code from the previous exercise or **`TroubledStreamingJobSolution2`**)

alex@ververica.com www.ververica.com @VervericaData