

CS 744: DATAFLOW

Shivaram Venkataraman

Spring 2024

ADMINISTRIVIA

Grading In Progress

- Course project proposal
- Assignment 2
- Midterm

MID-SEMESTER FEEDBACK

“...instead of having everyone submit full discussion answers regardless of what group they're in, maybe just have a single check box on the form....”

“I hope there can be more details about the diagrams in discussion and exams....”

....

“... I found midterm exercises were more tricky/challenging ...”

“... hard time in the midterm exam...”

“Its an early morning class”

“The morning timings of the class”

“I want reading groups to talk and understand the paper better.”

“... If we had paper reading groups like Distributed Systems course...”

Applications

Machine Learning

SQL

Streaming

Graph

Computational Engines

Scalable Storage Systems

Resource Management



Datacenter Architecture



DATAFLOW MODEL (?)

MOTIVATION

Streaming Video Provider

- How much to bill each advertiser ?
- Need per-user, per-video viewing sessions
- Handle out of order data

Goals

- Easy to program
- Balance correctness, latency and cost

APPROACH

Separate user API from execution

Decompose queries into

- What is being computed
- Where in time is it computed
- When is it materialized
- How does it relate to earlier results

STREAMING VS. BATCH

Streaming

Batch

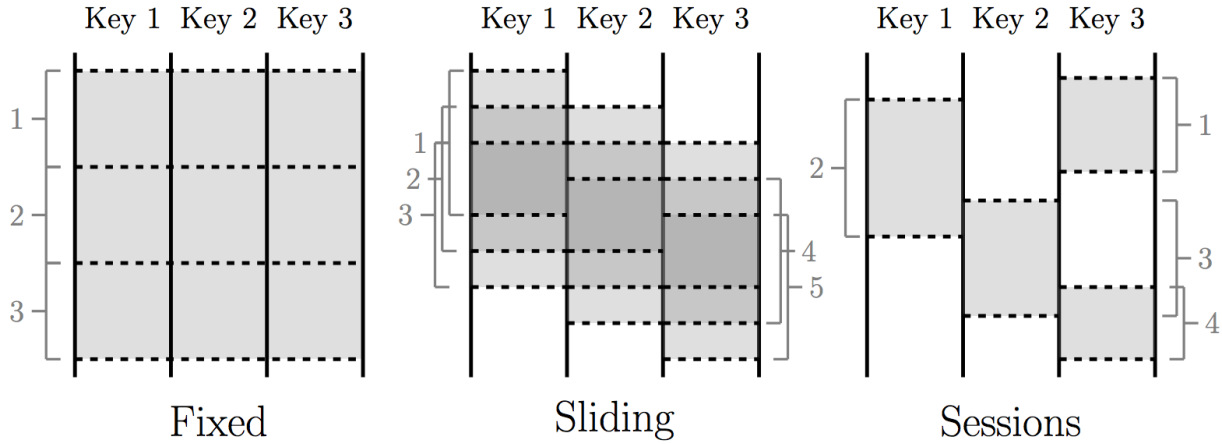


TIMESTAMPS

Event time:

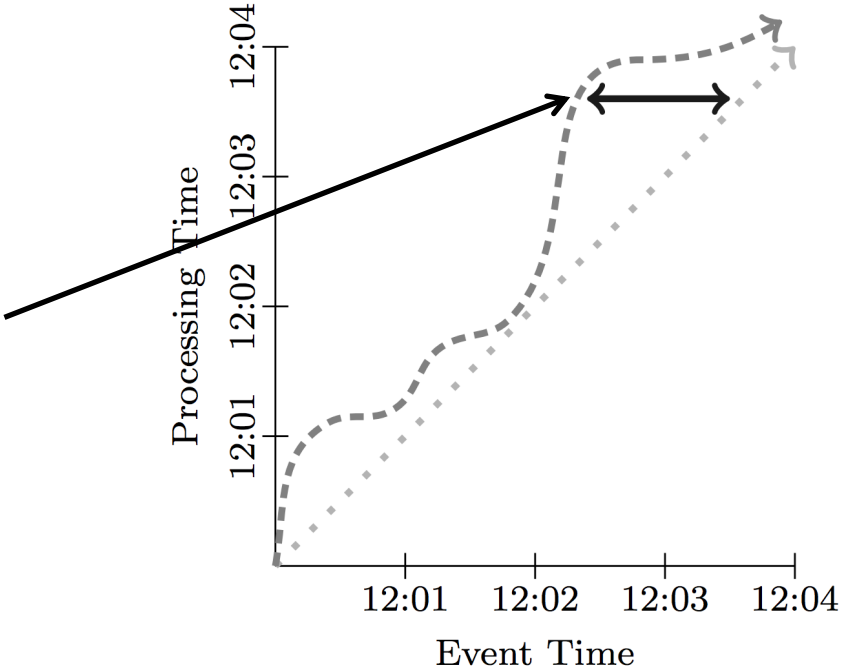
Processing time:




WINDOWING



WATERMARK OR SKEW

System has processed all events up to 12:02:30



- Actual watermark: 
- Ideal watermark: 
- Event Time Skew: 

API

ParDo:

GroupByKey:

Windowing

AssignWindow

MergeWindow

EXAMPLE

$(k_1, v_1, 13:02, [0, \infty))$,
 $(k_2, v_2, 13:14, [0, \infty))$,
 $(k_1, v_3, 13:57, [0, \infty))$,
 $(k_1, v_4, 13:20, [0, \infty))$

↓ *AssignWindows*(
 Sessions(30m))

$(k_1, v_1, 13:02, [13:02, 13:32))$,
 $(k_2, v_2, 13:14, [13:14, 13:44))$,
 $(k_1, v_3, 13:57, [13:57, 14:27))$,
 $(k_1, v_4, 13:20, [13:20, 13:50))$

↓ *DropTimestamps*

$(k_1, v_1, [13:02, 13:32))$,
 $(k_2, v_2, [13:14, 13:44))$,
 $(k_1, v_3, [13:57, 14:27))$,
 $(k_1, v_4, [13:20, 13:50))$

GroupByKey

$(k_1, [(v_1, [13:02, 13:32)),$
 $(v_3, [13:57, 14:27)),$
 $(v_4, [13:20, 13:50))])$,
 $(k_2, [(v_2, [13:14, 13:44))])$

↓ *MergeWindows*(
 Sessions(30m))

$(k_1, [(v_1, [13:02, 13:50)),$
 $(v_3, [13:57, 14:27)),$
 $(v_4, [13:02, 13:50))])$,
 $(k_2, [(v_2, [13:14, 13:44))])$

↓ *GroupAlsoByWindow*

$(k_1, [([v_1, v_4], [13:02, 13:50)),$
 $([v_3], [13:57, 14:27))])$,
 $(k_2, [([v_2], [13:14, 13:44))])$

↓ *ExpandToElements*

$(k_1, [v_1, v_4], 13:50, [13:02, 13:50))$,
 $(k_1, [v_3], 14:27, [13:57, 14:27))$,
 $(k_2, [v_2], 13:44, [13:14, 13:44))$

TRIGGERS AND INCREMENTAL PROCESSING

Windowing: **where** in event time are data grouped

Triggering: **when** in processing time are groups emitted

Strategies

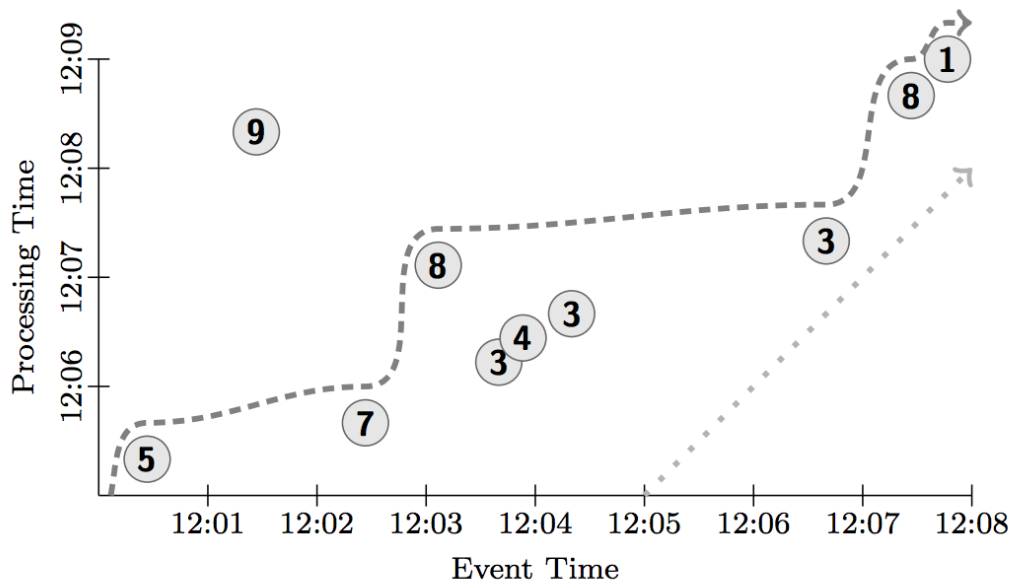
- Discarding

- Accumulating

- Accumulating & Retracting

RUNNING EXAMPLE

```
PCollection<KV<String, Integer>> input = IO.read(...);  
PCollection<KV<String, Integer>> output =  
    input.apply(Sum.integersPerKey());
```

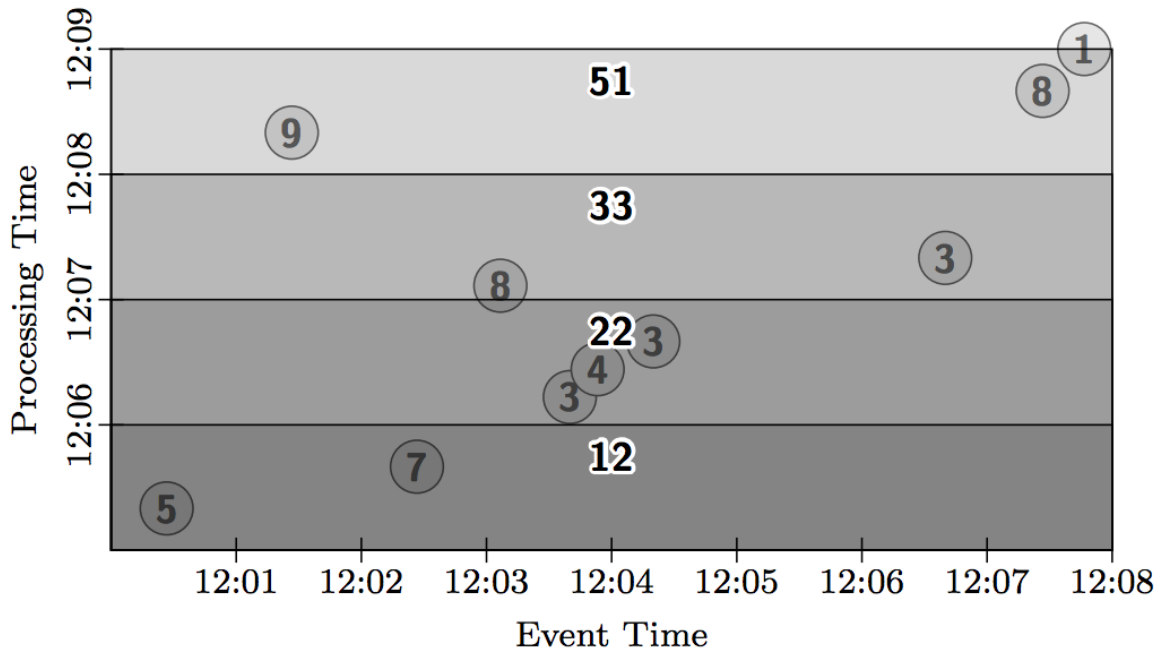


Actual watermark: ----->

Ideal watermark:>

GLOBAL WINDOWS, ACCUMULATE

```
PCollection<KV<String, Integer>> output = input  
  .apply(Window.trigger(Repeat(AtPeriod(1, MINUTE))))  
    .accumulating()  
  .apply(Sum.integersPerKey());
```



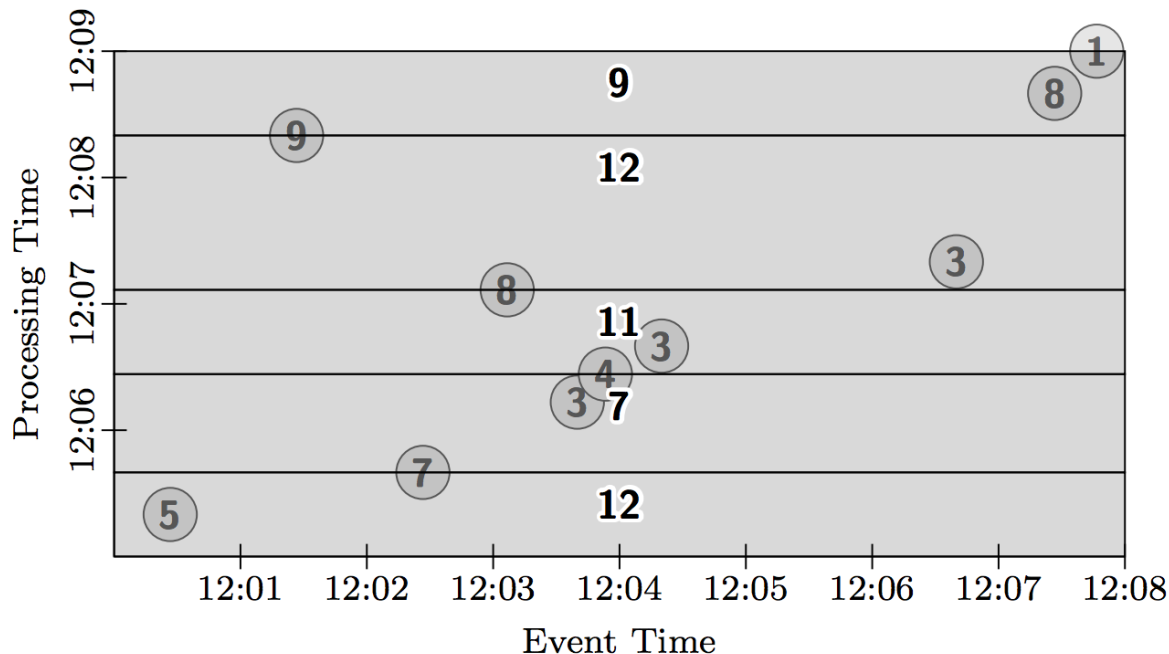
GLOBAL WINDOWS, COUNT, DISCARDING

```
PCollection<KV<String, Integer>> output = input
```

```
  .apply(Window.trigger(Repeat(AtCount(2))))
```

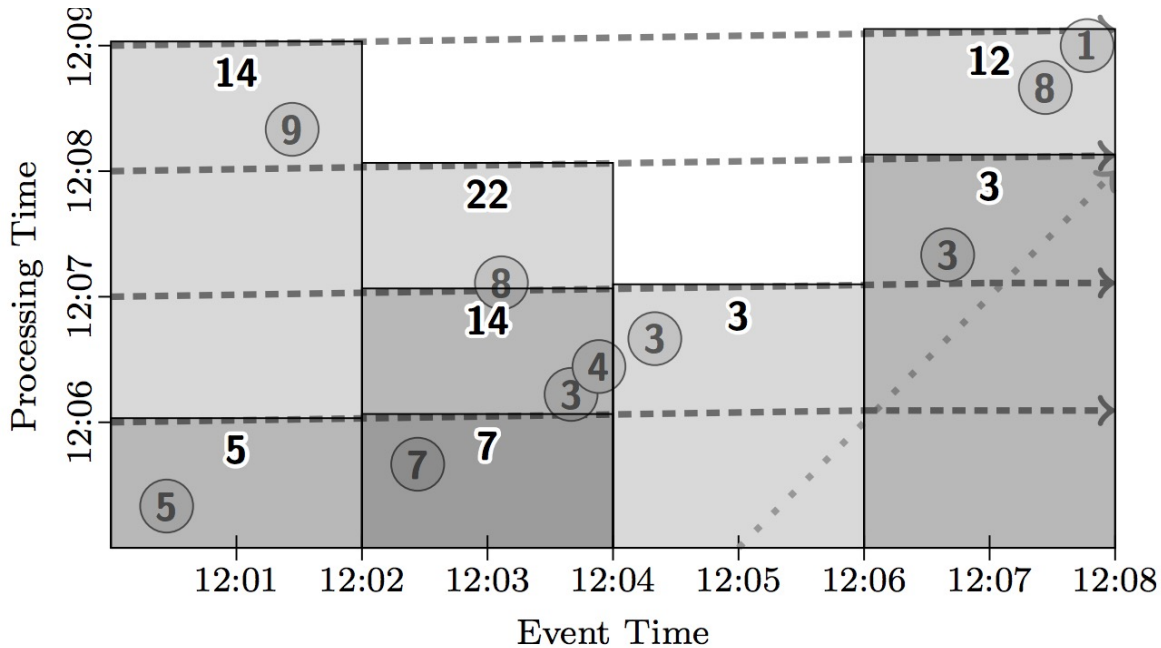
```
    .discarding()
```

```
  .apply(Sum.integersPerKey());
```



FIXED WINDOWS, MICRO BATCH

```
PCollection<KV<String, Integer>> output = input  
    .apply(Window.into(FixedWindows.of(2, MINUTES))  
           .trigger(Repeat(AtWatermark()))  
           .accumulating())
```



SUMMARY/LESSONS

Design for unbounded data: Don't rely on completeness

Be flexible, diverse use cases

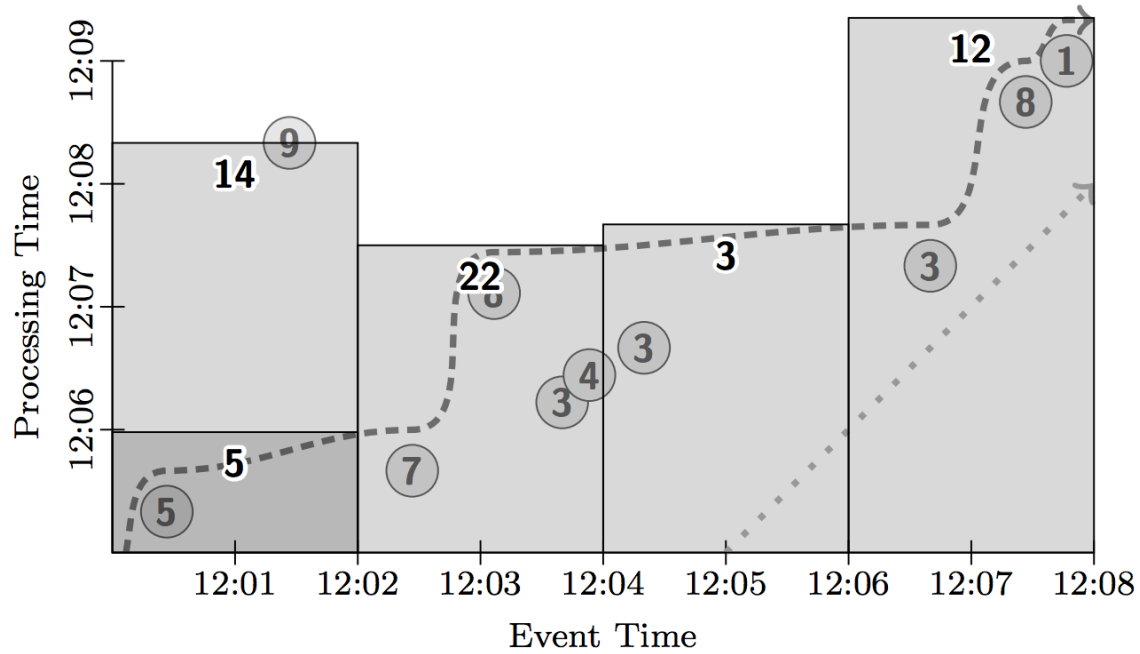
- Billing
- Recommendation
- Anomaly detection

Windowing, Trigger API to simplify programming on unbounded data



DISCUSSION

<https://forms.gle/TB5kz2cH3uYc6rjv6>



Consider you are implementing a micro-batch streaming API on top of Apache Spark. What are some of the bottlenecks/challenges you might have in building such a system?

NEXT STEPS

Next class: Apache Flink