CS 744: DATAFLOW

Shivaram Venkataraman
Fall 2022
Grading In Progress
  - Assignment 2
  - Course project proposal

About the Midterm
MID-SEMESTER FEEDBACK

More frequent change in paper reading group could be beneficial.
It was hard to find time for paper discussions, given the random assignments and fixed teammates
I would have liked to have chosen my reading group. I find it somewhat difficult to organize
...

I would like if more time was given for exam
Exams should be longer in time duration, very less time to actually think deep
...

It would've been nice if we had to … list any questions we had about the paper in the paper review form.

More time for in-class discussions.
Concentrating for 1.25 hours continuously is hard. A break in the middle for discussion?

maybe in-class quizzes??
Optional from next class (Nov 3rd)

Re-shuffle groups to have new “suggested” groups

You can do any of
- Use (part of) suggested group
- Discuss with your own group
- Read on your own!
Scalable Storage Systems

Resource Management

Computational Engines

Datacenter Architecture

Applications

Machine Learning  SQL  Streaming  Graph
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
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

Fixed

Sliding

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

ParDo:

GroupByKey:

Windowing
  AssignWindow

MergeWindow
Example

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

\[\text{AssignWindows(Sessions(30m))}\]
\(\uparrow\)
\[(k_1, v_1, 13:02, [13:02, 13:32)), \]
\[(k_2, v_2, 13:14, [13:14, 13:44)), \]
\[(k_1, v_3, 13:57, [13:57, 14:27)), \]
\[(k_1, v_4, 13:20, [13:20, 13:50))\]

\[\text{DropTimestamps}\]
\(\downarrow\)
\[(k_1, v_1, 13:02, 13:32)), \]
\[(k_2, v_2, 13:14, 13:44)), \]
\[(k_1, v_3, 13:57, 14:27)), \]
\[(k_1, v_4, 13:20, 13:50))\]

\[\text{GroupByKey}\]

\[(k_1, [(v_1, 13:02, 13:32)), \]
\[(v_3, [13:57, 14:27)), \]
\[(v_4, [13:20, 13:50))\]

\[\text{MergeWindows(Sessions(30m))}\]
\(\downarrow\)
\[(k_1, [(v_1, 13:02, 13:50)), \]
\[(v_3, [13:57, 14:27)), \]
\[(v_4, [13:02, 13:50))\]

\[\text{GroupAlsoByWindow}\]
\(\downarrow\)
\[(k_1, [[(v_1, v_4), 13:02, 13:50))], \]
\[([(v_3), [13:57, 14:27))\]
\[(k_2, [[(v_2), [13:14, 13:44))\]

\[\text{ExpandToElements}\]
\(\downarrow\)
\[(k_1, [v_1, v_4], 13:50, [13:02, 13:50)), \]
\[(k_1, [v_3], 14:27, [13:57, 14:27))\]
\[(k_2, [v_2], 13:44, [13:14, 13:44))\]
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());
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());
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/gUKw3ZP36JjBciABA
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: Flink