ADMINISTRIVIA

- Midterm grades this week
- Course Projects feedback
CONTINUOUS OPERATOR MODEL

- Long-lived operators
- Mutable State
- Distributed Checkpoints for Fault Recovery
- Stragglers?

Diagram:
- Driver → Control Message
- Task → Network Transfer

Tools: Flink, Naiad
CONTINUOUS OPERATORS

Diagram showing the flow of input through two nodes (node 1 and node 2) with mutable state and synchronization between nodes. Connections indicate the flow of data from input, through primaries, to replicas.
SPARK STREAMING: GOALS

1. Scalability to hundreds of nodes
2. Minimal cost beyond base processing (no replication)
3. Second-scale latency
4. Second-scale recovery from faults and stragglers
DISCRETIZED STREAMS (DSTREAMS)
pageViews = readStream(http://..., "1s")

ones = pageViews.map(
    event => (event.url, 1))

counts = ones.runningReduce(
    (a, b) => a + b)

Transformations

Stateless: map, reduce, groupBy, join

Stateful:

  window("5s") → RDDs with data in [0,5), [1,6), [2,7)

  reduceByWindow("5s", (a, b) => a + b)
SLIDING WINDOW

Add previous 5 each time

(a) Associative only
(b) Associative & invertible
Tracking State: streams of (Key, Event) → (Key, State)

events.track(
    (key, ev) => 1,

    (key, st, ev) => ev == Exit ? null : 1,

    "30s")
OPTIMIZATIONS

Timestep Pipelining
   No barrier across timesteps unless needed
   Tasks from the next timestep scheduled before current finishes

Checkpointing
   Async I/O, as RDDs are immutable
   Forget lineage after checkpoint
Worker failure
- Need to recomputate state RDDs stored on worker
- Re-execute tasks running on the worker

Strategy
- Run all independent recovery tasks in parallel
- Parallelism from partitions \textit{in timestep} and \textit{across timesteps}
```scala
pageViews = readStream(http://..., "1s")

ones = pageViews.map(
    event => (event.url, 1))

counts = ones.runningReduce((a, b) => a + b)
```
Straggler Mitigation

Use speculative execution

Task runs more than 1.4x longer than median task → straggler

Master Recovery

- At each timestep, save graph of DStreams and Scala function objects
- Workers connect to a new master and report their RDD partitions
- Note: No problem if a given RDD is computed twice (determinism).
SUMMARY

Micro-batches: New approach to stream processing

Simplifies fault tolerance, straggler mitigation

Unifying batch, streaming analytics
DISCUSSION

https://forms.gle/4Xbu9y9KTW5qph8H8
If the latency bound was made to 100ms, how do you think the above figure would change? What could be the reasons for it?
Consider the pros and cons of approaches in Naiad vs Spark Streaming. What application properties would you use to decide which system to choose?
NEXT STEPS

Next class: Graph processing!
Midterm grades ASAP!