dotnotes

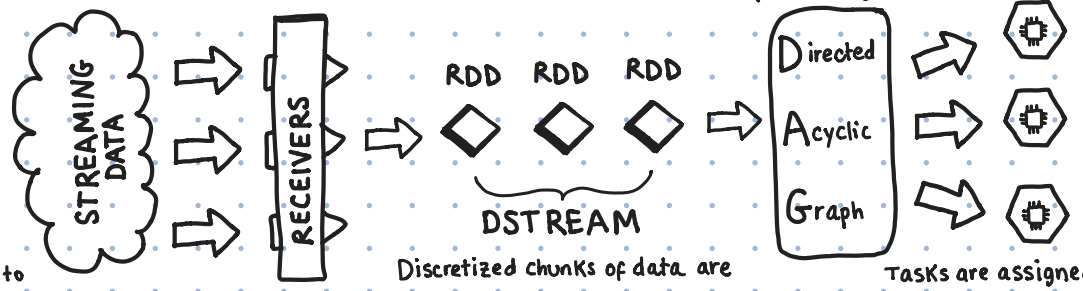
STREAMING BIG DATA

• POPULAR OPTIONS FOR INGESTING LIVE DATA STREAMS •

SPARK STREAMING

A framework for scalable, fault-tolerant processin of live data streams.

THE SETUP:

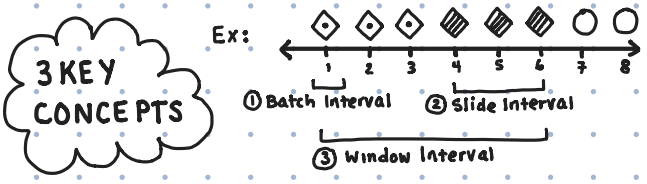


Discretized chunks of data are created at a fixed time interval.

Tasks are assigned to worker nodes by the DAG scheduler.

WINDOW Operations

Spark Streaming enables you to perform computations over a sliding "window" of data:



Operations performed on a DSTREAM are continuously applied to new microbatches of data as they are received.

① BATCH INTERVAL:
Sampling frequency - the size of each batch (here: 1 second)

② SLIDE INTERVAL:
Computational frequency - how often are calculations refreshed? (here: 3 seconds)

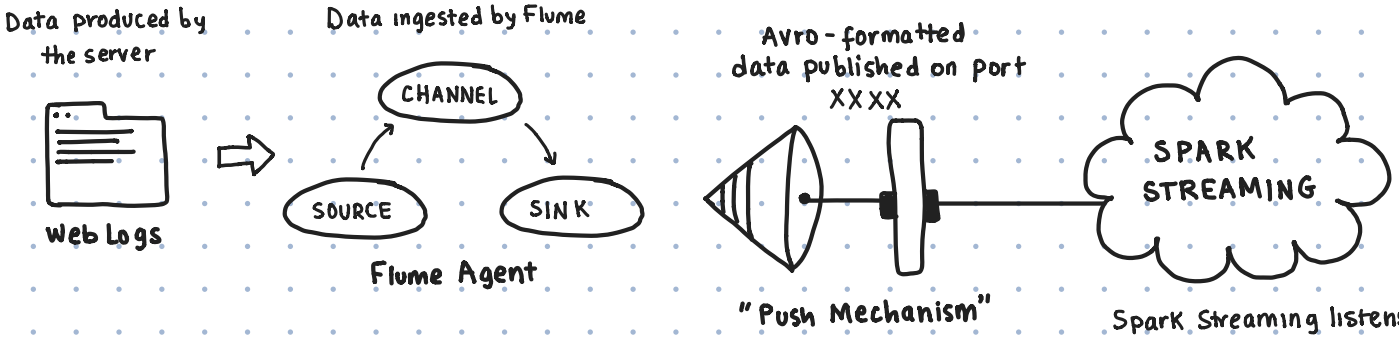
③ WINDOW INTERVAL:
Total history size for each calculation (here: 6 seconds)

Real World Example:
Average click through rate for the last hour (window interval), updated every 5 minutes (slide interval) with data collected every 30 seconds (batch interval).

```
>> SSC = StreamingContext(sc, 30)
>> result = data.reduceByKeyAndWindow(
       func, func, 3600, 300)
```

Batch — Spark Streaming Context — window — slide

EXAMPLE ARCHITECTURE:

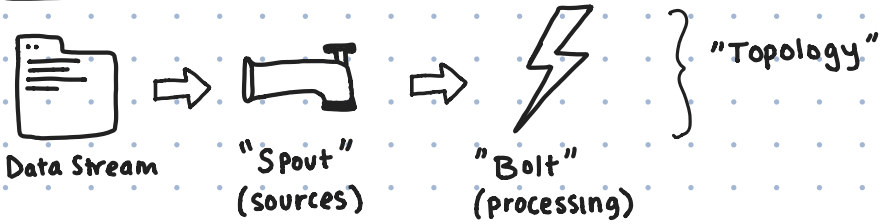


Spark Streaming listens on port XXXX & performs windowing operations on incoming minibatches.

POPULAR ALTERNATIVES TO SPARK:

STORM

Networks of Spouts & bolts => graph of computation



☑ Realtime processing performed on events, not mini batches

FLINK

The youngest technology in the streaming arena.

☑ Real-time processing.

SUPPORTS:

- Stream processing
- batch processing
- Machine learning
- graph analysis