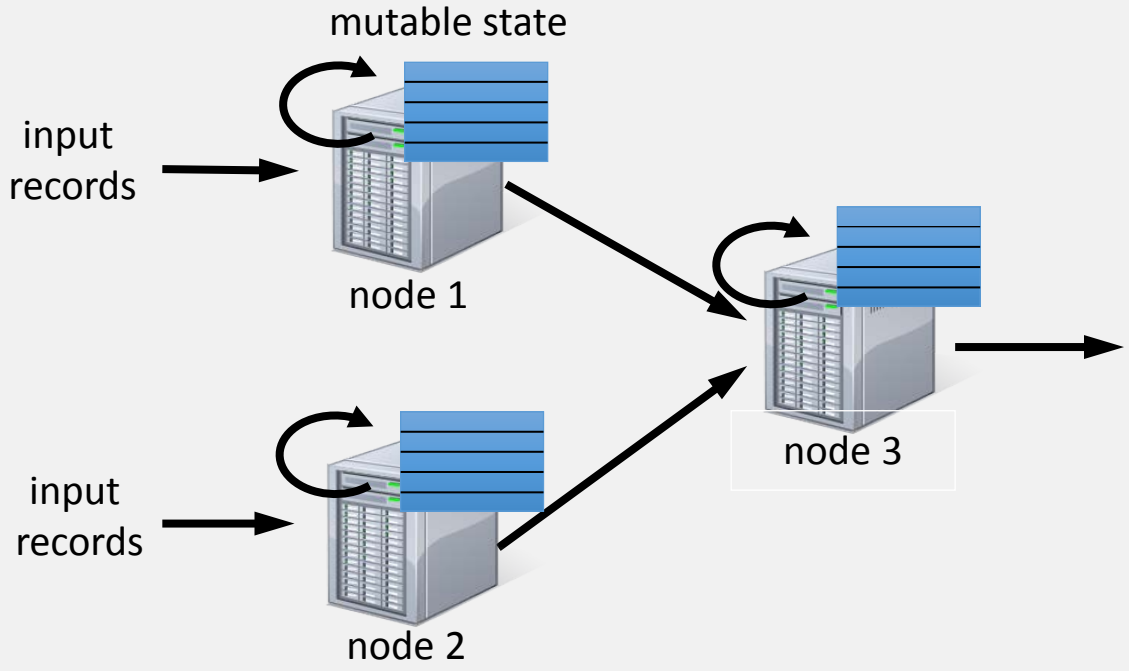
CLOUD COMPUTING APPLICATIONS

Spark Streaming

Roy Campbell & Reza Farivar

Stateful Stream Processing

- Traditional streaming systems have a record-at-a-time processing model
- Each node has mutable state
- For each record, update state and send new records
- State is lost if node dies!
  - Lambda Architecture
- Making stateful stream processing be fault-tolerant is challenging



Existing Streaming Systems

- Storm
  - Replays record if not processed by a node
  - Processes each record at least once
  - May update mutable state twice!
  - Mutable state can be lost due to failure!

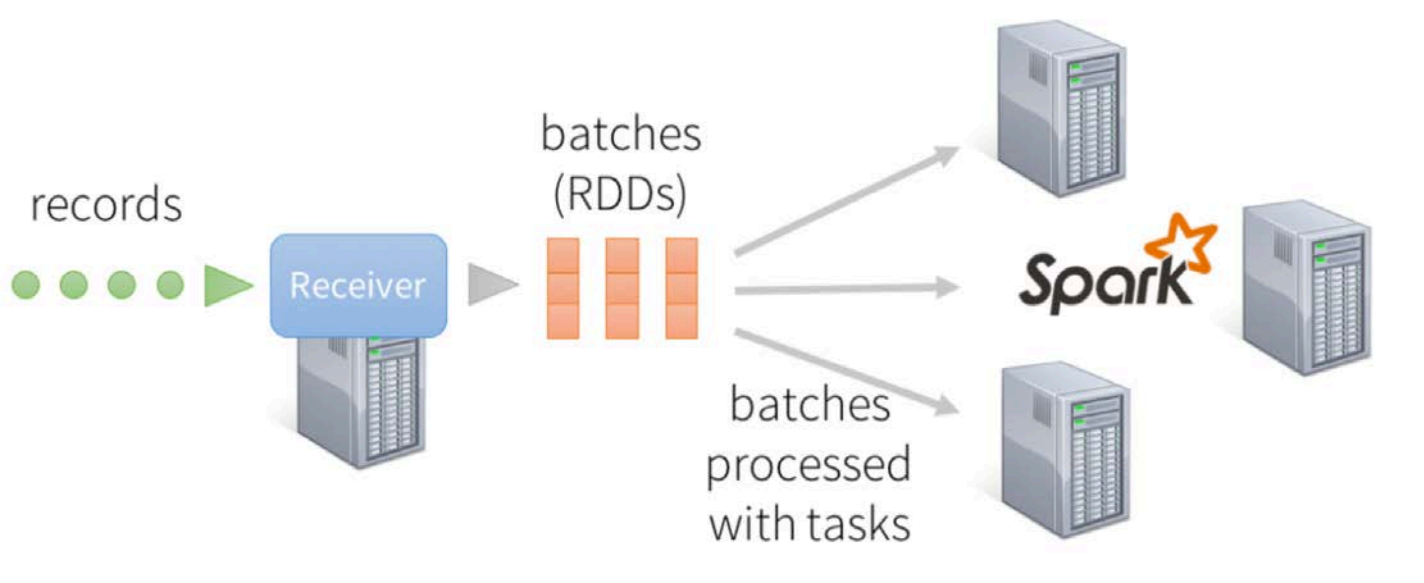
- Trident – Use transactions to update state
  - Processes each record exactly once
  - Per state transaction to external database is slow

Spark

- Spark was a project out of Berkeley from 2010
- Has become very popular
- Most contributed open source project in big-data domain
- RDD: Resilient Distributed Data Set

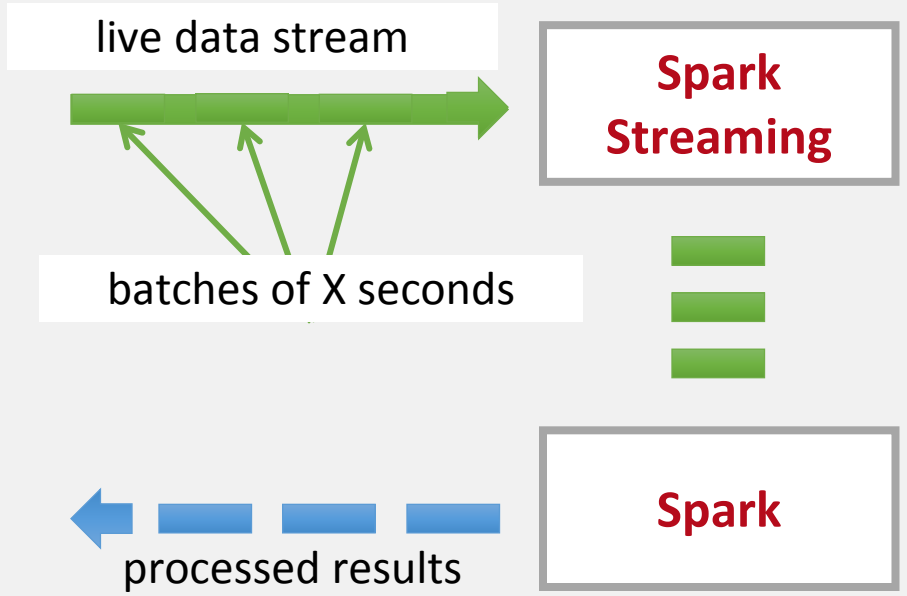
Spark Streaming

- Window a bit of data
- Run a batch
- Repeat



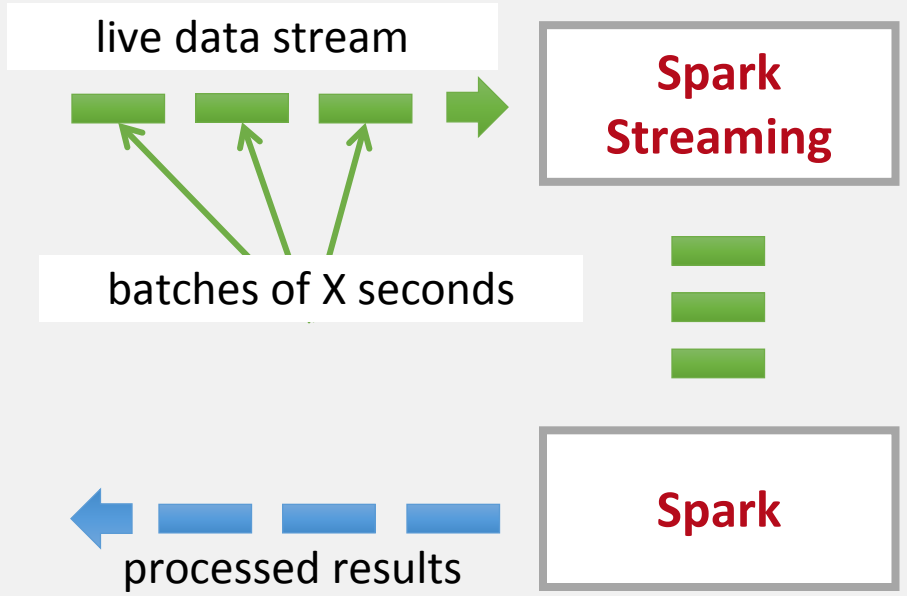
Discretized Stream Processing

- Chop up the live stream into batches of X seconds
- Spark treats each batch of data as RDDs and processes them using RDD operations
- Finally, the processed results of the RDD operations are returned in batches



Discretized Stream Processing

- Batch sizes as low as ½ second, latency of about 1 second
- Potential for combining batch processing and streaming processing in the same system



Spark Streaming example

```
sc = SparkContext("local[2]", "NetworkWordCount")
ssc = StreamingContext(sc, 1)
lines = ssc.socketTextStream("localhost", 9999)
# Split each line into words
words = l
# Count each word in each batch
pairs = words.map(lambda word: (word, 1))
wordCounts = pairs.reduceByKey(lambda x, y: x + y)
# Print the first ten elements of each RDD generated in this DStream to the console
wordCounts.pprint()ines.flatMap(lambda line: line.split(" "))
```

DStream Input Sources

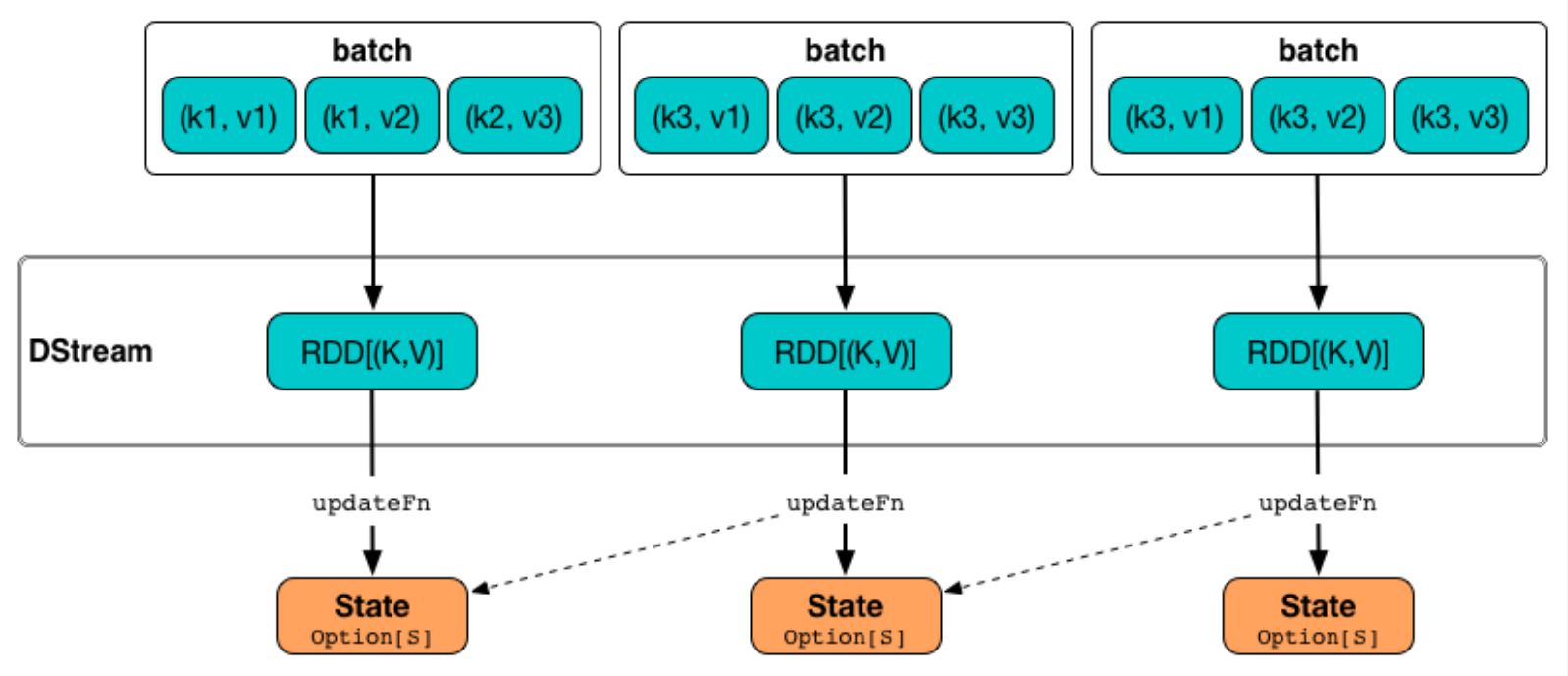
- Out of the box
  - Kafka
  - HDFS
  - Flume
  - Akka Actors
  - Raw TCP sockets

- Very easy to write a *receiver* for your own data source

Arbitrary Stateful Computations

- updateStateByKey
  - maintain arbitrary state while continuously updating it with new information
- How to use
  - Define the state - The state can be an arbitrary data type
  - Define the state update function - Specify with a function how to update the state using the previous state and the new values from an input stream
- state update function applied in every batch for all existing keys

Arbitrary Stateful Computations



Spark ML, Graph, etc.

- Advantage of Spark Streaming:
  - Rich ecosystem of big data tools
  - Spark SQL
  - Spark ML
  - Spark GraphX
  - SparkR
- Disadvantage:
  - Not really streaming