A (Probably not) Project Proposal: Spark Streaming vs Apache Storm for Real-time Event Detection

Niall Egan

November 2019

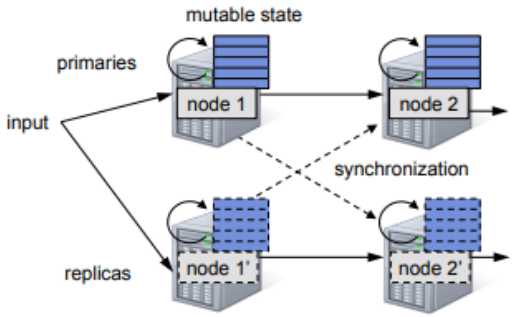
Streaming Dataflow

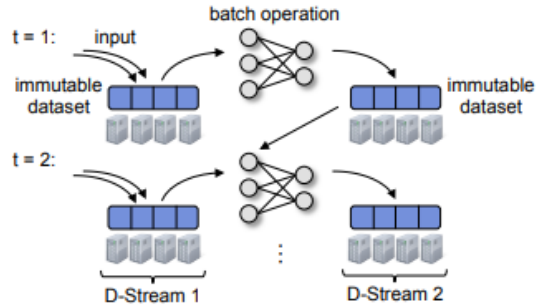
- Dataflow systems we've seen so far (e.g. MapReduce, Spark) are batch-processing systems
- Optimised for *throughput*, not *latency*

Spark Streaming

- Spark is a batch based system, based on RDDs: collections of objects spread across cluster
- Re-build on failure through lineage graph
- In memory RDDs faster than Hadoop
- How to get lower latencies?
- Micro-batching, exposed as D-Streams

Apache Storm





- Apache Storm is a streaming service from the ground up
- Consists of:
  - Streams, unbounded sequence of tuples
  - Spouts (sources of streams)
  - Bolts (processes streams)
  - Topologies

Proposed Application Comparison

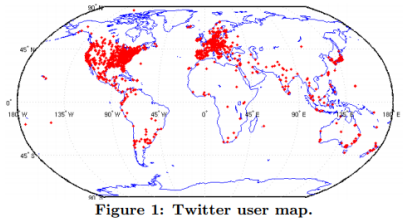


Figure 1: Twitter user map.

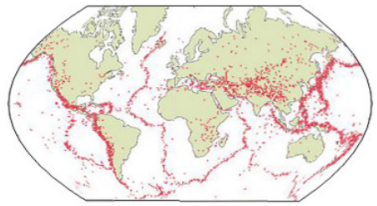


Figure 2: Earthquake map.

- Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors (Sakaki et al.)
- First step: tweet classification. Use SVM to classify tweets as positive or negatively relating to the target event. Have to avoid tweets such as 'The earthquake yesterday was scary'.
- Second step: tweet as a sensory value. Regard twitter user as sensor with associated time and place. Then use Kalman filters to predict where the earthquake is happening.
- Put this onto Spark and Storm to do real-time, large-scale tweet classification and Kalman filters

Things to Compare On

- Latency (Storm should win)
- Memory usage
- Fault recovery times
- Scalability to number of nodes

Project Plan

1. Think of a better idea
2. Write a new project plan