Application: Spark streaming kmeans
Scenario:

- Application is executed on wrangler
- 16 spark consumers
- 1 broker

Application UI:

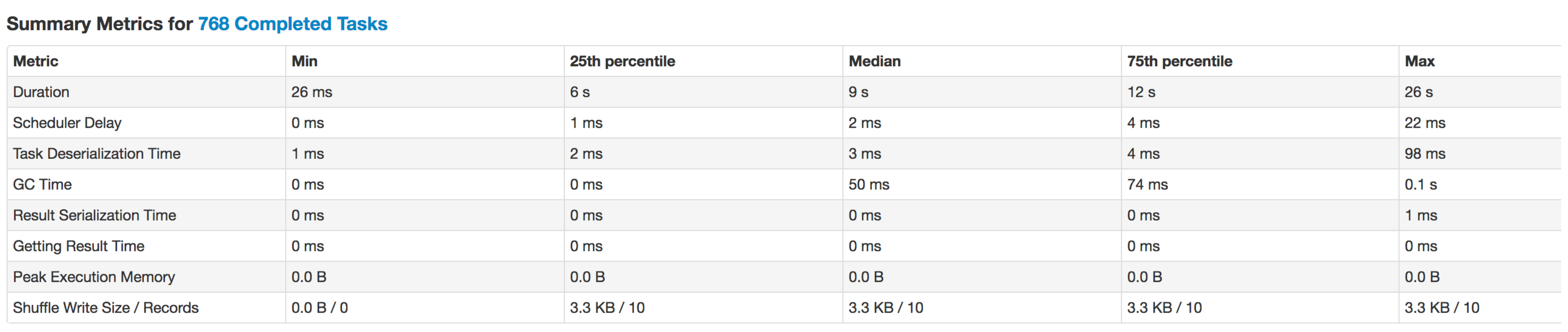
Here is the
http://129.114.62.161:18080/history/app-20180103041023-0000/jobs/

Algorithm:

- Create Dsteam
- Count Records:
  - Add.collect().   (Job1)
- Preprocess:
  - Map (job 2) > flatMap (job 3) -> Map (job 4)
- Model Update (for each rdd):
  - rdd.count(). (Job5)
  - Model.update

Job 34:
Kafka Stream: max: 21 seconds, avg: 6 seconds

Job 35:
Map Partitions -> Map max: 27 seconds, Shuffle write



Summary Metrics for 768 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	26 ms	6 s	9 s	12 s	26 s
Scheduler Delay	0 ms	1 ms	2 ms	4 ms	22 ms
Task Deserialization Time	1 ms	2 ms	3 ms	4 ms	98 ms
GC Time	0 ms	0 ms	50 ms	74 ms	0.1 s
Result Serialization Time	0 ms	0 ms	0 ms	0 ms	1 ms
Getting Result Time	0 ms	0 ms	0 ms	0 ms	0 ms
Peak Execution Memory	0.0 B	0.0 B	0.0 B	0.0 B	0.0 B
Shuffle Write Size / Records	0.0 B / 0	3.3 KB / 10	3.3 KB / 10	3.3 KB / 10	3.3 KB / 10

The average duration is 9 seconds but he maximum is 26. For some reason we have stragglers here.

-> aggregatebyKey (0.1 seconds) Shuffle Read

Job 36:
Kafka Stream: Duration: 3 sec Shuffle Write (45 KB)

Duration: 0.1 sec Shuffle Read(45KB)

Job 37:
Kafka stream: 0.1 seconds

What each job does is not clear to me though.

Jobs called:

1. Java_gateway.py: 2230
2. PythonRDD.scala: 446
3. PythonRDD.scala: 446
4. Java_gateway.py: 2230
5. Collect StreamingKmeans.scala:93. (aggregatebyKey.collect())