

Optimized Set Similarity Join

PROject 3 – COMP9313

Lei Luo (z5124677) | Big Data Management | Semester 2, 2018

# RS Join Optimization

In this project, I implemented the algorithm mainly base on the paper “Efficient Parallel Set-Similarity Joins Using MapReduce” by Spark Scala.

There is neither advanced language syntax in Scala nor low-level optimization on parallel and distributed computing. The key point from my view point is simply is to try not use useless duplicate operations like sort on an ordered list or distinct an array without duplicate element.

There are several general optimization tricks.

* Use Hash Map to quickly map from token to its frequency while sorting tokens in each line;
* Try to use RDD operations as much as possible because it contains optimizations on distributed systems;
* Avoid using groupByKey method since it requires shuffling.

## performance

In benchmark session, I tried three different node numbers 2, 3, 4 to see the power of distributed system. See the screenshot below:

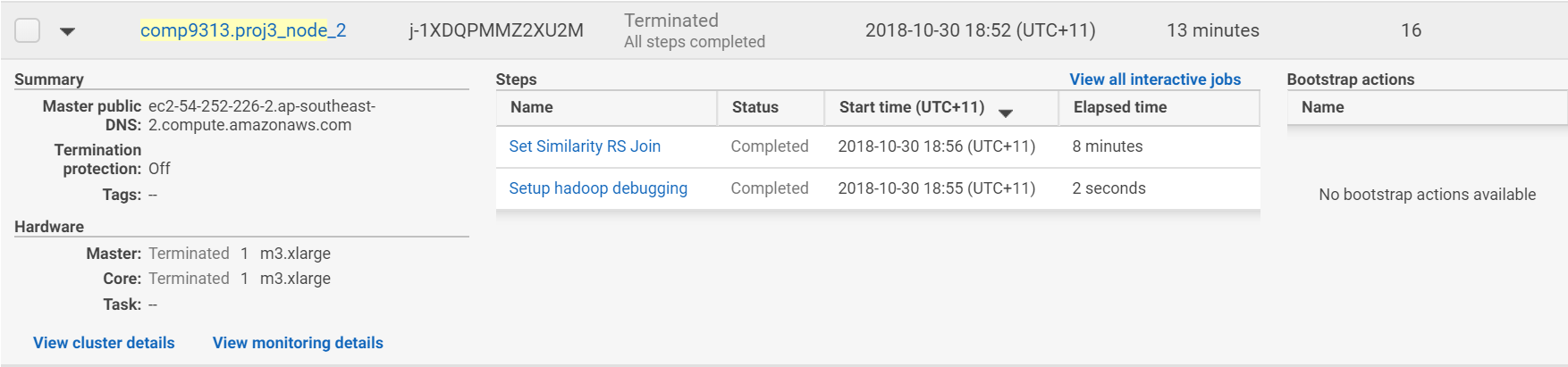


Figure 1 Benchmark on 2 nodes: Running time = 8 minutes

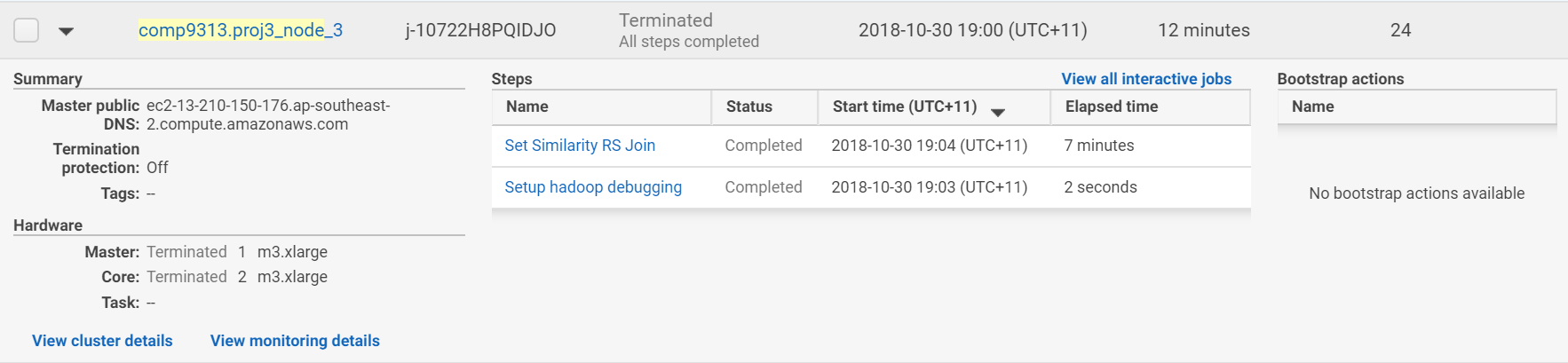


Figure 2 Benchmark 3 node: Running time = 7 minutes

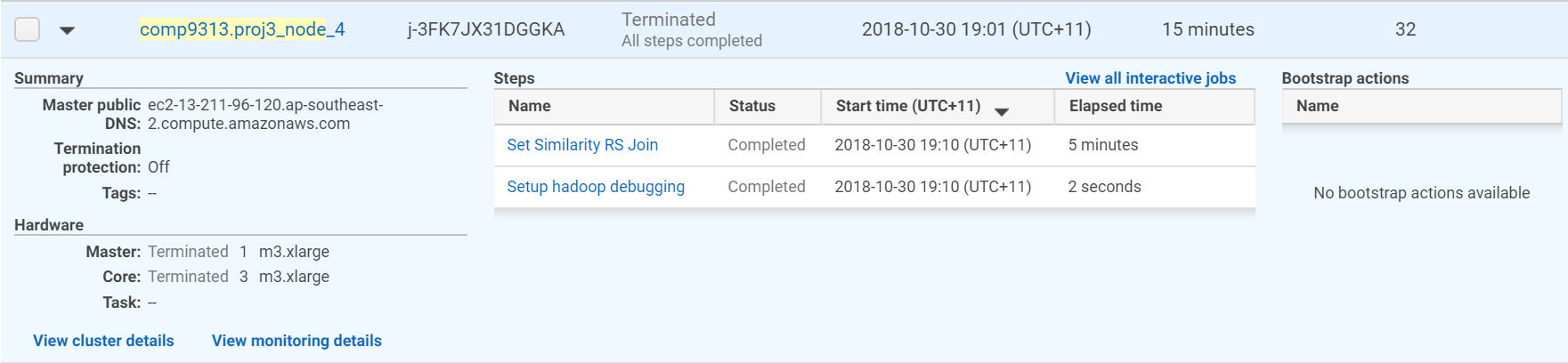


Figure 3 Benchmark on 4 nodes: Running time = 5 minutes

As the figures shows, the elapsed time decreases as the number of nodes increases. The line chart of number of nodes and running time is:

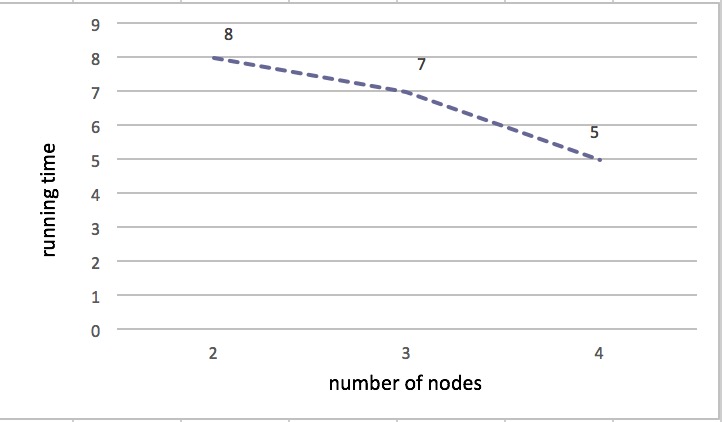


Figure 4 Running Time vs Number of Nodes