Spark: Cluster Computing with Working Sets
Aman Shankar (as5171)

1. Motivation

Most of the applications are built around acyclic data flow such as MapReduce, some of the applications needs cyclic data flow these include machine learning applications and data flow applications.

2. Goal

To create a workflow that works on cyclic data flow and at the same time uses the same fault tolerance and scalability of MapReduce.

3. Approach

Spark uses a Resilient Distributed Datasets(RDD) is a read-only collection. RDD is a read only collection of objects and uses a method called as lineage to allow data to be rebuild if it is lost. It provides two function one is the save function when the data is not going to get reused and a cache function which will keep the data in the memory to be reused at some later stage of the computation.

4. Result

Logistic Regression and Alternating Least Square was implemented using Spark. In Logistic Regression, each iteration took 127s because it was runs as a MapReduce task at every iteration. With Spark, it took 174s for the first iteration and the subsequent iteration took 6s. Hence the job ran at 10x faster pace in Spark. Similarly, the performance of Alternate Least Square was improved by 2.8x by using spark.

5. Conclusion and comments

Spark reduces the computation time by using cache. Spark provides three data abstractions for programming clusters: RDDs, broadcast variables and accumulators. In future, they plan to enhance the use case for RDDs and provide SQL on top on Spark. Spark provides a functionality to implement cyclic workflows and reduce the computation time. It’s a bit weak in implementing RDDs.