CS 378 – Big Data Programming

Lecture 23

Spark Basics

Review

- Assignment 10 – Download and run Spark
    - Logging output control

Apache Spark

- Open source project out of AMPLab at UC Berkeley

- A Spark program defines:
  - Transformations and actions on data sets
  - Data flow, or lineage graph among data sets, induced by the transformations

- Data sets in Spark are called RDDs
  - Resilient Distributed Datasets

Spark Features

- Provide domain specific libraries
  - Example: map-reduce library
  - Promotes functional programming model
- Access to multiple data (file) systems
  - Local, HDFS, Cassandra, S3, database tables, …
- Lazy evaluation, and caching for performance
  - Reduce or eliminate disk I/O
- Support multi-stage and iterative apps

Spark RDDs

- Resilient Distributed Dataset
  - One RDD has one or more partition
  - Partitions are distributed across machines
  - Rebuilt from base data on failure (versus replication)
  - Lazy evaluation – created on demand

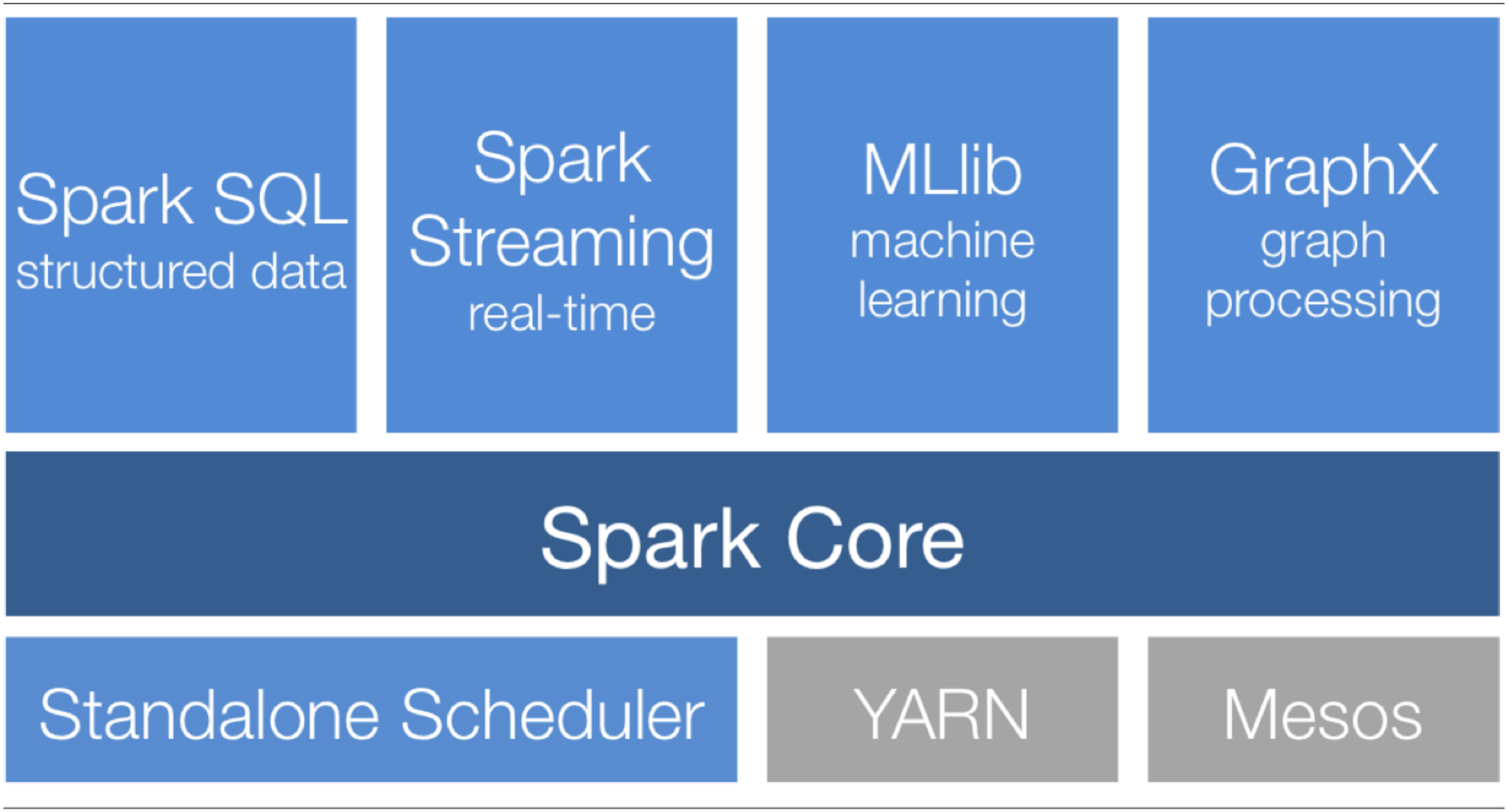
- RDD types offer various functions
  - map, reduce
  - groupBy, reduceByKey
  - joins (inner, leftOuterJoin, rightOuterJoin)
  - filter, sample

Spark

- Provides a higher level of abstraction for coding
  - Multi-stage map-reduce pipeline in Hadoop …
  - Can be composed functions in Spark
- RDD support and libraries
  - Spark SQL – RDD representing relational table
  - Streaming data – D-Stream, Twitter stream
  - Graph data – GraphX
  - …

Spark Stack

Learning Spark, Figure 1-1.



Spark SQL
structured data

Spark Streaming
real-time

MLlib
machine learning

GraphX
graph processing

Spark Core

Standalone Scheduler

YARN

Mesos

Spark Programs

- A Spark program defines:
  - RDDs
    - Input from external sources
    - Produced by a transformation
  - Transformations
    - Produce a new RDD from the input RDD
  - Actions
    - Compute something from the input RDD
      - Return non-RDD objects (e.g., number)
    - Write an RDD to external storage

Spark Example

- Interactive
  - – Scala

- Batch
  - – Java

Spark Actions

- Transformations do not initiate computation (lazy eval)
- Actions on RDDs initiate computation

- Counting
  - `rdd.count()`
- Extract some elements
  - `rdd.take(10)`
- Get all elements
  - `rdd.collect()`
  - Careful, as this assembles the entire RDD. Filter first.

Spark Transformations

- Transformations take functions as arguments
  - This function defines the details of the transform

- Filter
  - Apply a function to each element of an RDD, return those elements that evaluate to true
  - Source RDD element type: `T`
  - Result RDD element type: `T`
  - Java function (class) type: `Function<T, Boolean>`
  - Java method: `Boolean call(T t)`

Spark Transformations

- Map
  - Apply a function to each element of an RDD, return the result of applying the function
  - Source RDD element type: `T`
  - Result RDD element type: `R`
  - Java function (class) type: `Function<T, R>`
  - Java method: `R call(T t)`

Spark Transformations

- Flat Map
  - Apply a function to each element of an RDD, return the result of applying the function (an Iterable)
  - Source RDD element type: `T`
  - Result RDD element type: `R`
  - Java function (class) type: `FlatMapFunction<T, R>`
  - Java method: `Iterable<R> call(T t)`

Spark Transformations

- Map to Pair
  - Apply a function to each element of an RDD, return the result of applying the function (a key and a value)
  - Source RDD element type: `T`
  - Result RDD element type: `<K, V>`
  - Java function (class) type: `PairFunction<T, K, V>`
  - Java method: `Tuple2<K, V> call(T t)`

Spark Transformations

- Reduce by Key
  - Apply a function to each element of an RDD, return the result of applying the function (a key and a value)
  - Source RDD element type: `T`
  - Result RDD (JavaPairRDD) element type: `<K, V>`
  - Java function (class) type: `Function2<V, V, V>`
  - Java method: `V call(V v1, V v2)`

Spark Code Example

- WordCount again