Open Source Big Data Analytics Frameworks Written in Scala

John A. Miller, Casey Bowman, Vishnu Gowda Harish and Shannon Quinn

Department of Computer Science

University of Georgia

5th IEEE International Congress on Big Data

Outline

- Purpose of Big Data Frameworks
- Scala Based Ecosystem
- Evolution of Frameworks
- Scala-Based Open Source Frameworks
  - Spark
  - Kafka and Samza
  - ScalaTion
- Comparison of Frameworks
- Support for Parallel Execution
- Support for NoSQL Databases
- Conclusions

Purpose of Frameworks Supporting Big Data Analytics

- Support the storage and rapid access to large amount of data.
- Support the application of parallel and distributed processing techniques.
- Should provide simple API's, which are not hard to grasp, e.g., MapReduce.
- Internal details like partitioning of data, creation of threads, etc. should be hidden.
- Extensible libraries of analytics techniques (e.g., from statistics, data mining, and machine learning) should be provided.

Scala Based Ecosystem

- Scala supports both functional and object oriented programming paradigm.
- Functional Programming supports richer type systems, data flow processing, immutability for better concurrency, tail call optimization, primitives like map and reduce, etc.
- Has free access to Java ecosystem as it runs on JVM and it has its own libraries for handling data at scale, e.g. Akka Actors, Spark Mlib.
- E.g. Apache Spark, Apache Flink, Kafka + Samza, ScalaTion.

Evolution of Frameworks

- Message Passing Interface (MPI)
  - Provides very little out-of-the-box user-facing functionality such as fault tolerance.
- MapReduce
  - Hadoop includes fault-tolerant distributed data-structures out-of-the-box, making programming using Hadoop easier.
  - May cause serious bottlenecks for iterative algorithms.
- Apache Spark
  - Provides in-memory serialization of intermediate data, which results in significant performance gain over Hadoop.
- Apache Flink
  - Suitable for highly iterative streaming algorithm.

Scala-Based Open Source Frameworks

- Spark:
  - Foundation of Spark is Spark-core which is built in Scala.
  - Additional packages in Spark:
    1. Spark SQL: Allows for relational processing to be done on RDD's and on external datasets.
    2. Spark Streaming: Provides frameworks using RDD to handle stateful computation.
    3. Mlib: Spark's machine learning algorithms.
    4. GraphX: Spark's graph computational framework.

Kafka and Samza

- Kafka
  - Messaging system layer for frameworks like Samza.
  - Producers create messages and consumers subscribe to them.
  - Defines messages of specific type (topic) so that subscribers can efficiently find them.
- Samza
  - Works on distributed processing of real time data.
  - For parallel computation, each input stream is divided into partitions and a job into tasks that are assigned to machines.

ScalaTion

- Supports predictive analytics, graph analytics and simulation modeling.
- Integrated with two NoSQL database systems: columnar databases and graph databases.
- Current focus is on optimizing parallel execution.

Comparison of Frameworks

Features	Hadoop	Spark	Flink	Storm	ScalaTion
Vertical Scalability	Multiple Mappers, Multithreaded Mapper	Threadpool	Pipeline Parallelism (Also Horizontal)	Workers run multiple tasks on separate JVM processes	Multithreading for multicore
Horizontal Scalability	JobTracker sends jar files to TaskTrackers	Mesos, Remote Actors	JobManager Invokes TaskManagers	Nimbus distributes task to Workers	Future Work
Programming Abstraction	MapReduce	Actors and BSP	Coordinator Workers	Master Workers	Threads, .par, Workers
File	HDFS	HDFS	HDFS	HDFS	Files, Memory-mapped files
Databases	HBase	Apache Casandra and Hbase	HBase	HBase and MongoDB	Columnar Database, Graph Database

Comparison (cont.)

Features	Hadoop	Spark	Flink	Storm	ScalaTion
Packages	HIVE, PIG, FLUME, ZOOKEEPER	Spark SQL, MLlib, Spark Streaming, GraphX	Gelly, FlinkML	Storm Submitter, LogWriter	math, linear algebra, statistics, analytics, graph analytics, optimization, simulation
Data Processing	Batch Processing	Batch and streaming	Batch and streaming	Streaming	Interactive

Comparison (cont.)

Features	Hadoop	Spark	Flink	Storm	ScalaTion
Performance	Suitable for ETL like jobs, i.e. data-integration and data transformation.	Suitable for Iterative computation.	Apache Flink outperforms Apache Spark in processing machine learning & graph algorithms and relational queries but not in batch processing.	Applications are designed as directed acyclic graphs.	Optimized multithreading.

Support for Parallel Execution (Matrix Mult)

```
val aa = a()                                                    // def timempm
for (i <- a.range1.par; j <- b.range2) {                        //.par
    val a_i = aa(i); val bt_j = bt()(j)
    var sum = 0.0
    for (k <- a.range2) sum += a_i(k) * bt_j(k)
    c(i, j) = sum
} // for
```

```
val a_i = a(i)()                                                // def time_mv(i)
for (j <- b.range2) {                                           // Worker
    val bt_j = bt()(j)
    var sum = 0.0
    for (k <- b.range2) sum += a_i(k) * bt_j(k)
    c(i, j) = sum
} // for
for (i <- 0 until a.dim1) (new Worker (times_mv, i)).start
```

Support for Data Storage and Access

- Big Data is often stored in distributed file system e.g., Hadoop Distributed File System (HDFS).
- NoSQL databases are used for applications requiring greater storage and performance.
- NoSQL databases includes graph databases, key-value stores, document oriented databases and columnar databases.

Columnar Database

- Stores relations in columns.
- Columns are large and may contain repetitive values, which opens the possibility of compressing the columns.
- E.g. HP-Vertica and C-Store.

Graph Databases

- Database consists of one or more vertex/edge labelled multi-digraphs.
- ScalaTion provides support for efficient pattern matching on large graphs (e.g., Tight Simulation, DualIso).
- E.g. Neo4j and OrientDB.

Conclusions

- Scala based frameworks can provide better support for parallelism.
- Scala based frameworks can provide richer streaming API's.
- These frameworks have well designed libraries for scientific computing, linear algebra and random number generation.
  - https://www.dezyre.com/article/why-learn-scala-programming-for-apache-spark/198
- Google trends indicates that although Apache Hadoop is more popular, there is a recent strong uptick for Apache Spark.