Spark

Lightning-Fast Cluster Computing

www.spark-project.org

This Meetup

1. Project history and plans
2. Spark tutorial
   - » Running locally and on EC2
   - » Interactive shell
   - » Standalone jobs
3. Spark at Quantifind

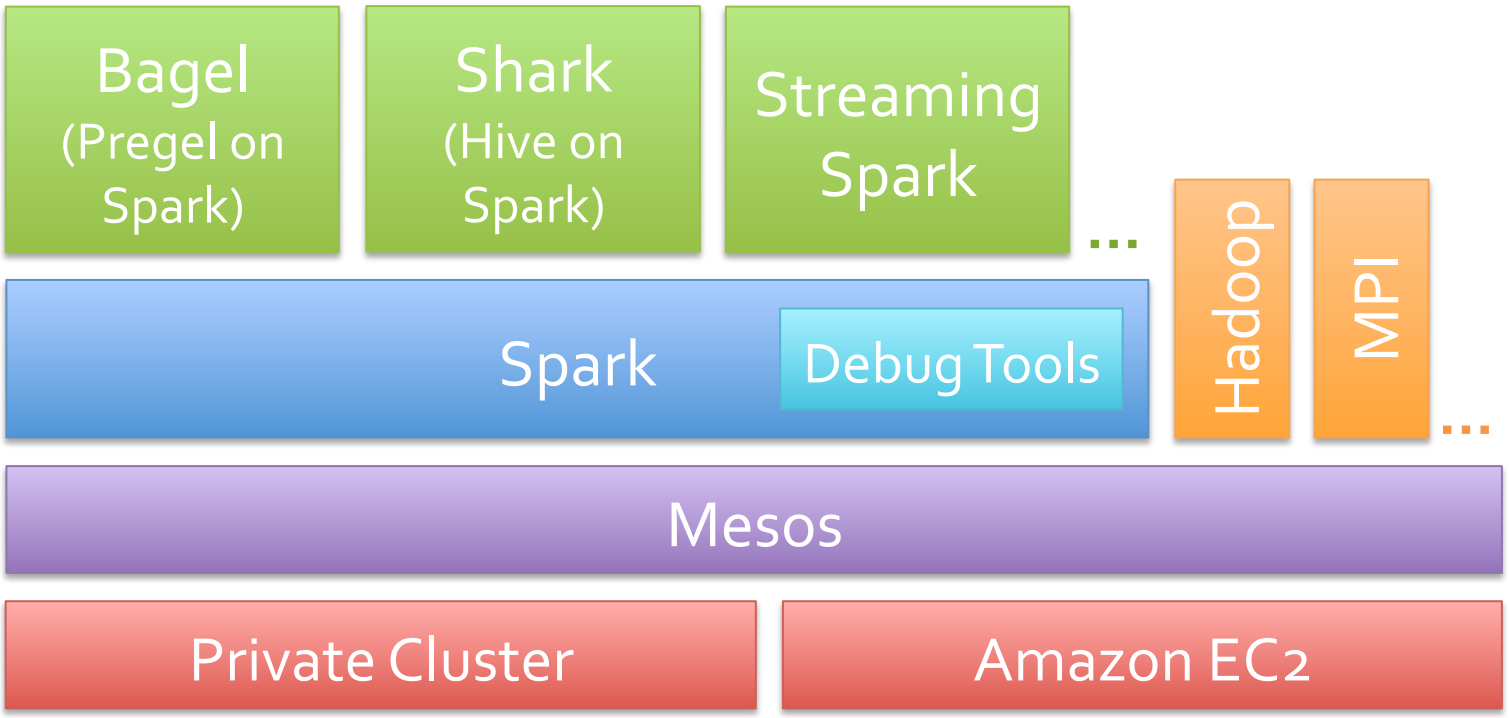
Project Goals

AMP Lab: design next-gen data analytics stack
- » By scaling up Algorithms, Machines and People

Mesos: cluster manager
- » Make it easy to write and deploy distributed apps

Spark: parallel computing system
- » General and efficient computing model supporting in-memory execution
- » High-level API in Scala language
- » Substrate for even higher APIs (SQL, Pregel, …)

Where We're Going



Some Users


CONVIVA


KLOUT


quantiFind


YAHOO!
RESEARCH


University of California
Berkeley


UCSF

Getting Spark

Requirements: Java 6+, Scala 2.9.1

```
git clone git://github.com/mesos/spark.git
cd spark
sbt/sbt compile
```

Wiki data: tinyurl.com/wikisample

These slides: tinyurl.com/sum-talk

Running Locally

```
# run one of the example jobs:

./run spark.examples.SparkPi local


# launch the interpreter:

./spark-shell
```

Running on EC2

```
git clone git://github.com/apache/mesos.git

cd mesos/ec2

./mesos-ec2 -k keypair -i id_rsa.pem -s slaves \
       [launch|stop|start|destroy] clusterName
```

Details: tinyurl.com/mesos-ec2

Programming Concepts

SparkContext: entry point to Spark functions

Resilient distributed datasets (RDDs):

- » Immutable, distributed collections of objects
- » Can be cached in memory for fast reuse

Operations on RDDs:

- » *Transformations*: define a new RDD (map, join, …)
- » *Actions*: return or output a result (count, save, …)

Creating a SparkContext

```
import spark.SparkContext
import spark.SparkContext._

val sc = new SparkContext(“master”, “jobName”)

// Master can be:
//   local    - run locally with 1 thread
//   local[K] - run locally with K threads
//   mesos://master@host:port
```

Creating RDDs

```
// turn a Scala collection into an RDD
sc.parallelize(List(1, 2, 3))

// text file from local FS, HDFS, S3, etc
sc.textFile("file.txt")
sc.textFile("directory/*.txt")
sc.textFile("hdfs://namenode:9000/path/file")

// general Hadoop InputFormat:
sc.hadoopFile(keyCls, valCls, inputFmt, conf)
```

RDD Operations

Transformations (define a new RDD)	`map`
	`filter`
	`sample`
	`groupByKey`
	`reduceByKey`
	`cogroup`
		`flatMap`
		`union`
		`join`
		`cross`
		`mapValues`
		`...`

Actions (output a result)	`collect`
	`reduce`
	`take`
	`fold`
		`count`
		`saveAsHadoopFile`
		`saveAsTextFile`
		`...`

Persistence	`cache`	(keep RDD in RAM)

Standalone Jobs

Without Maven: package Spark into a jar

```
sbt/sbt assembly
# use core/target/spark-core-assembly-*.jar
```

With Maven:

```
sbt/sbt publish-local
# add dep. on org.spark-project / spark-core
```

Standalone Jobs

Configure Spark's install location and your job's classpath as environment variables:

```
export SPARK_HOME=...
export SPARK_CLASSPATH=...
```

Or pass extra args to SparkContext:

```
new SparkContext(master, name, sparkHome, jarList)
```

Where to Go From Here

Programming guide:
www.spark-project.org/documentation.html

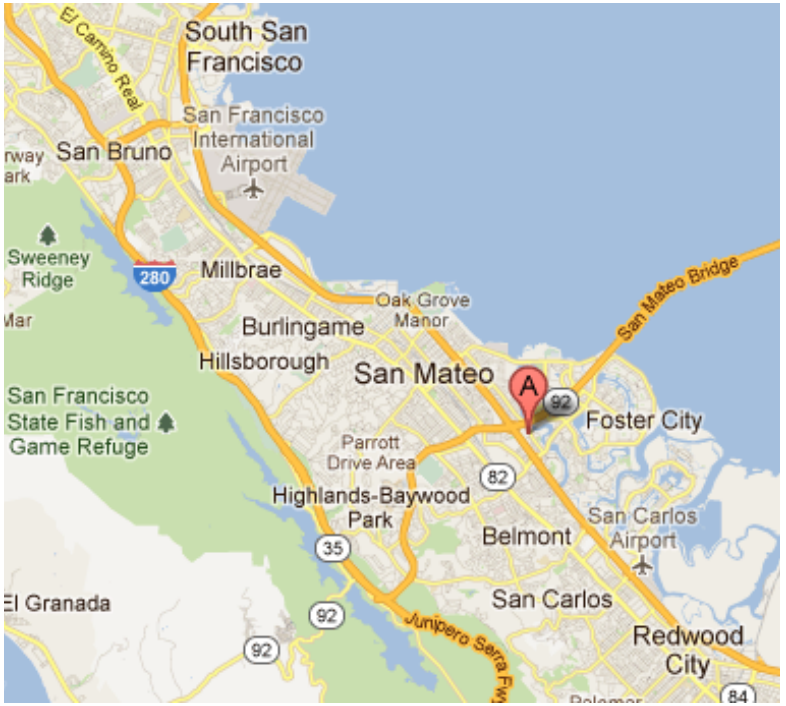
Example programs: examples/src/main/scala

RDD ops: RDD.scala, PairRDDFunctions.scala

Next Meetup

Thursday, Feb 23rd at 6:30 PM

Conviva, Inc
2 Waters Park Drive
San Mateo, CA 94403



Give us feedback!
tinyurl.com/firstsparkmeetup