Big Data Processing

Patrick Wendell
Databricks
About me

Committer and PMC member of Apache Spark
“Former” PhD student at Berkeley
Left Berkeley to help found Databricks
Now managing open source work at Databricks
Focus is on networking and operating systems
Outline

Overview of today
The Big Data problem
The Spark Computing Framework
Straw poll. Have you:

- Written code in a numerical programming environment (R/Matlab/Weka/etc)?

- Written code in a general programming language (Python/Java/etc)?

- Written multi-threaded or distributed computer programs?
Today’s workshop

Overview of trends in large-scale data analysis

Introduction to the Spark cluster-computing engine with a focus on numerical computing

Hands-on workshop with TA’s covering Scala basics and using Spark for machine learning
Hands-on Exercises

You’ll be given a cluster of 5 machines*

(a) Text mining of full text of Wikipedia
(b) Movie recommendations with ALS

5 machines * 200 people = 1,000 machines
Outline

Overview of today

The Big Data problem

The Spark Computing Framework
The Big Data Problem

Data is growing faster than computation speeds

Growing data sources
  » Web, mobile, scientific, …

Cheap storage
  » Doubling every 18 months

Stalling CPU speeds

Storage bottlenecks
Examples

Facebook’s daily logs: 60 TB
1000 genomes project: 200 TB
Google web index: 10+ PB

Cost of 1 TB of disk: $50
Time to read 1 TB from disk: 6 hours (50 MB/s)
The Big Data Problem

Single machine can no longer process or even store all the data!

Only solution is to distribute over large clusters
Google Datacenter

How do we program this thing?
What’s hard about cluster computing?

How to divide work across nodes?
- Must consider network, data locality
- Moving data may be very expensive

How to deal with failures?
- 1 server fails every 3 years => 10K nodes see 10 faults/day
- Even worse: stragglers (node is not failed, but slow)
A comparison:

How to divide work across nodes?

What if certain elements of a matrix/array became 1000 times slower to access than others?

How to deal with failures?

What if with 10% probability certain variables in your program became undefined?
Outline

Overview of today

The Big Data problem

The Spark Computing Framework
The Spark Computing Framework

Provides a programming abstraction and parallel runtime to hide this complexity.

“Here’s an operation, run it on all of the data”

» I don’t care where it runs (you schedule that)
» In fact, feel free to run it twice on different nodes
Resilient Distributed Datasets

Key programming abstraction in Spark

Think “parallel data frame”

Supports collections/set API’s

```
# get variance of 5th field of a tab-delimited dataset
rdd = spark.hadoopFile("big-file.txt")

rdd.map(line => line.split("\t")(4))
  .map(cell => cell.toDouble())
  .variance()
```
RDD Execution

Automatically split work into many small, idempotent tasks

Send tasks to nodes based on data locality

Load-balance dynamically as tasks finish
Fault Recovery

1. If a task crashes:
   » Retry on another node
     • OK for a map because it had no dependencies
     • OK for reduce because map outputs are on disk

Requires user code to be **deterministic**
Fault Recovery

2. If a node crashes:
   » Relaunch its current tasks on other nodes
   » Relaunch any maps the node previously ran
     • Necessary because their output files were lost
Fault Recovery

3. If a task is going slowly (straggler):
   » Launch second copy of task on another node
   » Take the output of whichever copy finishes first, and kill the other one
Spark Compared with Earlier Approaches

Higher level, declarative API.

Built from the “ground up” for performance – including support for leveraging distributed cluster memory.

Optimized for iterative computations (e.g. machine learning)
Spark Platform

Spark RDD API
Spark Platform: GraphX

```
graph = Graph(vertexRDD, edgeRDD)
graph.connectedComponents() # returns a new RDD
```
Spark Platform: MLLib

```python
model = LogisticRegressionWithSGD.train(trainRDD)
dataRDD.map(point => model.predict(point))
```
Spark Platform: Streaming

dstream = spark.networkInputFileStream()
dstream.countByWindow(Seconds(30))
Spark Platform: SQL

\[
rdd = \text{sql"select * from rdd1 where age > 10"}
\]
Performance

[1] https://amplab.cs.berkeley.edu/benchmark/
Composition

Most large-scale data programs involve parsing, filtering, and cleaning data.

Spark allows users to compose patterns elegantly. E.g.:

*Select input dataset with SQL, then run machine learning on the result.*
Spark Community

One of the largest open source projects in big data

150+ developers contributing

30+ companies contributing

Contributors in past year

[Graph showing contributors]
Community Growth

Spark 0.6: 17 contributors

Spark 0.7: 31 contributors

Spark 0.8: 67 contributors

Spark 0.9: 83 contributors
Databricks

Primary developers of Spark today

Founded by Spark project creators

A nexus of several research areas:

➔ OS and computer architecture

➔ Networking and distributed systems

➔ Statistics and machine learning
Today’s Speakers

Holden Karau – Worked on large-scale storage infrastructure @ Google.

*Intro to Spark and Scala*

Hossein Falaki – Data scientist at Apple.

*Numerical Programming with Spark*

Xiangrui Meng – PhD from Stanford ICME. Lead contributor on MLLib.

*Deep dive on Spark’s MLLib*
Questions?