Apache Spark and the Emerging Technology Landscape for Big Data

Universidade da Coruña
2015-05-27

Paco Nathan **@pacoid**
slides **http://goo.gl/A0WL8y**

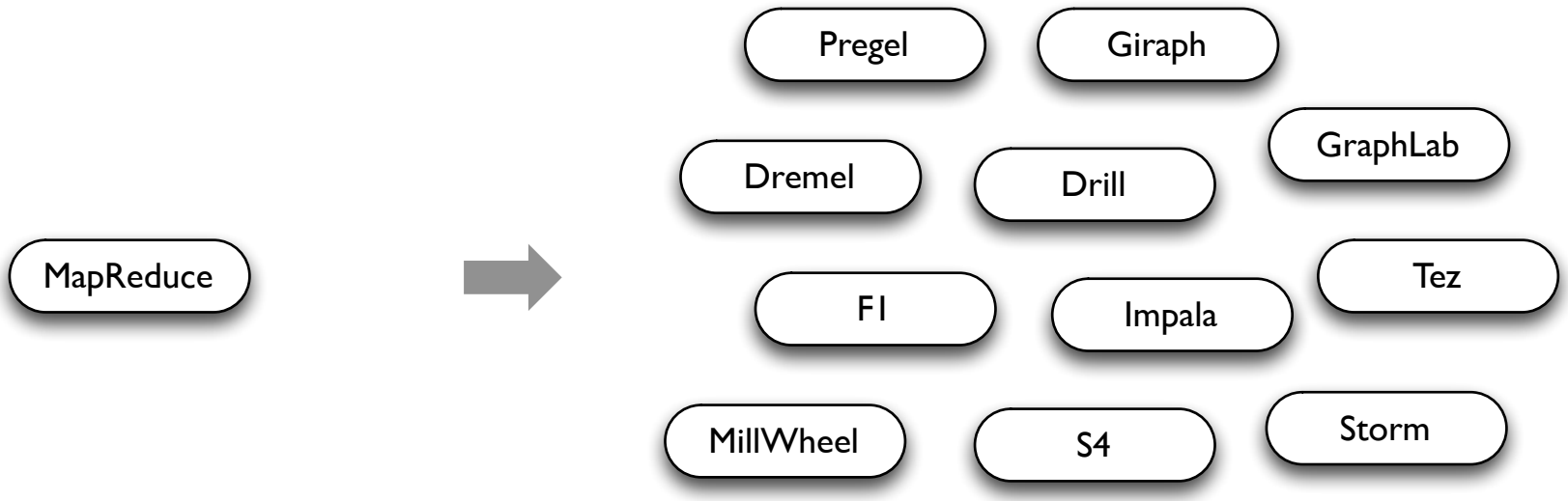
Big Data: *Intentions*

Getting started working with Big Data may seem like going after a big fish…

Big Data: *Realities*

However, learning a variety of complex Big Data frameworks feels more like the perspective of a tuna caught in the labyrinth of *La Almadraba*

Big Data: *Many Specialized Systems*

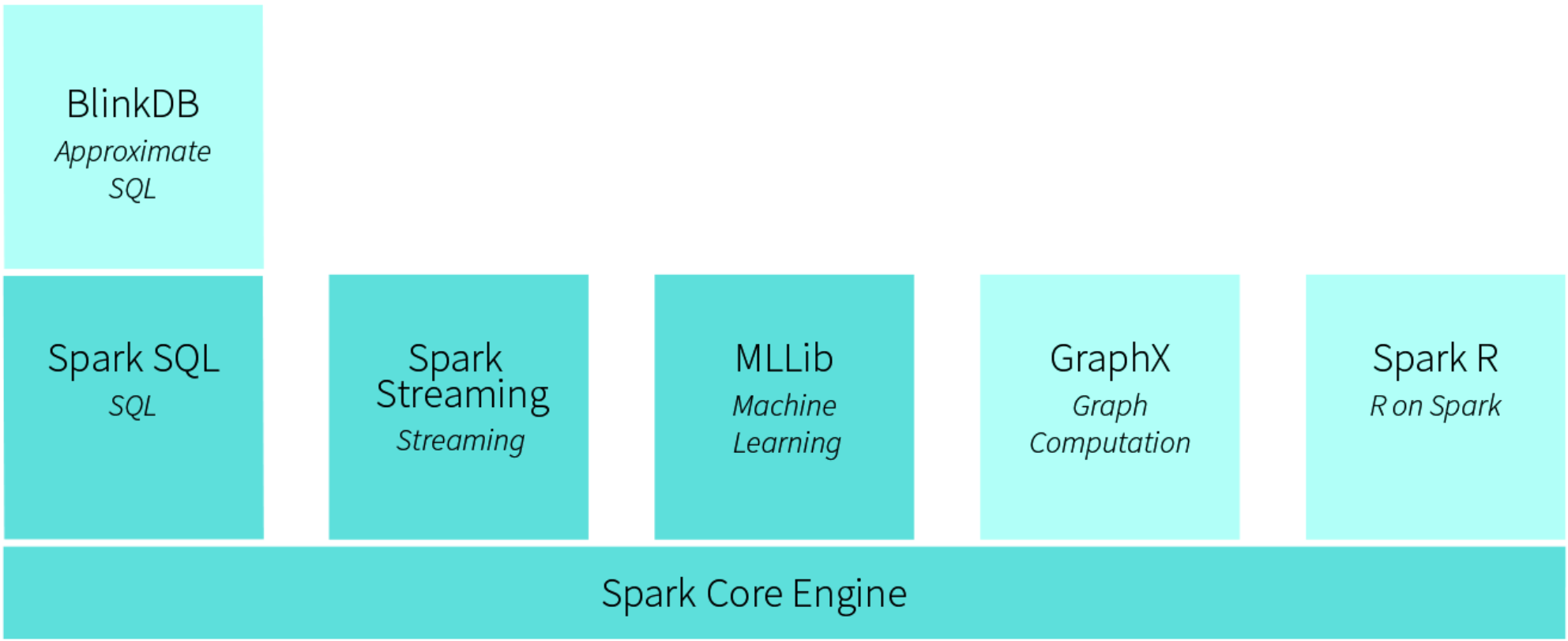


General Batch Processing

Specialized Systems:
iterative, interactive, streaming, graph, etc.

MR doesn't compose well for large applications, and so *specialized systems* emerged as workarounds

Big Data: *Unified Workflows based on Apache Spark*



Big Data: *What is Spark?*

- leverages current generation of commodity hardware
- organizes data as Resilient Distributed Datasets (RDD)
- provides fault tolerance and parallel processing at scale
- lazy eval of DAG optimizes pipelining in cluster computing
- functional programming simplifies SQL, Streaming, ML, Graphs, etc.
- unified engine removes need for many specialized systems

Big Data: *What is Spark?*

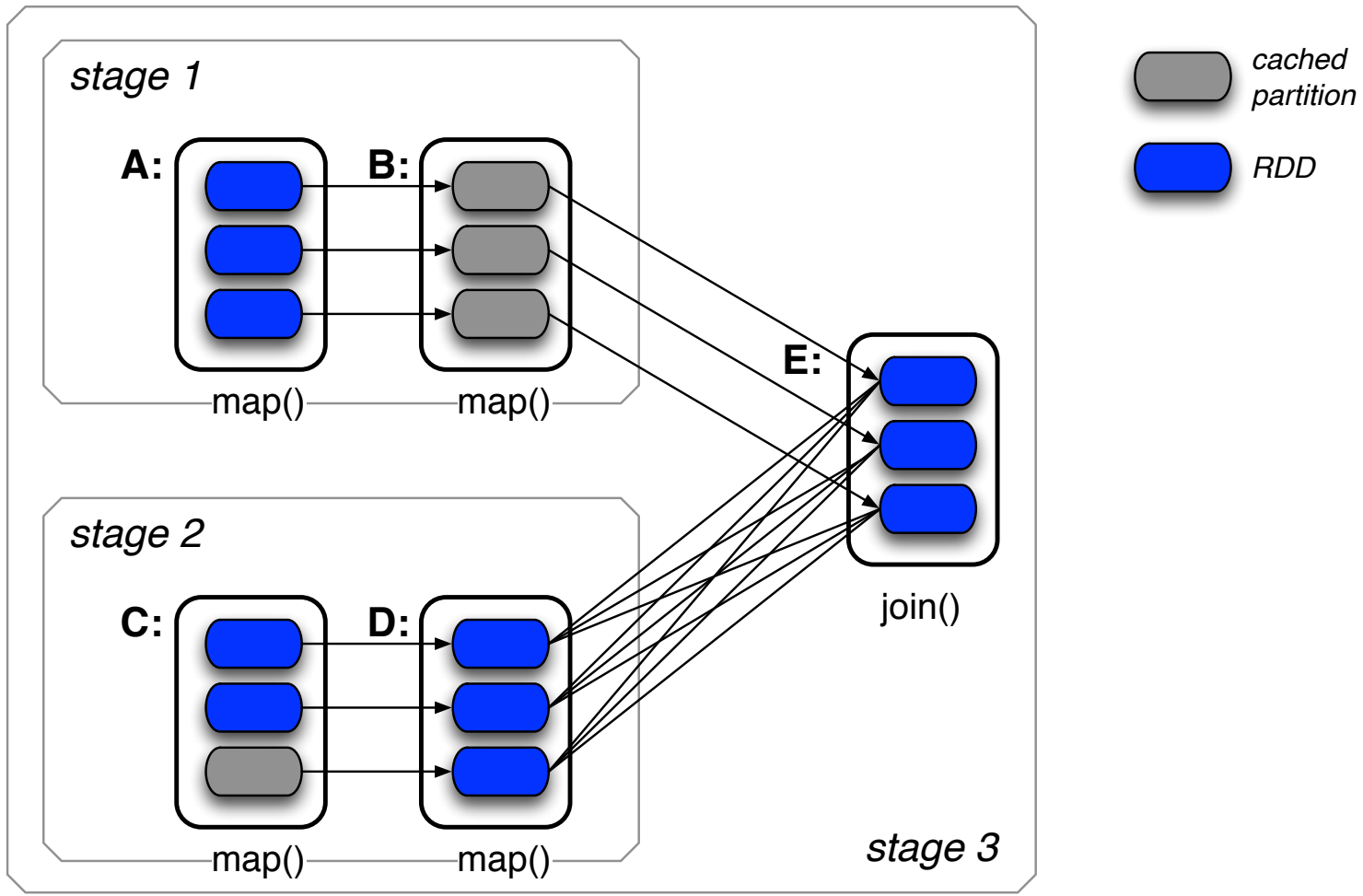
```java
public class WordCount {
  public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{

    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();

    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }

  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }

  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
    if (otherArgs.length < 2) {
      System.err.println("Usage: wordcount <in> [<in>...] <out>");
      System.exit(2);
    }
    Job job = new Job(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    for (int i = 0; i < otherArgs.length - 1; ++i) {
      FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
    }
    FileOutputFormat.setOutputPath(job,
      new Path(otherArgs[otherArgs.length - 1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

WordCount in 50+ lines of Java MR

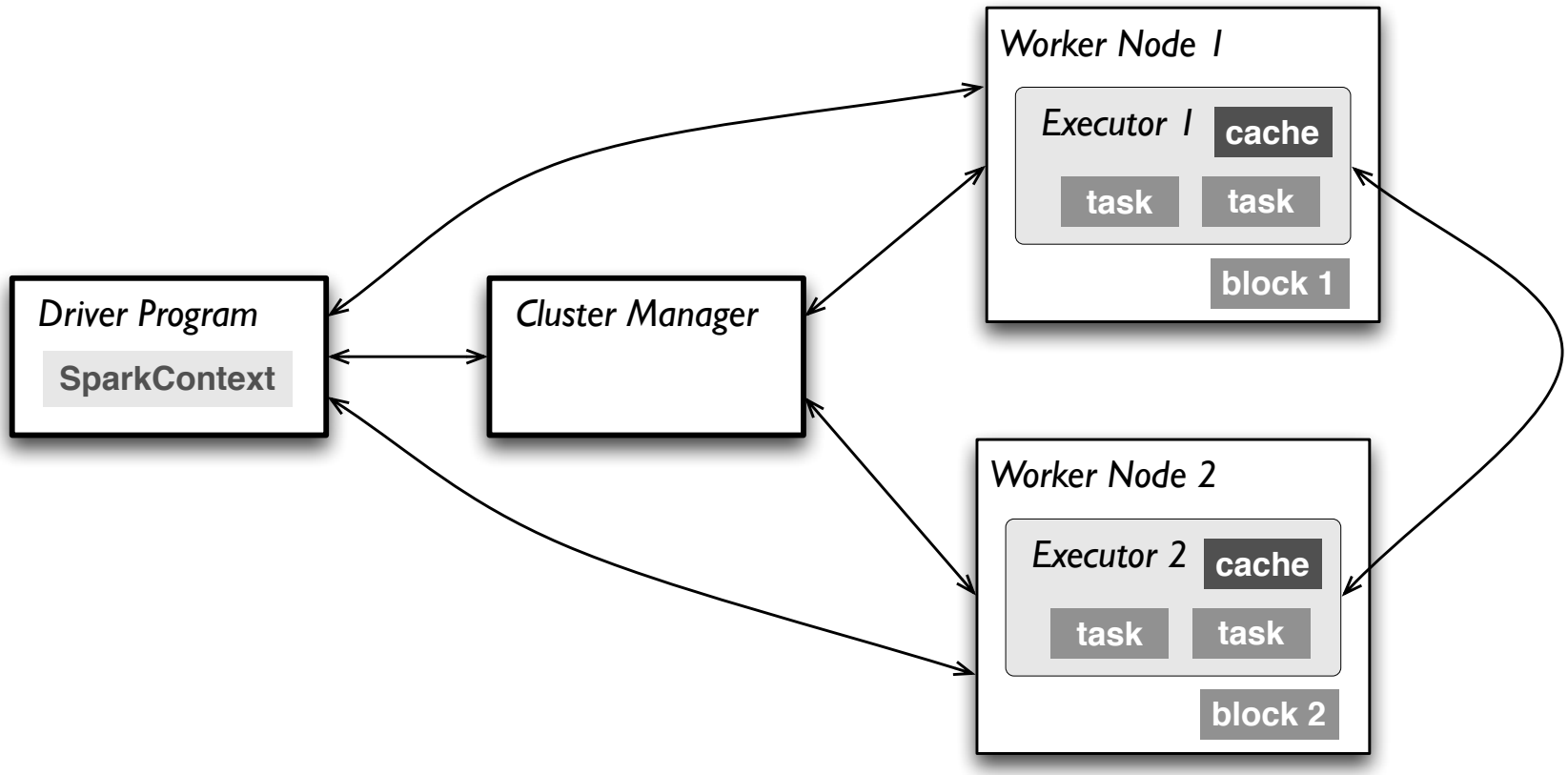
```scala
val f = sc.textFile(inputPath)
val w = f.flatMap(l => l.split(" ")).map(word => (word, 1)).cache()
w.reduceByKey(_ + _).saveAsText(outputPath)
```

WordCount in 3 lines of Spark

Big Data: *What is Spark? – Logical Architecture*



Big Data: *What is Spark? – Physical Architecture*



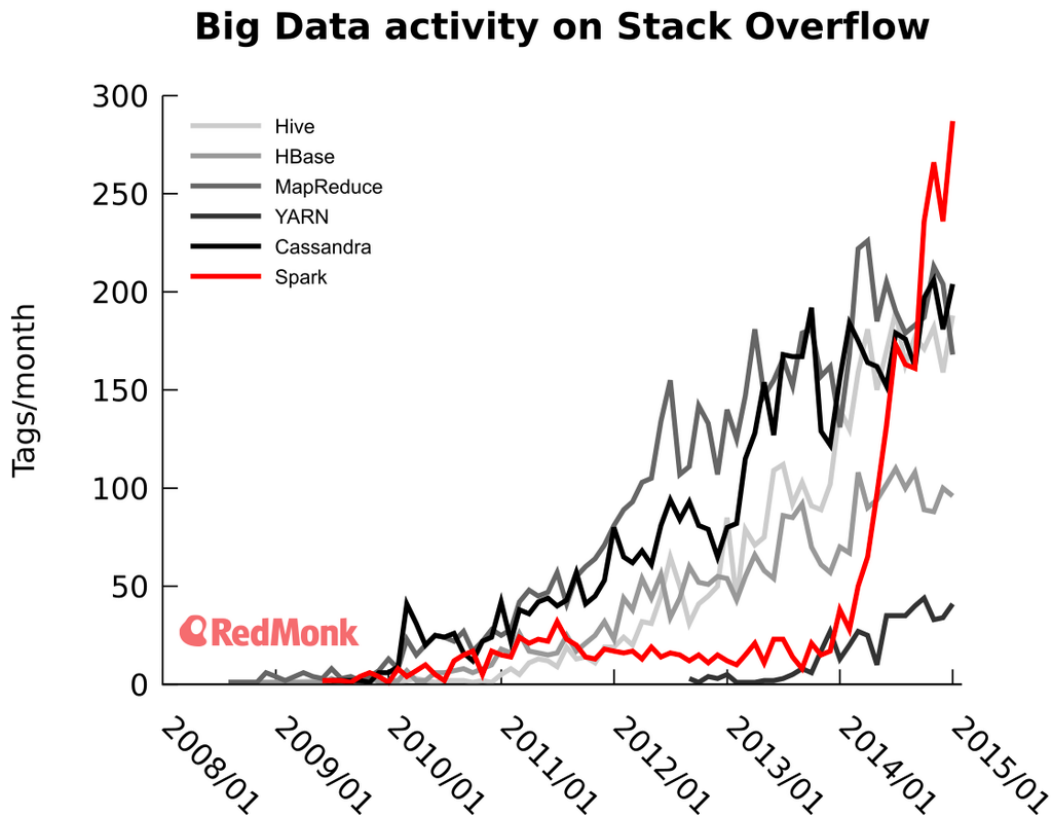
Results: *Gray Sort Challenge – World Record*

databricks.com/blog/2014/11/05/spark-officially-sets-a-new-record-in-large-scale-sorting.html

	Hadoop MR Record	**Spark Record**	**Spark 1 PB**
Data Size	102.5 TB	100 TB	1000 TB
Elapsed Time	72 mins	23 mins	234 mins
# Nodes	2100	206	190
# Cores	50400 physical	6592 virtualized	6080 virtualized
Cluster disk throughput	3150 GB/s (est.)	618 GB/s	570 GB/s
Sort Benchmark Daytona Rules	Yes	Yes	No
Network	dedicated data center, 10Gbps	virtualized (EC2) 10Gbps network	virtualized (EC2) 10Gbps network
Sort rate	**1.42 TB/min**	**4.27 TB/min**	**4.27 TB/min**
Sort rate/node	**0.67 GB/min**	**20.7 GB/min**	**22.5 GB/min**

Results: *Spark on StackOverflow*

twitter.com/dberkholz/status/568561792751771648



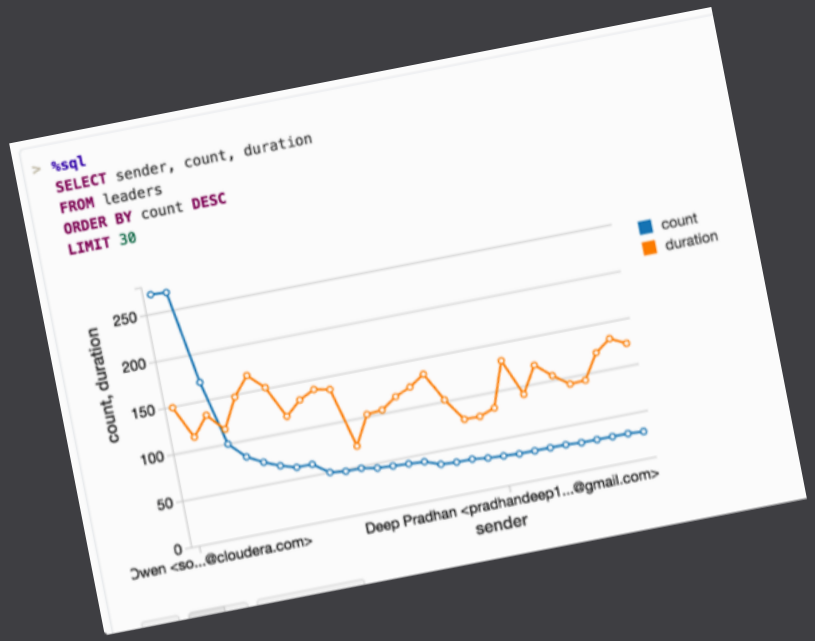
Big Data: *Personal Observation*

Apache Spark represents an opportunity for collaboration between industry and academia:

More so than previous Big Data frameworks, Spark is pure open source and successful use cases at scale leverage algorithms and mathematics in a more fundamental way

- *academia needs real-world data from industry*
- *industry needs experts in latest techniques from academia*

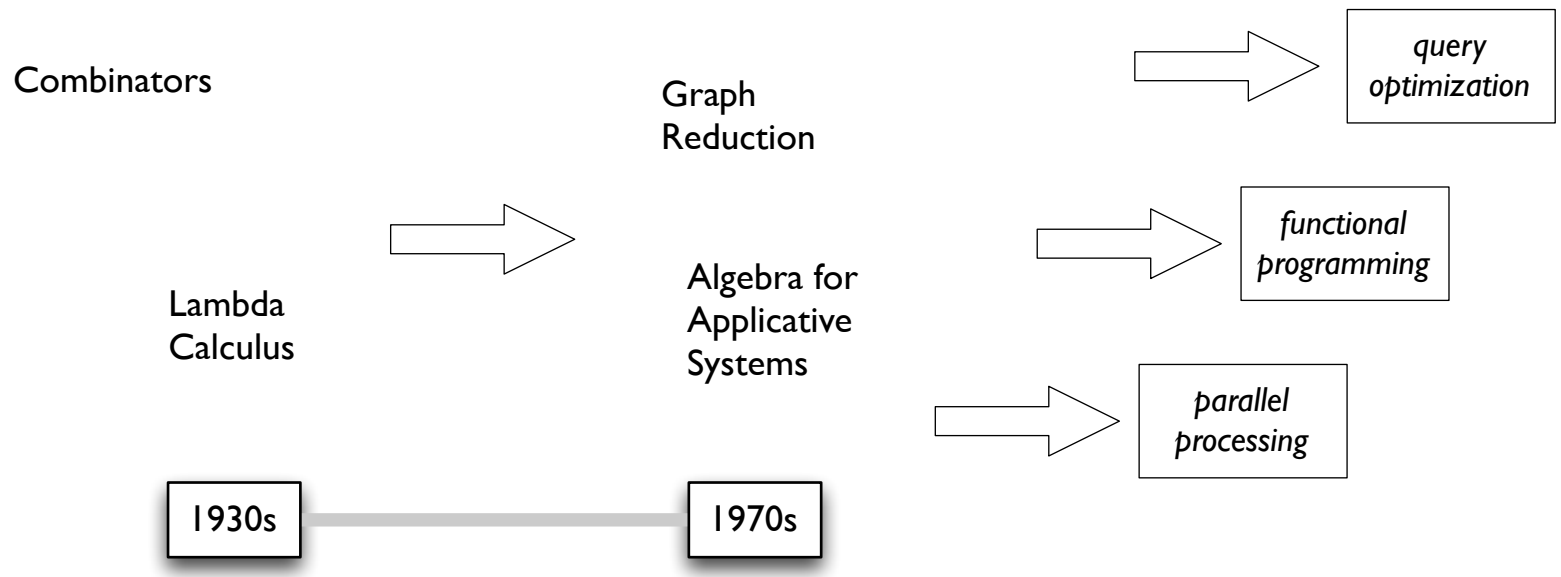
Backstory: themes and primary sources



databricks

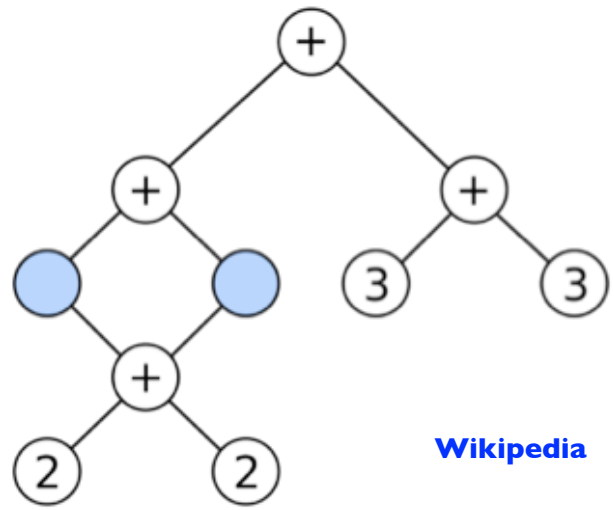
Backstory: *1930s to 1970s*

Early work in the theory of computation led to foundations for functional programming, parallel processing, and query optimization methods…



Backstory: *1930s to 1970s – Themes*

- abstraction layers provided a structured context for defining functions
- algebraic properties of functions allowed for improved compiler optimizations
- parallelism leveraged semigroup structure of the data
- lazy evaluation of a graph, used in functional programming
- lacked the needed compute power



Wikipedia

Backstory: *1930s to 1970s – Primary Sources*

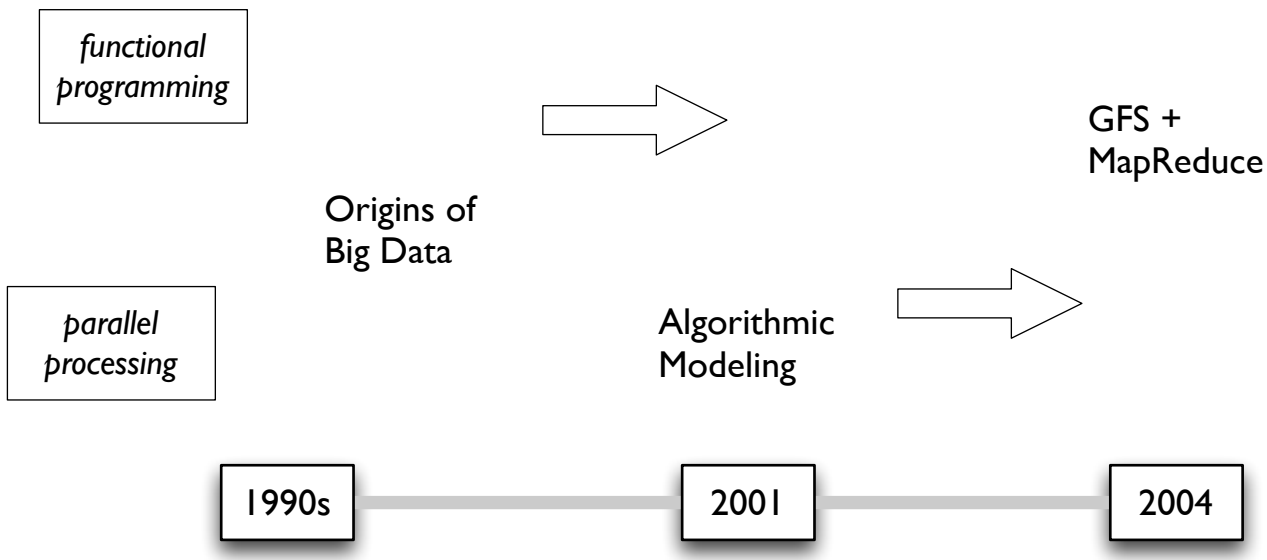
"Computability and λ-Definability"
Alan Turing
The Journal of Symbolic Logic 2 (4): (1937), 153-163
10.2307/2268280

"Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs"
John Backus
ACM Turing Award (1977)
stanford.edu/class/cs242/readings/backus.pdf

"A new implementation technique for applicative languages"
David Turner
Softw: Pract. Exper., 9: (1979), 31-49
10.1002/spe.4380090105

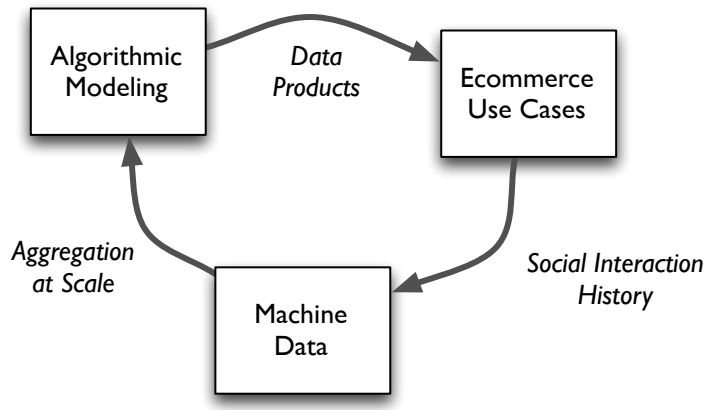
Backstory: *1990s to early 2000s*

Initial successes in e-commerce led to the origins of algorithmic modeling, horizontal scale-out, the Big Data "flywheel", then subsequently MapReduce…



Backstory: *1990s to early 2000s – Themes*

- AMZN, EBAY, GOOG, YHOO avoided paying $$ to IBM, ORCL, etc., by scaling out clusters of commodity hardware
- Big Data "flywheel" pushed horizontal scale-out
- culture of "Data Modeling" shifted to a culture of "Algorithmic Modeling" at scale
- GOOG required fault-tolerance for large ML jobs
- circa 2002 hardware drove MapReduce design



Backstory: *1990s to early 2000s – Primary Sources*

"Social information filtering"
Upendra Shardanand, Pattie Maes
CHI (1995), 210-217
dl.acm.org/citation.cfm?id=223931

"Early Amazon: Splitting the website"
Greg Linden
glinden.blogspot.com/2006/02/early-amazon-splitting-website.html

"The eBay Architecture"
Randy Shoup, Dan Pritchett
addsimplicity.com/downloads/eBaySDForum2006-11-29.pdf

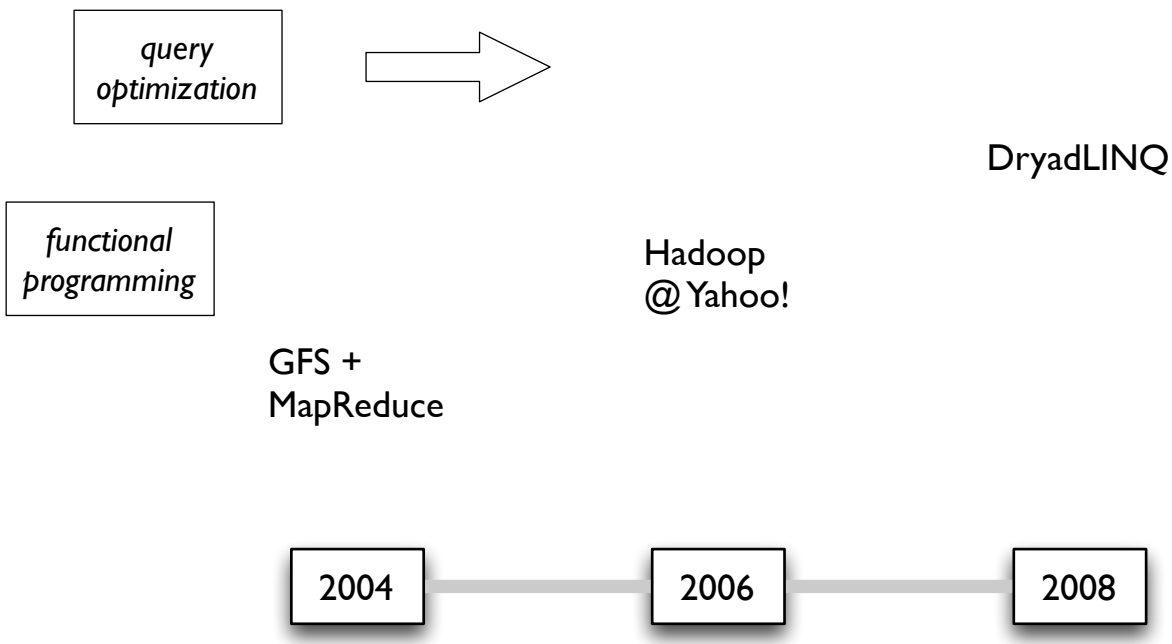
"Inktomi's Wild Ride"
Erik Brewer (0:05:31 ff)
youtu.be/E91oEn1bnXM

"Statistical Modeling: The Two Cultures"
Leo Breiman
Statist. Sci. 16:3 (2001), 199-231
10.1214/ss/1009213726

MapReduce: Simplified Data Processing on Large Clusters
Jeffrey Dean, Sanjay Ghemawat
OSDI (2004)
research.google.com/archive/mapreduce.html

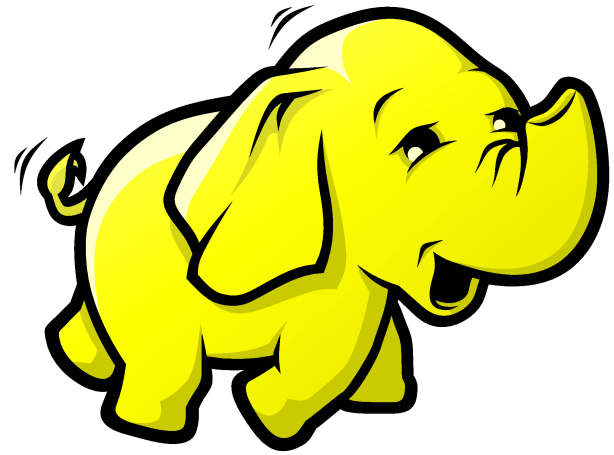
Backstory: *late 2000s*

Projects built on top of MapReduce, with Apache Hadoop gaining significant traction in industry…



Backstory: *late 2000s – Themes*

- batch jobs using MapReduce on clusters of commodity hardware proved valuable in industry
- specialized systems emerged because many use cases had requirements beyond batch (SQL, real-time, etc.)
- abstraction layers emerged because it is difficult to hire enough engineers to "think" in MapReduce
- functional programming reduced software engineering costs for Big Data apps



Apache Hadoop

Backstory: *late 2000s – Primary Sources*

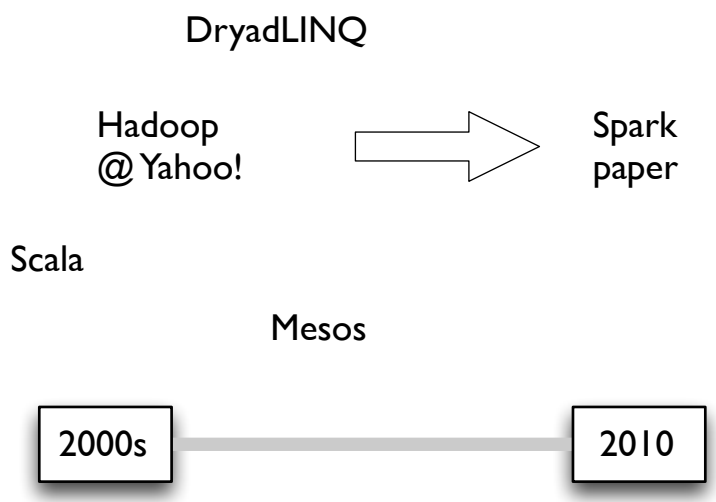
"Hadoop, a brief history"
Doug Cutting
Yahoo! (2006)
research.yahoo.com/files/cutting.pdf

"Improving MapReduce Performance in Heterogeneous Environments"
Matei Zaharia, et al.
OSDI: (2008), 29-42
dl.acm.org/citation.cfm?id=1855744

"DryadLINQ: A System for General-Purpose Distributed
Data-Parallel Computing Using a High-Level Language"
Yuan Yu, et al.
OSDI: (2008) 1-4
research.microsoft.com/en-us/projects/DryadLINQ/

Backstory: *early 2010s*

Apache Spark emerged from the Apache Mesos project, leveraging Scala, with the capability to subsume many of the specialized systems that had become popular for Big Data…



Backstory: *early 2010s – Themes*

- generalized patterns led to a unified engine for many use cases
- lazy evaluation of the lineage graph reduced wait states, improved pipelining (less synchronization barriers)
- generational differences in hardware, e.g., off-heap use of large memory spaces
- functional programming improved ease of use, reduced costs to maintain large apps
- lower overhead for starting jobs, less expensive shuffles

Backstory: *early 2010s – Primary Sources*

"The Origins of Scala"
Bill Venners, Frank Sommers
Scalazine (2009-05-04)
artima.com/scalazine/articles/origins_of_scala.html

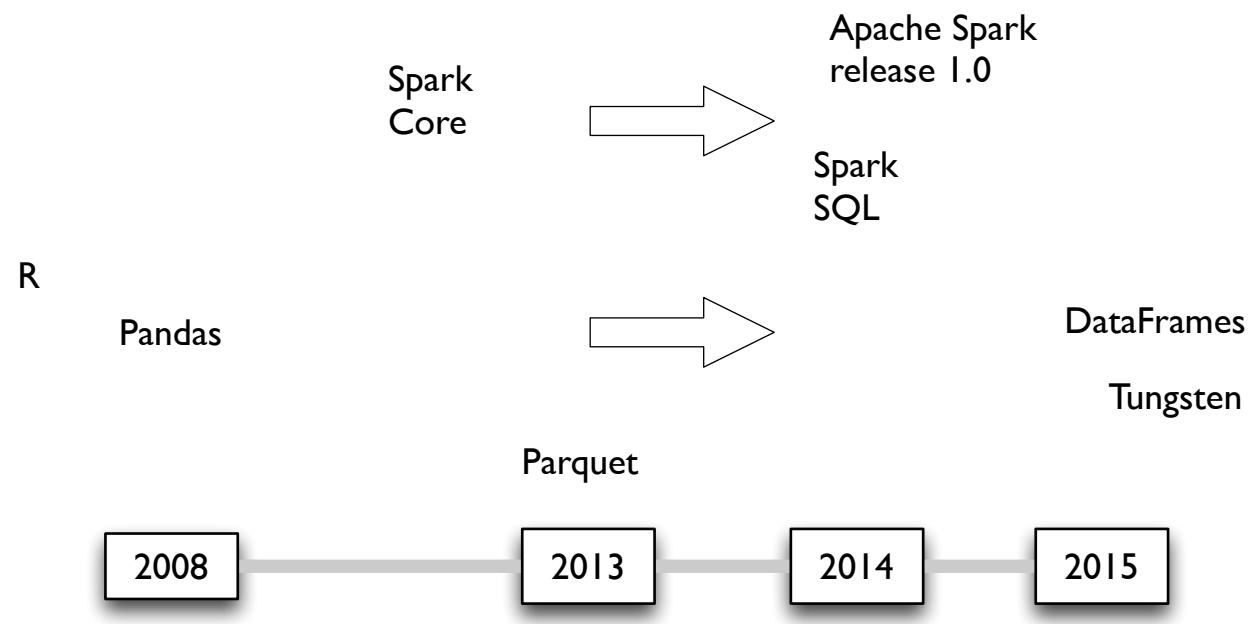
"Spark: Cluster Computing with Working Sets"
Matei Zaharia, et al.
HotCloud: (2010)
dl.acm.org/citation.cfm?id=1863103.1863113

"Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center"
Benjamin Hindman, et al.
NSDI: (2011), 295-308
dl.acm.org/citation.cfm?id=1972488

"Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing"
Matei Zaharia, et al.
NSDI: (2012)
dl.acm.org/citation.cfm?id=2228301

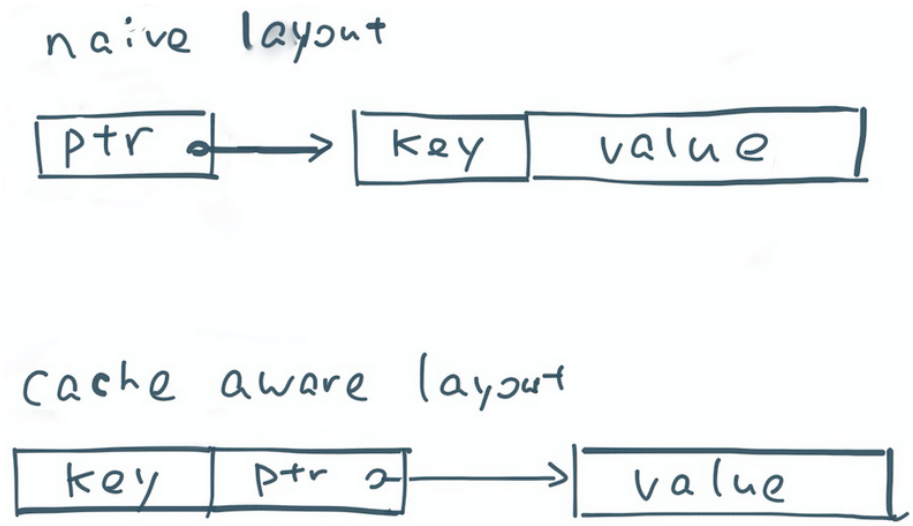
Backstory: *mid 2010s*

Apache Spark became one of the most popular frameworks for Big Data…



Backstory: *mid 2010s – Themes*

- *DataFrames* provide an excepted metaphor, as a higher abstraction layer than RDDs, leveraging *Catalyst* optimizer
- *Parquet* as a best practice: columnar store, excellent compression, preserves schema, pushdown predicates, etc.
- *Tungsten*: application semantics help mitigate the overhead of JVM object model and GC; cache-aware computation leverages memory hierarchy; code generation exploits modern compilers and CPUs



Backstory: *mid 2010s – Primary Sources*

Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython
Wes McKinney
O'Reilly Media (2012)
http://shop.oreilly.com/product/0636920023784.do

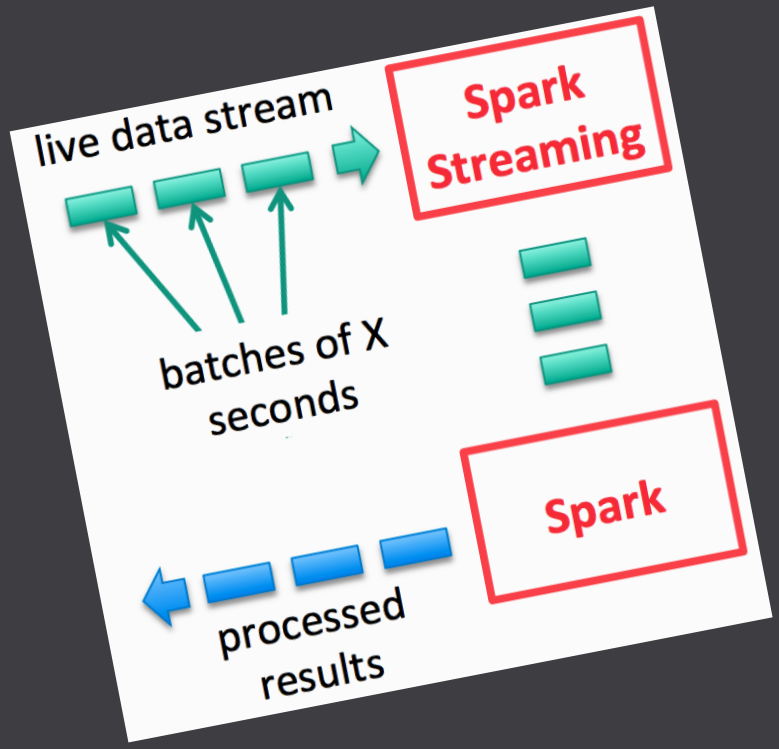
"Parquet: Columnar storage for the people"
Julien Le Dem
Strata + Hadoop World, New York (2013)
parquet.apache.org/presentations/

"Spark SQL: Manipulating Structured Data Using Spark"
Michael Armbrust, Reynold Xin
databricks.com/blog/2014/03/26/Spark-SQL-manipulating-structured-data-using-Spark.html

"Introducing DataFrames in Spark for Large Scale Data Science"
Reynold Xin, Michael Armbrust, Davies Liu
databricks.com/blog/2015/02/17/introducing-dataframes-in-spark-for-large-scale-data-science.html

"Project Tungsten: Bringing Spark Closer to Bare Metal"
Reynold Xin, Josh Rosen
databricks.com/blog/2015/04/28/project-tungsten-bringing-spark-closer-to-bare-metal.html

Disruptive Patterns



Disruptive Patterns:

Three suggested areas for R&D, among the most disruptive innovations based on Spark:

1. *Streaming analytics, especially stateful apps that leverage approximation algorithms – with demand driven by IoT*

2. *Generalized ML workflows, especially large-scale matrix factorization and convex optimization for industry uses*

3. *Cloud-based notebooks, building on containers, IPython, and DataFrames, as tools for collaboration and teaching – ultimately as disruptive as spreadsheets in the 1980s*

Disruptive Patterns:

Three suggested areas for R&D, among the most disruptive innovations based on Spark:

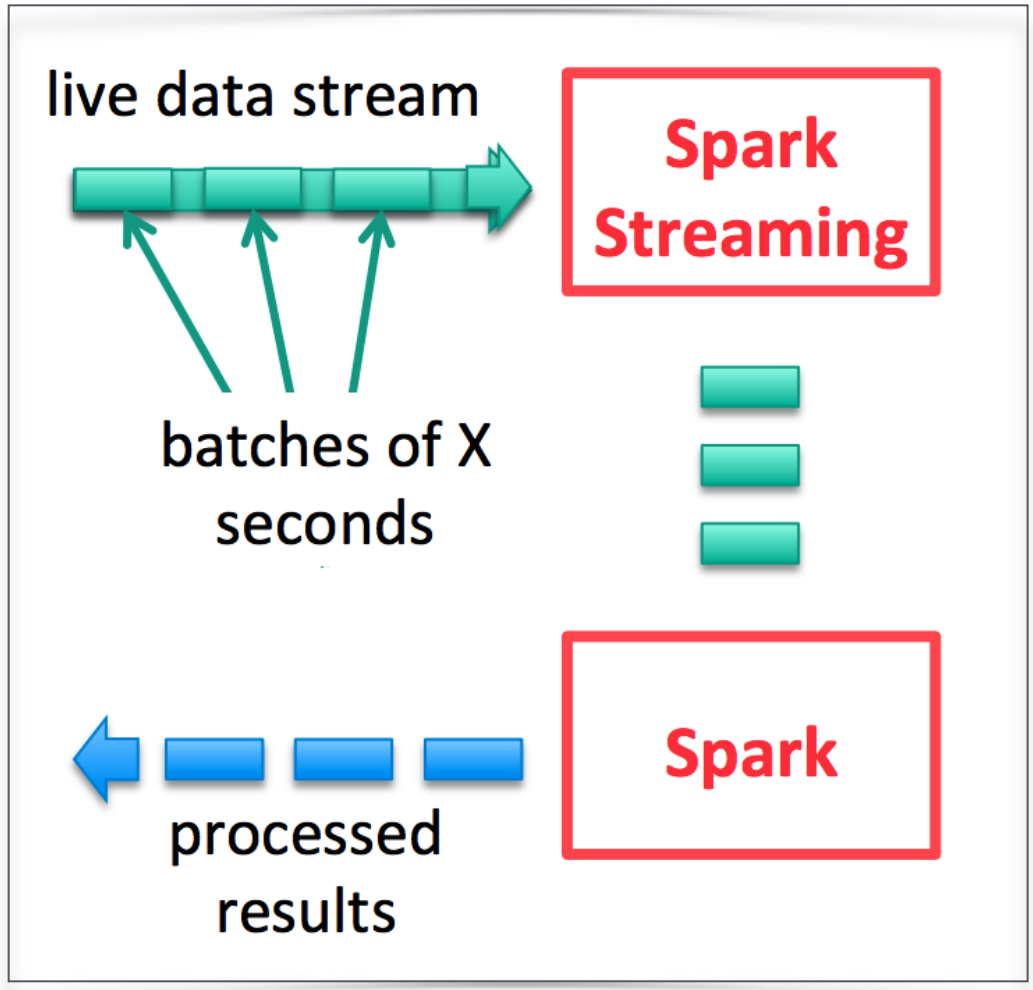
1. *Streaming analytics, especially stateful apps that leverage approximation algorithms – with demand driven by IoT*
2. *Generalize… factorizati… …ix …s*
3. *Cloud-based notebooks, building on containers, IPython, and DataFrames, as tools for collaboration and teaching – ultimately as disruptive as spreadsheets in the 1980s*

What is the story here?

Spark Streaming: *micro-batch approach*

Run a streaming computation in Spark as:
a series of very small, deterministic batch jobs

- *Chop up the live stream into batches of X seconds*
- *Spark treats each batch of data as RDDs and processes them using RDD operations*
- *Finally, the processed results of the RDD operations are returned in batches*



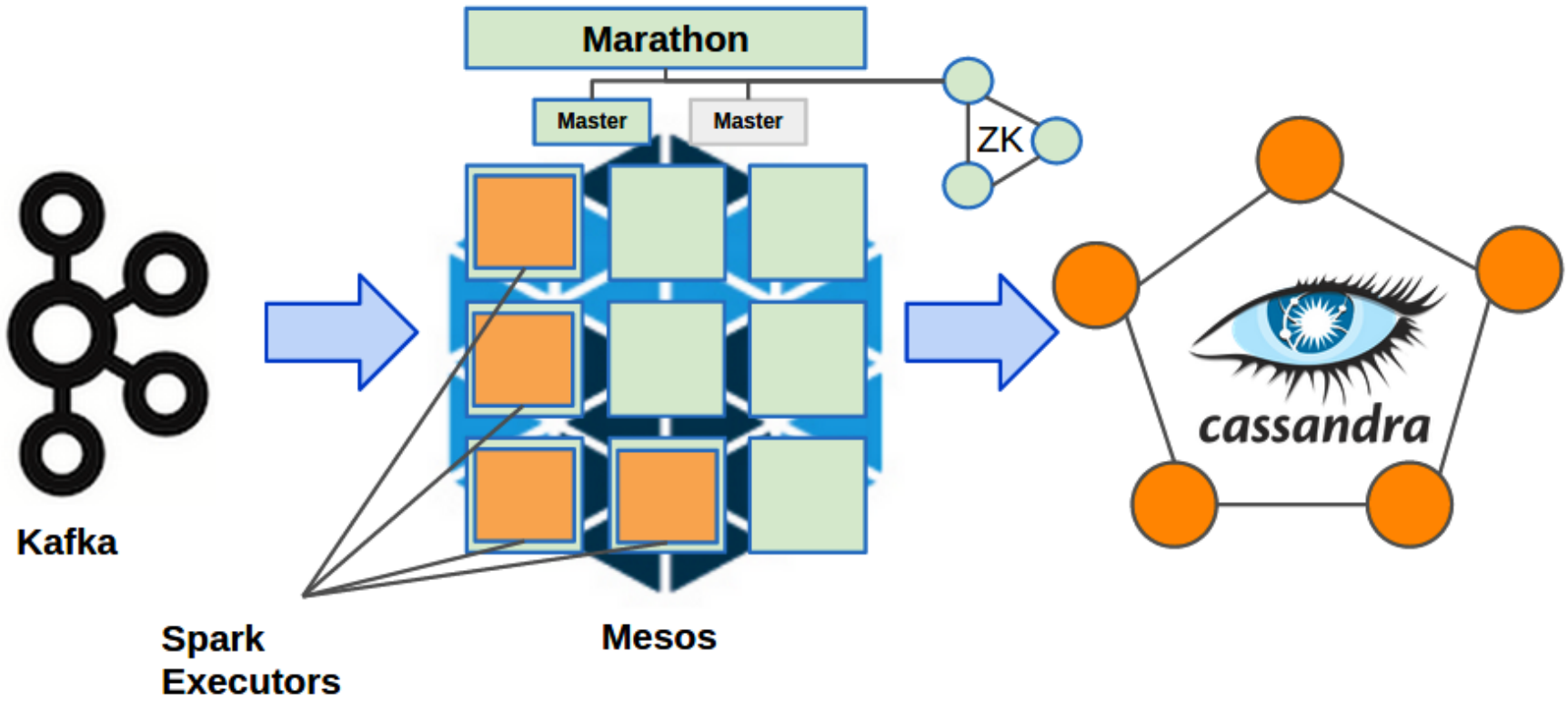
Spark Streaming: *example – stateful streaming app*

```
import sys
from pyspark import SparkContext
from pyspark.streaming import StreamingContext

def updateFunc (new_values, last_sum):
    return sum(new_values) + (last_sum or 0)

sc = SparkContext(appName="PyStreamNWC", master="local[*]")
ssc = StreamingContext(sc, 5)
ssc.checkpoint("checkpoint")

lines = ssc.socketTextStream(sys.argv[1], int(sys.argv[2]))

counts = lines.flatMap(lambda line: line.split(" ")) \
              .map(lambda word: (word, 1)) \
              .updateStateByKey(updateFunc) \
              .transform(lambda x: x.sortByKey())

counts.pprint()

ssc.start()
ssc.awaitTermination()
```

Tuning Spark Streaming for Throughput

Gerard Maas, 2014-12-22

virdata.com/tuning-spark/

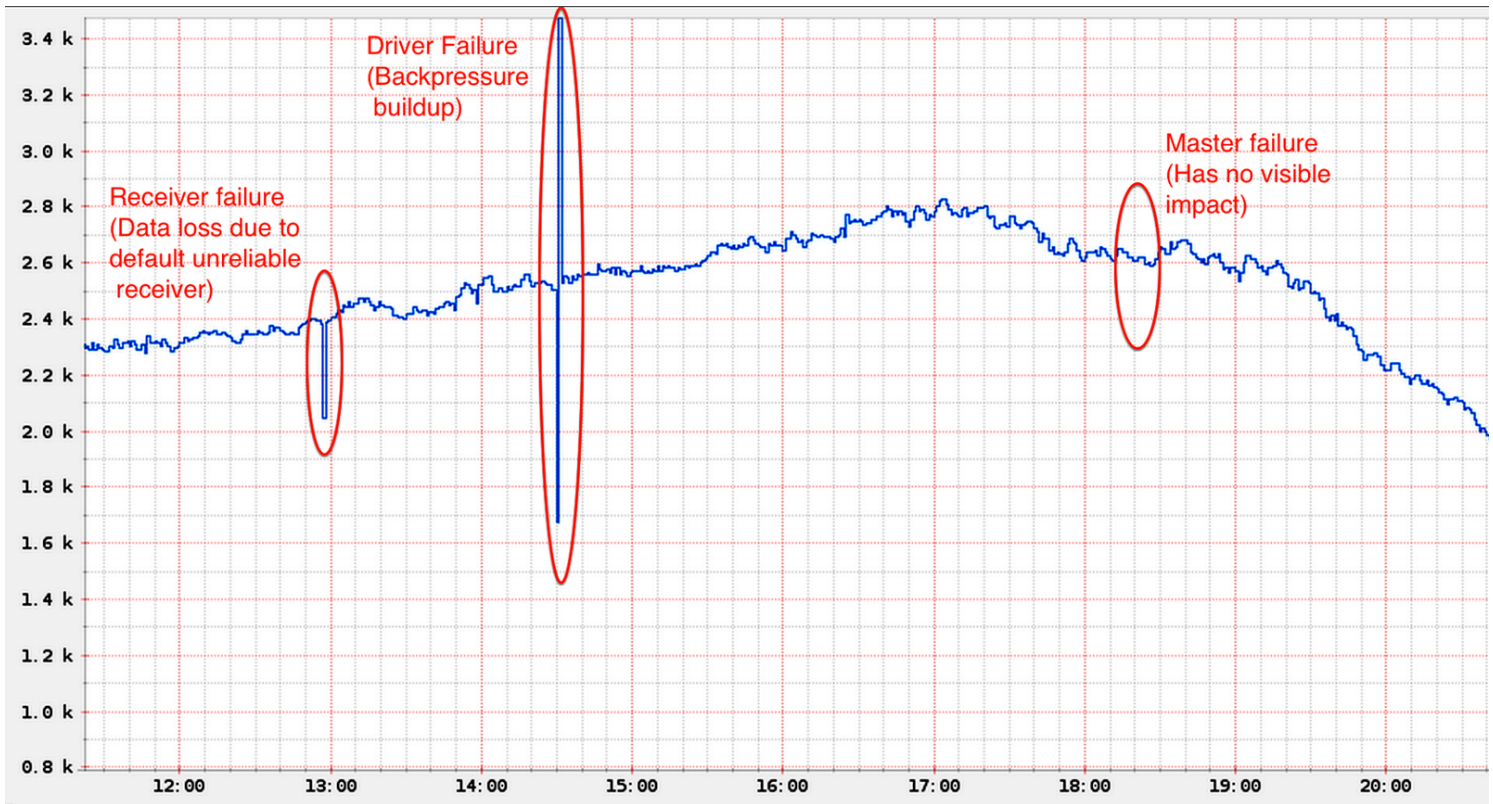


Spark Streaming: *Netflix tutorial – resiliency*

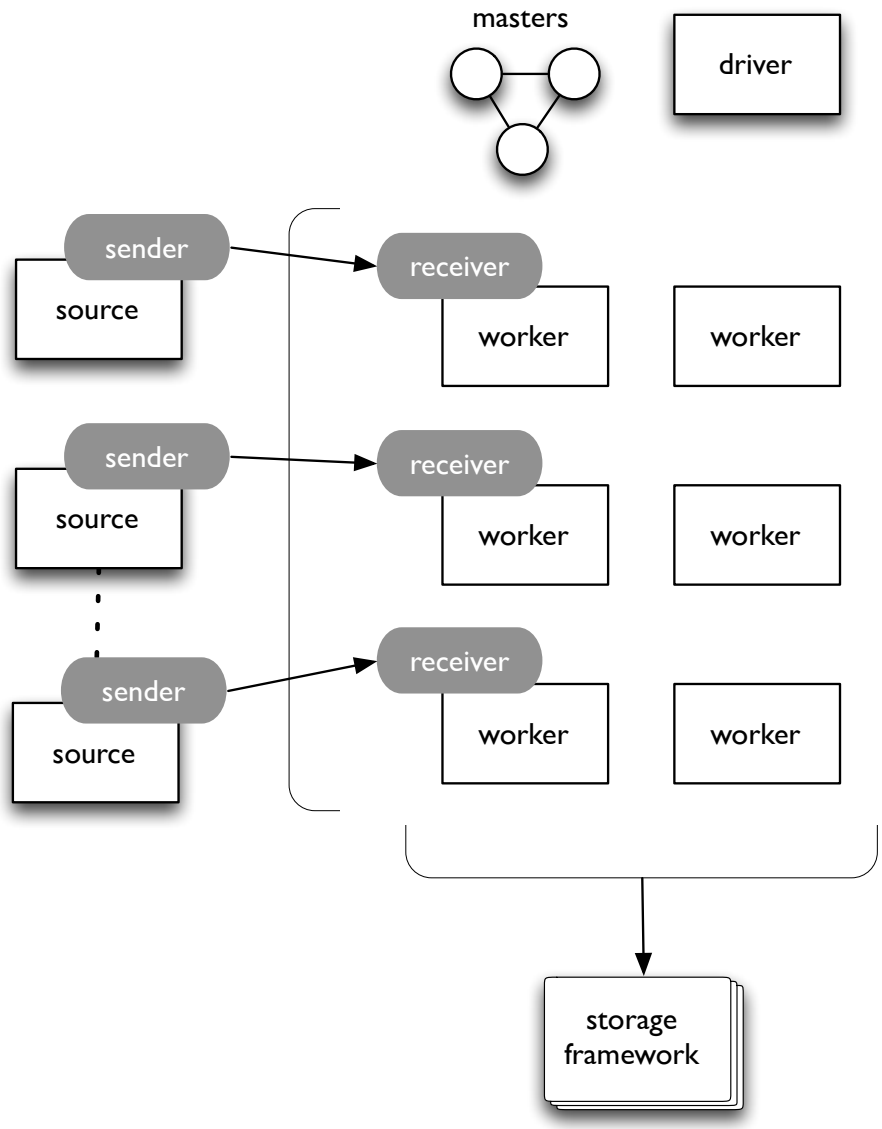
Can Spark Streaming survive Chaos Monkey?

Bharat Venkat, Prasanna Padmanabhan, Antony Arokiasamy, Raju Uppalapati

techblog.netflix.com/2015/03/can-spark-streaming-survive-chaos-monkey.html



Spark Streaming: *resiliency illustrated*



backpressure
(flow control is a hard problem)

reliable receiver

in-memory replication
write ahead log (data)

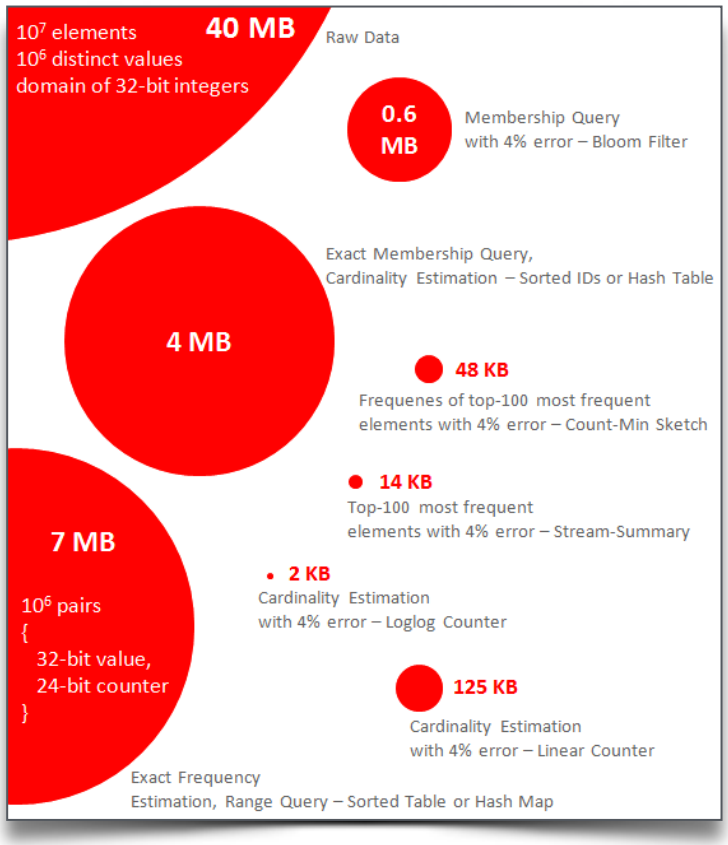
driver restart
checkpoint (metadata)

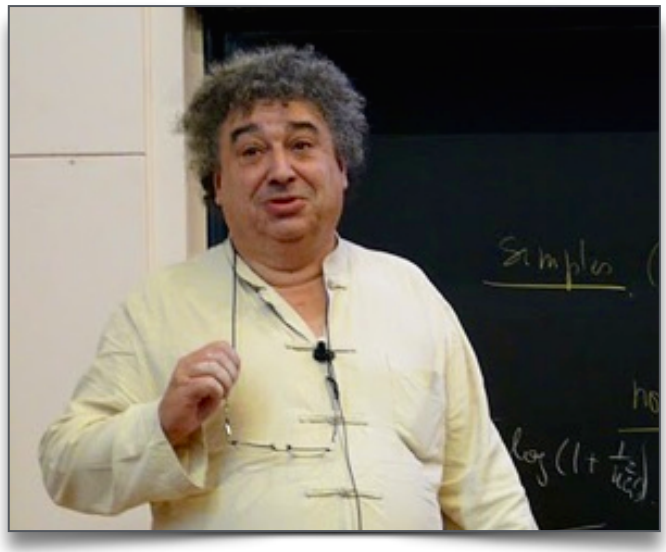
multiple masters

worker relaunch
executor relaunch

A Big Picture...

21c. shift towards modeling based on probabilistic approximations: trade bounded errors for greatly reduced resource costs

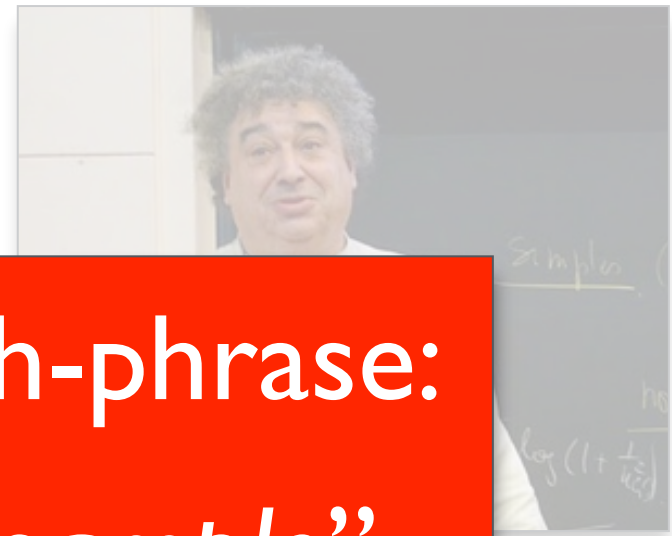




highlyscalable.wordpress.com/2012/05/01/probabilistic-structures-web-analytics-data-mining/

A Big Picture...

21c. shift towards modeling based on probabil approximations: trade bounded errors for greatly reduced resource costs



Twitter catch-phrase:

"Hash, don't sample"

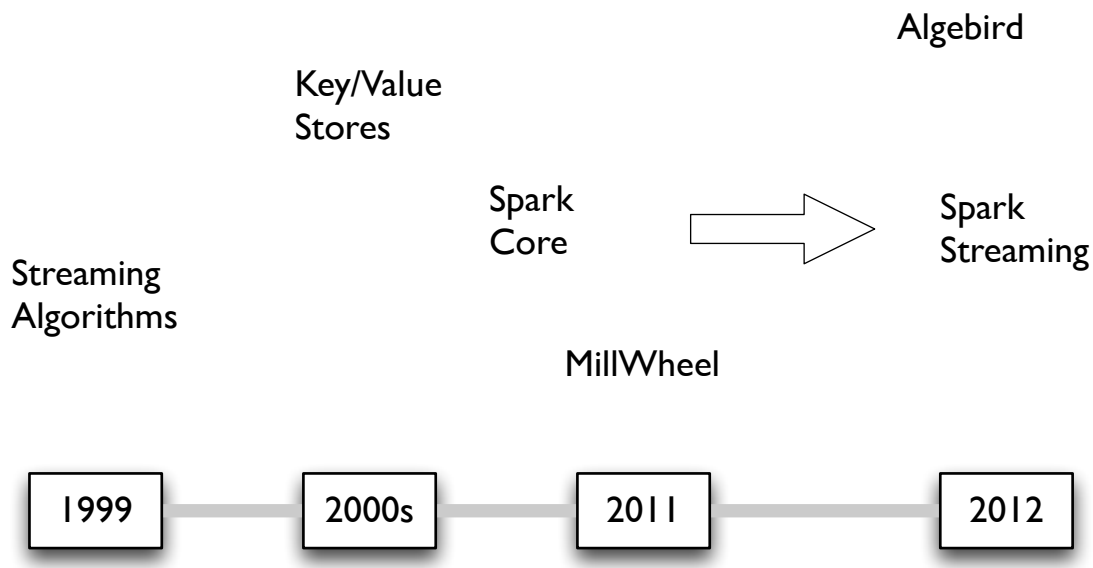
highlyscalable.wordpress.com/2012/05/01/probabilistic-structures-web-analytics-data-mining/

Streaming Algorithms: *example usage*

algorithm	*use case*	*example*
Count-Min Sketch	frequency summaries	**code**
HyperLogLog	set cardinality	**code**
Bloom Filter	set membership	
MinHash	set similarity	
DSQ	streaming quantiles	
SkipList	ordered sequence search	

Streaming Algorithms: *Themes*

This pattern of Kafka, Spark, Cassandra, and generally Mesos is sometimes called "Team Apache" – frameworks distinct from the Hadoop stack and its vendors, displacing them as industry demands real-time insights at scale, leading to an IoT "flywheel" effect…



Streaming Algorithms: *Primary Sources*

"The Space Complexity of Approximating the Frequency Moments"
Noga Alon, Yossi Matias, Mario Szegedy
JCSS 58:1, (Feb 1999), 137-147
10.1006/jcss.1997.1545

"Cassandra - A Decentralized Structured Storage System"
Avinash Lakshman, Prashant Malik
ACM SIGOPS 44:2 (Apr 2010), 35-40
10.1145/1773912.1773922

Algebird
Avi Bryant, Oscar Boykin, et al.
Twitter (2012)
engineering.twitter.com/opensource/projects/algebird

"Discretized Streams: A Fault-Tolerant Model for Scalable Stream Processing"
Matei Zaharia, Tathagata Das, et al.
Berkeley EECS (2012-12-14)
www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-259.pdf

"MillWheel: Fault-Tolerant Stream Processing at Internet Scale"
Tyler Akidau, et al.
VLDB (2013)
research.google.com/pubs/pub41378.html

Streaming Algorithms: *performance at scale*

Add ALL the Things:
Abstract Algebra Meets Analytics

infoq.com/presentations/abstract-algebra-analytics

Avi Bryant, Strange Loop (2013)

Avi Bryant
@avibryant

- *grouping doesn't matter (associativity)*
- *ordering doesn't matter (commutativity)*
- *zeros get ignored*

In other words, while partitioning data at scale is quite difficult, you can let the math allow your code to be flexible at scale

Spark Streaming: *system integrator*

Stratio Streaming: a new approach to Spark Streaming

David Morales, Oscar Mendez

2014-06-30

spark-summit.org/2014/talk/stratio-streaming-a-new-approach-to-spark-streaming

- Stratio Streaming is the union of a real-time messaging bus with a complex event processing engine using Spark Streaming
- allows creation of streams and queries on the fly
- paired with Siddhi CEP engine and Apache Kafka
- added global features to the engine such as auditing and statistics
- use cases: large banks, retail, travel, etc.
- using Apache Mesos

Spark Streaming: *neuroscience research*

hhmi | janelia Research Campus

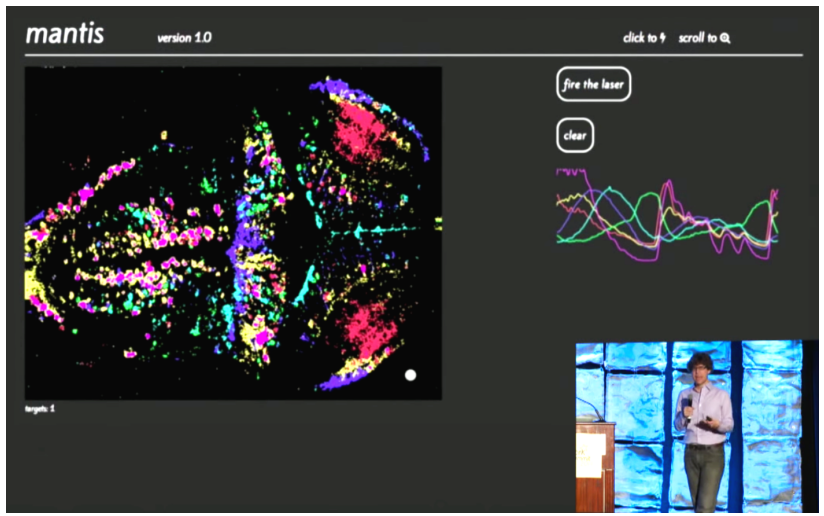
Analytics + Visualization for Neuroscience:
Spark, Thunder, Lightning

Jeremy Freeman

2015-01-29

youtu.be/cBQm4LhHn9g?t=28m55s

- neuroscience studies: zebrafish, rats, etc.
- see **http://codeneuro.org/**
- real-time ML for laser control
- 2 TB/hour per fish
- 80 HPC nodes



Spark Streaming: *geospatial analytics*

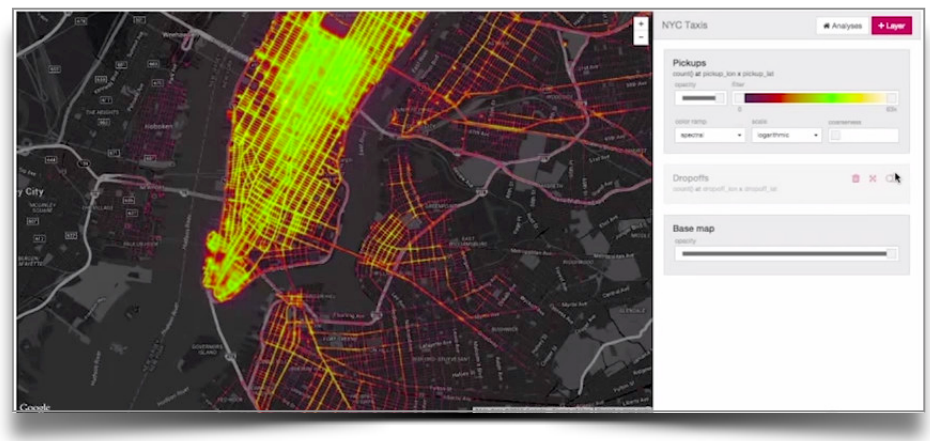
Plot all the data – interactive visualization of massive datasets

Rob Harper, **Nathan Kronenfeld**

2015-05-20

uncharted.software/spark-summit-east-2015-presentation/

- geospatial analytics – especially given sensor data, remote sensing from micro satellites, etc.
- sophisticated interactive visualization
- open source **http://aperturetiles.com/**
- see **http://pantera.io/**



Further Resources

databricks

Spark Developer Certification

- go.databricks.com/spark-certified-developer
- defined by Spark experts @Databricks
- assessed by O'Reilly Media
- establishes the bar for Spark expertise

community:

spark.apache.org/community.html

events worldwide: goo.gl/2YqJZK

YouTube channel: goo.gl/N5Hx3h

video+preso archives: spark-summit.org

resources: databricks.com/spark/developer-resources

workshops: databricks.com/spark/training

MOOCs:

Anthony Joseph
UC Berkeley
begins Jun 2015
edx.org/course/uc-berkeleyx/uc-berkeleyx-cs100-1x-introduction-big-6181

Introduction to Big Data with Apache Spark

Learn how to apply data science techniques using parallel programming in Apache Spark to explore big (and small) data.

Scalable Machine Learning

Learn the underlying principles required to develop scalable machine learning pipelines and gain hands-on experience using Apache Spark.

Ameet Talwalkar
UCLA
begins Jun 2015
edx.org/course/uc-berkeleyx/uc-berkeleyx-cs190-1x-scalable-machine-6066

confs:

Strata EU
London, May 5-7
strataconf.com/big-data-conference-uk-2015

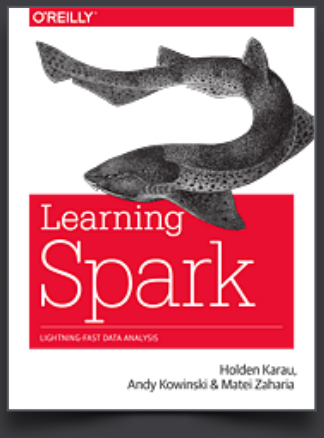
GOTO Chicago
Chicago, May 11-14
gotocon.com/chicago-2015

Spark Summit 2015
SF, Jun 15-17
spark-summit.org

Spark Summit EU
(to be announced)

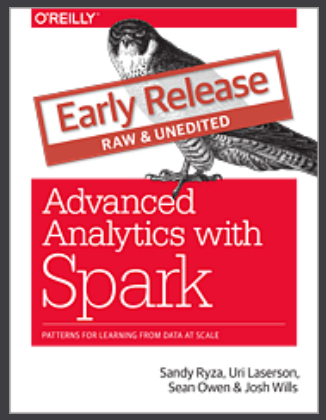
books+videos:

Learning Spark
Holden Karau, Andy Konwinski, Parick Wendell, Matei Zaharia
O'Reilly (2015)
shop.oreilly.com/product/06369200285l2.do



Intro to Apache Spark
Paco Nathan
O'Reilly (2015)
shop.oreilly.com/product/0636920036807.do

Advanced Analytics with Spark
Sandy Ryza, Uri Laserson, Sean Owen, Josh Wills
O'Reilly (2014)
shop.oreilly.com/product/0636920035091.do



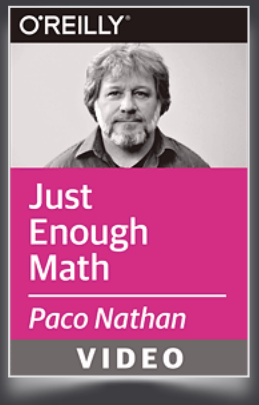
Spark in Action
Chris Fregly
Manning (2015)
sparkinaction.com/

Fast Data Processing with Spark
Holden Karau
Packt (2013)
shop.oreilly.com/product/9781782167068.do

presenter:

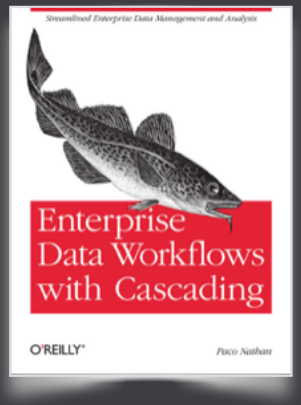
monthly newsletter for updates, events, conf summaries, etc.:

liber118.com/pxn/



Just Enough Math
O'Reilly, 2014

justenoughmath.com
preview: youtu.be/TQ58cWgdCpA



Enterprise Data Workflows with Cascading
O'Reilly, 2013

shop.oreilly.com/product/0636920028536.do

Moitas graciñas
¿Preguntas?