Emerging Trends in Big Data Software

(via the Berkeley Data Analytics Stack)

Michael Franklin
24 March 2016
Paris Database Summit

amplab
UC BERKELEY

Big Data – A Bad Definition

Data sets, typically consisting of billions or trillions of records, that are so vast and complex that they require new and powerful computational resources to process.

- Dictionary.com

Big Data as a Resource

"For a long time, we thought that Tamoxifen was roughly 80% effective for breast cancer patients. But now we know much more:

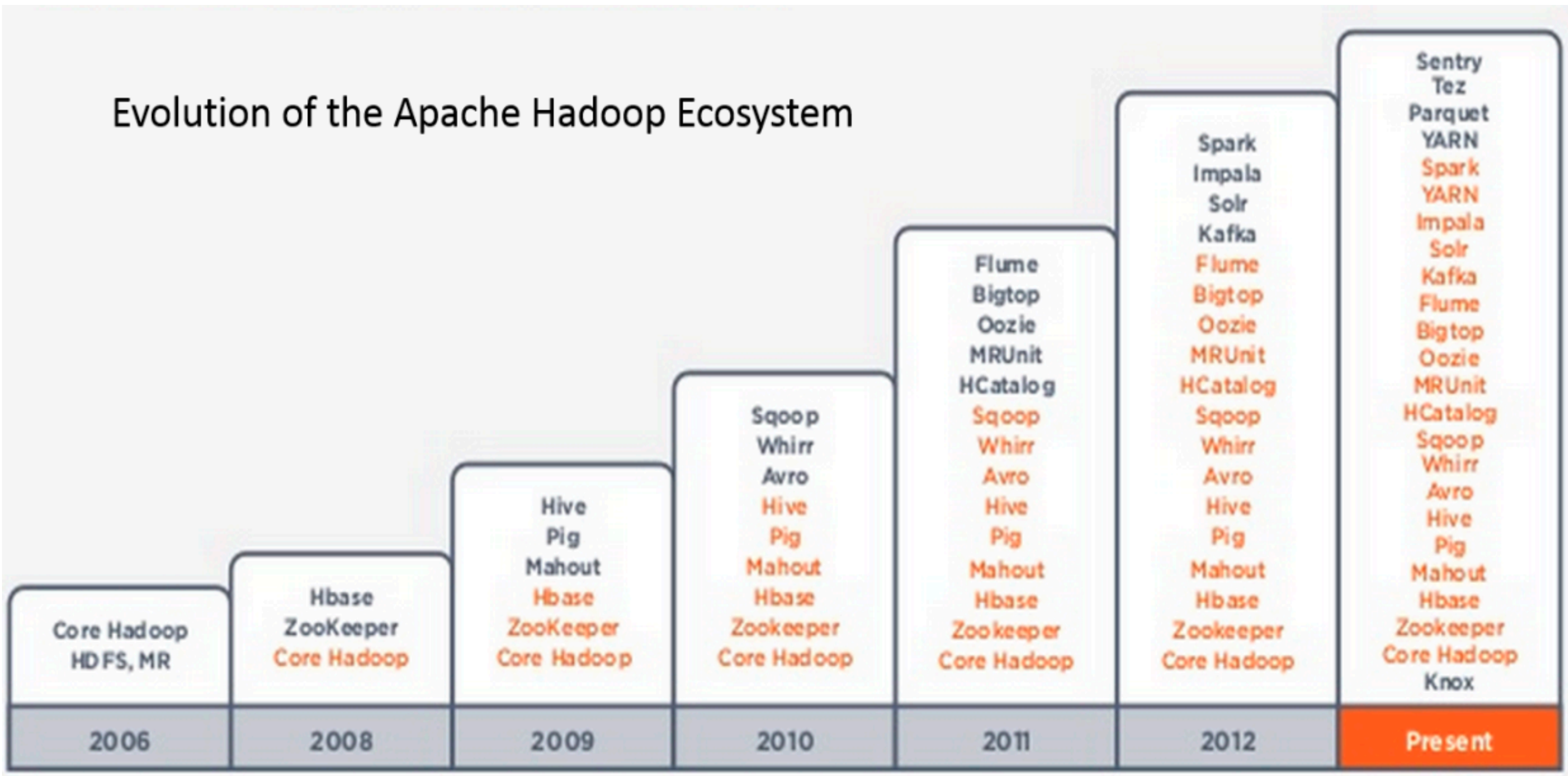
we know that it's 100% effective in 70% to 80% of the patients, and ineffective in the rest."

- Tim O'Reilly et al. "How Data Science is Transforming Health Care" 2012

With enough of the right data you can determine precisely who the treatment will work for.

Thus, even a 1% effective drug could save lives

Big Data Software Growth



...

Previously: 1980's & 90's: Parallel Database systems;
early 2000's Google MapReduce

Regardless of your definition…

The technology has fundamentally changed.

- Massively scalable processing and storage
- Pay-as-you-go processing and storage
- Flexible schema on read vs. schema on write
- Easier integration of search, query and analysis
- Variety of languages for interface/interaction
- Open source ecosystem driving innovation

Big Data Software Moves Fast

We'll look at the following trends:

1) Integrated Stacks vs Silos
2) "Real-Time" Redux
3) Machine Learning and Advanced Analytics
4) Serving Data and Models
5) Big Data Software + X <HPC, People, IoT,…>

Trend 1

INTEGRATED STACKS VS SILOS

AMPLab: A Public/Private Partnership

Launched 2011; ~90 Students, Postdocs, and Faculty

Scheduled to run through 2016

National Science Foundation Expedition Award

Darpa XData; DoE/Lawrence Berkeley National Lab

Industrial Sponsors:

AMP: 3 Key Resources

Algorithms
- Machine Learning, Statistical Methods
- Prediction, Business Intelligence

Machines
- Clusters and Clouds
- Warehouse Scale Computing

People
- Crowdsourcing, Human Computation
- Data Scientists, Analysts

Berkeley Data Analytics Stack

In House Applications – Genomics, IoT, Energy

Sample Clean | BlinkDB | GraphX | MLbase

Access and Interfaces

Spark Streaming | Spark SQL | Spark + R | KeystoneML

Spark | Processing Engine | VELOX

TACHYON | Storage

MESOS | Resource Virtualization

Apache Spark Meetups (Jan 2015)

Groups	Members	Interested	Cities	Countries
43	12,039	828	35	14

Apache Spark Meetups (Dec 2015)

Groups	Members	Interested	Cities	Countries
126	52,643	1,441	86	36

+ 193%

+339%

This Morning: 166 groups with 75739 members
(including 3 groups in Africa)

2015: Typical Spark Coverage

November 4, 2015

Skip the Ph.D and Learn Spark, Data Science Salary Survey Says

Alex Woodie

Prospective data scientists can boost their salary more by learning Apache Spark and its tied-at-the-hip language Scala than obtaining a Ph.D., a recent data science survey by O'Reilly suggests.

Big Data Ecosystem Evolution

MapReduce

Pregel Giraph

Dremel Drill Tez

Impala GraphLab

Storm S4 …

General batch processing

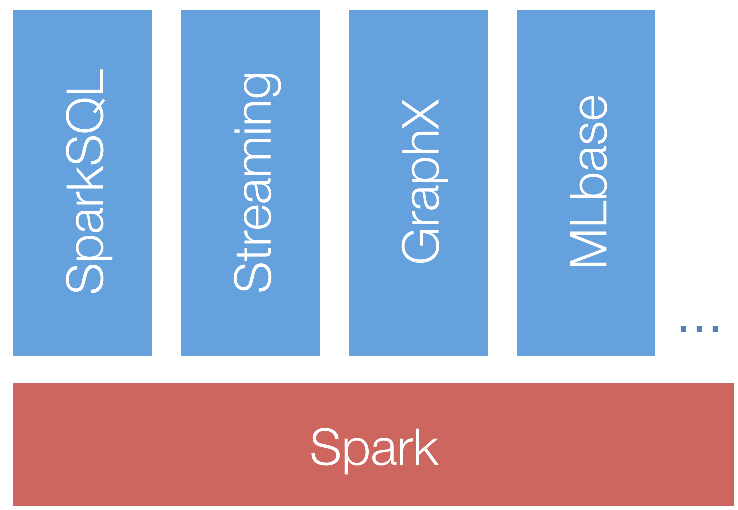
Specialized systems
(iterative, interactive and streaming apps)

AMPLab Unification Strategy

Don't specialize MapReduce – Generalize it!

Two additions to Hadoop MR:

1. General Task DAGs
2. Data Sharing



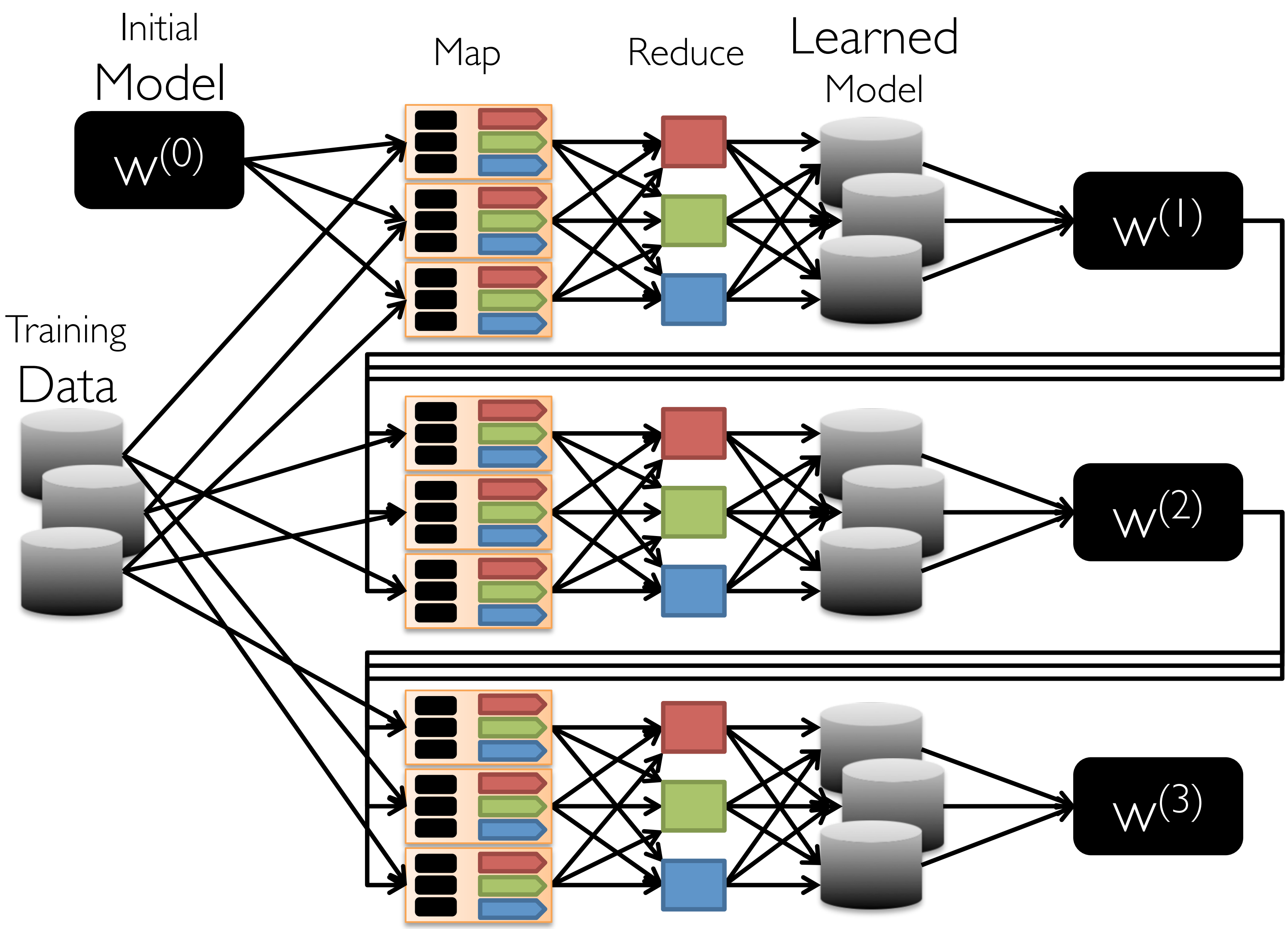
Productivity:

Fewer Systems to Master

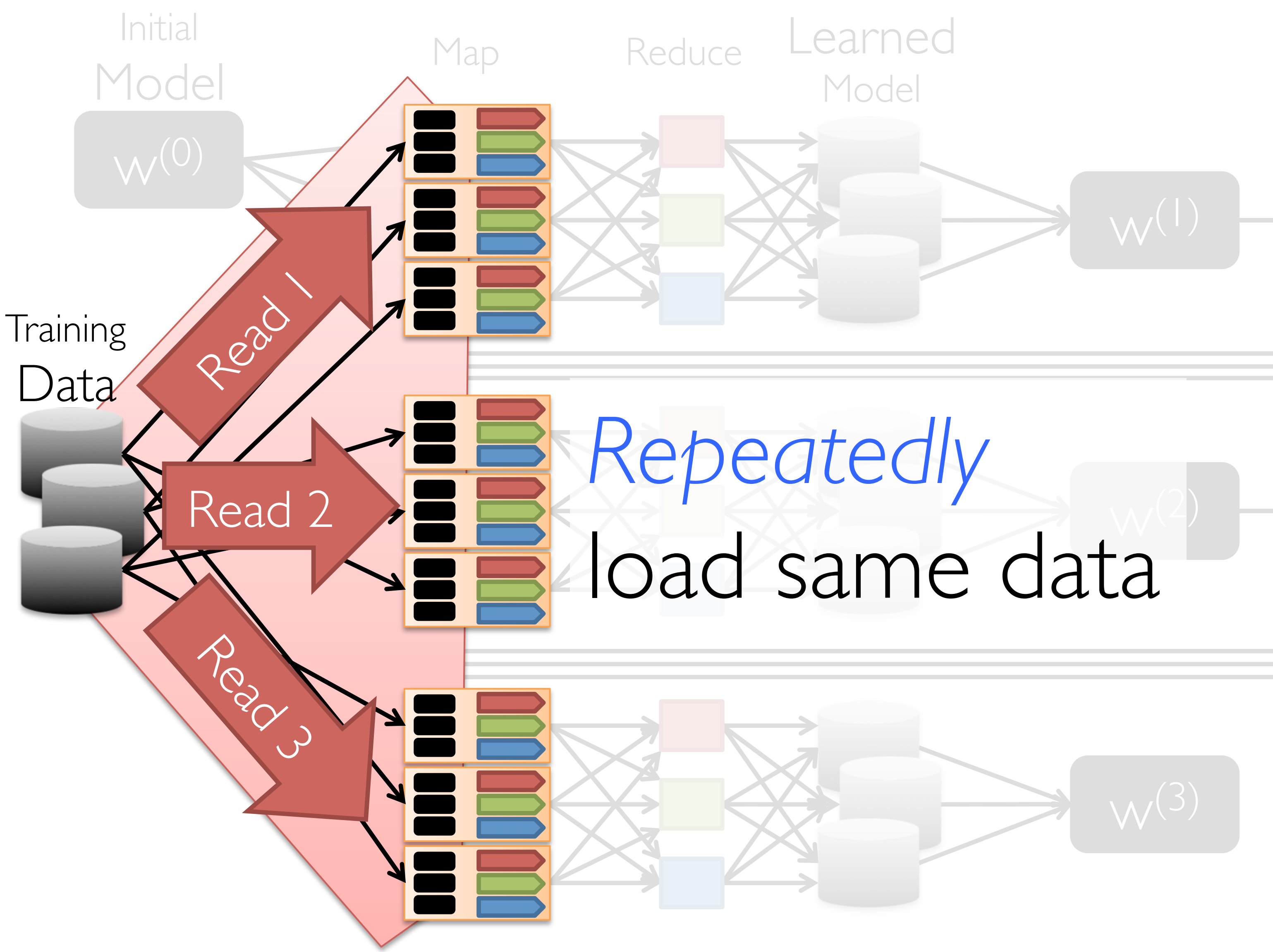
Less Data Movement

M. Zaharia, M. Choudhury, M. Franklin, I. Stoica, S. Shenker, "Spark: Cluster Computing with Working Sets, USENIX HotCloud, 2010.

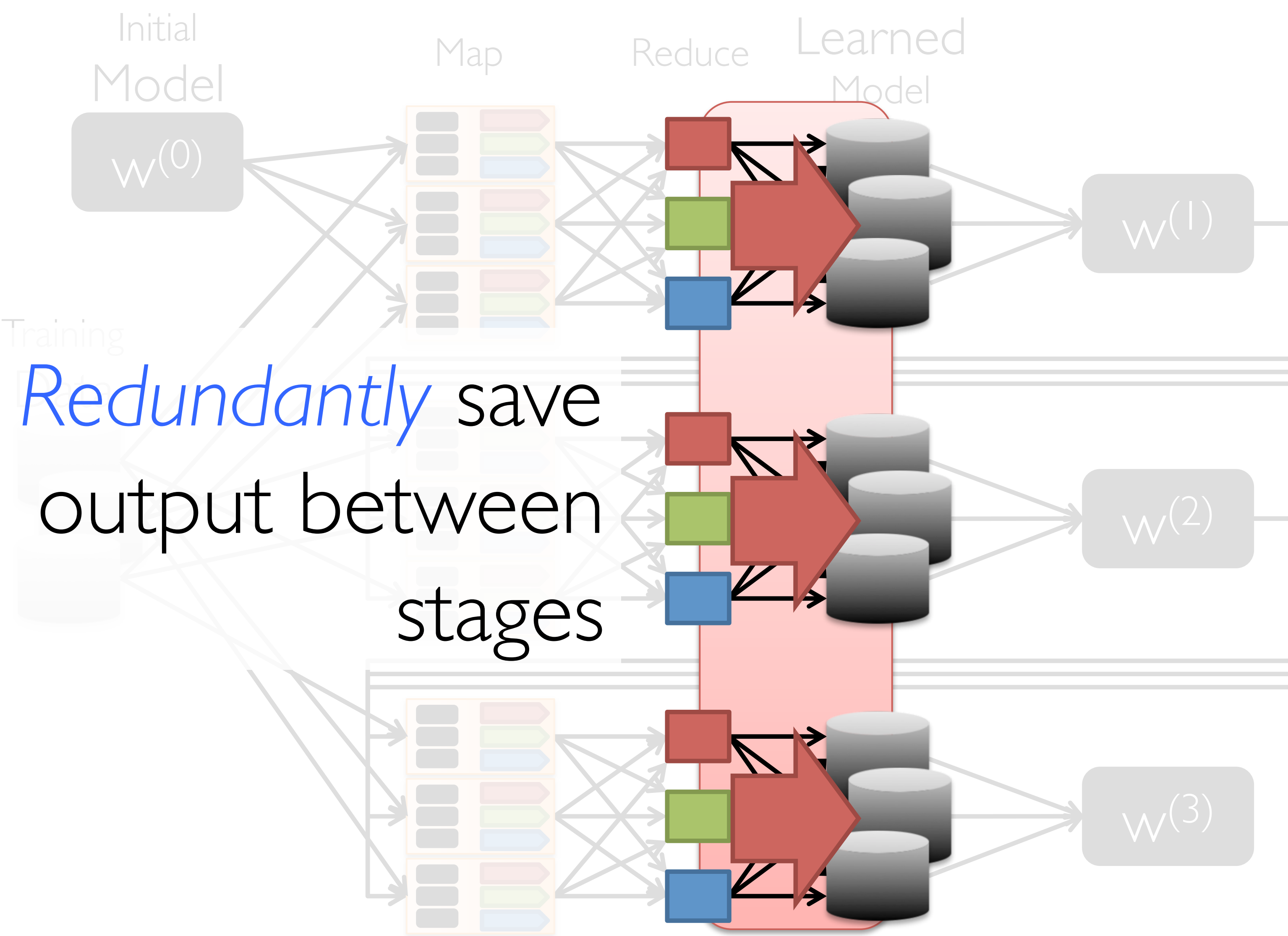
Iteration in Map-Reduce



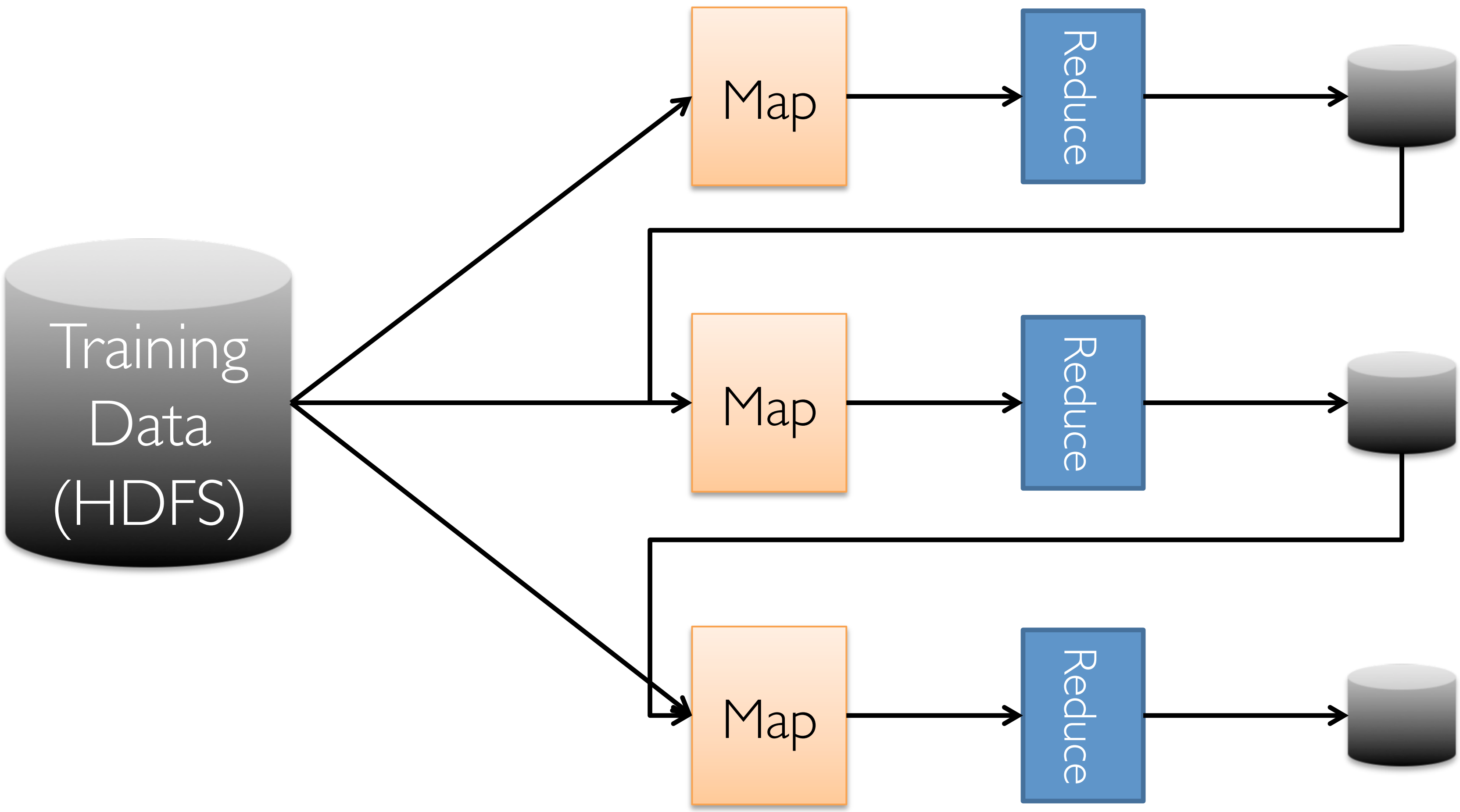
Cost of Iteration in Map-Reduce



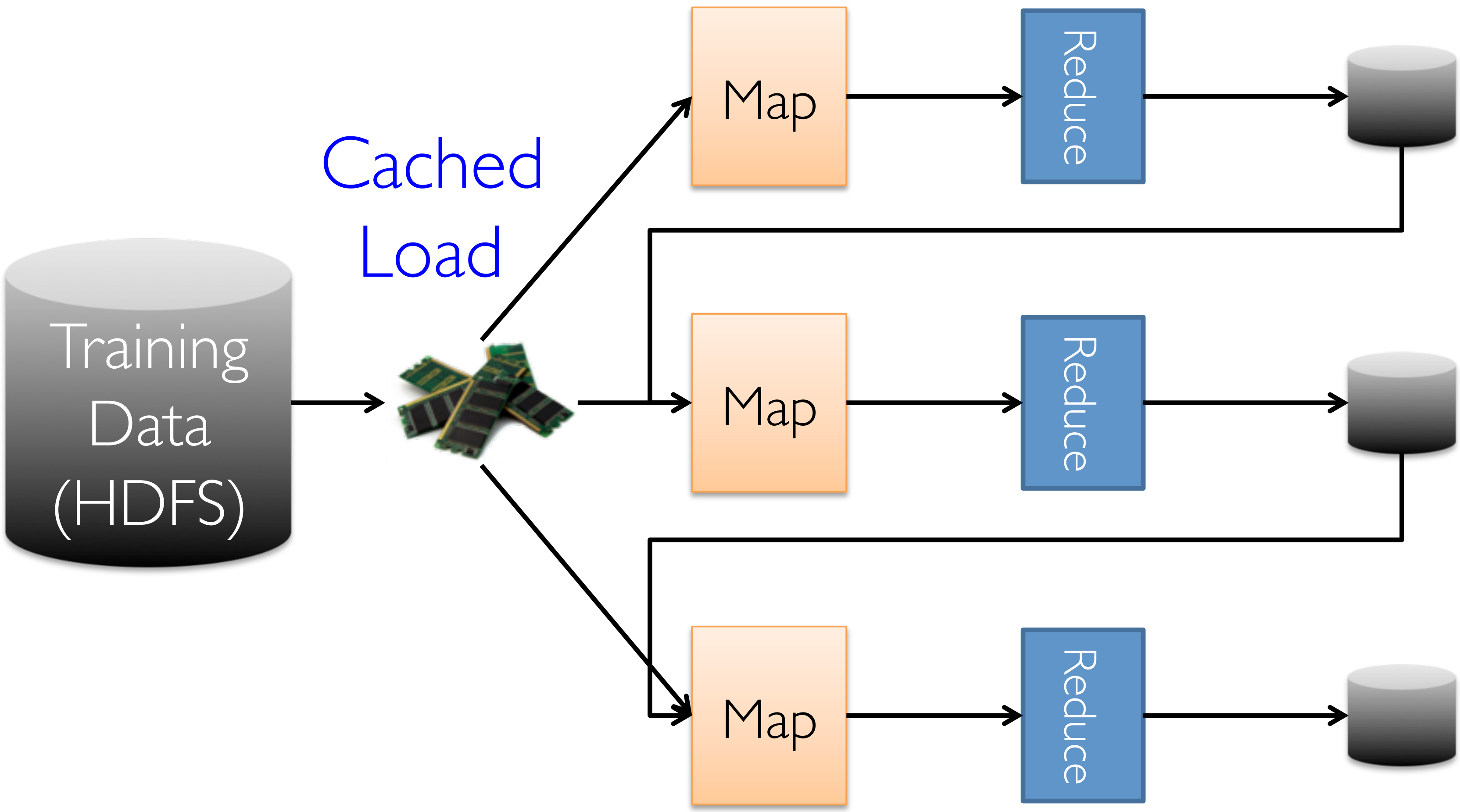
Cost of Iteration in Map-Reduce



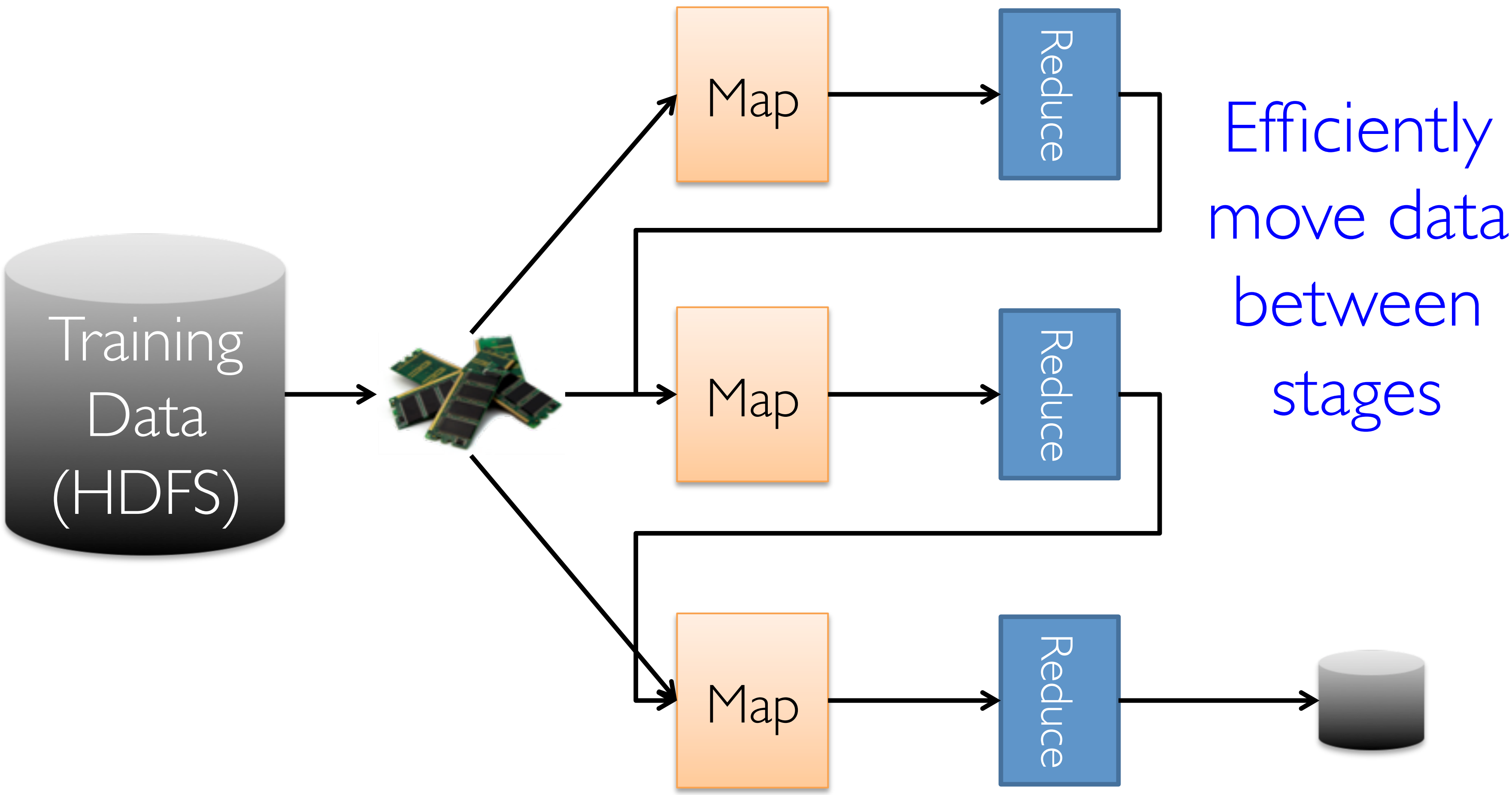
Dataflow View



Memory Opt. Dataflow



Memory Opt. Dataflow View



Spark: 10-100× faster than Hadoop MapReduce

Resilient Distributed Datasets (RDDs)

API: coarse-grained *transformations* (map, group-by, join, sort, filter, sample,…) on immutable collections

Resilient Distributed Datasets (RDDs)

» Collections of objects that can be stored in memory or disk across a cluster
» Built via parallel transformations (map, filter, …)
» Automatically rebuilt on failure

Rich enough to capture many models:

» **Data flow models**: MapReduce, Dryad, SQL, …
» **Specialized models**: Pregel, Hama, …

M. Zaharia, et al, Resilient Distributed Datasets: A fault-tolerant abstraction for in-memory cluster computing, NSDI 2012.

Abstraction: *Dataflow Operators*

`map`	`reduce`	`sample`
`filter`	`count`	`take`
`groupBy`	`fold`	`first`
`sort`	`reduceByKey`	`partitionBy`
`union`	`groupByKey`	`mapWith`
`join`	`cogroup`	`pipe`
`leftOuterJoin`	`cross`	`save`
`rightOuterJoin`	`zip`	`...`

Fault Tolerance with RDDs

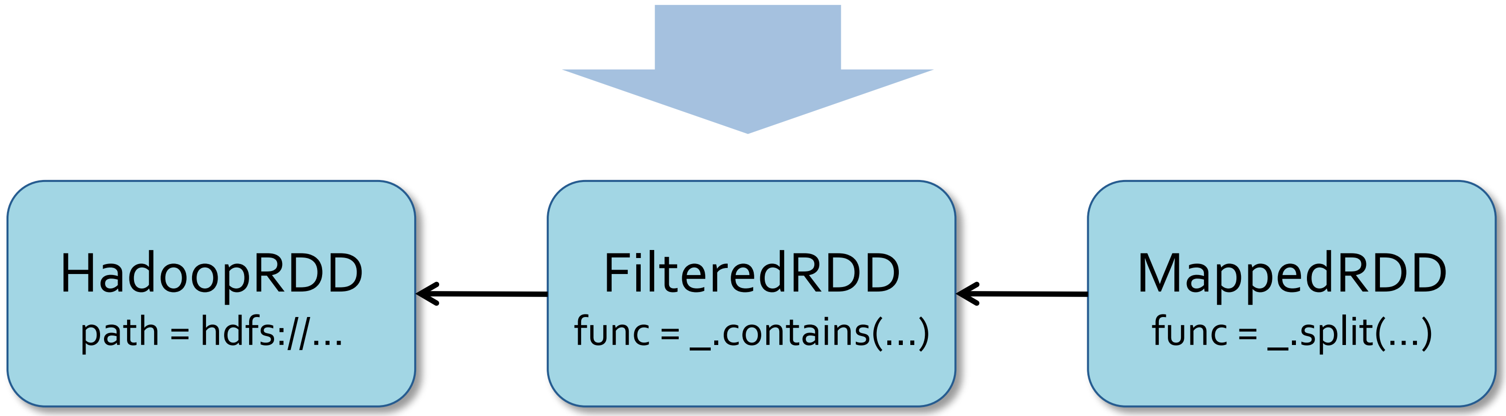
RDDs track the series of transformations used to build them (their *lineage*)

» Log one operation to apply to many elements

» No cost if nothing fails

Enables per-node recomputation of lost data

```
messages = textFile(...).filter(_.contains("error"))
                        .map(_.split('\t')(2))
```



Spark SQL – Deeper Integration

Replaces "Shark" – Spark's implementation of Hive

- Hive dependencies were cumbersome
- Missed integration opportunities

Spark SQL has two main additions

1) Tighter Spark integration, including Data Frames
2) Catalyst Extensible Query Optimizer

First release May 2014; in production use

- e.g., large Internet co has deployed on 8000 nodes; >100PB with typical queries covering 10's of TB

R. Xin, J. Rosen, M. Zaharia, M. Franklin,S. Shenker, I. Stoica, "Shark: SQL and Rich Analytics at Scale, SIGMOD 2013.

M. Armbrust, R. Xin et al., "Spark SQL: Relational Data Processing in Spark", SIGMOD 2015.

SQL + ML + Streaming

```
// Load historical data as an RDD using Spark SQL
val trainingData = sql(
  "SELECT location, language FROM old_tweets")

// Train a K-means model using MLlib
val model = new KMeans()
  .setFeaturesCol("location")
  .setPredictionCol("language")
  .fit(trainingData)

// Apply the model to new tweets in a stream
TwitterUtils.createStream(...)
  .map(tweet => model.predict(tweet.location))
```

DataFrames

```
employees
.join(dept, employees("deptId") === dept("id"))
.where(employees("gender") === "female")
.groupBy(dept("id"), dept("name"))
.agg(count("name"))
```

Some people think this is an improvement over SQL ☺

Recently added: a binding for R dataframes

Catalyst Optimizer

Extensibility via Optimization Rules written in Scala

Code generation for inner-loops

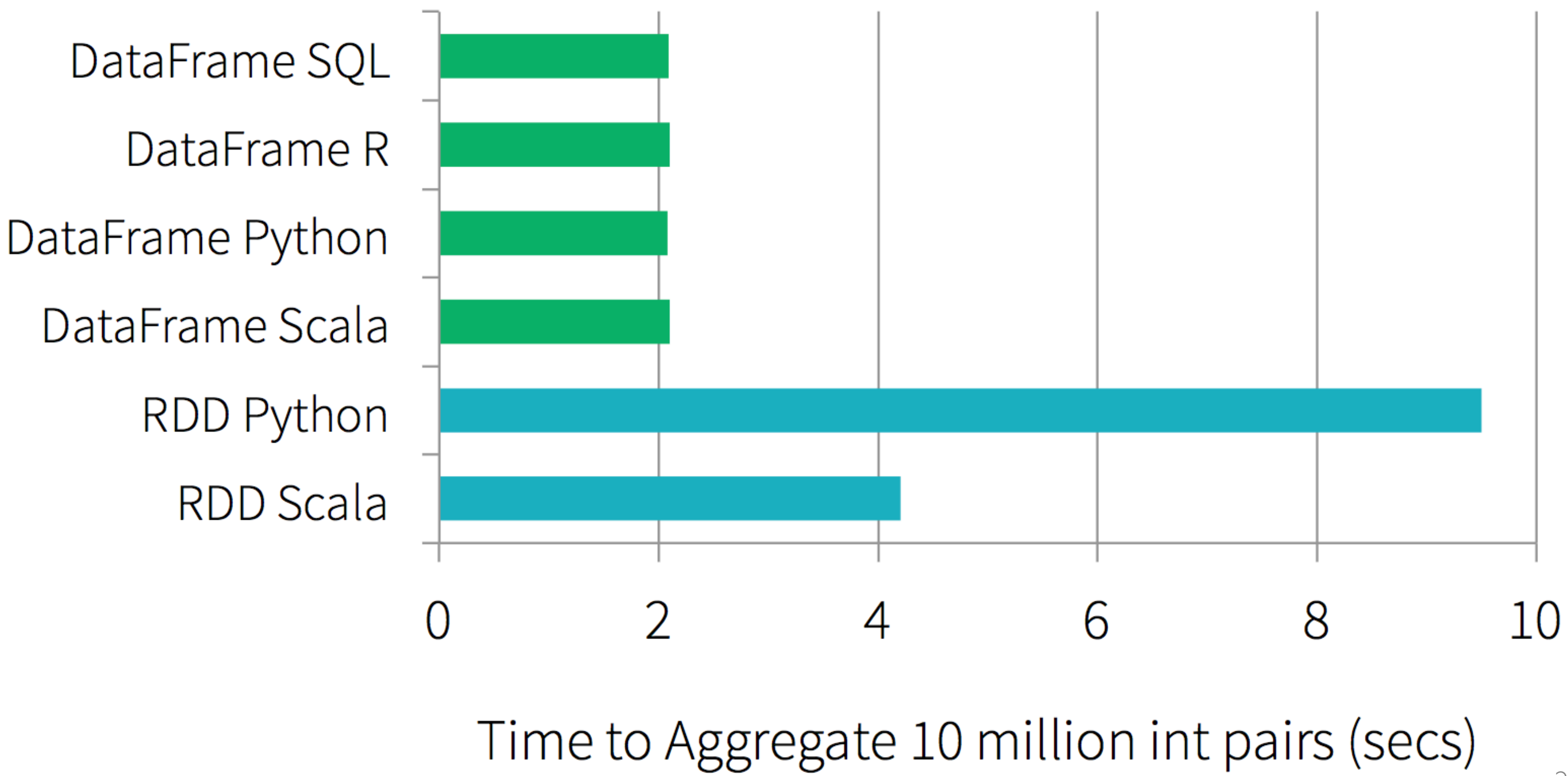
Extension Points:

Data Sources: e.g., CSV, Avro, Parquet, JDBC, …

- via TableScan (all cols), PrunedScan (project), FilteredPrunedScan(push advisory selects and projects) CatalystScan (push advisory full Catalyst expression trees)

User Defined Types

An interesting thing about SparkSQL Performance



JSON Type Inference

```
{
"text": "This is a tweet about #Spark",
"tags": ["#Spark"],
"loc": {"lat": 45.1, "long": 90}
}
{
"text": "This is another tweet",
"tags": [],
"loc": {"lat": 39, "long": 88.5}
}
{
"text": "A #tweet without #location",
"tags": ["#tweet", "#location"]
}
```

```
text STRING NOT NULL ,
tags ARRAY<STRING> NOT NULL,
loc STRUCT<lat FLOAT NOT NULL,
           long FLOAT NOT NULL>
```

Currently can also do Type Inference for Python RDDs; CSV and XML in progress

Query Federation made Easy?

A join of a MySQL Table and a JSON file using Spark SQL

```
CREATE TEMPORARY TABLE users
USING jdbc
OPTIONS(driver "mysql" url "jdbc:mysql://userDB/users")
```

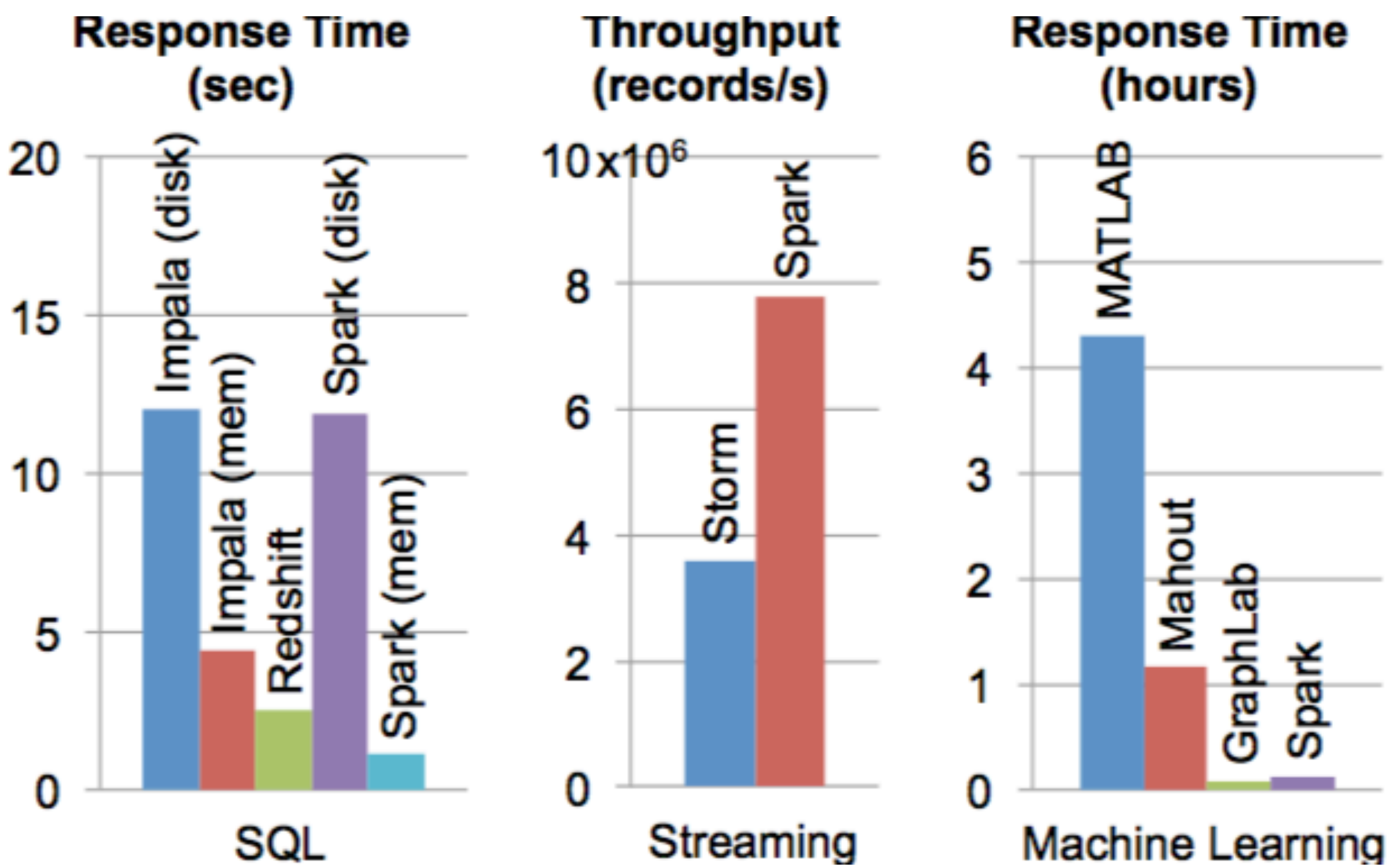
Don't Forget About Approximation

BDAS Uses Approximation in two main ways:

1) BlinkDB (Agarwal et al. EuroSys 13)
    - Run queries on a sample of the data
    - Returns answer and confidence interval
    - Can adjust time vs confidence

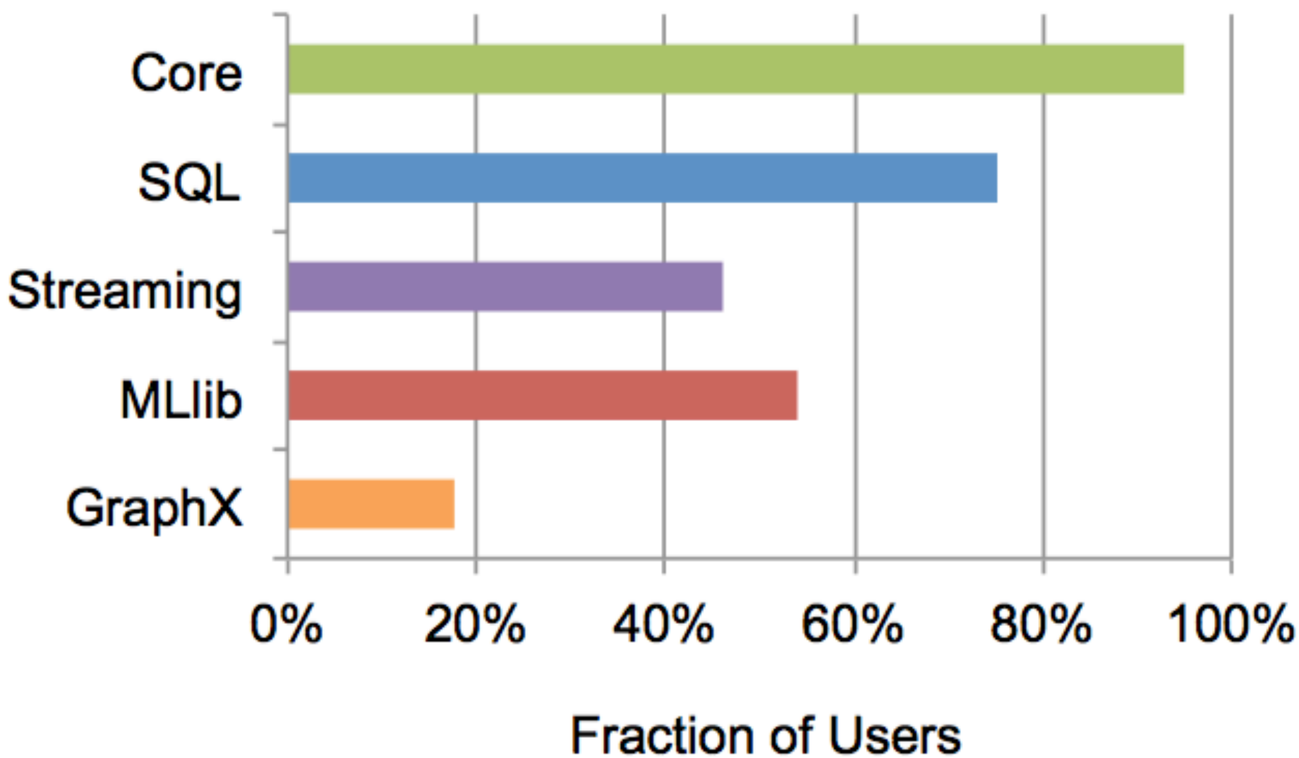
2) Sample Clean (Wang et al. SIGMOD 14)
    - Clean a sample of the data rather than whole data set
    - Run query on sample (get error bars) OR
    - Run query on dirty data and correct the answer

Performance vs. Specialized



Zaharia et al., "Spark: Building a Unified Engine for Big Data Processing", CACM 2016 to appear

Spark User Survey 7/2015 (One Size Fits Many)



~1400 respondents; 88% Use at least 2 components; 60% at least 3; 27% at least 4;
Source: Databricks

Trend II

"REALTIME" REDUX

Renewed Excitement Around Streaming

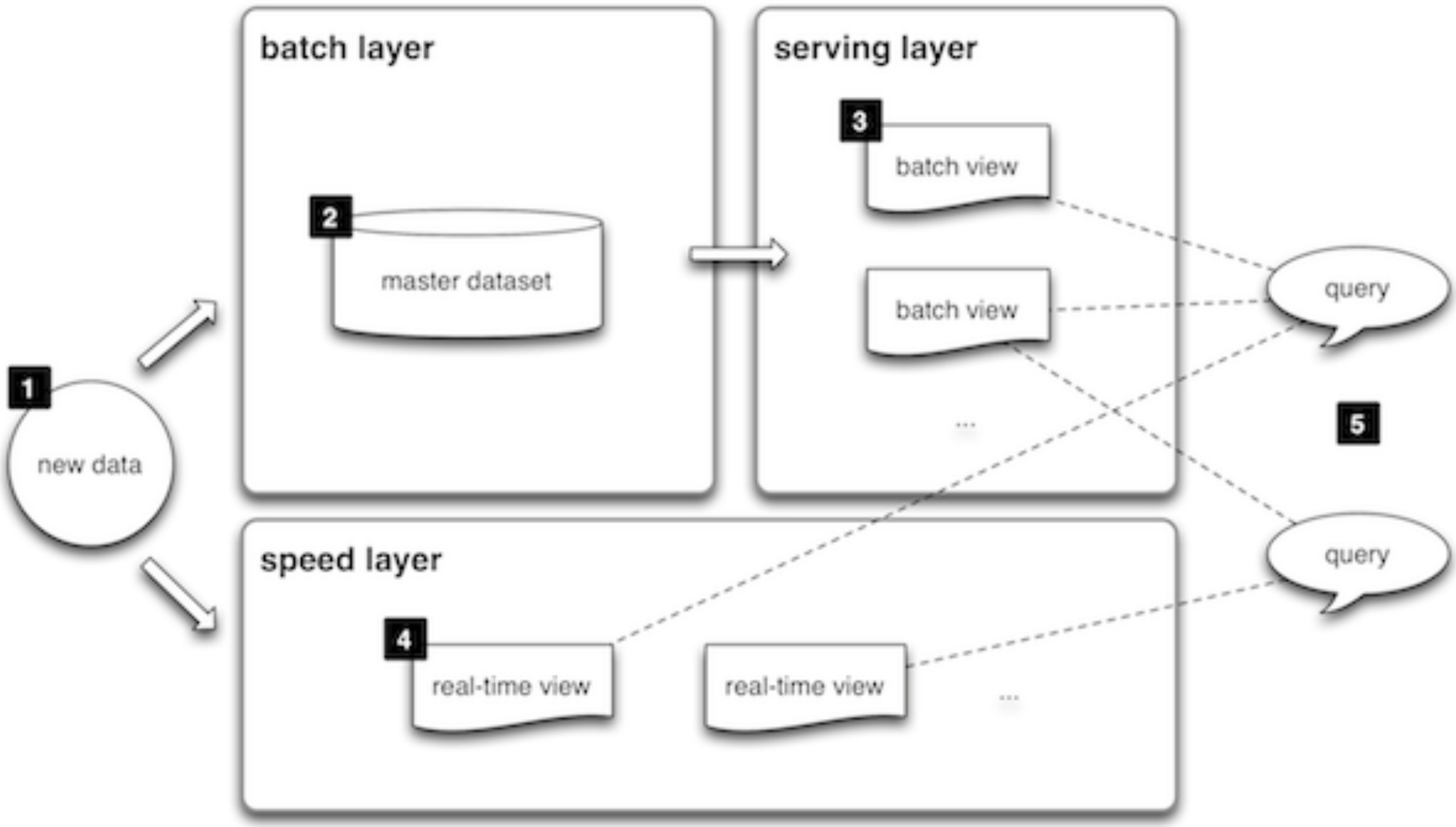
Stream Processing (esp. Open Source)

- » Spark Streaming
- » Samza
- » Storm
- » Flink Streaming
- » Google Millwheel and Cloud Dataflow
- » <YOUR FAVORITE SYSTEM HERE>

Message Transport

- » Kafka
- » Kenesis
- » Flume

Lambda Architecture: Real-Time + Batch



lambda-architecture.net

Lambda: How Unified Is It?

Have to write everything twice!

Have to fix everything (maybe) twice.

Subtle differences in semantics

how much Duct Tape required?

What about Graphs, ML, SQL, etc.?

see e.g., Jay Kreps: http://radar.oreilly.com/2014/07/questioning-the-lambda-architecture.html
and Franklin et al., CIDR 2009.

Spark Streaming

Scalable, fault-tolerant stream processing system

High-level API

joins, windows, …
often 5x less code

Fault-tolerant

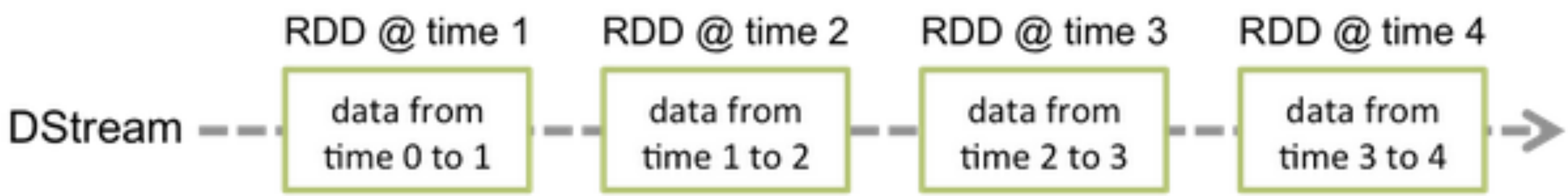
Exactly-once semantics, even for stateful ops

Integration

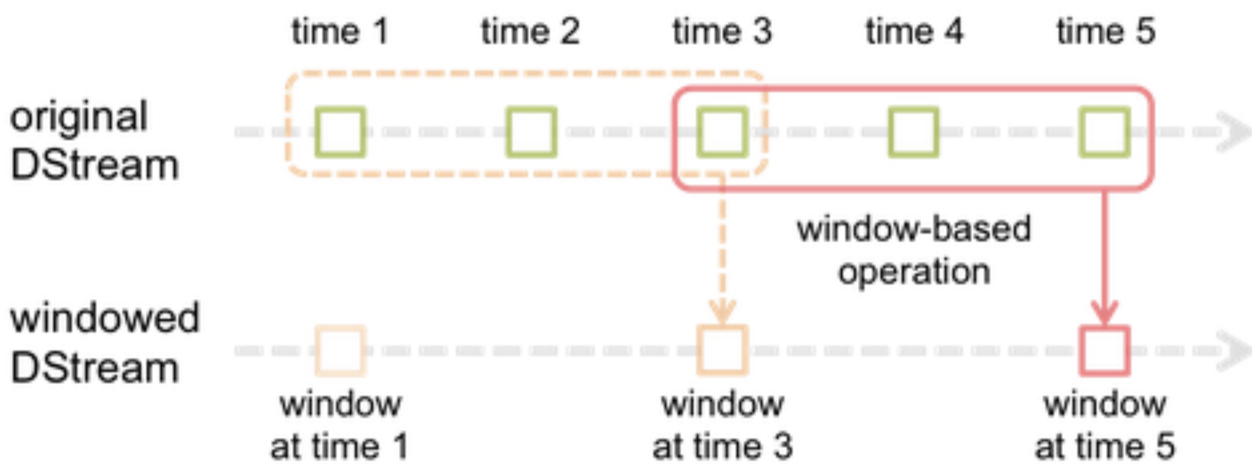
Integrate with MLlib, SQL, DataFrames, GraphX

Spark Streaming

Microbatch approach provides low latency



Additional operators provide windowed operations



M. Zaharia, et al, Discretized Streams: Fault-Tollerant Streaming Computation at Scale, SOSP 2013.

Batch/Streaming Unification

Batch and streaming codes virtually the same

» Easy to develop and maintain consistency

```
// count words from a file (batch)
val file = sc.textFile("hdfs://.../pagecounts-*.gz")
val words = file.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
wordCounts.print()
```

```
// count words from a network stream, every 10s (streaming)
val ssc = new StreamingContext(args(0), "NetCount", Seconds(10), ..)
val lines = ssc.socketTextStream("localhost", 3456)
val words = lines.flatMap(_.split(" "))
val wordCounts = words.map(x => (x, 1)).reduceByKey(_ + _)
wordCounts.print()
ssc.start()
```

Spark Streaming - Comments

Mini-batch approach appears to be "low latency" enough for many applications.

Integration with the rest of the BDAS/Spark stack is a big deal for users

We're also adding a "timeseries" capability to BDAS (see AMPCamp 6 `ampcamp.berkeley.edu`)

- initially batch but streaming integration planned

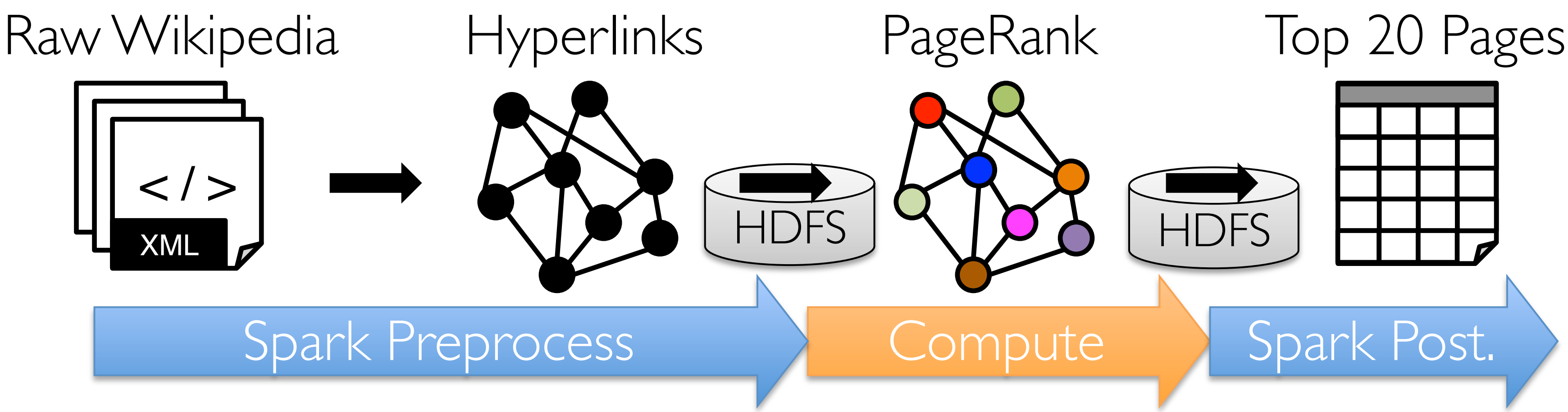
Trend III

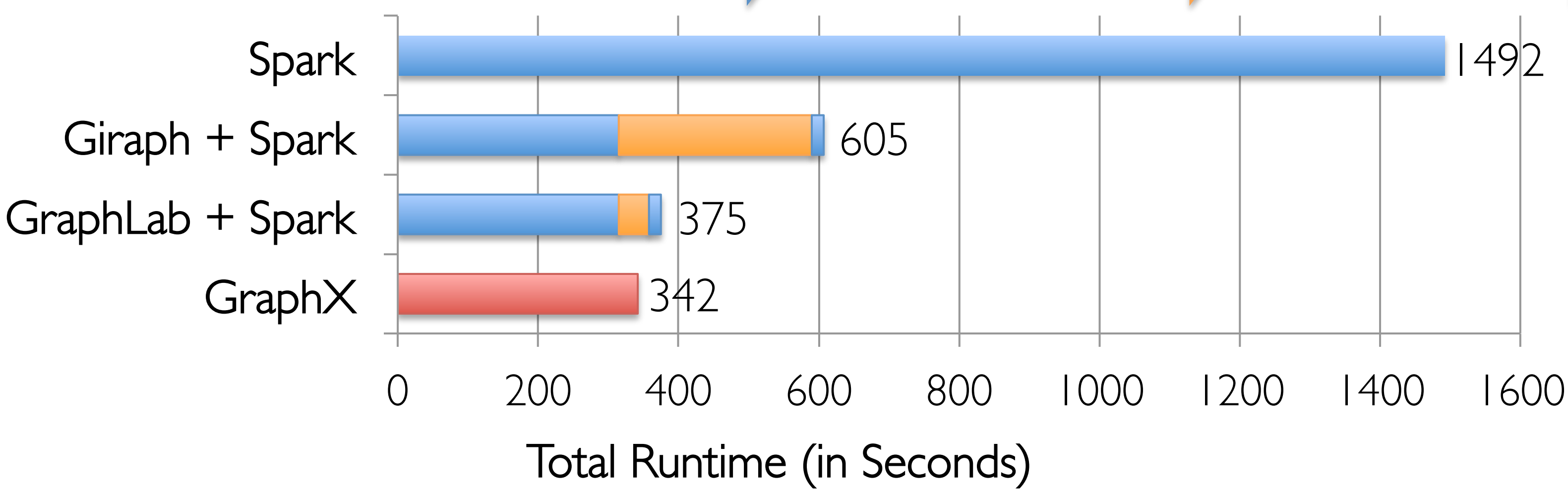
MACHINE LEARNING PIPELINES

Beyond ML Operators

- Data Analytics is a complex process
- Rare to simply run a single algorithm on an existing data set
- Emerging systems support more complex workflows:
  - Spark MLPipelines
  - Google TensorFlow
  - KeystoneML (BDAS)

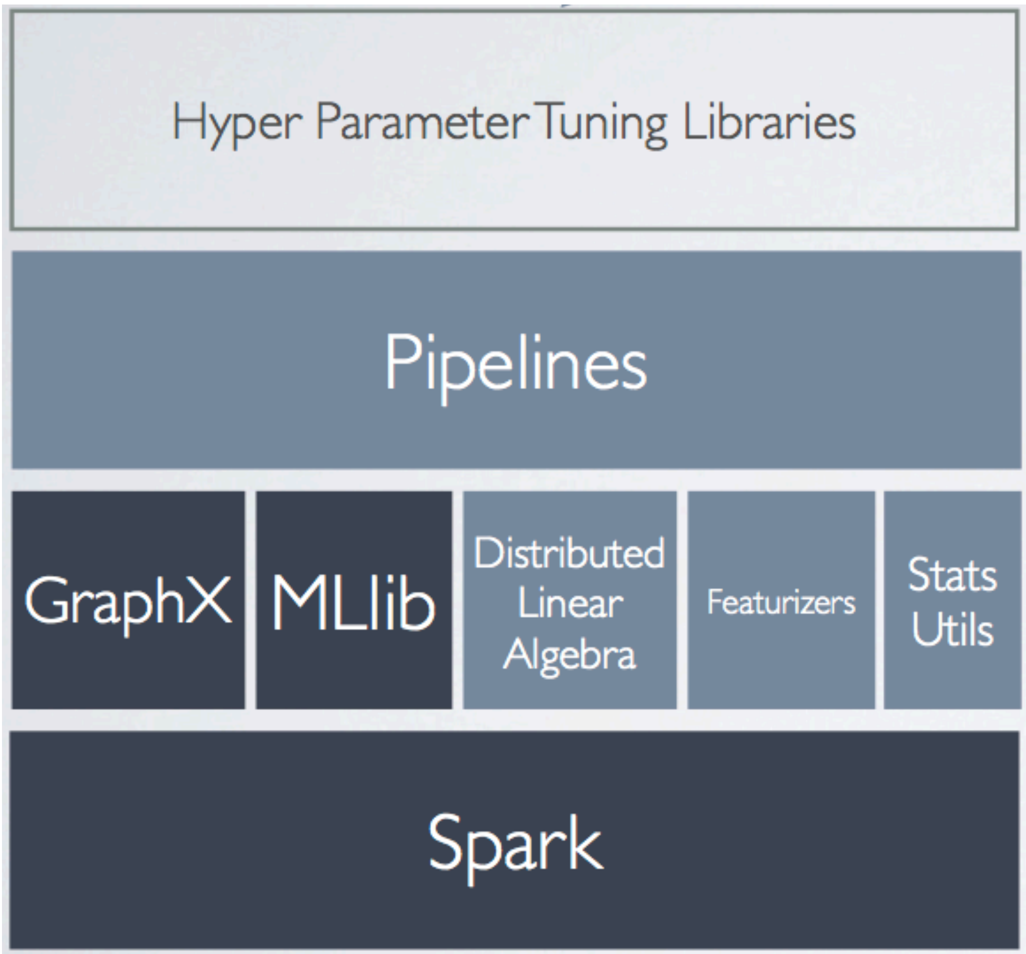
A Small Pipeline in GraphX





Need to measure End-to-End Performance

MLBase: Distributed ML Made Easy



DB Query Language Analogy:

Specify **What** not **How**

MLBase chooses:

- Algorithms/Operators
- Ordering and Physical Placement
- Parameter and Hyperparameter Settings
- Featurization

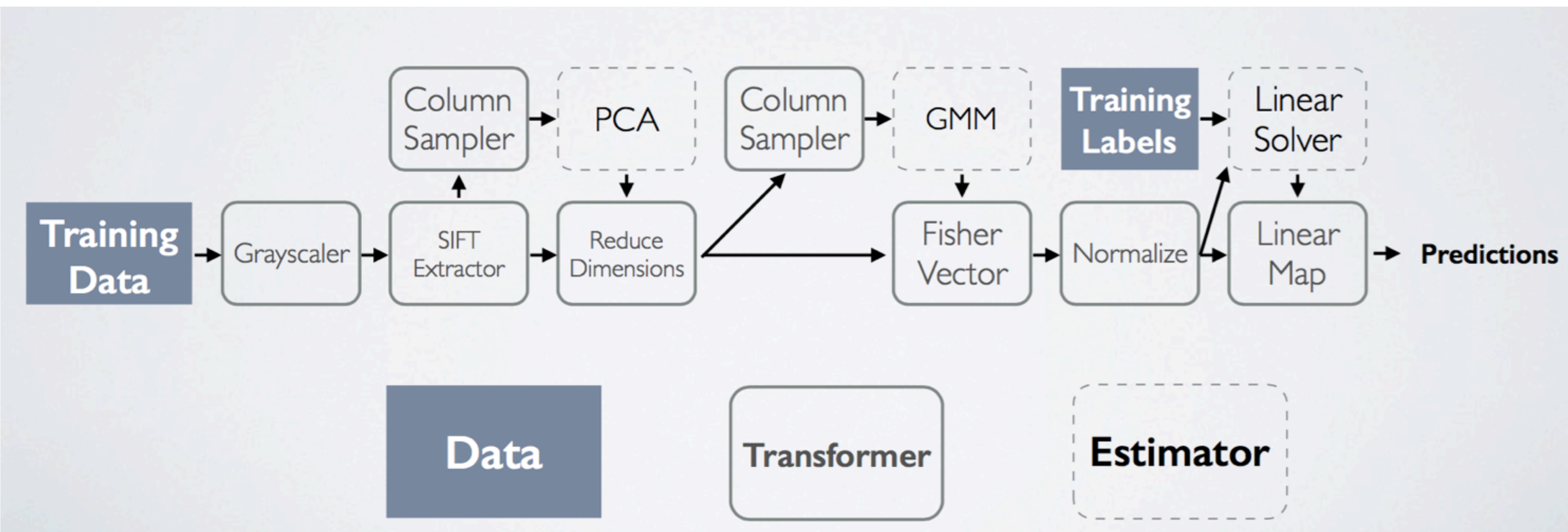
Leverages Spark for Speed and Scale

T. Kraska, A. Talwalkar, J. Duchi, R. Griffith, M. Franklin, M. Jordan, "MLBase: A Distributed Machine Learning System", CIDR 2013.

KeystoneML

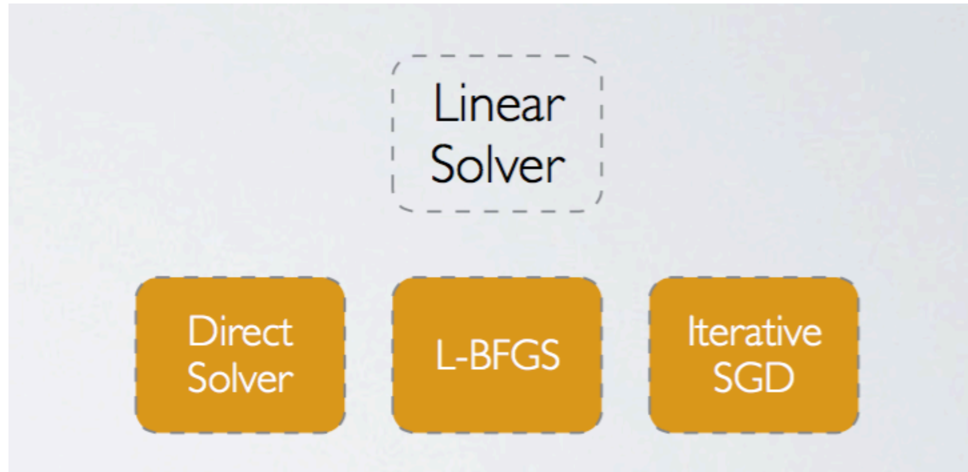
Software framework for describing complex ***machine learning pipelines*** built on Apache Spark.

Pipelines are specified using domain specific and general purpose ***logical operators.***

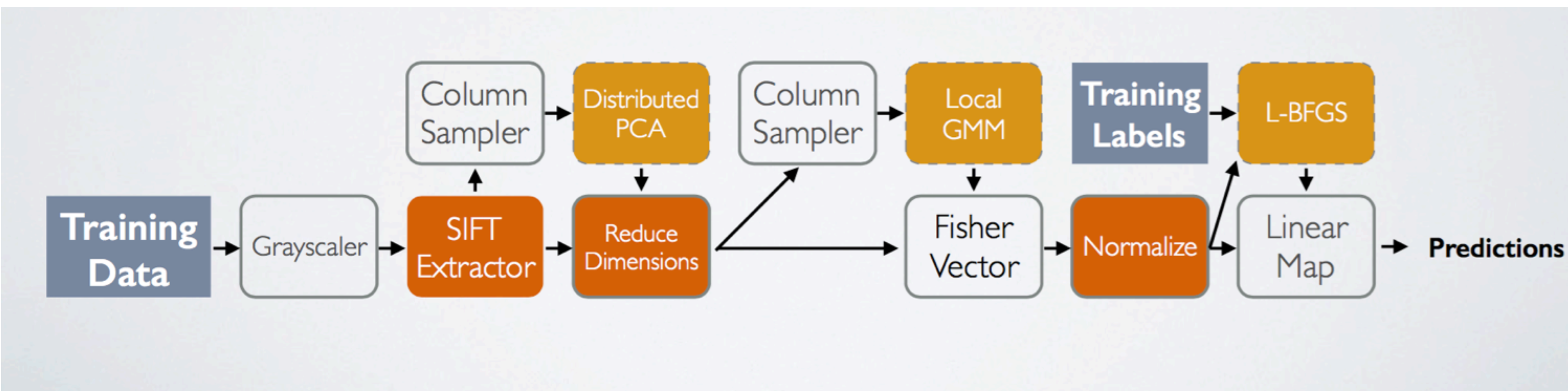


High-level API ➔ Optimizations

Automated ML operator selection



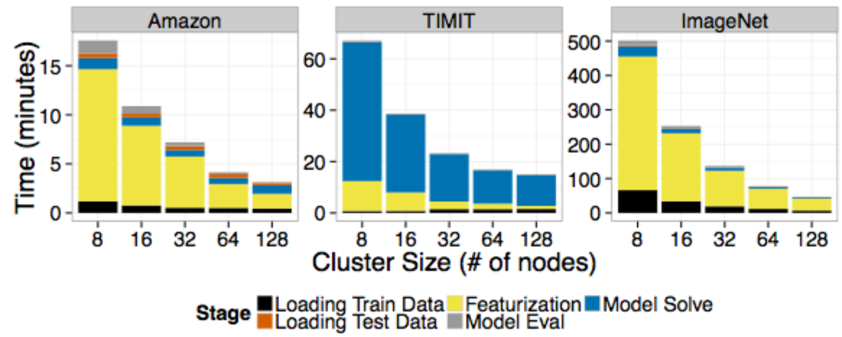
Auto-caching for iterative workloads



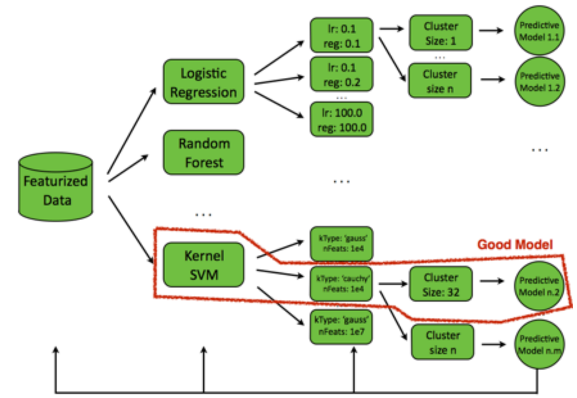
KeystoneML: Latest News

v0.3 to be released this week.

Scale-out performance on 10s of TBs of training features on 100s of machines. apps: Image Classification, Speech, Text.



First versions of node-level and whole-pipeline optimizations.

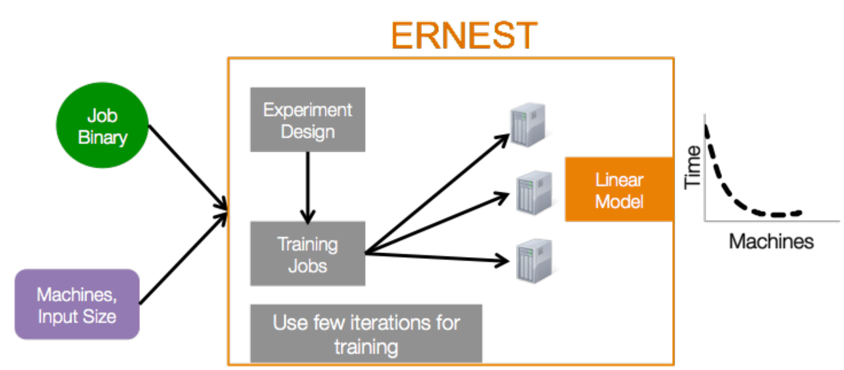


Many new high-speed, scalable operators

Coming soon:

» Principled, scalable hyperparameter tuning. (TuPAQ - SoCC 2015)

» Advanced cluster sizing/job placement algorithms. (Ernest - NSDI 2016)



Trend IV

MODEL AND DATA SERVING

Introducing Velox: Model Serving

Data $\xrightarrow{\textit{Training}}$ Model

Where do models go?

- Conference Papers
- Sales Reports
- Drive Actions

Driving Actions

Suggesting Items at Checkout

Fraud Detection

Cognitive Assistance

Internet of Things

Low-Latency

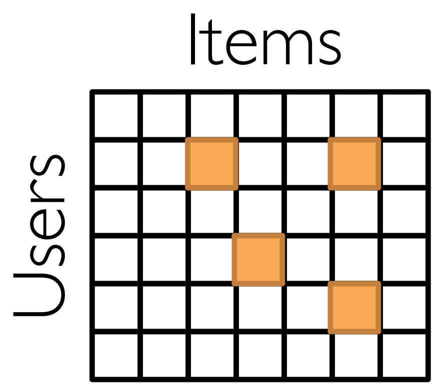
Personalized

Rapidly Changing

Current Solutions & Limitations

Materialize Everything: Pre-compute all Predictions

Train model
on old data



Low Latency
Data Serving

Specialized Service: Build a Prediction Service

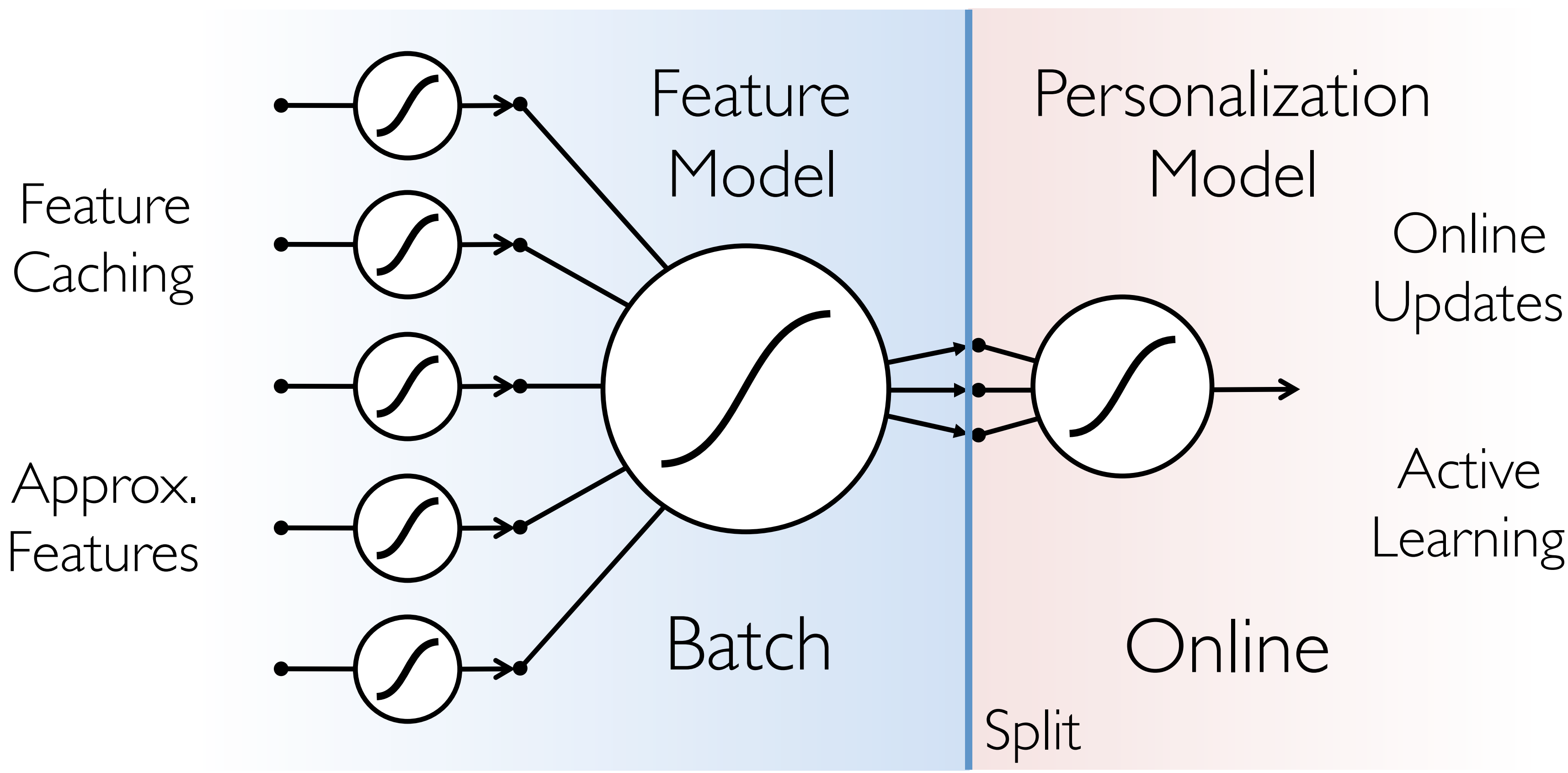
Train model
on old data

High-Latency
One-off
Prediction Service

Velox Model Serving System

[CIDR'15]

Decompose *personalized* predictive models:



Order-of-magnitude reductions in prediction latencies.

Hybrid Learning

Update feature functions *offline* using batch solvers

- Leverage high-throughput systems (Apache Spark)
- Exploit slow change in population statistics

$$f(x;\theta)^T w_u$$

Update the user weights *online*:

- Simple to train + more robust model
- Address rapidly changing user statistics

Serving Data

- Intelligent services also require serving data (in addition to predictions).
- KV Stores such as Cassandra, HBase, etc. provide this functionality.
- Traditional problems of merging analytics and serving (or OLTP and OLAP) remain.

Trend V

BIG DATA FOR IOT, HIGH PERFORMANCE COMPUTING AND MORE…

High-Performance Computing

HPC used to have a monopoly on "big iron"

Completely different scale/pace of innovation

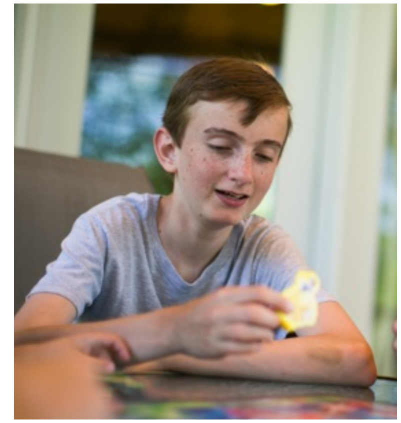
White House "National Strategic Computing Initiative" Includes combining HPC and Big Data

Scientific Computing Meets Big Data Technology: An Astronomy Use Case

Zhao Zhang[*,°] Kyle Barbary[°,†] Frank Austin Nothaft[*,‡] Evan Sparks[*]
Oliver Zahn[†] Michael J. Franklin[*,°] David A. Patterson[*,‡] Saul Perlmutter[°,†]
[*] AMPLab, University of California, Berkeley
[°] Berkeley Institute for Data Science, University of California, Berkeley
[†] Berkeley Center for Cosmological Physics, University of California, Berkeley
[‡] ASPIRE Lab, University of California, Berkeley

IEEE Conf. on Big Data 2015

AMPLab Genomics



- SNAP (Scalable Nucleotide Alignment): alignment in hours vs. days
- Why Speed Matters – A real-world use case

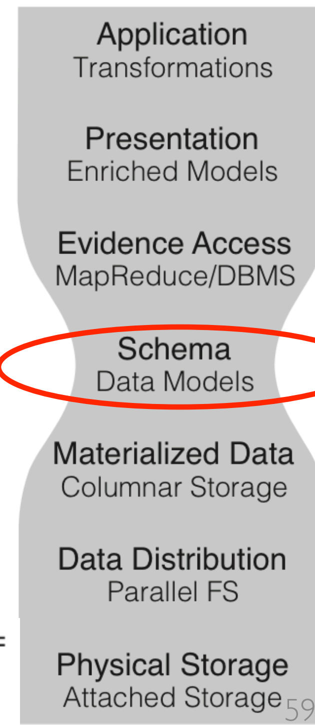
M. Wilson, …, and C. Chiu, "Actionable Diagnosis of Neruoleptospirosis by Next-Generation Sequencing", June 4, 2014, *New England Journal of Medicine,*

Table 1: Summary Performance on NA12878

Tool	EC2	BQSR	IR	DM	Sort	FS	Total
[14]	1†	**1283m**	**658m**	—	—	—	
[32]	1†	—	—	509m	203m	54m41	2075m1 $214.39
[50]	1†	—	—	**44m50**	**83m**	**6m11**	
[51]	1†	—	—	160m	562m	—	
ADAM	1†	1602m 1/1.25×	366m 1.7×	143m 1/3.8×	108m 1/1.3×	2m17 2.7×	2221m17 1/1.07×
ADAM	32⋆	74m 17×	64m 10×	34m56 1.2×	39m23 2.1×	0m43 8.6×	223m2 9.3×
ADAM	64⋆	41m52 30×	35m39 18×	21m35 2.0×	18m56 4.3×	0m49 7.5×	118m51 17×
ADAM	128⋆	25m59 49×	20m27 32×	15m27 2.9×	10m31 7.9×	1m20 4.3×	73m44 28× $78.92

	Machine	Cost	Description
†	i2.8xlarge	$6.20	32 proc, 244G RAM, 8 SDD
⋆	r3.2xlarge	$0.70	8 proc, 61G RAM, 1 SDD

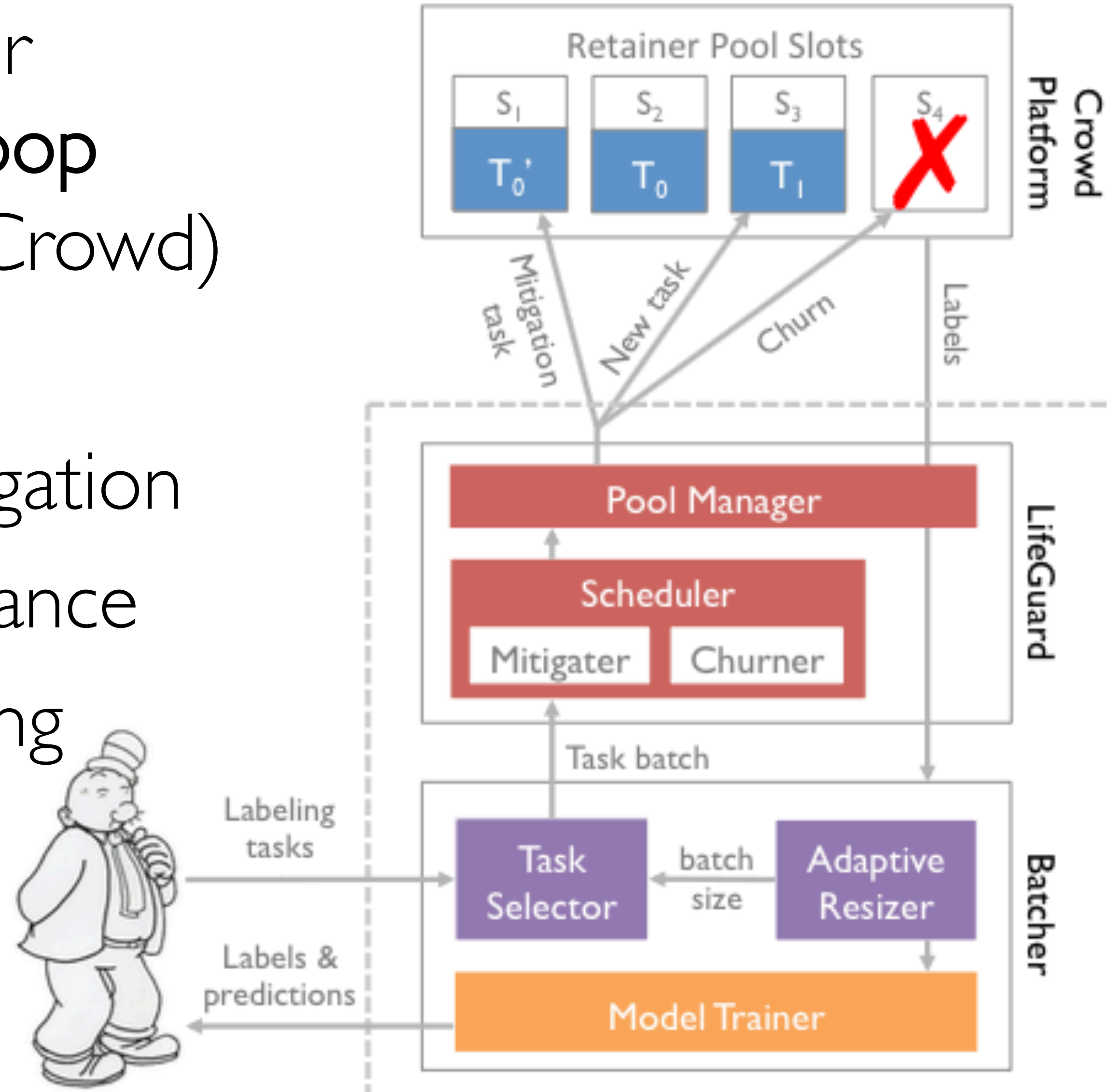
F. Nothaft, et. al., "Rethinking Data-Intensive Science Using Scalable Analytics Systems", *ACM SIGMOD Conf.,* June 2015.



Integrating the “P” in AMP

Optimization for **human-in-the-loop** analtyics (AMPCrowd)

- SampleClean
- Straggler Mitigation
- Pool Maintenance
- Active Learning



BDS Meets Internet of Things

Streaming and Real Time

What to keep, what to drop

Edge Processing

Privacy

Partitions, Fault Tolerance, Eventual Consistency, Order-dependence

Big Data Software Moves Fast

We looked at the following trends:

1) Integrated Stacks vs Silos
2) "Real-Time" Redux
3) Machine Learning and Advanced Analytics
4) Serving Data and Models
5) Big Data Software + X <HPC, People, IoT,…>

To find out more or get involved:

amplab.berkeley.edu

franklin@berkeley.edu

Thanks to NSF CISE Expeditions in Computing, DARPA XData,
Founding Sponsors: Amazon Web Services, Google, IBM, and SAP,
the Thomas and Stacy Siebel Foundation,
all our industrial sponsors and partners, and all the members of the AMPLab Team.