Big Data Software: What's Next?
(*and what do we have to say about it?*)

Michael Franklin

43rd VLDB Conference

Munich

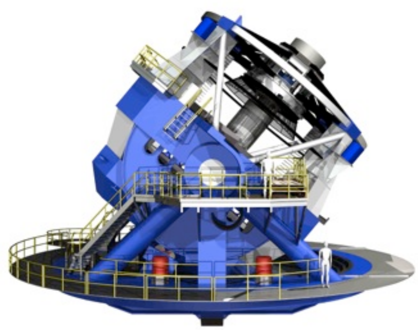
August 2017

THE UNIVERSITY OF CHICAGO

The VLDB Keynote "Sandwich"

The Data Center under your Desk - How Disruptive is Modern Hardware for DB System Design?
Wolfgang Lehner (Technische Universitä̈t Dresden)
Tuesday 29 August, 8:30-10:00
While we are already used to see more than 1,000 cores within a single machine, the next processing platforms for database engines will be widely heterogeneous with built-in GPU-style processors as well as specialized FP- GAs and chips with domain-specific instruction sets tak- ing advantage of the "Dark Silicon" effect. Moreover, the traditional volatile as well as the upcoming non-volatile RAM with capacities in the 100s of TBytes per machine will provide great opportunities for storage engines but also call for radical changes on

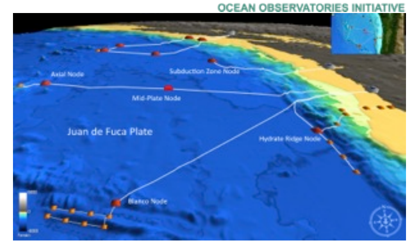
Big Data = Nearly every field of endeavor is transitioning from “data poor” to “data rich”



Astronomy: LSST

Physics: LHC

OCEAN OBSERVATORIES INITIATIVE



Oceanography

Neuroscience: EEG, fMRI

Sociology: The Web

Biology: Sequencing

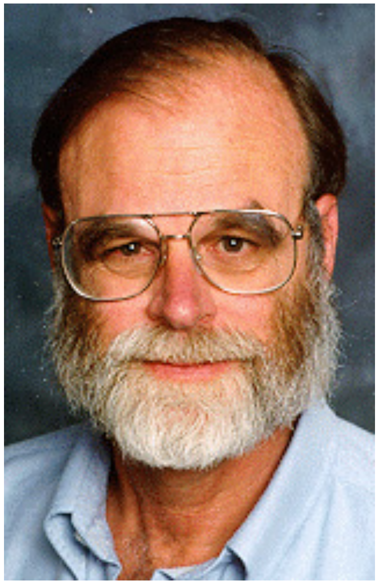
Economics: mobile, POS terminals

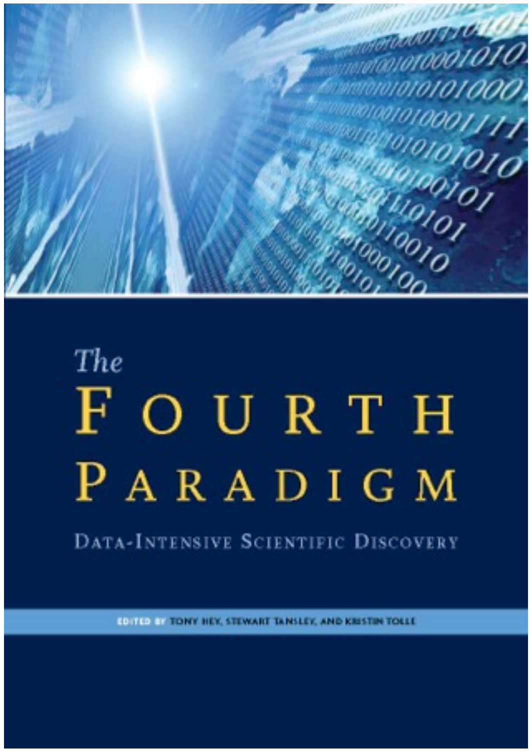
Data-Driven Medicine

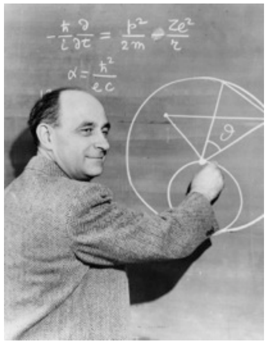
Sports

The Fourth Paradigm of Science

1. Empirical + experimental
2. Theoretical
3. Computational
4. Data-Intensive







Open Source Ecosystem & Context

2006-2010
Autonomic Computing & Cloud

2011-2016
Big Data Analytics

Usenix HotCloud Workshop 2010

Spark: Cluster Computing with Working Sets

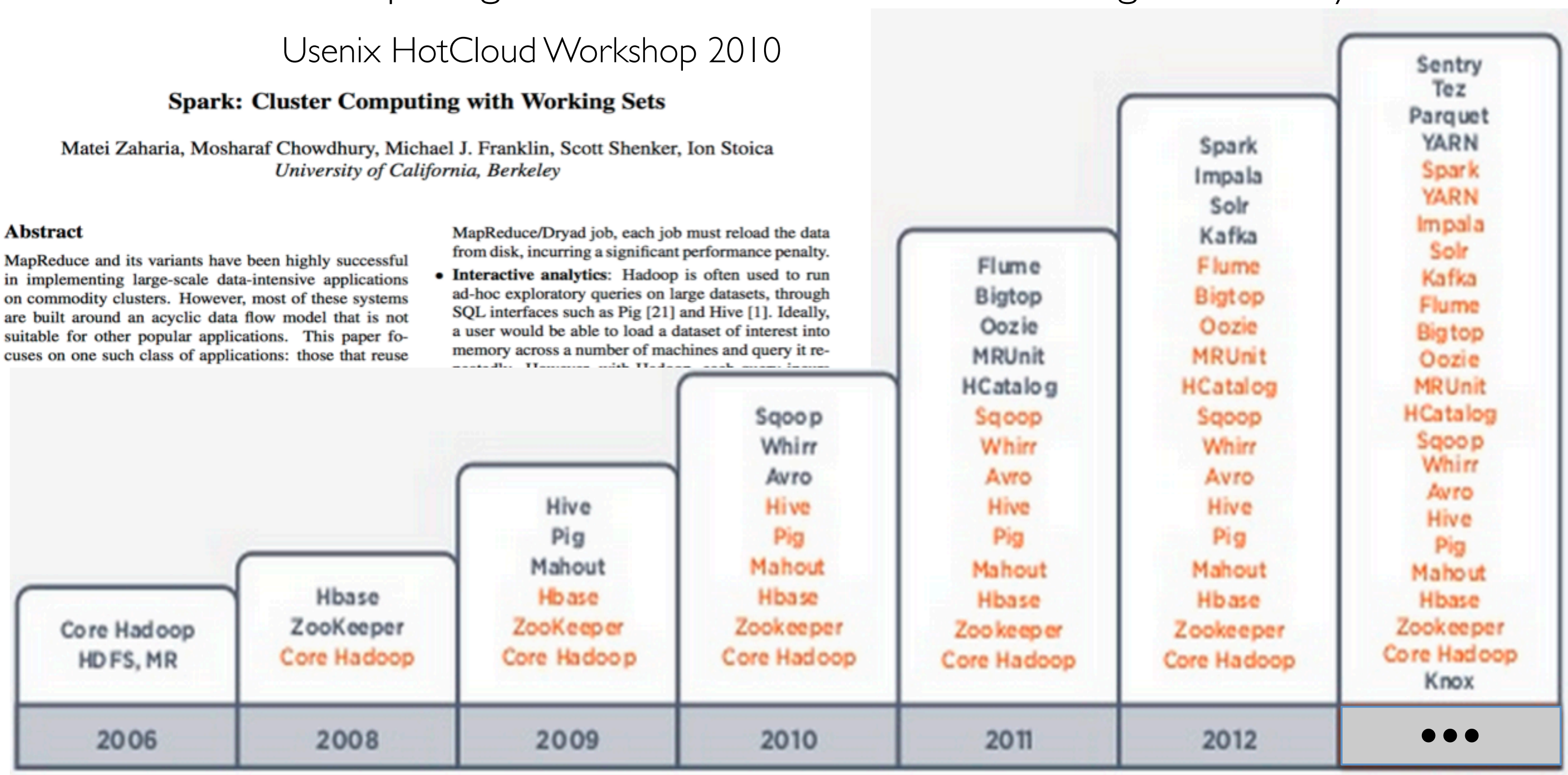
Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley

Abstract

MapReduce and its variants have been highly successful in implementing large-scale data-intensive applications on commodity clusters. However, most of these systems are built around an acyclic data flow model that is not suitable for other popular applications. This paper focuses on one such class of applications: those that reuse

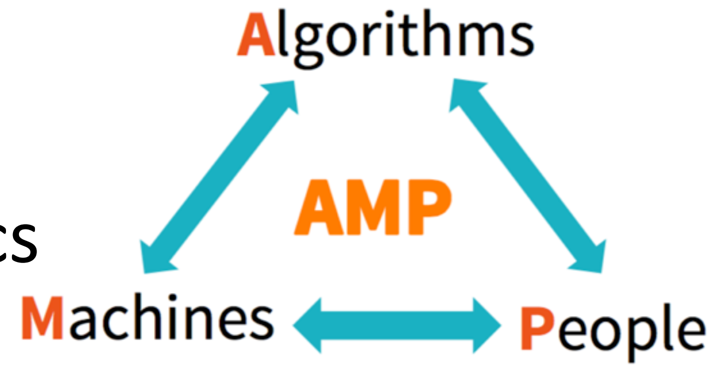
MapReduce/Dryad job, each job must reload the data from disk, incurring a significant performance penalty.

- **Interactive analytics**: Hadoop is often used to run ad-hoc exploratory queries on large datasets, through SQL interfaces such as Pig [21] and Hive [1]. Ideally, a user would be able to load a dataset of interest into memory across a number of machines and query it re-



amplab UC BERKELEY

"Making Sense at Scale"



6 years (2011-2016)

~12 faculty; ~120 PhD & Postdocs

DB+Systems+ML

NSF Expeditions, DARPA, DOE, DHS, 40+ Companies

Pubs in SIGMOD/VLDB/ICDE, OSDI/NSDI/SOSP/SOCC/SIGCOMM, NIPS/ICML/ICDM, HCOMP...

Some Stats:

- 3 ACM Dissertation Awards (1 + 2 HMs)
- 2 CACM Research Highlights
- 4 Spinout companies: ~$400M in venture funding
- 3 Marriages (and numerous long term relationships)

Berkeley Data Analytics Stack

In House Applications – Genomics, IoT, Energy, Cosmology

GraphX BlinkDB Sample Clean ML base

Access and Interfaces

Spark Streaming SHARK Spark SQL Spark + R KeystoneML Spark MLlib

Spark Processing Engines Clipper

ALLUXIO Storage Succinct Enabling Queries on Compressed Data

MESOS Resource Virtualization

A CONFLUENCE OF ML, SYSTEMS AND DATABASE THINKING

DB Thinking Meets Systems Thinking?

MapReduce: A major step backwards

By David DeWitt on January 17, 2008

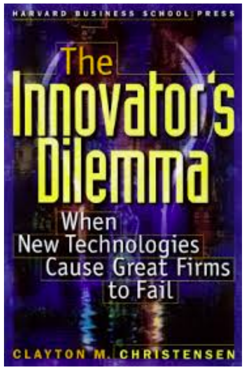
[Note: Although the system attributes this post to a single author, it was written by David J. DeWitt and Michael Stonebraker]

DB Thinking Meets Systems Thinking?

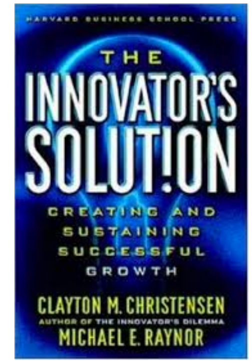
"MapReduce may be a good idea for writing certain types of general-purpose computations, but to the database community, it is:

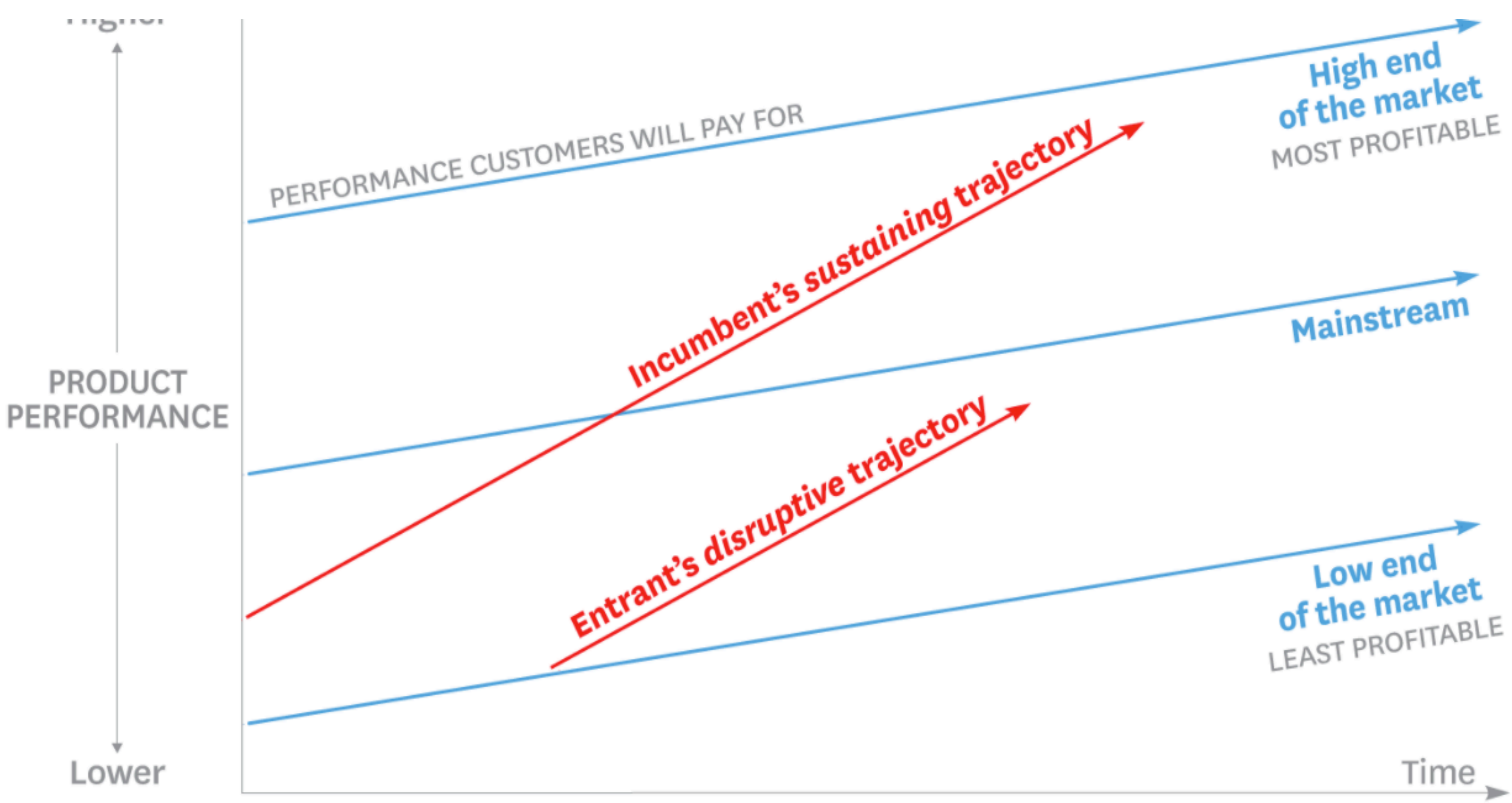
1. **A giant step backward** in the programming paradigm for large-scale data intensive applications
2. **A sub-optimal implementation**, in that it uses brute force instead of indexing
3. **Not novel at all** - it represents a specific implementation of well known techniques developed **nearly 25 years ago**
4. **Missing most of the features** that are routinely included in current DBMS
5. **Incompatible with all of the tools** DBMS users have come to depend on"

AT THE TIME, MANY IN THE DB CAMP AGREED



Disruptive Technology (low end/new market)





SOURCE CLAYTON M. CHRISTENSEN, MICHAEL RAYNOR, AND RORY MCDONALD

DB Thinking Meets Systems Thinking?

"MapReduce may be a good idea for writing certain types of general-purpose computations, but to the database community, it is:

1. **A giant step backward** in the programming paradigm for large-scale data intensive applications
2. **A sub-optimal implementation**, in that it uses brute force instead of indexing
3. **Not novel at all** - it represents a specific implementation of well known techniques developed **nearly 25 years ago**
4. **Missing most of the features** that are routinely included in current DBMS
5. **Incompatible with all of the tools** DBMS users have come to depend on"

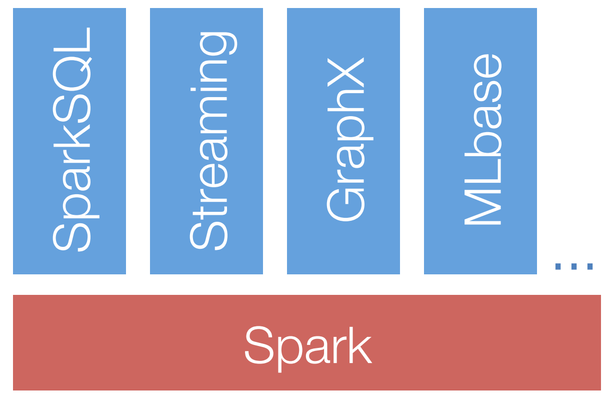
BUT “DATABASE THINKING” IS DRIVING THE IMPROVEMENT PROCESS

Spark's Philosphy

- Specializing MapReduce leads to stovepiped systems
- Instead, **generalize** MapReduce:

   1. Richer Programming Model
      ➔Fewer Systems to Master

   2. Memory Management
      ➔Less data movement leads to better performance for complex analytics



Abstraction: *Dataflow Operators*

- `map`
- `filter`
- `groupBy`
- `sort`
- `union`
- `join`
- `leftOuterJoin`
- `rightOuterJoin`

- `reduce`
- `count`
- `fold`
- `reduceByKey`
- `groupByKey`
- `cogroup`
- `cross`
- `zip`

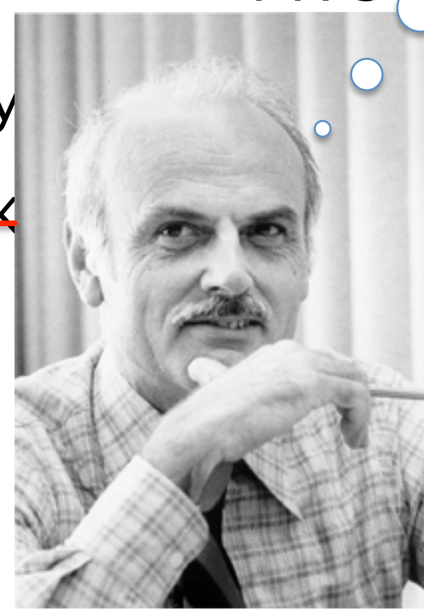
`sample`
`take`
`first`
`partitionBy`
`mapWith`
`pipe`
`save`
`...`

Abstraction: *Dataflow Operators*

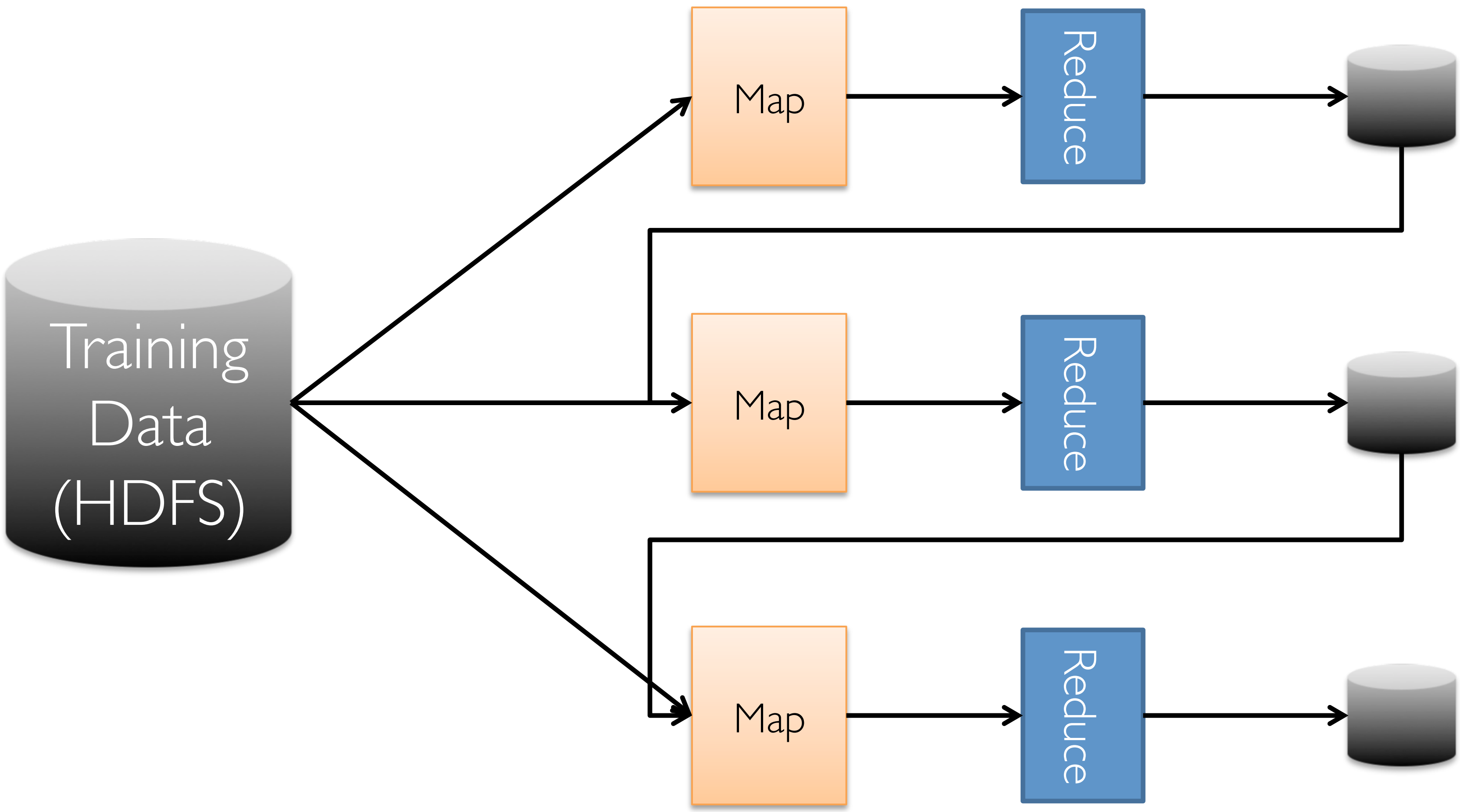
- map
- filter
- groupBy
- sort
- union
- join
- leftOuterJoin
- rightOuterJoin

- reduce
- count
- fold
- reduceBy
- groupByK
- cogroup
- cross
- zip

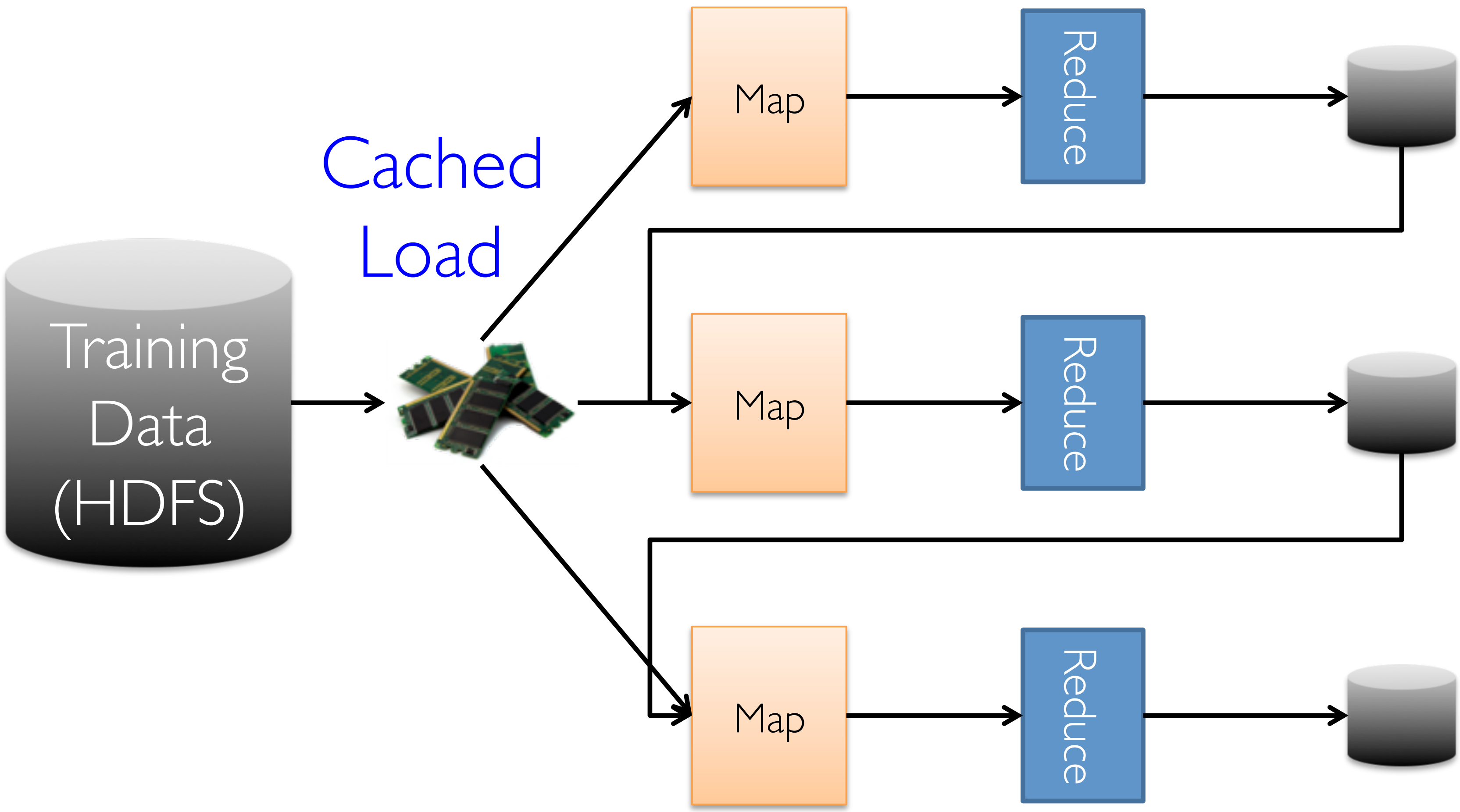
sample
take
first
tionBy
th



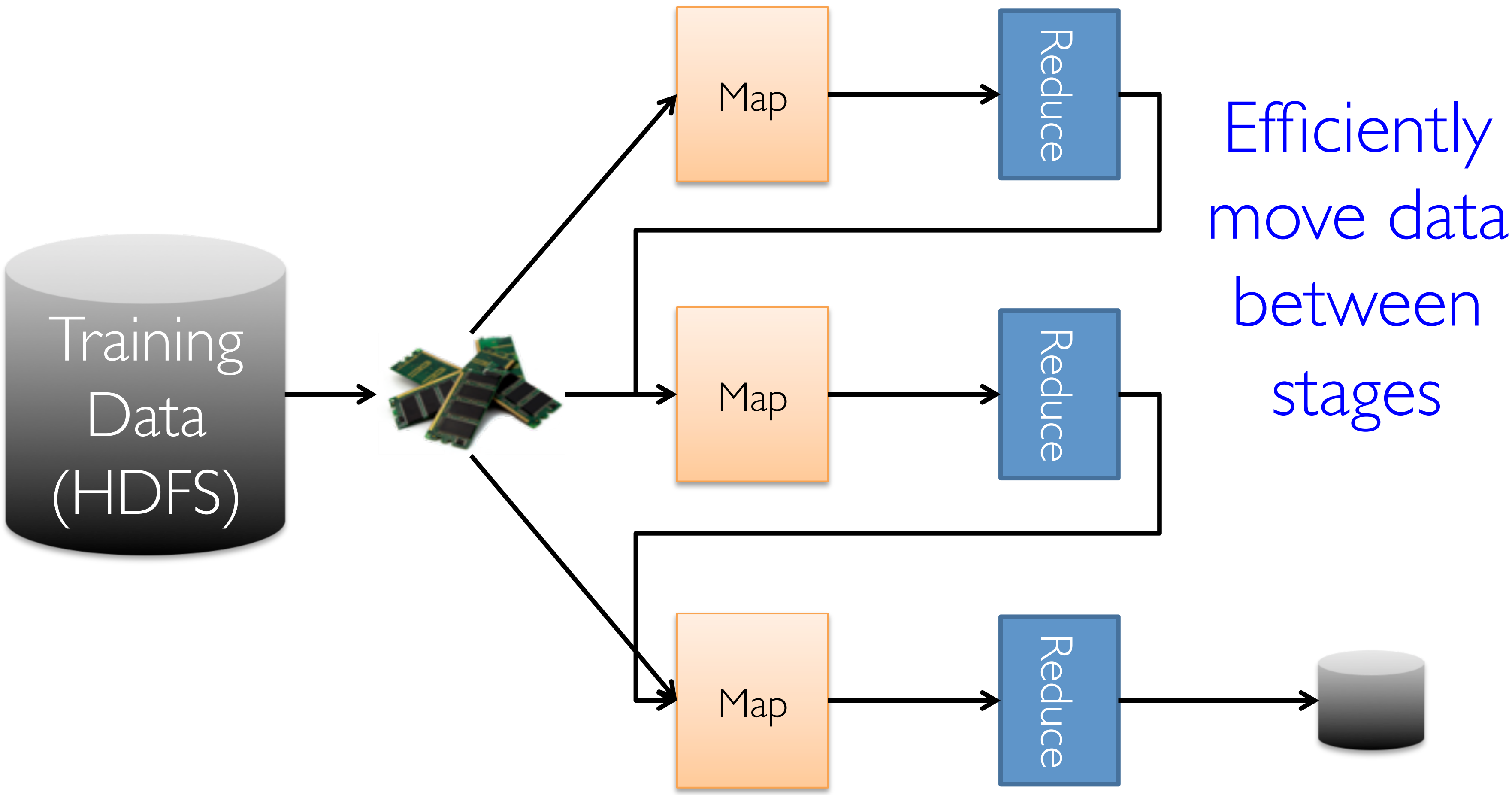
Memory Mgmt in Hadoop MR



Memory Mgmt in Spark

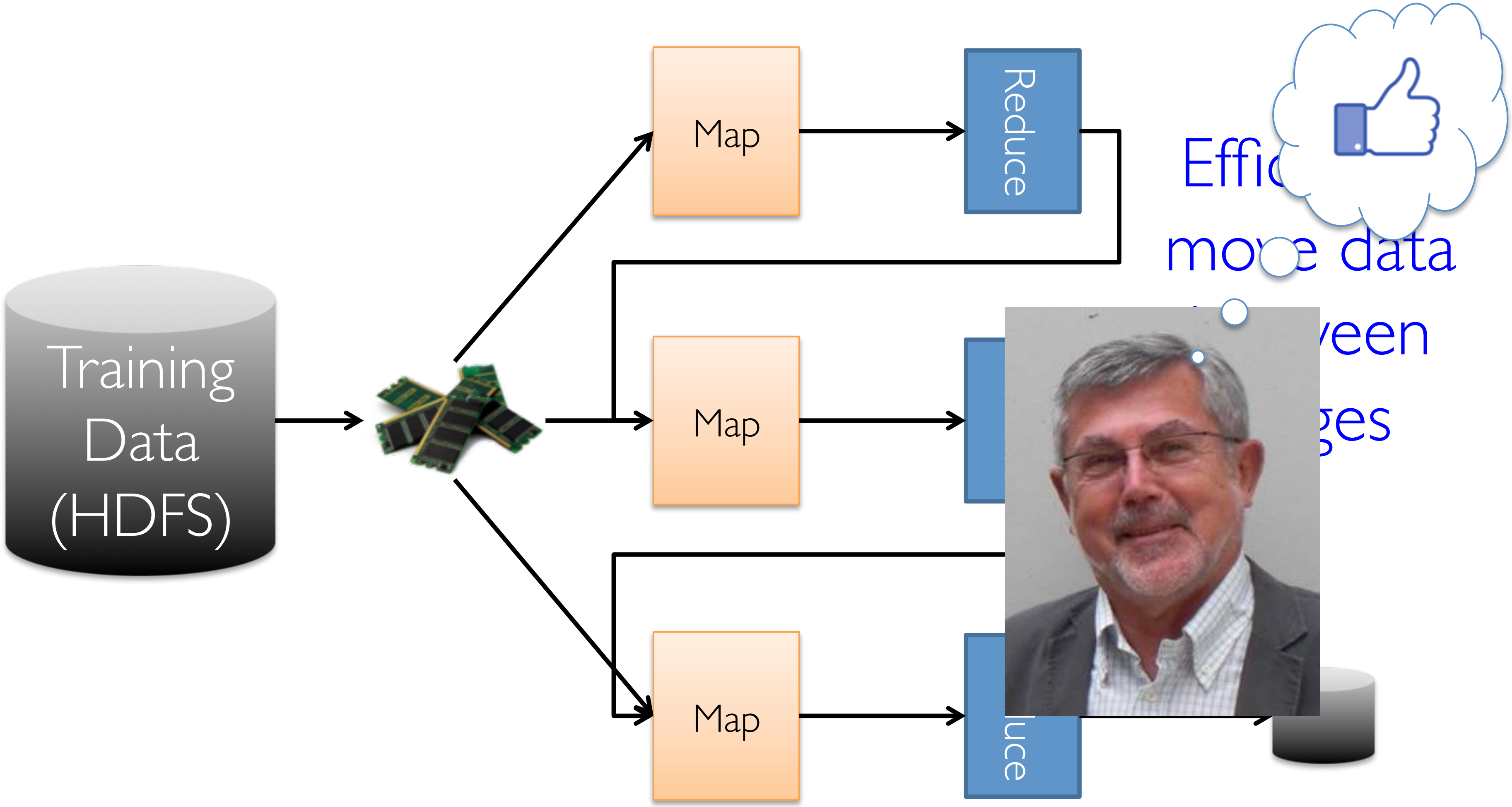


Memory Management in Spark



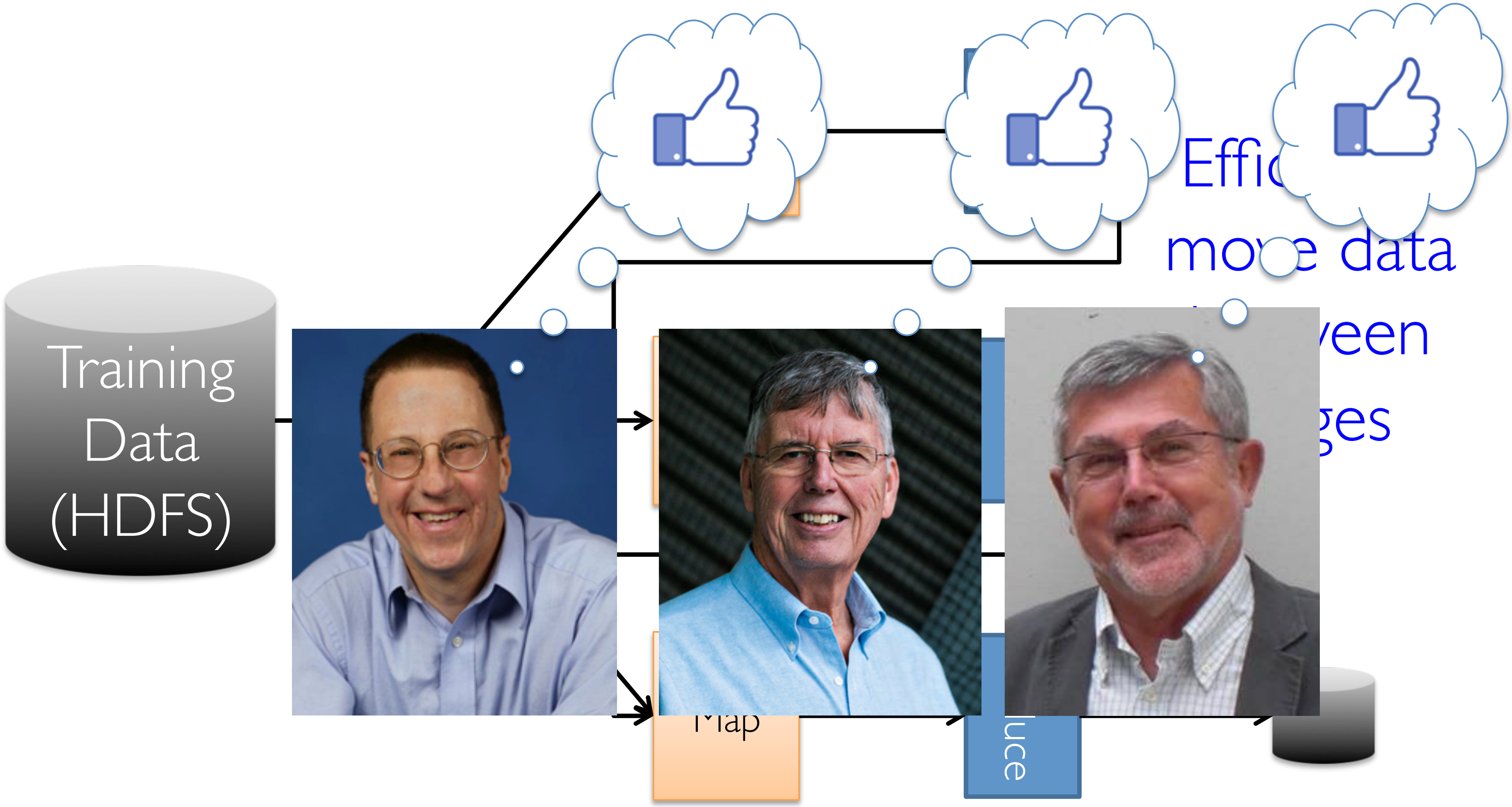
10-100× speed up vs.Hadoop MapReduce
with no HDFS data migration needed

Memory Management in Spark



10-100× speed up vs.Hadoop MapReduce
with no HDFS data migration needed

Memory Management in Spark

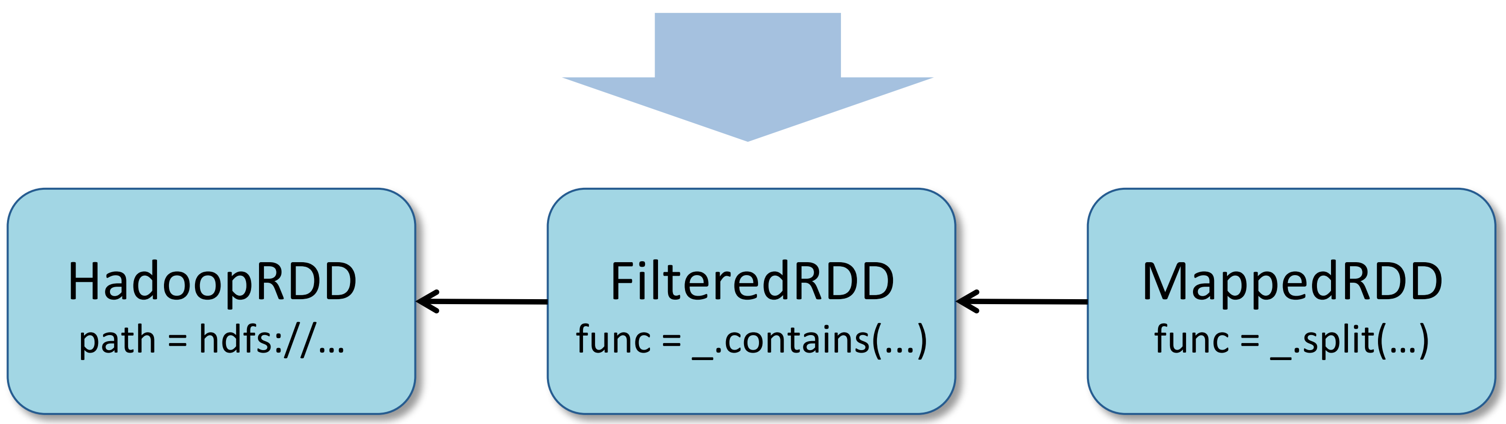


10-100× speed up vs.Hadoop MapReduce
with no HDFS data migration needed

Lineage (aka Logical Logging)

- **RDDs:** **Immutable** collections of objects that can be stored in memory or disk across a cluster
  - Built via parallel transformations (map, filter, …)
  - Automatically rebuilt on (partial) failure

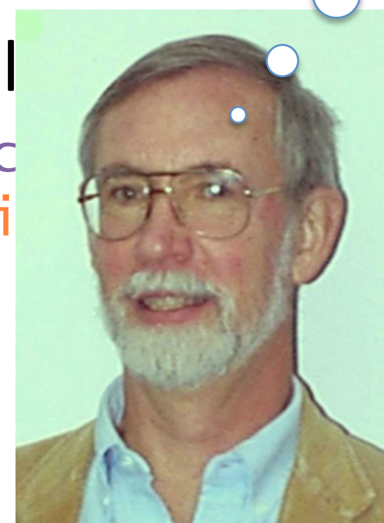
```
messages = textFile(...).filter(_.contains(“error”))
                        .map(_.split(‘\t’)(2))
```



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

Lineage (aka Logical Logging)

- **RDDs:** **Immutable** collectio  bjects  n
  be stored in memory or dis … a clu
  - Built via parallel transformatio … (map, filter, …)
  - Automatically r … ial



```
messages = textFile( … .c … ror”))
                       … pli
```

HadoopRDD
path = hdfs://…

FilteredRDD
func = _.contains(...)

MappedRDD
func = _.split(…)

HadoopRDD ← FilteredRDD ← MappedRDD

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

Spark Native SQL Support

Spark 2.2.0 Overview Programming Guides API Docs Deploying More

Spark SQL, DataFrames and Datasets Guide

- Overview
  - SQL
  - Datasets and DataFrames
- Getting Started
  - Starting Point: SparkSession
  - Creating DataFrames
  - Untyped Dataset Operations (aka DataFrame Operations)
  - Running SQL Queries Programmatically
  - Global Temporary View
  - Creating Datasets
  - Interoperating with RDDs
    - Inferring the Schema Using Reflection
    - Programmatically Specifying the Schema
  - Aggregations
    - Untyped User-Defined Aggregate Functions
    - Type-Safe User-Defined Aggregate Functions
- Data Sources
  - Generic Load/Save Functions
    - Manually Specifying Options
    - Run SQL on files directly
    - Save Modes
    - Saving to Persistent Tables
    - Bucketing, Sorting and Partitioning
  - Parquet Files
    - Loading Data Programmatically
    - Partition Discovery

VLDB 2017 Munich · Germany

DataFrames
(main abstraction in Spark 2.0)

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

Notes:

1) Some people prefer this to SQL ☺
2) Dataframes can be typed (called "Datasets")

Catalyst Optimizer

- Typical DB optimizations across SQL and Dataframes
  - Extensibility via Optimization Rules written in Scala
  - Open Source optimizer evolution!
- Code generation for inner-loops, iterator removal
- Extensible Data Sources: CSV, Avro, Parquet, JDBC, …

  via TableScan (all cols), PrunedScan (project),
  FilteredPrunedScan(push advisory selects and projects)
  CatalystScan (push advisory full Catalyst expression trees)
- Extensible (User Defined) Types
- Cost-based (as of v2.2)

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

Catalyst Optimizer

- Typical DB optimizations across SQL and Dataframes
  – Extensibility via Optimization Rules wr Scala
  – Open Source optimizer evolution!

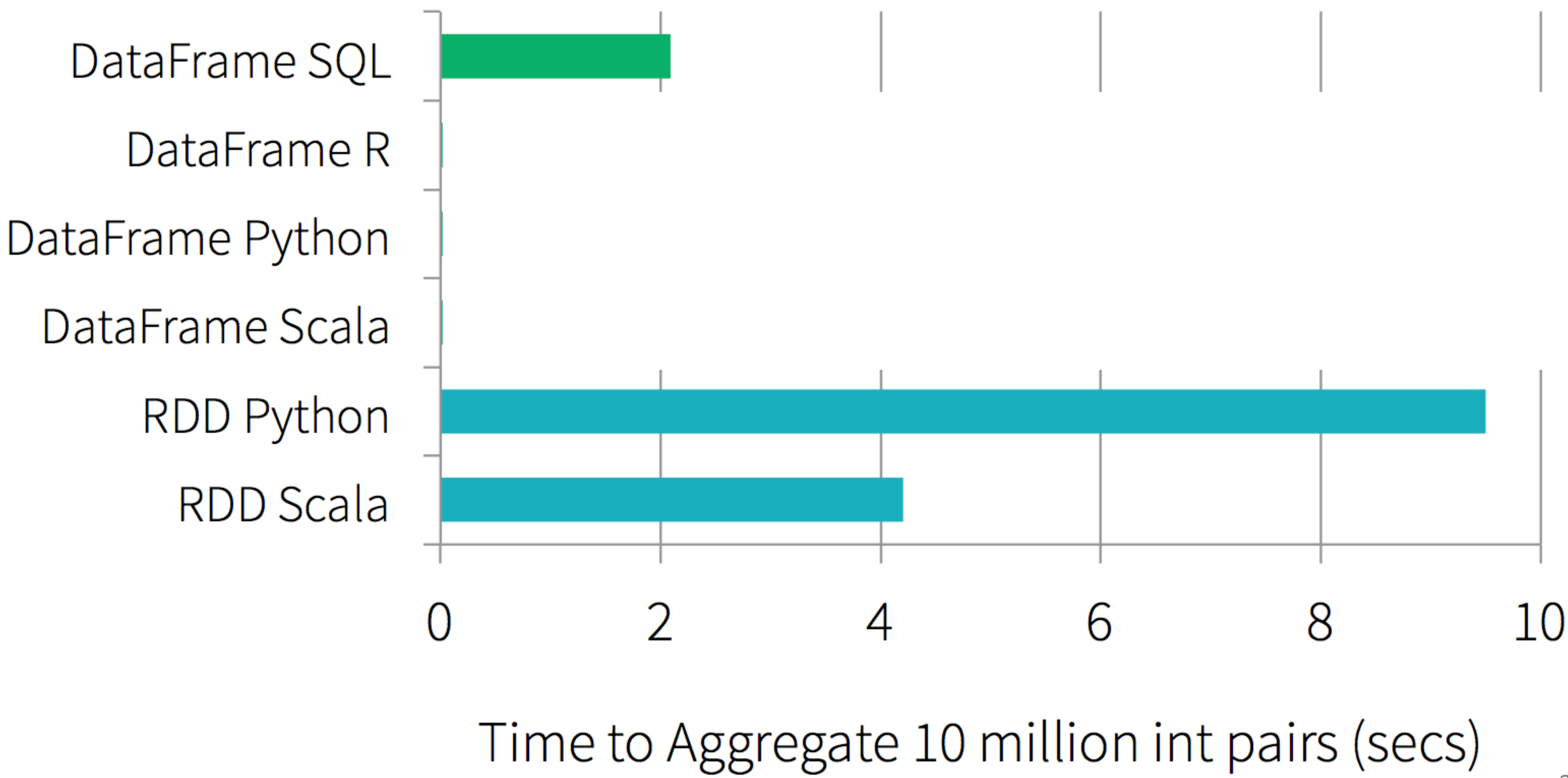
- Code generation for inner-loops, ite emoval
- Extensible Data Sources: CSV, Avro, Parquet, JDBC, …

  via TableScan (all cols), Prun c
  FilteredPrunedScan(push ad a
  CatalystScan (push advisory x s)

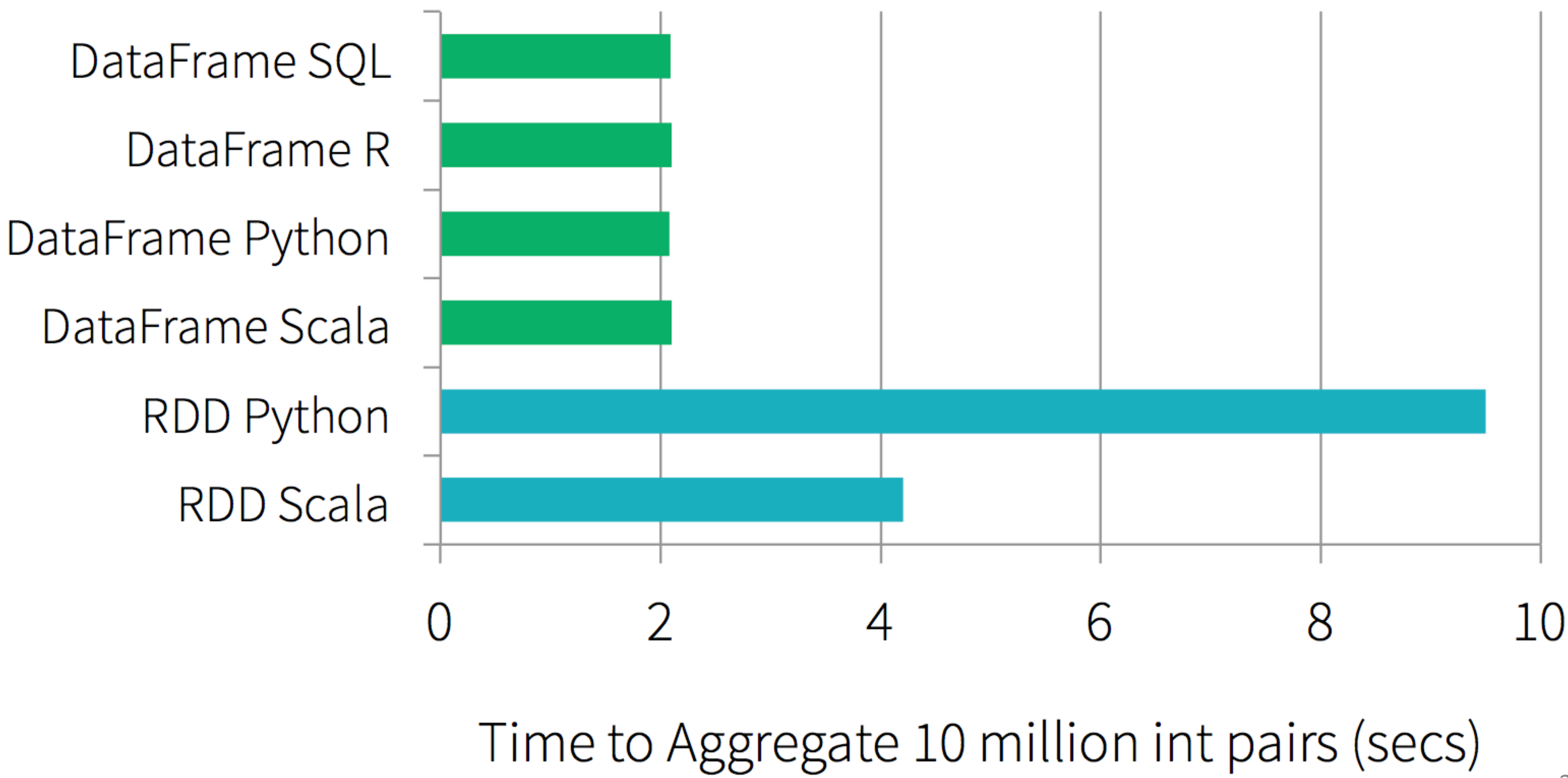
- Extensible (User Definec
- Cost-based (as of v2.2)

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

An interesting thing about SparkSQL Performance



An interesting thing about SparkSQL Performance



Spark Structured Streams (unified)

Batch Analytics

```
// Read data once from an S3 location
val inputDF = spark.read.json("s3://logs")

// Do operations using the standard DataFrame API and write to MySQL
inputDF.groupBy($"action", window($"time", "1 hour")).count()
       .write.format("jdbc")
       .save("jdbc:mysql//...")
```

Streaming Analytics

```
// Read data continuously from an S3 location
val inputDF = spark.readStream.json("s3://logs")

// Do operations using the standard DataFrame API and write to MySQL
inputDF.groupBy($"action", window($"time", "1 hour")).count()
       .writeStream.format("jdbc")
       .start("jdbc:mysql//...")
```

Spark Structured Streams (unified)

Batch Analytics

```
// Read data once from an S3 location
val inputDF = spark.read.json("s3://logs")

// Do operations using the standard DataFram                    to MySQL
inputDF.groupBy($"action", window($"time", "                    )
       .write.format("jdbc")
       .save("jdbc:mysql//...")
```

Streaming Analytics

```
// Read data continuously from an S3 location
val inputDF = spark.readStream.json("s3://logs")

// Do operations using the standard DataFrame API and write to MySQL
inputDF.groupBy($"action", window($"time", "1 hour")).count()
       .writeStream.format("jdbc")
       .start("jdbc:mysql//...")
```

Putting it all Together: Multi-modal Analytics

SQL

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

Machine Learning

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

Streaming

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

Current release has similar support for Deep Learning models as well

SPARK MOMENTUM

Spark Meetups (February 2013)

1 group with 538 members
spark.meetup.com

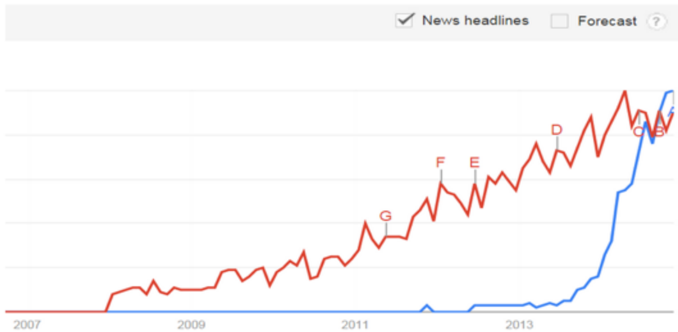
Apache Spark Meetups (August 2017)

619 groups with 406,114 members
spark.meetup.com

Open Source Impact

November 21, 2014

Spark Just Passed Hadoop in Popularity on the Web– Here's Why



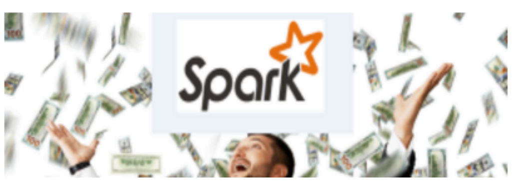
In October Apache Spark (blue line) passed Apache Hadoop (red line) in popularity according to Google Trends

datanami
BIG DATA • BIG ANALYTICS • BIG INSIGHTS
Big Data • Big Analytics • Big Insight

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.

O'REILLY®

A Data Management Inflection Point

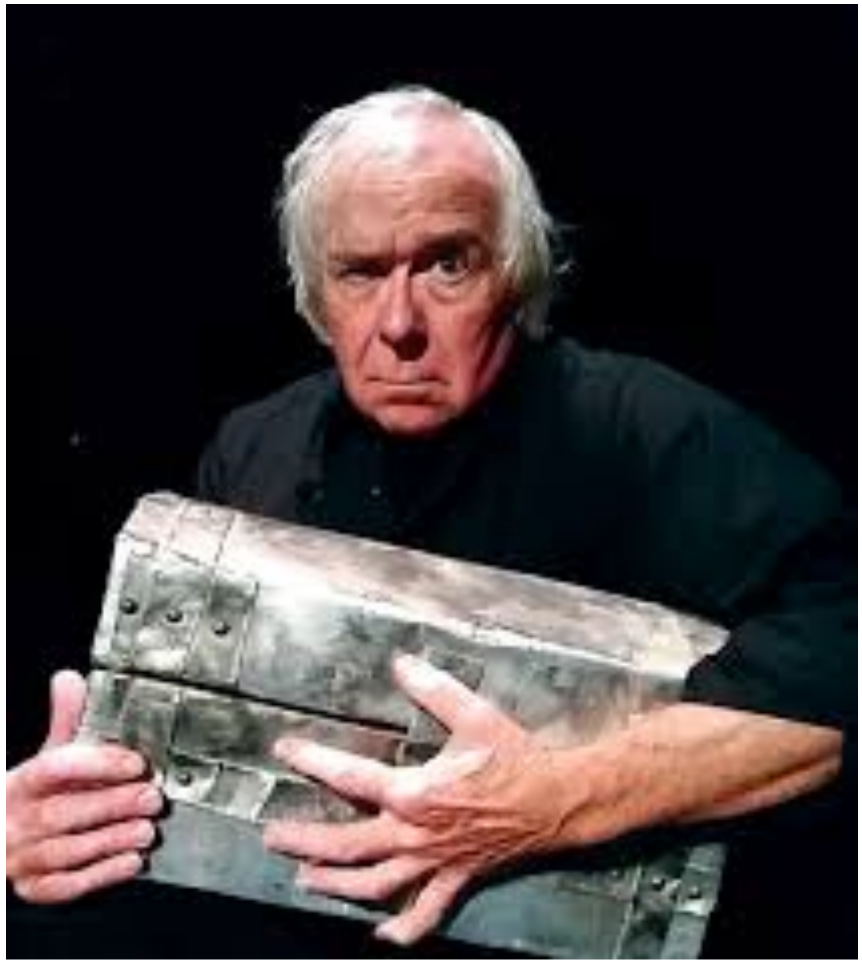
Scale Out Computing	• Processing • Storage
Elastic Resources	• Pay-as-you-go Processing • Pay-as-you-go Storage
Flexible Data Formats	• Schema on Read vs. on Write • Direct access to stored data
Multimodal Advanced Analytics	• Search, Query, Analytics • Machine Learning, AI
Open Source Ecosystem	• Rapid Adoption • Rapid Innovation

WHERE “DATABASE THINKING” CAN GET IN THE WAY

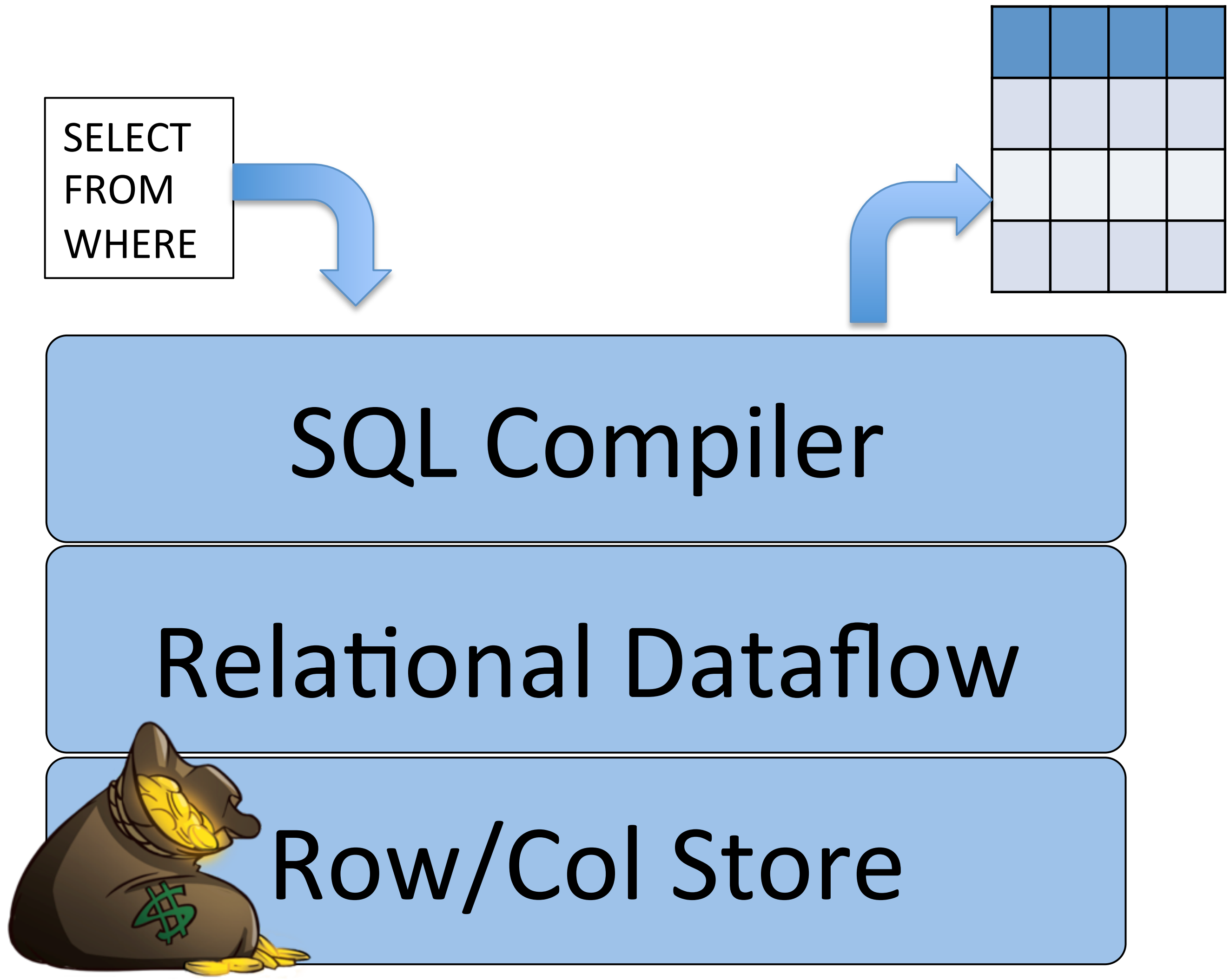
Traditional Database Thinking (analytics subset)

+ Declarative Queries and Data Independence
    - Rich Query Operators, Plans and Optimization
    - Separation of Physical and Logical Layers
+ Data existing independently of applications
    - Not as natural to most people as you'd think
+ Importance of managing the storage hierarchy
- Monolithic Systems and Control
- Schema First & High Friction
- The DB Lament: "We've seen it all before"

How Database Systems Treat Data

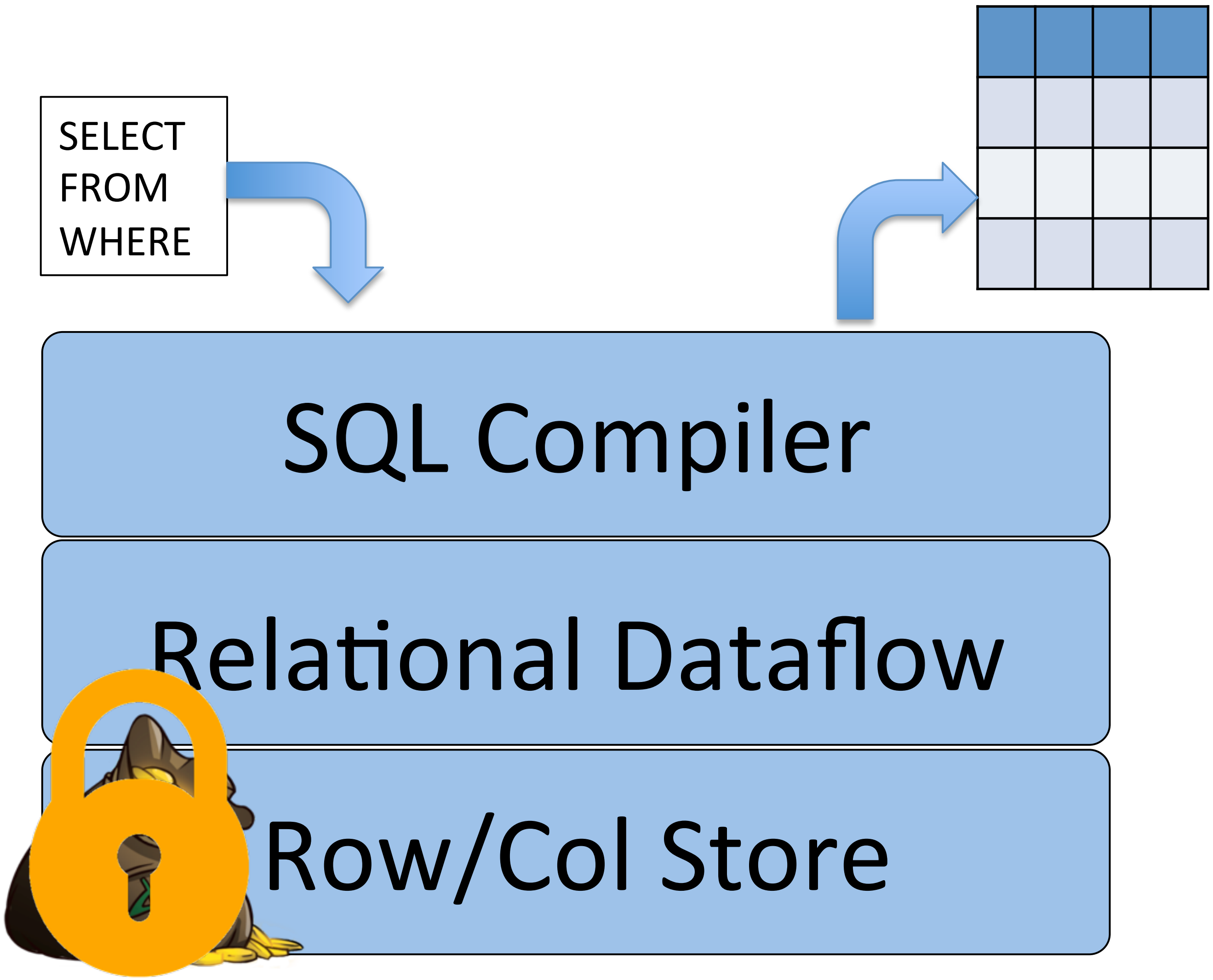


Database Systems: One way in/out



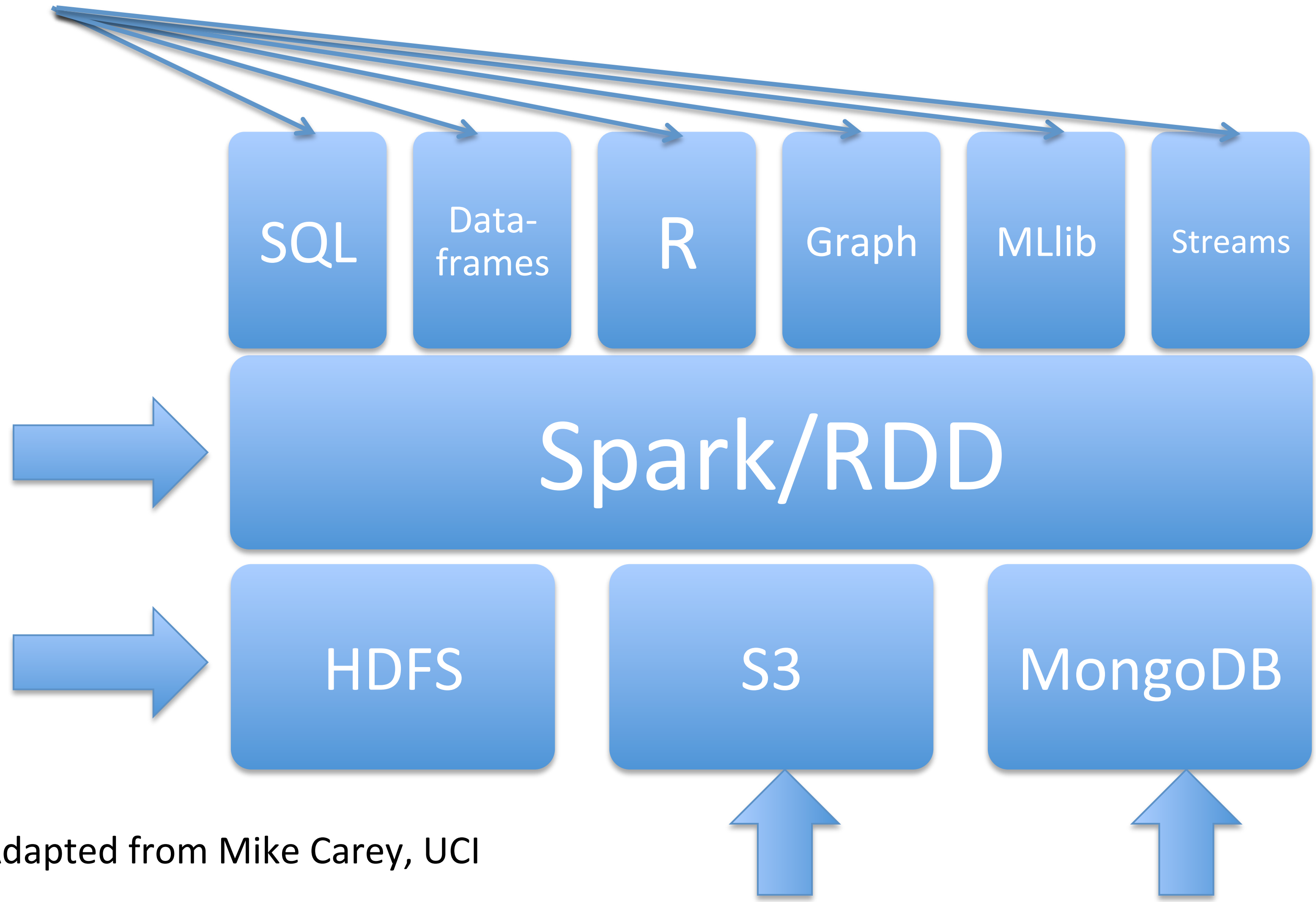
Adapted from Mike Carey, UCI

Database Systems: One way in/out



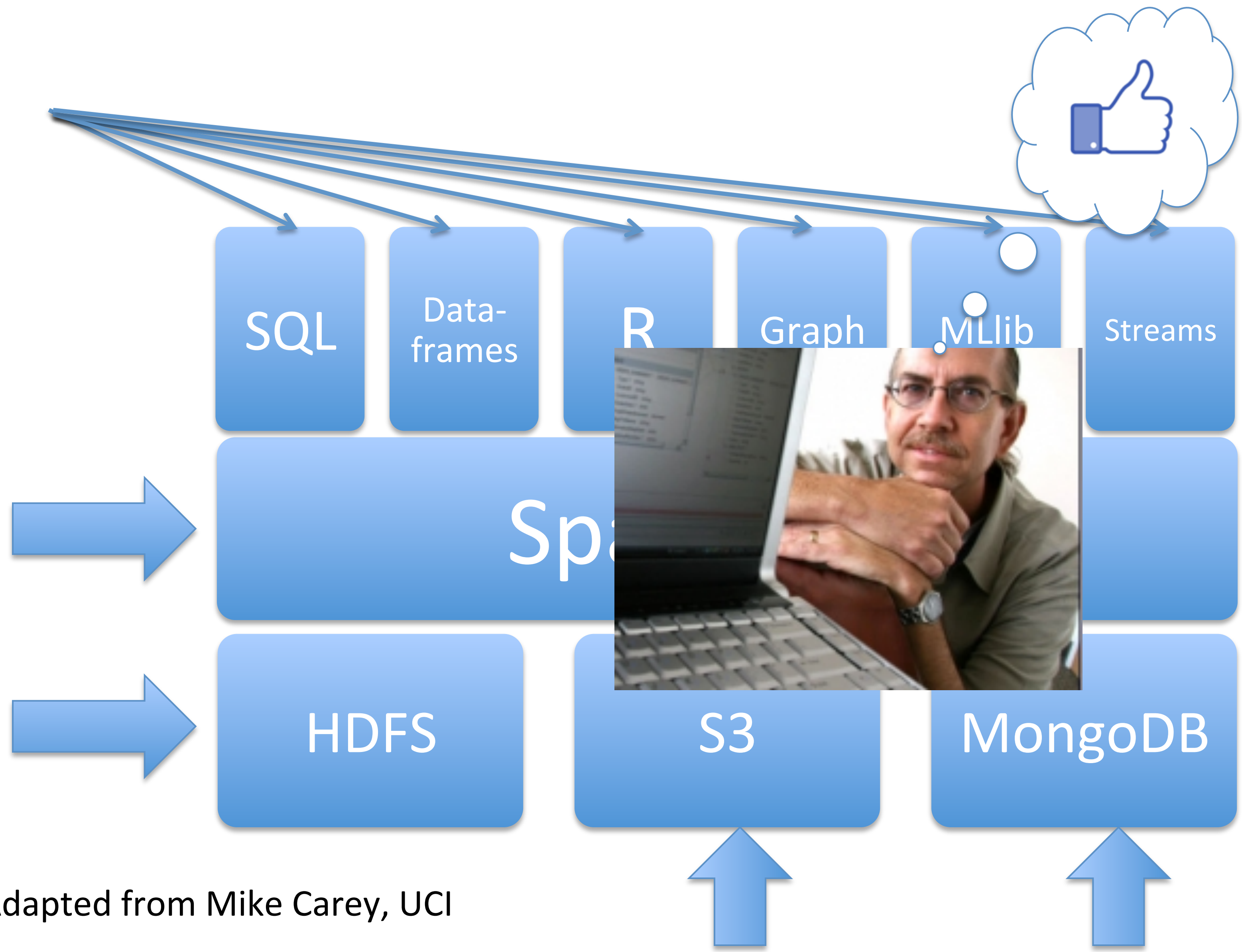
Adapted from Mike Carey, UCI

Mix and Match Data Access

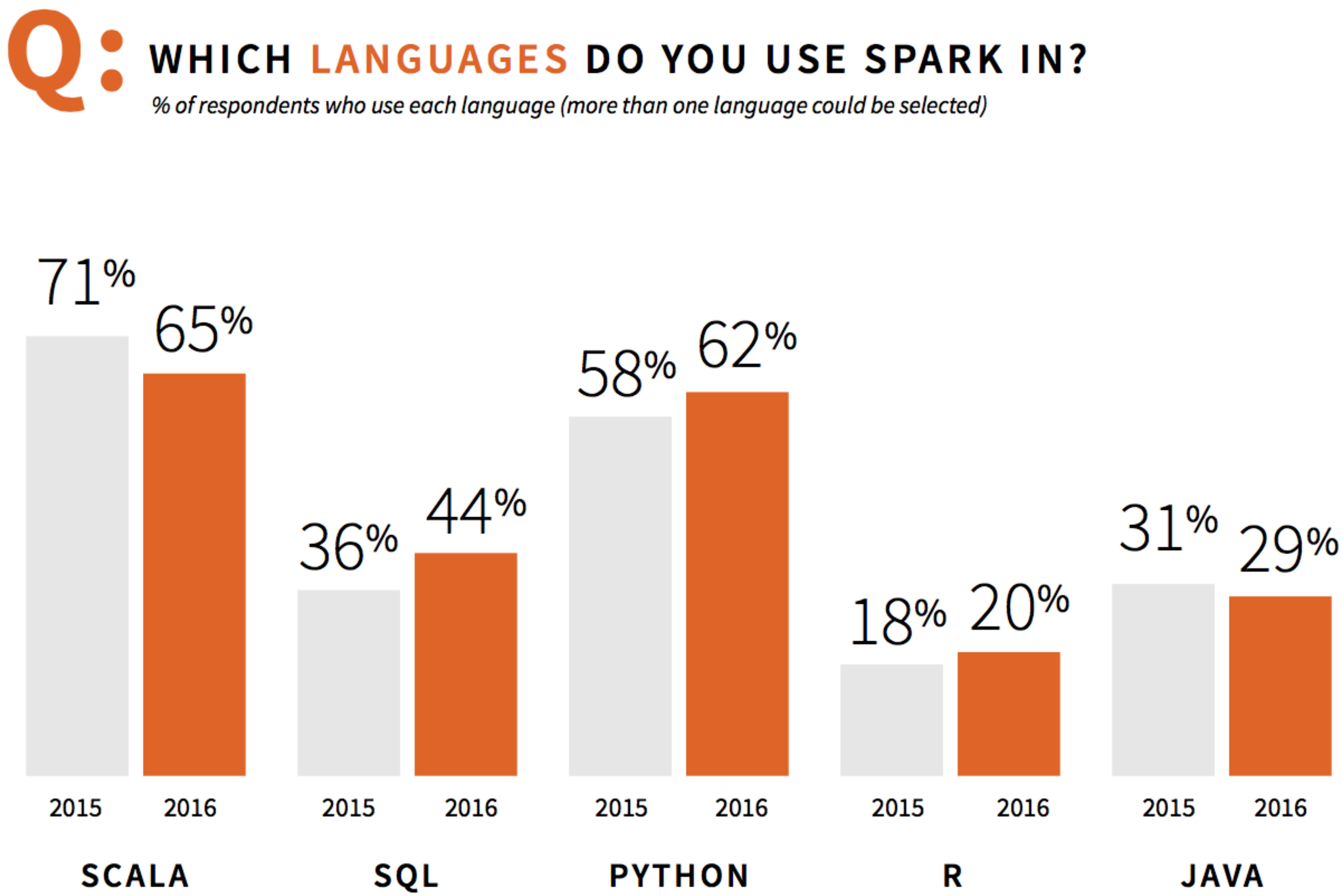


Adapted from Mike Carey, UCI

Mix and Match Data Access



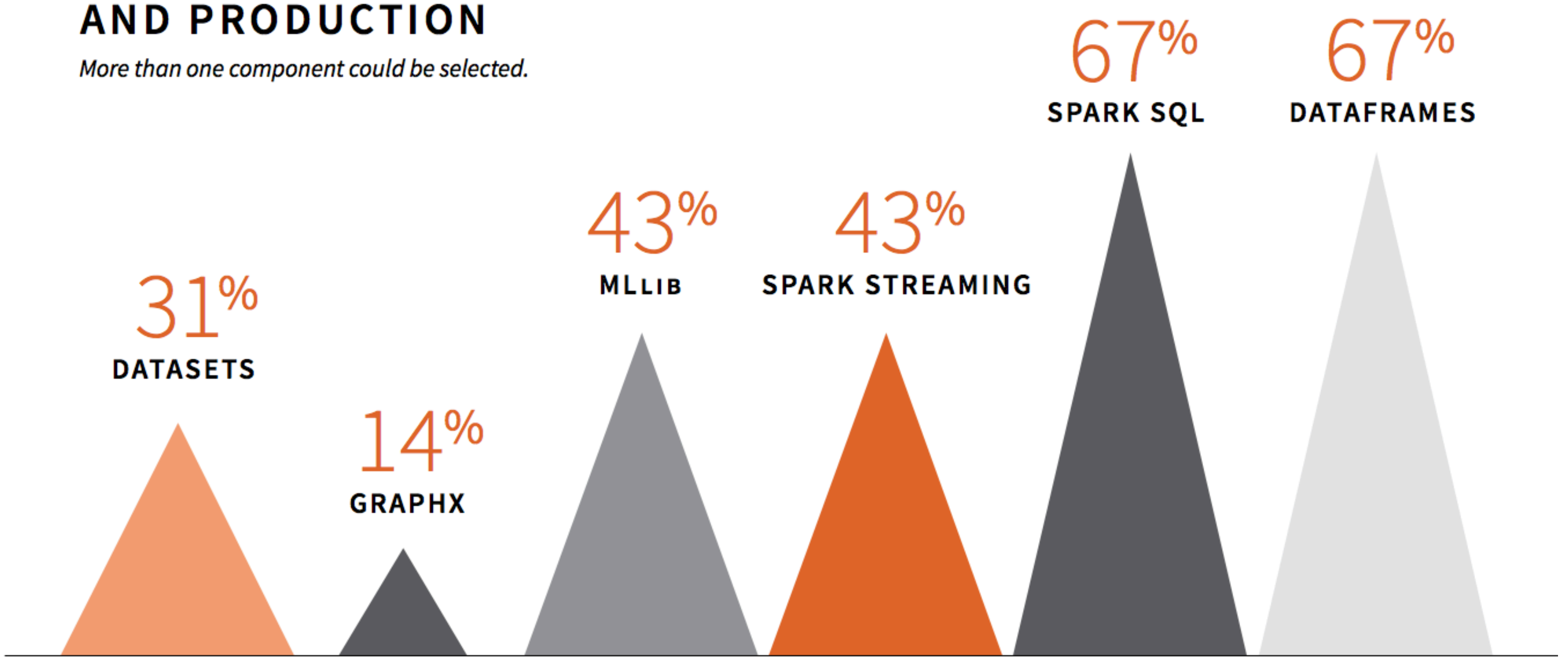
Adapted from Mike Carey, UCI



From: Spark User Survey 2016, 1615 respondents from 900 organizations
http://go.databricks.com/2016-spark-survey

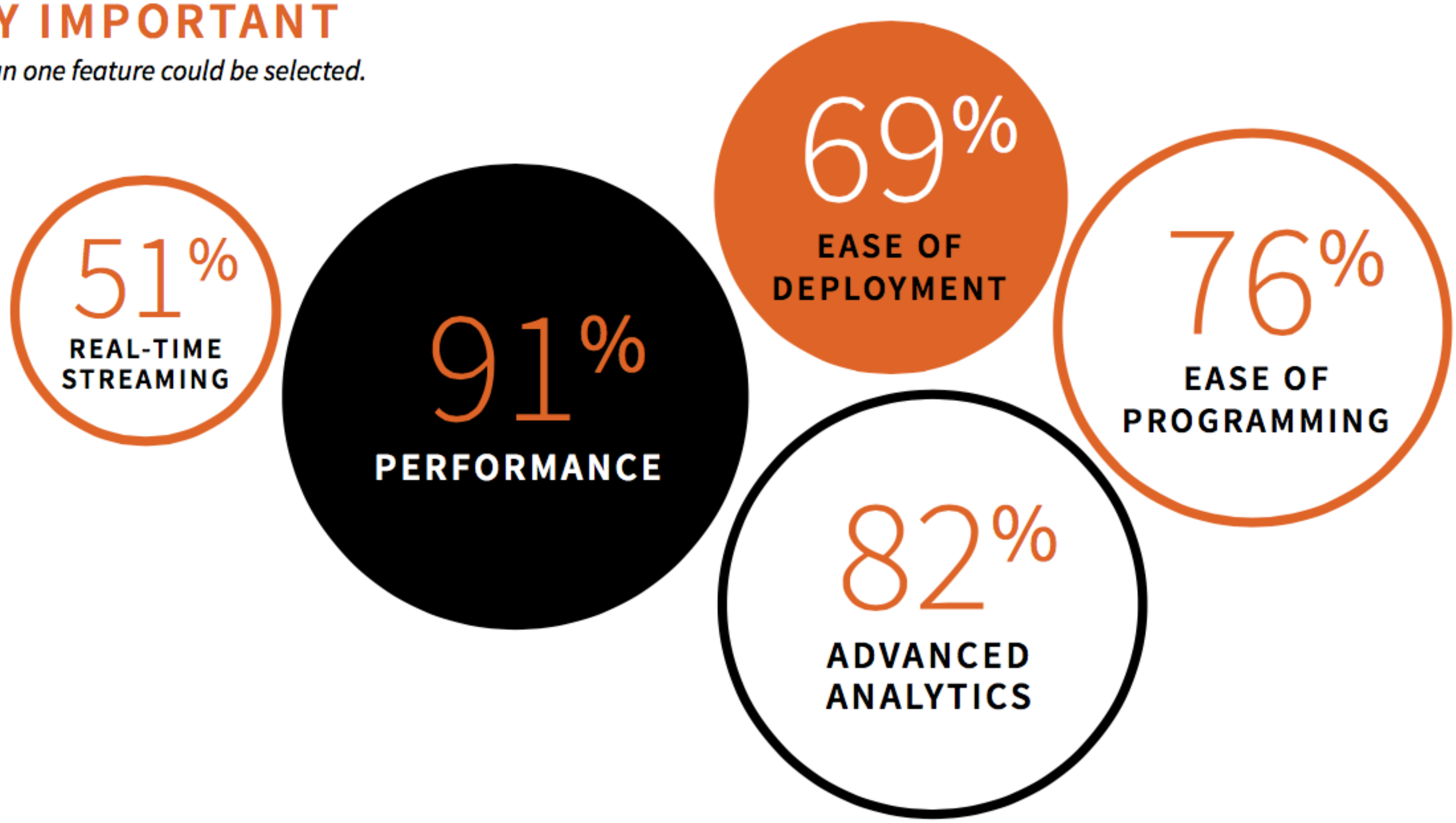
COMPONENTS USED IN PROTOTYPING AND PRODUCTION

More than one component could be selected.



% OF RESPONDENTS WHO CONSIDERED THE FEATURE VERY IMPORTANT

More than one feature could be selected.



Spark Ecosystem Attributes

- Spark focus was initially on
  - Performance + Scalability with Fault Tolerance
- Rapid evolution of functionality kept it growing
  - especially across multiple modalities: DB, Graph, Stream, ML, etc.

Database thinking is moving Spark and much of the Hadoop ecosystem up the disruptive technology value curve

Some Other Lessons

- Leverage (create) a popular ecosystem
- Build community - agree on standards: de facto or otherwise
- Solve the most common use cases and avoid complexity from others
- Ease of use + scale up/out trumps raw speed (although winning benchmarks is good for buzz)
- Hellerstein and Brewer’s 262 CS&OS merger at Berkeley set the intellectual stage

What's Next?

As we heard yesterday, rapidly changing hardware means that there is still a lot of research to be done in performance, scalability and fault tolerance!

But a new set of concerns is moving to the fore…

1) Data Science/Analytics Full Lifecycle Concerns
2) Ease of Development and Deployment
3) "Safe" Data Science and Human Factors

And how will DB Thinking help???

Data Science – NSF CISE December 2016

National Science Foundation
WHERE DISCOVERIES BEGIN

If NSF can help foster the evolution and development of both Data Science and Data Scientists over the next decade, we can begin to meet the potential of Data Science to drive new discovery and innovation…

This should include not only a focus on fundamental Data Science, but also on **translational efforts** to move ideas from research to practice across the broadest landscape of commercial applications.

CISE AC Data Science Report

REALIZING THE POTENTIAL OF DATA SCIENCE

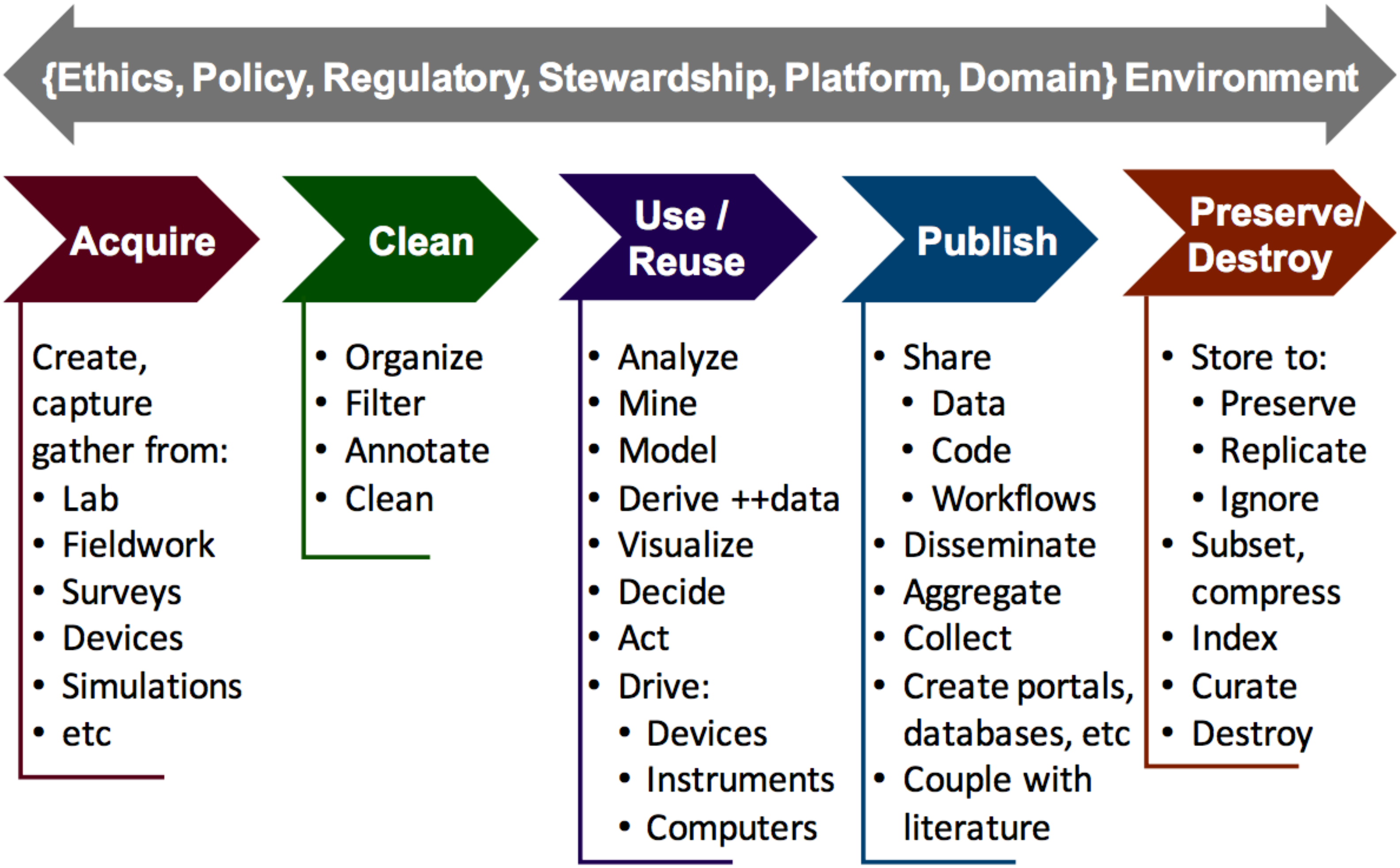
Final Report from the National Science Foundation Computer and Information Science and Engineering Advisory Committee Data Science Working Group

Francine Berman and Rob Rutenbar, co-Chairs
Henrik Christensen, Susan Davidson, Deborah Estrin, Michael Franklin, Brent Hailpern, Margaret Martonosi, Padma Raghavan, Victoria Stodden, Alex Szalay

December 2016

The function of Federal advisory committees is advisory only. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the Advisory Committee, and do not necessarily reflect the views of the National Science Foundation.

Data Science & Analtyics: A Lifecycle View



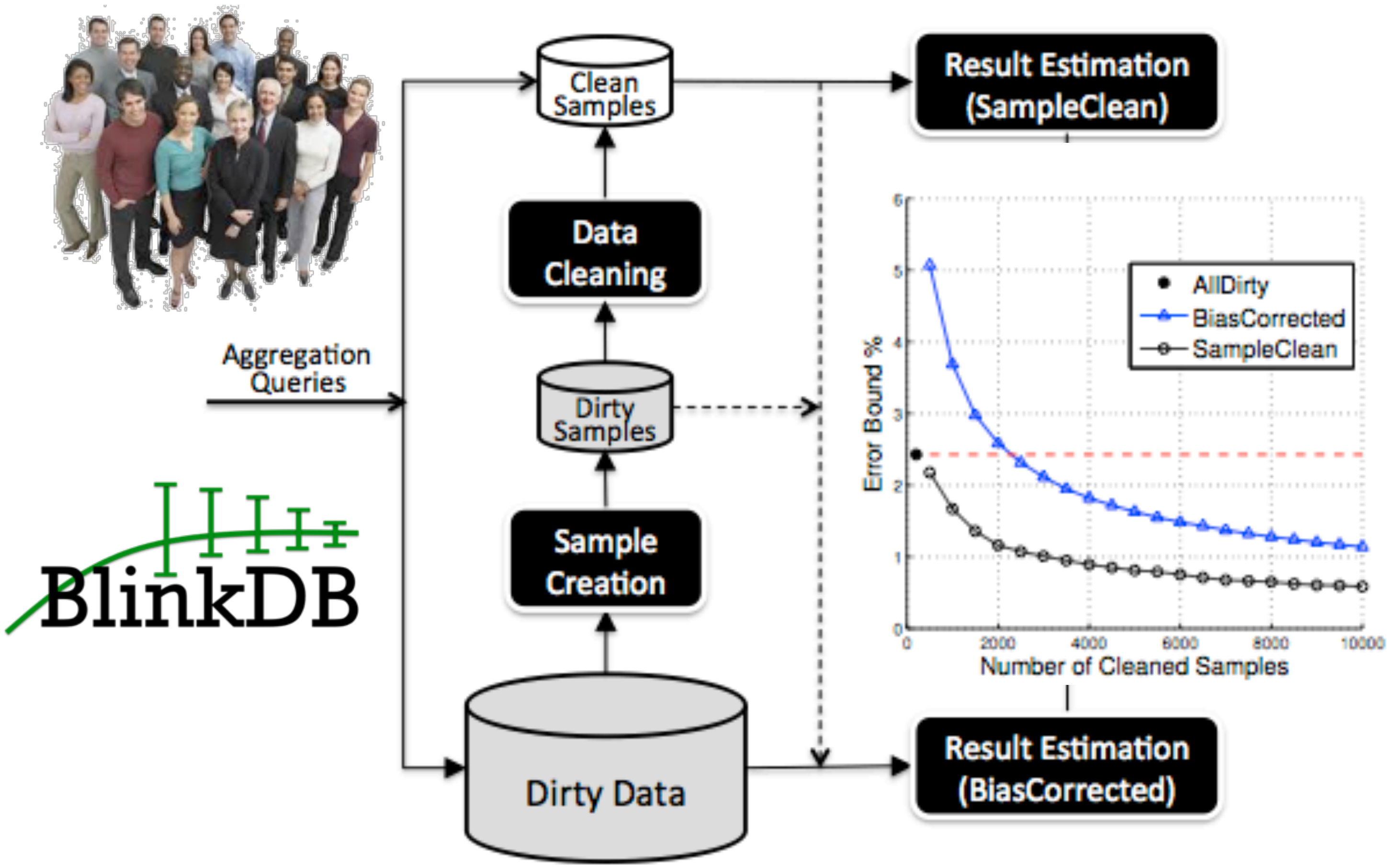
from the National Science Foundation CISE AC Data Science Report, October 2016

Data “Wrangling”

- Claim: Up to 80% of time spent on cleaning, integrating and preparing data for analysis
- Problems include:
  - Data acquisition and characterization
  - Correcting values and imputing missing data
  - (Re) Formatting
  - Dynamic and evolving data sources
- Data Integration from heterogeneous sources
- Semantic and Performance issues arise
- Machine Learning and Human Processing solutions

Data Cleaning: SampleClean

Key Systems Issues – how to deal with latency and cost of the crowd?



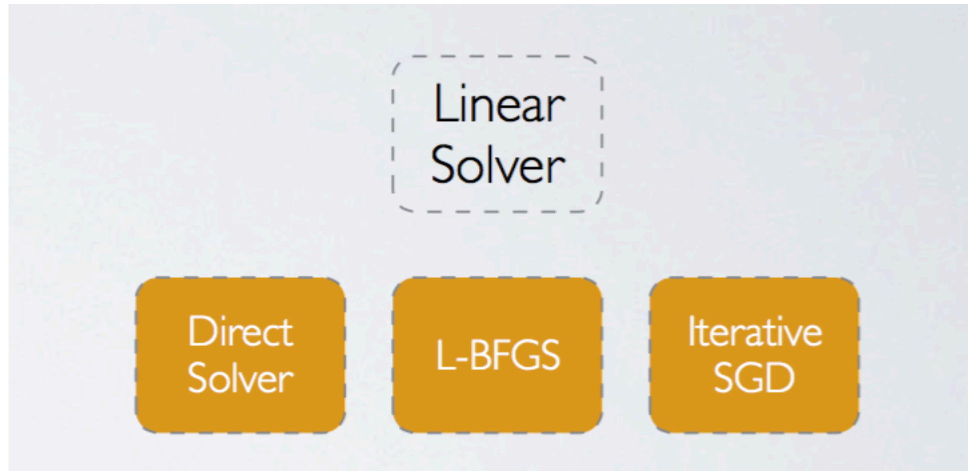
J. Wang, S. Krishnan, *et al.*, A Sample-and-Clean Framework for Fast and Accurate Query Processing on Dirty Data, *SIGMOD 2014*

Ease of Development/Deployment

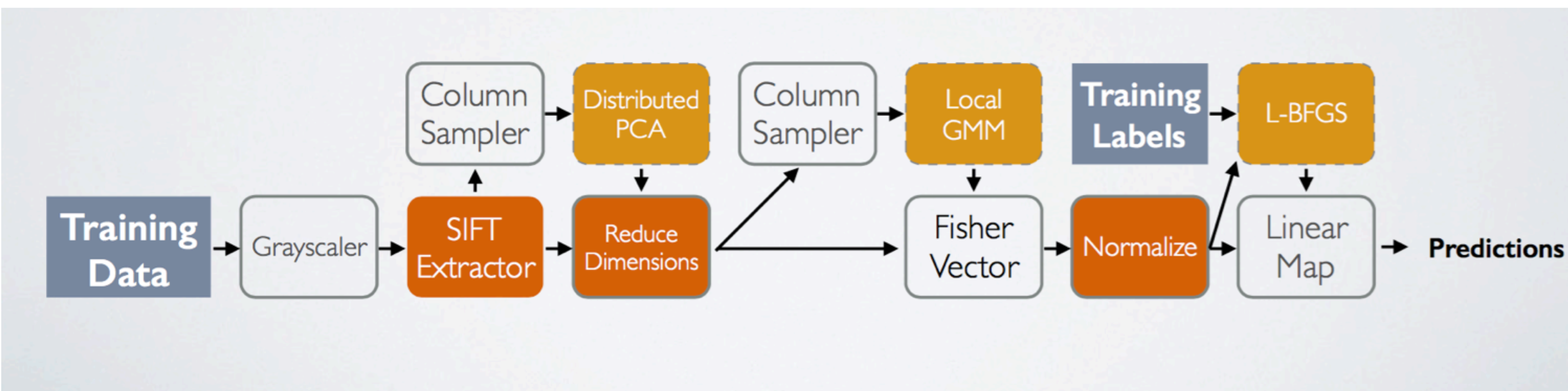
- Data Analytics is a complex process
- Rare to simply run a single algorithm on an existing data set
- Emerging systems support more complex workflows:
    - Spark MLPipelines
    - Google TensorFlow
    - KeystoneML and Clipper Model Serving (BDAS)

Declarative API → Optimizations
(c.f., Database Query Optimization)

Automated ML operator selection

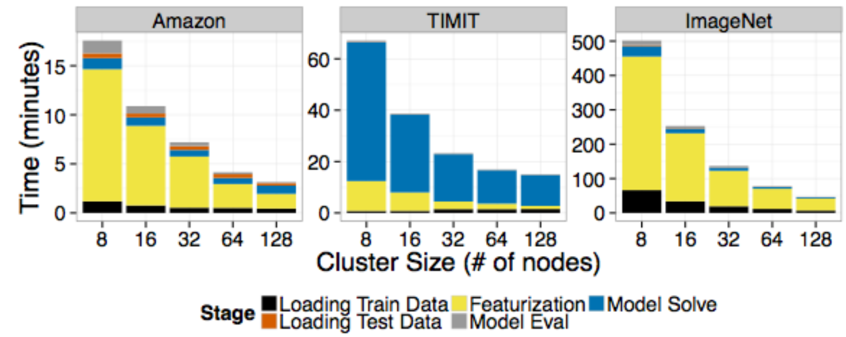


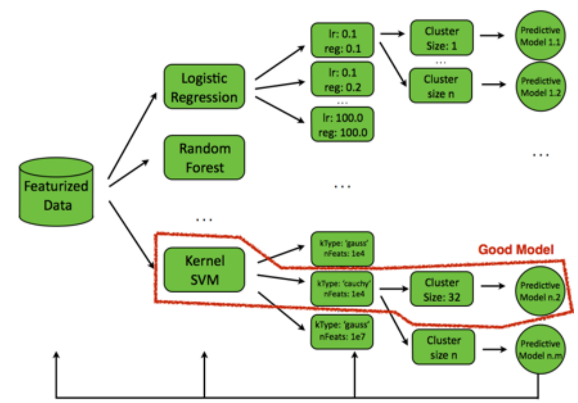
Auto-caching for iterative workloads

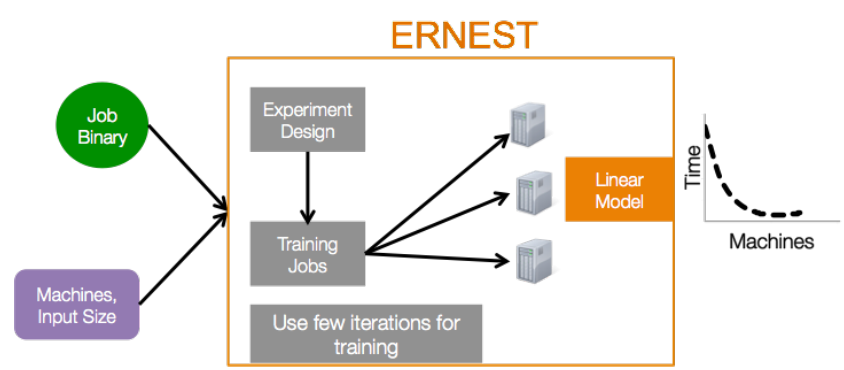


KeystoneML

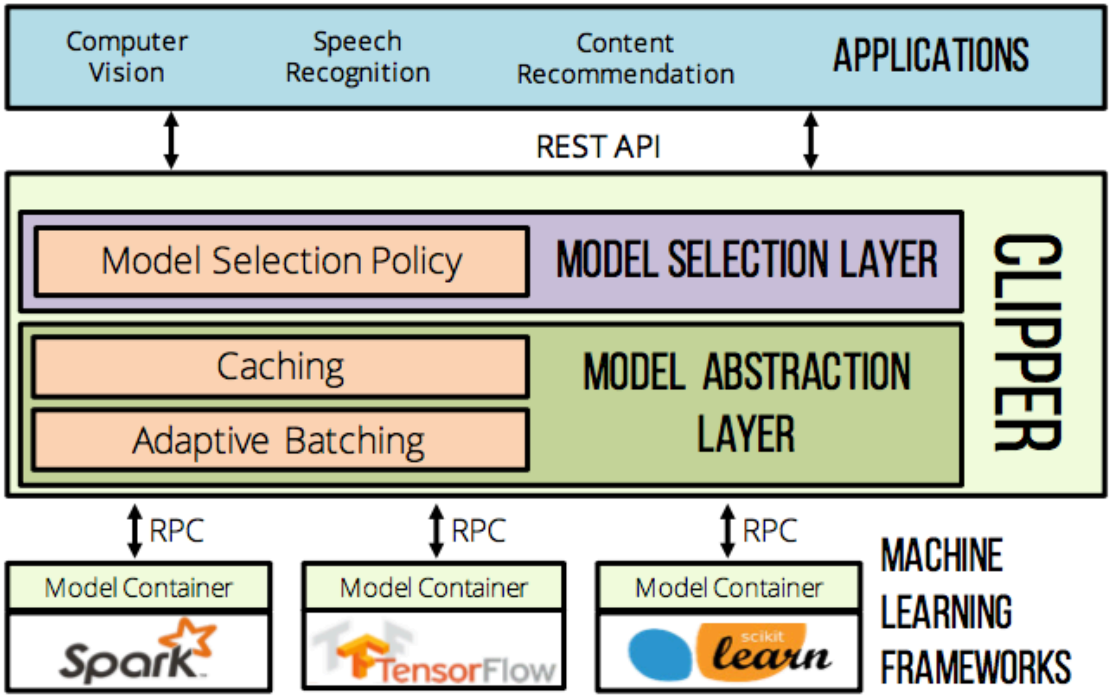
- Current version: v0.3
- Scale-out performance on 10s of TBs of training features on 100s of machines. apps: Image Classification, Speech, Text.
- First versions of node-level and whole-pipeline optimizations.
- KeystoneML system design – ICDE 2017
- Other Results:
  – Principled, scalable hyperparameter tuning. (TuPAQ - SoCC 2015)
  – Advanced cluster sizing/job placement algorithms. (Ernest - NSDI 2016)







Deployment: Model Serving



Note the similarity with the traditional RDS/RSS Split

Clipper: A prediction serving system that spans multiple ML frameworks

- Simplifies model serving
- Bounds latency and increases prediction throughput
- Enables real-time learning and personalization across machine learning frameworks

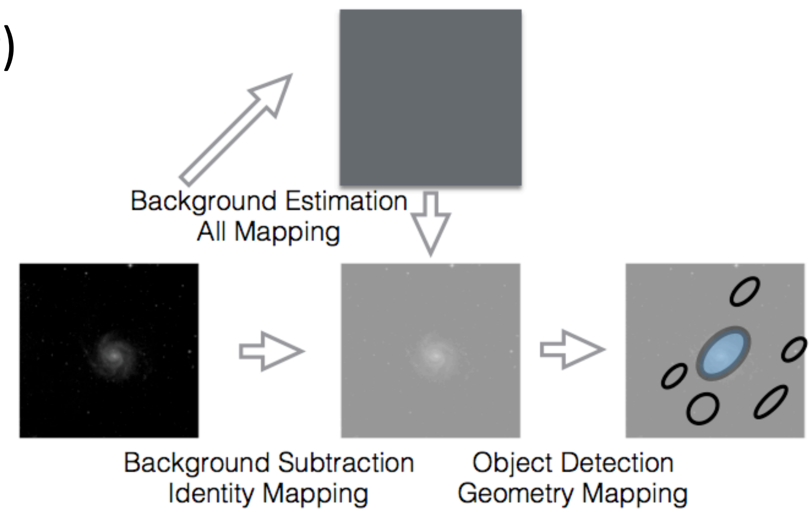
https://github.com/ucbrise/clipper

D. Crankshaw et al., "Clipper: A Low-Latency Online Prediction Serving System", *NSDI Conf.,* March 2017

Curation and Reproducibility

Data outlives any particular application:

"[database systems] let you use one set of data in multiple ways, including ways that are unforeseen at the time the database is built and the 1st applications are written." (Curt Monash, analyst/blogger)



Z. Zhang et al. HPDC 17:

- Efficient fine-grained lineage for machine learning and advanced analytics pipelines
- Supports code debugging, result analysis, data anomaly removal and computation replay
- Provides interactive answers to queries over lineage

Bias, Privacy and Ethical Issues

SALON VIDEO NEWS POLITICS ENTERTAINMENT LIFE INNOVATION

SUNDAY, MAY 14, 2017 09:00 AM HKT

Why big-data analysis of police activity is inherently biased

One predictive policing algorithm targeted black neighborhoods at roughly twice the rate of white neighborhoods

sign in become a supporter subscribe search jobs dating more International edition

theguardian

UK world sport football opinion culture business lifestyle fashion environment tech travel all sections

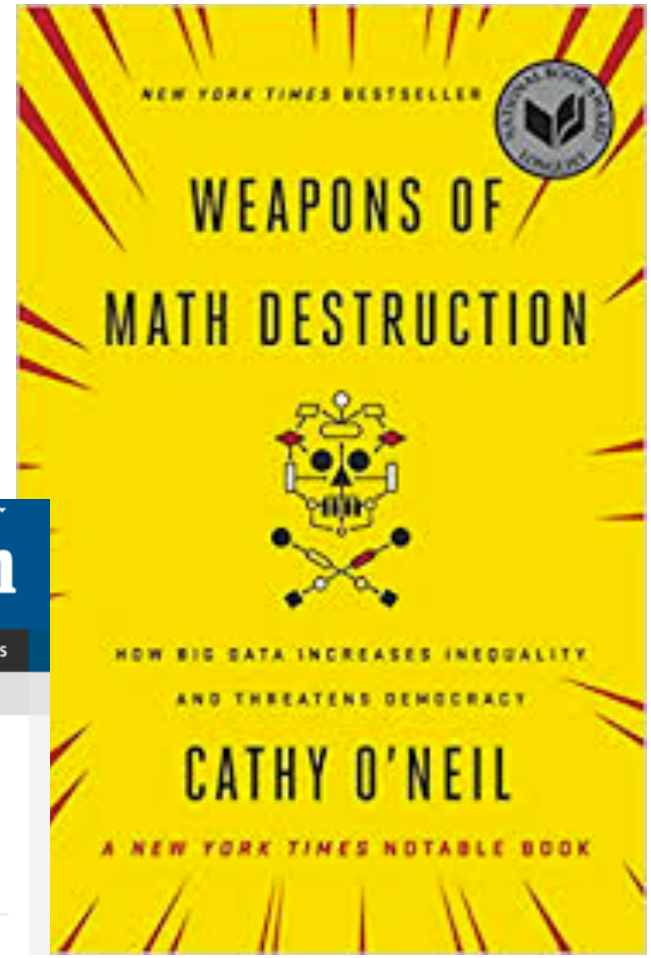
home › tech

Big data
Big data development (series)

Little privacy in the age of big data

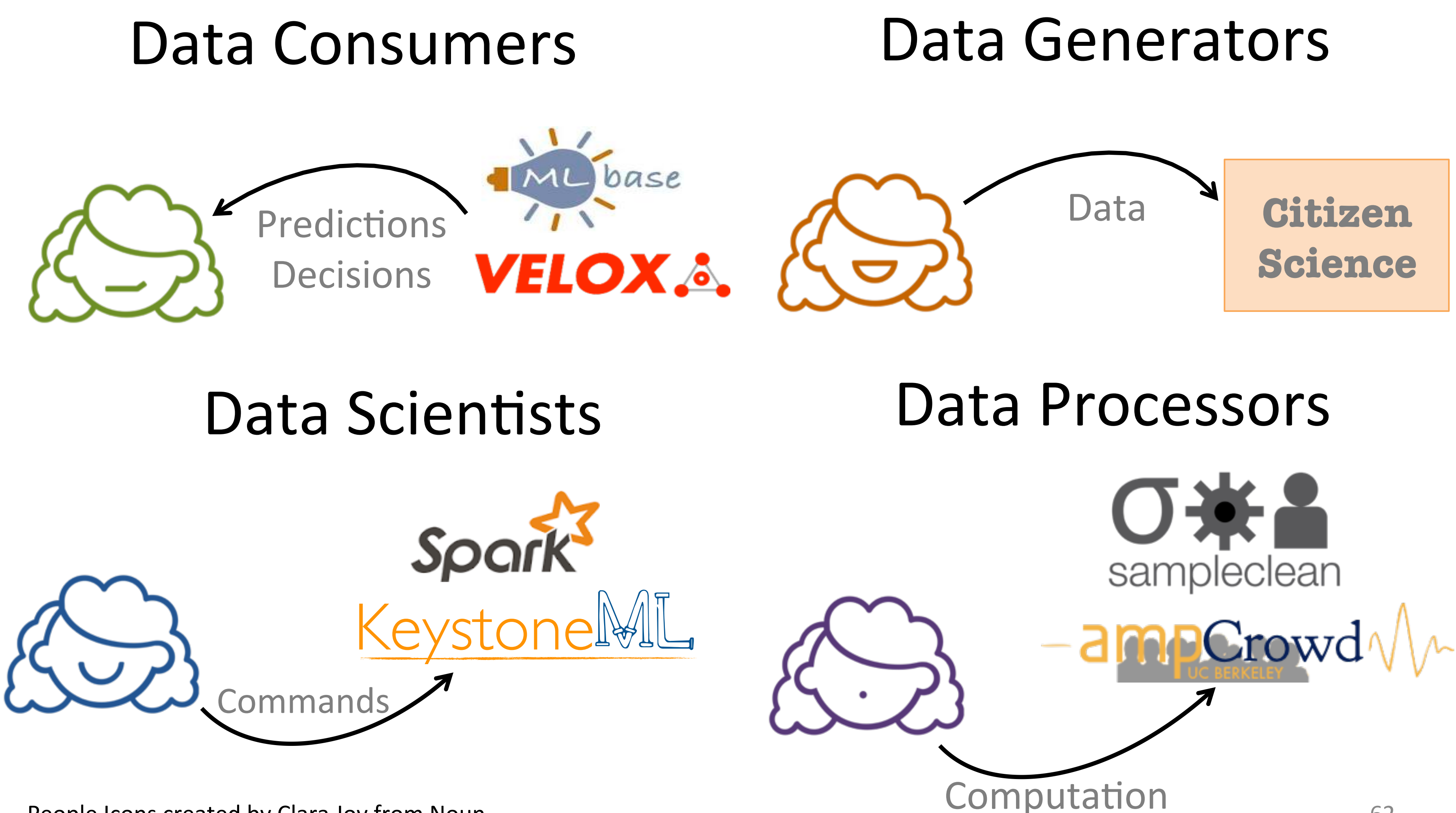
With massive amounts of our personal data now being routinely collected and stored, privacy breaches are almost inevitable

Most popular



“With Big Data comes Big Resposnibility”

Humans in the loop

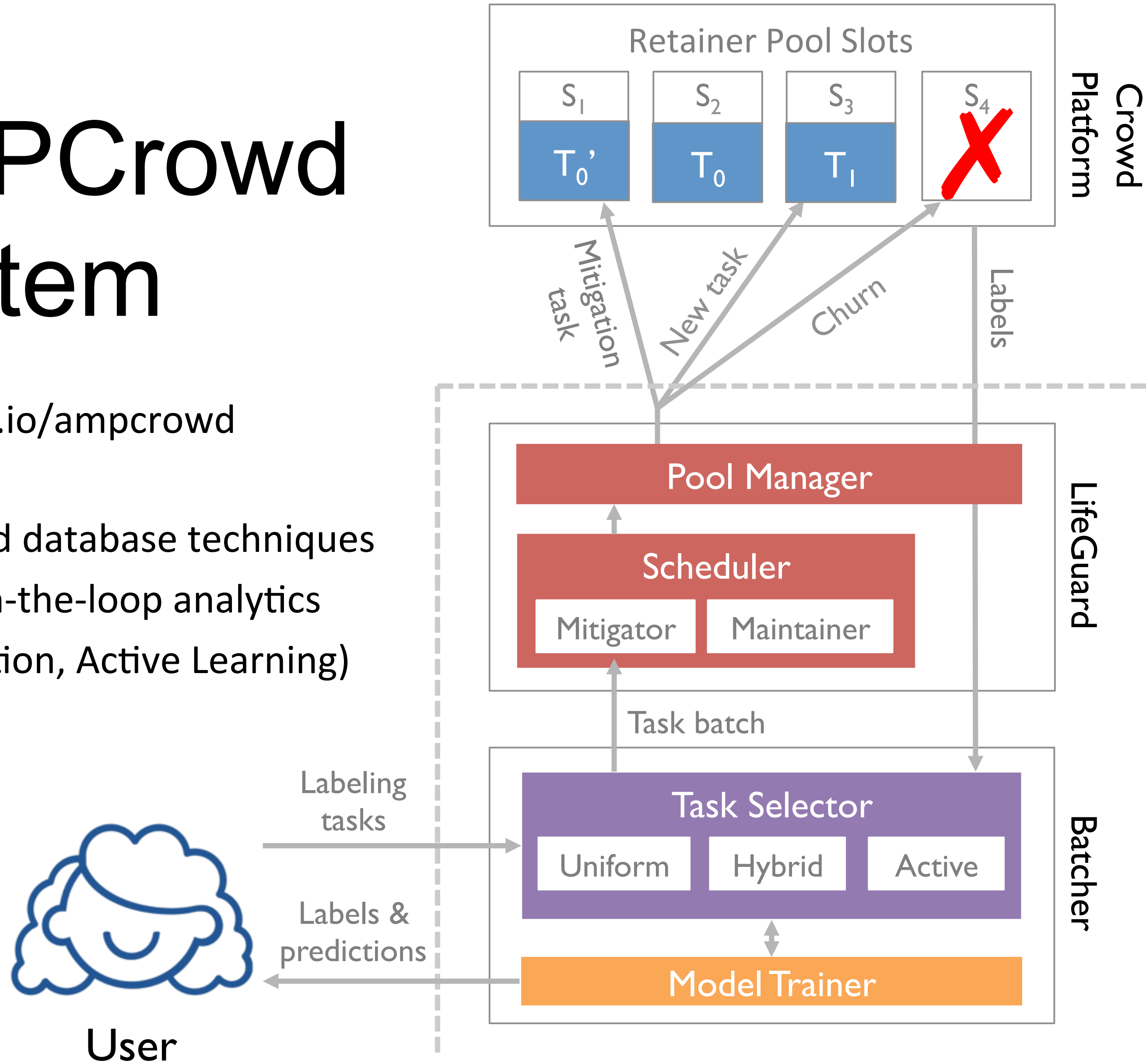


People Icons created by Clara Joy from Noun Project

The AMPCrowd System

amplab.github.io/ampcrowd

Leveraging systems and database techniques
for hybrid human-in-the-loop analytics
(e.g. Straggler Mitigation, Active Learning)



D. Haas, *et al.*, Clamshell: Scaling Up Crowds for Low Latency Data Labeling, *PVLDB 9(4)*
Haas & Franklin, Cioppino: Multi-tenant Crowdsourcing, *HCOMP 2017*

Closer Integration With Domains

- Jim Gray and Alex Szalay showed the mutual benefits between databases and science that can gained by close collaboration
- The widespread creation of new Data Science Institutes provides institutional support for such efforts
- DB program committees much be encouraged to recognize this type of work
- (this was the topic of yesterday’s panel)

New Challenges Summary

Performance, Scalability, and Fault Tolerance remain important, but we face new challenges, <u>including</u>:

Data Science Lifecycle

- Data Acquisition, Integration, Cleaning (i.e., wrangling)
- Data Integration remains a "wicked problem"
- Model Building
- Communicating results, Curation, "Translational Data Science"

Ease of Development and Deployment

- Can leverage database ideas (e.g., declarative query optimization)
- New components for "model serving" and "model management"

"Safe" Data Science

- end-to-end Bias Mitigation
- Security, Ethics and Data Privacy
- Explaining and influencing decisions
- Human-in-the-loop

(and don't ignore Deep Learning…)

Conclusions

- The Database field is seeing tremendous change from above and below
- Big Data software is a classic Disruptive Technology
- Database Thinking is key to moving up the value chain
- But we'll also have to shed some of our traditional inclinations in order to make progress

Acknowledgements

Thanks to all the amazing AMPLab students, staff, faculty and sponsors
and to the pioneers who developed
our increasingly central field
as well as to those who continue to push the boundaries
(apologies to anyone left out of the pictures!)

Thanks and for More Info

Mike Franklin

mjfranklin@uchicago.edu