Spark Libraries

Justin Pihony

@JustinPihony | justin-pihony.blogspot.com

Course Overview

- Basics of Spark
- Core API
- Cluster Managers
- Spark Maintenance

- Libraries
  - SQL
  - Streaming
  - MLlib/GraphX
- Troubleshooting / Optimization
- Future of Spark

Section Overview

- Libraries
  - SQL
  - Streaming
  - MLlib/GraphX

Spark SQL

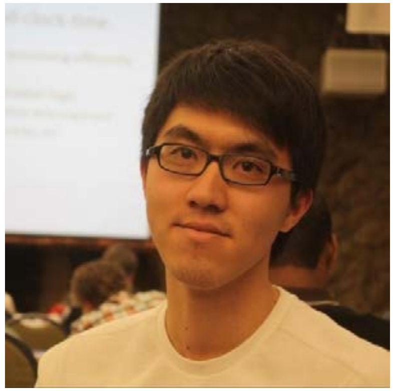
```
mystructuredData.registerTempTable(“SparkTable”)


sqlContext.sql(“SELECT * FROM SparkTable
                WHERE SomeColumn == ‘SomeData’”)
```

> As Spark continues to grow, we want to enable wider audiences beyond "Big Data" engineers to leverage the power of distributed processing.

— Databricks blog (http://bit.ly/17NM70s)

Spark SQL Maintainers

Michael Armbrust



Reynold Xin

Spark SQL

Data Sources

JSON

AMAZON REDSHIFT

Parquet

AVRO™

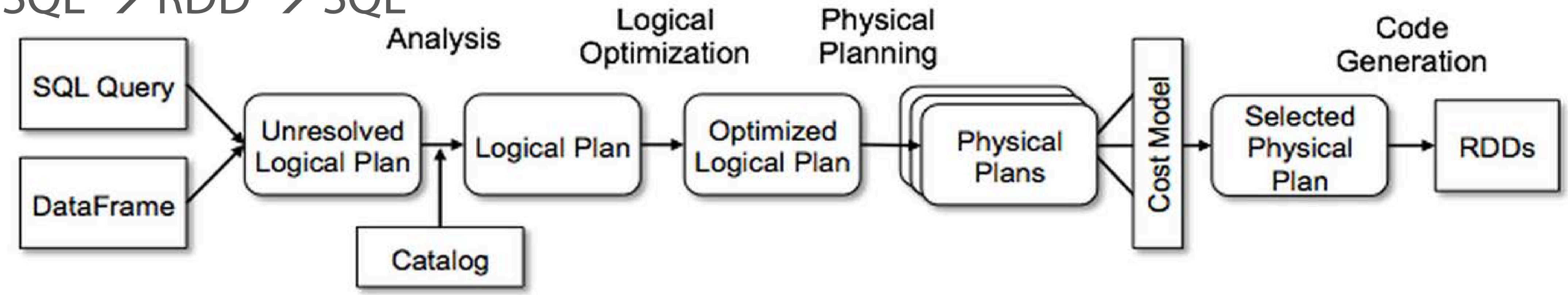
HIVE

CSV

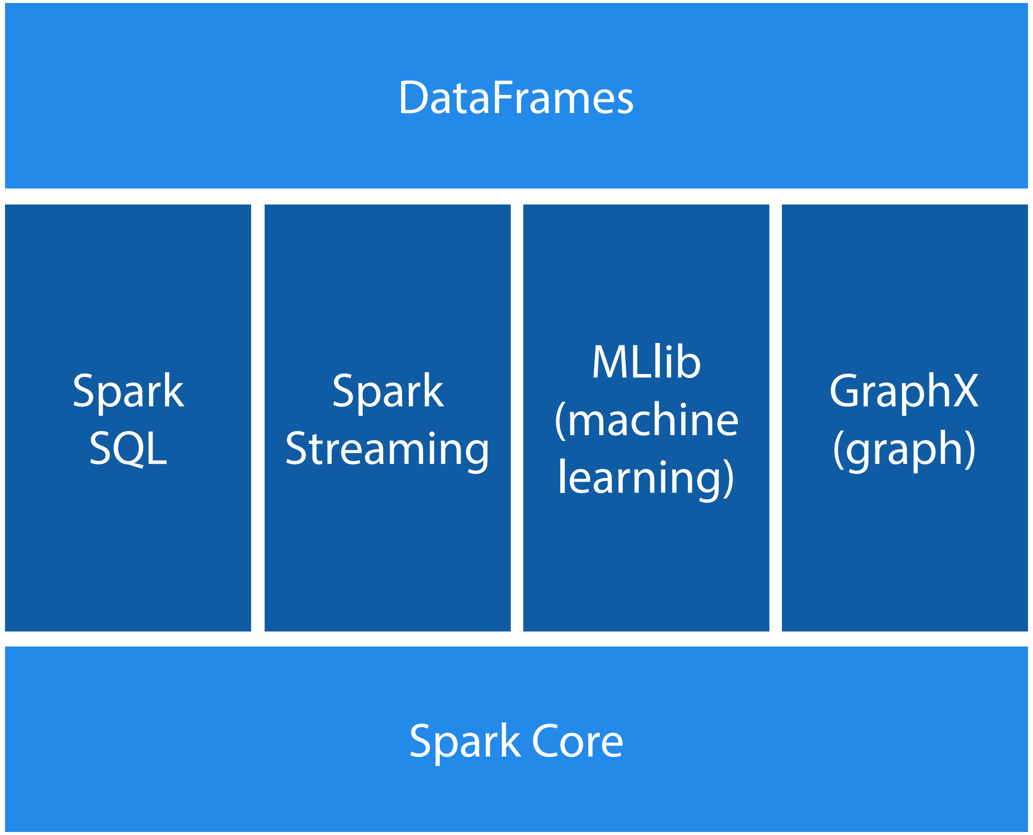
• • •

Spark SQL

- Optimizations
  - Predicate push down
  - Column pruning
- Uniform API
- Code generation == Performance gains
- SQL → RDD → SQL



Spark SQL



DataFrames

Spark SQL

Spark Streaming

MLlib (machine learning)

GraphX (graph)

Spark Core

> This new API makes Spark programs more concise and easier to understand, and at the same time exposes more application semantics to the engine.
>
> — Databricks blog (http://bit.ly/1FBRTBA)

DataFrames

Experimental

SPARK-6116

```
sqlContext.createDataFrame(pandas)
dataFrame.toPandas()
```

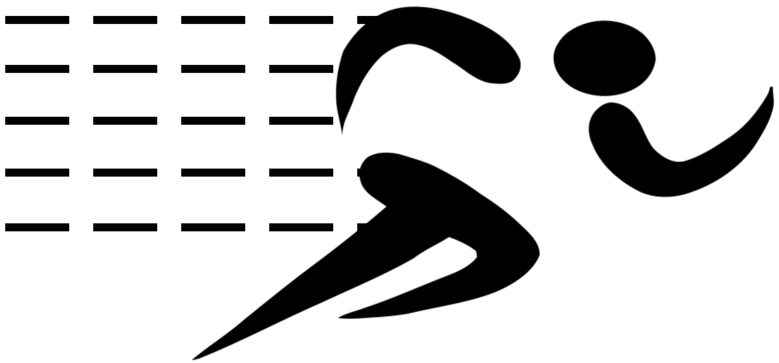
```
createDataFrame(sqlContext, RDataFrame)
collect(df)
```

Spark Streaming Maintainers

Tathagata Das

Matei Zaharia

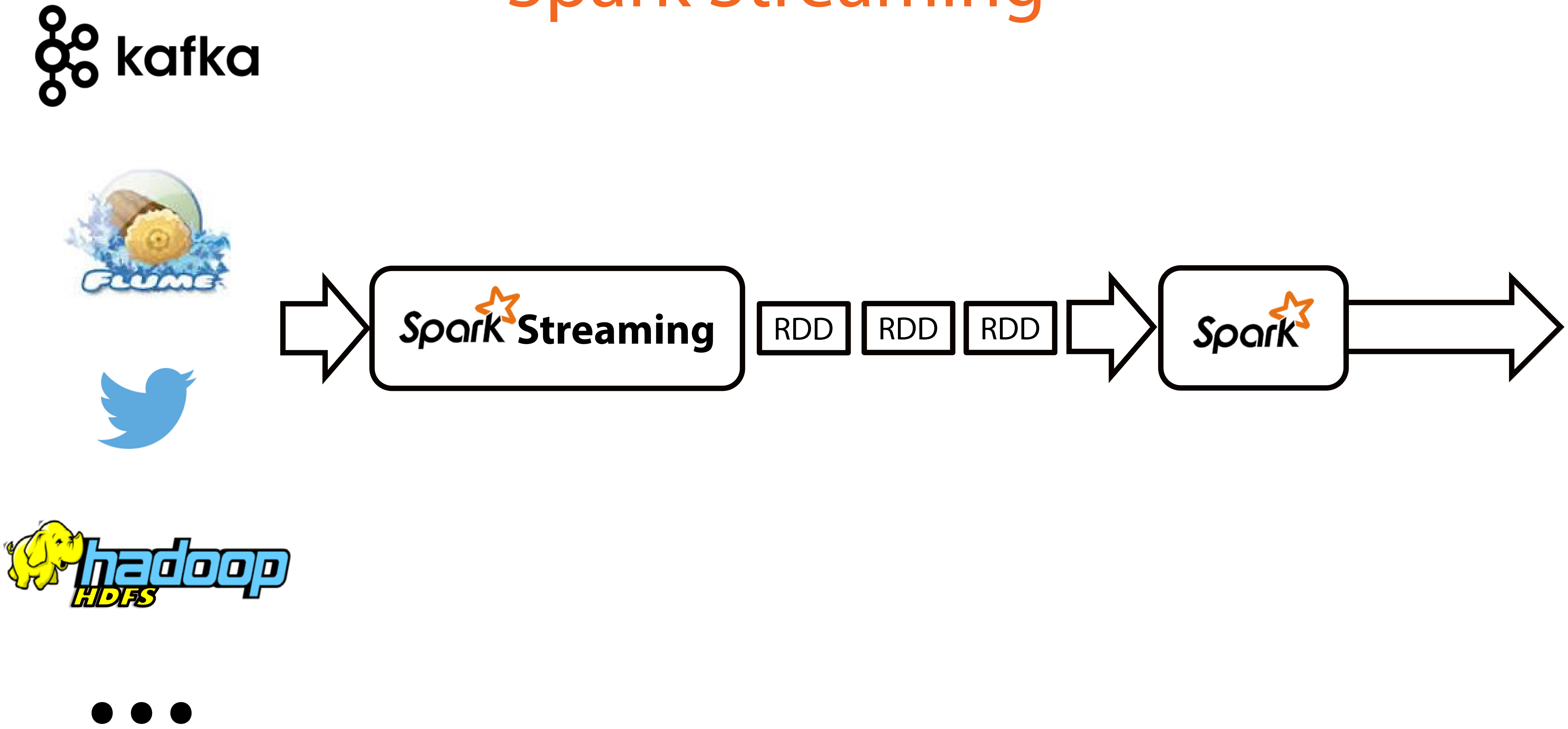
Spark Streaming



1

Spark Streaming

Spark Streaming



Spark Streaming

Python API

MLlib Maintainers

Joseph Bradley

Xiangrui Meng

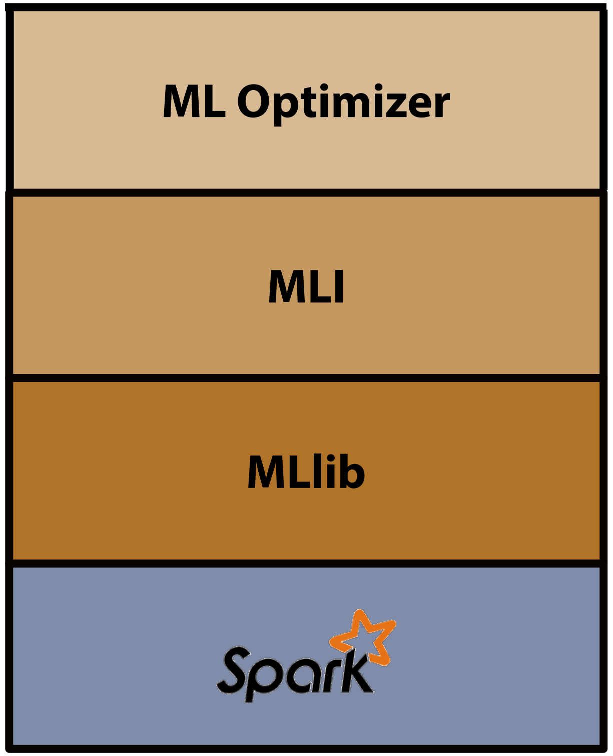
Shivaram Venkataraman

Matei Zaharia

MLlib

MATLAB

R



ML Optimizer

MLI

MLlib

Spark

mahout

GraphLab

MLlib

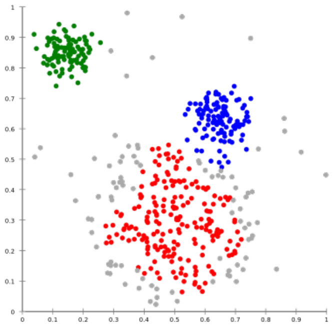
`org.apache.spark.mllib`

MLlib

```
org.apache.spark.mllib
                .ml
```

MLlib

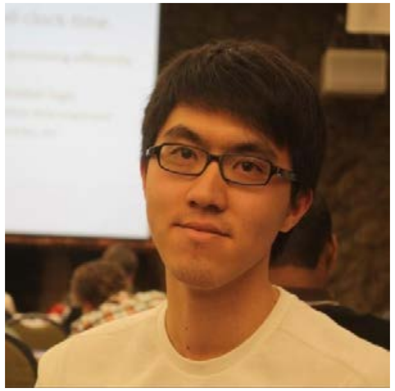
- Algorithms
  - Classification
  - Regression
  - Collaborative Filtering
  - Clustering
  - Dimensional Reduction
- Feature Extraction and Transformation



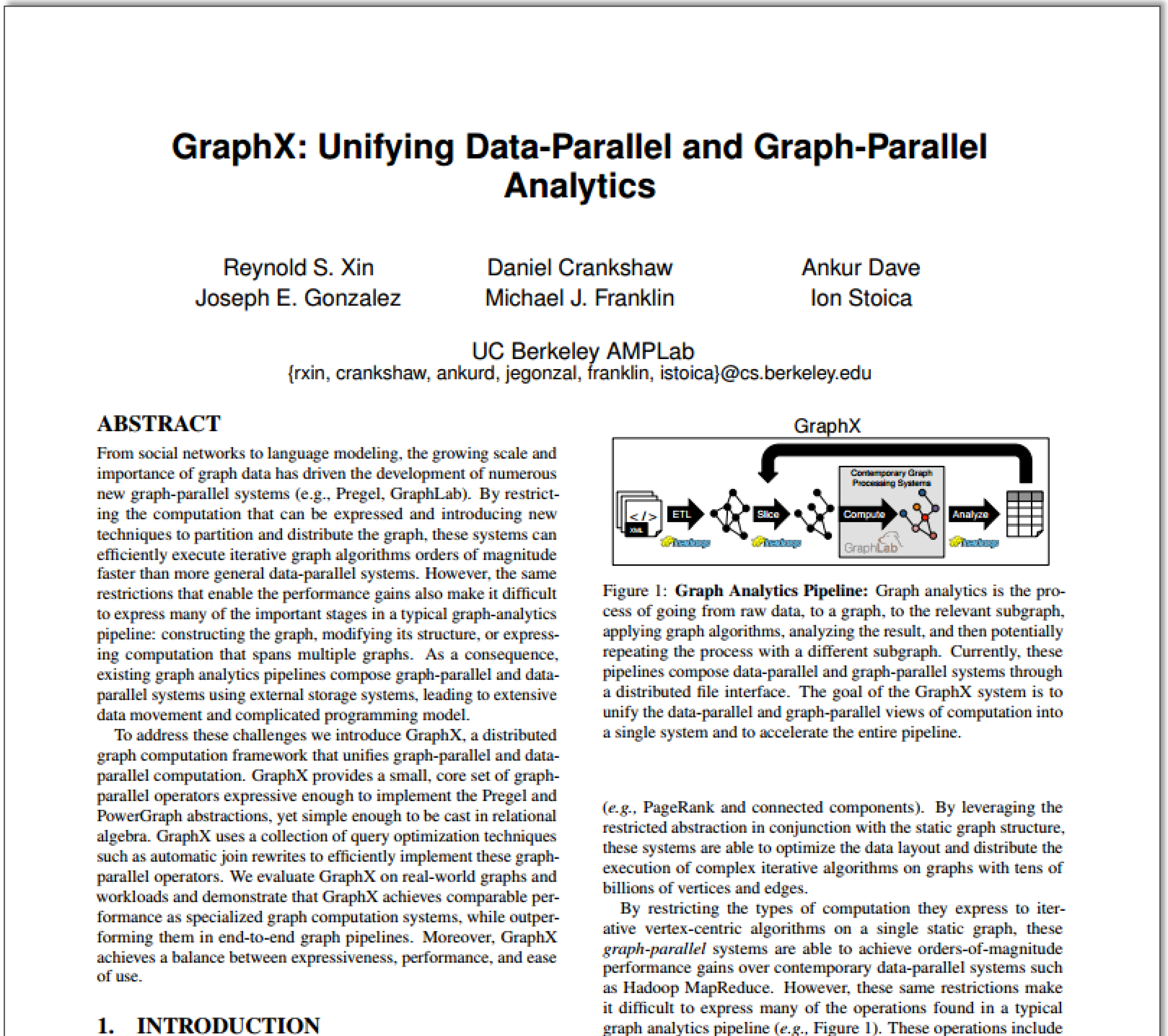
GraphX Maintainers

Ankur Dave

Joseph Gonzalez



Reynold Xin



GraphX: Unifying Data-Parallel and Graph-Parallel Analytics

Reynold S. Xin, Daniel Crankshaw, Ankur Dave
Joseph E. Gonzalez, Michael J. Franklin, Ion Stoica

UC Berkeley AMPLab
{rxin, crankshaw, ankurd, jegonzal, franklin, istoica}@cs.berkeley.edu

ABSTRACT

From social networks to language modeling, the growing scale and importance of graph data has driven the development of numerous new graph-parallel systems (e.g., Pregel, GraphLab). By restricting the computation that can be expressed and introducing new techniques to partition and distribute the graph, these systems can efficiently execute iterative graph algorithms orders of magnitude faster than more general data-parallel systems. However, the same restrictions that enable the performance gains also make it difficult to express many of the important stages in a typical graph-analytics pipeline: constructing the graph, modifying its structure, or expressing computation that spans multiple graphs. As a consequence, existing graph analytics pipelines compose graph-parallel and data-parallel systems using external storage systems, leading to extensive data movement and complicated programming model.

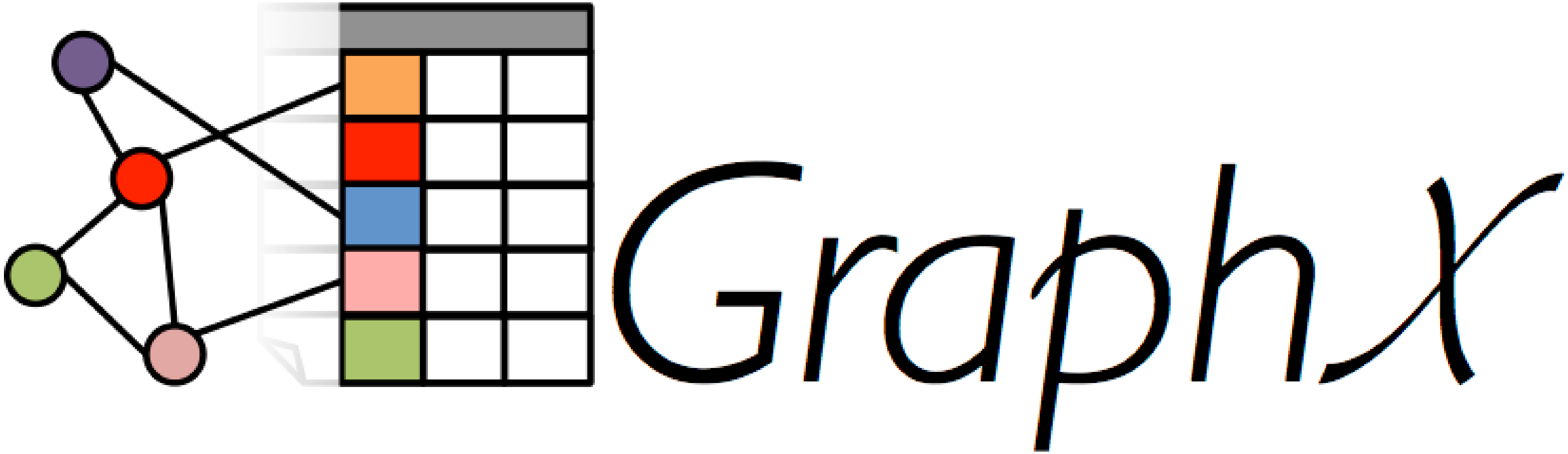
To address these challenges we introduce GraphX, a distributed graph computation framework that unifies graph-parallel and data-parallel computation. GraphX provides a small, core set of graph-parallel operators expressive enough to implement the Pregel and PowerGraph abstractions, yet simple enough to be cast in relational algebra. GraphX uses a collection of query optimization techniques such as automatic join rewrites to efficiently implement these graph-parallel operators. We evaluate GraphX on real-world graphs and workloads and demonstrate that GraphX achieves comparable performance as specialized graph computation systems, while outperforming them in end-to-end graph pipelines. Moreover, GraphX achieves a balance between expressiveness, performance, and ease of use.

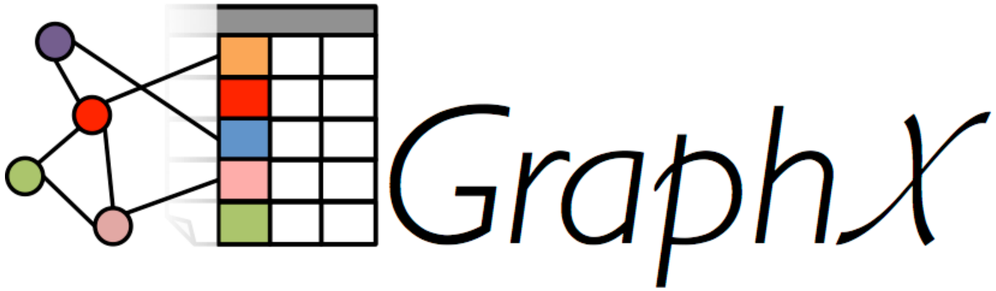
1. INTRODUCTION

Figure 1: **Graph Analytics Pipeline:** Graph analytics is the process of going from raw data, to a graph, to the relevant subgraph, applying graph algorithms, analyzing the result, and then potentially repeating the process with a different subgraph. Currently, these pipelines compose data-parallel and graph-parallel systems through a distributed file interface. The goal of the GraphX system is to unify the data-parallel and graph-parallel views of computation into a single system and to accelerate the entire pipeline.

(*e.g.,* PageRank and connected components). By leveraging the restricted abstraction in conjunction with the static graph structure, these systems are able to optimize the data layout and distribute the execution of complex iterative algorithms on graphs with tens of billions of vertices and edges.

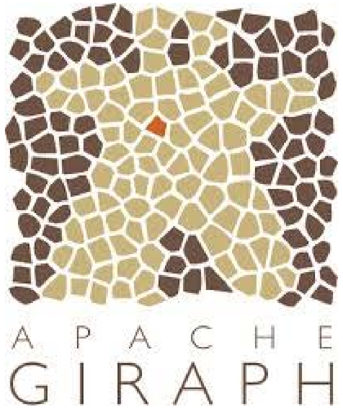
By restricting the types of computation they express to iterative vertex-centric algorithms on a single static graph, these *graph-parallel* systems are able to achieve orders-of-magnitude performance gains over contemporary data-parallel systems such as Hadoop MapReduce. However, these same restrictions make it difficult to express many of the operations found in a typical graph analytics pipeline (*e.g.,* Figure 1). These operations include


GraphX


GraphX

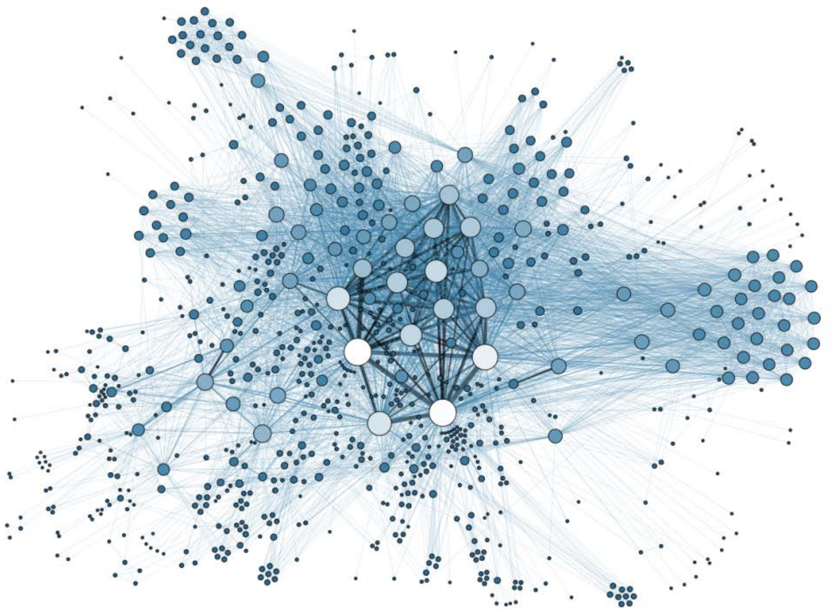
GraphLab

hadoop


APACHE
GIRAPH

GraphX

Google

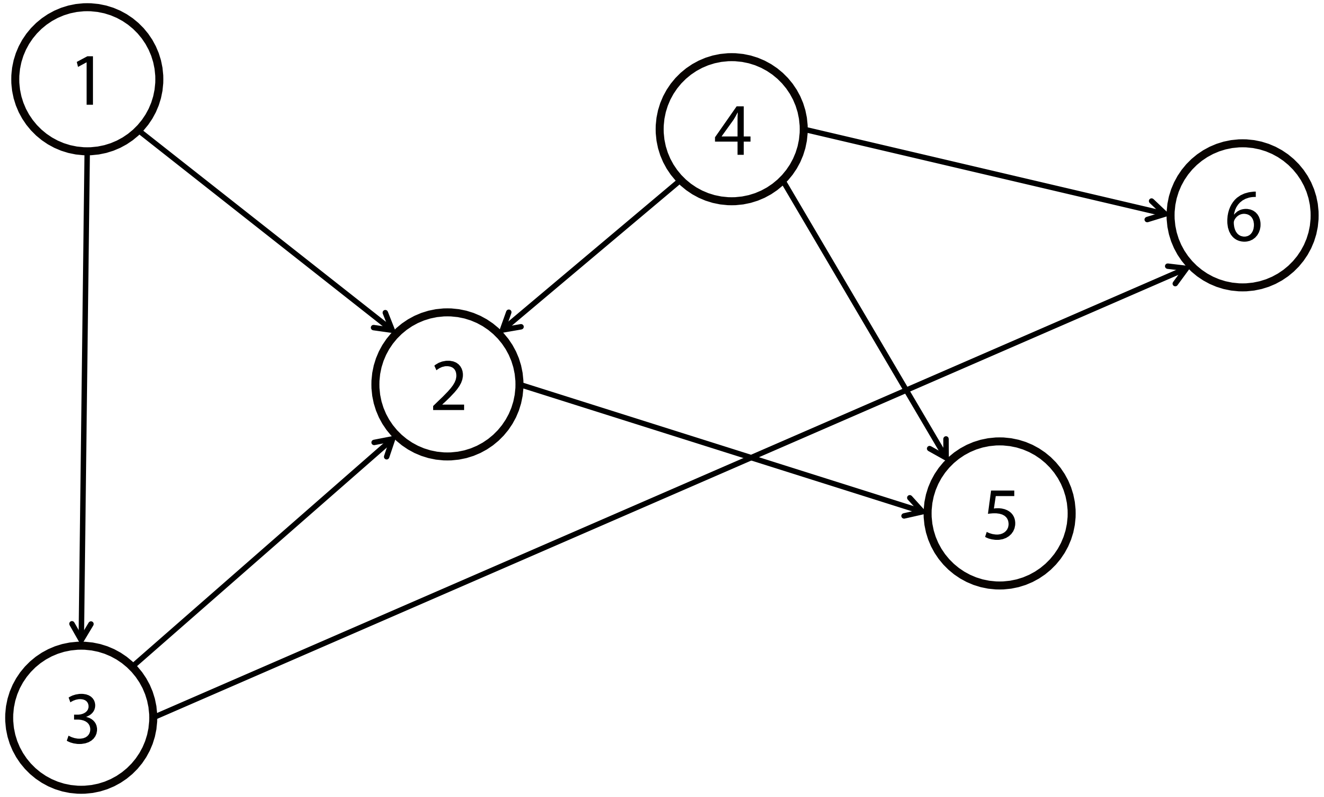


WIKIPEDIA
The Free Encyclopedia

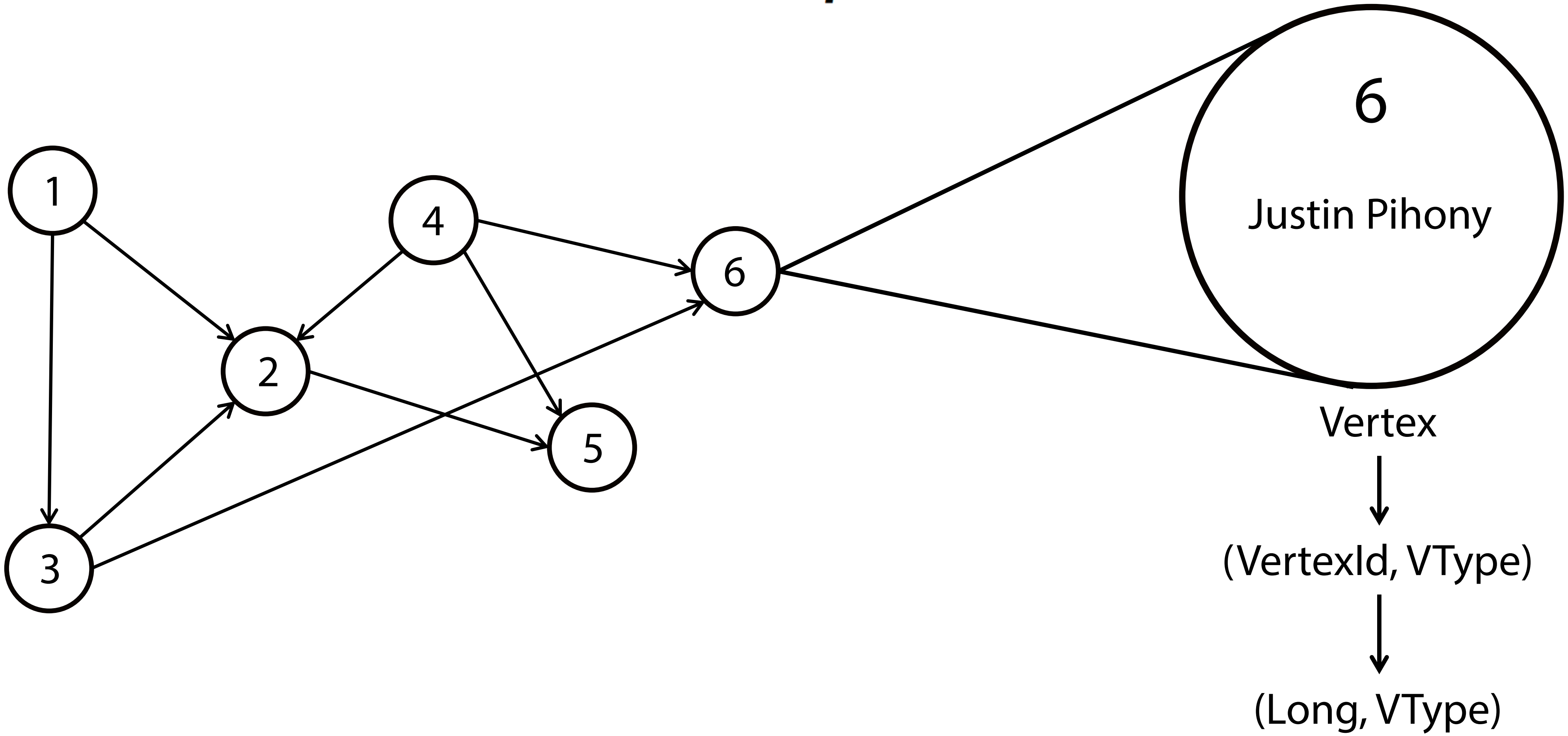
NETFLIX

amazon.com

GraphX



GraphX



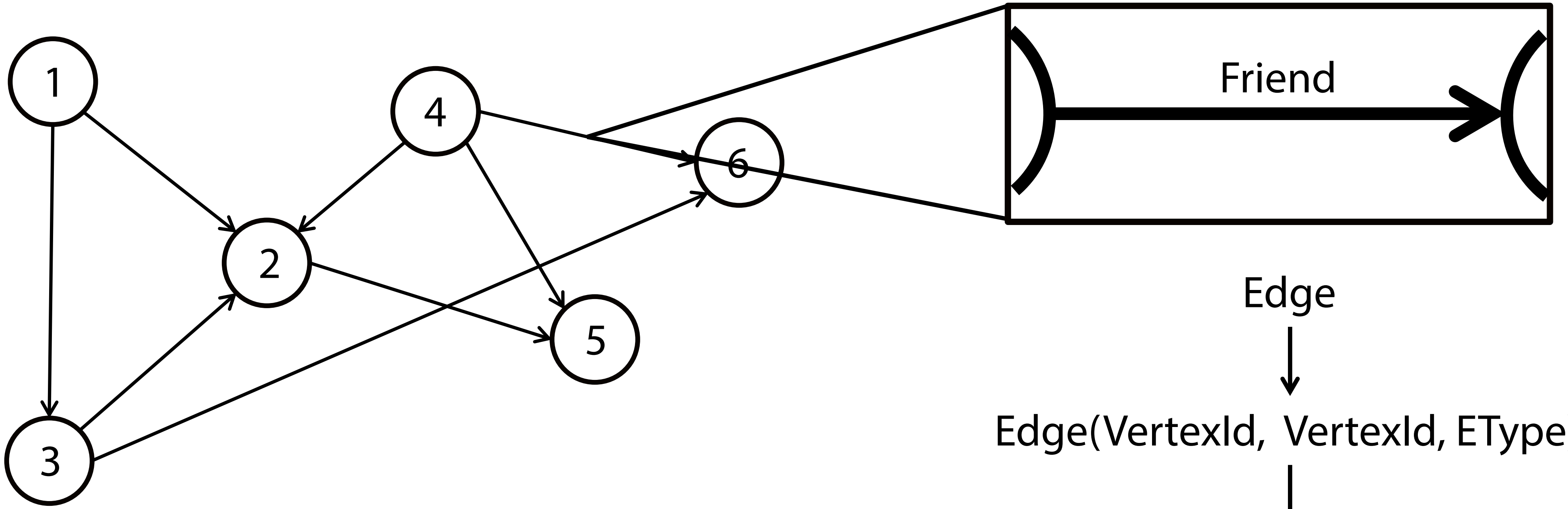
Justin Pihony

Vertex

(VertexId, VType)

(Long, VType)

GraphX



Edge

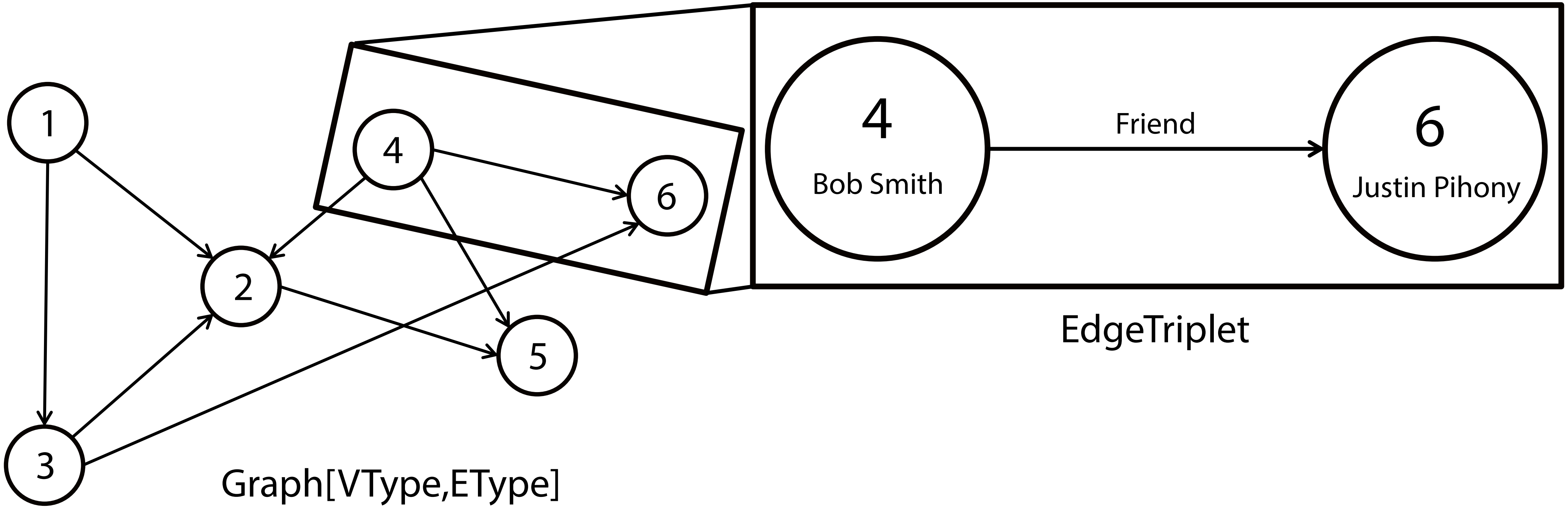
↓

Edge(VertexId, VertexId, EType)

↓

Edge(Long, Long, EType)

GraphX



Graph[VType,EType]

↓

Graph(RDD[Vertex], RDD[Edge])

Resources

- SQL
  - https://ogirardot.wordpress.com/2015/05/29/rdds-are-the-new-bytecode-of-apache-spark/
  - https://databricks.com/blog/2015/03/24/spark-sql-graduates-from-alpha-in-spark-1-3.html
  - Michael Armbrust: https://www.youtube.com/watch?v=xWkJCUcD55w
- Streaming
  - http://techblog.netflix.com/2015/03/can-spark-streaming-survive-chaos-monkey.html
  - https://databricks.com/blog/2015/07/30/diving-into-spark-streamings-execution-model.html
  - TD: https://www.youtube.com/watch?v=mKdm4NCtYgk
- MLlib
  - Ameet Talwalkar: https://www.youtube.com/watch?v=qSPqh7DiREM
- GraphX
  - https://amplab.cs.berkeley.edu/wp-content/uploads/2014/02/graphx.pdf
  - Joseph Gonzalez & Reynold Xin: https://www.youtube.com/watch?v=mKEn9C5bRck
- https://spark.apache.org/docs/latest/**[LIBRARY]**-programming-guide.html (mllib-guide)

Summary

- SQL
  - DataFrames
- Streaming
- MLlib
- GraphX