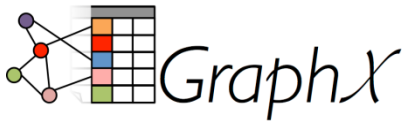
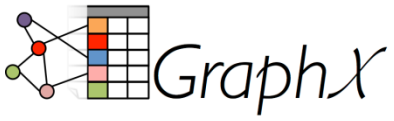
Introduction to Spark

#5  Graph Analytics on Spark

Srini Penchikala
srinipenchikala@gmail.com
08/06/14

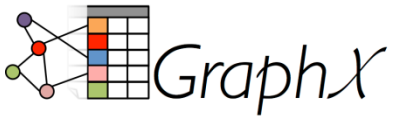
About Me

- Solutions Architect
- Lead Editor for NoSQL at InfoQ.com
- Started as a Java Developer
- Current Focus: NoSQL Databases
  - Cassandra
  - MongoDB
  - Neo4j
  - Hazelcast
- Not a Data Scientist



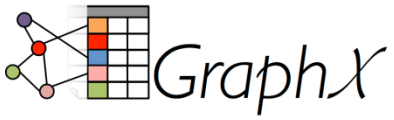
Sources for this presentation

- Graph Analytics with Graphx** (https://databricks-training.s3.amazonaws.com/graph-analytics-with-graphx.html)
- GraphX Programming Guide (http://spark.apache.org/docs/latest/graphx-programming-guide.html)
- Spark Summit 2014 Presentations
- AmpCamp 2014 Presentations



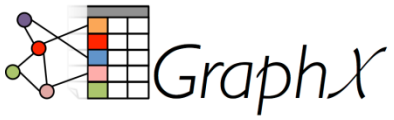
Software Used for Today's Demos

- Spark 1.0.1
- JDK 1.6 (http://www.oracle.com/technetwork/java/javase/downloads/index-jsp-138363.html)
- Scala IDE in Eclipse
- Windows



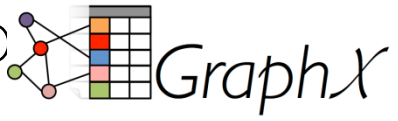
How to Install Spark

- Spark 1.0.1 (https://spark.apache.org/downloads.html)
- Cloudera VM (http://www.cloudera.com/content/support/en/downloads/quickstart_vms/cdh-5-1-x.html)
- Vagrant/VirtualBox/Chef
- Others?



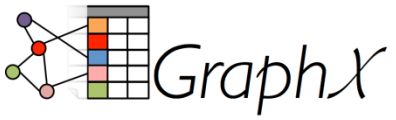
How to run Spark

- Four ways to run Spark
- Run Spark locally with one worker thread(no parallelism)
  - `spark-shell.cmd or spark-shell.cmd local`
- Run locally with K worker threads (ideally set to # of cores)
  - `spark-shell.cmd local[2]`
- Connect to a Spark standalone cluster
  - `spark-shell.cmd spark://HOST:PORT`
  - Example: `MASTER=spark://localhost:7077`
- Connect to a Mesos cluster
  - `spark-shell.cmd mesos://HOST:PC`



Agenda

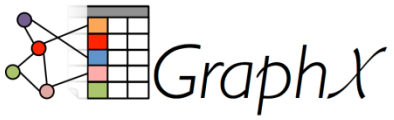
- Spark
- GraphX
- Use Cases
- Demos
- Q&A



Spark

Spark Background

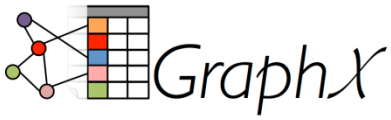
- Distributed Systems are the new standard (specifically, Data Parallel systems)
- Spark: An SDK for Big Data Applications
- Unified System With Libraries to Build a Complete Solution
- Full-featured Programming Environment
- Single, Consistent Interface for Developers to Write Against
- Runtimes available on several platforms

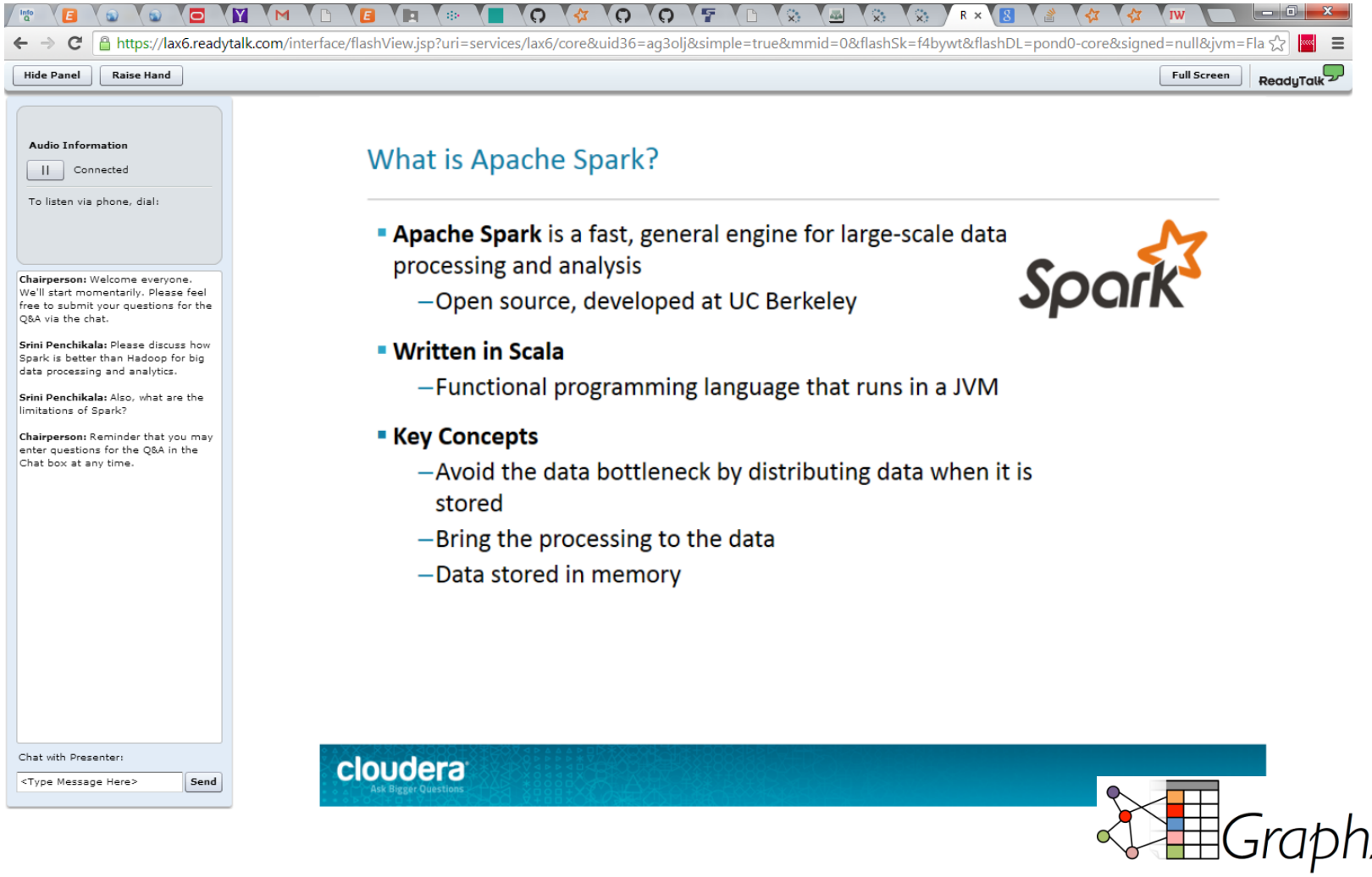


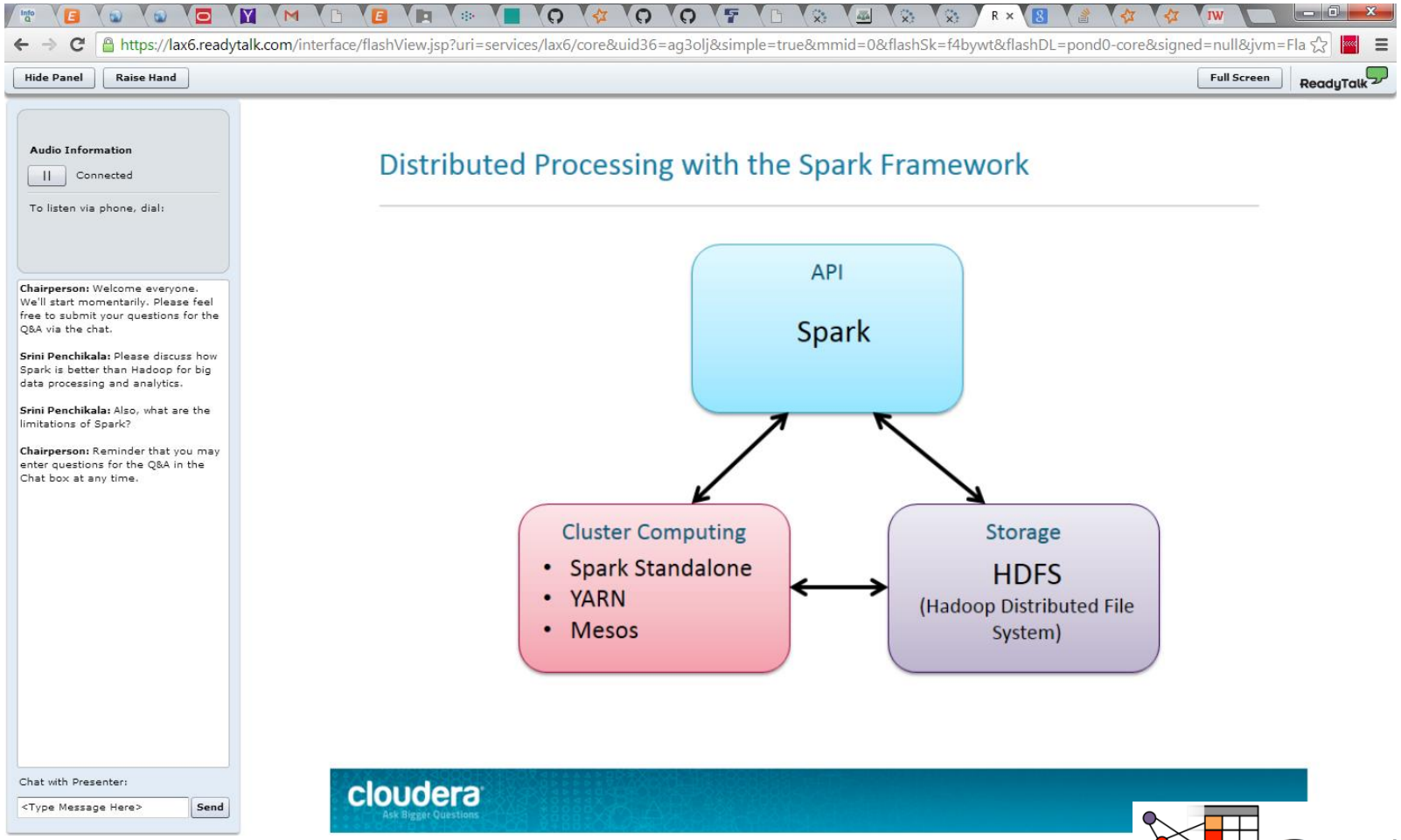
A Brief History: *Spark*

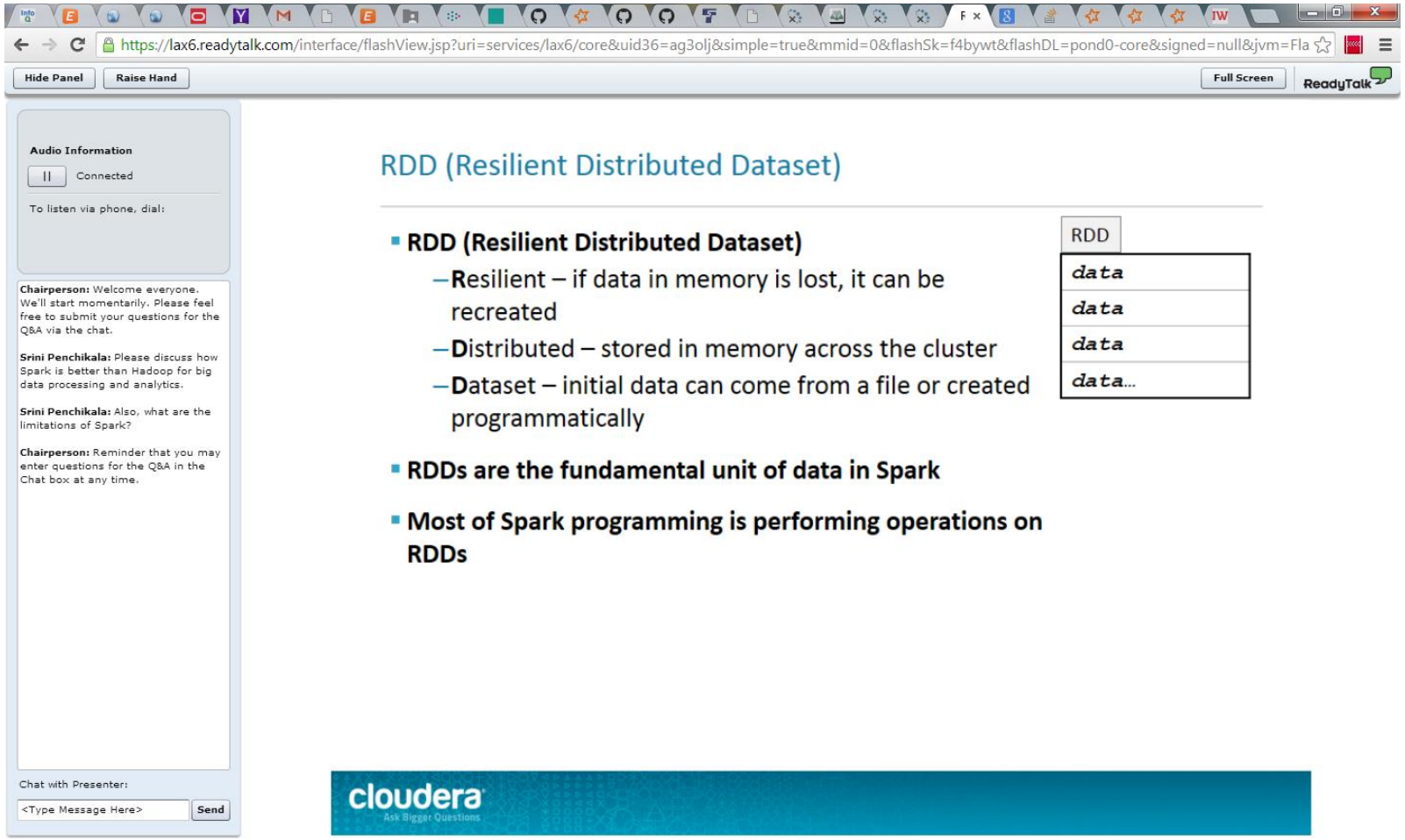
Some key points about Spark:

- handles batch, interactive, and real-time within a single framework
- native integration with Java, Python, Scala
- programming at a higher level of abstraction
- more general: map/reduce is just one set of supported constructs




https://lax6.readytalk.com/interface/flashView.jsp?uri=services/lax6/core&uid36=ag3olj&simple=true&mmid=0&flashSk=f4bywt&flashDL=pond0-core&signed=null&jvm=Fla
Hide Panel
Raise Hand
Full Screen
ReadyTalk
Audio Information
Connected
To listen via phone, dial:
Chairperson: Welcome everyone. We'll start momentarily. Please feel free to submit your questions for the Q&A via the chat.
Srini Penchikala: Please discuss how Spark is better than Hadoop for big data processing and analytics.
Srini Penchikala: Also, what are the limitations of Spark?
Chairperson: Reminder that you may enter questions for the Q&A in the Chat box at any time.
Chat with Presenter:
<Type Message Here>
Send
What is Apache Spark?
Apache Spark is a fast, general engine for large-scale data processing and analysis
Open source, developed at UC Berkeley
Written in Scala
Functional programming language that runs in a JVM
Key Concepts
Avoid the data bottleneck by distributing data when it is stored
Bring the processing to the data
Data stored in memory
Spark
cloudera
Ask Bigger Questions
GraphX


https://lax6.readytalk.com/interface/flashView.jsp?uri=services/lax6/core&uid36=ag3olj&simple=true&mmid=0&flashSk=f4bywt&flashDL=pond0-core&signed=null&jvm=Fla
Hide Panel
Raise Hand
Full Screen
ReadyTalk
Audio Information
Connected
To listen via phone, dial:
Chairperson: Welcome everyone. We'll start momentarily. Please feel free to submit your questions for the Q&A via the chat.
Srini Penchikala: Please discuss how Spark is better than Hadoop for big data processing and analytics.
Srini Penchikala: Also, what are the limitations of Spark?
Chairperson: Reminder that you may enter questions for the Q&A in the Chat box at any time.
Chat with Presenter:
<Type Message Here>
Send
Distributed Processing with the Spark Framework
API
Spark
Cluster Computing
Spark Standalone
YARN
Mesos
Storage
HDFS
(Hadoop Distributed File System)
cloudera
Ask Bigger Questions
GraphX


https://lax6.readytalk.com/interface/flashView.jsp?uri=services/lax6/core&uid36=ag3olj&simple=true&mmid=0&flashSk=f4bywt&flashDL=pond0-core&signed=null&jvm=Fla
Hide Panel
Raise Hand
Full Screen
ReadyTalk
Audio Information
Connected
To listen via phone, dial:
Chairperson: Welcome everyone. We'll start momentarily. Please feel free to submit your questions for the Q&A via the chat.
Srini Penchikala: Please discuss how Spark is better than Hadoop for big data processing and analytics.
Srini Penchikala: Also, what are the limitations of Spark?
Chairperson: Reminder that you may enter questions for the Q&A in the Chat box at any time.
Chat with Presenter:
<Type Message Here>
Send
RDD (Resilient Distributed Dataset)
RDD (Resilient Distributed Dataset)
–Resilient – if data in memory is lost, it can be recreated
–Distributed – stored in memory across the cluster
–Dataset – initial data can come from a file or created programmatically
RDDs are the fundamental unit of data in Spark
Most of Spark programming is performing operations on RDDs
RDD
data
data
data
data...
cloudera
Ask Bigger Questions

Spark Execution Model

1. Create DAG of RDDs to represent computation
2. Create logical execution plan for DAG
3. Schedule and execute individual tasks

From Challenges to Solutions

Challenges	Solutions
Clusters hard to set up and manage	Hosted platform
Need to integrate a zoo of tools	Apache Spark
Tools are hard to use	Interactive Workspace

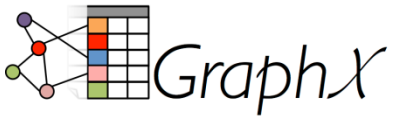
GraphX

What is GraphX

- New Spark API for:
  - Graphs (e.g., Web-Graphs and Social Networks)
  - Graph-parallel computation (e.g., PageRank and Collaborative Filtering)
- Extends Spark RDD abstraction by introducing Resilient Distributed Property Graph
- Property Graph: a directed multi-graph with properties attached to each vertex and edge
- Set of operators (e.g., subgraph, joinVertices, & mapReduceTriplets)
- Collection of graph algorithms
- Builders to simplify graph analytics tasks

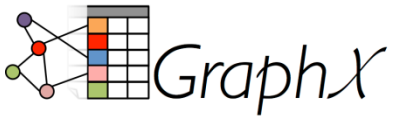
What is GraphX - 2

- Currently only available in Scala
- Java and Python bindings (future)
- ToDo: Link to GraphX API



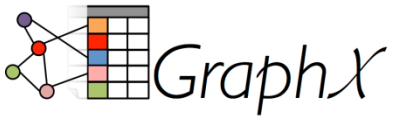
Property Graphs

- Like RDDs, property graphs are immutable, distributed, and fault-tolerant.

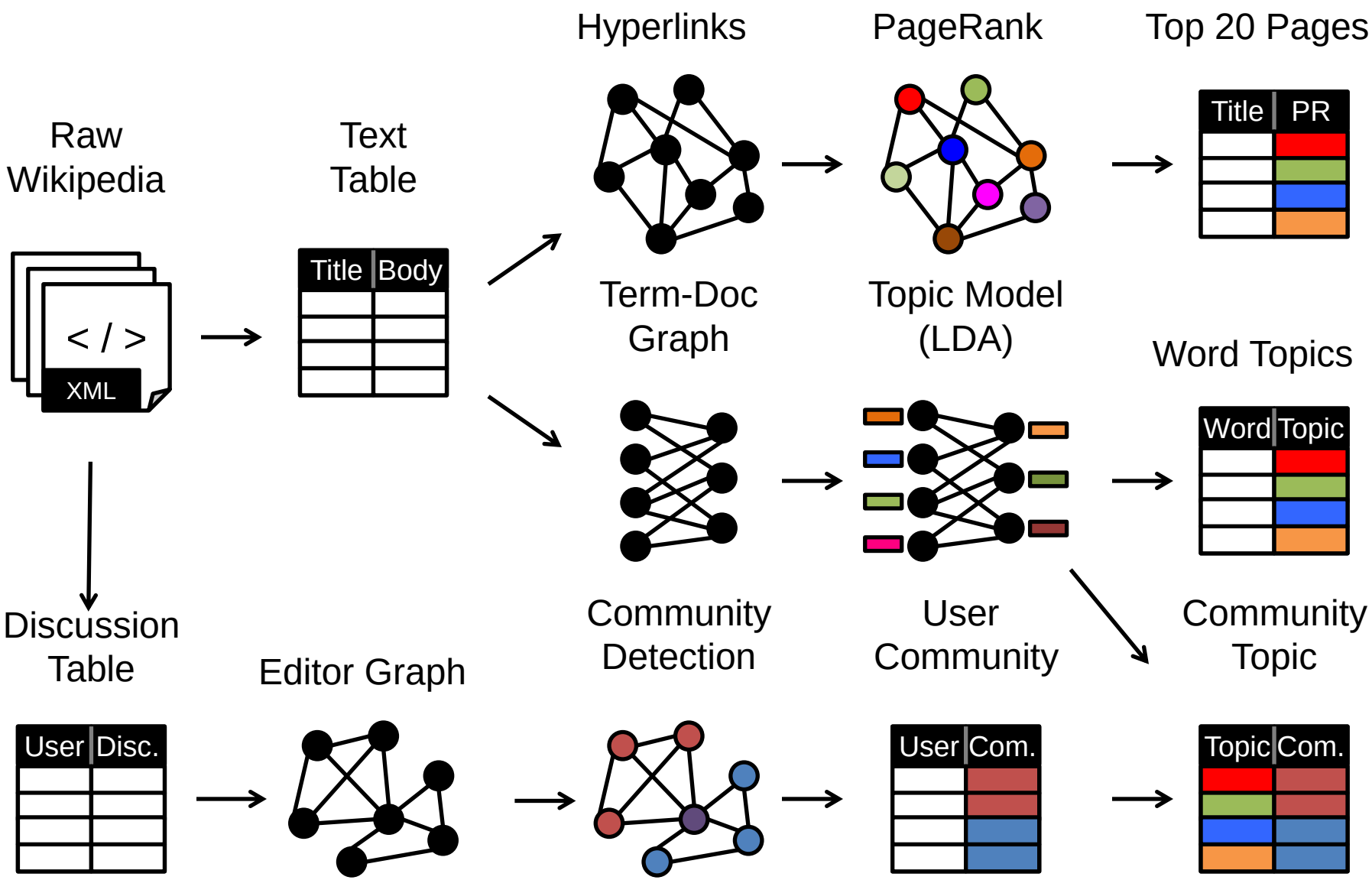


Graph Algorithms

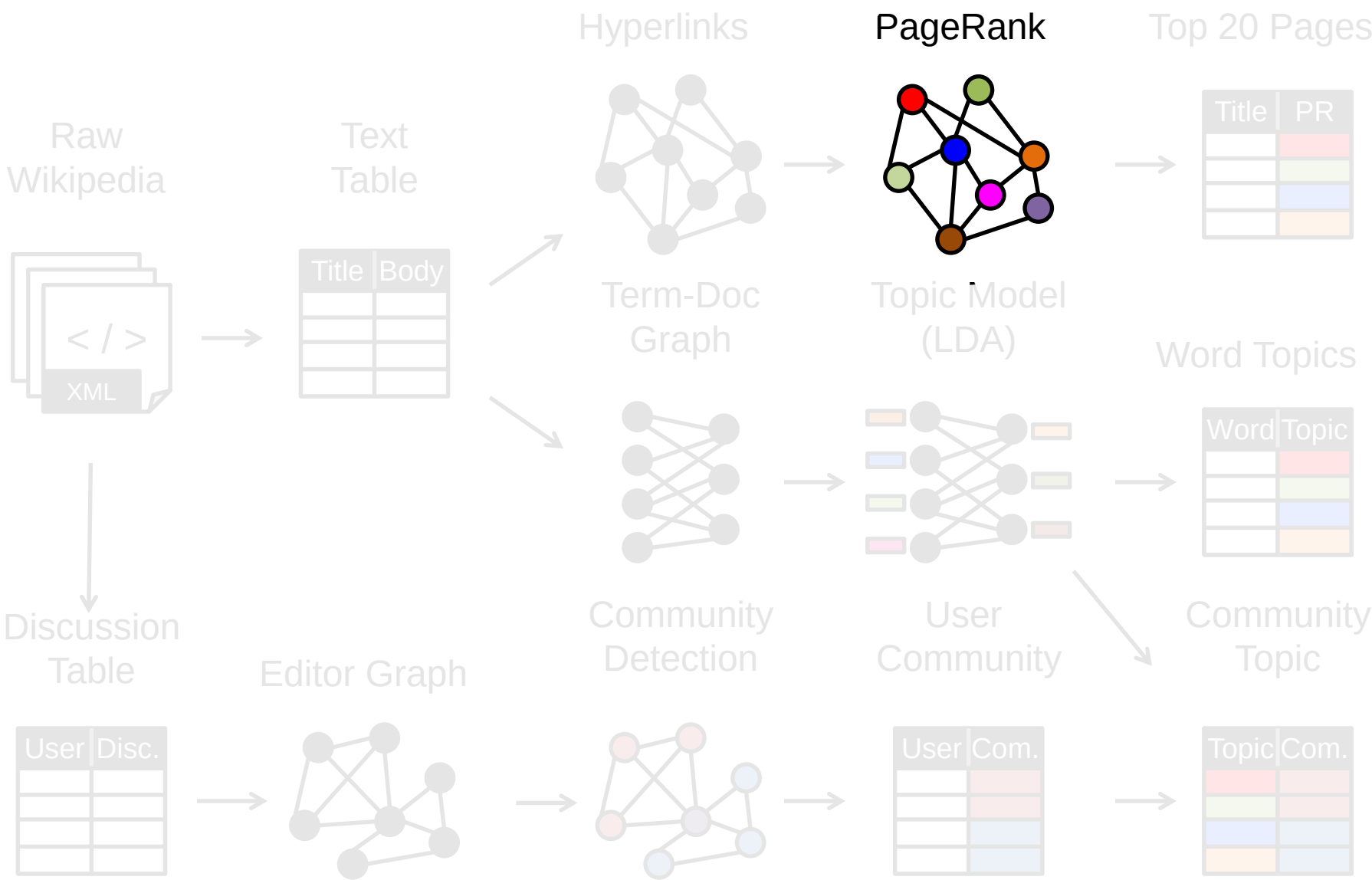
- PageRank
- Connected Components
- Triangle Counting Example



Graphs are Central to Analytics

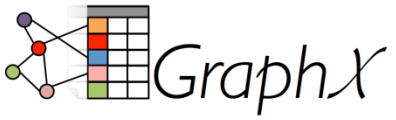


Graphs are Central to Analytics

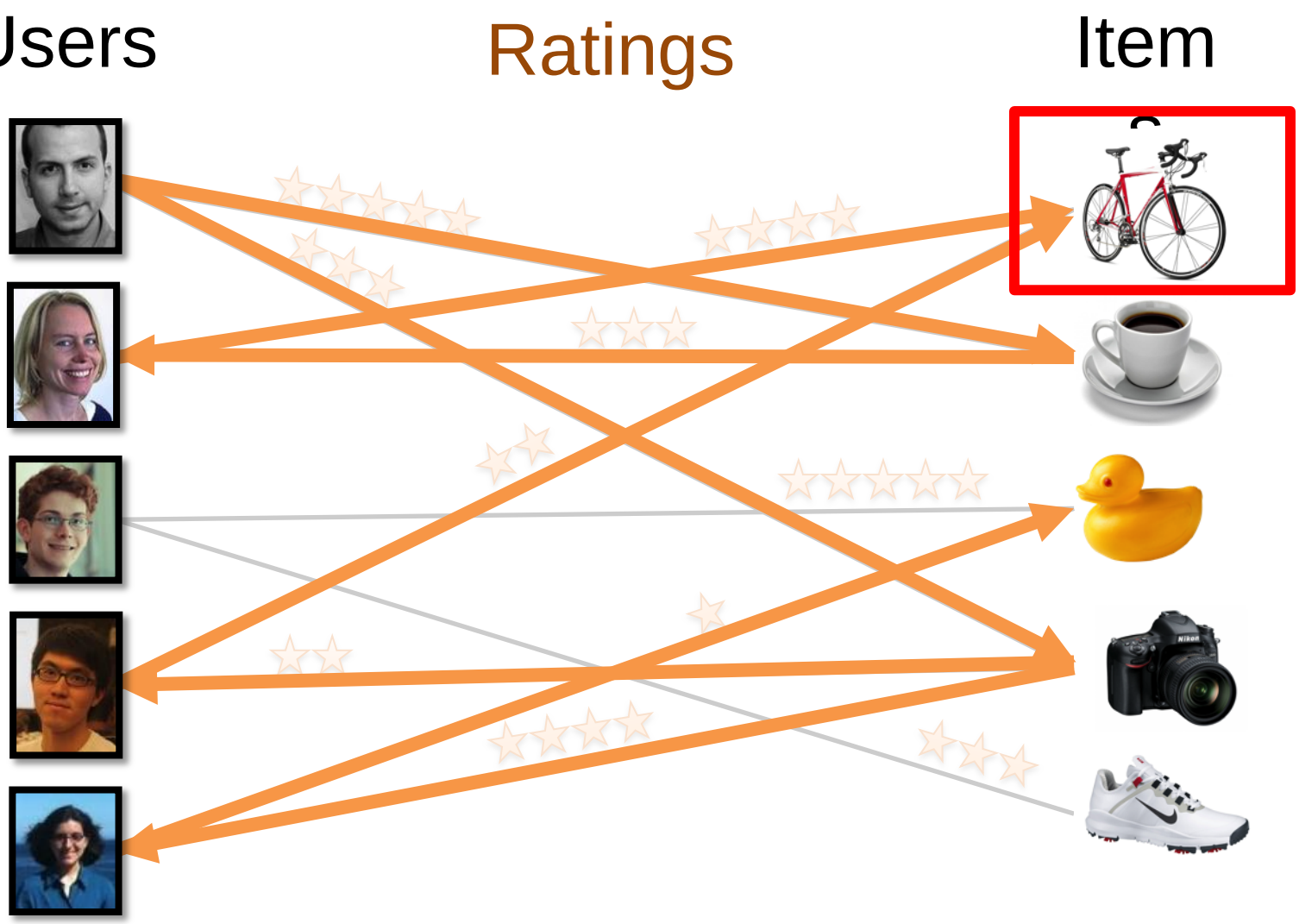


Graphs are Essential to Data Mining and Machine Learning

- Identify influential people and information
- Find communities
- Understand people's shared interests
- Model complex data dependencies

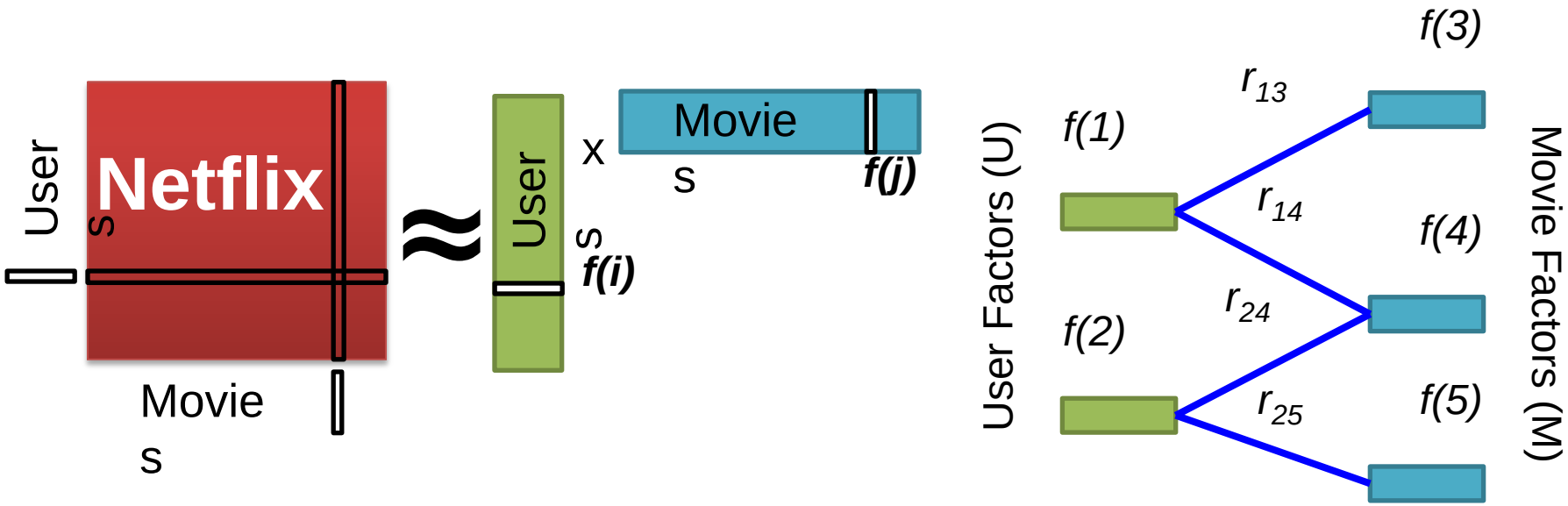


Recommending Products



Recommending Products

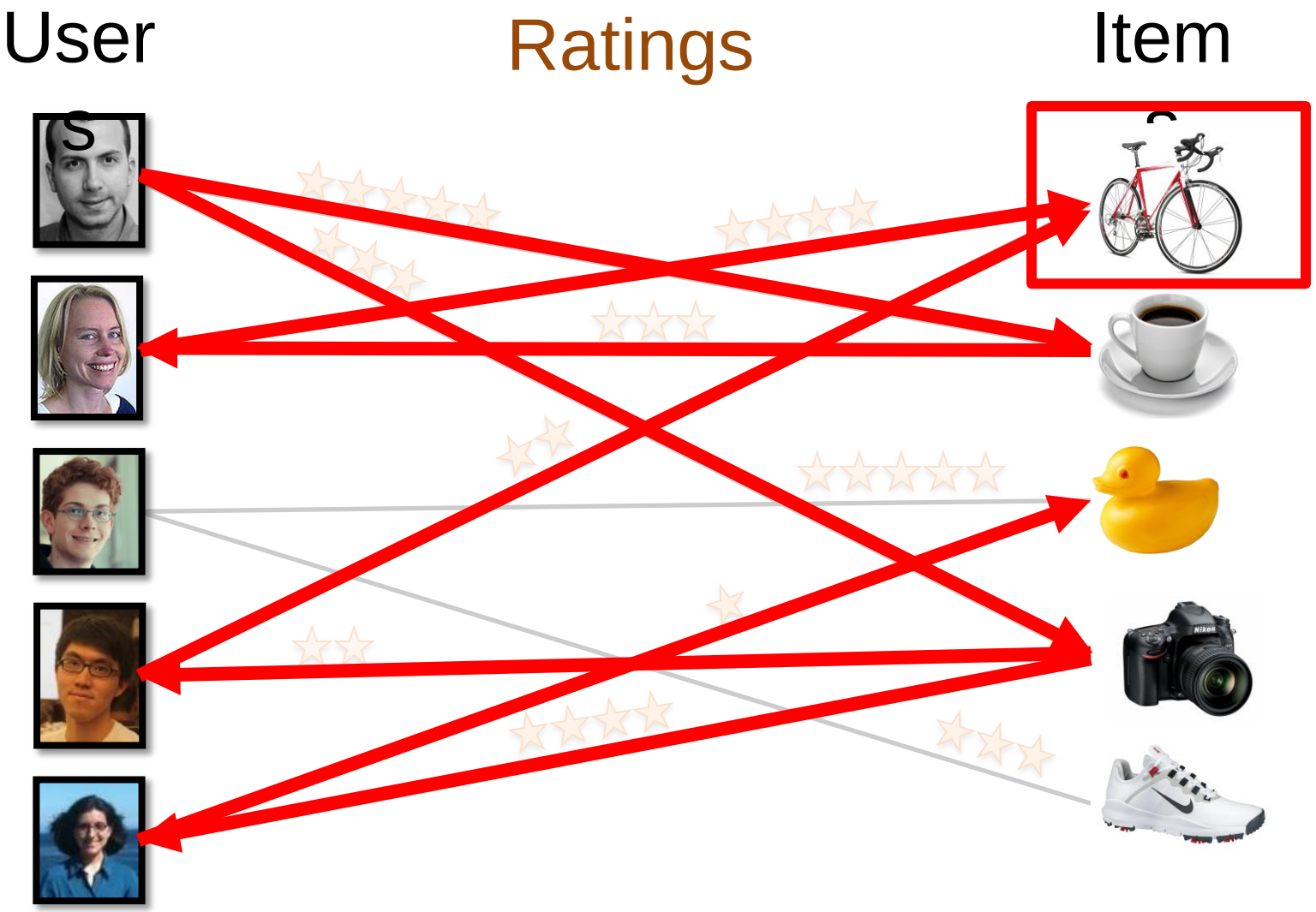
Low-Rank Matrix Factorization:



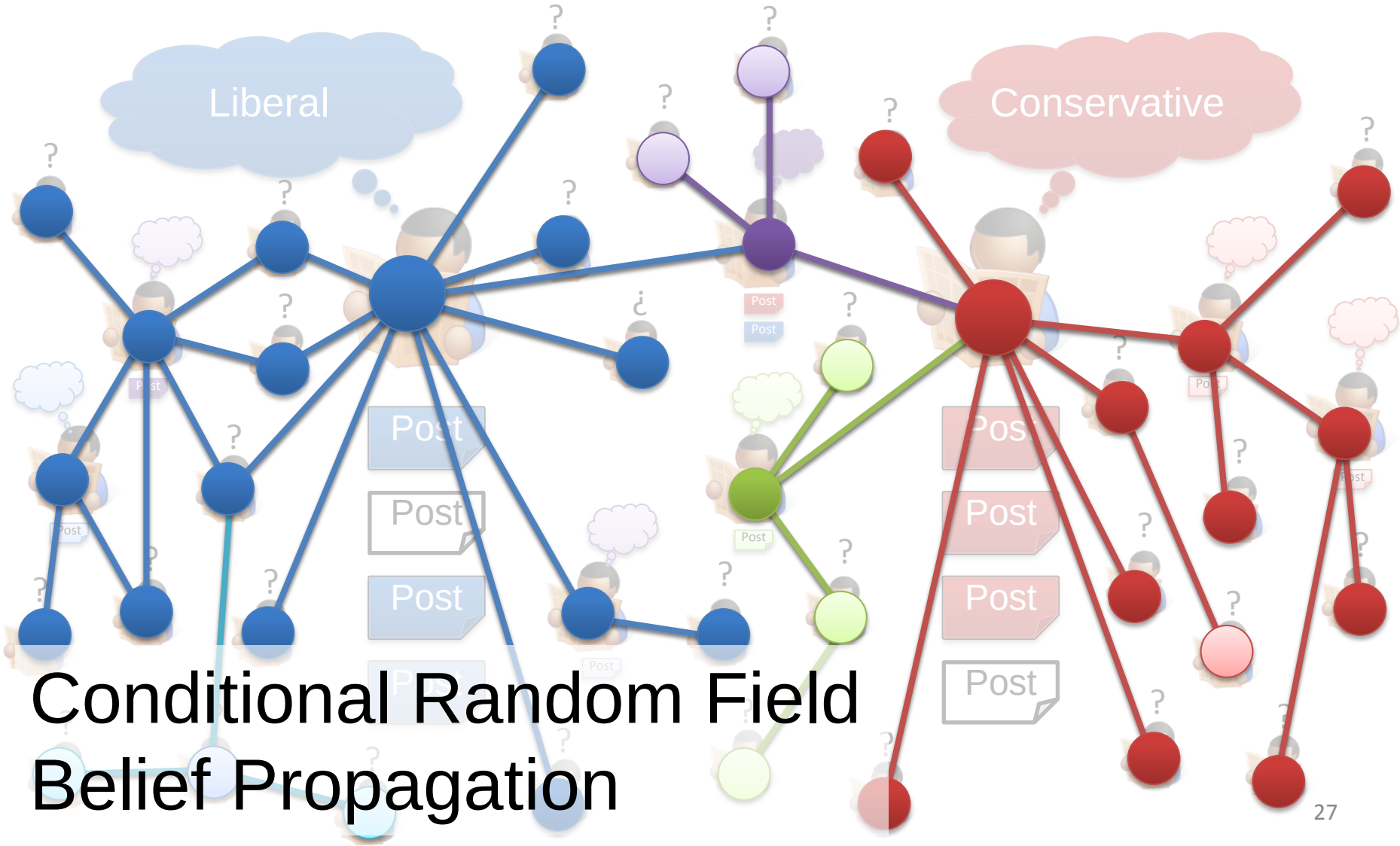
Iterate:

$$f[i] = \arg\min_{w \in \mathbb{R}^d} \sum_{j \in \mathrm{Nbrs}(i)} \left(r_{ij} - w^T f[j]\right)^2 + \lambda ||w||_2^2$$

Collaborative Filtering



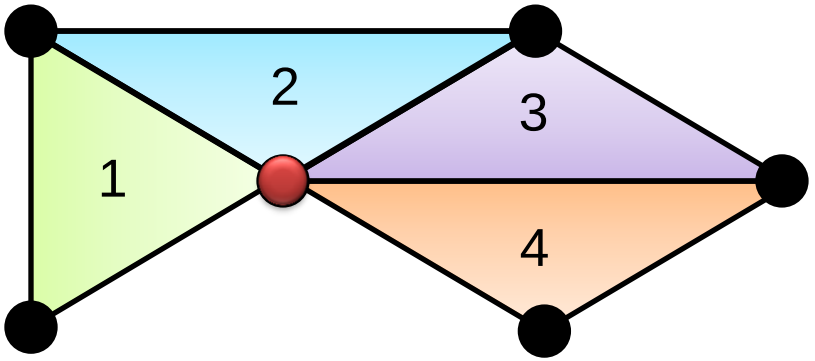
Predicting User Behavior



Conditional Random Field
Belief Propagation

Finding Communities

- Count triangles passing through each vertex:
-



- Measures “cohesiveness” of local community

Fewer Triangles
Weaker Community

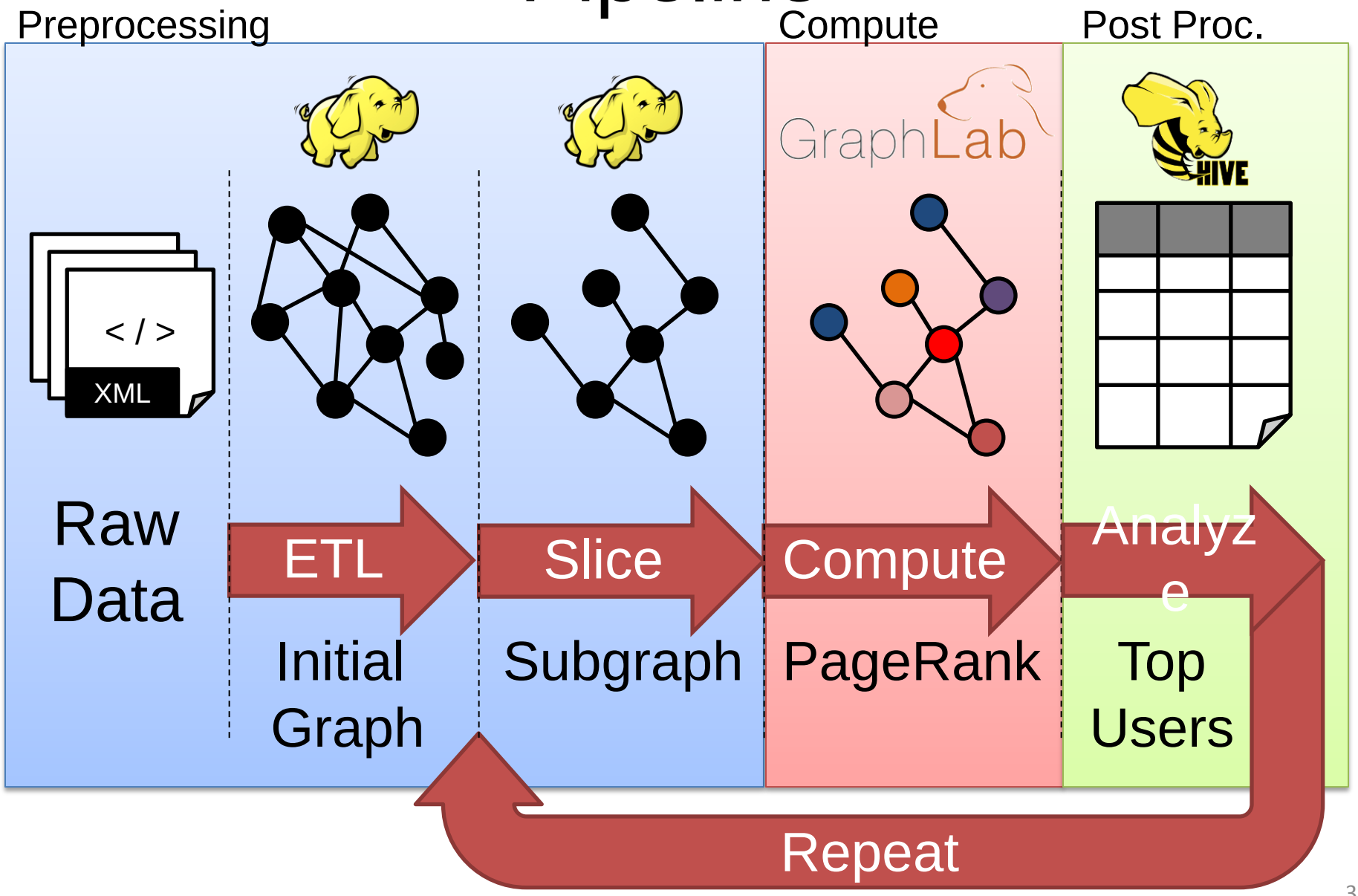
More Triangles
Stronger Community

Many More Graph Algorithms

- **Collaborative Filtering**
  - Alternating Least Squares
  - Stochastic Gradient Descent
  - Tensor Factorization
  - SVD
- **Structured Prediction**
  - Loopy Belief Propagation
  - Max-Product Linear Programs
  - Gibbs Sampling
- **Semi-supervised ML**
  - Graph SSL
  - CoEM
- **Graph Analytics**
  - PageRank
  - Single Source Shortest Path
  - Triangle-Counting
  - Graph Coloring
  - K-core Decomposition
  - Personalized PageRank
- **Classification**
  - Neural Networks
  - Lasso

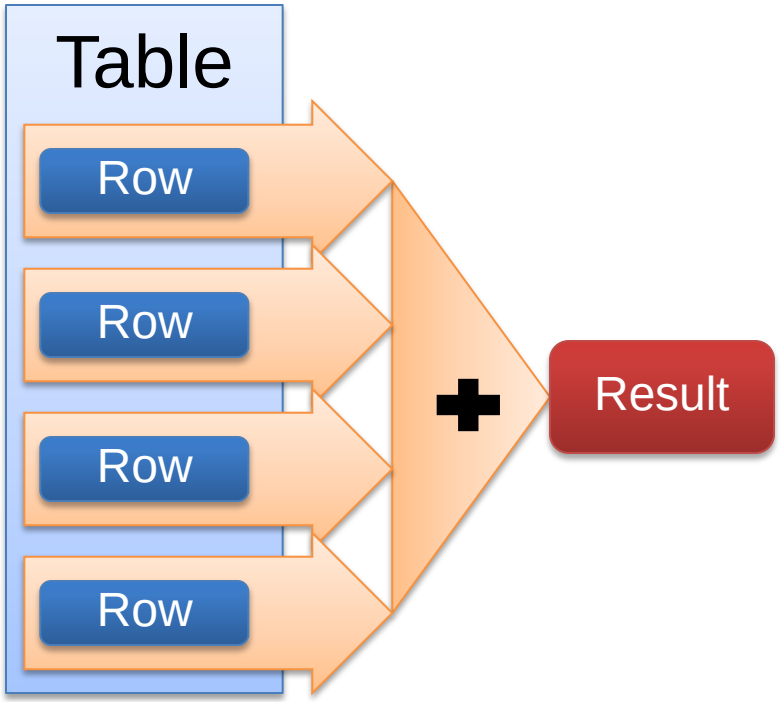
…

Example Graph Analytics Pipeline

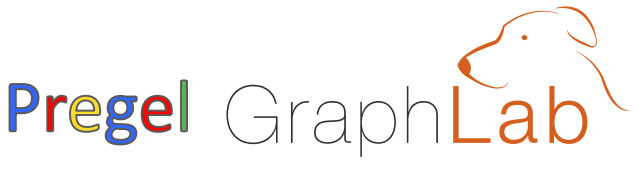


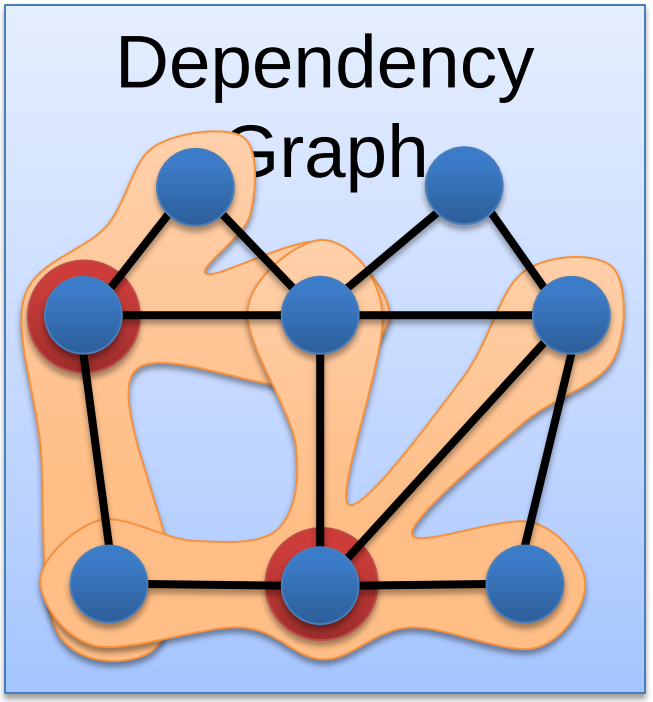
Separate Systems to Support Each View

Table View

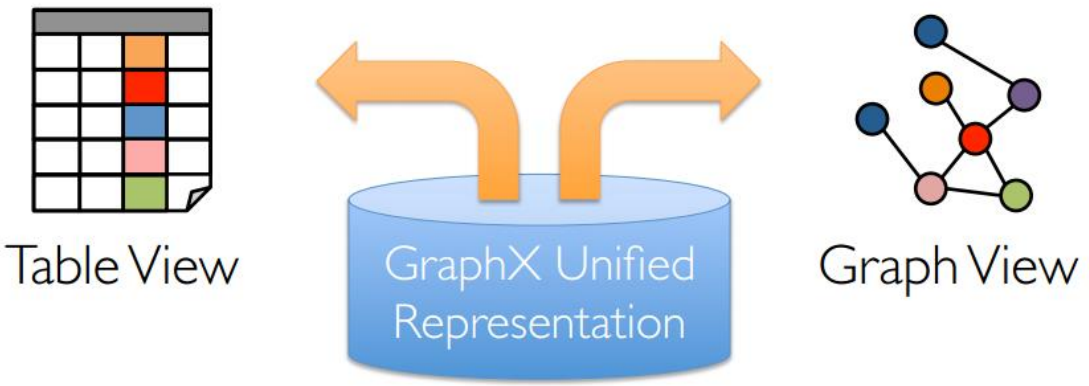


Graph View





GraphX – Adding Graphs to the Mix



Tables and Graphs are composable *views* of the *same physical* data

Each view has its own operators that exploit the semantics of the view to achieve efficient execution

R. Xin, J. Gonzalez, M. Franklin, I. Stoica, "GraphX: In-situ Graph Computation Made Easy" GRADES Workshop at SIGMOD, June 2013.

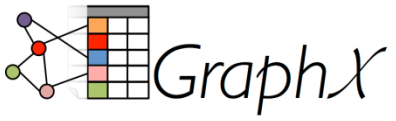
GraphX: Graphs → Dataflow

1. Encode graphs as distributed tables
2. Express graph computation in relational ops.
3. Recast graph systems optimizations as:
   1. Distributed join optimization
   2. Incremental materialized maintenance

Integrate Graph and Table data processing systems.

Achieve performance parity with specialized systems.

Having separate systems for each view is difficult to use and inefficient



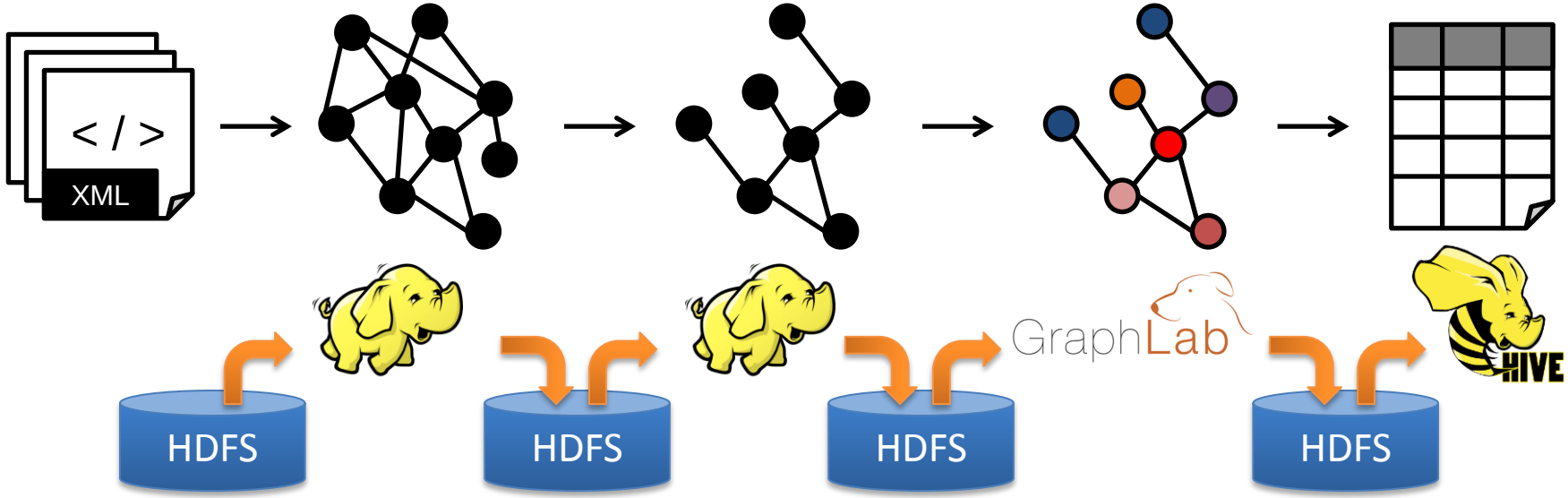
Difficult to Program and Use

Users must ***Learn***, ***Deploy***, and ***Manage*** multiple systems

Leads to brittle and often complex interfaces

Inefficient

Extensive data movement and duplication across the network and file system

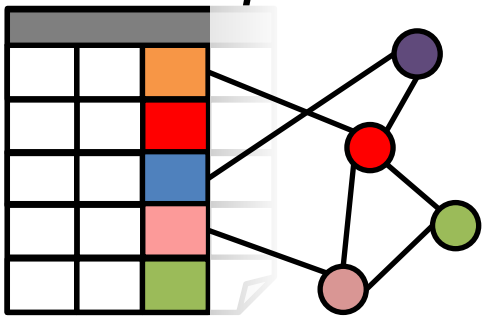


Limited reuse internal data-structures across stages

Solution: The GraphX Unified Approach

New API

Blurs the distinction between Tables and Graphs



New System

Combines Data-Parallel Graph-Parallel Systems

Enabling users to easily and efficiently express the entire graph analytics pipeline

Tables and Graphs are composable *views* of the *same physical* data

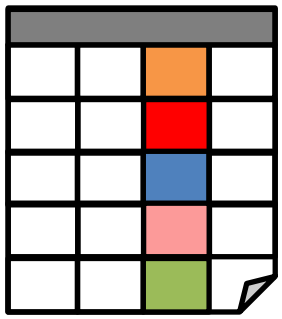
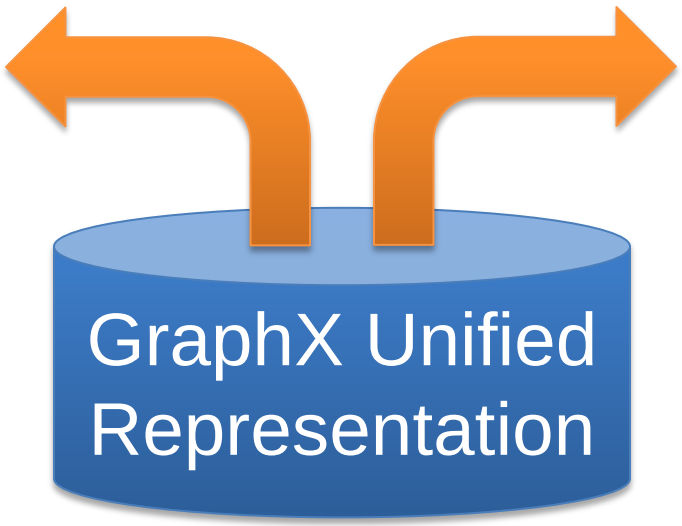
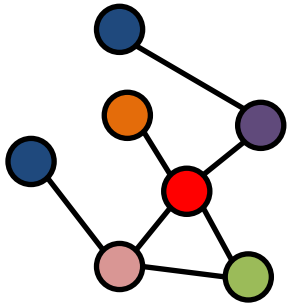


Table View



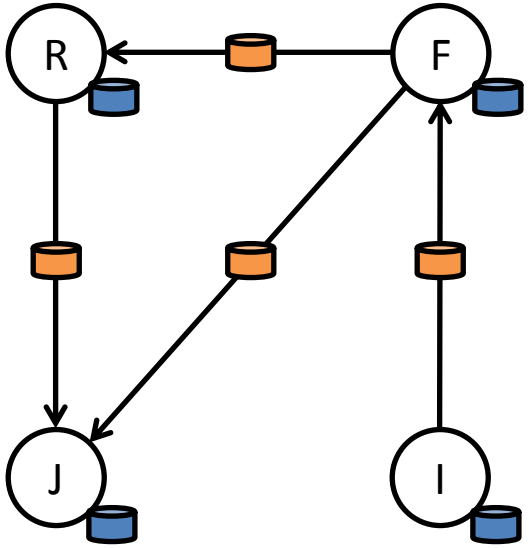


Graph View

Each view has its own operators that exploit the semantics of the view to achieve efficient execution

View a Graph as a Table

Property Graph



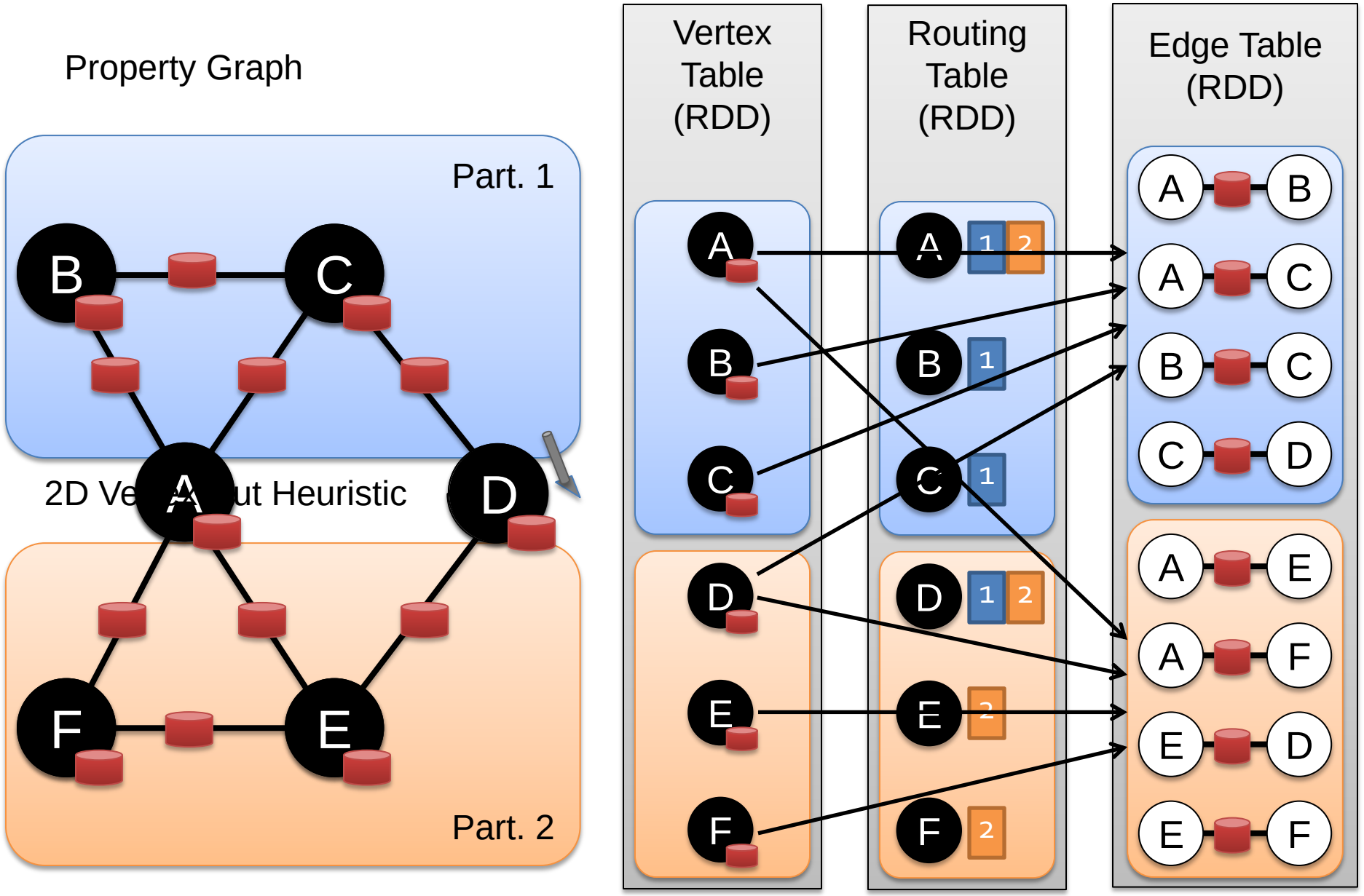
Vertex Property Table

Id	Property (V)
Rxin	(Stu., Berk.)
Jegonzal	(PstDoc, Berk.)
Franklin	(Prof., Berk)
Istoica	(Prof., Berk)

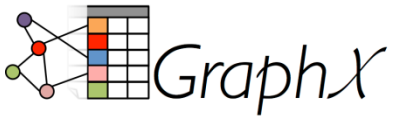
Edge Property Table

SrcId	DstId	Property (E)
rxin	jegonzal	Friend
franklin	rxin	Advisor
istoica	franklin	Coworker
franklin	jegonzal	PI

Distributed Graphs as Tables (RDDs)



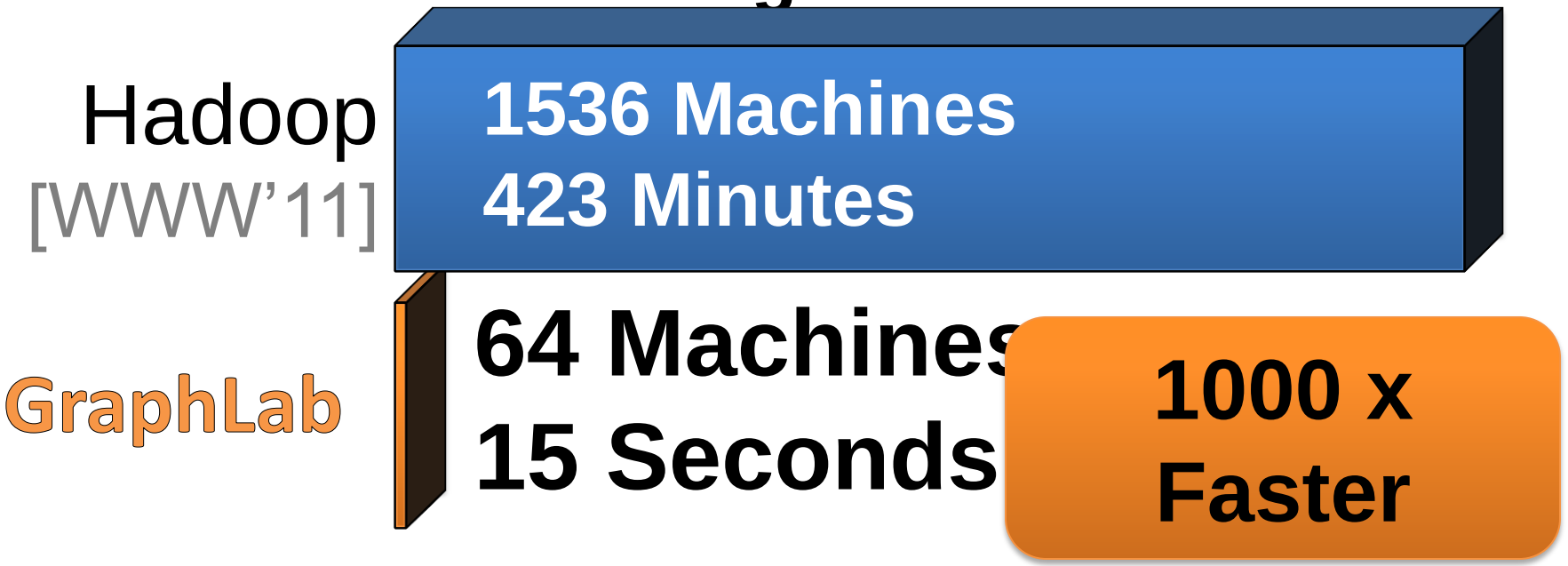
By exploiting ***graph-structure*** *Graph-Parallel systems can be* ***orders-of-magnitude faster****.*



Triangle Counting on Twitter

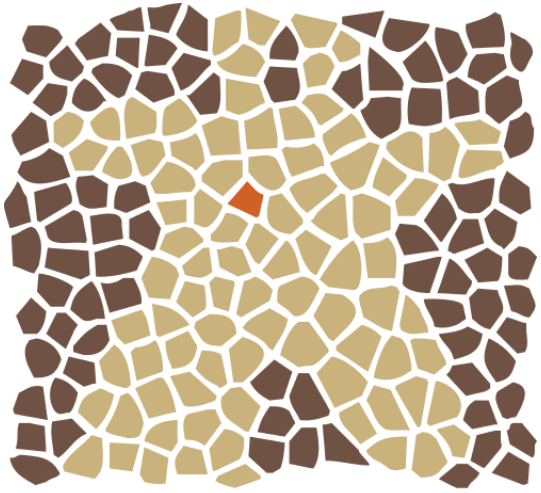
40M Users, 1.4 Billion Links

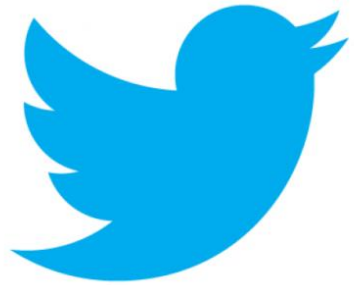
Counted: 34.8 Billion Triangles



S. Suri and S. Vassilvitskii, "Counting triangles and the curse of the last reducer," WWW'11

Specialized Graph Systems



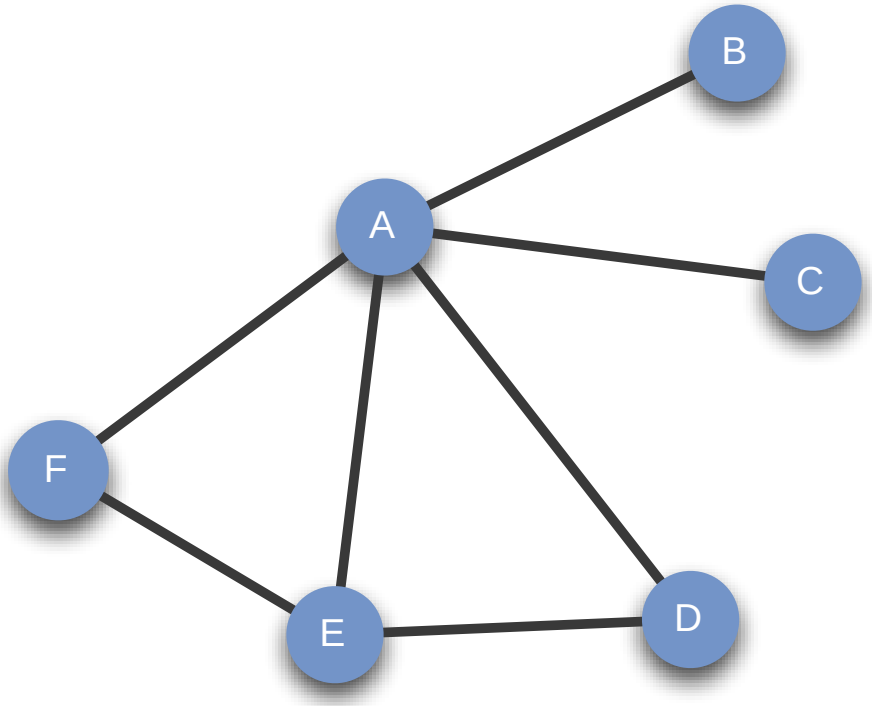


Pregel

Specialized Graph Systems

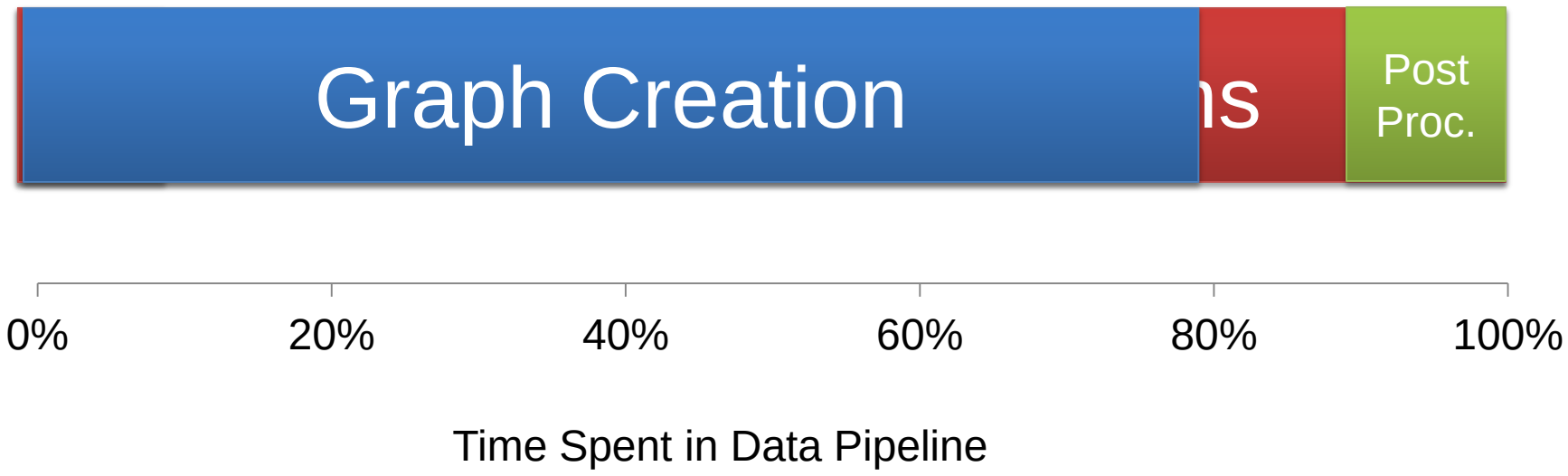
1. APIs to capture complex graph dependencies

2. Exploit graph structure to reduce communication and computation



Why GraphX?

The Bigger Picture



GraphX Unifies Data-Parallel and Graph-Parallel Systems

Spark

Table API

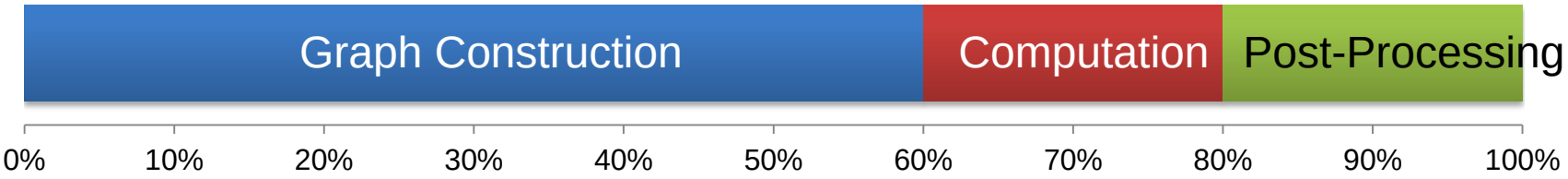
RDDs, Fault-tolerance, and task scheduling

GraphLab

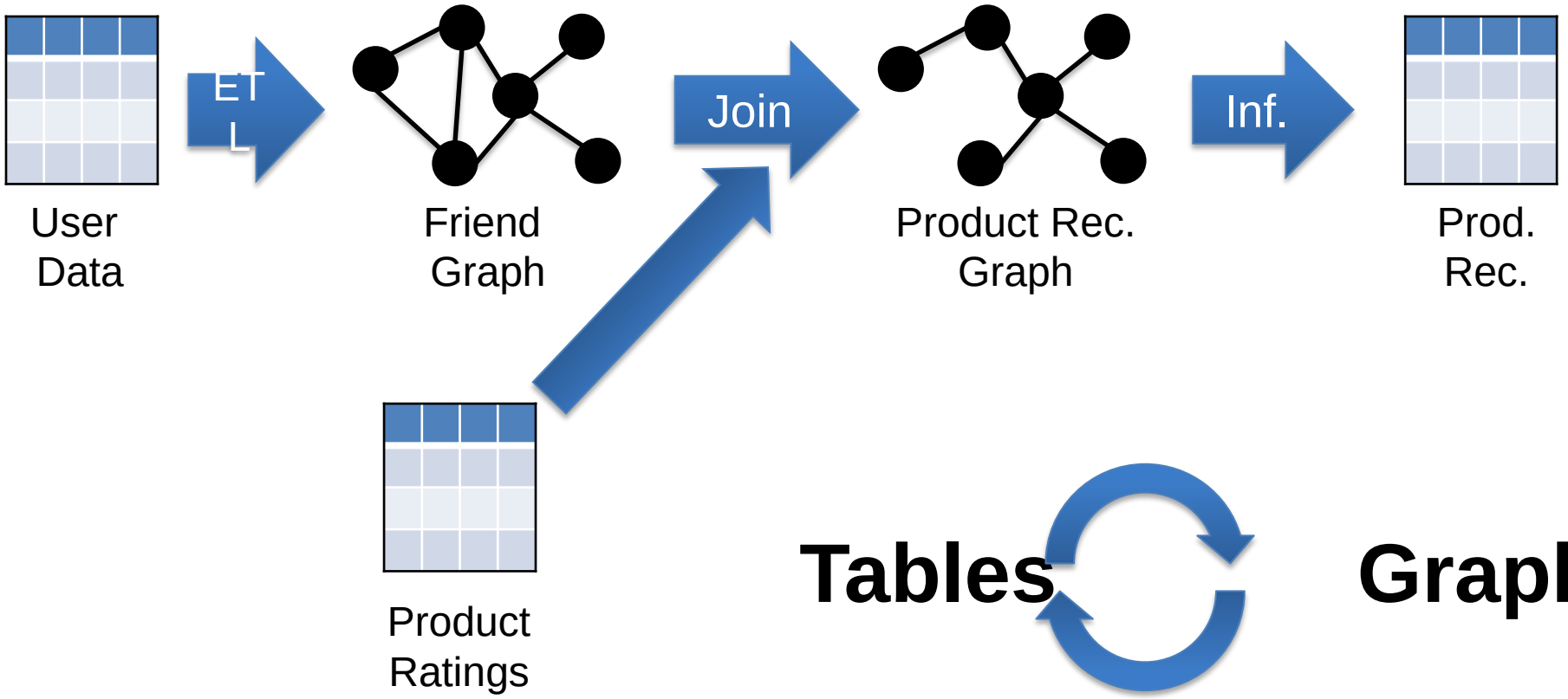
Graph API

graph representation and execution

one system for the entire graph pipeline

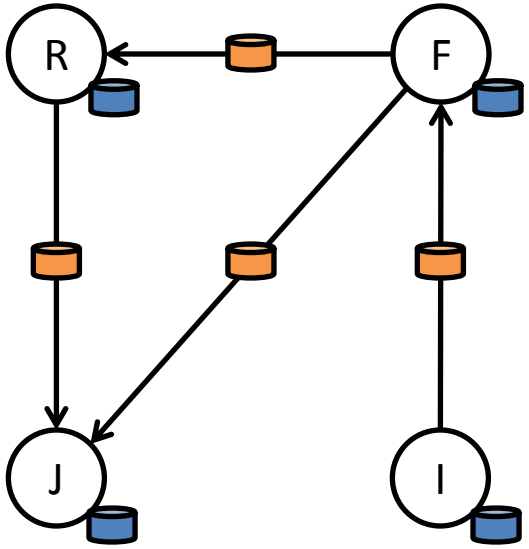


Enable Joining Tables and Graphs



The GraphX
Resilient Distributed Graph

Id	Attribute (V)
Rxin	(Stu., Berk.)
Jegonzal	(PstDoc, Berk.)
Franklin	(Prof., Berk)
Istoica	(Prof., Berk)



SrcId	DstId	Attribute (E)
rxin	jegonzal	Friend
franklin	rxin	Advisor
istoica	franklin	Coworker
franklin	jegonzal	PI

Table Operators

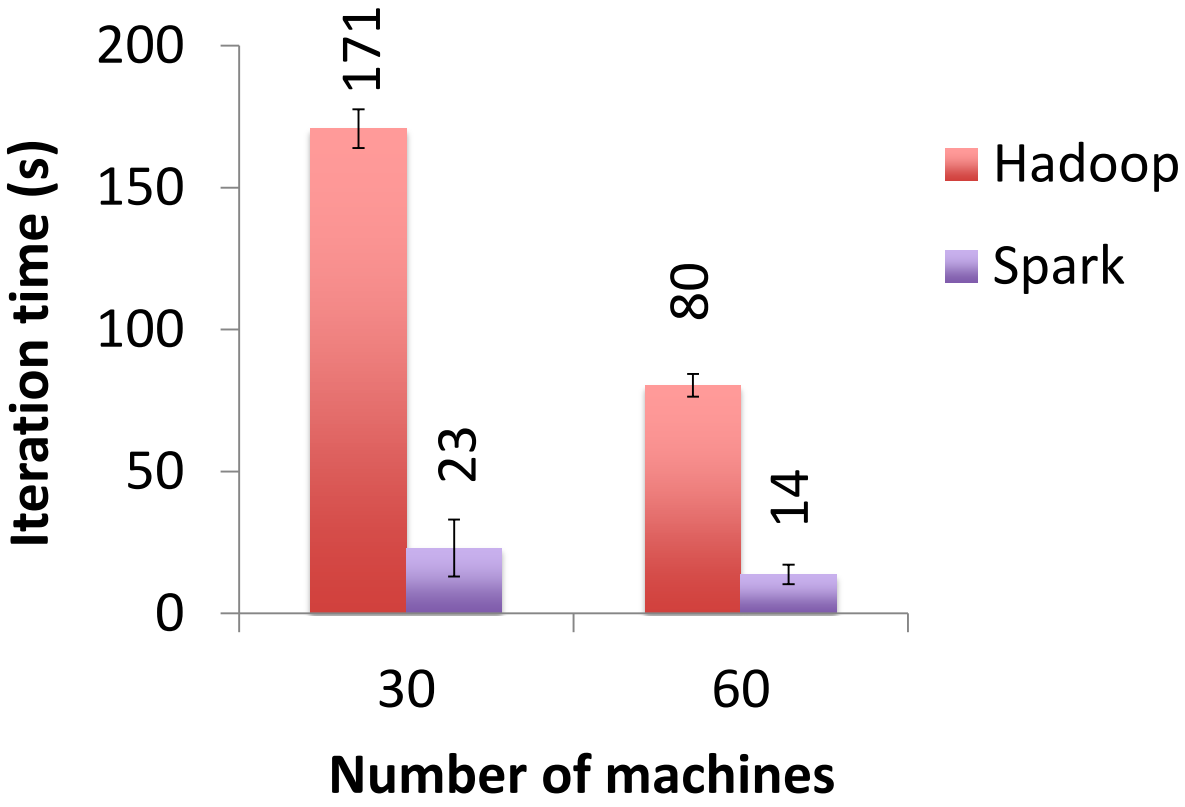
- Table (RDD) operators are inherited from Spark:

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

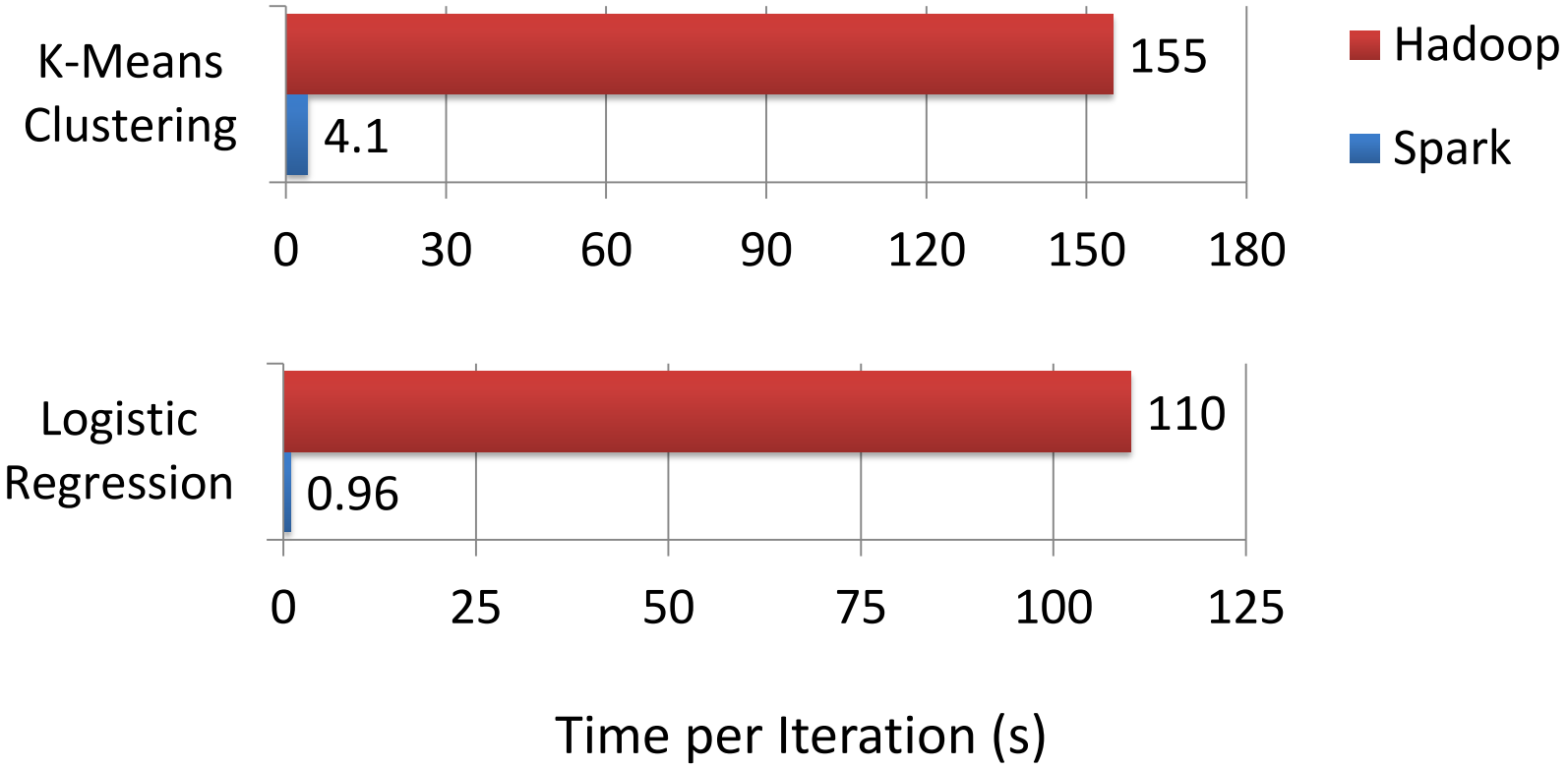
GraphX API

```
class Graph [ V, E ] {
    // Table Views -----------------
    def vertices: RDD[ (Id, V) ]
    def edges: RDD[ (Id, Id, E) ]
    def triplets: RDD[ ((Id, V), (Id, V), E) ]

    // Transformations ------------------------------
    def reverse: Graph[V, E]
    def filterV(p: (Id, V) => Boolean): Graph[V,E]
    def filterE(p: Edge[V,E] => Boolean): Graph[V,E]
    def mapV[T](m: (Id, V) => T ): Graph[T,E]
    def mapE[T](m: Edge[V,E] => T ): Graph[V,T]

    // Joins ----------------------------------------
    def joinV[T](tbl: RDD[(Id, T)]): Graph[(V, Opt[T]), E ]
    def joinE[T](tbl: RDD[(Id, Id, T)]): Graph[V, (E, Opt[T])]

    // Computation ----------------------------------
    def aggregateNeighbors[T](mapF: (Edge[V,E]) => T,
                              reduceF: (T, T) =>

                              direction:
EdgeDir): Graph[T, E]
}
```

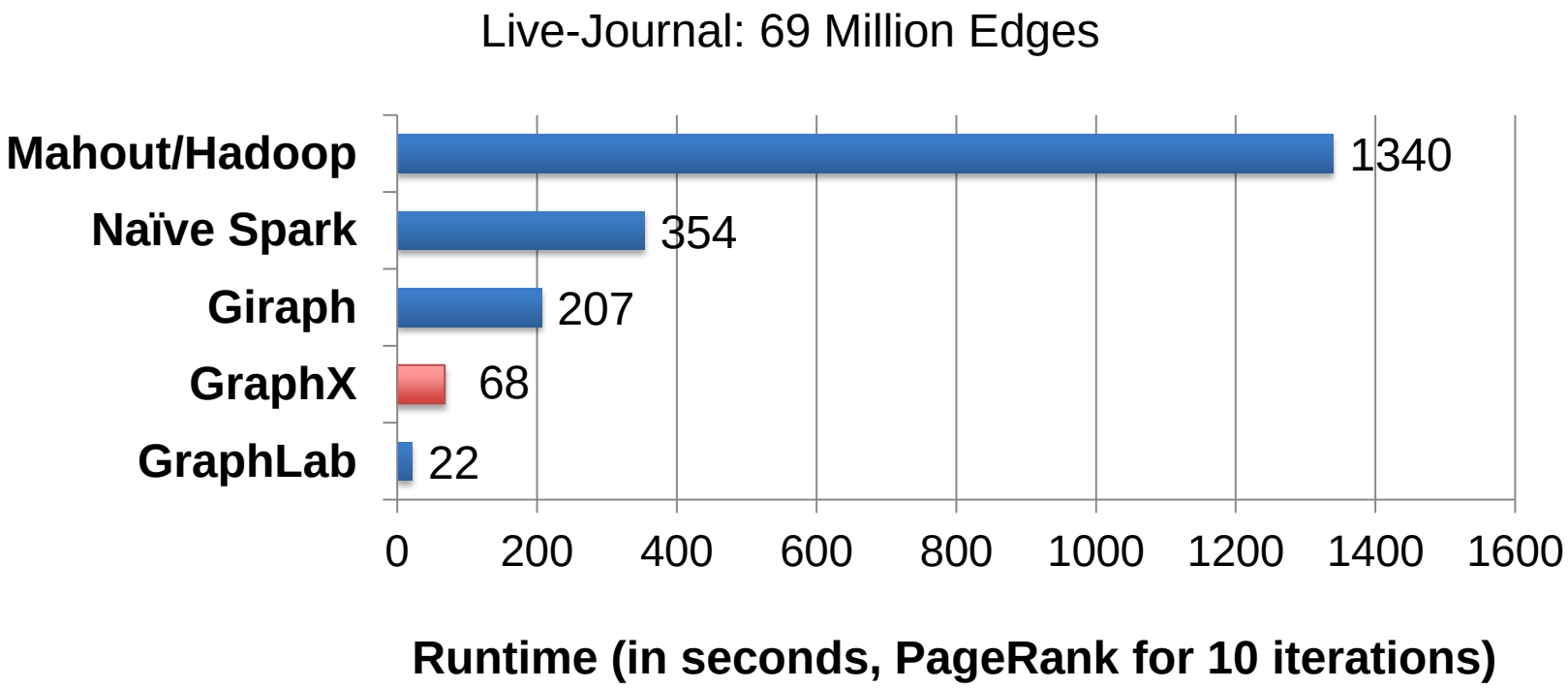
PageRank Performance



Other Iterative Algorithms

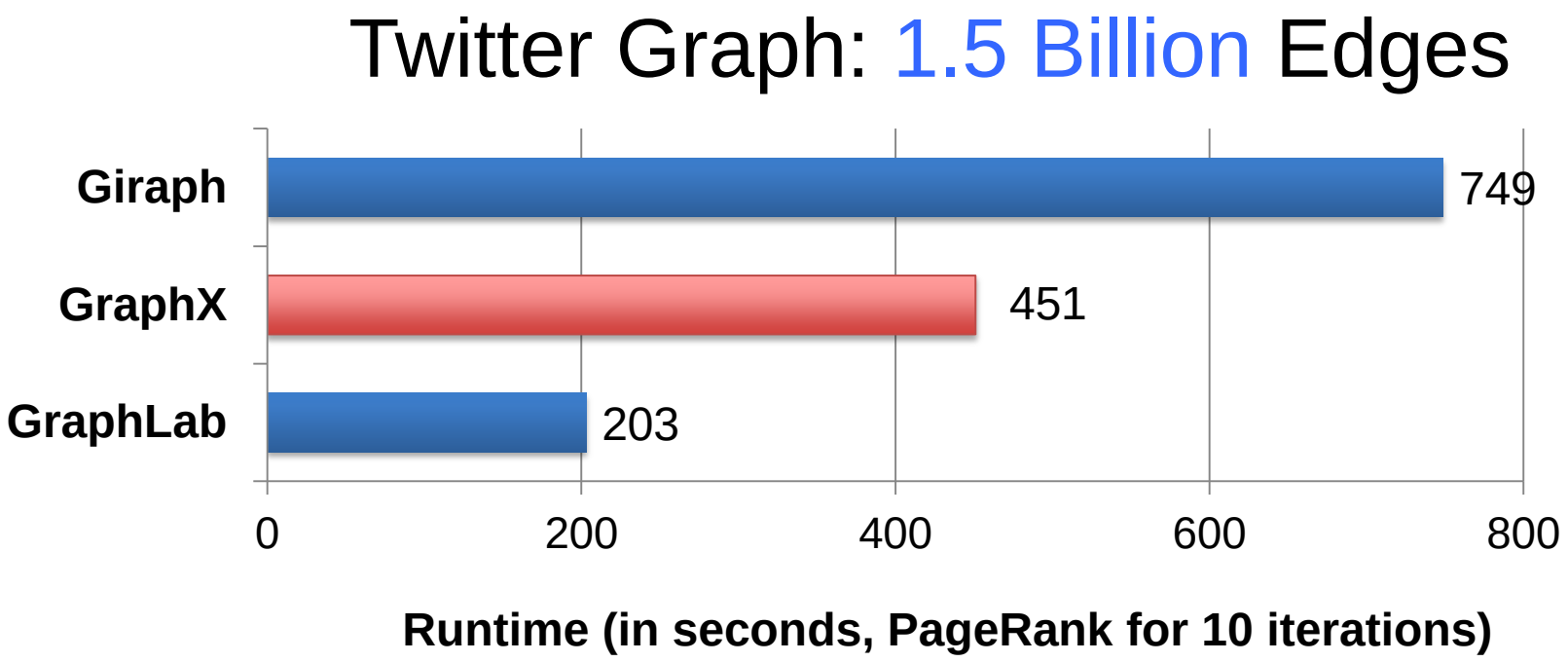


Performance Comparisons



GraphX is roughly 3x slower than GraphLab

GraphX scales to larger graphs



GraphX is roughly 2x slower than GraphLab

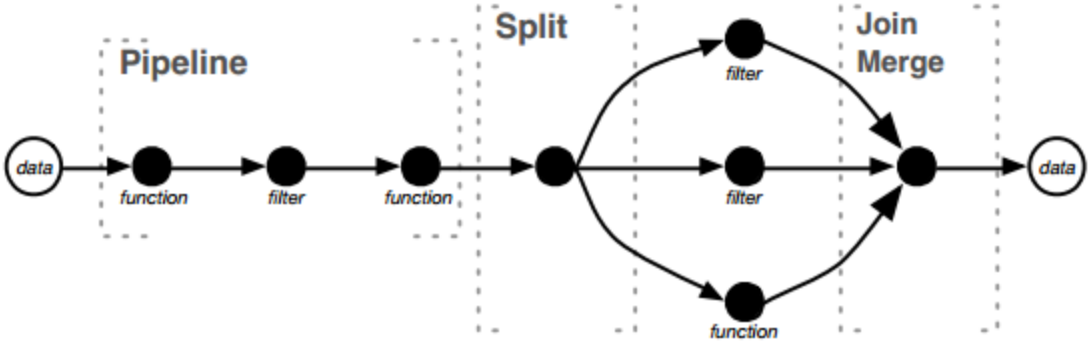
- Scala + Java overhead: Lambdas, GC time, …
- No shared memory parallelism: 2x increase in comm.

PageRank is just one stage….

What about a pipeline?

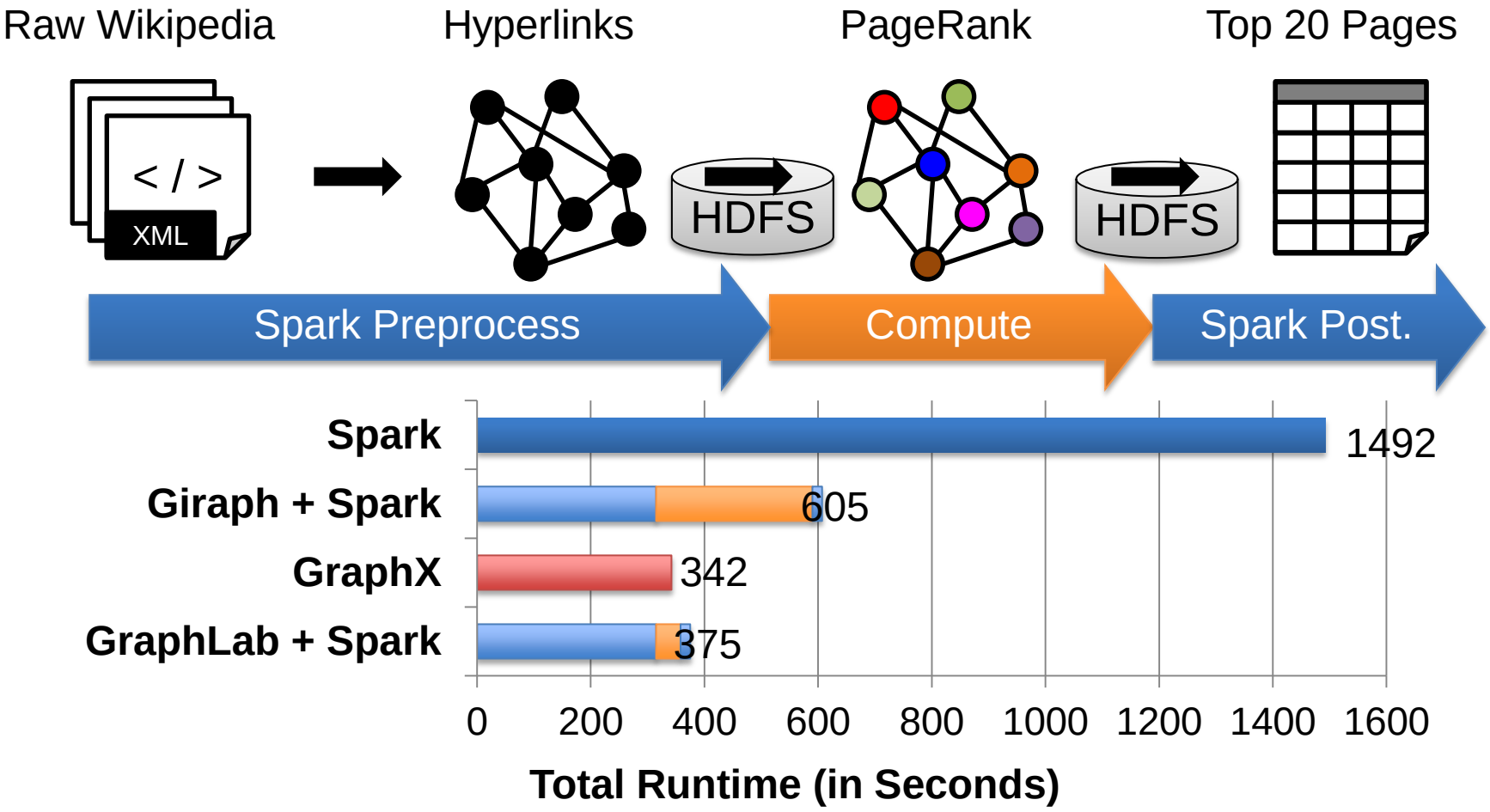
SOME COMMON PATTERNS

- Functions
- Filters
- Joins
  - Inner / Outer / Mixed
  - Asymmetrical / Symmetrical
- Merge (Union)
- Grouping
  - Secondary Sorting
  - Unique (Distinct)
- Aggregations
  - Count, Average, etc



Topology

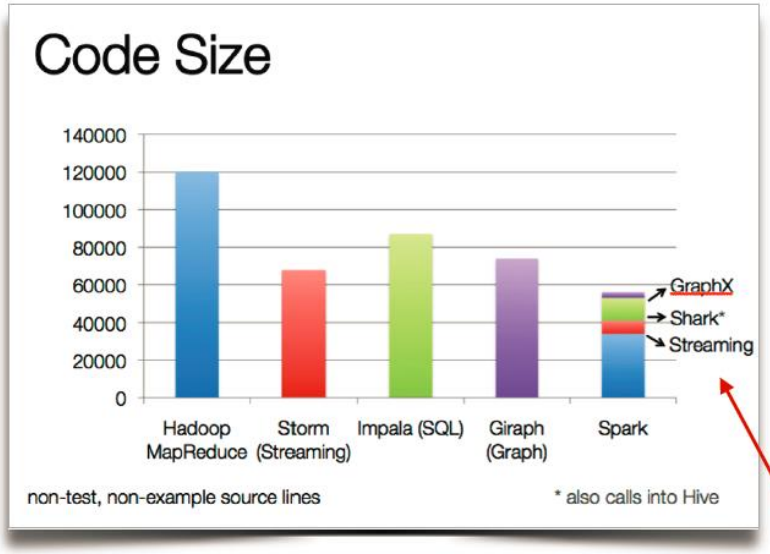
A Small Pipeline in GraphX



Timed end-to-end GraphX is *faster* than GraphLab

Spark v Hadoop - Code Size

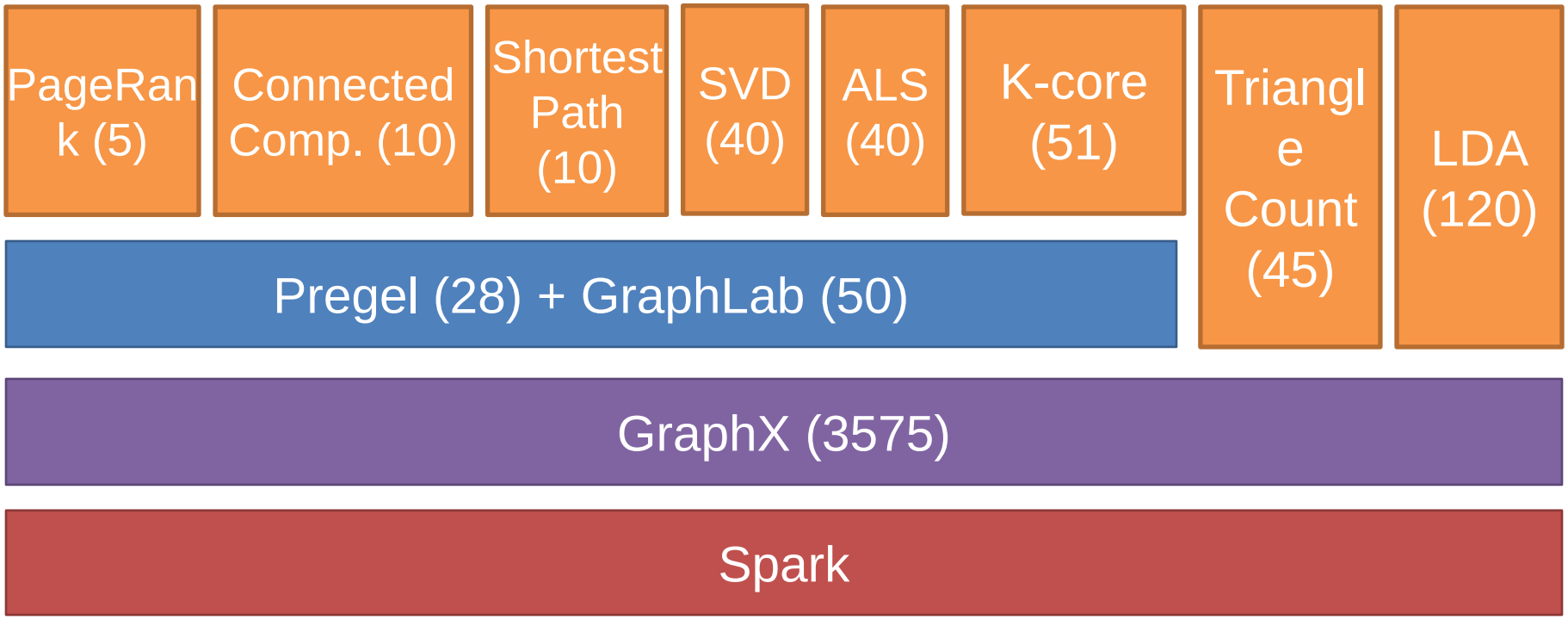
A Brief History: *Spark*



The State of Spark, and Where We're Going Next
Matei Zaharia
Spark Summit (2013)
youtu.be/nU6vO2EJAb4

used as libs, instead of specialized systems

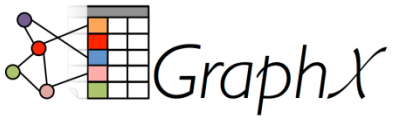
The GraphX Stack (Lines of Code)



Graph Analytics Code Example

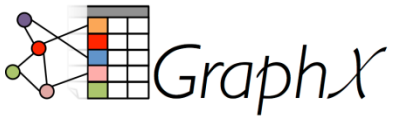
Code Example – Use Case

- Use Case:
  - Use GraphX to analyze Wikipedia data and implement graph algorithms in Spark.
- Data Sets:
  - Articles (Nodes/Vertices): `graphx-wiki-vertices.txt`
  - Links to articles (Edges/Relationships): `graphx-wiki-edges.txt`



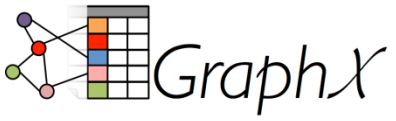
Code Example Steps

- Import Spark GraphX API
- Load Article data from text file into RDD
- Load Links data from text file into RDD
- Construct the Graph using the two datasets
- Parse article rows into pairs of vertex ID and title
- Parse link rows into Edge objects with placeholder 0 attribute
- Create graph by calling Graph constructor with vertex RDD, our edge RDD & default vertex attribute
- Cache the resulting graph in memory to avoid reloading it from disk each time we use it
- Verify the data created so far



Code Example Steps - 2

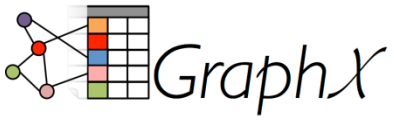
- Run graph analytics**
- Run PageRank on Wikipedia
- Find titles of ten most important articles in Berkeley subgraph of Wikipedia



Demo

Serialization

- GraphX requires Kryo serializer to achieve maximum performance
- `spark.serializer` property in Spark UI
- By default Kryo Serialization is not enabled
- Enable Kryo Serialization for the spark shell
- Set Serialization setting in "spark-shell.cmd" file

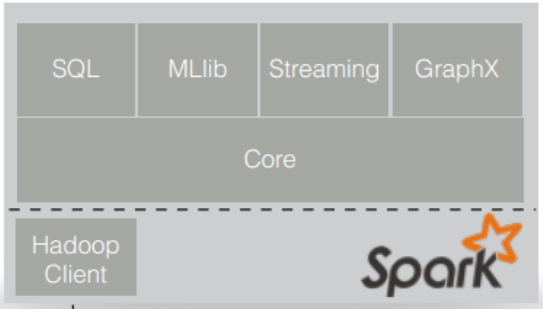


Demo with Serialization

Spark Distro Platforms

Spark Support Included on Big Data Platforms

- While this build process is very easy, it's even easier to have the runtime pre-built…
- Platform support also indicates stronger integration testing, supported, and integrated management tools



cloudera MAPR DATASTAX

Hortonworks IBM Pivotal ORACLE

DATABRICKS

http://spark.apache.org/docs/latest/hadoop-third-party-distributions.html

Conclusions

Spark and GraphX Resources

Spark:

- Spark Main Site (http://spark.apache.org/docs/latest/index.html)
- Spark Summit 2014 Conference (http://spark-summit.org/2014)
- Spark Configuration (http://spark.apache.org/docs/latest/configuration.html)
- Tuning Spark (http://spark.apache.org/docs/latest/tuning.html)
- Monitoring and Instrumentation (http://spark.apache.org/docs/latest/monitoring.html)
- Spark Overview ("Where to Go from Here" section) (http://spark.apache.org/docs/latest/index.html)

API Docs:

- Scala (http://spark.apache.org/docs/latest/api/scala/index.html#org.apache.spark.package)
- Java (http://spark.apache.org/docs/latest/api/java/index.html)
- Python (http://spark.apache.org/docs/latest/api/python/index.html)

GraphX:

- Graph Analytics with Graphx (https://databricks-training.s3.amazonaws.com/graph-analytics-with-graphx.html)
- GraphX Programming Guide (http://spark.apache.org/docs/latest/graphx-programming-guide.html)
- GraphX on Github (http://amplab.github.io/graphx/)

Speaker Contact Information

- **InfoQ Lead Editor:** http://www.infoq.com/author/Srini-Penchikala
- **Email:** srinipenchikala@gmail.com
- **LinkedIn:** http://linkedin.com/in/SriniPenchikala
- **Twitter:** @srinip
- Interested in writing articles for InfoQ?
  - Send an email to: srini@infoq.com

Q&A
Questions & Comments