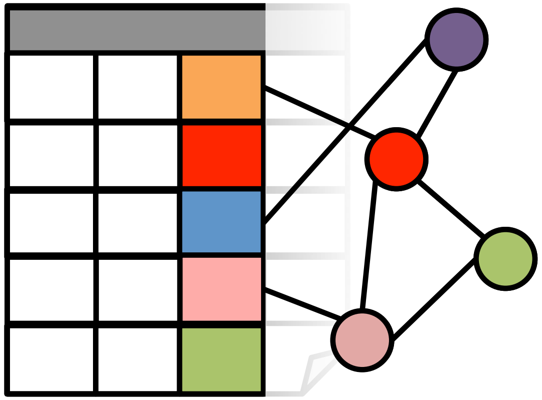
GraphX

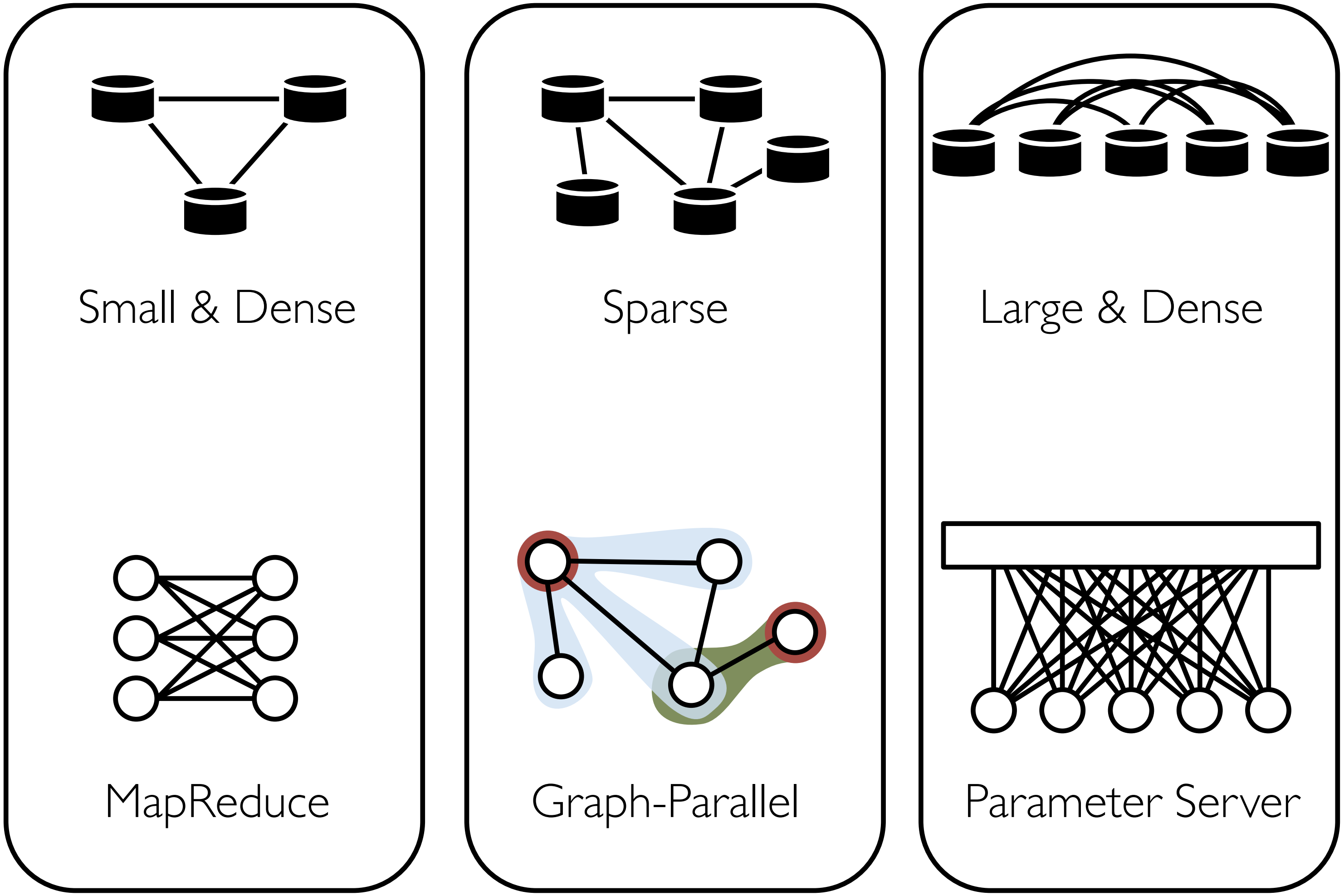


Graph Analytics in Spark

Ankur Dave
Graduate Student, UC Berkeley AMPLab

Joint work with Joseph Gonzalez, Reynold Xin, Daniel Crankshaw, Michael Franklin, and Ion Stoica

Machine Learning Landscape



Model & Dependencies

Small & Dense

Sparse

Large & Dense

Architecture

MapReduce

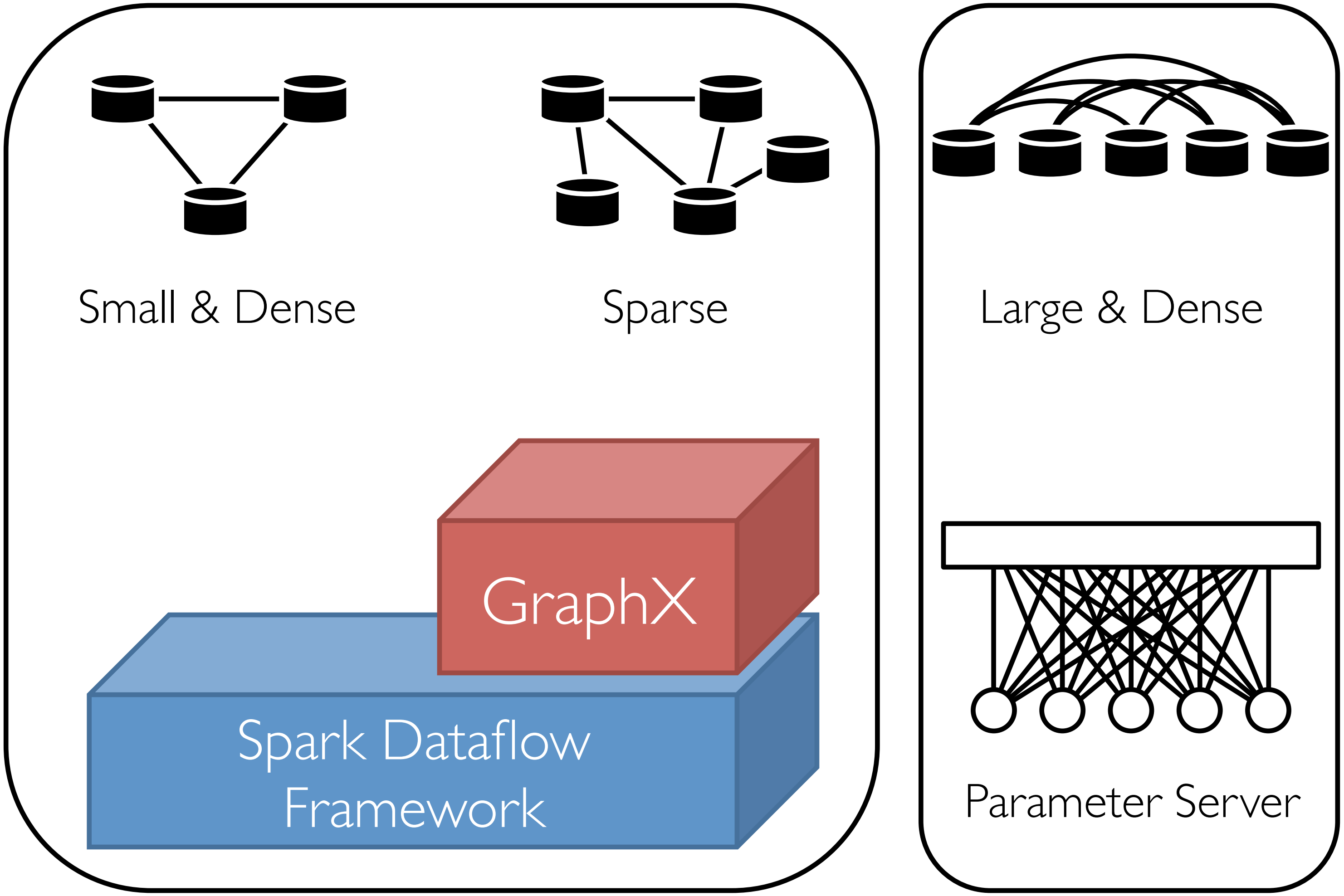
Graph-Parallel

Parameter Server

Machine Learning Landscape

Model & Dependencies

Architecture



Small & Dense

Sparse

Large & Dense

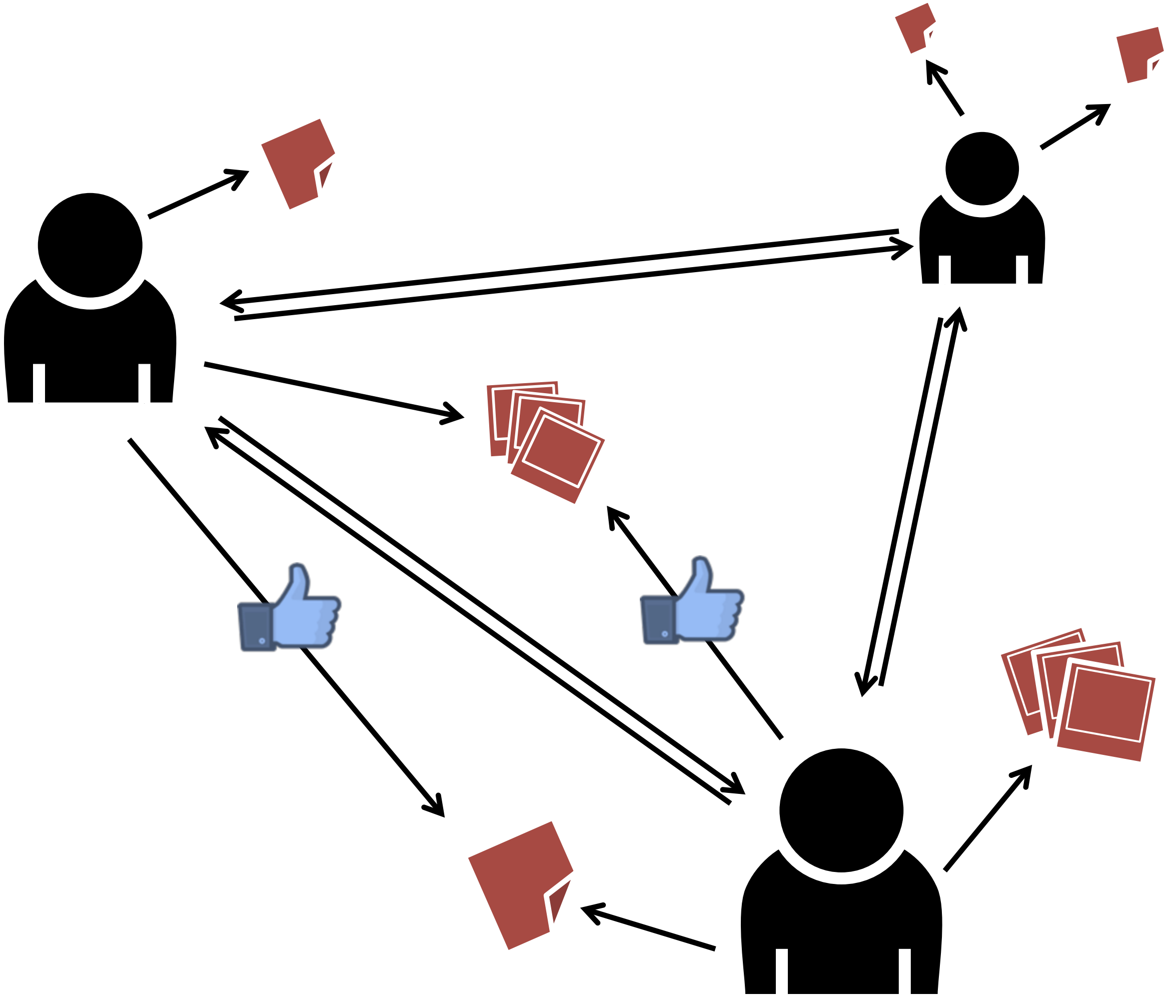
GraphX

Spark Dataflow Framework

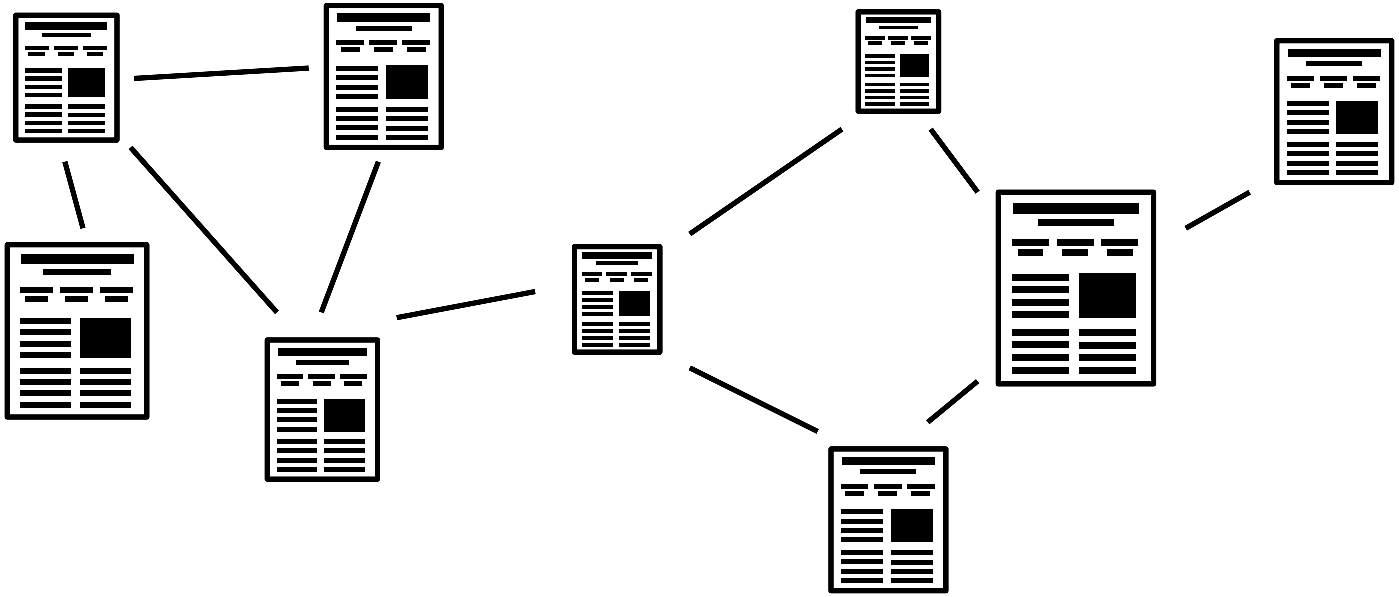
Parameter Server

Graphs

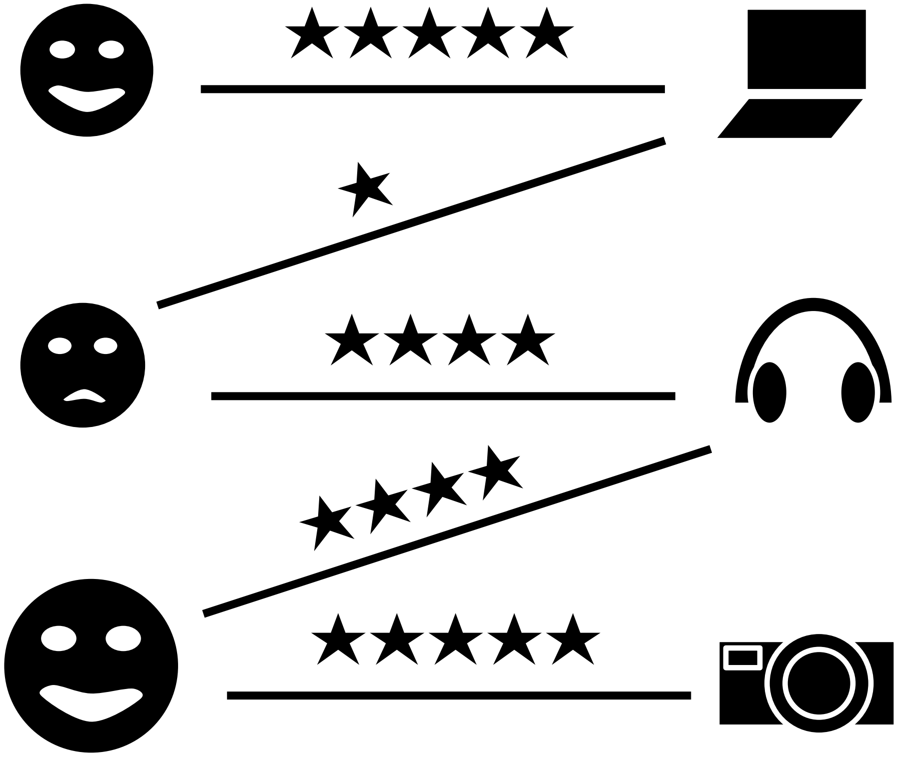
Social Networks



Web Graphs

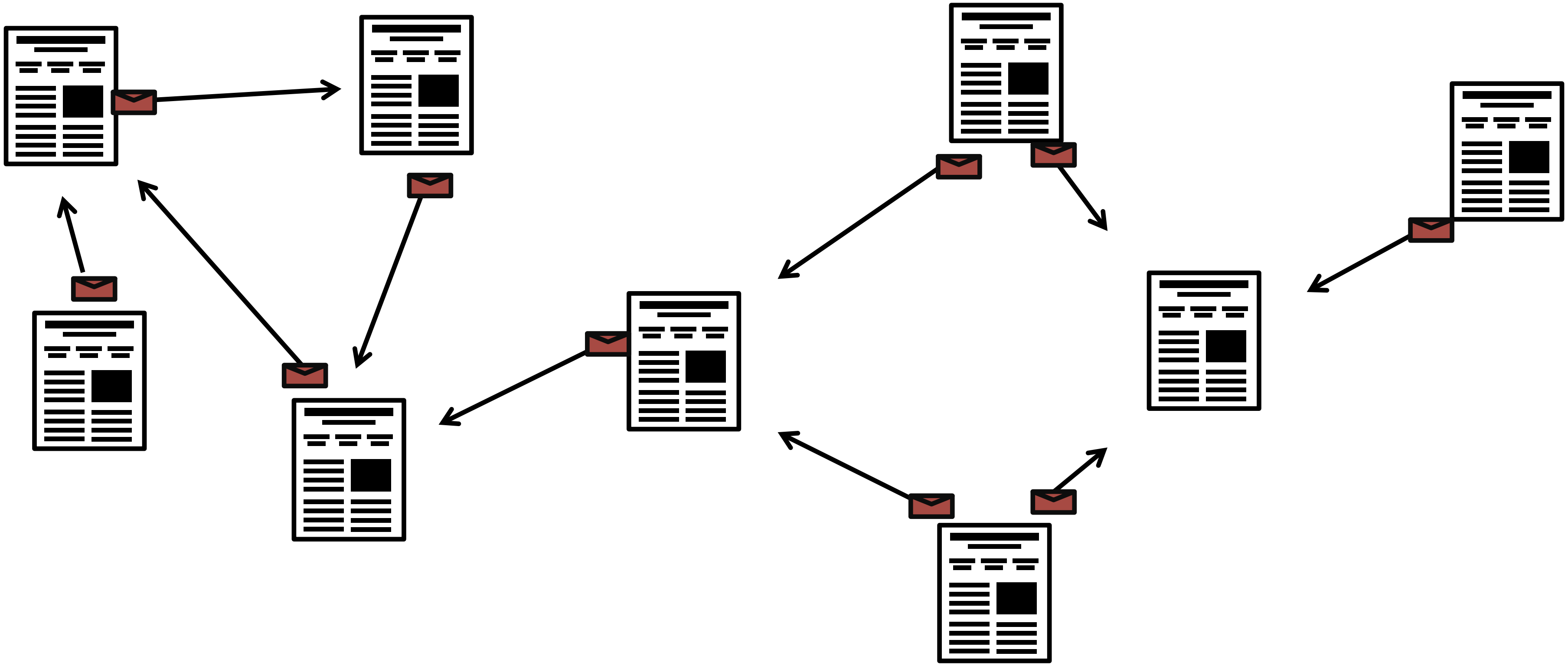


User-Item Graphs

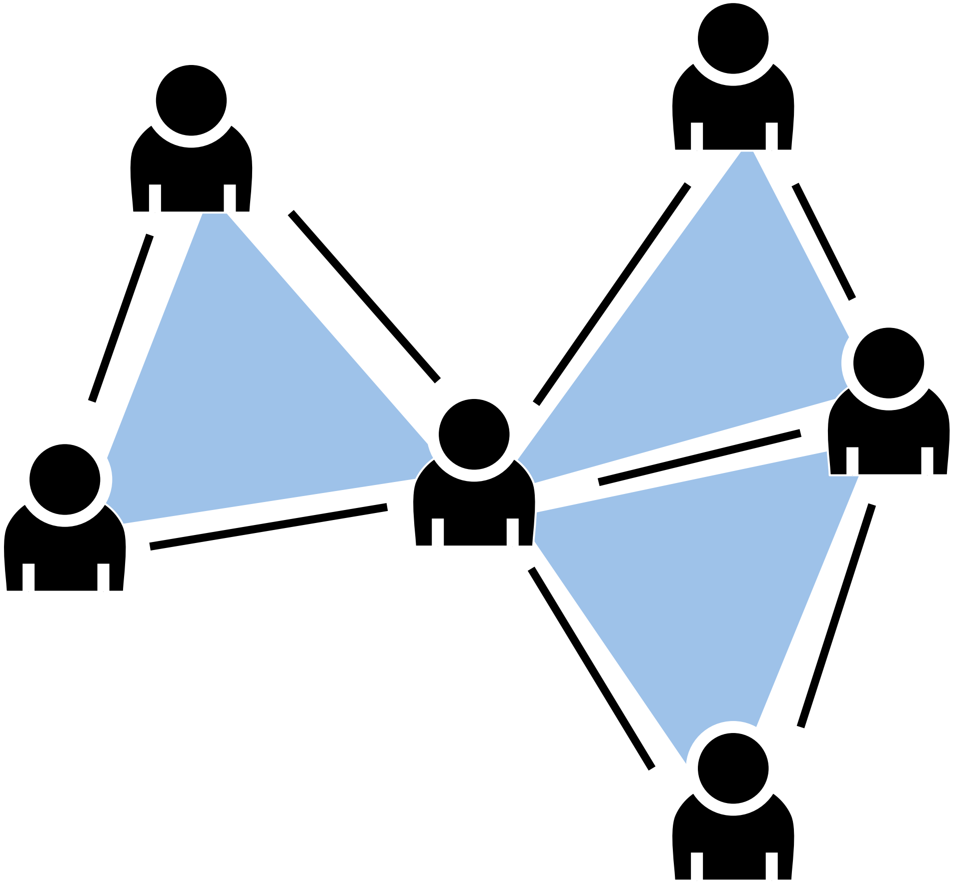


Graph Algorithms

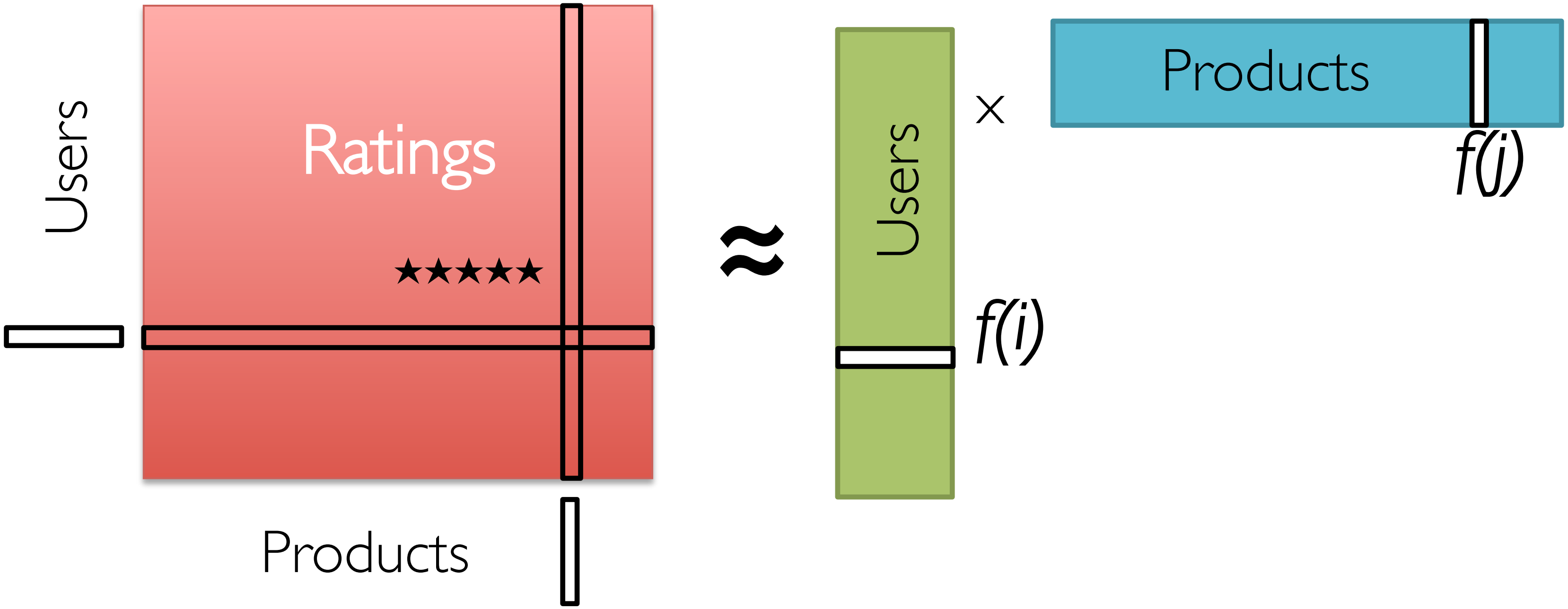
PageRank



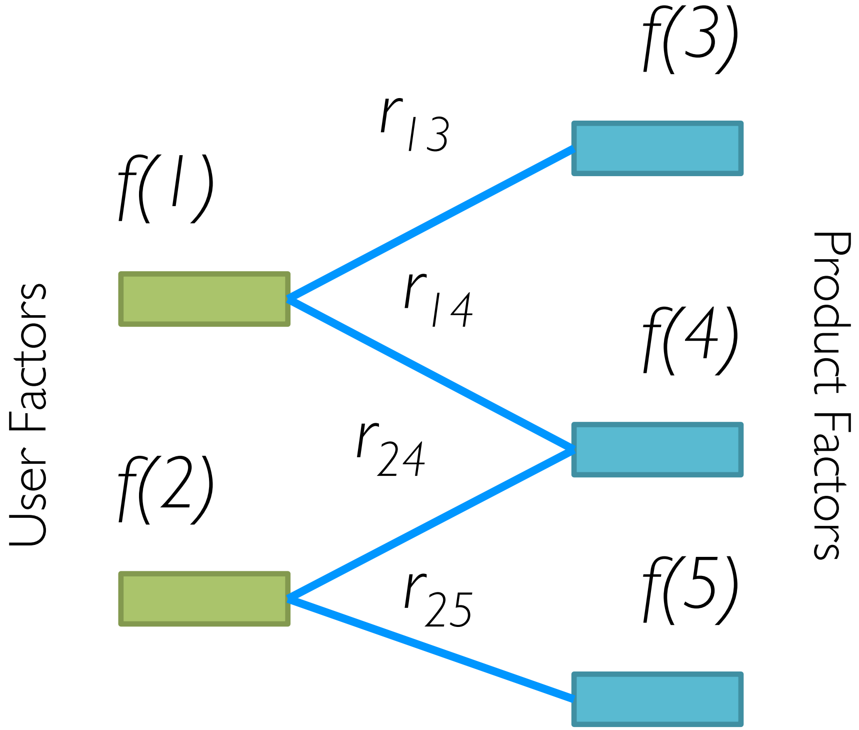
Triangle Counting



Collaborative Filtering

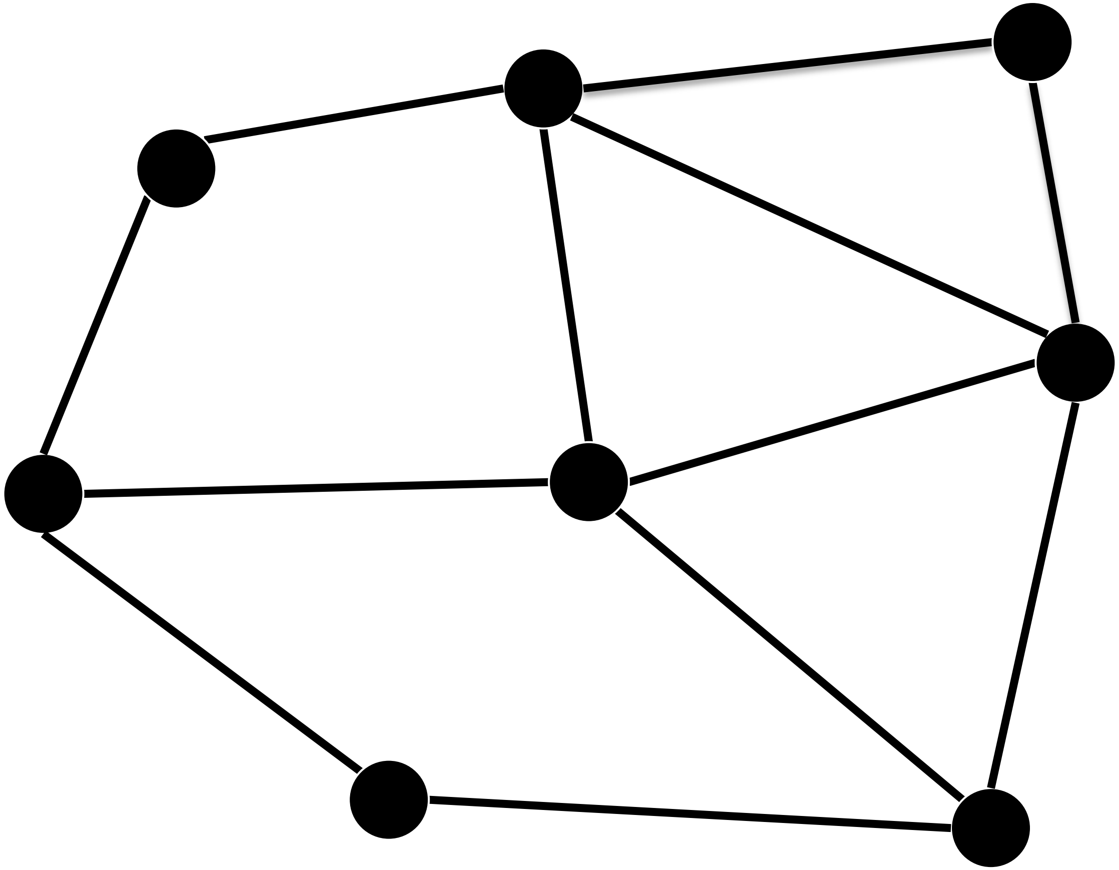


Collaborative Filtering

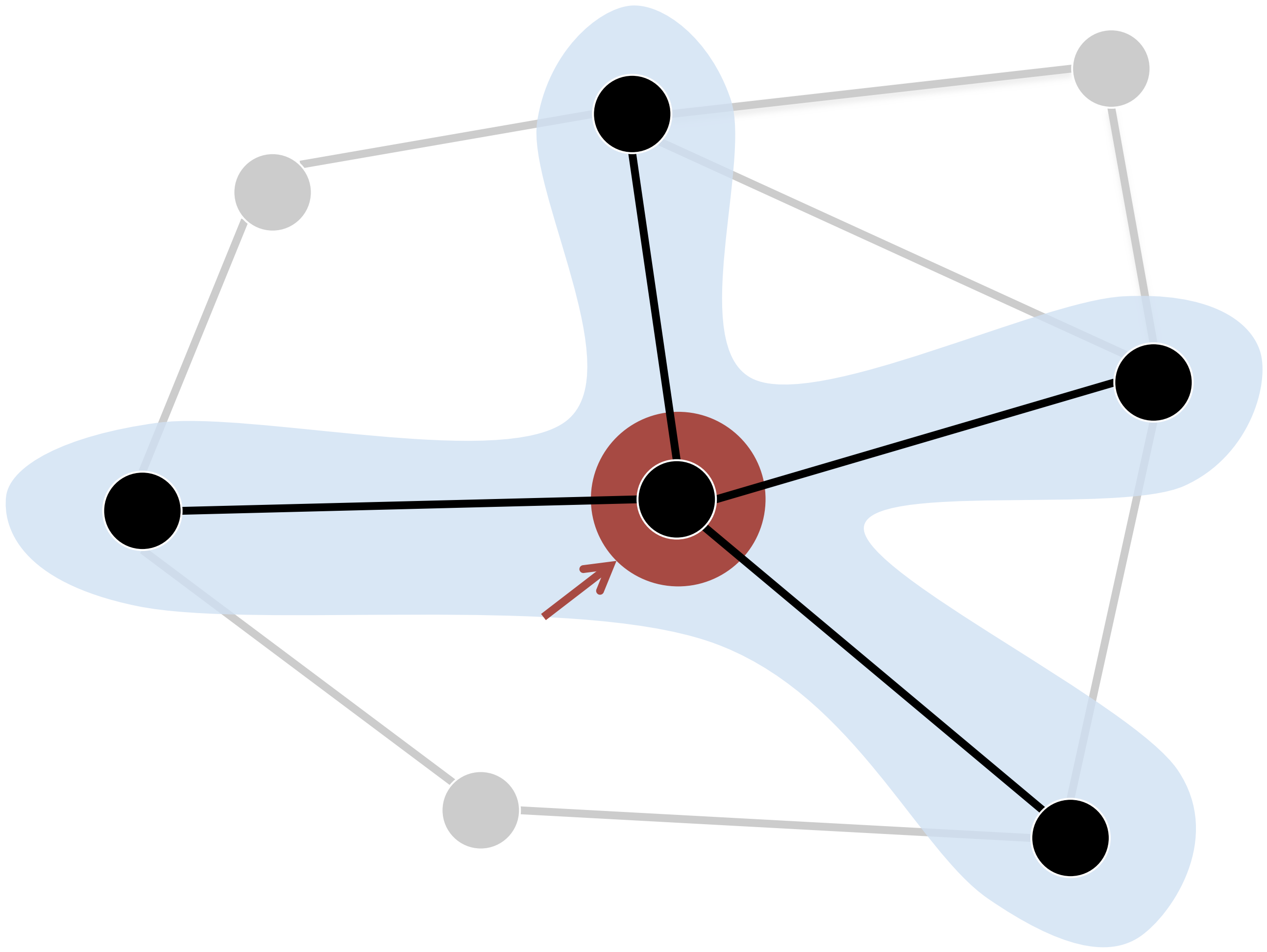


$$f[i] = \arg\min_{w \in \mathbb{R}^d} \sum_{j \in \mathrm{Nbrs}(i)} \left(r_{ij} - w^T f[j] \right)^2 + \lambda ||w||_2^2$$

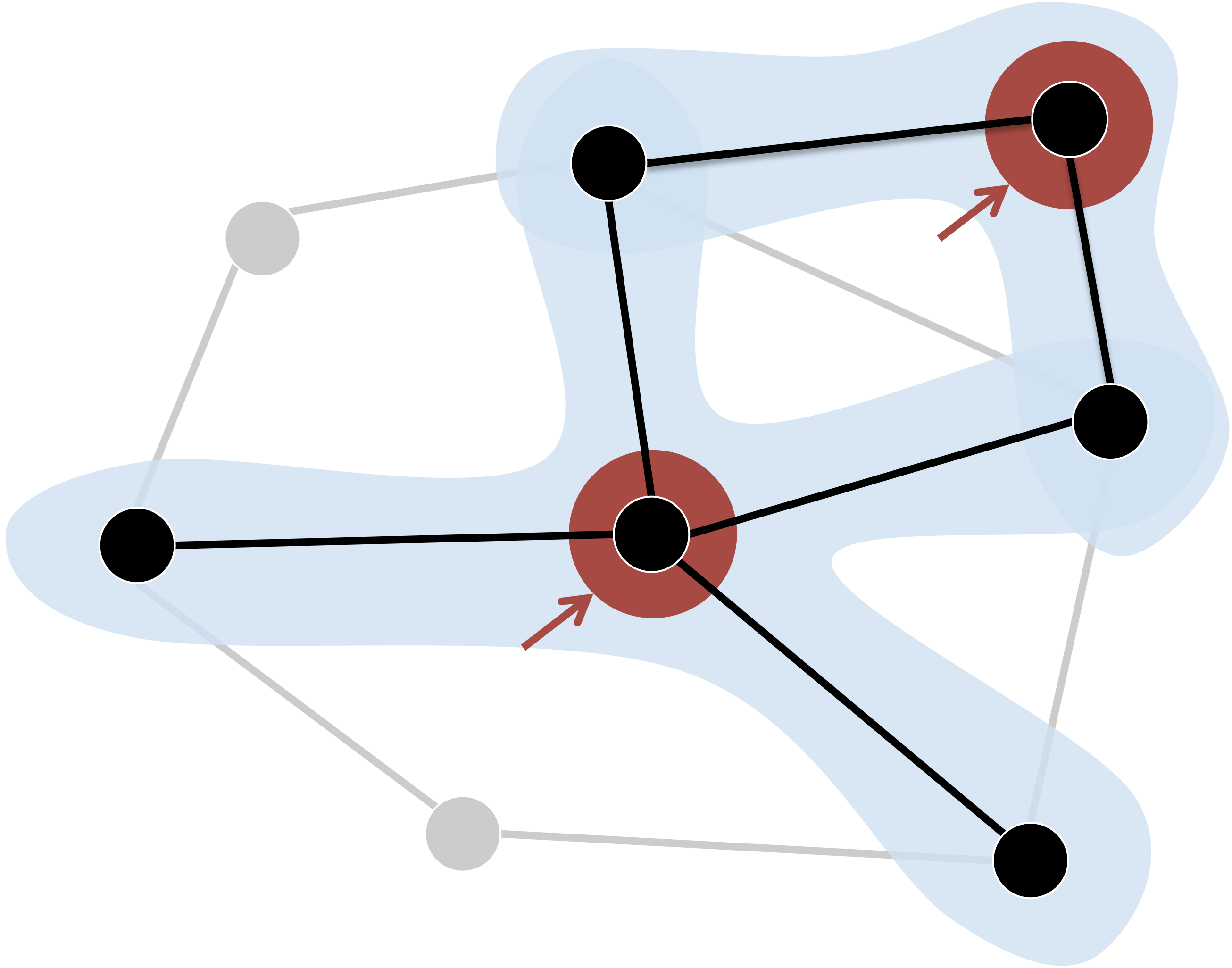
The Graph-Parallel Pattern



The Graph-Parallel Pattern



The Graph-Parallel Pattern



Many Graph-Parallel Algorithms

Collaborative Filtering

- » Alternating Least Squares
- » Stochastic Gradient Descent
- » Tensor Factorization

Structured Prediction

- » Loopy Belief Propagation
- » Max-Product Linear Programs
- » Gibbs Sampling

Semi-supervised ML

- » Graph SSL
- » CoEM

Community Detection

- » Triangle-Counting
- » K-core Decomposition
- » K-Truss

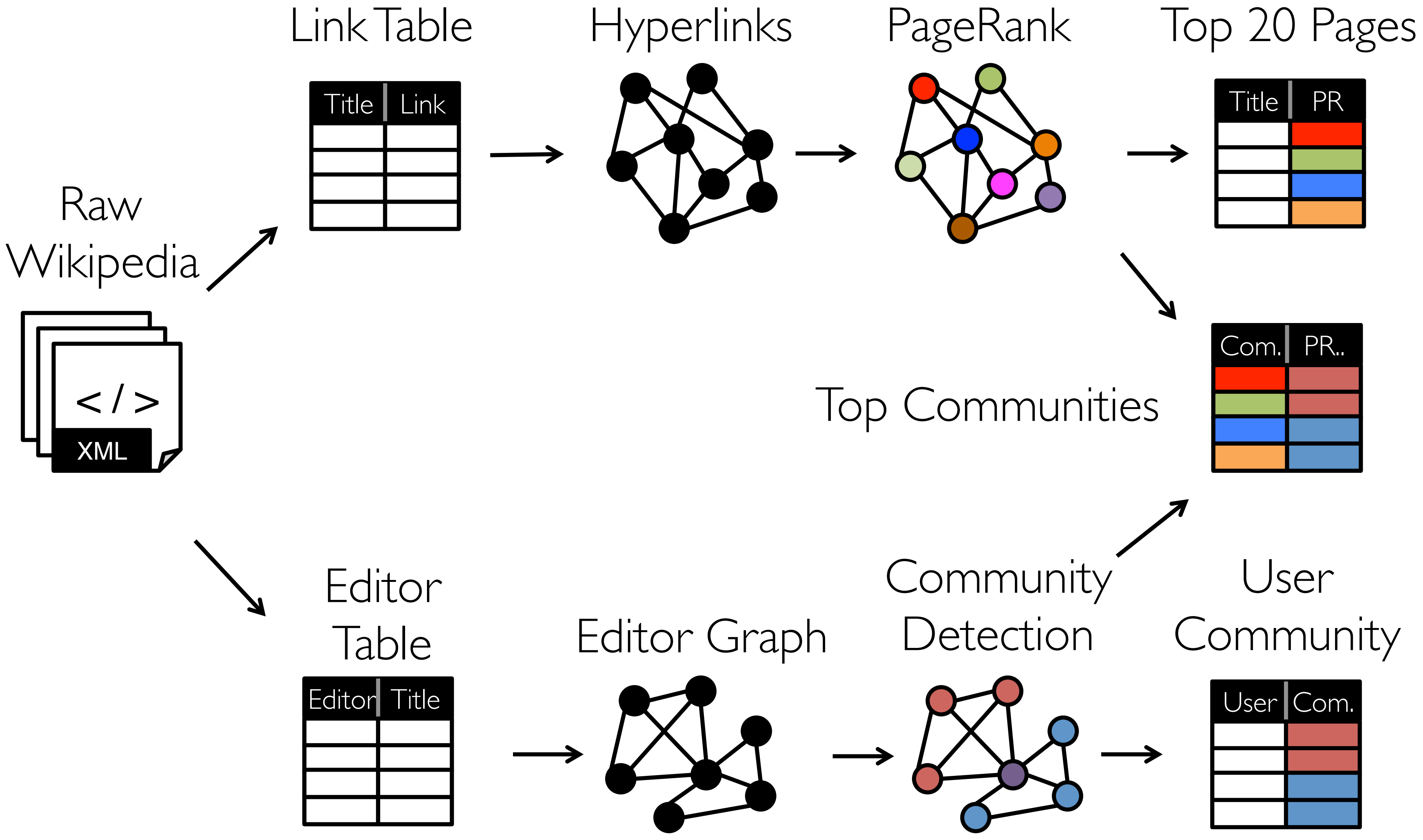
Graph Analytics

- » PageRank
- » Personalized PageRank
- » Shortest Path
- » Graph Coloring

Classification

- » Neural Networks

Modern Analytics



Raw Wikipedia

< / >

XML

Link Table

Title Link

Hyperlinks

PageRank

Top 20 Pages

Title PR

Top Communities

Com. PR..

Editor Table

Editor Title

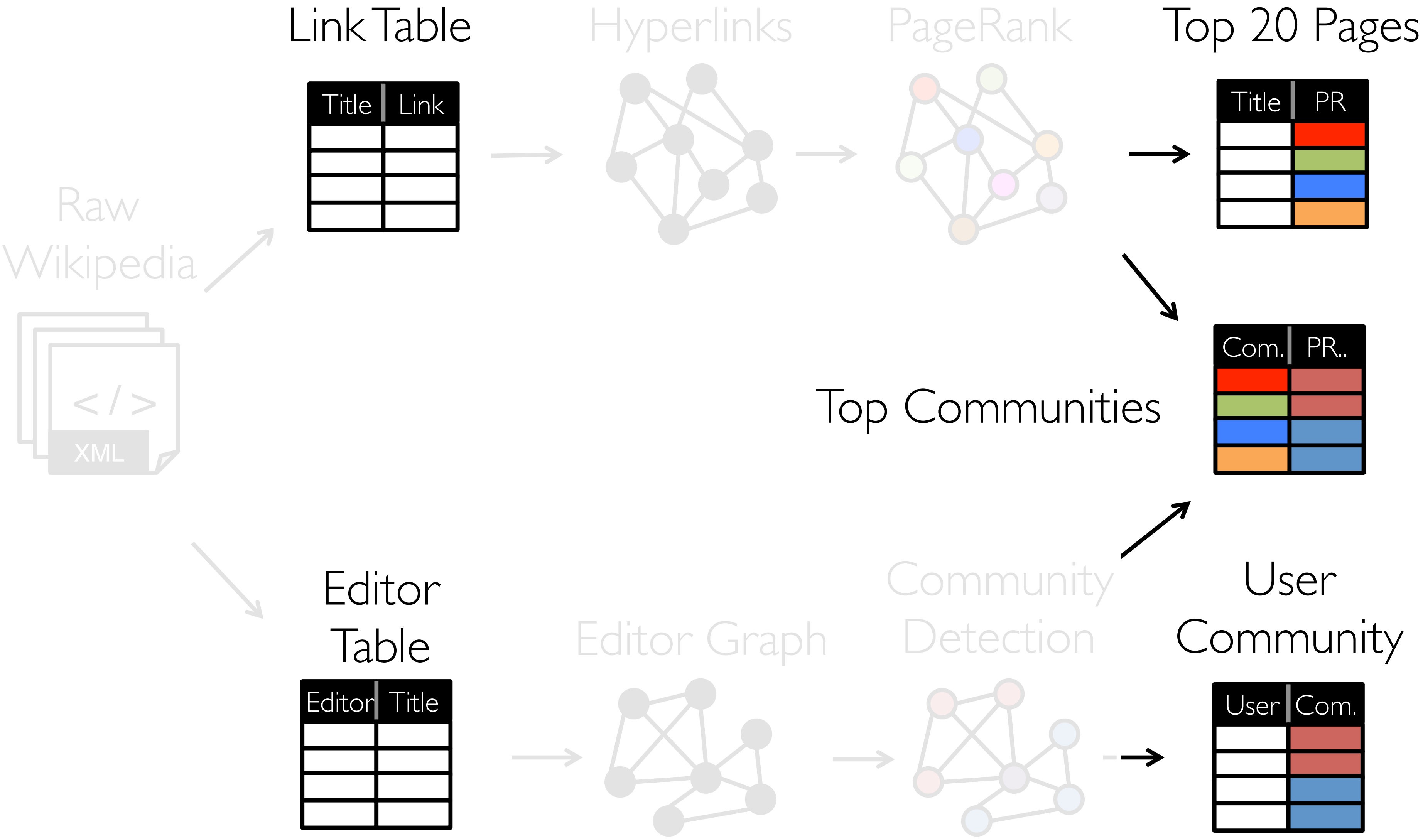
Editor Graph

Community Detection

User Community

User Com.

Tables



Raw Wikipedia

Link Table

Title	Link

Hyperlinks

PageRank

Top 20 Pages

Title	PR
	🟥
	🟩
	🟦
	🟧

Top Communities

Com.	PR..
🟥	🟥
🟩	🟥
🟦	🟦
🟧	🟦

Editor Table

Editor	Title

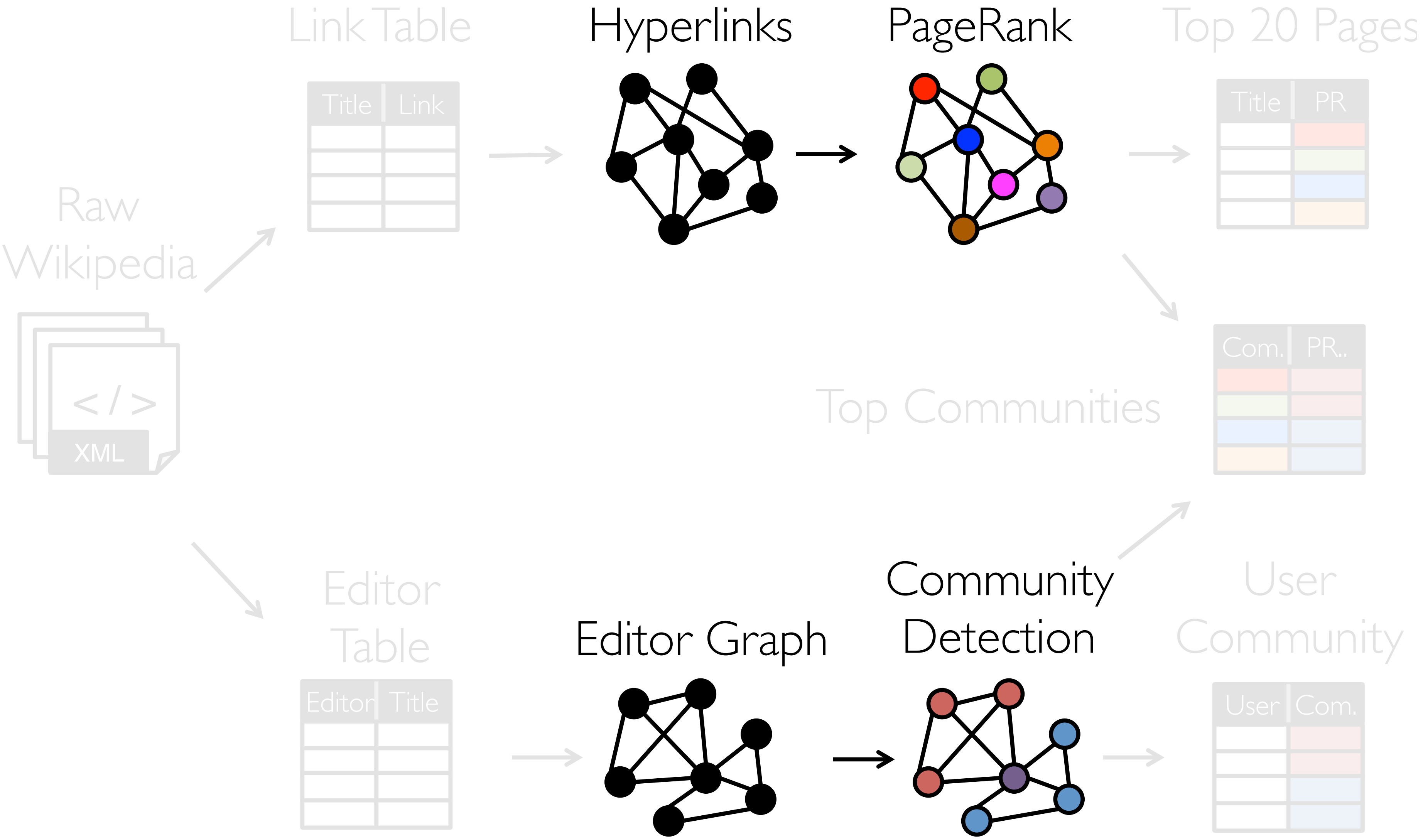
Editor Graph

Community Detection

User Community

User	Com.
	🟥
	🟥
	🟦
	🟦

Graphs



Raw Wikipedia

XML

Link Table

Title | Link

Hyperlinks

PageRank

Top 20 Pages

Title | PR

Top Communities

Com. | PR..

Editor Table

Editor | Title

Editor Graph

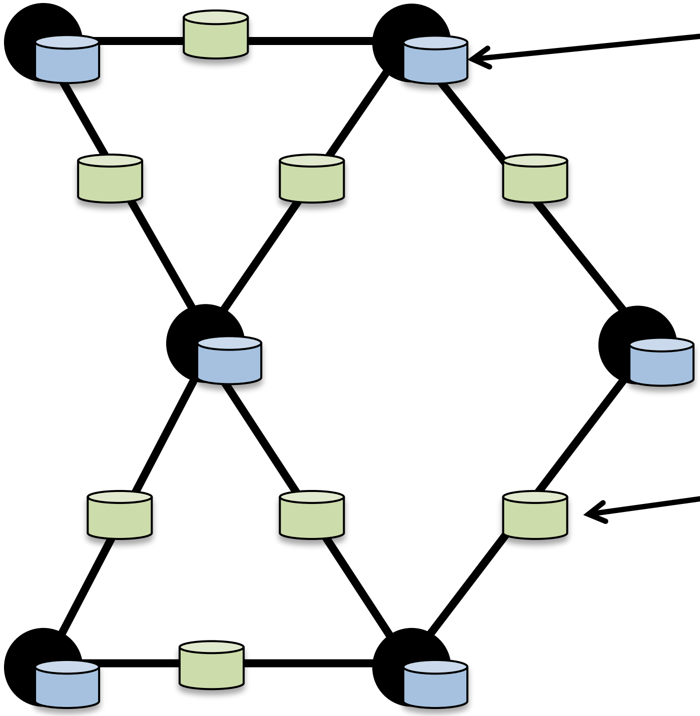
Community Detection

User Community

User | Com.

The GraphX API

Property Graphs



Vertex Property:

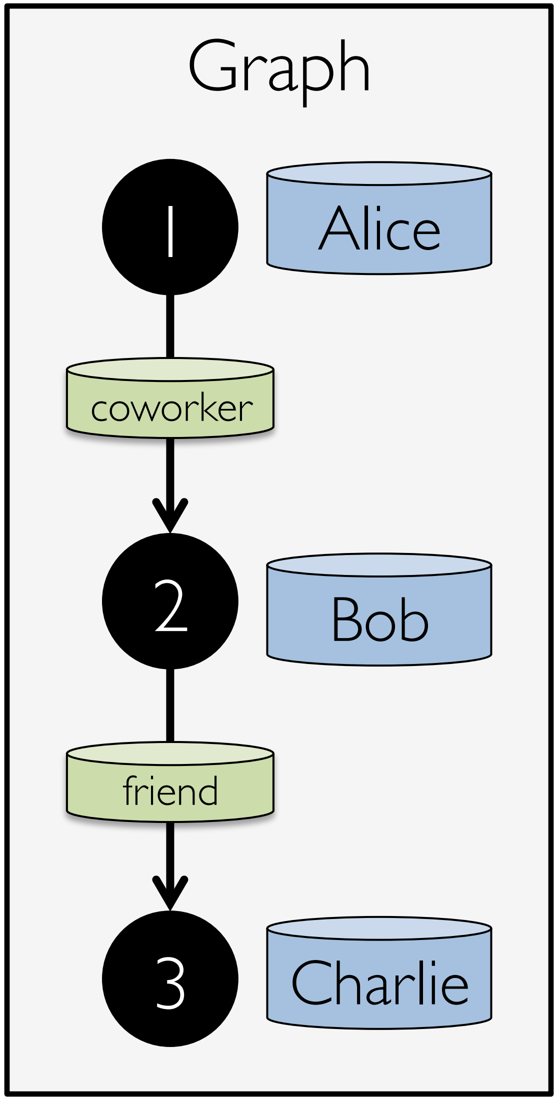
- User Profile
- Current PageRank Value

Edge Property:

- Weights
- Relationships
- Timestamps

Creating a Graph (Scala)

```
type VertexId = Long

val vertices: RDD[(VertexId, String)] =
  sc.parallelize(List(
    (1L, “Alice”),
    (2L, “Bob”),
    (3L, “Charlie”)))

class Edge[ED](
  val srcId: VertexId,
  val dstId: VertexId,
  val attr: ED)

val edges: RDD[Edge[String]] =
  sc.parallelize(List(
    Edge(1L, 2L, “coworker”),
    Edge(2L, 3L, “friend”)))

val graph = Graph(vertices, edges)
```



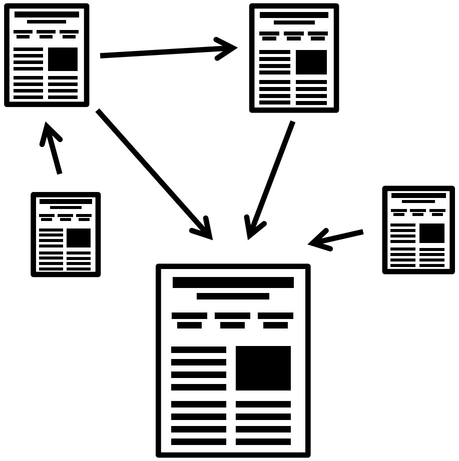
Graph Operations (Scala)

```scala
class Graph[VD, ED] {
    // Table Views -----------------------
    def vertices: RDD[(VertexId, VD)]
    def edges: RDD[Edge[ED]]
    def triplets: RDD[EdgeTriplet[VD, ED]]
    // Transformations ------------------------------------------
    def mapVertices[VD2](f: (VertexId, VD) => VD2): Graph[VD2, ED]
    def mapEdges[ED2](f: Edge[ED] => ED2): Graph[VD2, ED]
    def reverse: Graph[VD, ED]
    def subgraph(epred: EdgeTriplet[VD, ED] => Boolean,
                 vpred: (VertexId, VD) => Boolean): Graph[VD, ED]
    // Joins ----------------------------------------
    def outerJoinVertices[U, VD2]
        (tbl: RDD[(VertexId, U)])
        (f: (VertexId, VD, Option[U]) => VD2): Graph[VD2, ED]
    // Computation ----------------------------------
    def mapReduceTriplets[A](
        sendMsg: EdgeTriplet[VD, ED] => Iterator[(VertexId, A)],
        mergeMsg: (A, A) => A): RDD[(VertexId, A)]
```

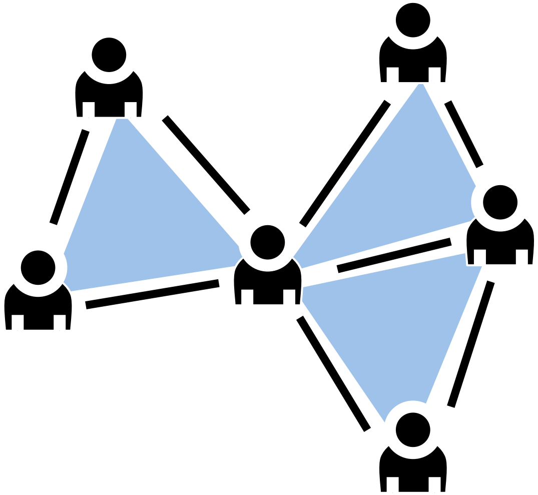
Built-in Algorithms (Scala)

```
    // Continued from previous slide
    def pageRank(tol: Double): Graph[Double, Double]
    def triangleCount(): Graph[Int, ED]
    def connectedComponents(): Graph[VertexId, ED]
    // ...and more: org.apache.spark.graphx.lib
}
```

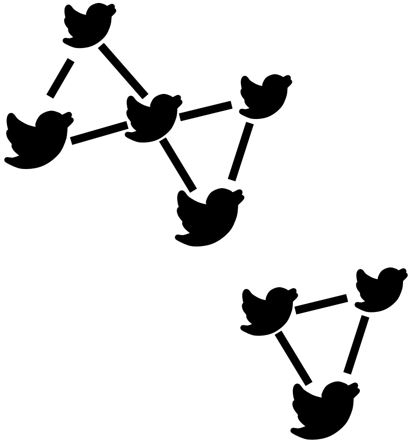
PageRank



Triangle Count

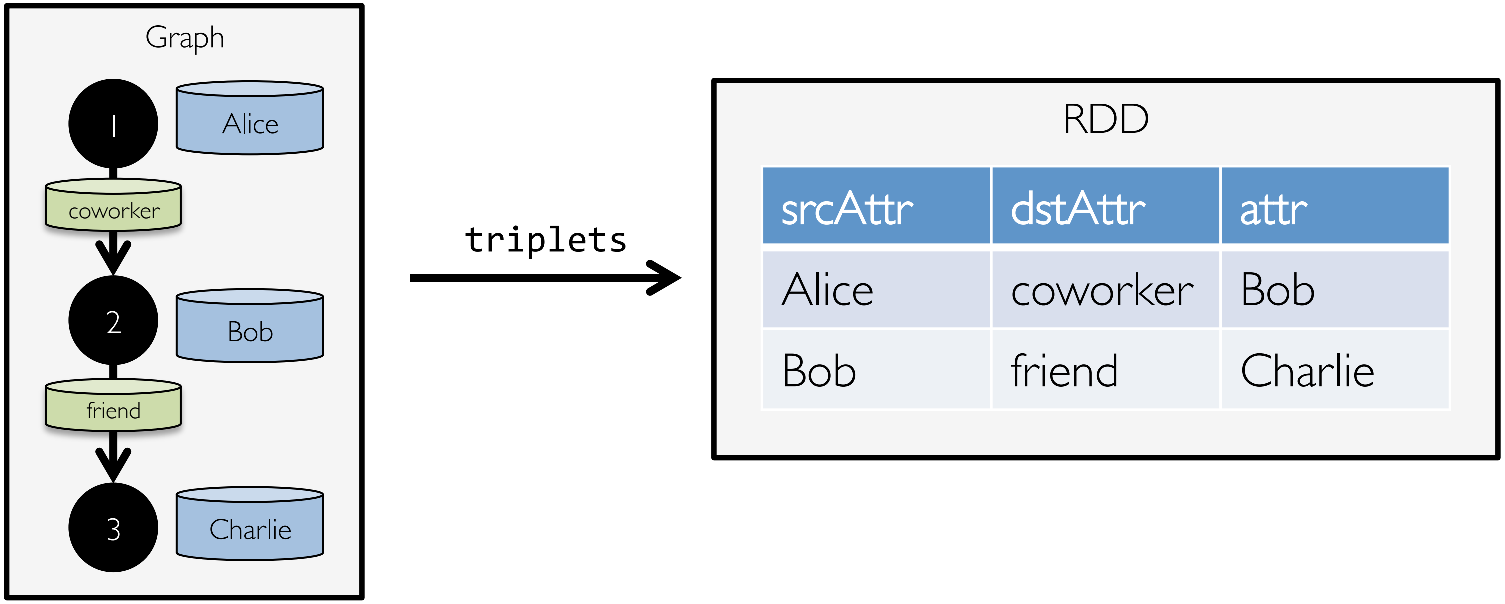


Connected Components



The triplets view

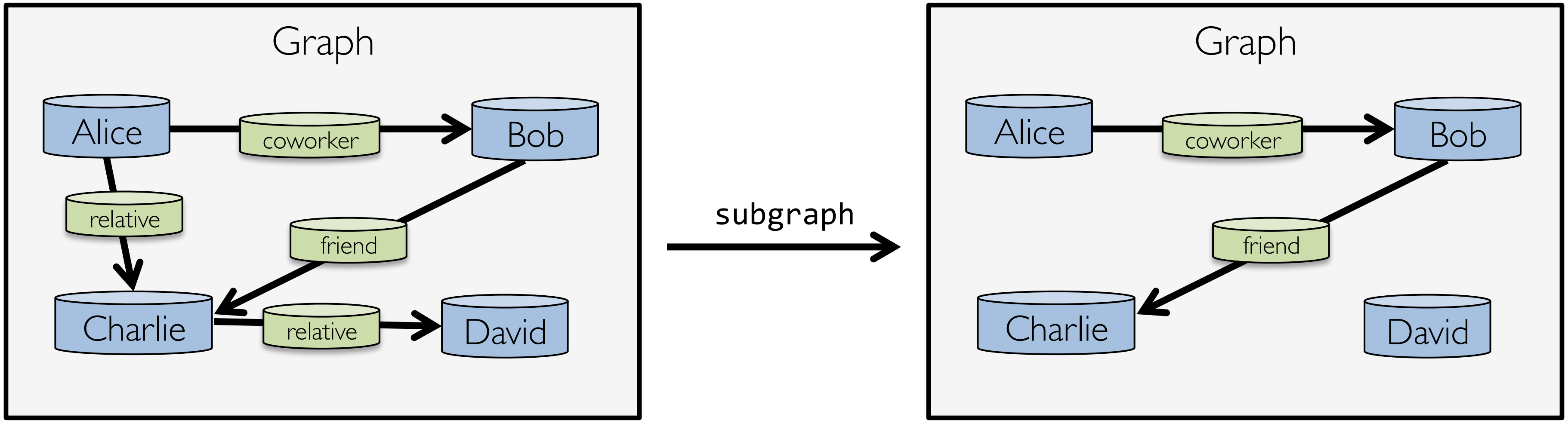
```
class Graph[VD, ED] {
    def triplets: RDD[EdgeTriplet[VD, ED]]
}

class EdgeTriplet[VD, ED](
  val srcId: VertexId, val dstId: VertexId, val attr: ED,
  val srcAttr: VD, val dstAttr: VD)
```



srcAttr	dstAttr	attr
Alice	coworker	Bob
Bob	friend	Charlie

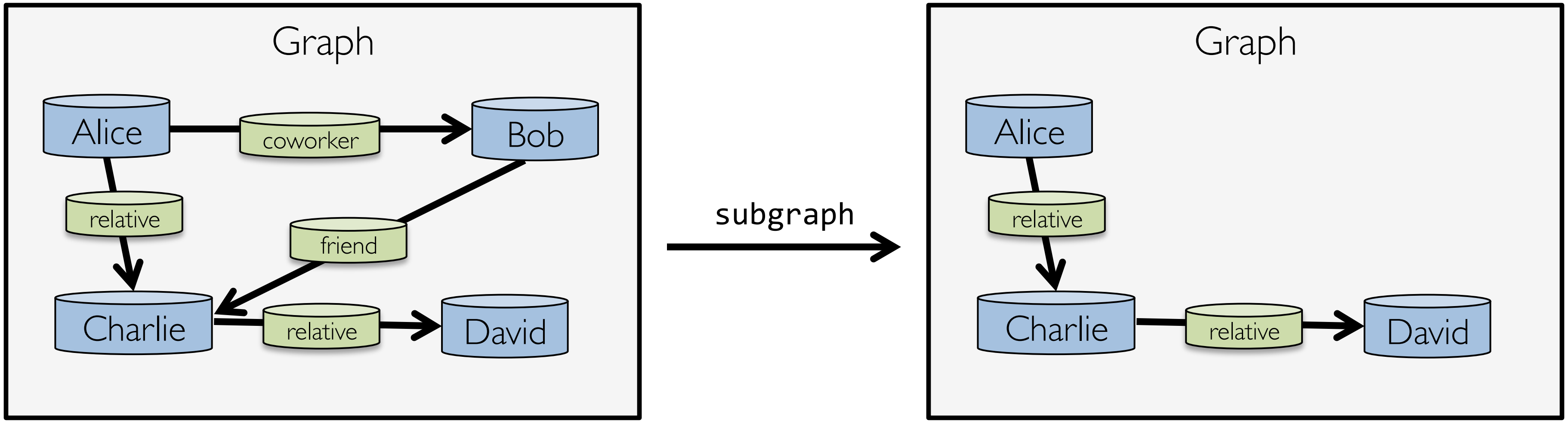
The **subgraph** transformation

```
class Graph[VD, ED] {
    def subgraph(epred: EdgeTriplet[VD, ED] => Boolean,
                 vpred: (VertexId, VD) => Boolean): Graph[VD, ED]
}

graph.subgraph(epred = (edge) => edge.attr != “relative”)
```



The **subgraph** transformation

```
class Graph[VD, ED] {
    def subgraph(epred: EdgeTriplet[VD, ED] => Boolean,
                 vpred: (VertexId, VD) => Boolean): Graph[VD, ED]
}

graph.subgraph(vpred = (id, name) => name != “Bob”)
```

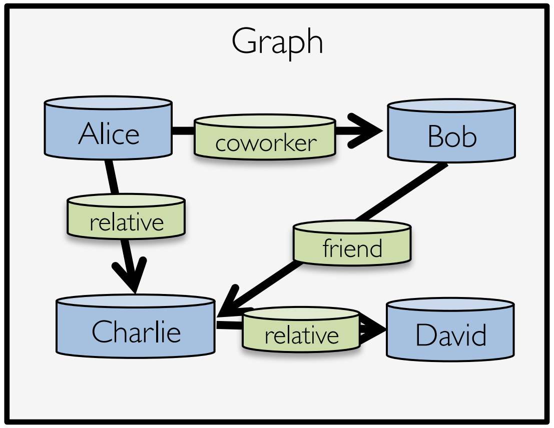


Computation with `mapReduceTriplets`

```
class Graph[VD, ED] {
  def mapReduceTriplets[A](
    sendMsg: EdgeTriplet[VD, ED] => Iterator[(VertexId, A)],
    mergeMsg: (A, A) => A): RDD[(VertexId, A)]
}
```

⚠ upgrade to `aggregateMessages` in Spark 1.2.0

```
graph.mapReduceTriplets(
  edge => Iterator(
    (edge.srcId, 1),
    (edge.dstId, 1)),
  _ + _)
```



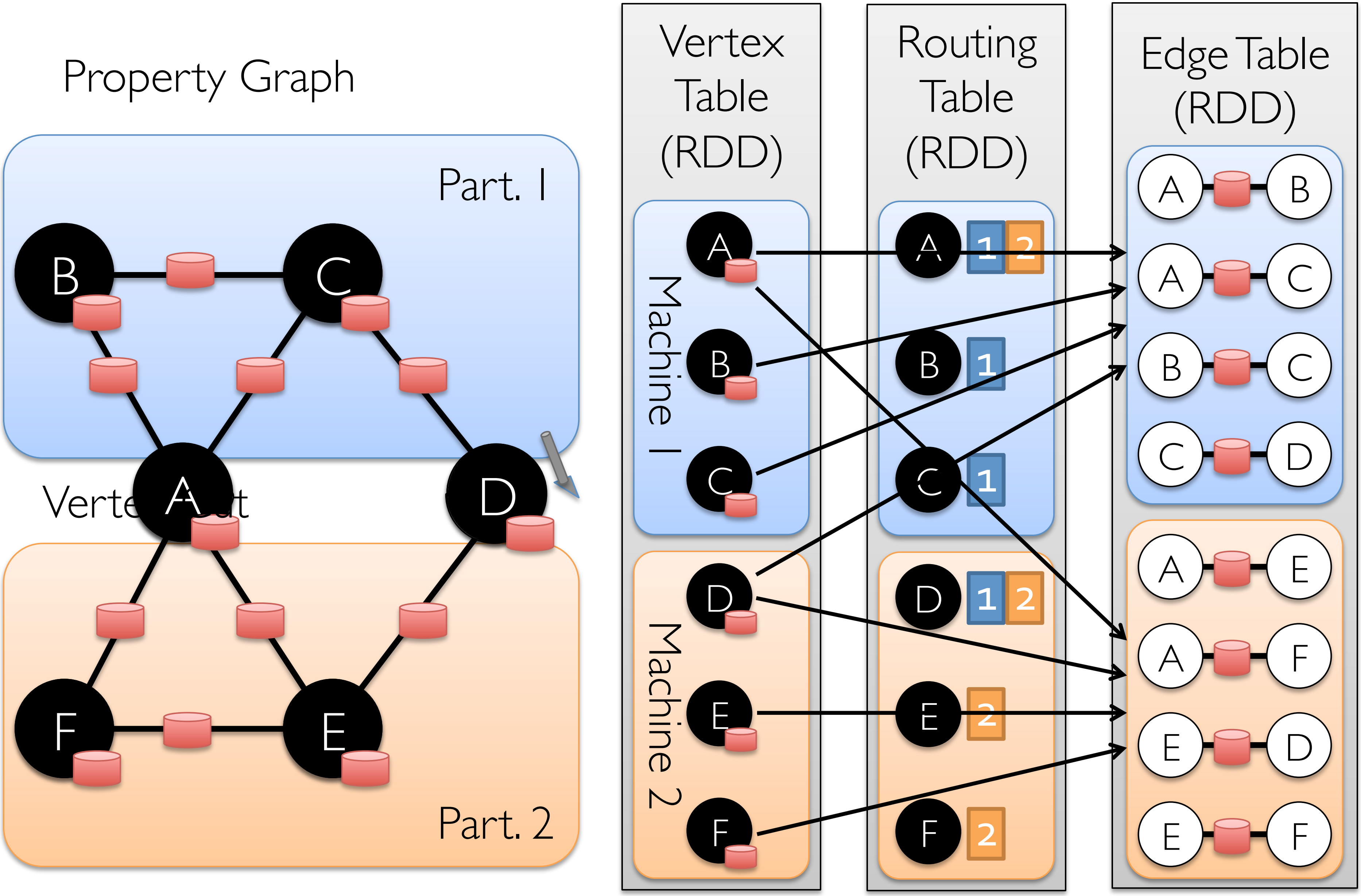
`mapReduceTriplets` →

RDD

vertex id	degree
Alice	2
Bob	2
Charlie	3
David	1

How GraphX Works

Encoding Property Graphs as RDDs



Property Graph

Part. 1

Vertex Cut

Part. 2

Vertex Table (RDD)

Machine 1

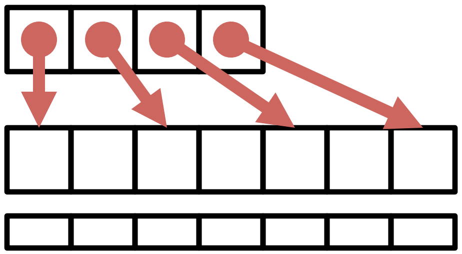
Machine 2

Routing Table (RDD)

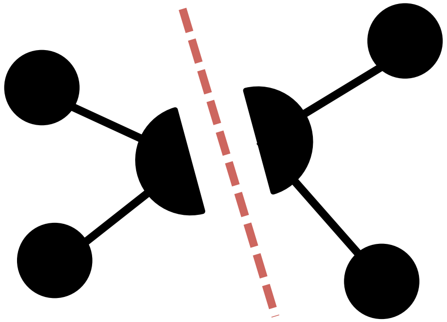
Edge Table (RDD)

Graph System Optimizations

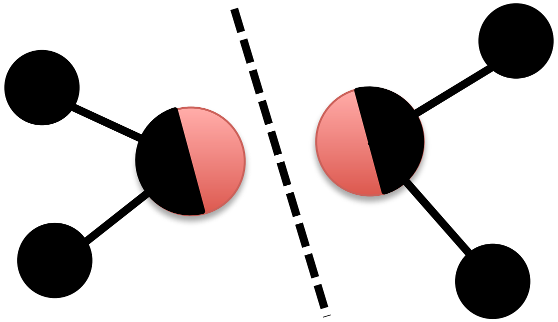
Specialized
Data-Structures



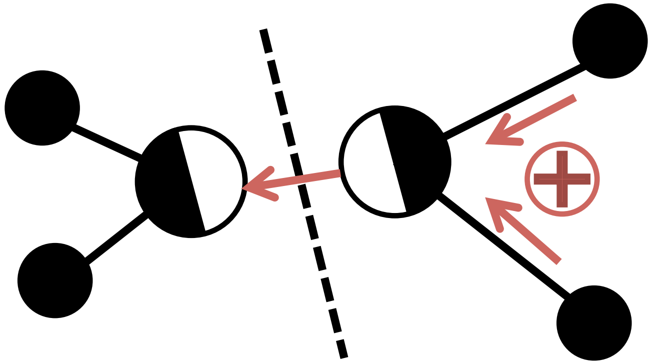
Vertex-Cuts
Partitioning



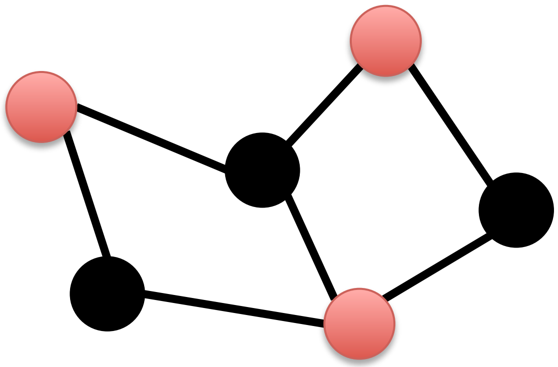
Remote
Caching / Mirroring



Message Combiners

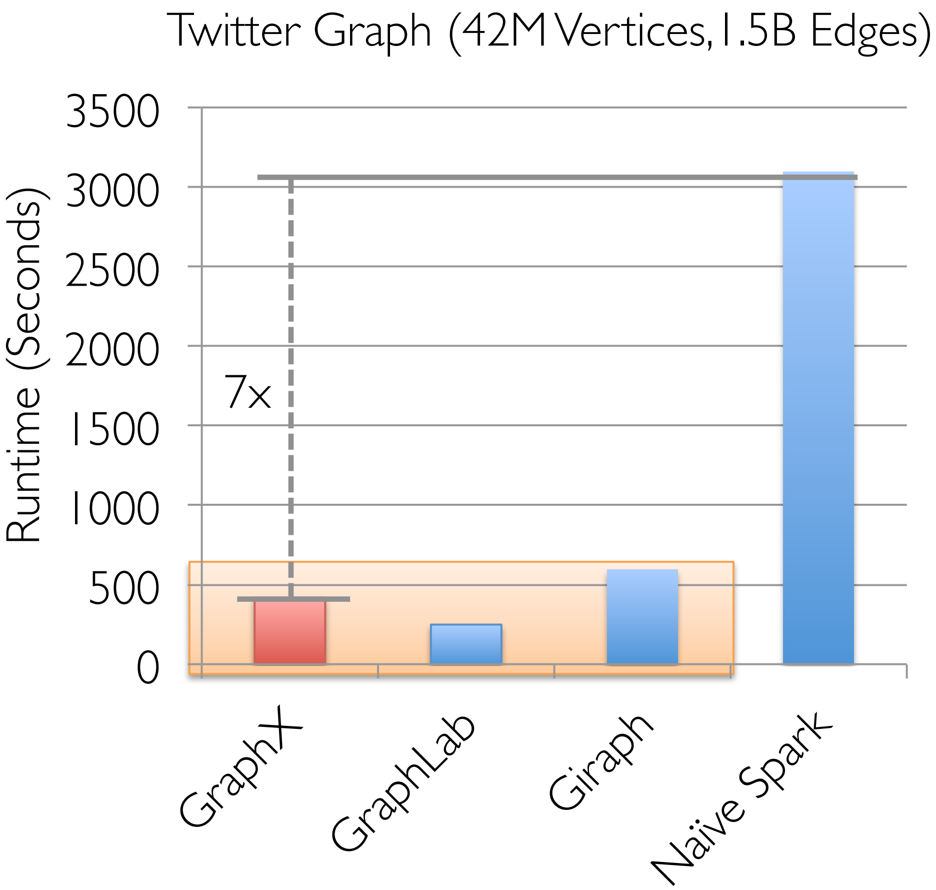


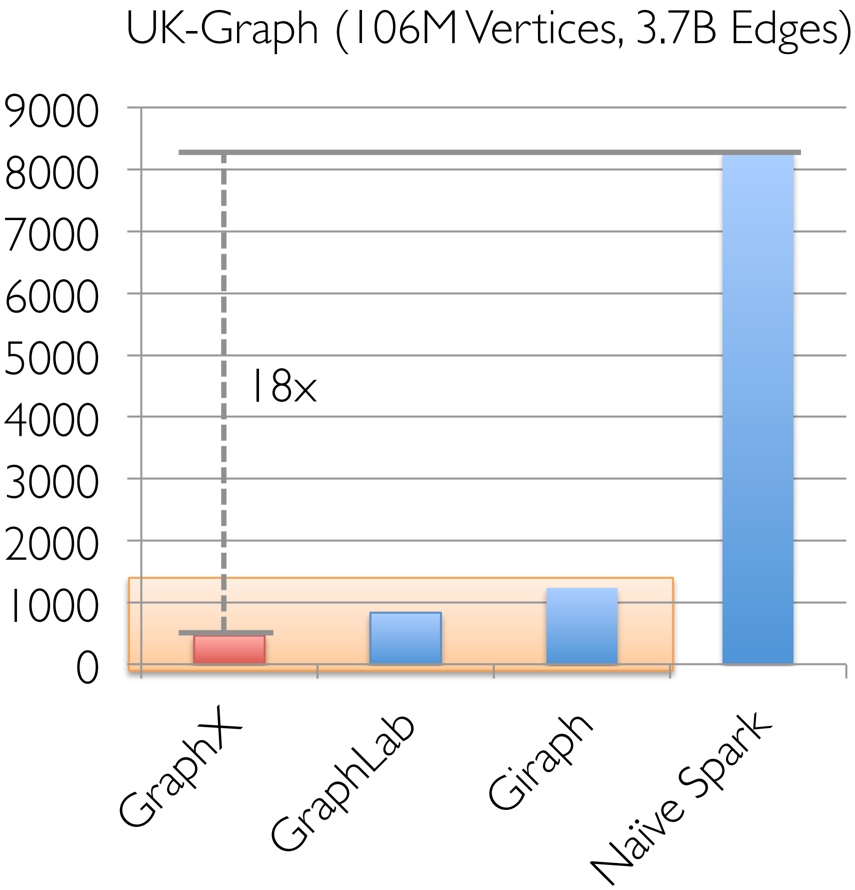
Active Set Tracking



PageRank Benchmark

EC2 Cluster of 16 x m2.4xLarge (8 cores) + 1GigE





GraphX performs comparably to state-of-the-art graph processing systems.

Future of GraphX

1. Language support
   a) Java API: PR #3234
   b) Python API: collaborating with Intel, SPARK-3789

2. More algorithms
   a) LDA (topic modeling): PR #2388
   b) Correlation clustering
   c) Your algorithm here?

3. Speculative
   a) Streaming/time-varying graphs
   b) Graph database–like queries

Thanks!

http://spark.apache.org/graphx

ankurd@eecs.berkeley.edu

jegonzal@eecs.berkeley.edu
rxin@eecs.berkeley.edu
crankshaw@eecs.berkeley.edu