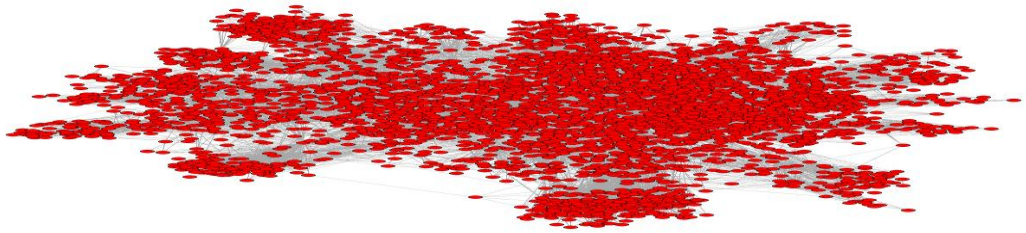
HelP: High-level Primitives for Large-Scale Graph Processing

Semih Salihoglu — Stanford University

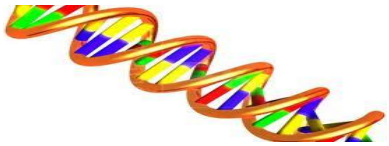
Jennifer Widom — Stanford University

Large-scale Graph Processing

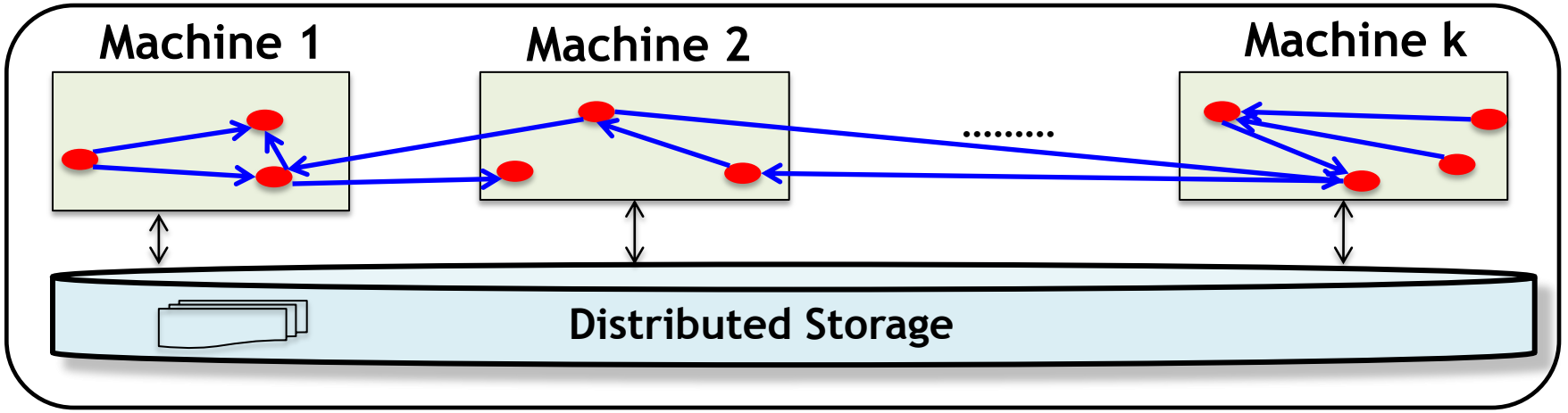
- 10s or 100s billion vertices and edges



Google facebook twitter



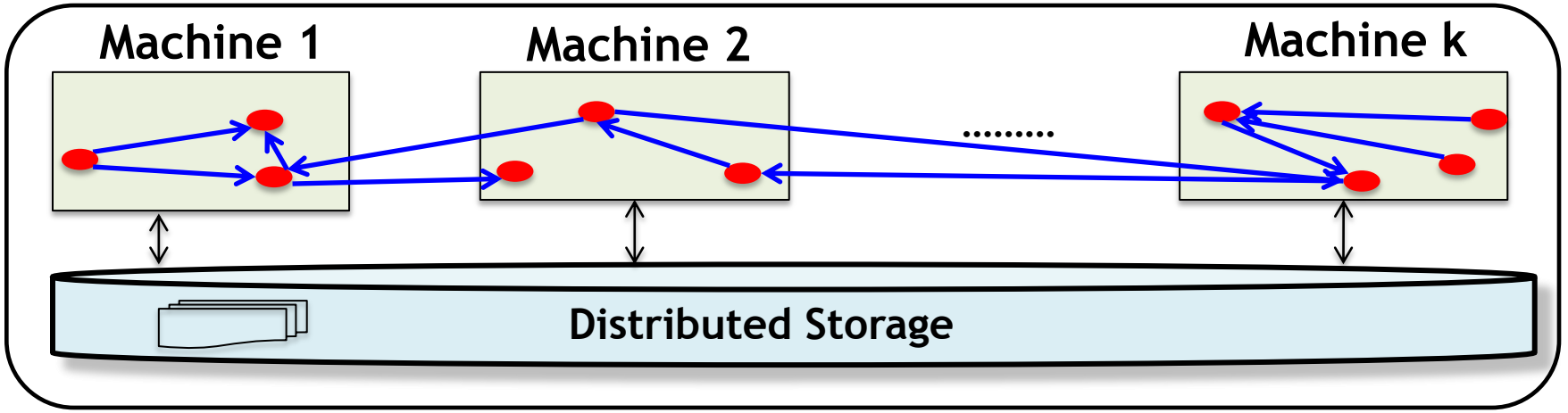
- Distributed Shared-Nothing Systems



Pregel APACHE GIRAPH PowerGraph GraphX

APIs of Existing Systems

- Specialized map() and reduce() type APIs
  - Pregel's compute()
  - PowerGraph's gather(), apply(), scatter()
- Vertex-centric/Graph-parallel
- Message-passing



Advantages

- Transparent parallelism
- Flexible. Can express many graph algorithms:

PageRank	HITS
Shortest Paths	Collaborative Filtering
Affinity Propagation	Loopy Belief Propagation
Weakly Connected Components	Triangle Counting
Strongly Connected Components	Betweenness-Centrality
Minimum Spanning Tree	Diameter Estimation
…	…

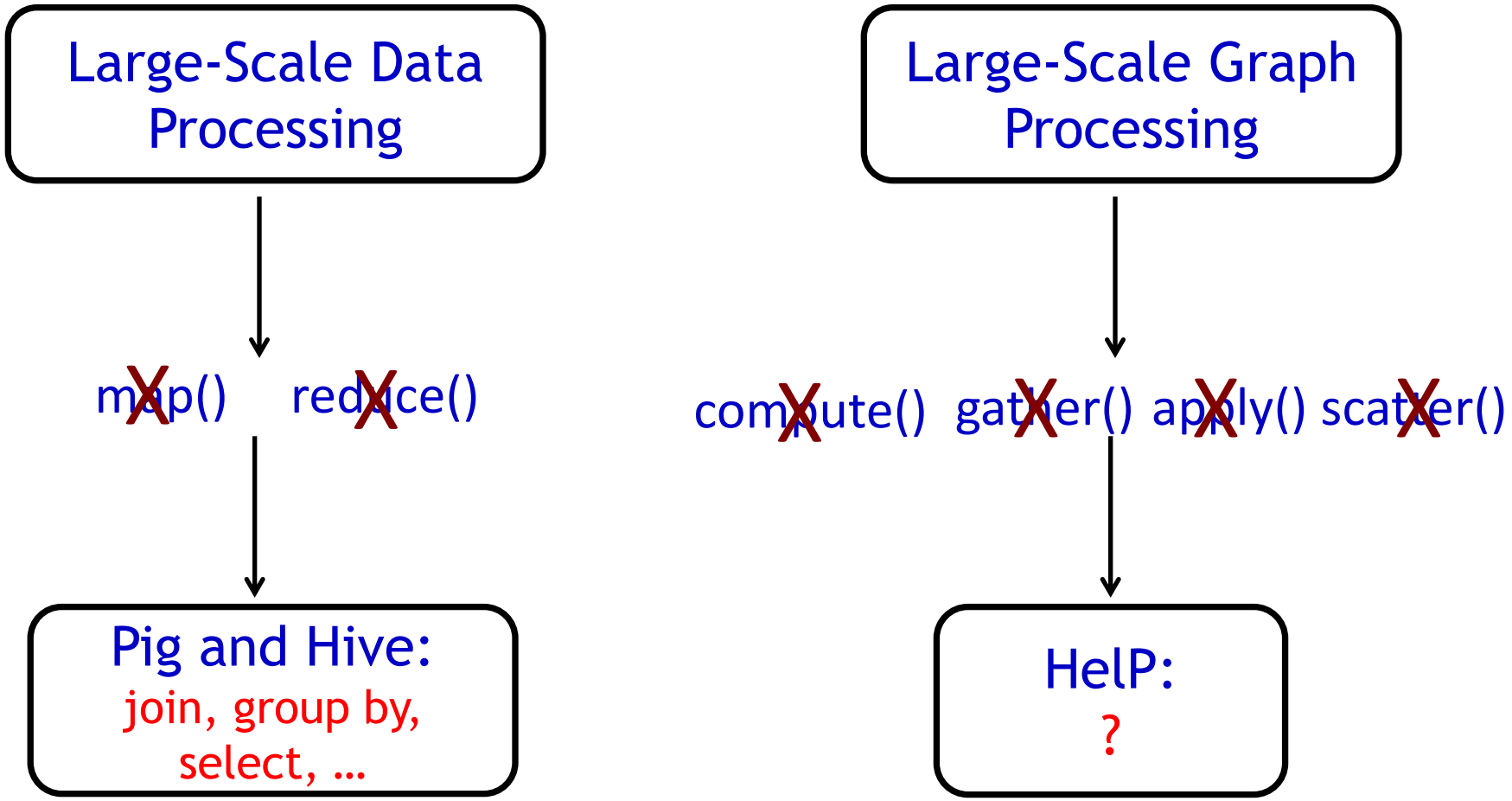
Disadvantages

- Custom code for common operations, such as:
  - Initializing vertex values
  - Aggregating neighbor values
- Difficult to read and understand some programs:
  - Complex UDFs hide higher-level graph operations

```
…
graph = Pregel.compute(UDF1)
graph = Pregel.compute(UDF2)
graph = Pregel.compute(UDF3)
…
```

- Too low-level for some operations
  - E.g: forming super vertices in a minimum spanning tree
    - ➢ Multiple rounds of complex messaging inside compute()

HelP Primitives



Steps in Our Work

1. Implemented a wide suite of distributed graph algorithms
2. Identified the commonly appearing operations
3. Abstracted the operations into *HelP* primitives
4. Implemented HelP on GraphX
5. Reimplemented the suite of algorithms on GraphX

Graph Algorithms We Implemented

Algorithm

- PageRank
- HITS
- Conductance
- Approx. Betweenness Centrality
- Clustering Coefficient
- Semi-clustering
- Multi-level clustering
- Approx. Maximum Weight Matching
- Random Bipartite Matching
- Weakly Connected Components
- Strongly Connected Components
- Single Source Shortest Paths
- Graph Coloring
- Maximal Independent Set
- K-core
- Triangle Counting
- Diameter Estimation
- K-truss
- Minimum Spanning Forest

HelP Primitives

Primitive	Type of Operation
Aggregate Neighbor Values (ANV)	Vertex-centric Update
Local Update of Vertices (LUV)	Vertex-centric Update
Update Vertices Using One Other Vertex (UVUOV)	Vertex-centric Update
Filter	Topology Modification
Form Supervertices (FS)	Topology Modification
Aggregate Global Value (AGV)	Global Aggregation

Algorithms & HelP Primitives

Algorithm	Filter	ANV	LUV	UVUOV	FS	AGV
PageRank		x	x			
HITS		x	x			x
Conductance		x				x
Approx. Betweenness Centrality		x	x			x
Clustering Coefficient	x	x				
Semi-clustering		x	x			x
Multi-level clustering	x			x	x	
Approx. Maximum Weight Matching	x			x		
Random Bipartite Matching	x		x	x		
Weakly Connected Components		x	x			
Strongly Connected Components	x	x	x			x
Single Source Shortest Paths		x	x			
Graph Coloring	x	x	x			
Maximal Independent Set	x	x	x			
K-core	x					x
Triangle Counting		x				
Diameter Estimation		x	x			x
K-truss		x				
Minimum Spanning Forest	x		x	x	x	

Example: Aggregate Neighbor Values

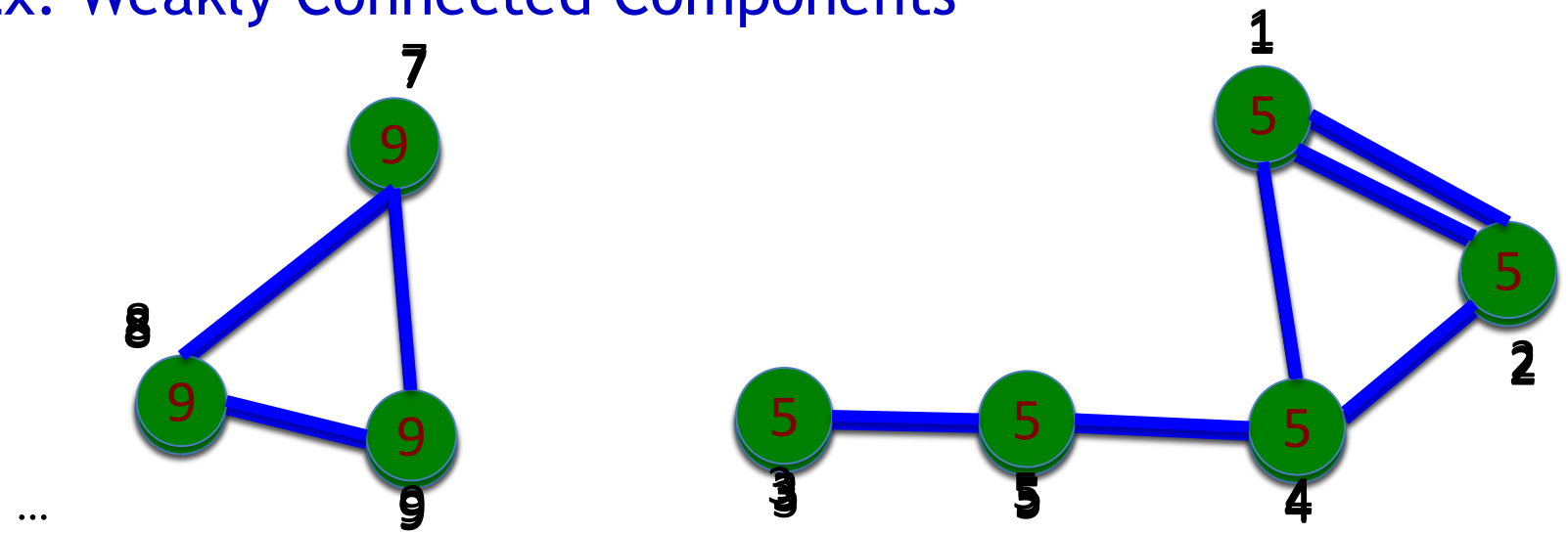
- Vertices aggregate some or all of their neighbors' values
- Update own value with the aggregated value
- Version 1: Non-iterative => *aggregateNeighborValues*

PageRank

```
…
for (i=0; i < 10; ++i) {
  g.aggregateNeighborValues(
    v -> true /* aggregate all vertices */,
    nbr -> true /* which neighbors to aggregate */,
    nbr -> nbr.val.pr/nbr.degree,
    AggrFnc.SUM,
    (v, sumPr)->{v.val.pr = 0.85*sumPr + 0.15/g.numV;})
}
```

Version 2: Iterative => propagateAndAggregate

- Continue aggregations until vertex values converge
- Ex: Weakly Connected Components



…

```
g.propagateAndAggregate(
    EdgeDirection.BOTH,
    v -> true, /* start propagation from all */
    v -> v.val.wccID,
    AggrFnc.MAX,
    (v, aggrWCCID) -> {v.val.wccID = aggrWCCID;})
```

Related Work (see paper)

- Vertex-centric APIs
- MapReduce-based APIs
- Higher-Level Data Analysis Languages
- Domain-Specific Graph Languages
- MPI-based Libraries

GraphX Implementation, Limitations, Future Work

See Our Paper
& Poster!

Questions?

GraphX Implementation (Non-iterative Version)