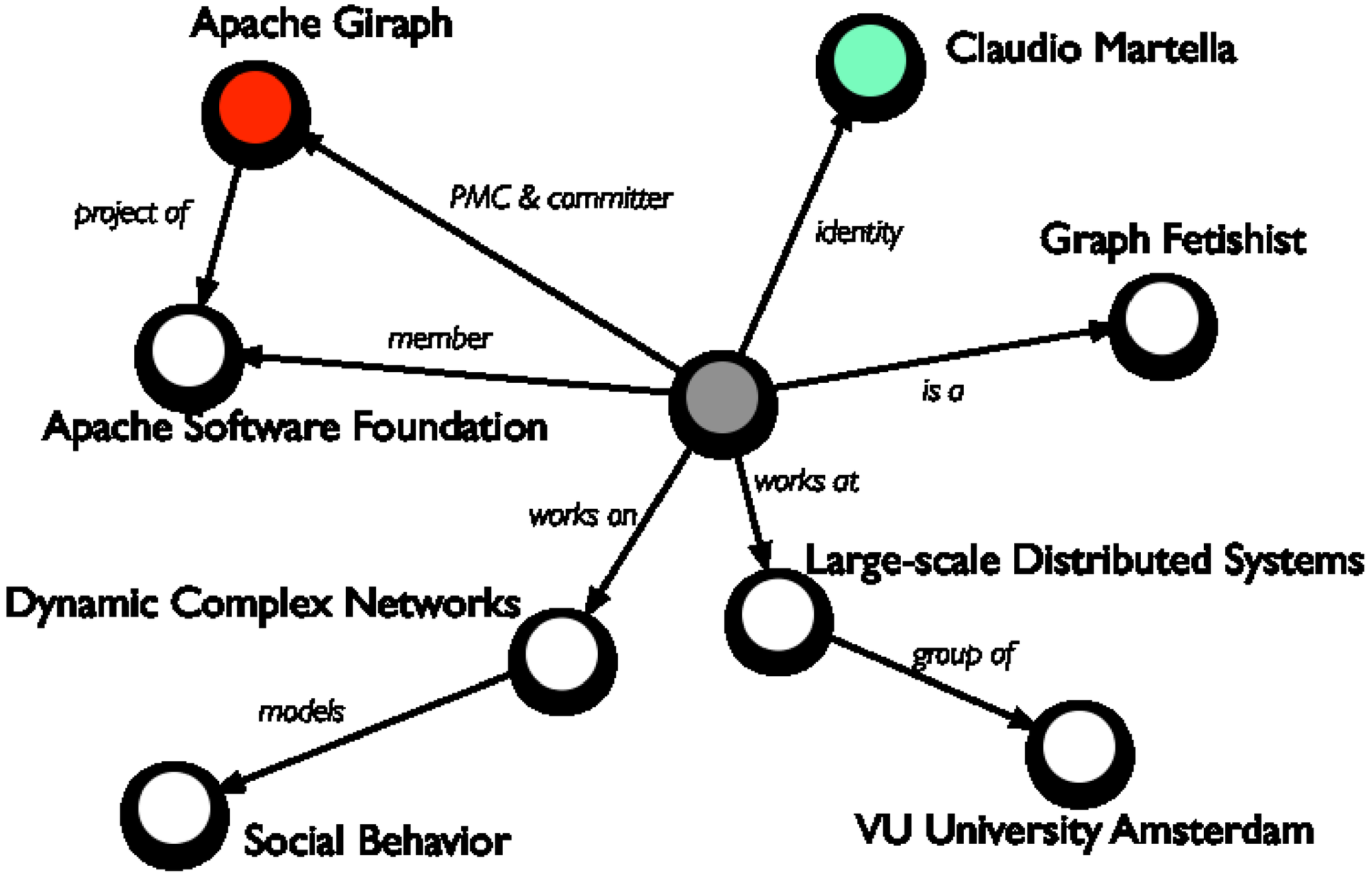
Apache Giraph

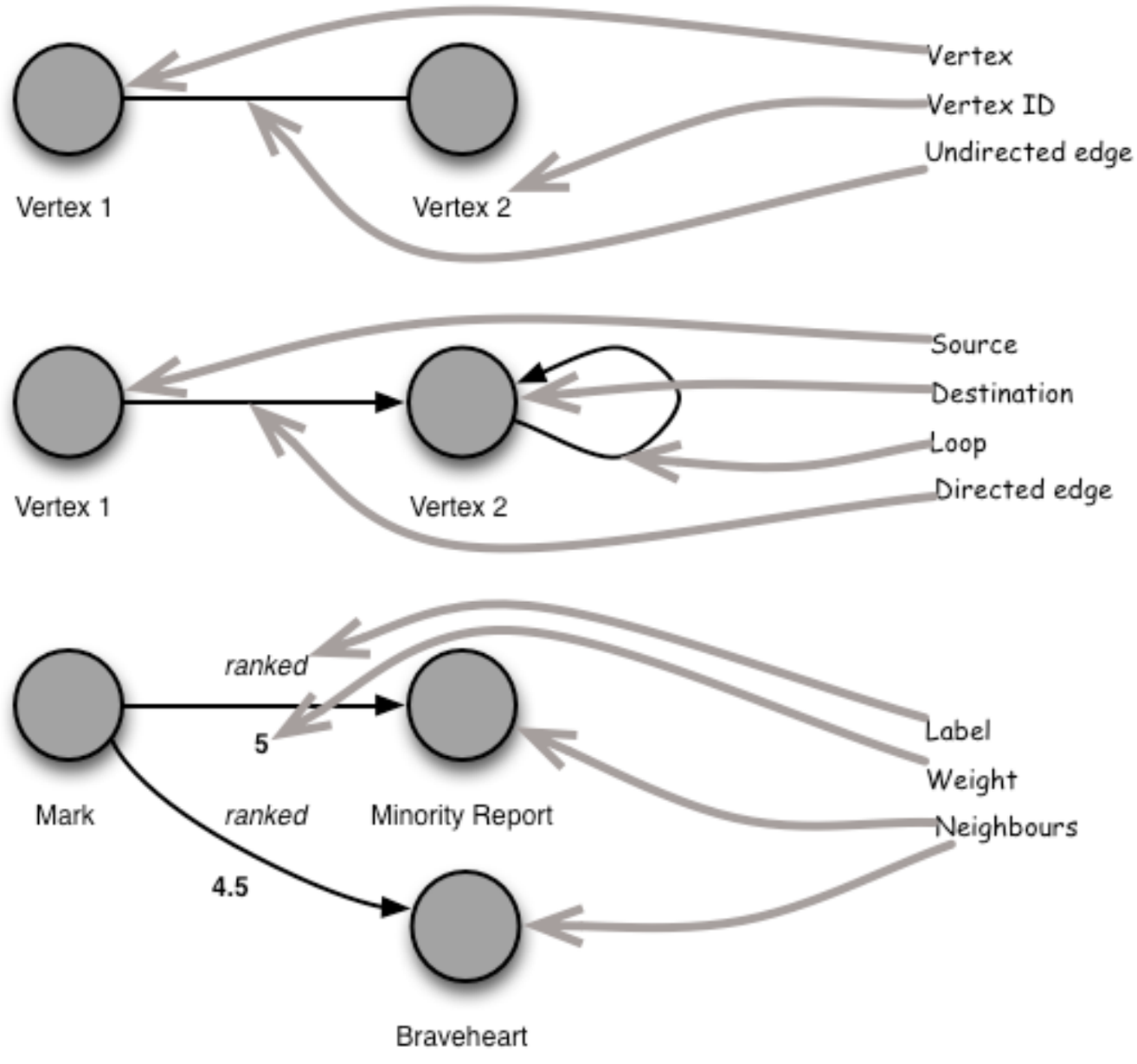
Large-scale Graph Processing on Hadoop

Claudio Martella <claudio@apache.org>
@claudiomartella

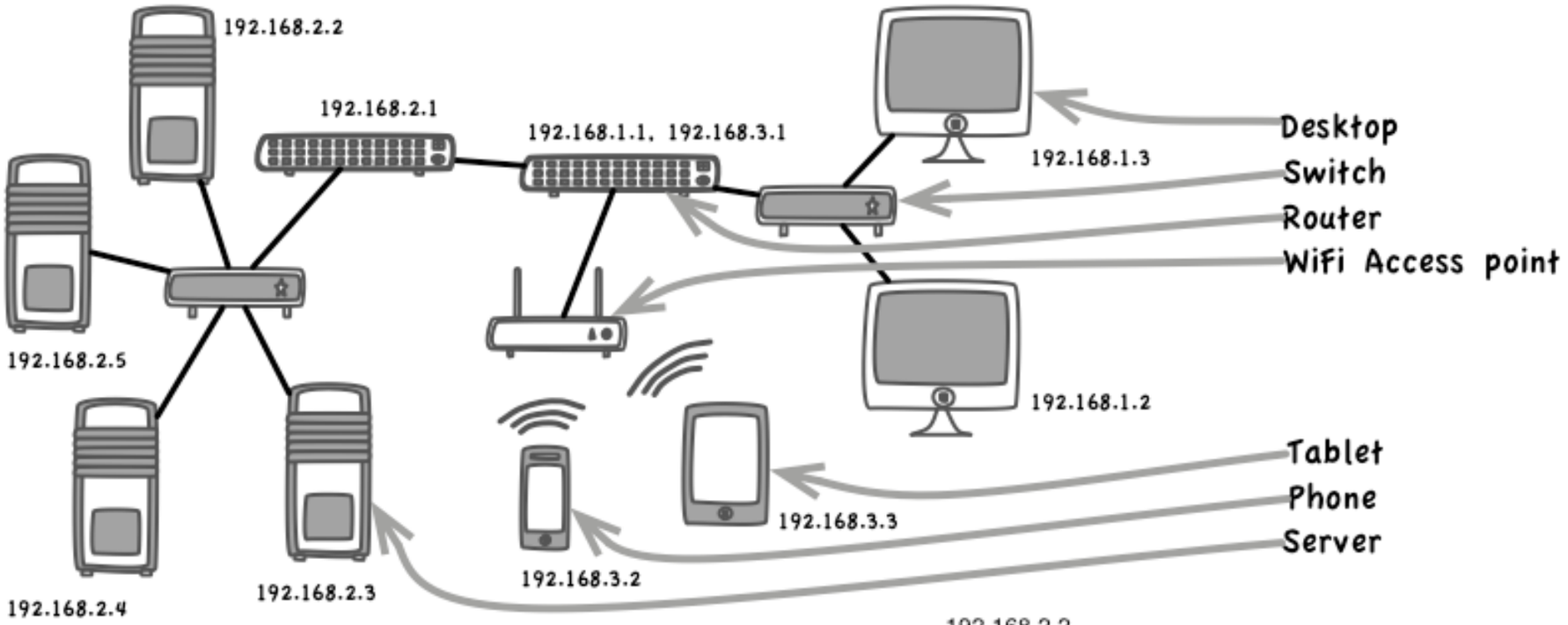
Hadoop Summit @ Amsterdam - 3 April 2014


Apache Giraph
Claudio Martella
project of
PMC & committer
identity
Graph Fetishist
member
is a
Apache Software Foundation
works on
works at
Large-scale Distributed Systems
Dynamic Complex Networks
group of
models
Social Behavior
VU University Amsterdam

Graphs are simple

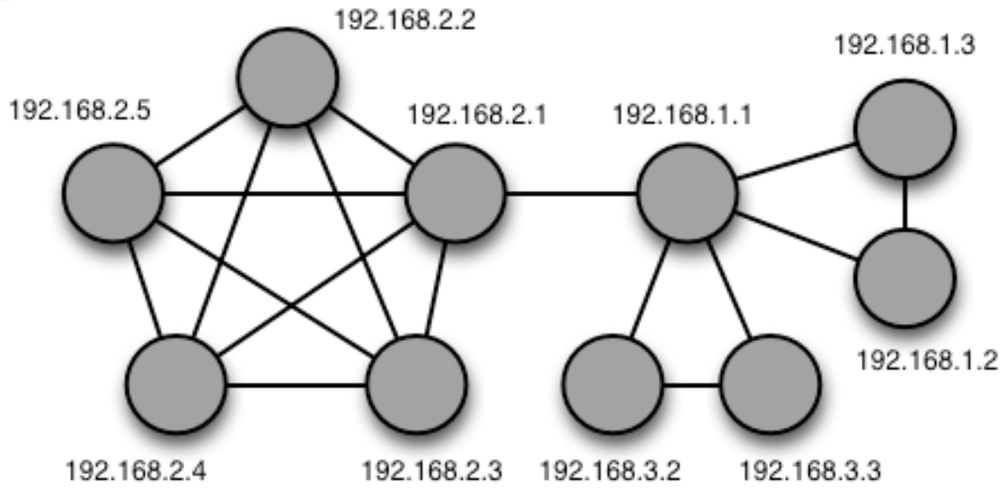


A computer network

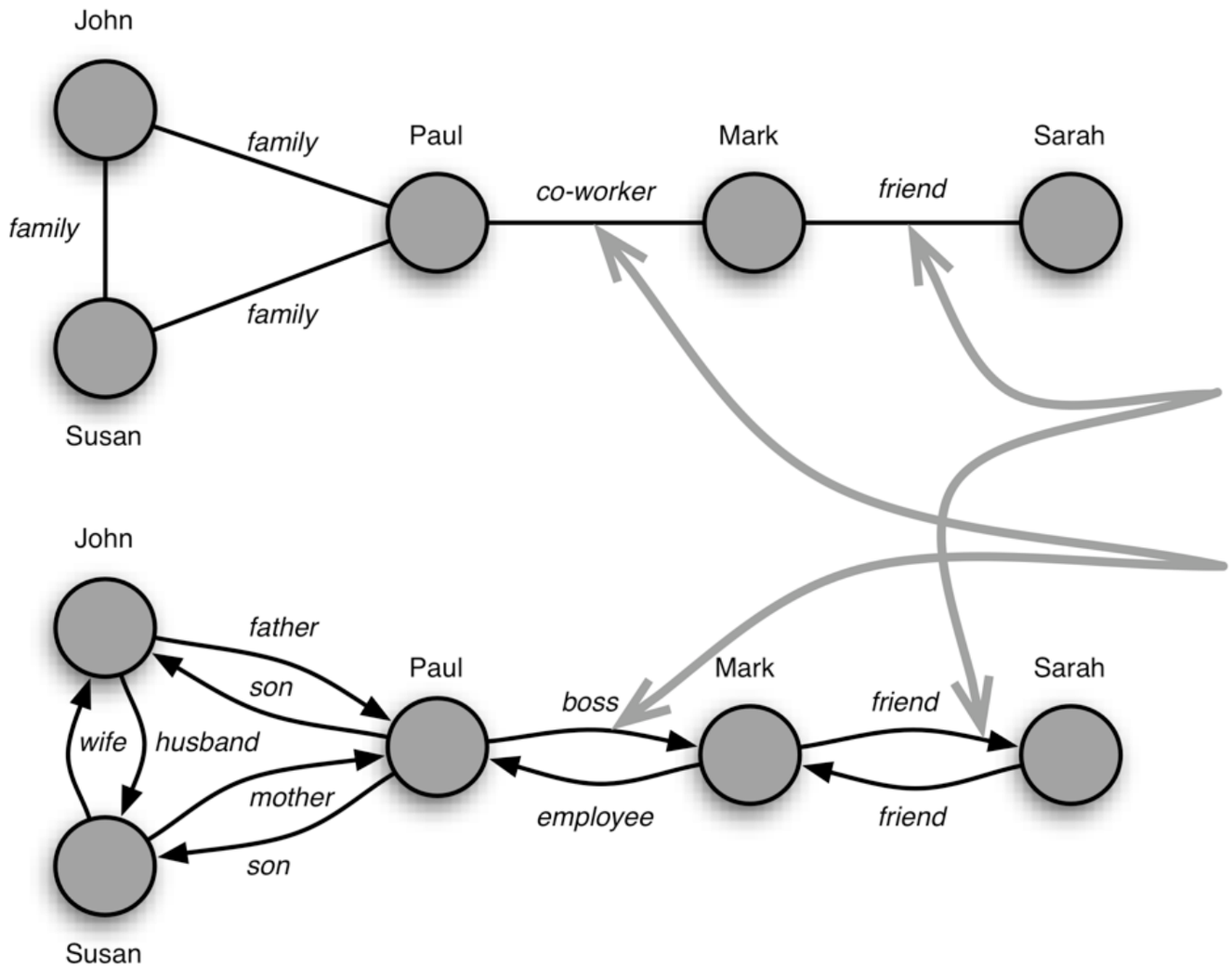


Note

1. There are three networks: servers, desktops and mobile.
2. They are connected through two routers/firewalls.
3. We ignored the switches and the access point in the graph.
4. Router 192.168.1.1 has two interfaces but we used one as vertex ID.



A social network

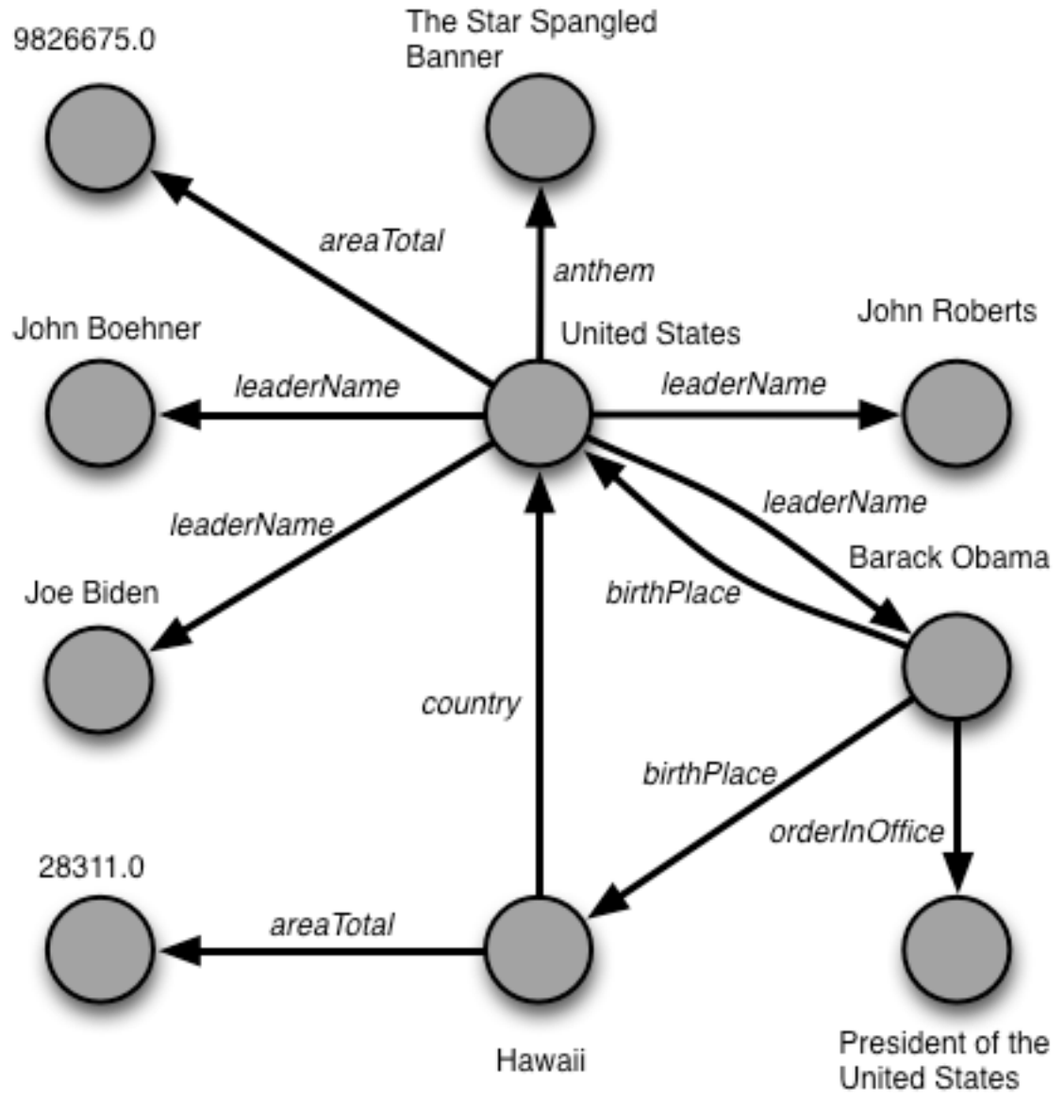


Note

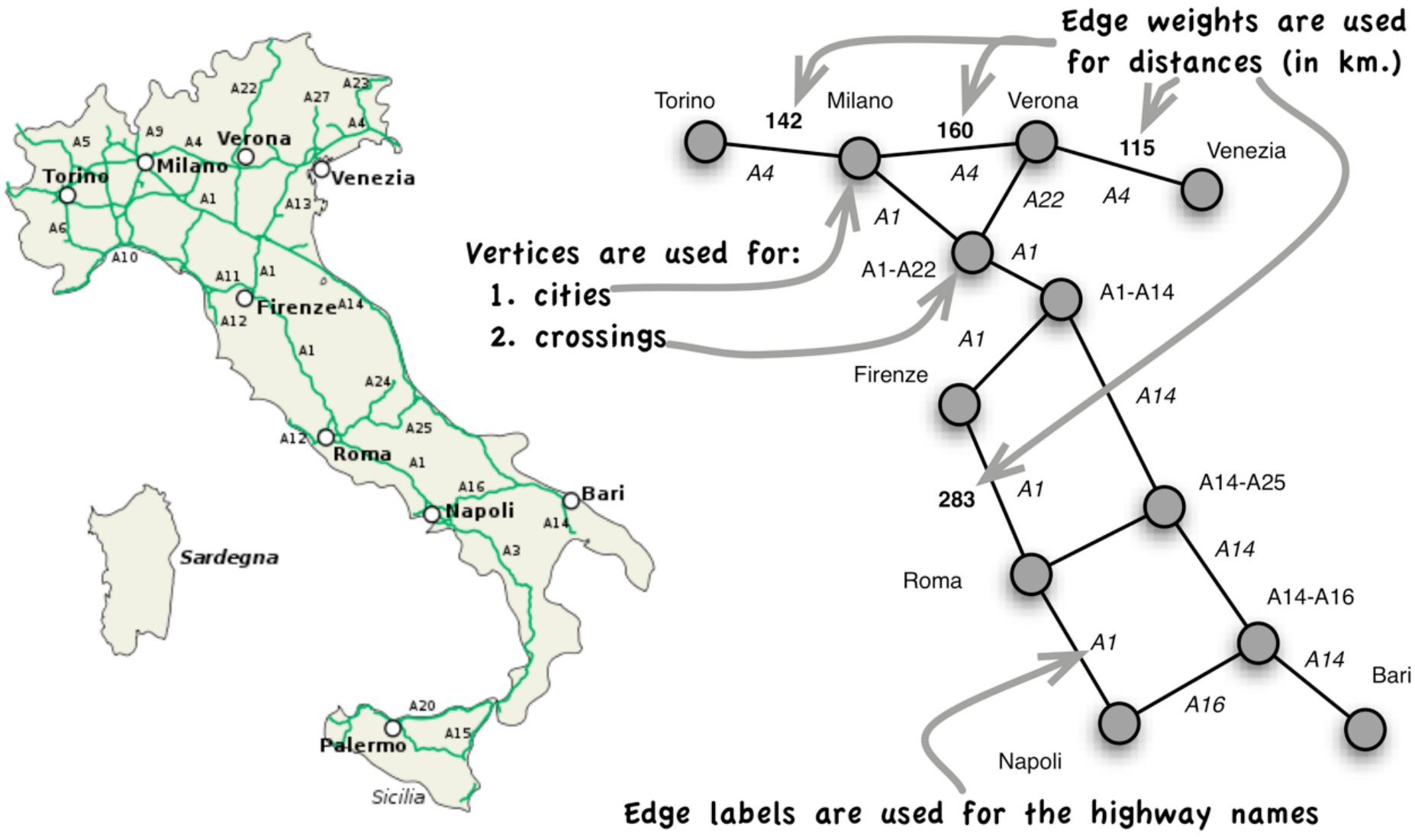
1. A symmetric relationship is substituted by two directed edges.
2. A relationship does not have to be substituted by two edges, but e.g. by a more specific one.

A semantic network

Subject	Predicate	Object
United States	areaTotal	9826675.0
United States	anthem	The Star Spangled Banner
United States	leaderName	Barack Obama
United States	leaderName	Joe Biden
United States	leaderName	John Boehner
United States	leaderName	John Roberts
Barack Obama	birthPlace	United States
Barack Obama	birthPlace	Hawaii
Barack Obama	orderInOffice	President of the United States
Hawaii	areaTotal	28311.0
Hawaii	country	United States



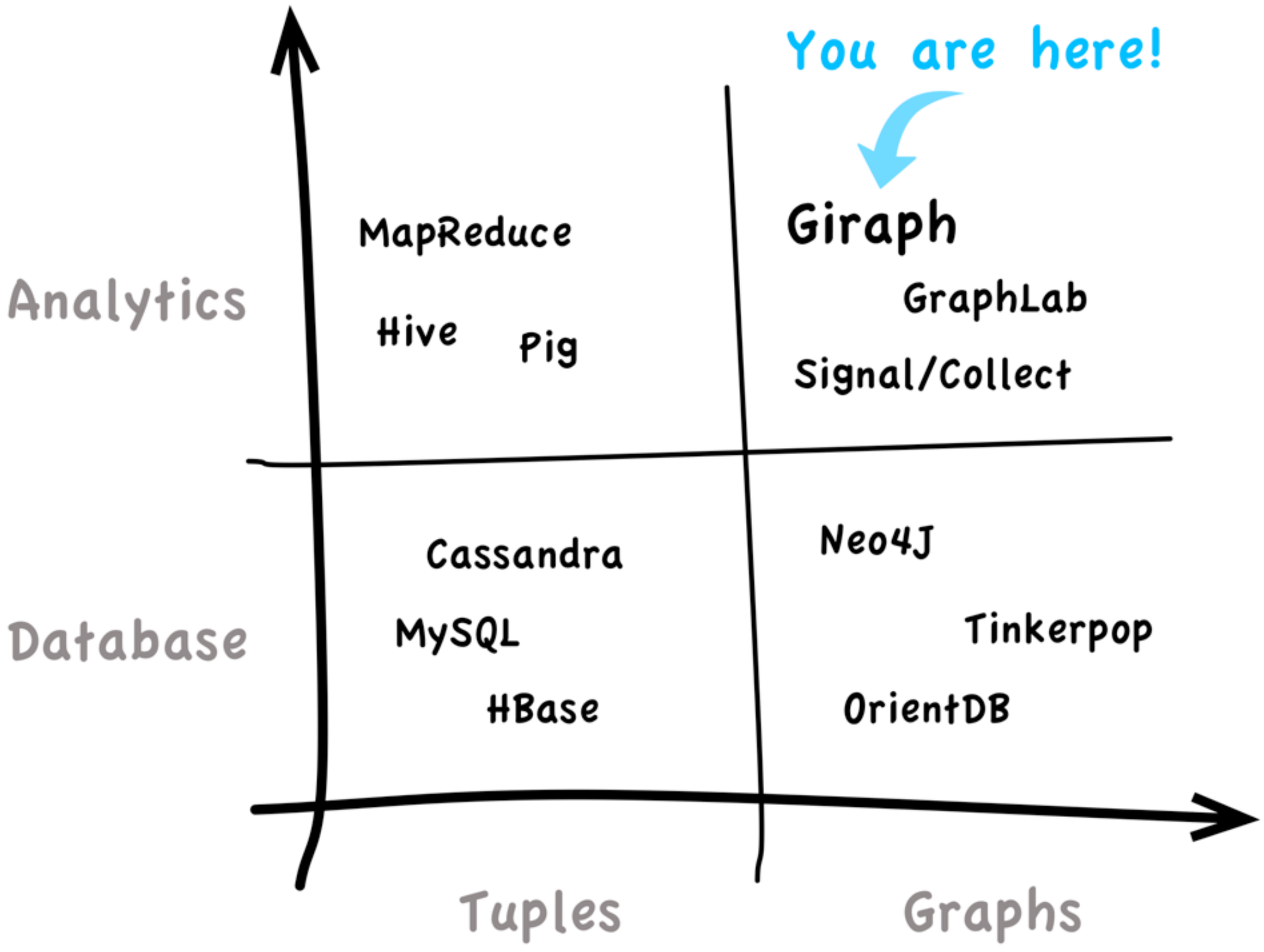
A map


Torino
Milano
Verona
Venezia
Firenze
Roma
Napoli
Bari
Palermo
Sardegna
Sicilia
A1
A4
A5
A6
A9
A10
A11
A12
A13
A14
A15
A16
A20
A22
A23
A24
A25
A27
A3
Edge weights are used for distances (in km.)
142
160
115
283
Vertices are used for:
1. cities
2. crossings
A1-A22
A1-A14
A14-A25
A14-A16
A1
A14
A16
A22
Edge labels are used for the highway names

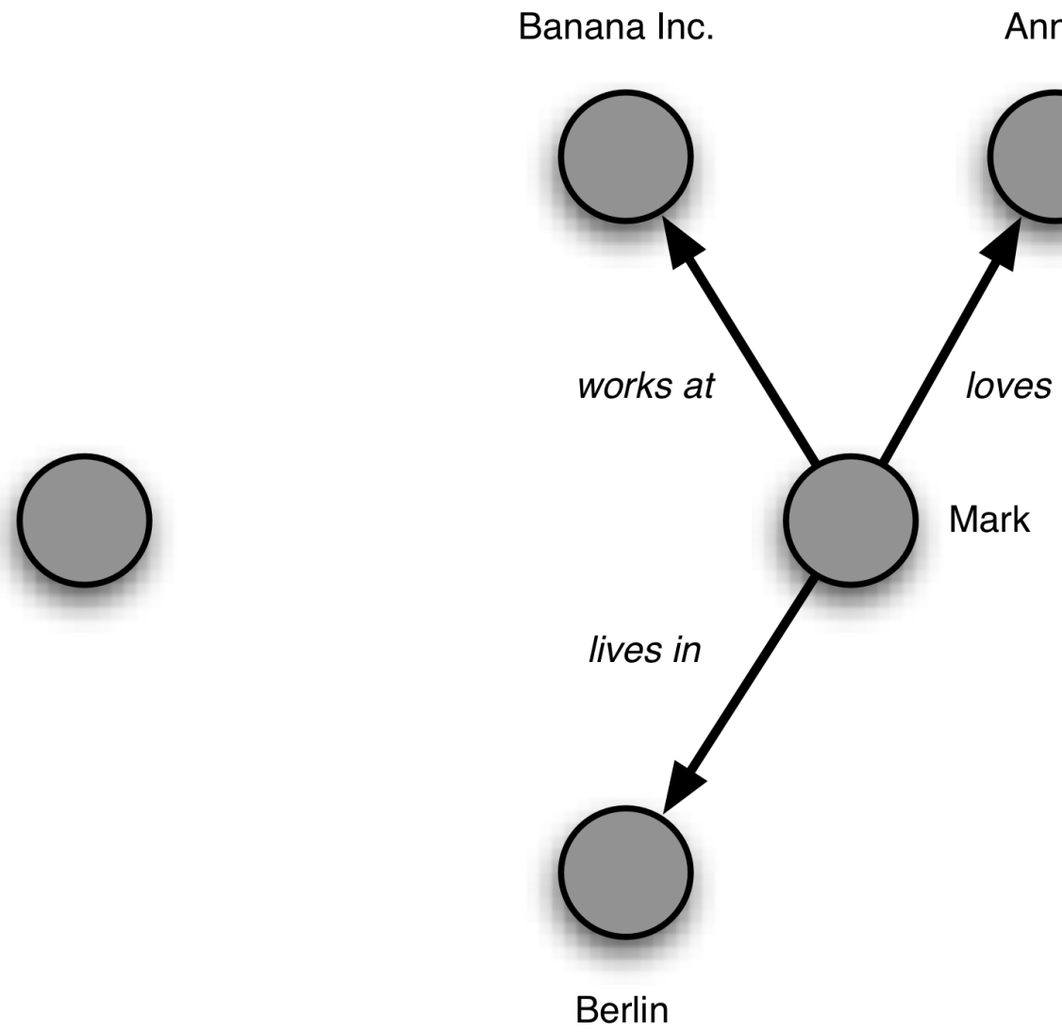
Graphs are huge

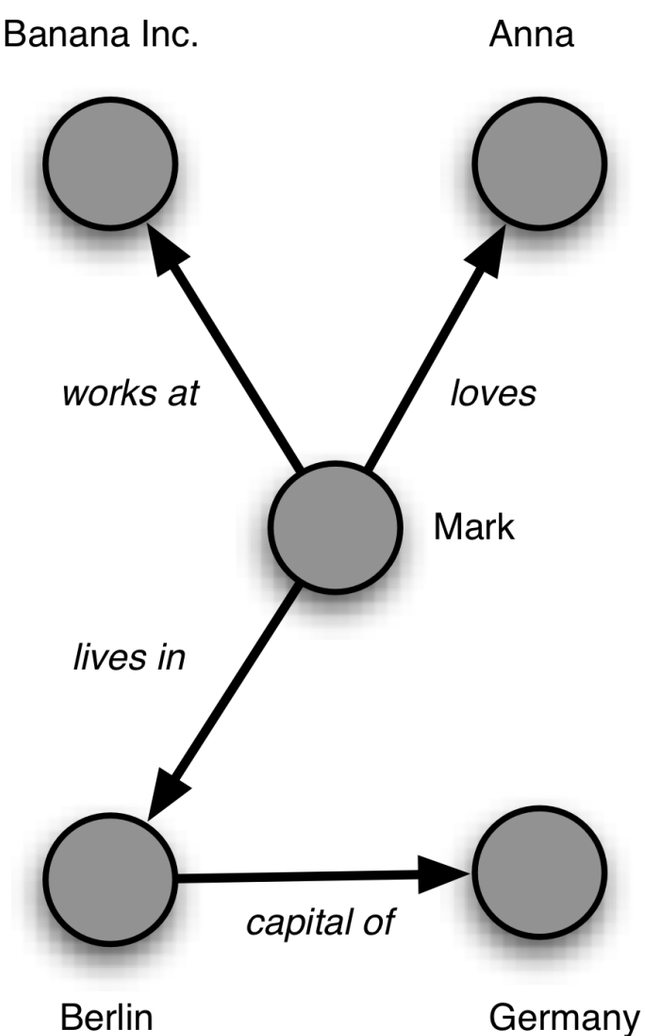
- Google's index contains 50B pages
- Facebook has around1.1B users
- Google+ has around 570M users
- Twitter has around 530M users

VERY rough
estimates!


You are here!
Analytics
MapReduce
Hive
Pig
Giraph
GraphLab
Signal/Collect
Database
Cassandra
MySQL
HBase
Neo4J
Tinkerpop
OrientDB
Tuples
Graphs

Graphs aren't easy





Graphs are nasty.

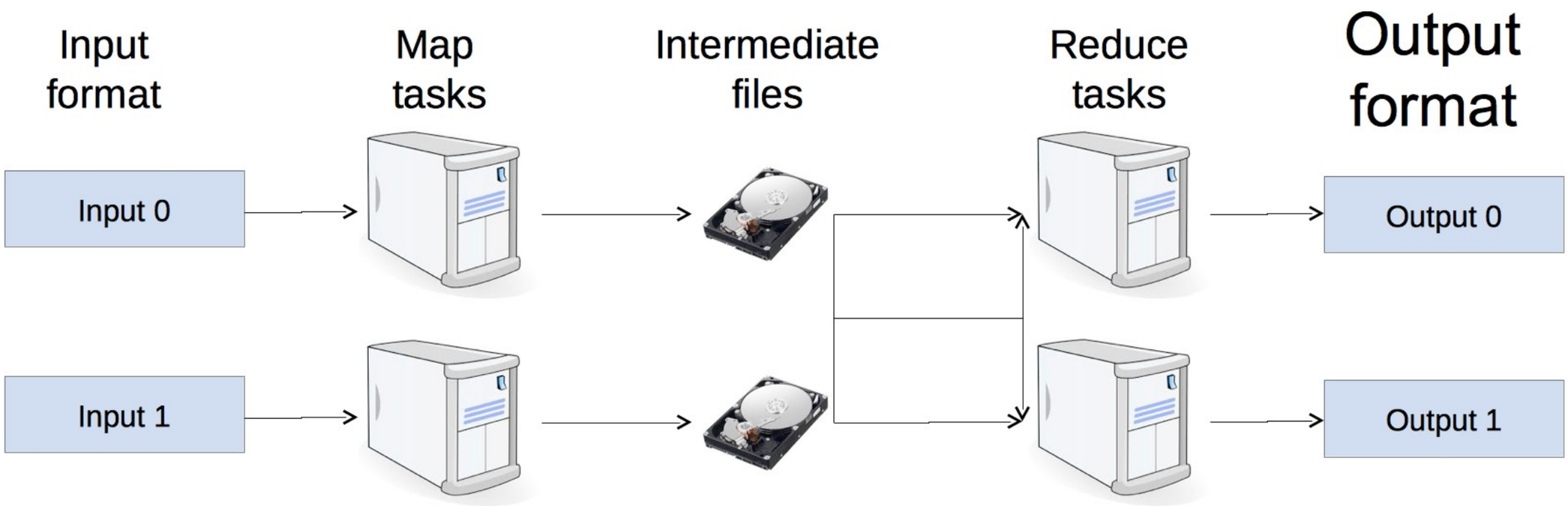
Each vertex depends on its neighbours, recursively.

Recursive problems are nicely solved iteratively.

PageRank in MapReduce

- Record: < v_i, pr, [v_j, ..., v_k] >
- Mapper: emits < v_j, pr / #neighbours >
- Reducer: sums the partial values

MapReduce dataflow



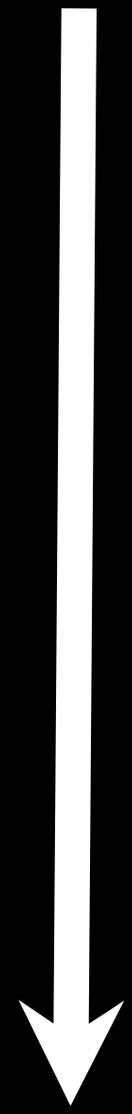
Drawbacks

- Each job is executed N times
- Job bootstrap
- Mappers send PR values and structure
- Extensive IO at input, shuffle & sort, output

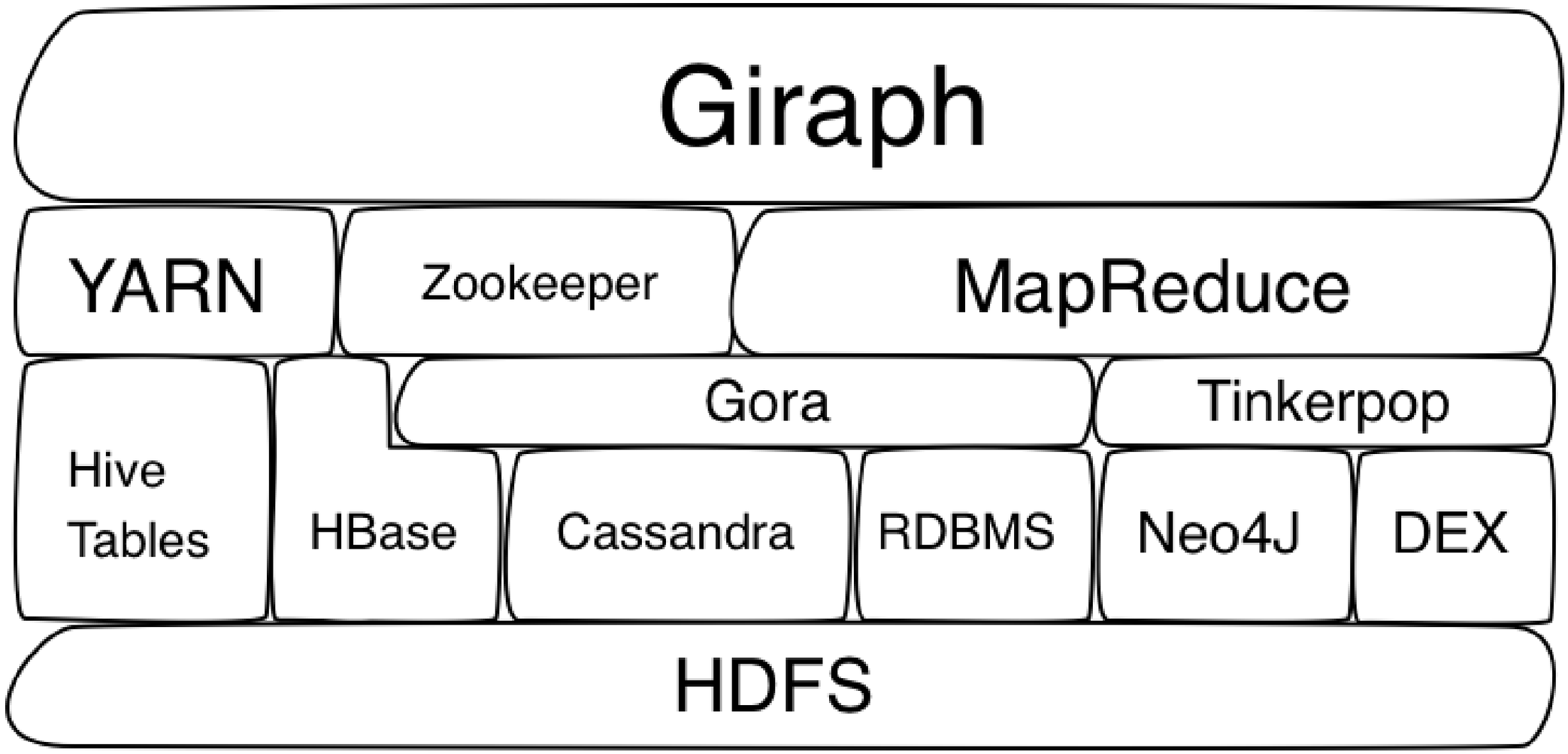

APACHE
GIRAPH

Timeline

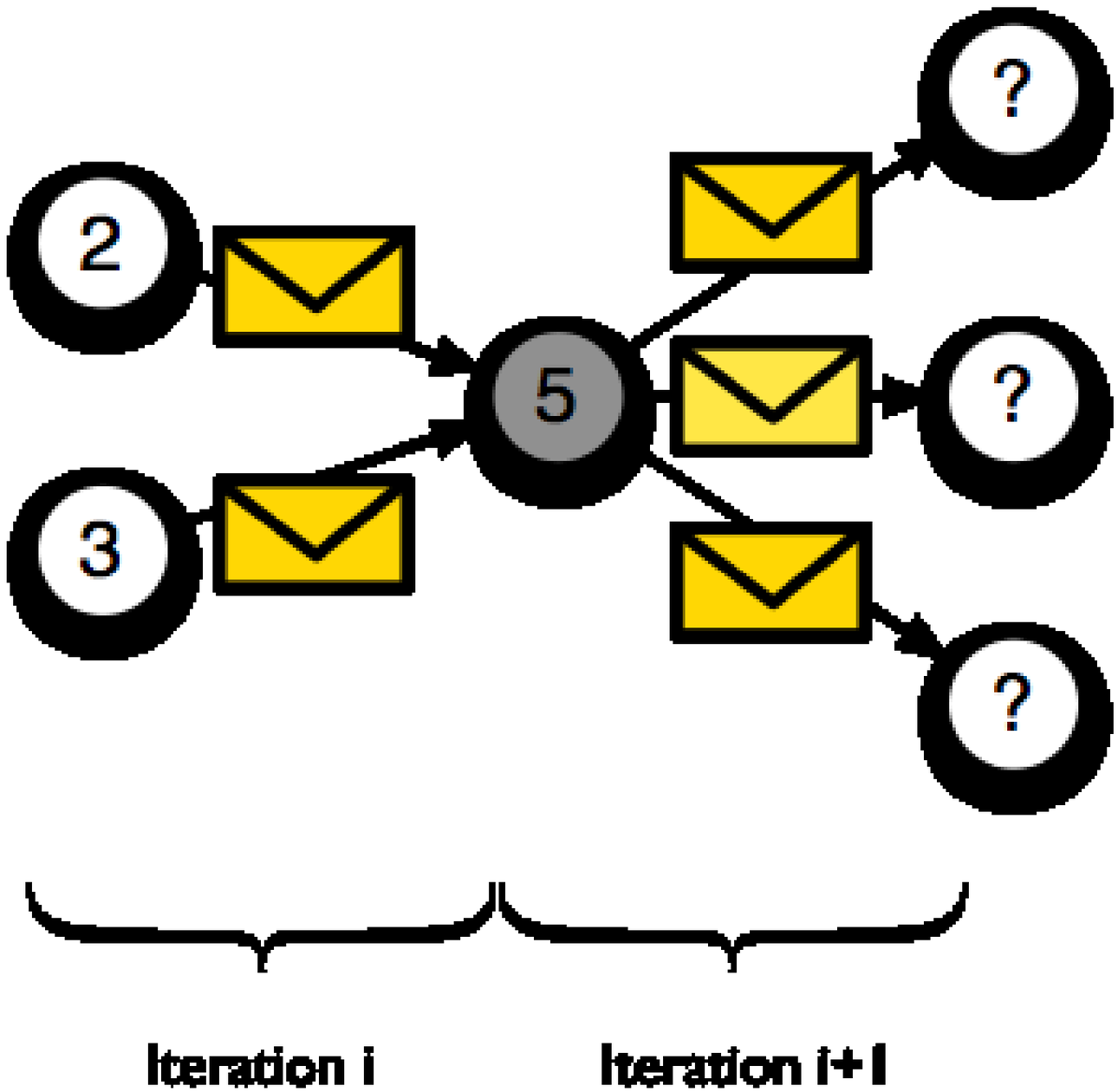
- Inspired by Google Pregel (2010)
- Donated to ASF by Yahoo! in 2011
- Top-level project in 2012
- 1.0 release in January 2013
- 1.1 release in days 2014



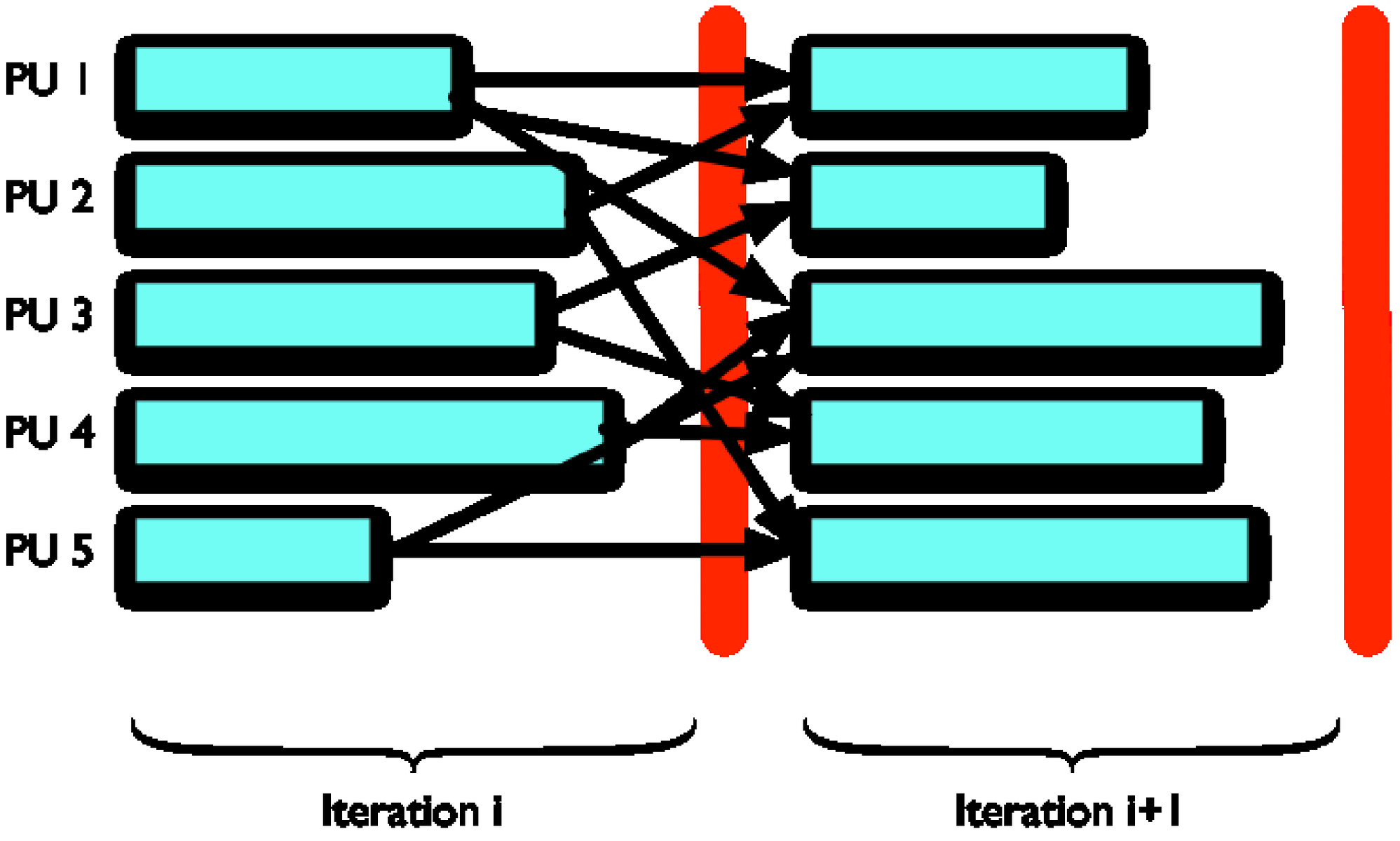
Plays well with Hadoop



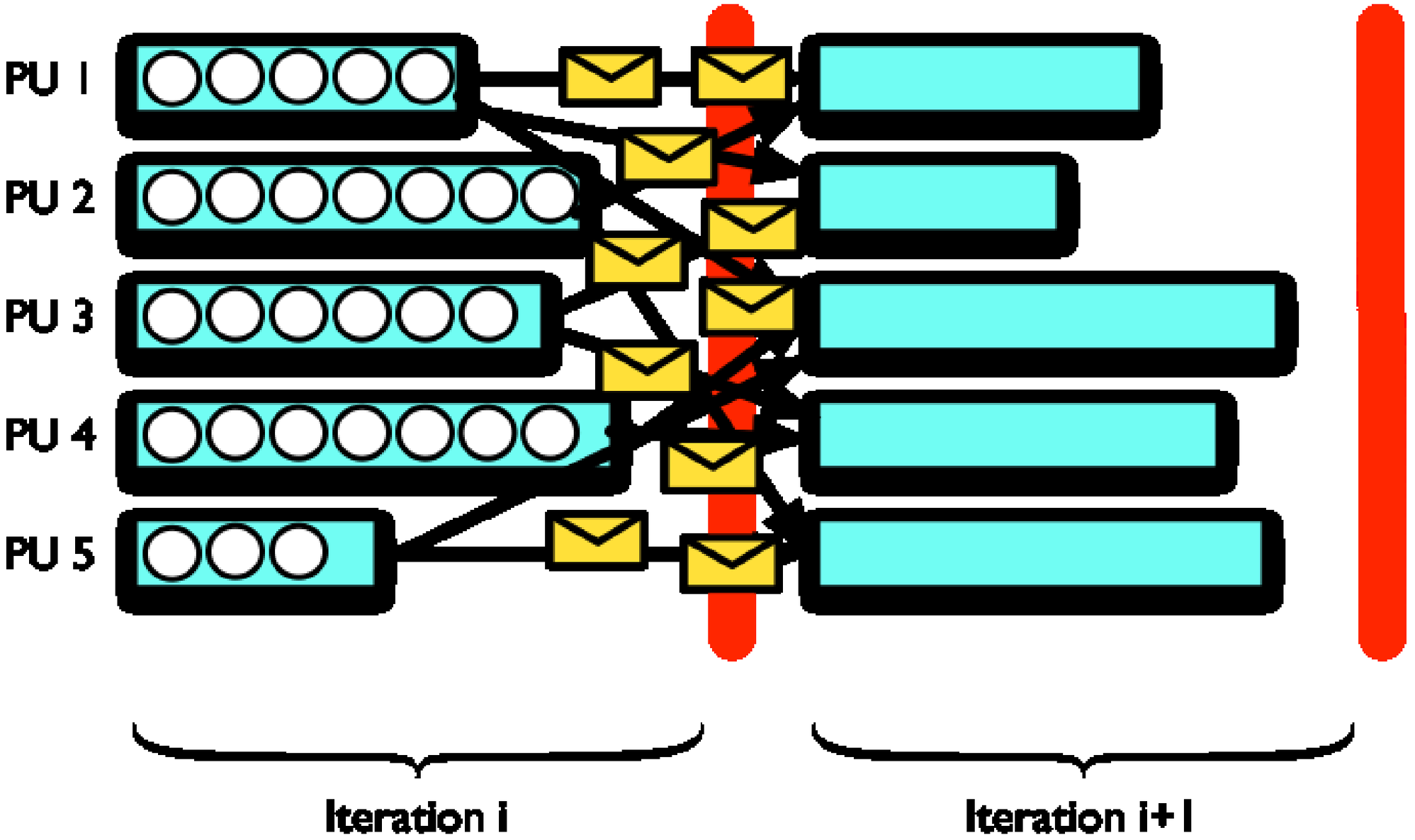
Vertex-centric API



BSP machine



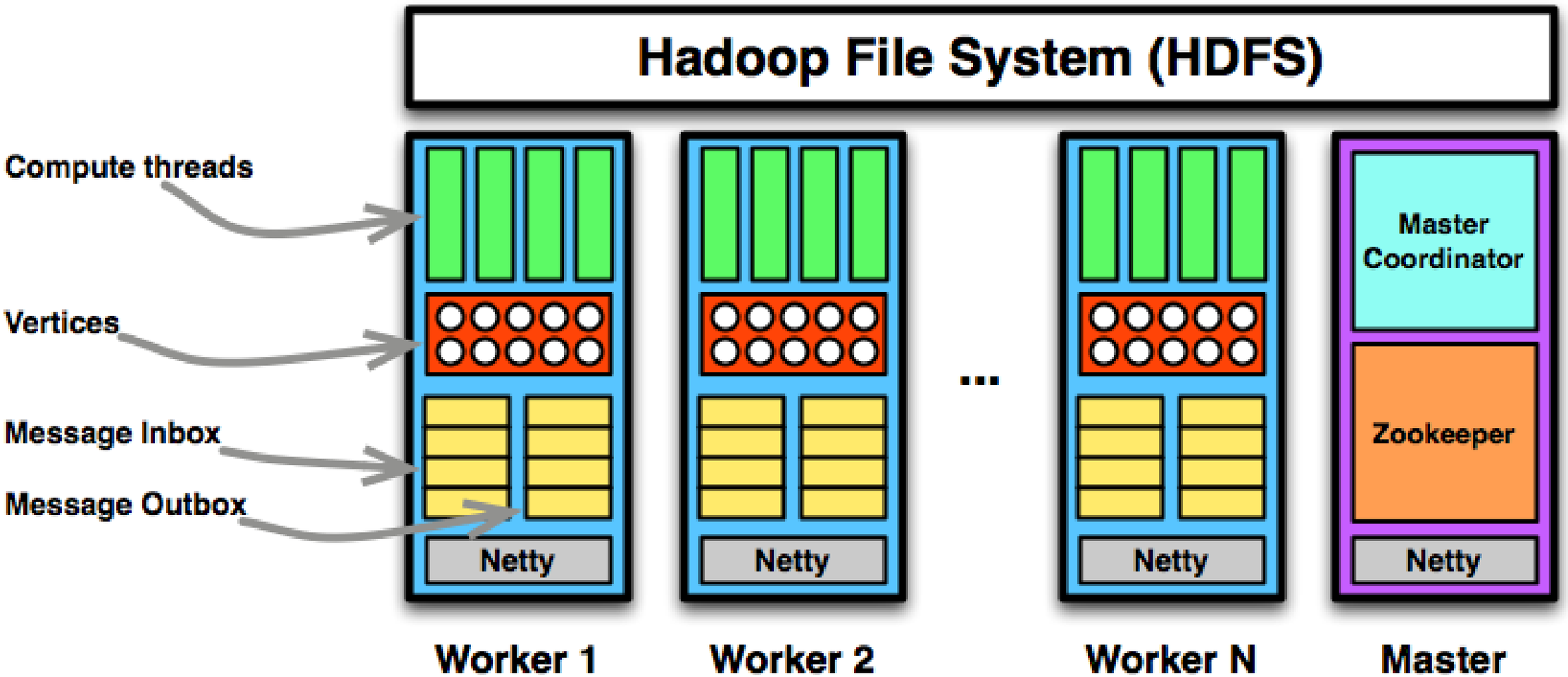
BSP & Giraph



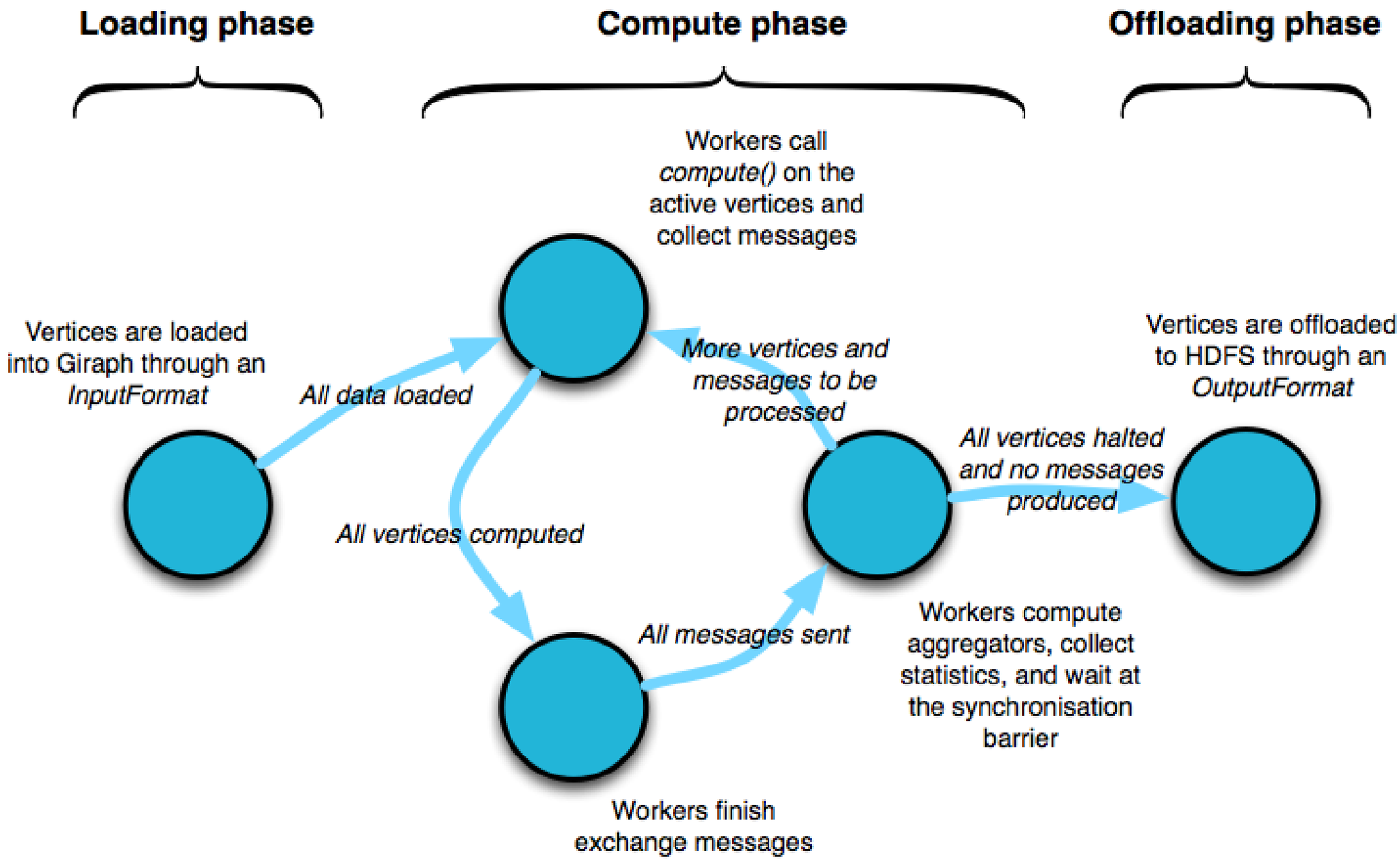
Advantages

- No locks: message-based communication
- No semaphores: global synchronization
- Iteration isolation: massively parallelizable

Architecture



Giraph job lifetime



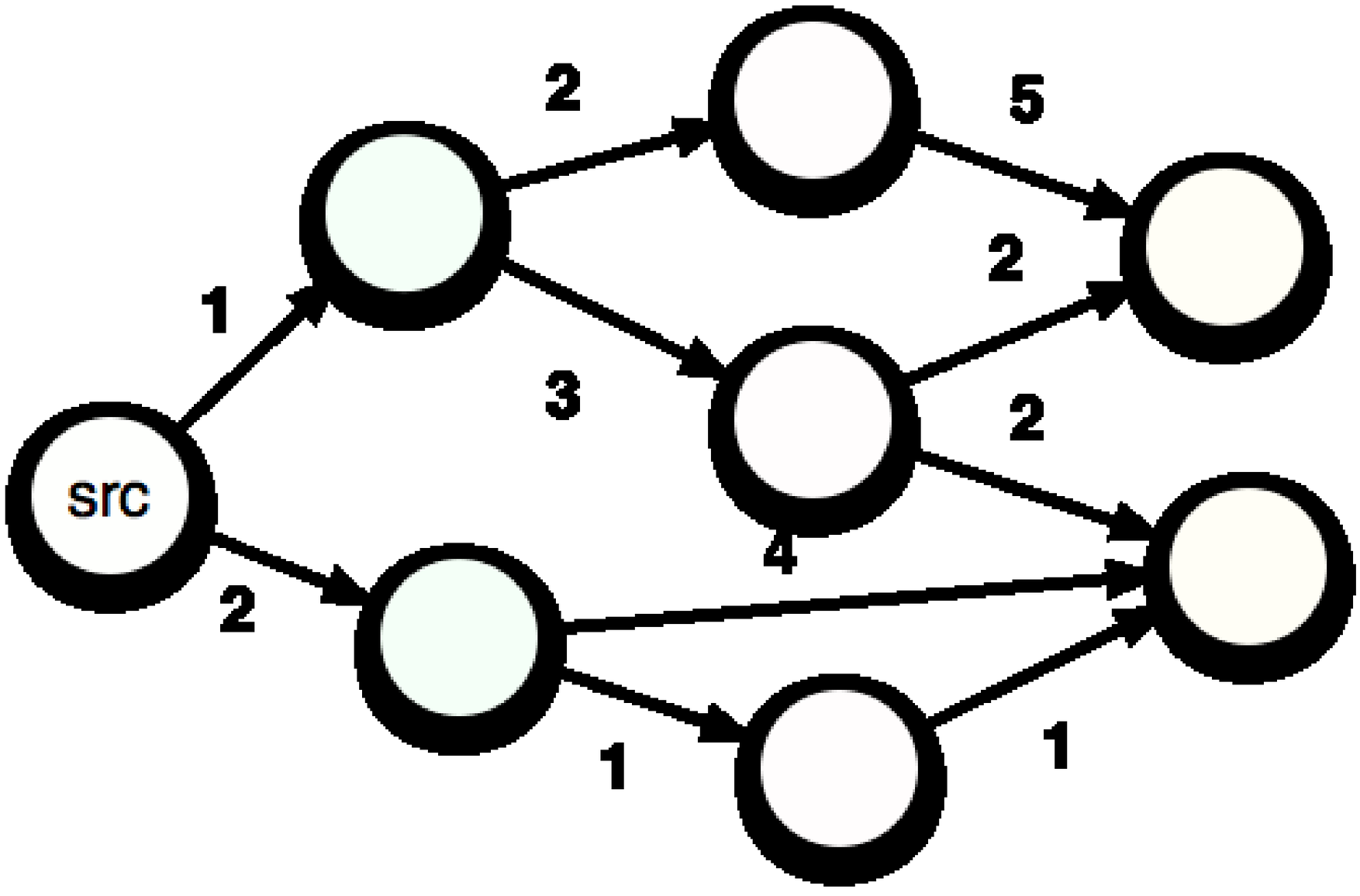
Designed for iterations

- Stateful (in-memory)
- Only intermediate values (messages) sent
- Hits the disk at input, output, checkpoint
- Can go out-of-core

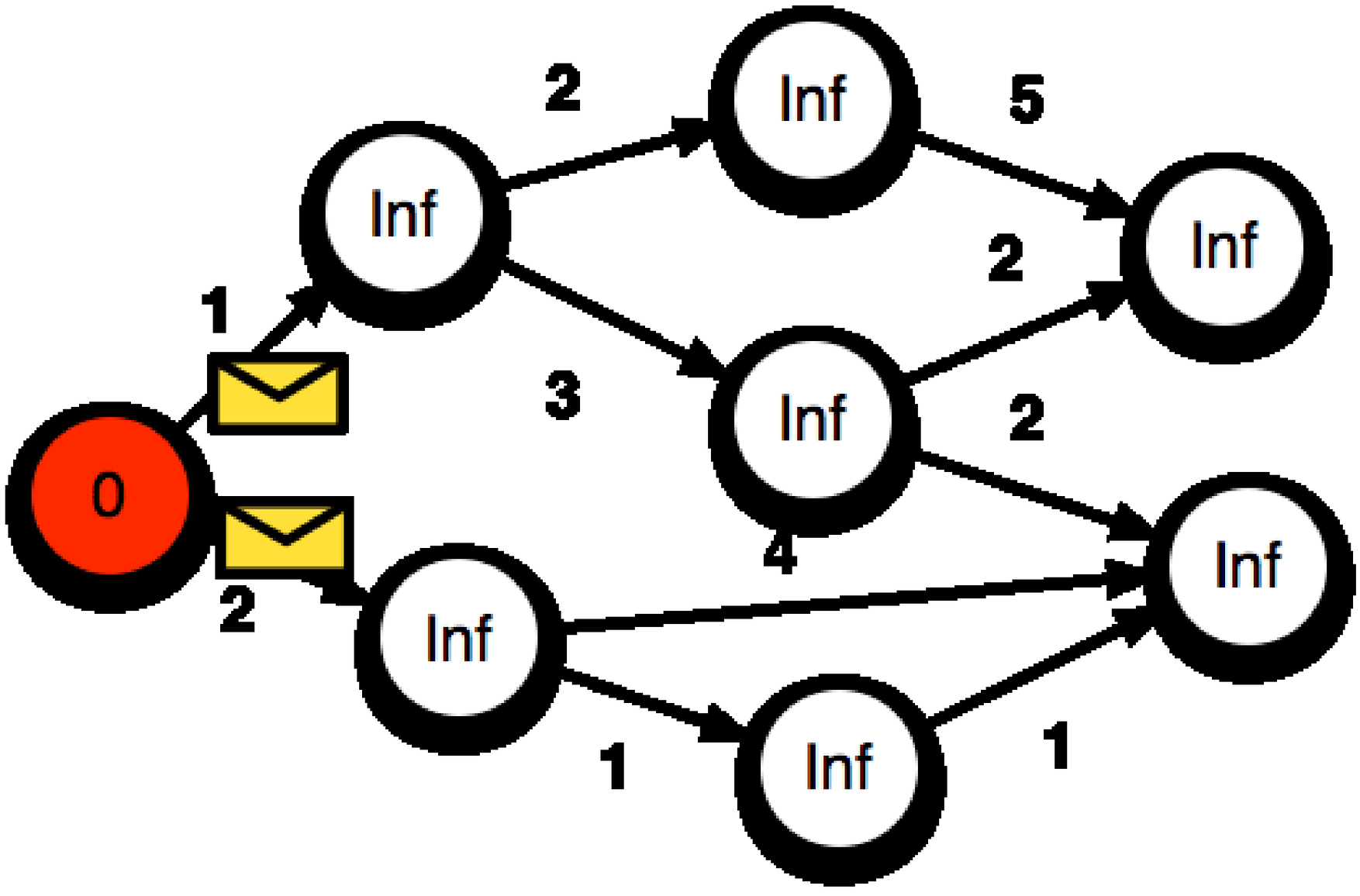
A bunch of other things

- Combiners (minimises messages)
- Aggregators (global aggregations)
- MasterCompute (executed on master)
- WorkerContext (executed per worker)
- PartitionContext (executed per partition)

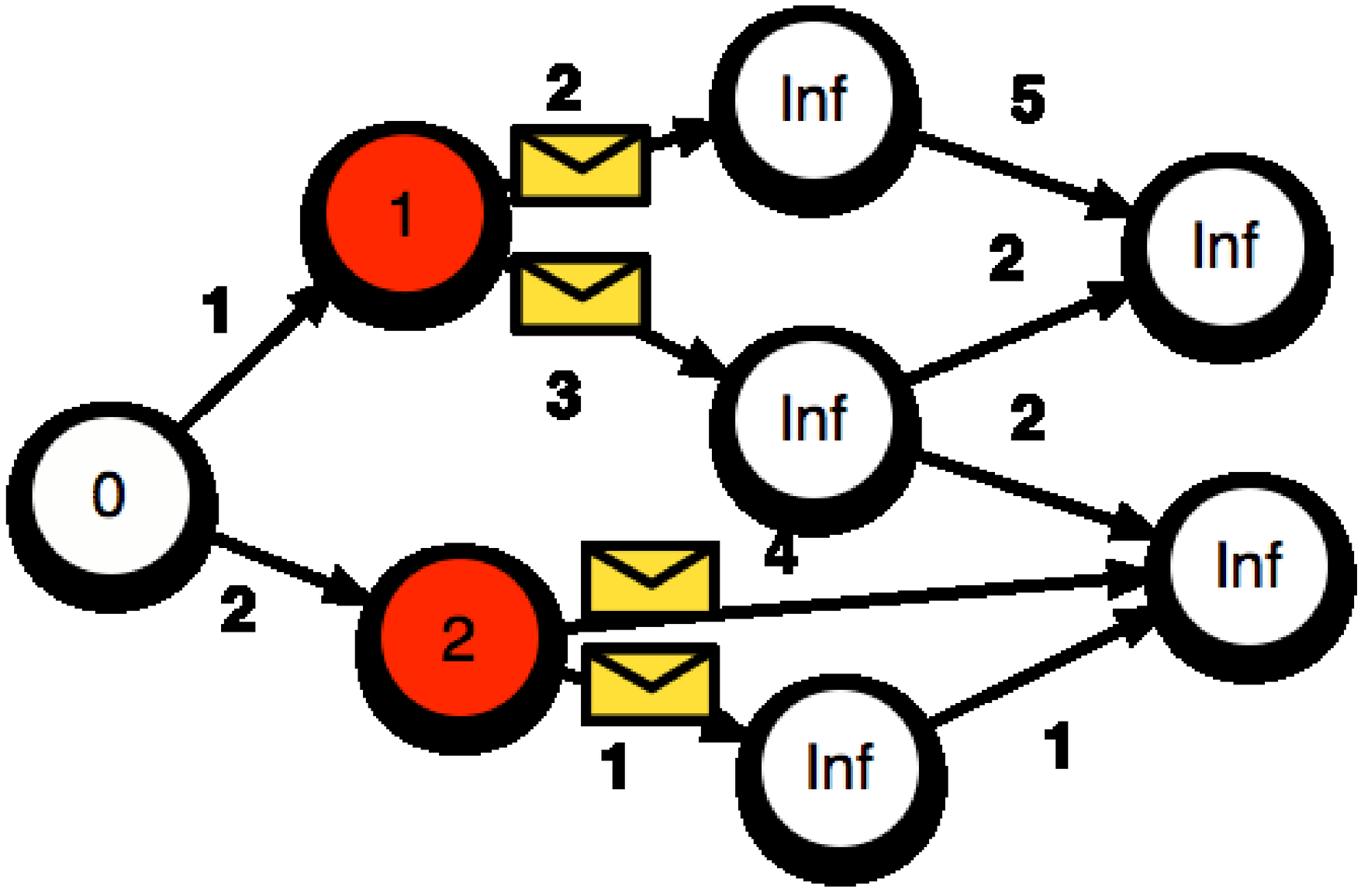
Shortest Paths



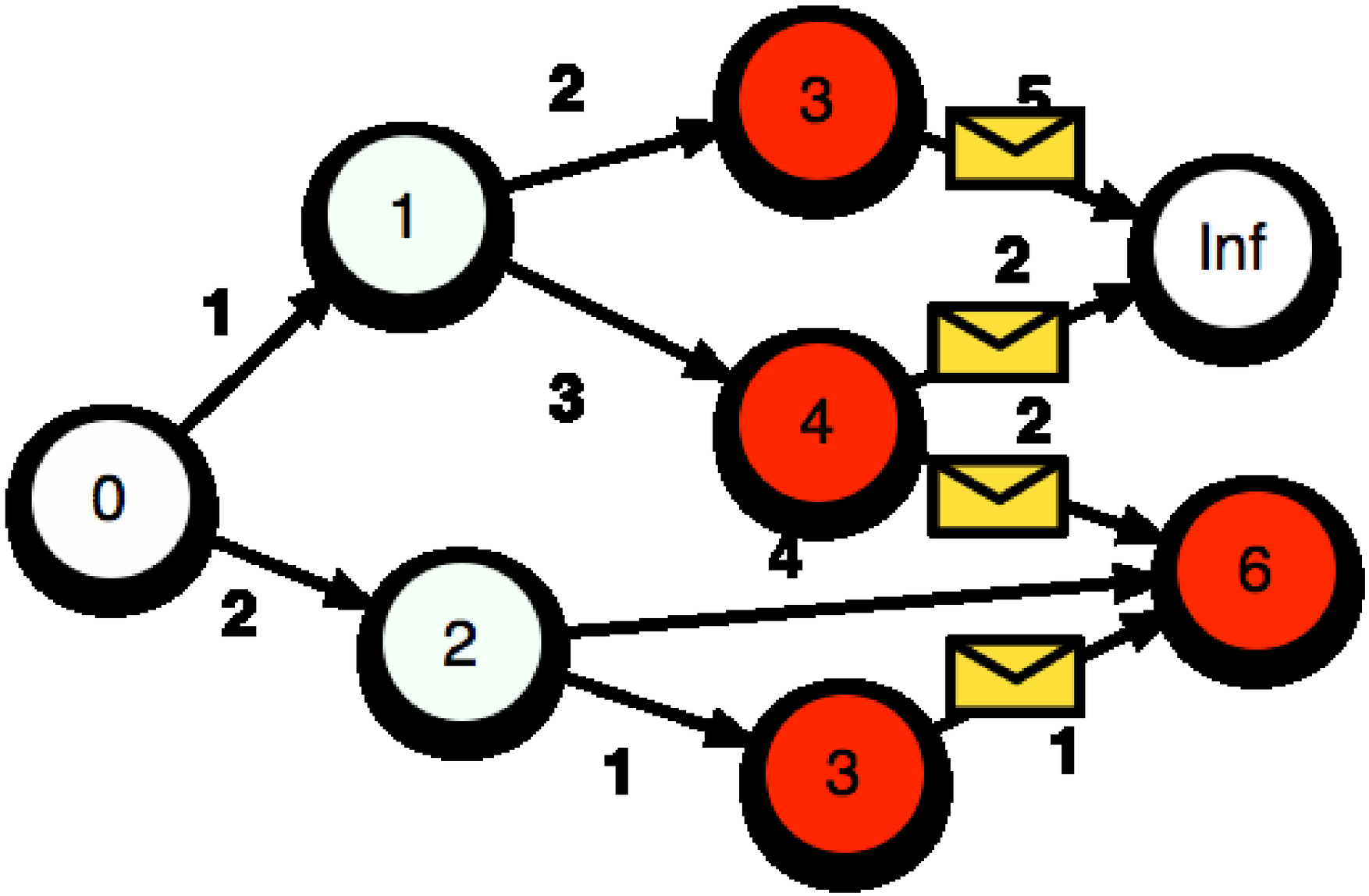
Shortest Paths



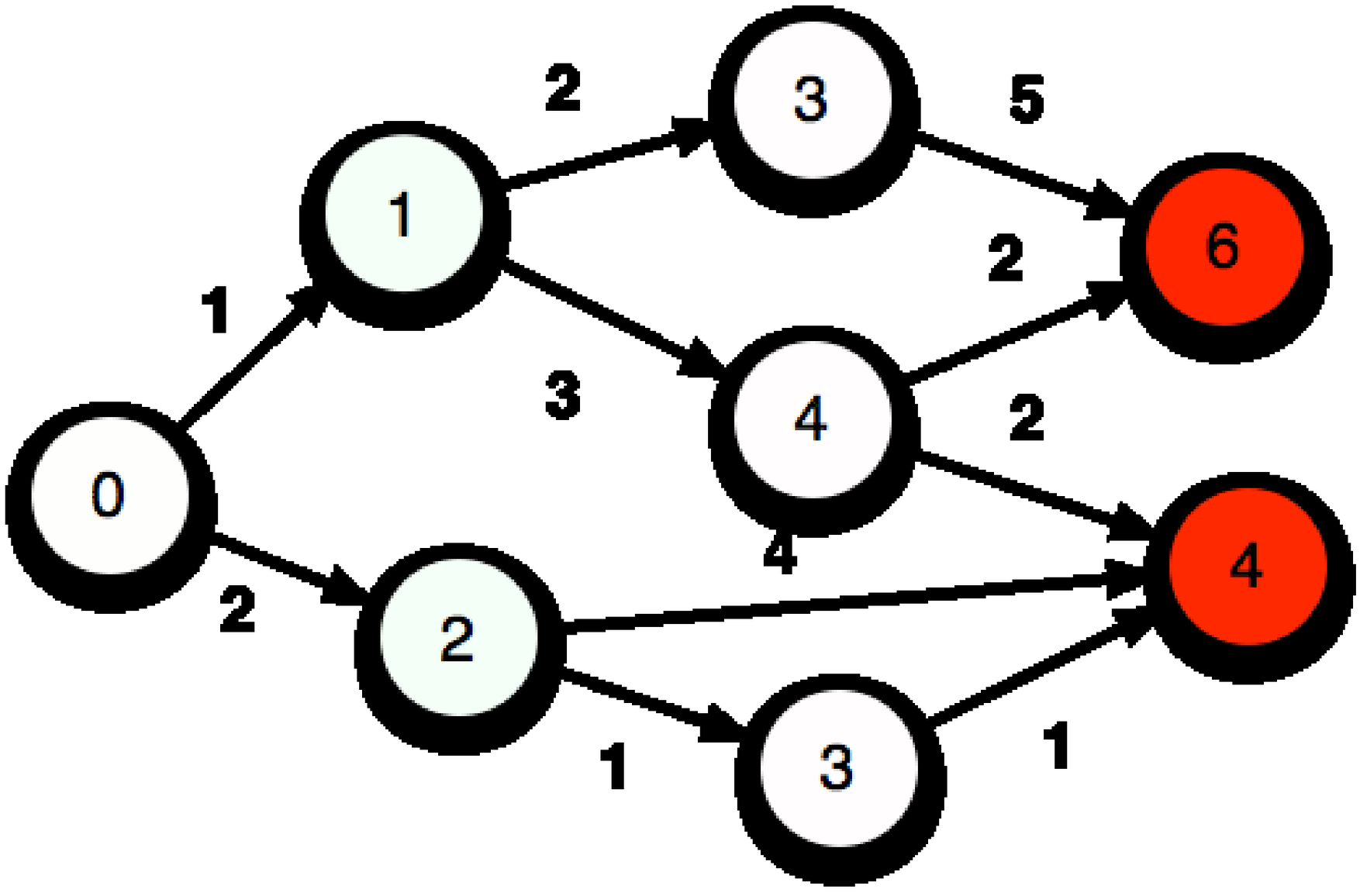
Shortest Paths



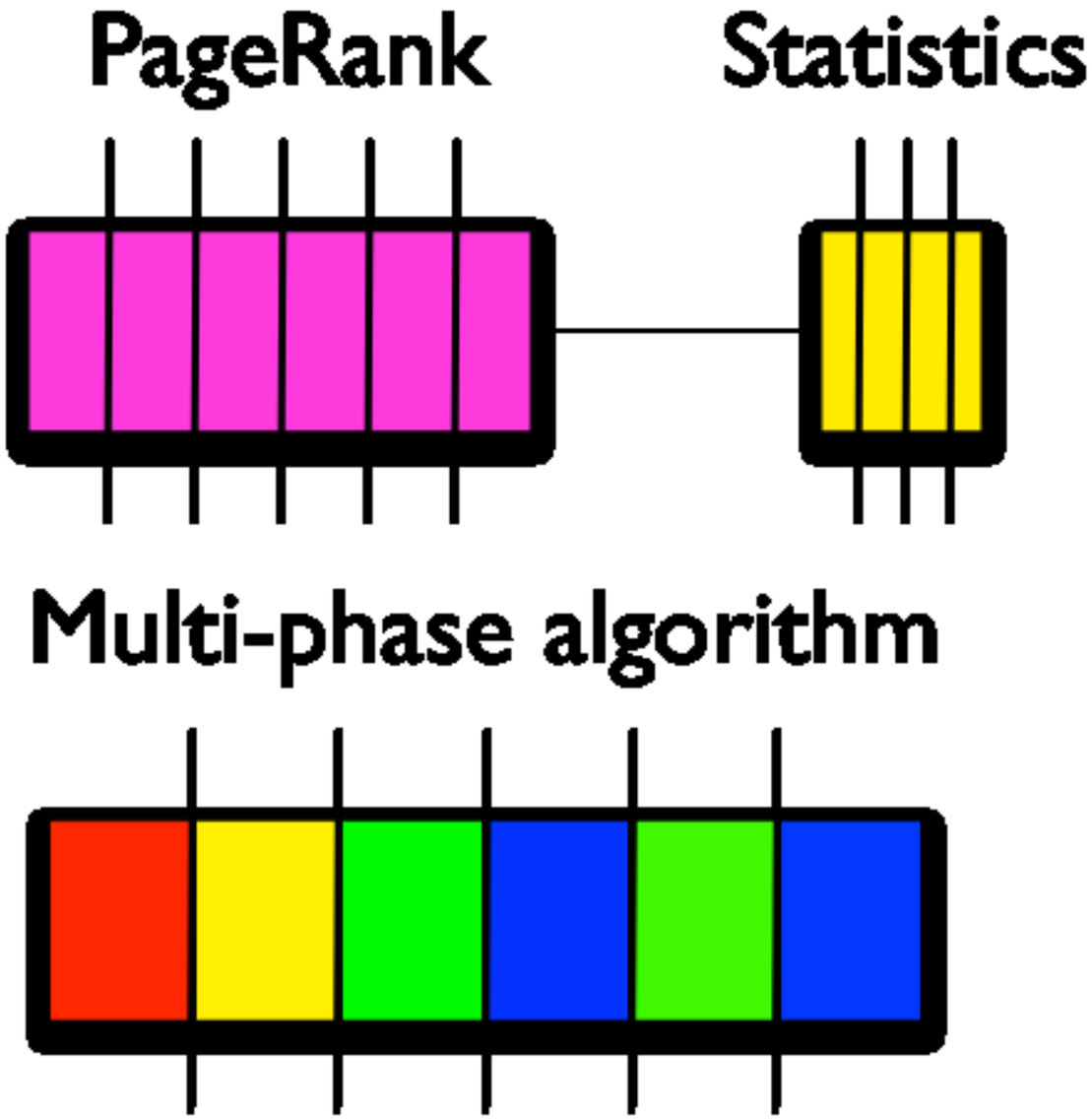
Shortest Paths



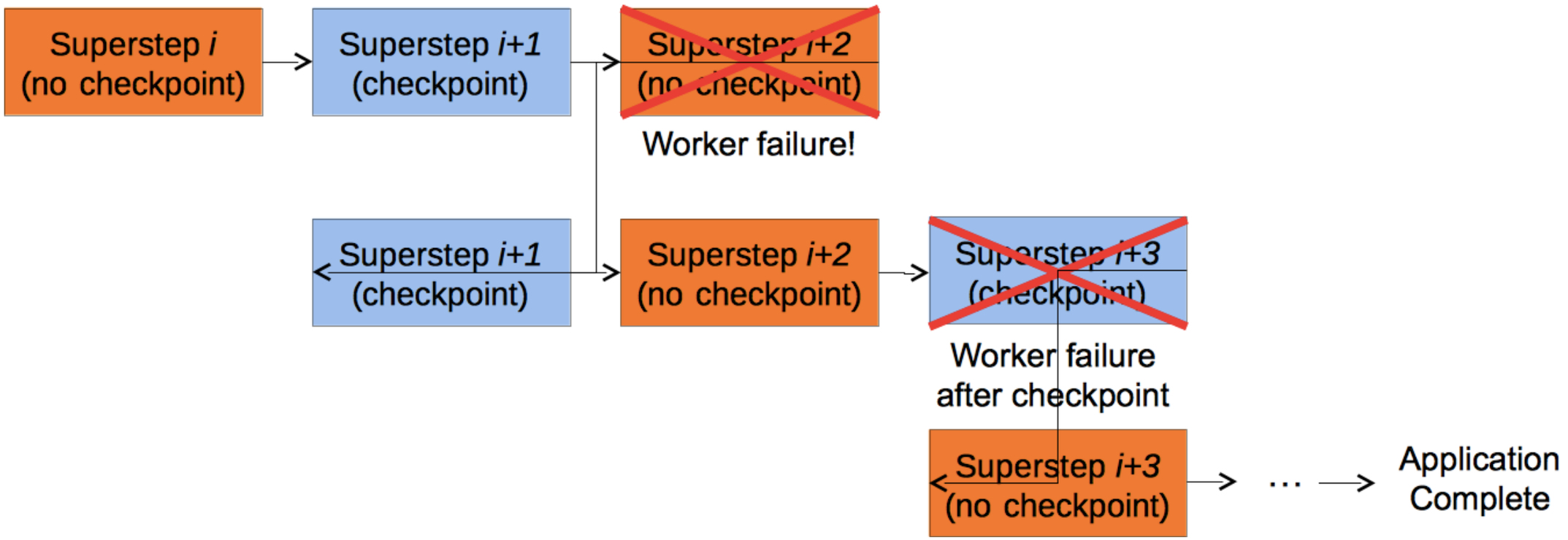
Shortest Paths



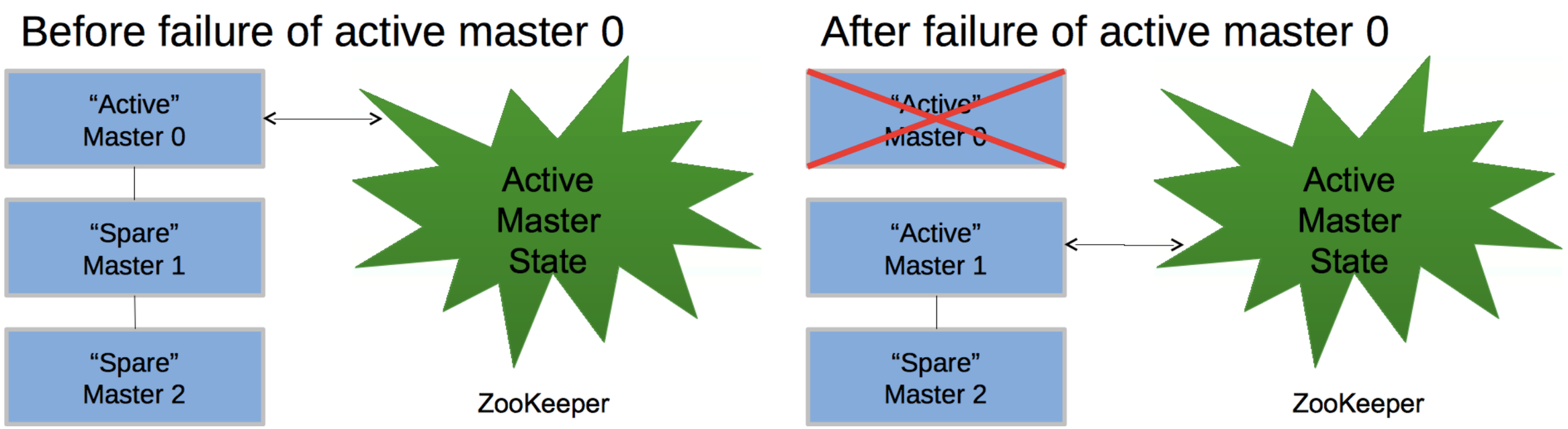
Composable API



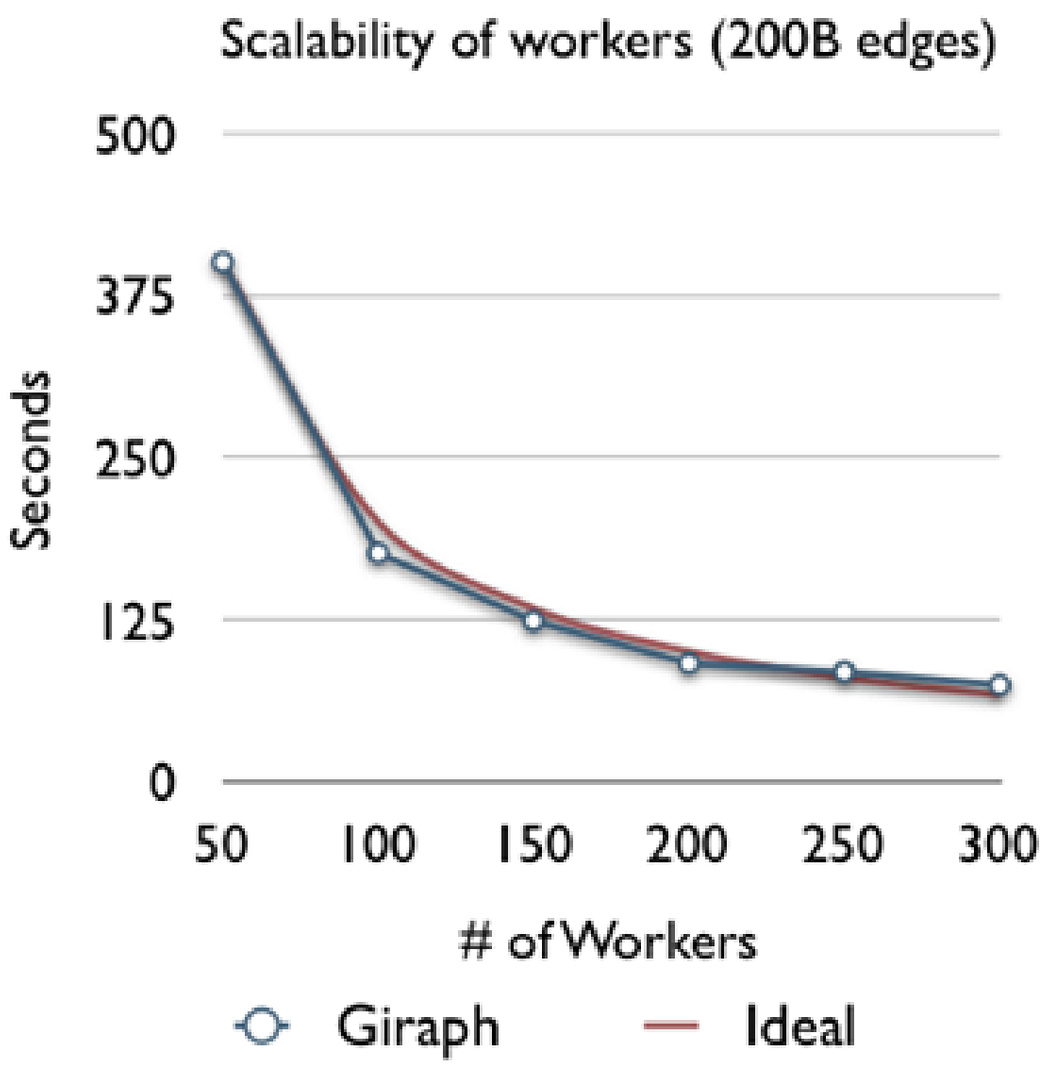
Checkpointing

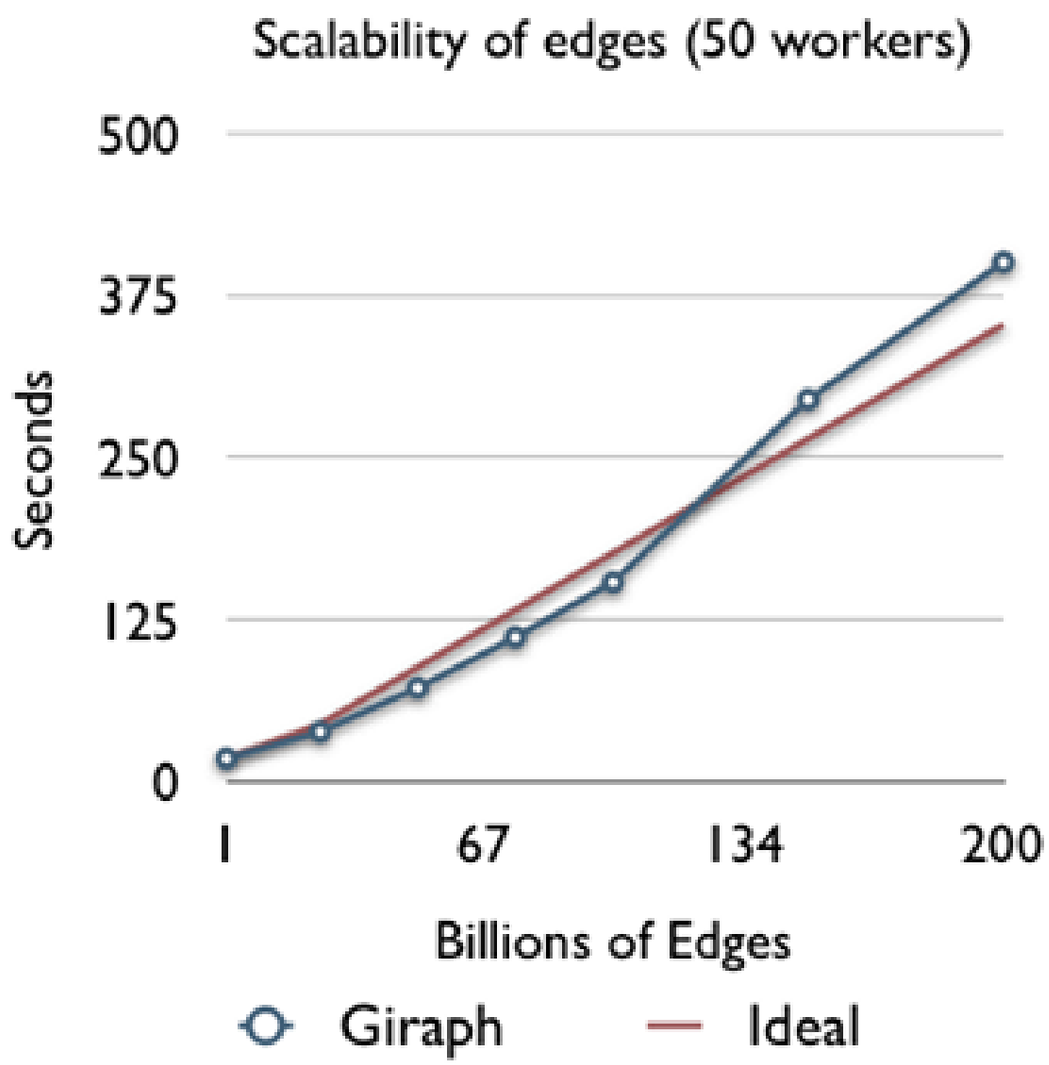


No SPoFs


Before failure of active master 0
"Active" Master 0
"Spare" Master 1
"Spare" Master 2
Active Master State
ZooKeeper
After failure of active master 0
"Active" Master 0
"Active" Master 1
"Spare" Master 2
Active Master State
ZooKeeper

Giraph scales



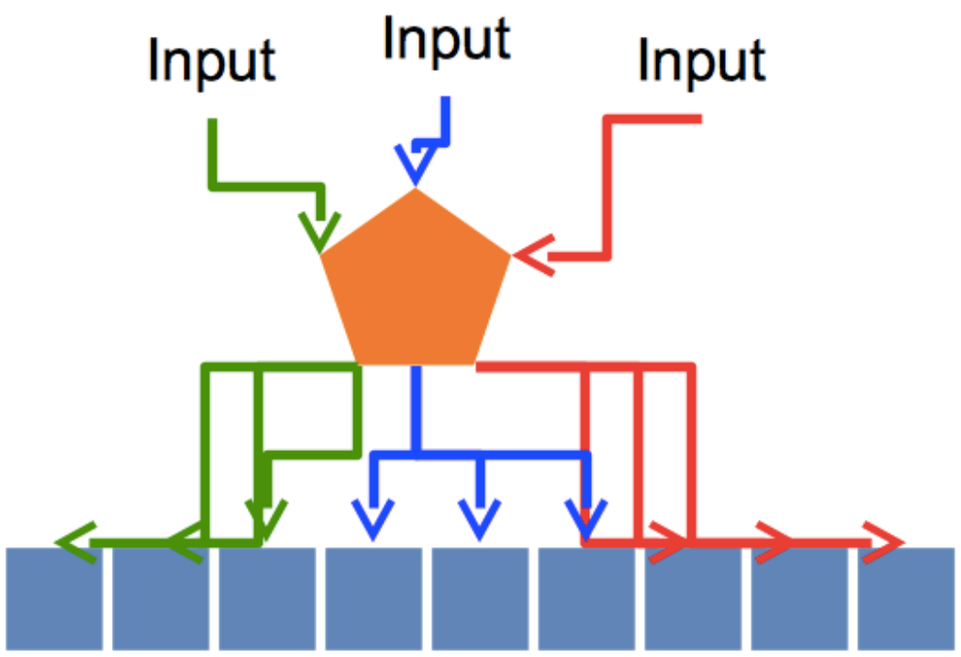


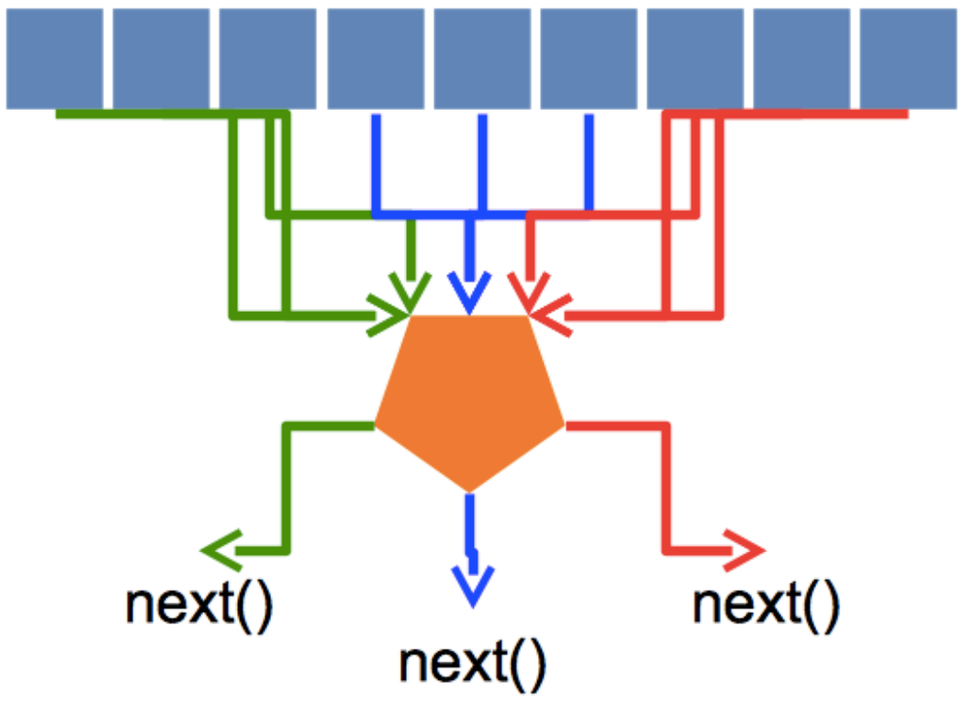
ref: https://www.facebook.com/notes/facebook-engineering/scaling-apache-giraph-to-a-trillion-edges/10151617006153920

Giraph is fast

- 100x over MR (Pr)
- jobs run within minutes
- given you have resources ;-)

Serialised objects

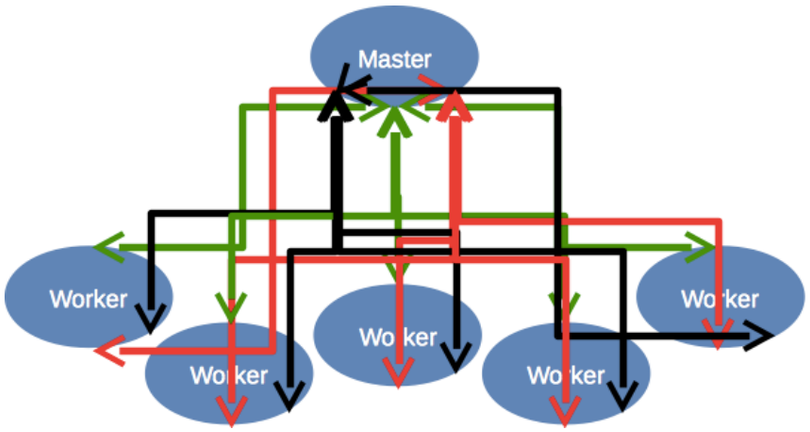




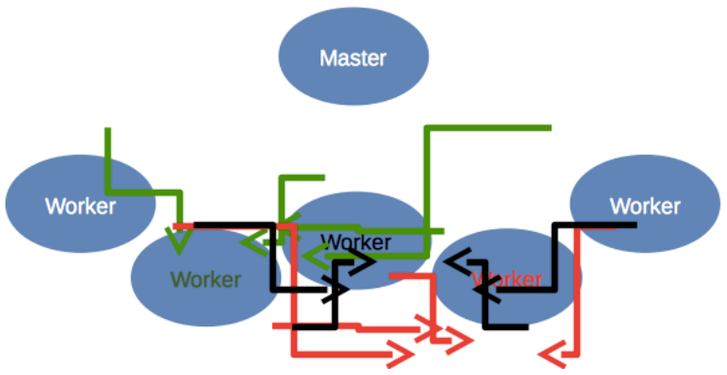
Primitive types

- Autoboxing is expensive
- Objects overhead (JVM)
- Use primitive types on your own
- Use primitive types-based libs (e.g. fastutils)

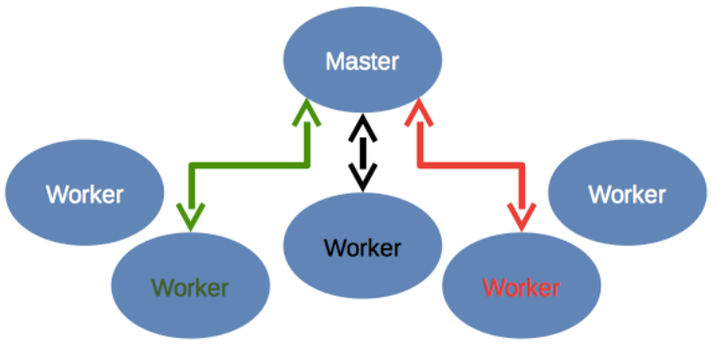
Sharded aggregators



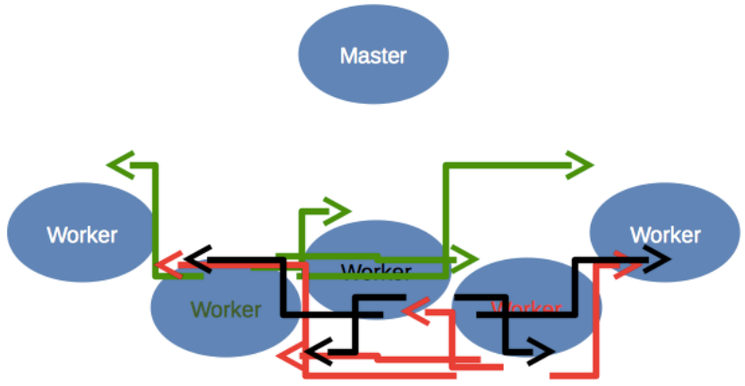
Workers own aggregators



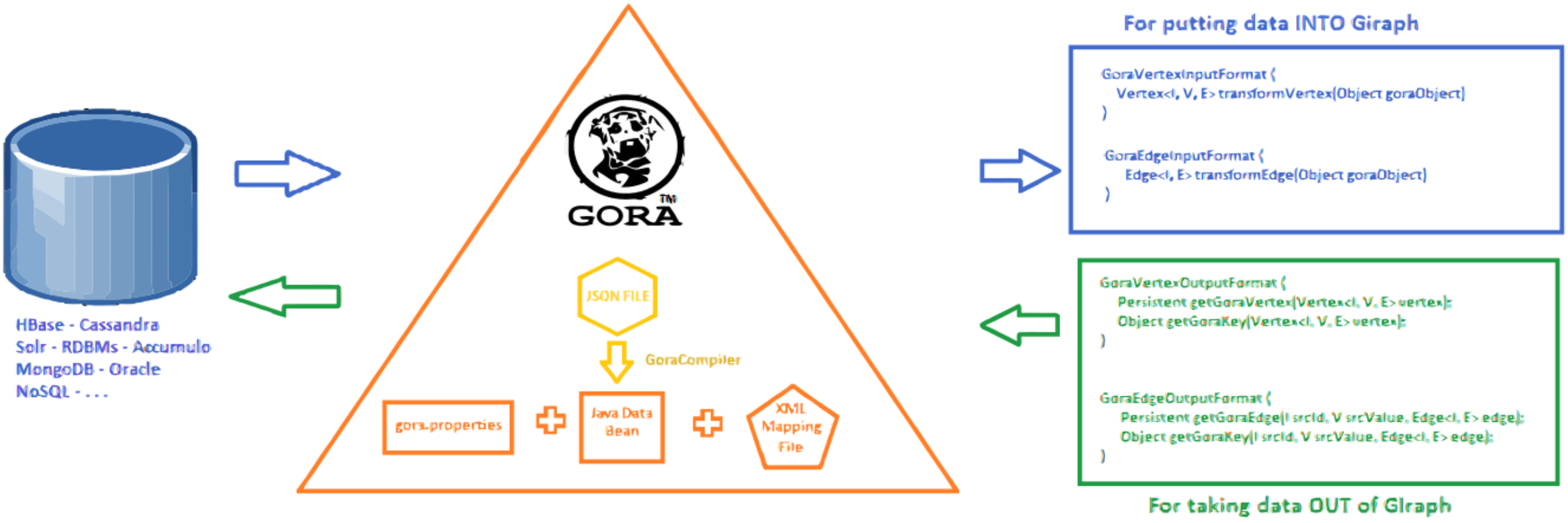
Aggregator owners communicate with Master



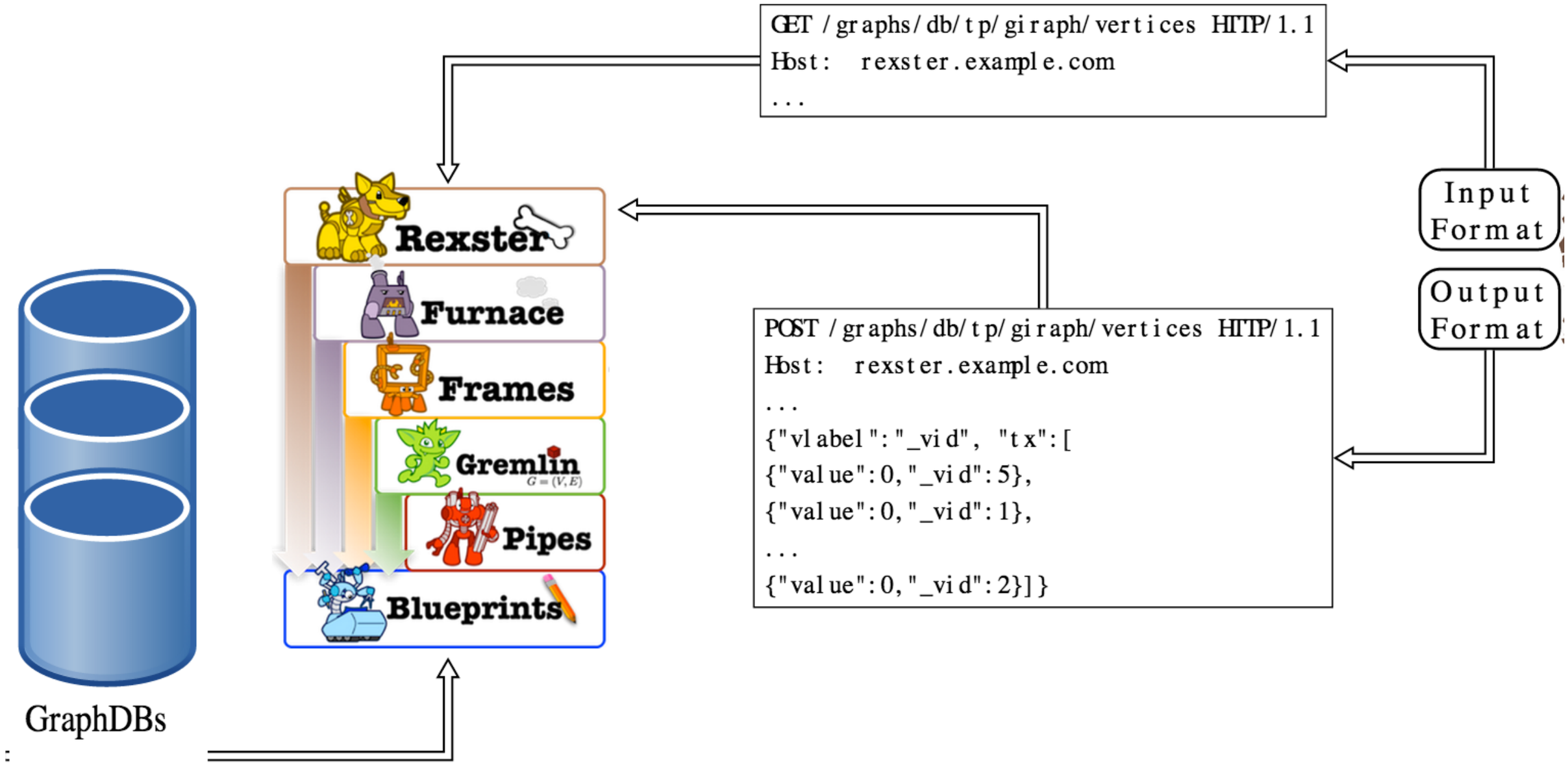
Aggregator owners distribute values



Many stores with Gora



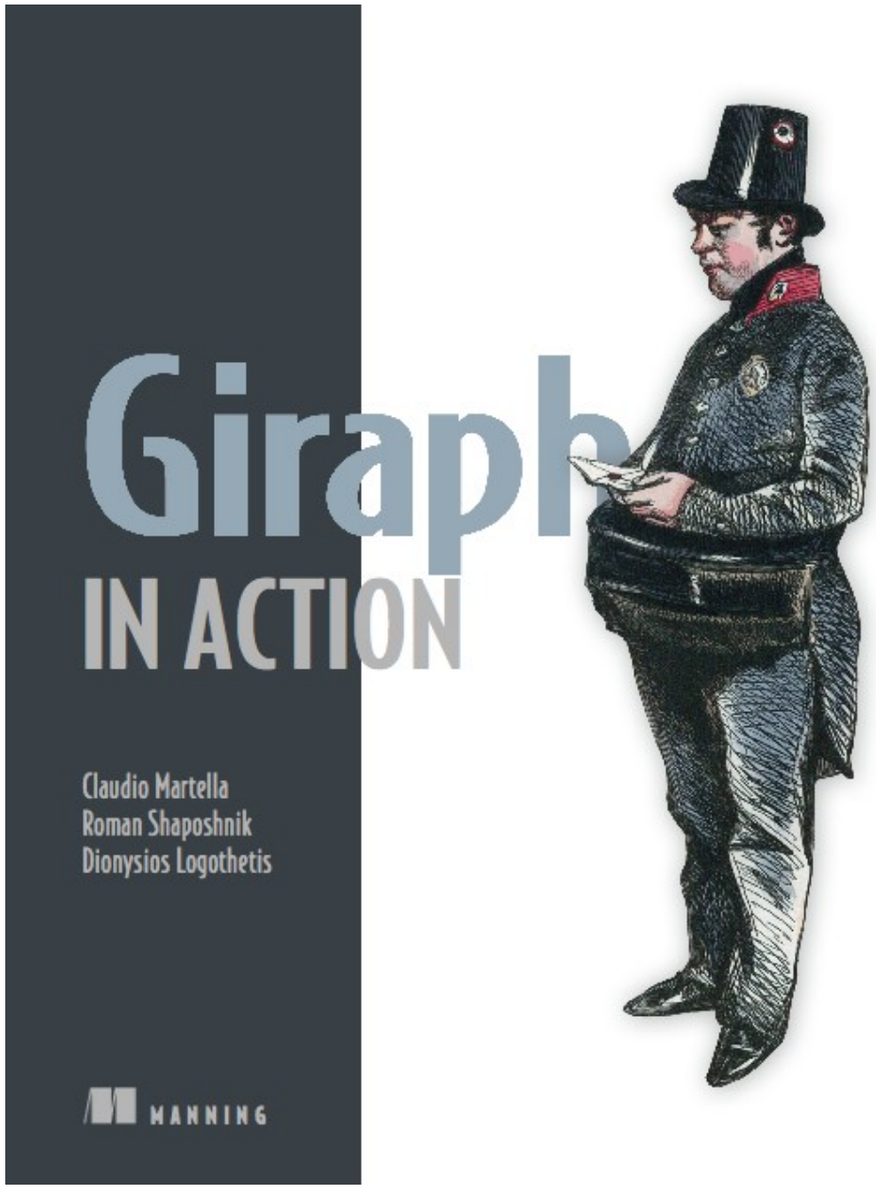
And graph databases



Current and next steps

- Out-of-core graph and messages
- Jython interface
- Remove Writable from < I V E M >
- Partitioned supernodes
- More documentation

Giraph in Action



- Published by Manning
- MEAP now
- Complete Q3 2014 (well...)
- Part 1: Graphs and Algorithms
- Part 2: Giraph Basic Topics
- Part 3: Giraph Advanced Topics
- http://www.manning.com/martella

Okapi

- Apache Mahout for graphs
- Graph-based recommenders: ALS, SGD, SVD++, etc.
- Graph analytics: Graph partitioning, Community Detection, K-Core, etc.

Thank you

http://giraph.apache.org

Claudio Martella <claudio@apache.org>
@claudiomartella

Some figures gently borrowed from Nitay Joffe:
http://www.slideshare.net/nitayj/20130910-giraph-at-london-hadoop-users-group