One Trillion edges : Graph Processing at Facebook-Scale

Tong Niu
tong.niu.cn@outlook.com

Outline

- Introduction
    - Improvements
- Experiment Results
- Conclusion& Future Work
- Discussion

Introduction

- Graph Structures(entities, connections)
  - social networks
    - Facebook manages a social graph that is composed of people, their friendships, subscriptions, likes, posts, and many other connections.

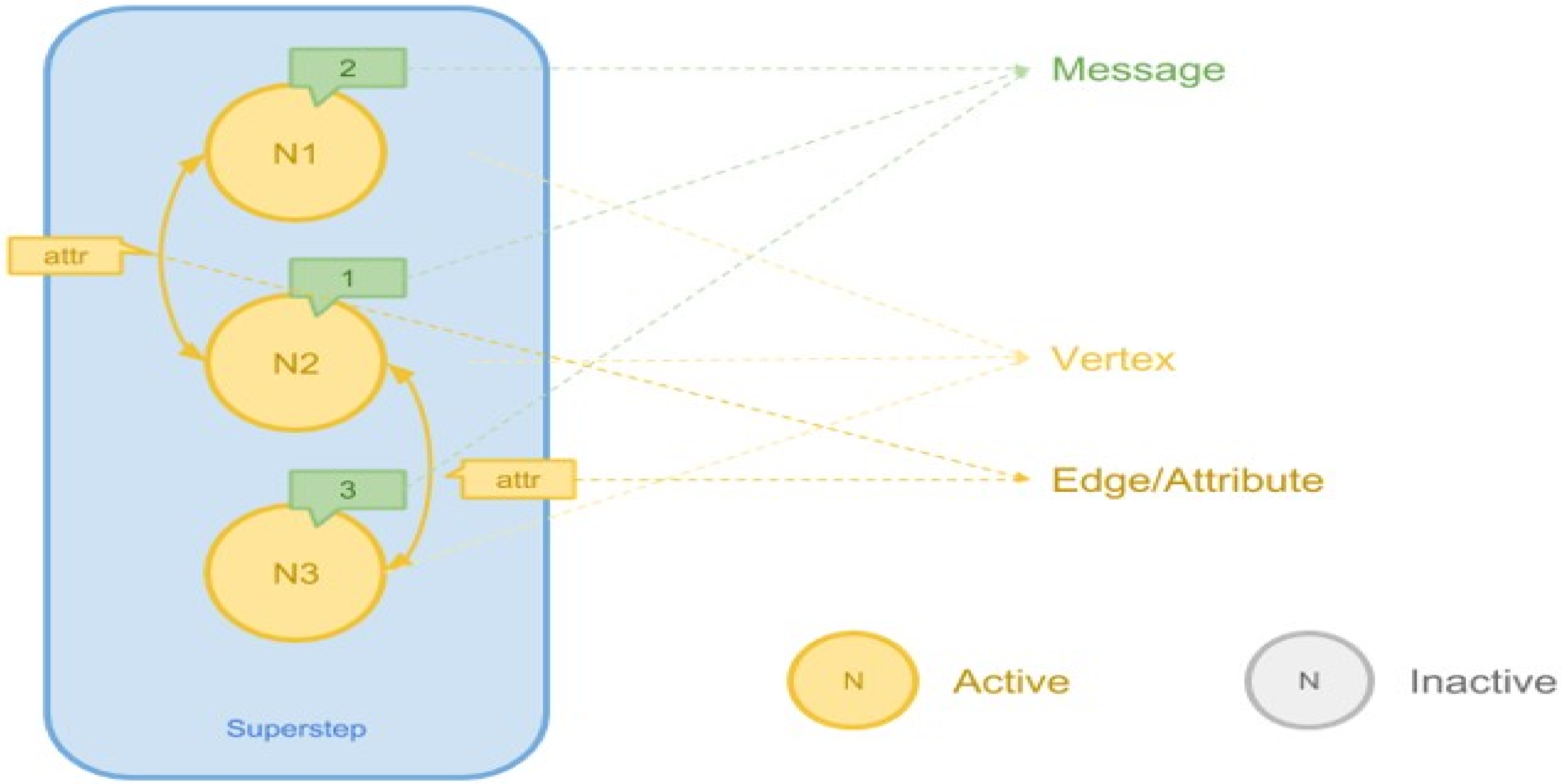
1.39B active users in 2014
with more than 400B edges

Introduction

- What is Apache Giraph?
    - "Think like a vertex"
        - Each vertex has an id, a value, a list of adjacent neighbors and corresponding edge values
    - Bulk synchronous processing(BSP)
        - Break up to several supersteps(iteration)
        - Messages are sent during a superstep from one vertex to another and then delivered in the following supersteps

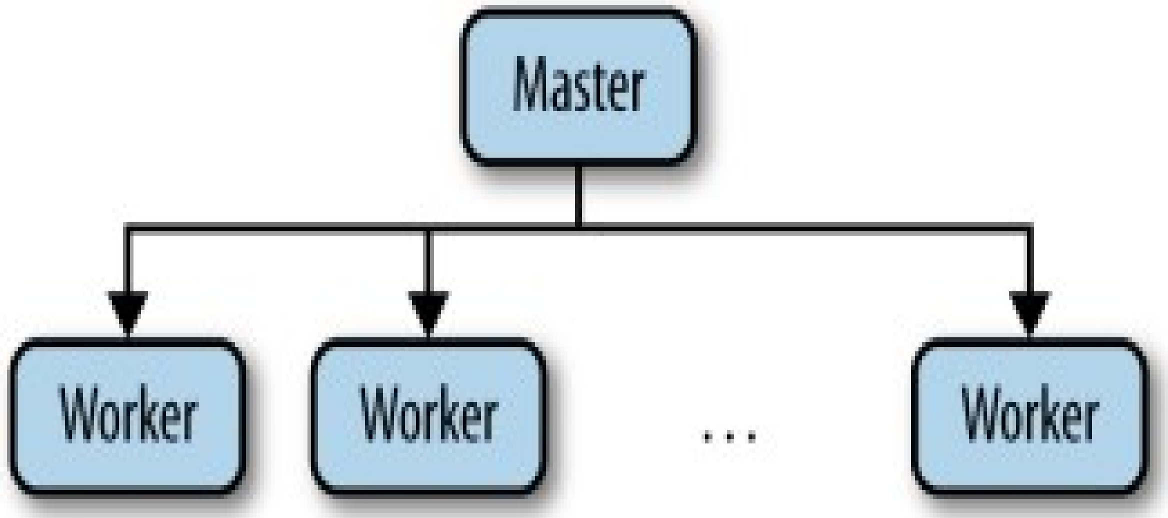
Introduction

- What is Apache Giraph?



Introduction

- What is Apache Giraph?



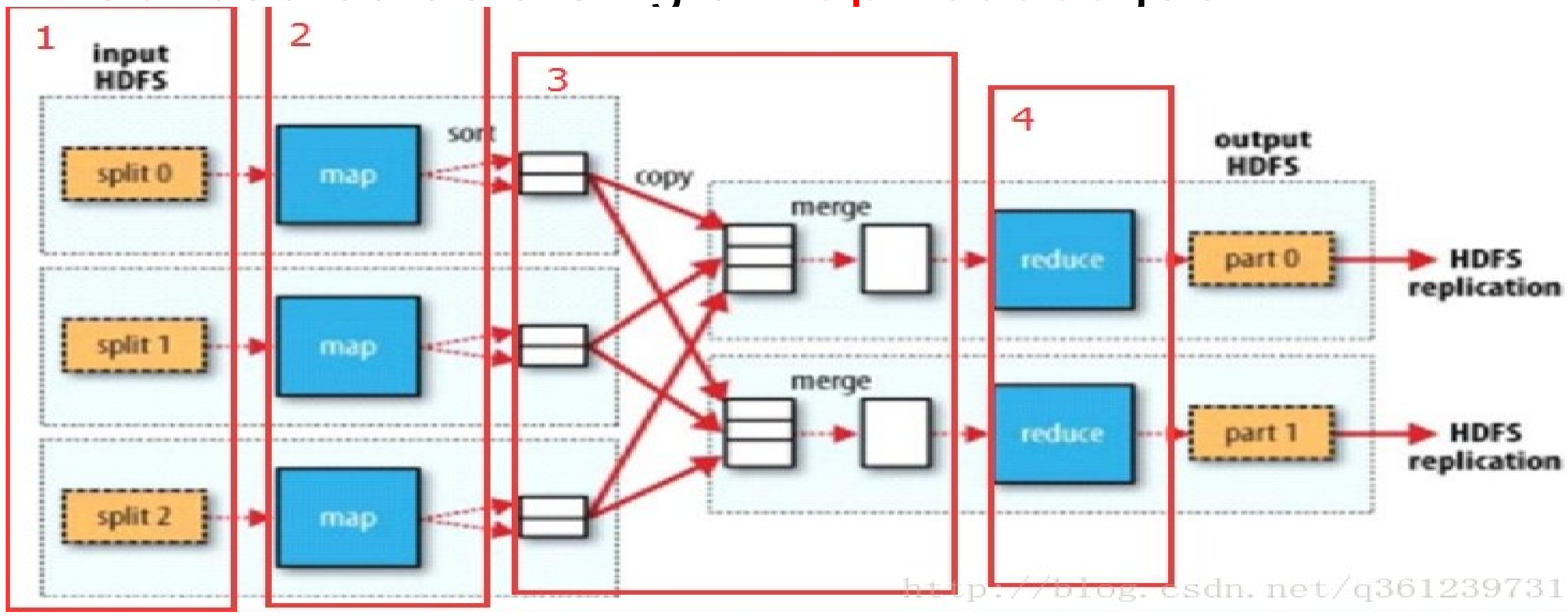
- Master – Application coordinator
  - Assigns partitions to workers
  - Synchronizes supersteps
- Worker – Computation, messaging
  - Load the graph from input splits
  - Does the computation/messaging of its assigned partitions

1. Flexible vertex/edge based input

- Original input:
  - All data(vertex/edge) need to be read from the same record and assumed to the same data source
- Modified input:
  - Allow loading vertex data and edges from separate sources
  - Add an arbitrary number of data sources

2. Parallelization support

- Original:
  - Scheduled as a single MapReduce job

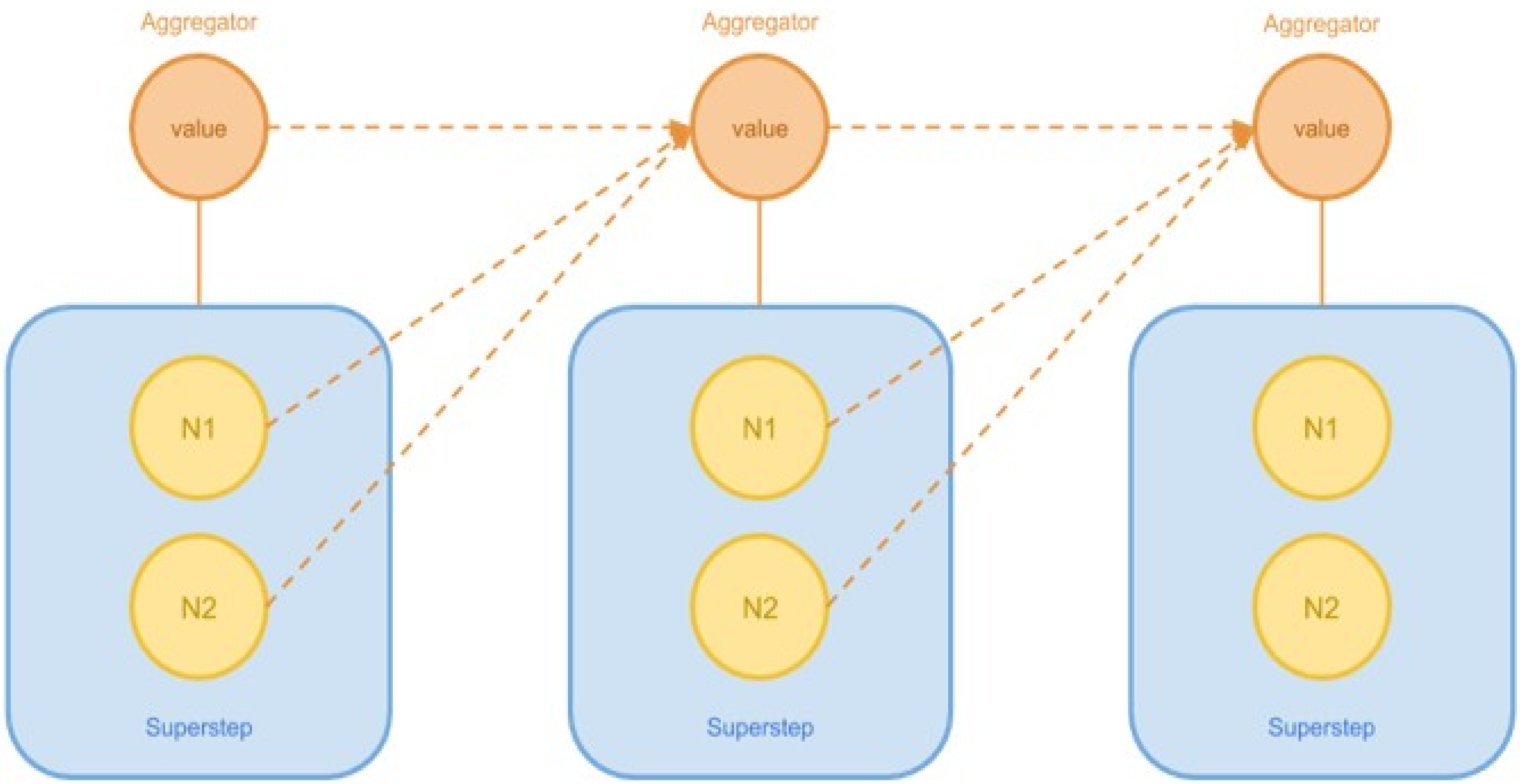


- Modified:
  - Add more workers per machine
  - Use local multithreading to maximize resource utilization

3. Memory optimization

- Original:
  - Large memory overhead because of flexibility
- Modified:
  - Serialize the edges of every vertex into a bit array rather than using native direct serialization methods
  - Create an OutEdges interface that allow developers to achieve edge stores

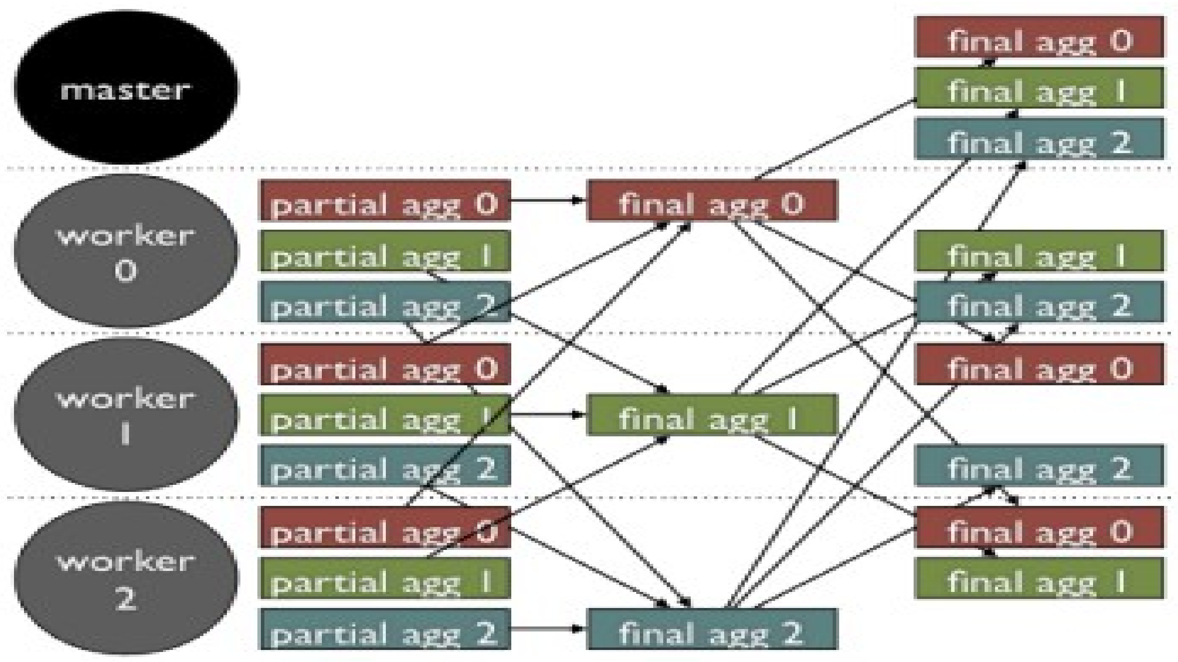
4. Sharded aggregators



- global computation(min/max value)
- provide efficient shared state across workers
- make the values available in the next superstep

4. Sharded aggregators

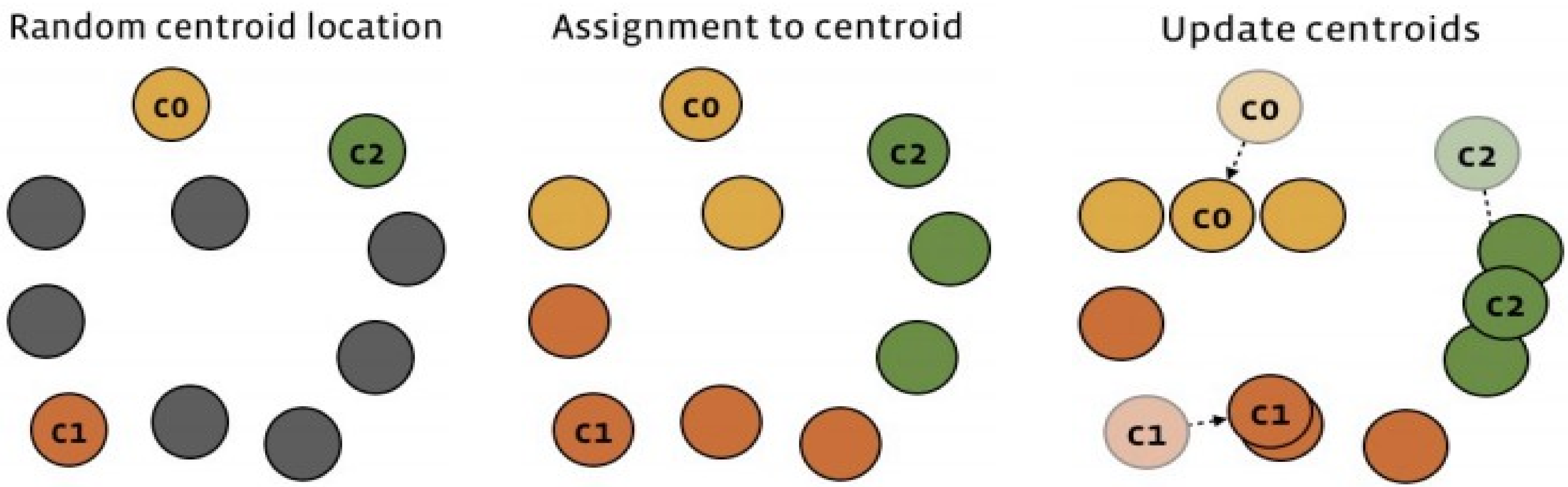
- Original:
  - Use znodes in zookeeper to store partial aggregated data from workers, master aggregate all of them and write result back to znode for workers to access it
  - every worker has plenty of data that need to be aggregated
- Modified:



Randomly assigned to one of the workers

Distribute final values to master/workers

K-Means clustering



In a graph application, input vectors are vertices, and centroids are aggregators.

1. Worker phases

- Add preApplication() to initialize positions of centroids
- Add preSuperstep() to calculate the new position for each of the centroids before next superstep

2. Master computation

- Centralized computation prior to every superstep that can communicate with the workers via aggregators

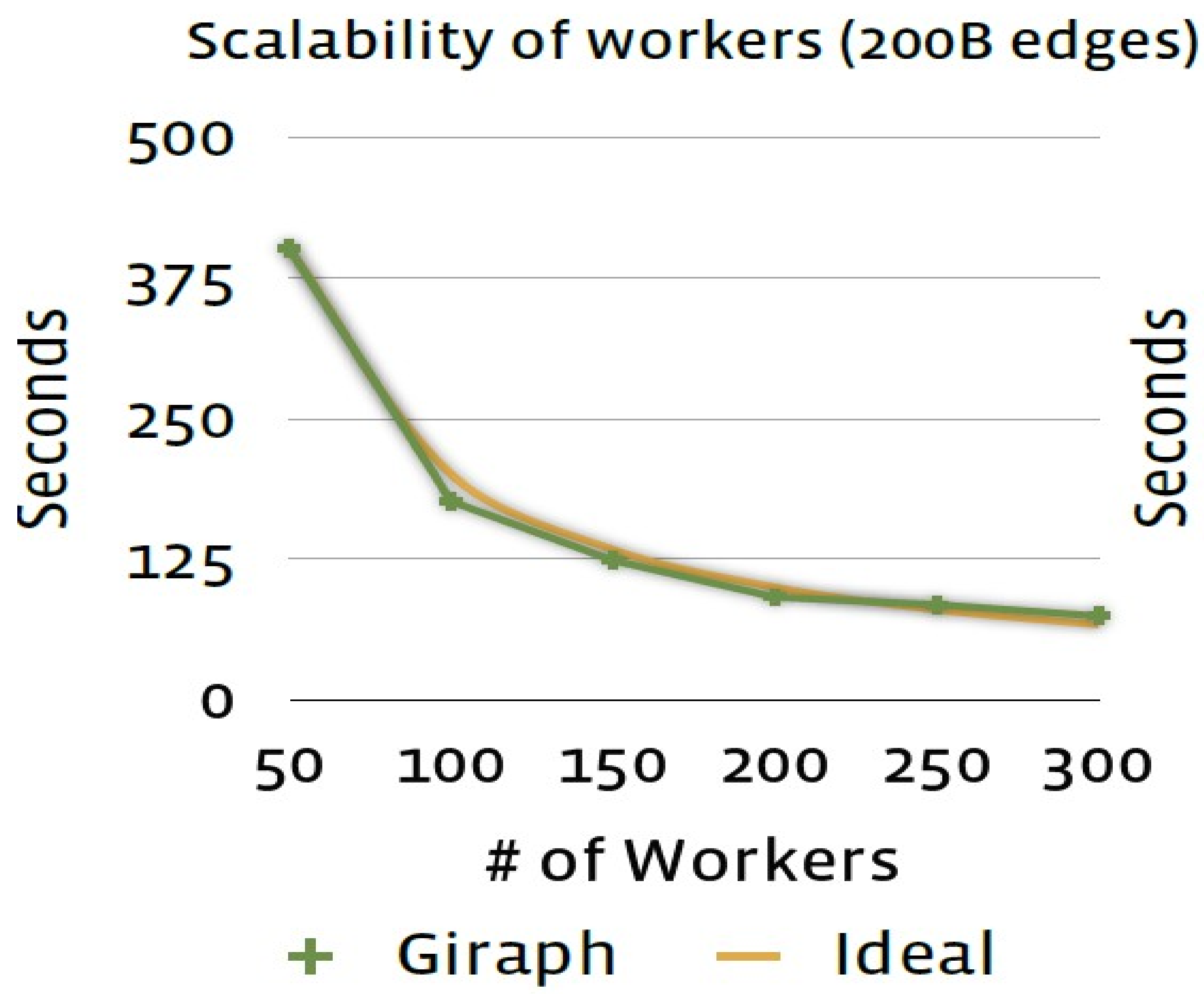
3. Composable computation

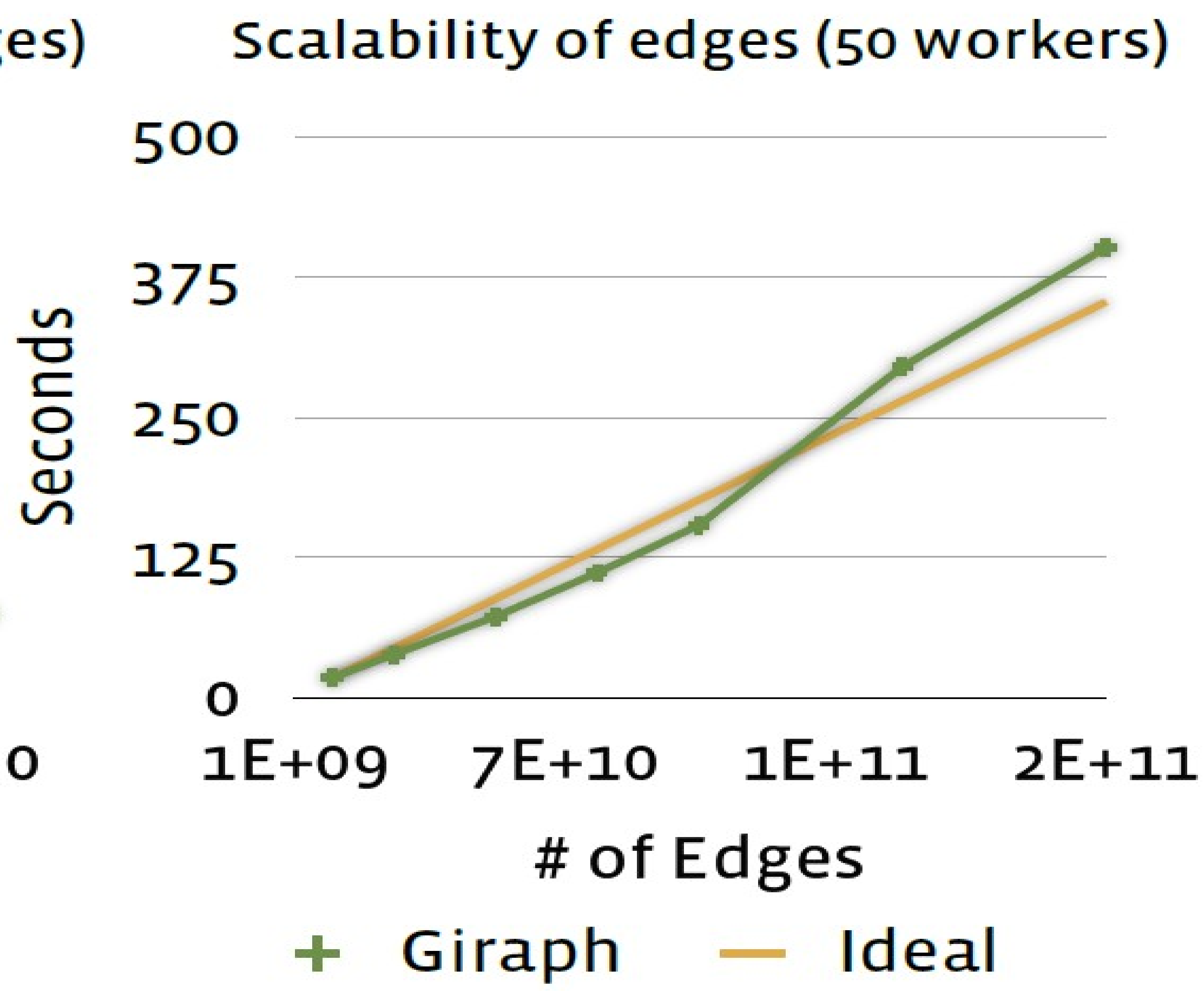
- Allows us to use different message types ,combiners and computation to build a powerful k-means application

4. Superstep splitting

- For a message heavy superstep
- send a fragment of messages to the destinations and do a partial computation during each iteration

Experiment results





Experiment results

- Giraph(200 machines) vs Hive(at least 200 machines)
  - compare CPU time and elapsed time
    - label propagation algorithm

Graph size	Hive	Giraph	Speedup
701M+ vertices 48B+ edges	Total CPU 9.631M secs	Total CPU 1.014M secs	9x
	Elapsed Time 1,666 mins	Elapsed Time 19 mins	87x

    - Weighted PageRank

Graph size	Hive	Giraph	Speedup
2B+ vertices 400B+ edges	Total CPU 16.5M secs	Total CPU 0.6M secs	26x
	Elapsed Time 600 mins	Elapsed Time 19 mins	120x

Conclusion & Future work

How a processing framework supports Facebook-scale production workloads. We have described the improvements to Giraph.

1. Determine a good quality graph partitioning prior to our computation.
2. Make our computation more asynchronous to improve convergence speed.
3. Leverage Giraph as a parallel machine-learning platform

Discussion