DREAM:Lab

Presented By
Abhilash Sharma

Overview

- Need for Graph Processing Systems
- Overview of Pregel
  - Bulk Synchronous Parallel
  - Pregel
  - Giraph Architecture

Need for Graph processing system

- Graphs are everywhere
  - Web and Social Graph, eg. twitter,facebook
  - Internet of Things
  - CyberSecurity
- Algorithms on Graphs
  - Traversals
  - Clustering
  - Centrality
- Scale of Graphs
  - Graphs are large
  - Clueweb12(978,408,098 V/42,574,107,469 E)
  - fb-2011(562,368,789 V/95,057,125,765 E)

Challenges

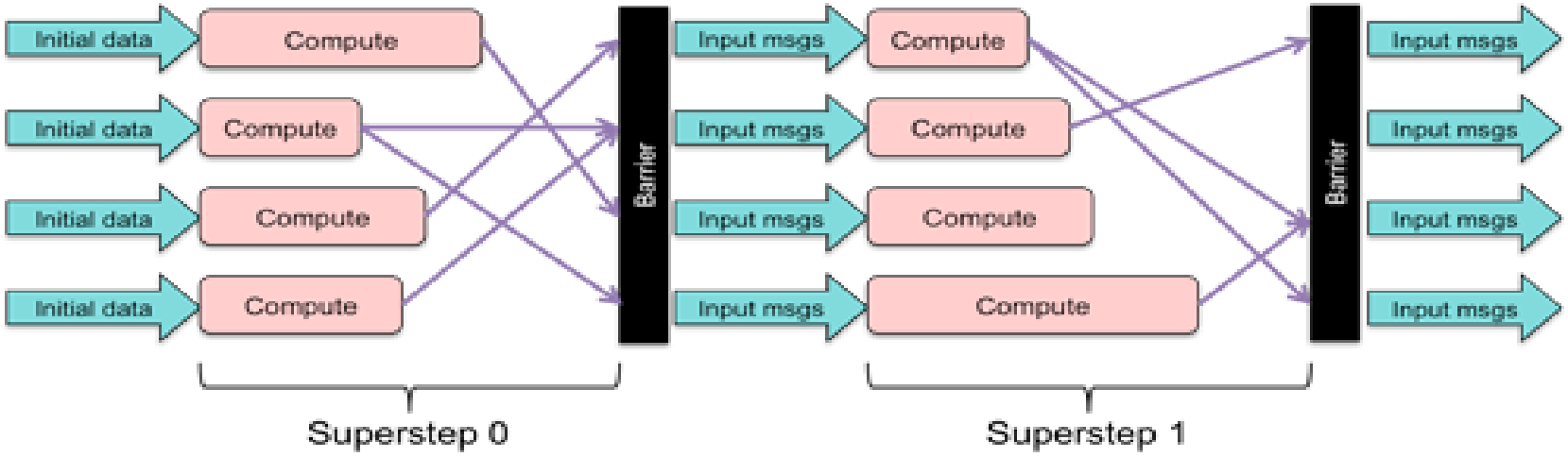
- Graphs don’t fit on memory of a single machine
- Graph algorithms are computationally expensive

Overview of Pregel

- Vertex-centric Model for writing Graph algorithms
  - Scalability
  - Expressibility in writing algorithms
  - Fault-tolerance
- Uses Bulk Synchronous parallel abstraction for communication and synchronization
- Apache Giraph is an open-source implementation of pregel

Bulk Synchronous Parallel

- Computations consist of a sequence of iterations, called supersteps
  - Concurrent computation
  - Communication
  - Barrier synchronisation



https://www.cs.rutgers.edu/~pxk/417/exam/images/bsp-500.png

Pregel Abstraction

- Algorithm written from a perspective of a vertex
  - Think like a vertex paradigm

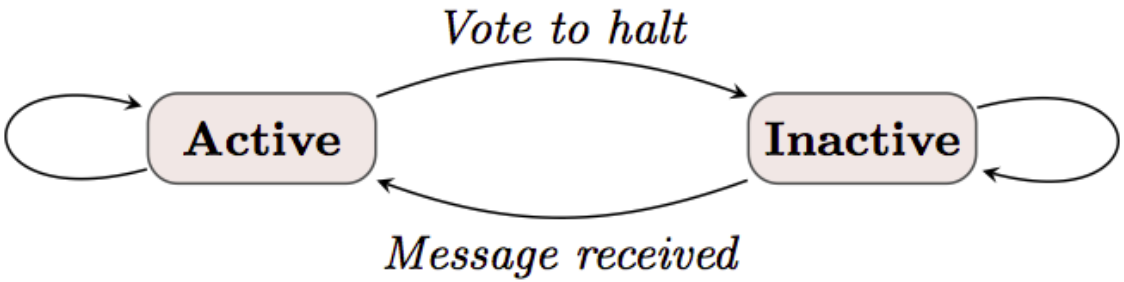
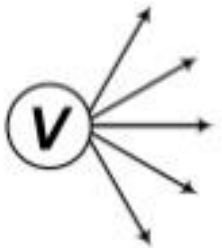
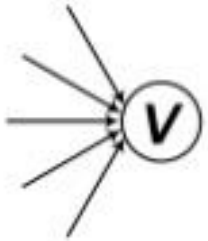


Figure 1: Vertex State Machine

Vertex-centric Computation Model

- Think like a vertex
- vertex_scatter(vertex *v*)
  - send updates over outgoing edges of *v*
- vertex_gather(vertex *v*)
  - apply updates from inbound edges of *v*





- repeat the computation iterations
  - for all vertices *v*
    - vertex_scatter(*v*)
  - for all vertices *v*
    - vertex_gather(*v*)

https://image.slidesharecdn.com/lingliu-part01biggraphprocessing-160616082617/95/ling-liu-part-01big-graph-processing-16-638.jpg?cb=1466065639

Giraph API

- void compute(Iterator<IntWritable> msgs)
  - getSuperstep()
  - getVertexValue()
  - edges = iterator() //list of edges
  - sendMsg(edge, value)
  - sendMsgToAllEdges(value)
  - VoteToHalt()
- Messages Passing
  - Message ordering not guaranteed
  - Can send messages to any node
  - Message is delivered exactly once

Max Vertex-value

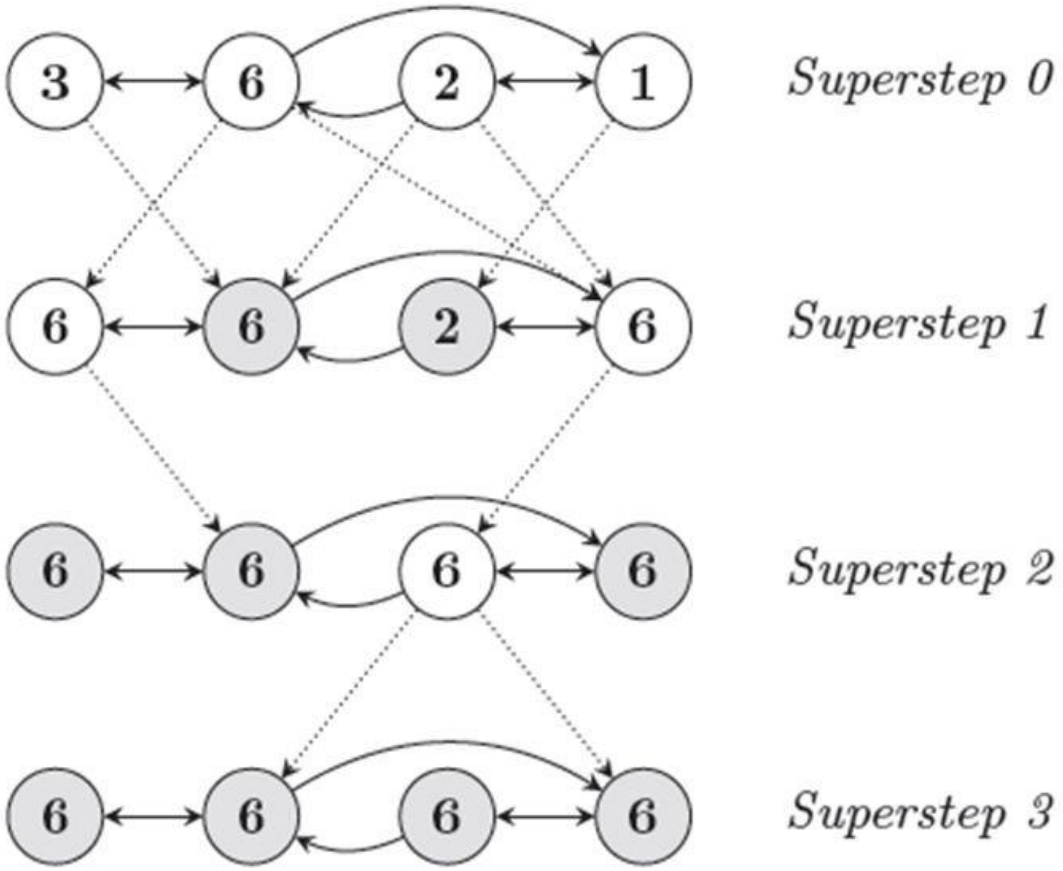
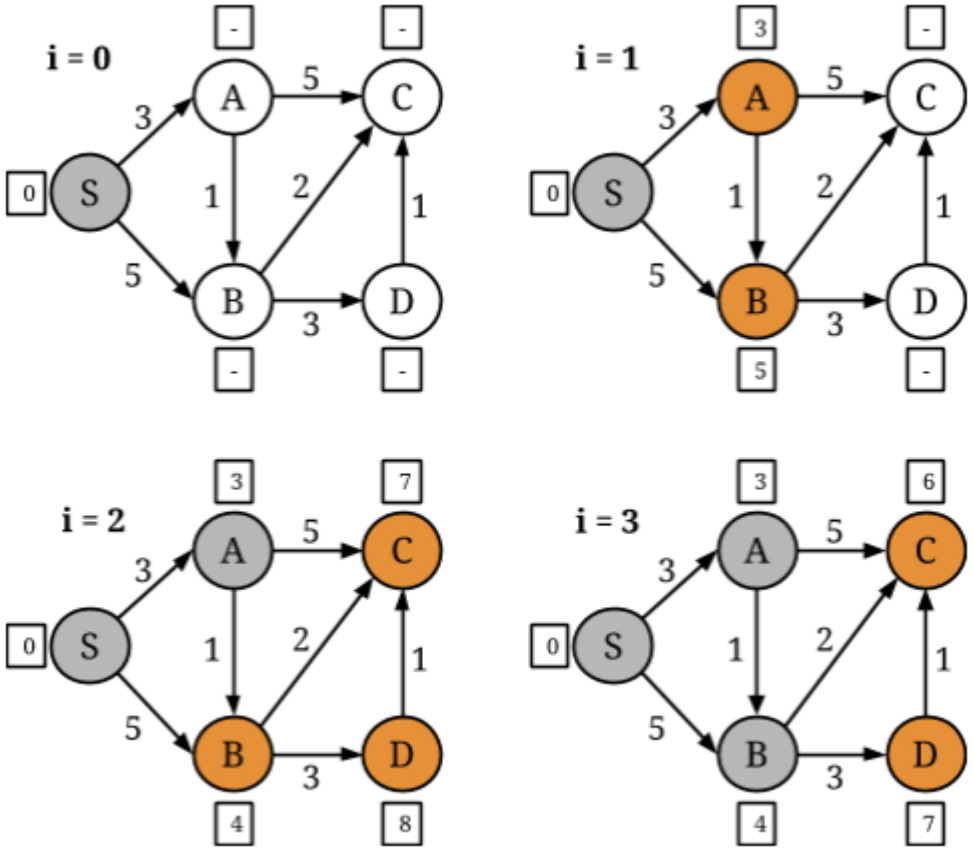


Figure 2: Maximum Value Example. Dotted lines are messages. Shaded vertices have voted to halt.

http://images.slideplayer.com/24/7406460/slides/slide_11.jpg

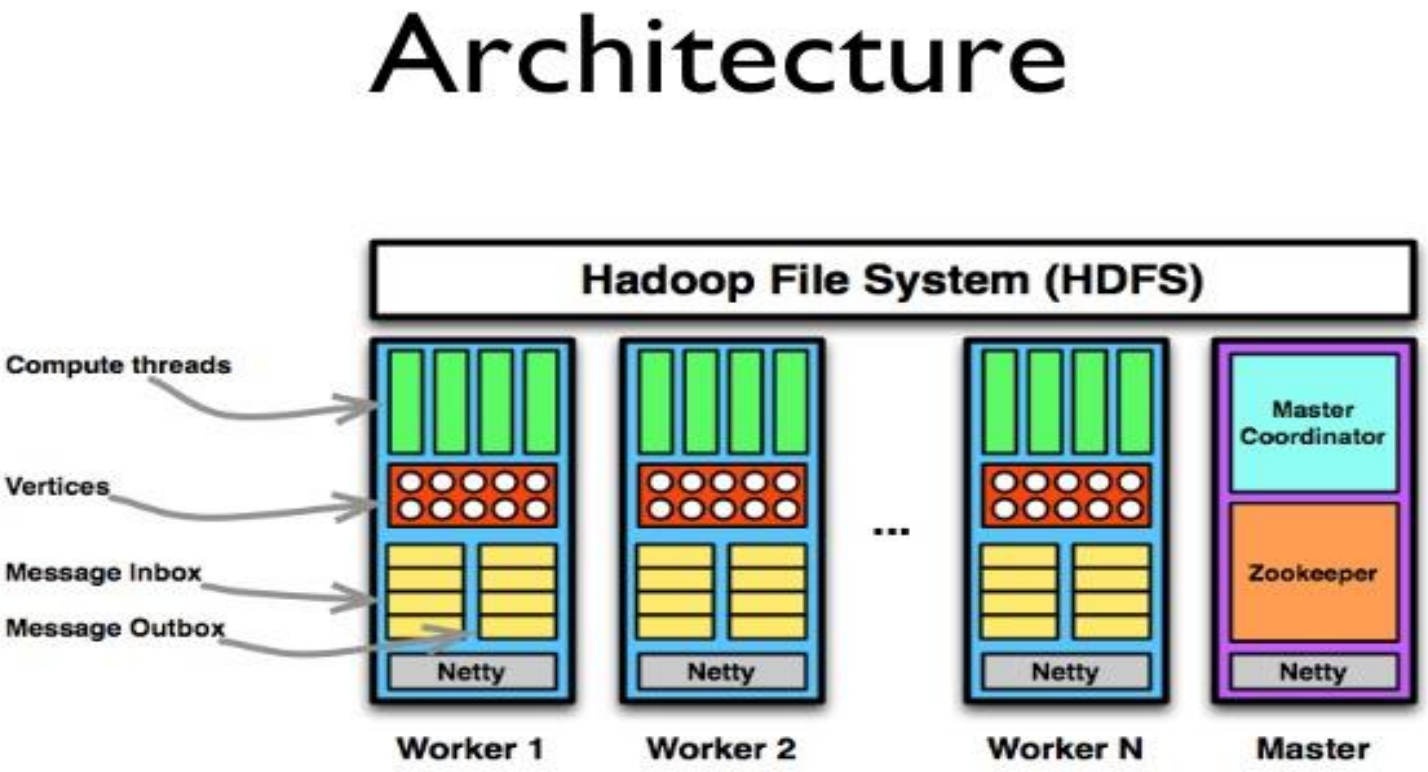
Single Source shortest path



SSSP:Code

```
  public void compute(Iterable<DoubleWritable> messages)
{
    double minDist = Double.MAX_VALUE;
    for (DoubleWritable message : messages) {
      minDist = Math.min(minDist, message.get());
    }
    if (minDist < getValue().get()) {
      setValue(new DoubleWritable(minDist));
      for (Edge<LongWritable, FloatWritable> edge :
getEdges()) {
        double distance = minDist +
edge.getValue().get();
        sendMessage(edge.getTargetVertexId(), new
DoubleWritable(distance));
      }
    }
    voteToHalt();
  }
```

Giraph Architecture



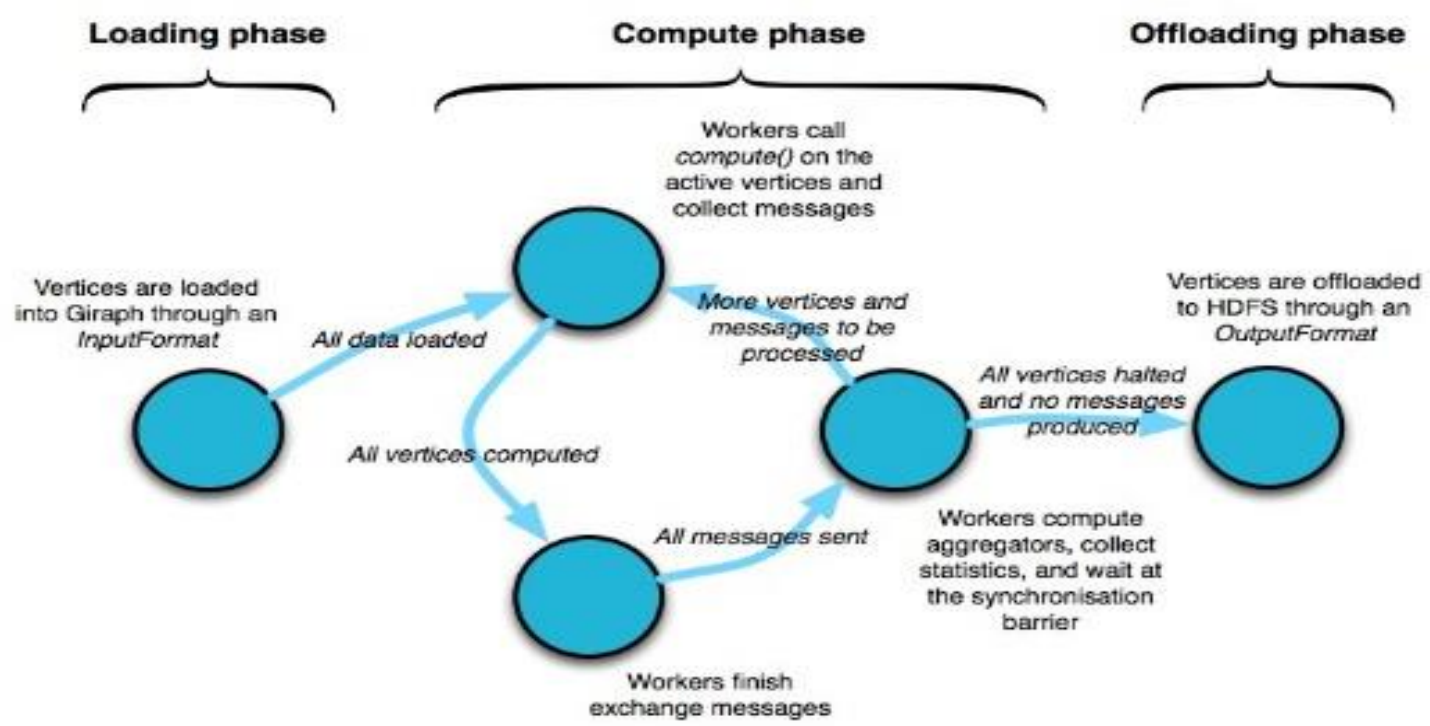
https://image.slidesharecdn.com/hadoopsummit2014-140403113102-phpapp02/95/giraph-at-hadoop-summit-2014-24-638.jpg?cb=1396526440

Giraph Architecture

- Master/Coordinator
  - Assigns partitions to workers,Synchronization
- Zookeeper
  - Keeps track of the computation state
- Netty
  - Java library used for messaging
- Workers
  - Operates on set of vertices called partitions
  - Invokes active vertices, sends/receive and assign messages

- Message Inbox: Messages received
- Message Outbox: Messages to be sent
- HDFS: Distributed file system reading initial graph

Giraph job lifetime



References

- Pregel: a system for large-scale graph processing, *SIGMOD 2010*
- Apache Giraph: Large Scale Graph Processing on Hadoop, Hadoop Summit 2014
- Distributed Graph Processing, SE256 *http://cds.iisc.ac.in/wp-content/uploads/L12.Pregel.pdf*