Data-enabled Science and Engineering: Scalable High Performance Data Analytics

ISL/NCSA Colloquium Series

March 14, 2016

Judy Qiu

Computer Science Department, Indiana University
E-mail: xqiu@indiana.edu

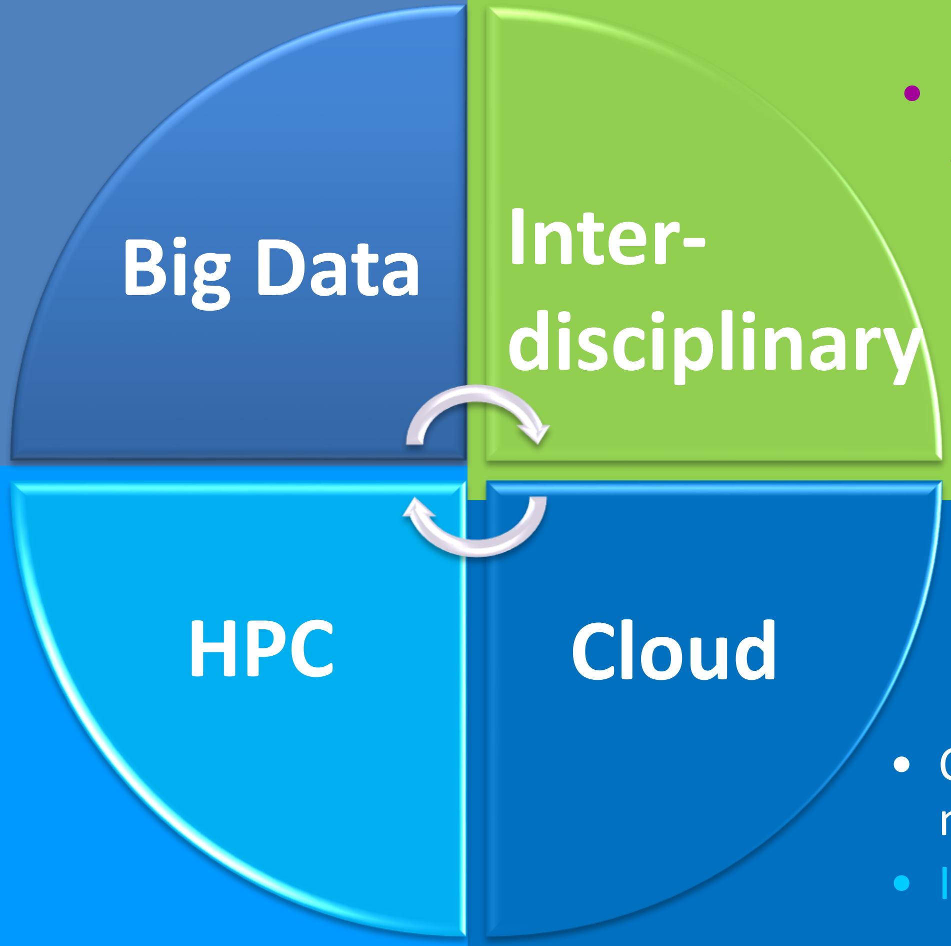
SALSA

Outline

1. Introduction: Big Data (Batch and Streaming), interdisciplinary, HPC and Clouds
2. Clouds are important for Big Data Analysis
3. Clouds are important for Education and Training
4. Interdisciplinary Applications and Technologies
5. Enhancing Commodity systems (Apache Big Data Stack) to HPC-ABDS
6. Summary and Future

- Impacts preservation, access/use, programming model
- Data Analysis/Machine Learning
- Batch and Stream Processing

- In all fields of science and daily life
- Health, social, financial, policy, national security, environment
- Better understanding the world surrounding us



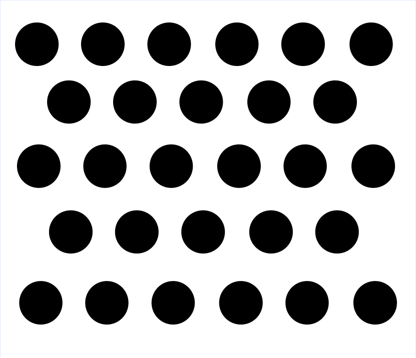
- Parallel computing is important
- Performance from Multicore (Manycore or GPU)

- Commercially supported data center model
- IaaS, PaaS, SaaS

What is Big Data ?

Big Data is defined by IBM as “any data that cannot be captured, managed and/or processed using traditional data management components and techniques.”

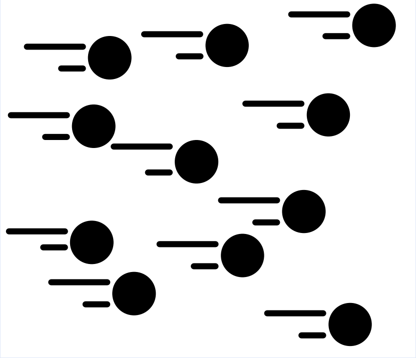
Volume



Data at Scale

Terabytes to Petabytes of Data

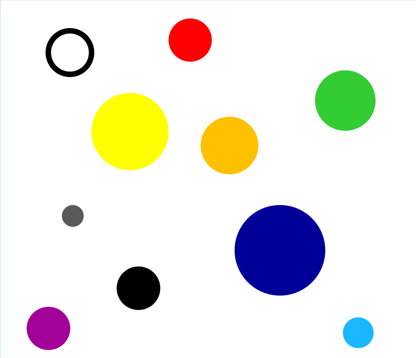
Velocity



Data in Motion

- Streaming data
- Real-time or near real-time to respond

Variety



Data in Many Forms

- Structured and unstructured data
- Text, numbers, and pixels

Veracity

Data Uncertainty

Inconsistent, incomplete, ambiguous, and approximated data

SALSA

Challenges and Opportunities

- Large-scale parallel simulations and data analysis drive scientific discovery across many disciplines

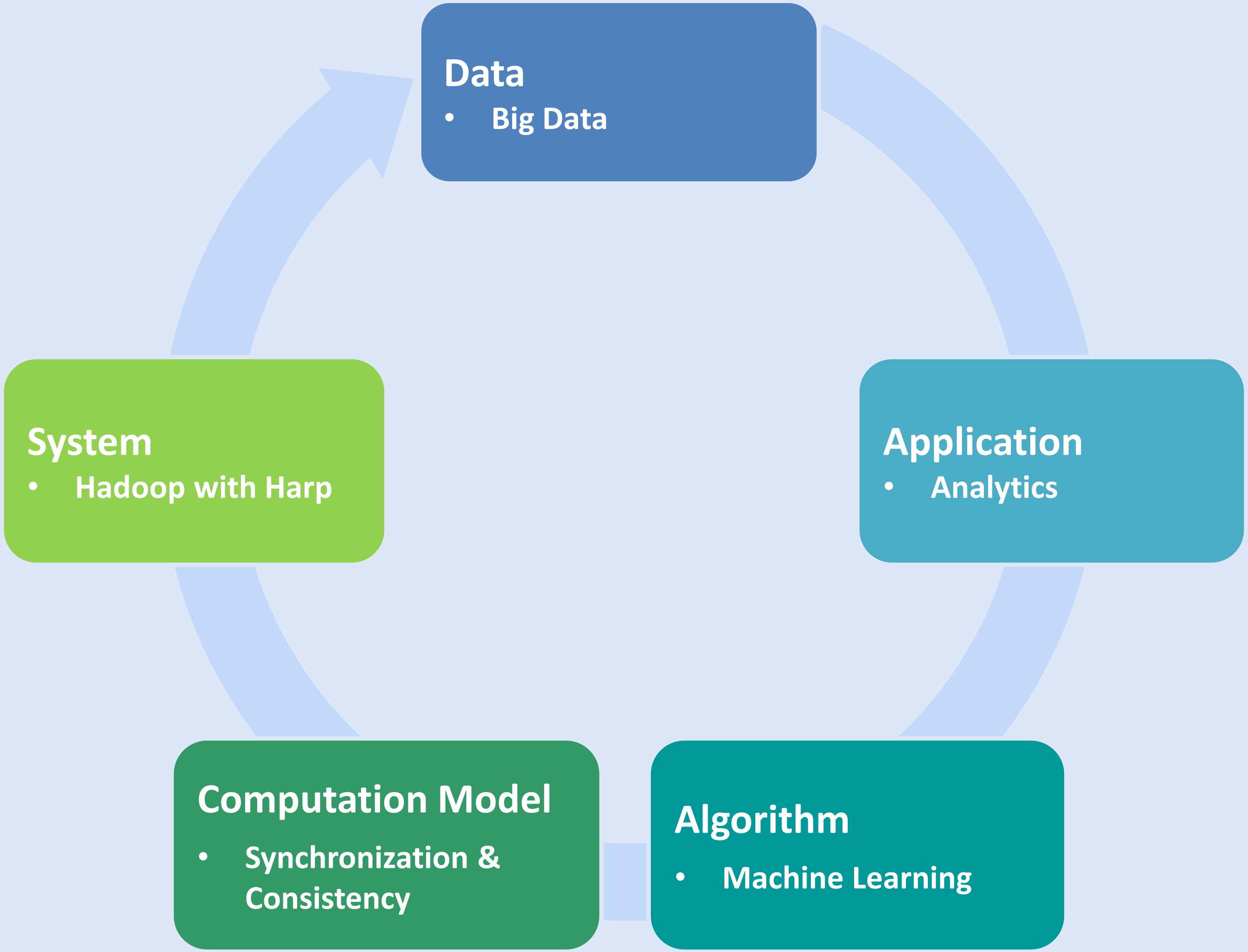
- Research a holistic approach that will enable performance portability to any machine, while increasing developer productivity and accelerating the advance of science

- Organize my research as **Data-Enabled Discovery Environments for Science and Engineering** (DEDESE)

— NSF Career: Map-Collective model for DEDESE and HPC-Cloud Integration

— DOE Workshop Report: Machine Learning and Understanding for Intelligent Extreme Scale Scientific Computing and Discovery, January 7-9, 2015

The System Solution to Big Data Problems



SALSA

Outline

1. Introduction: Big Data (Batch and Streaming), interdisciplinary, HPC and Clouds
2. Clouds are important for Big Data Analysis
3. Clouds are important for Education and Training
4. Interdisciplinary Applications and Technologies
5. Enhancing Commodity systems (Apache Big Data Stack) to HPC-ABDS
6. Summary and Future

SALSA

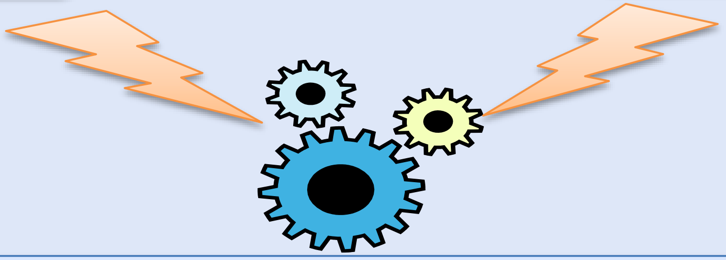
Motivation of Iterative MapReduce

MapReduce

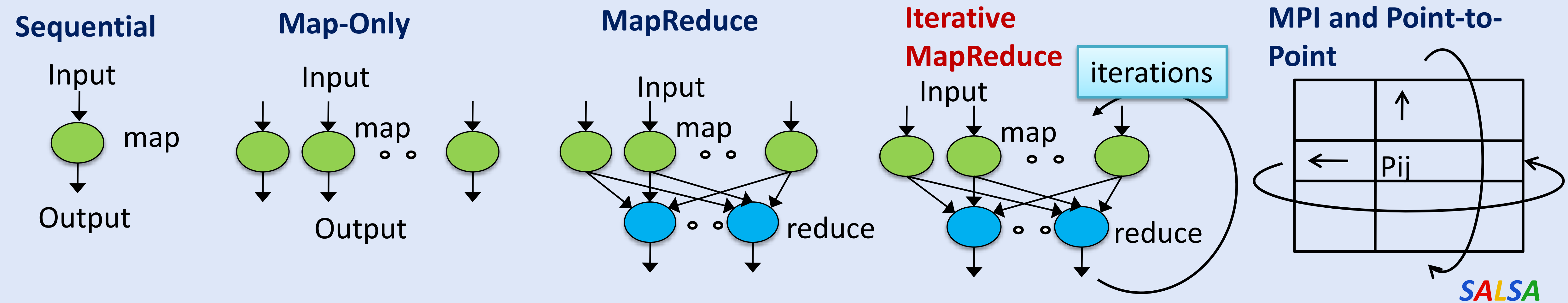
Data Centered, QoS

Classic Parallel Runtimes (MPI)

Efficient and Proven techniques



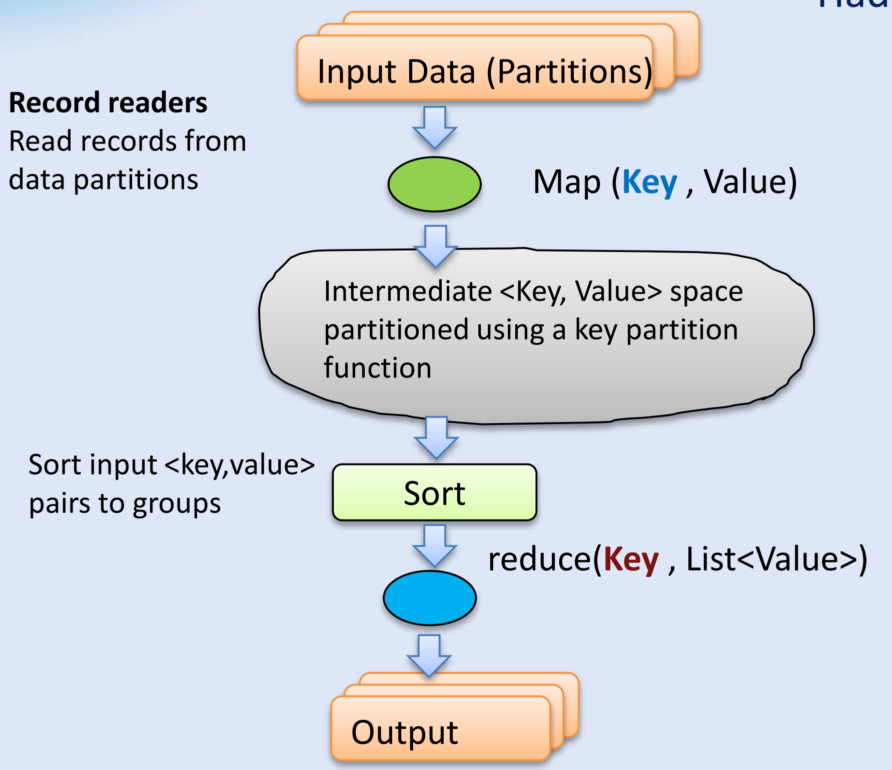
Expand the Applicability of MapReduce to more **classes** of Applications

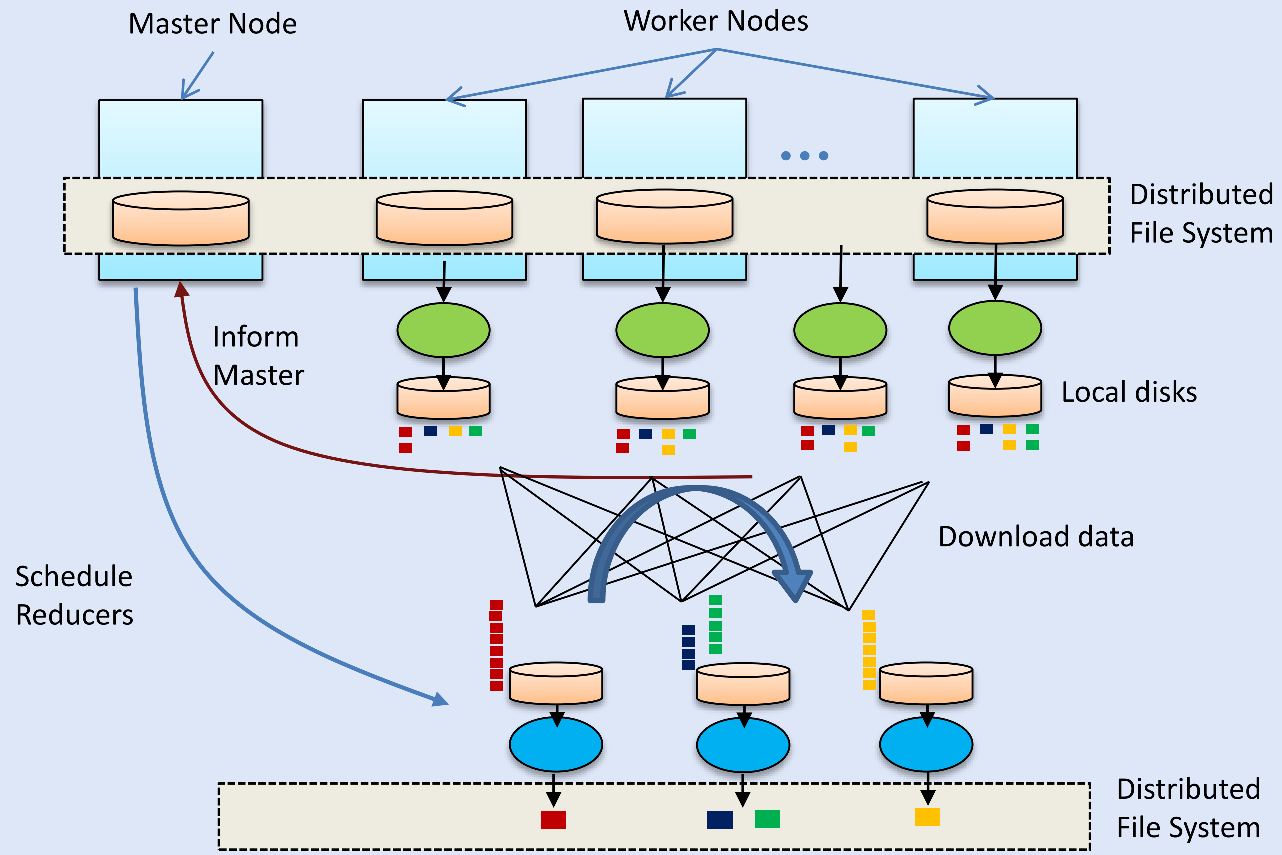


SALSA

MapReduce Programming Model & Architecture

Google MapReduce , Apache Hadoop

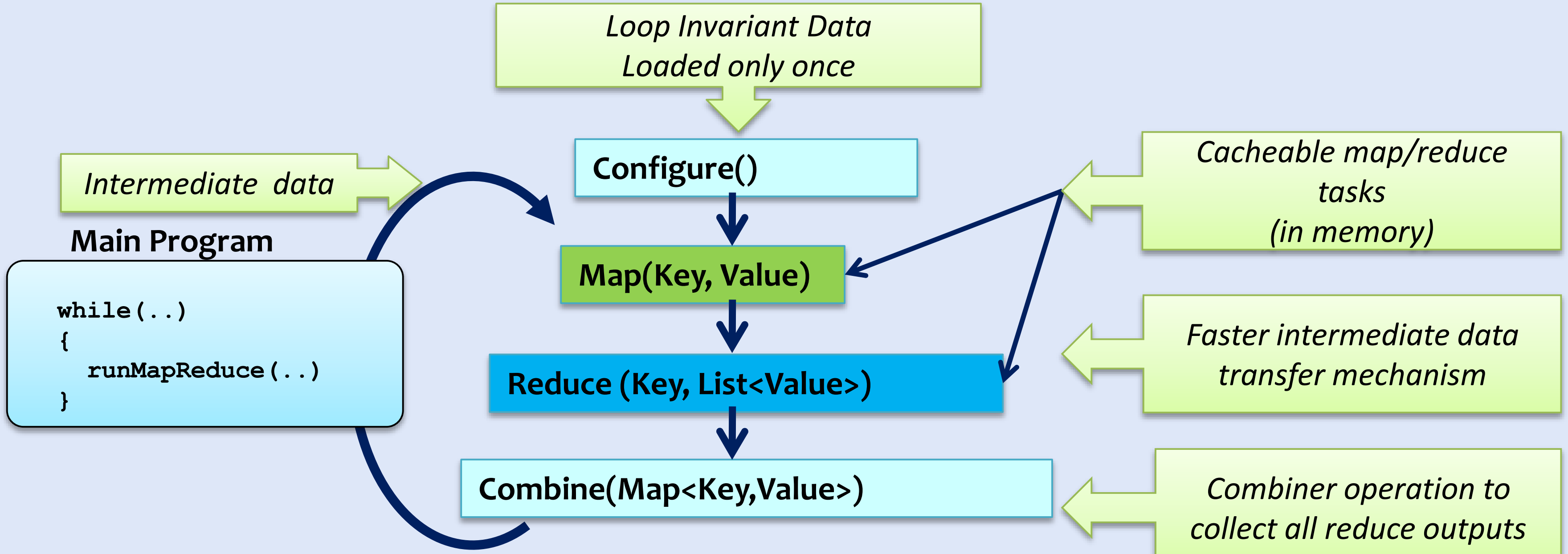




- Map(), Reduce(), and the intermediate key partitioning strategy determine the algorithm
- Input and Output => Distributed file system
- Intermediate data => **Disk -> Network -> Disk**
- Scheduling =>Dynamic
- Fault tolerance (Assumption: Master failures are rare)

SALSA

Programming Model for Iterative MapReduce



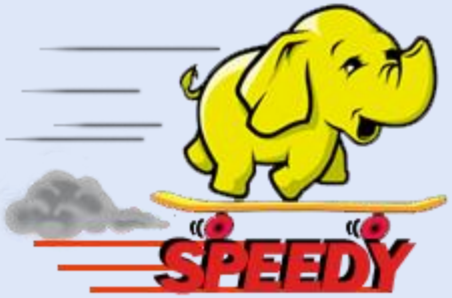
Twister was our initial implementation with first paper having 585 Google Scholar citations

Iterative MapReduce is a programming model that applies a computation (e.g. Map task) or function repeatedly, using output from one iteration as the input of the next iteration. By using this pattern, it can solve complex computation problems by using apparently simple (user defined) functions.

- Distinction on loop invariant (e.g. input) data and variable (e.g. intermediate) data
- Cacheable map/reduce tasks (in-memory)
- Combine operation

MapReduce Optimized for Iterative Computations

Twister
Cross Platform Iterative MapReduce



Twister: the speedy elephant

Abstractions

In-Memory
- Cacheable map/reduce tasks

Data Flow
- Iterative
- Loop Invariant
- Variable data

Thread
- Lightweight
- Local aggregation

Map-Collective
- Communication patterns optimized for large intermediate data transfer

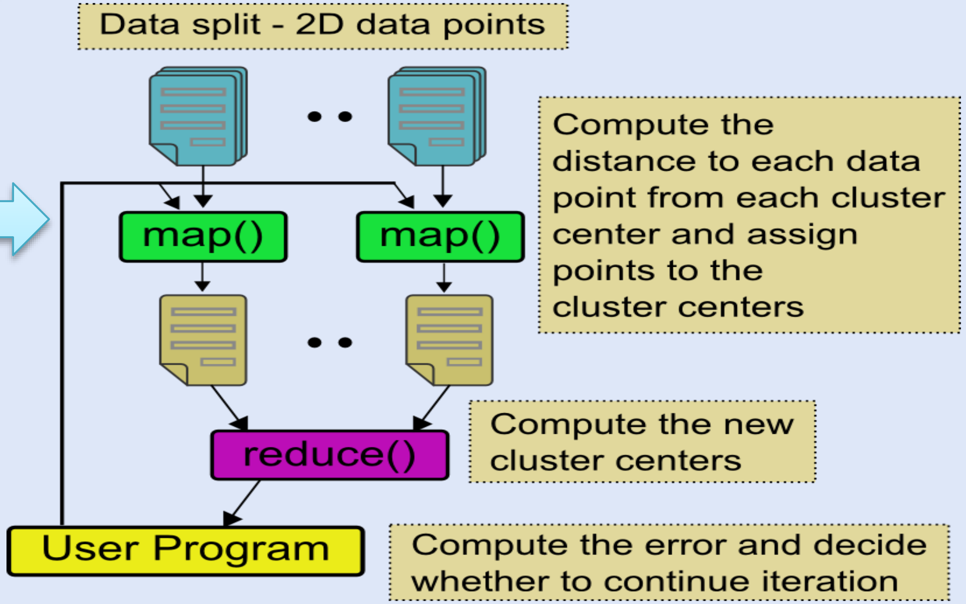
Portability
- HPC (Java)
- Azure Cloud (C#)
- Supercomputer (C++, Java)

- Microsoft has developed Daytona, an Iterative MapReduce runtime, which is based on Twister
- Twister4Azure is our prototype that demonstrates portability of Iterative MapReduce from HPC to PaaS/Azure Cloud Azure Queues for scheduling, Tables to store metadata and monitoring data, Blobs for input/output/intermediate data storage.

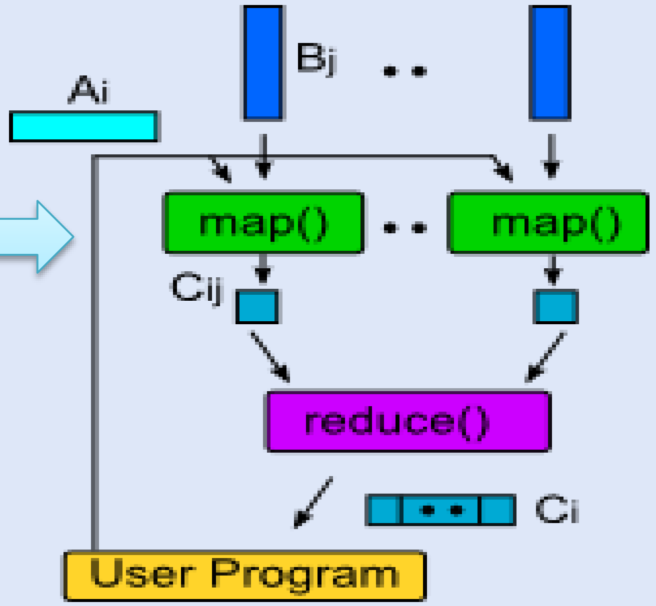
SALSA

Iterative Computations

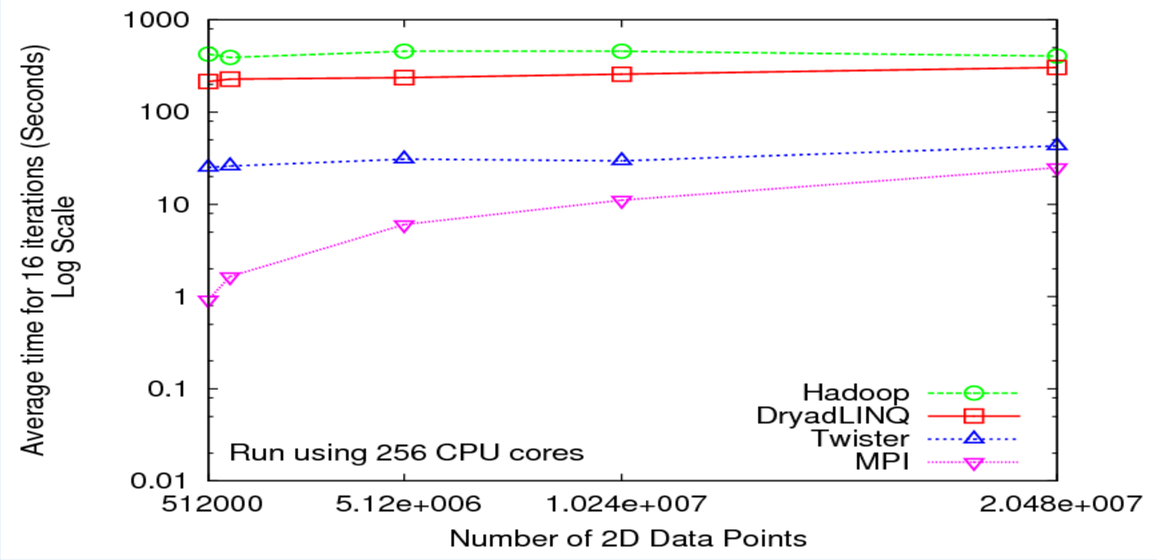
K-means



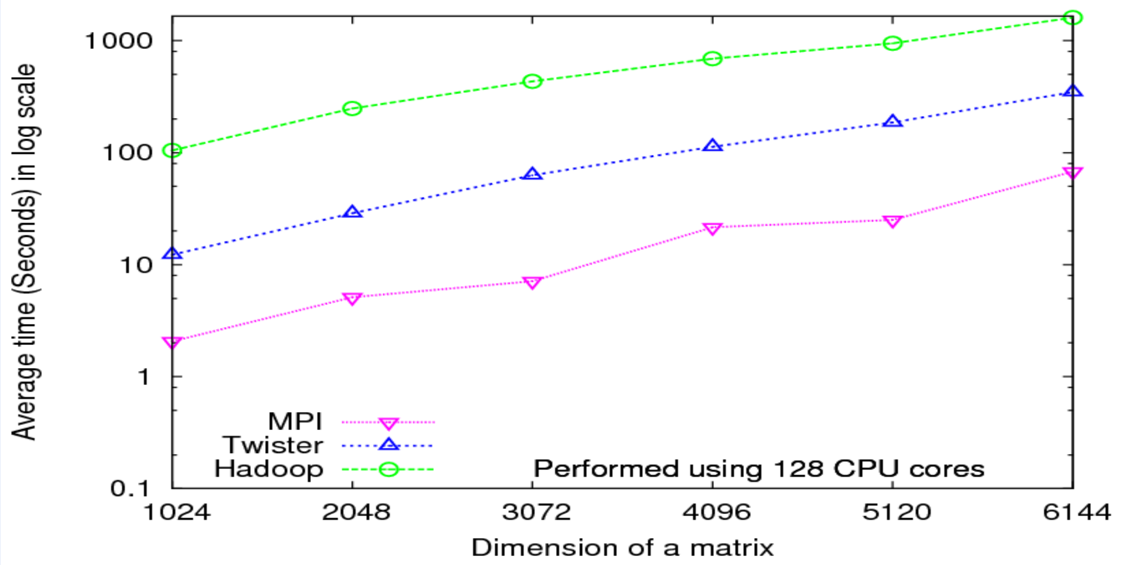
Matrix Multiplication



Performance of K-Means

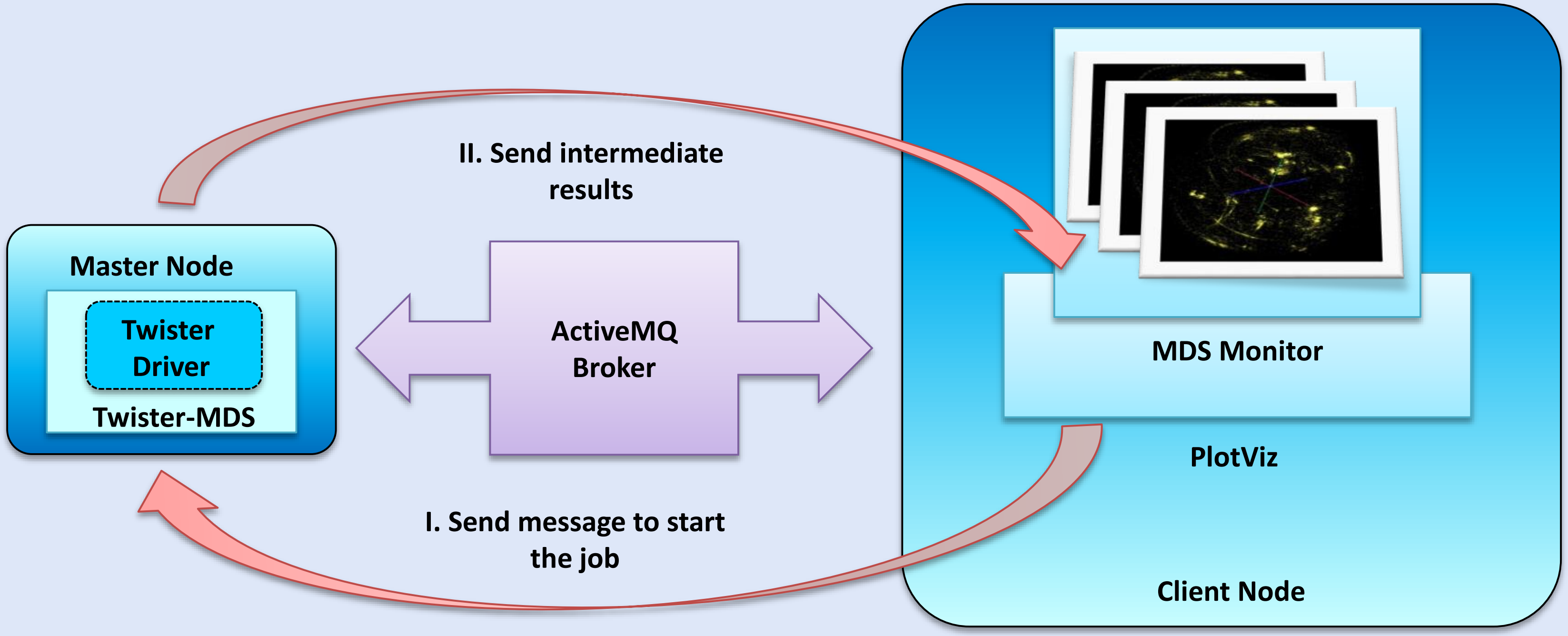


Parallel Overhead Matrix Multiplication



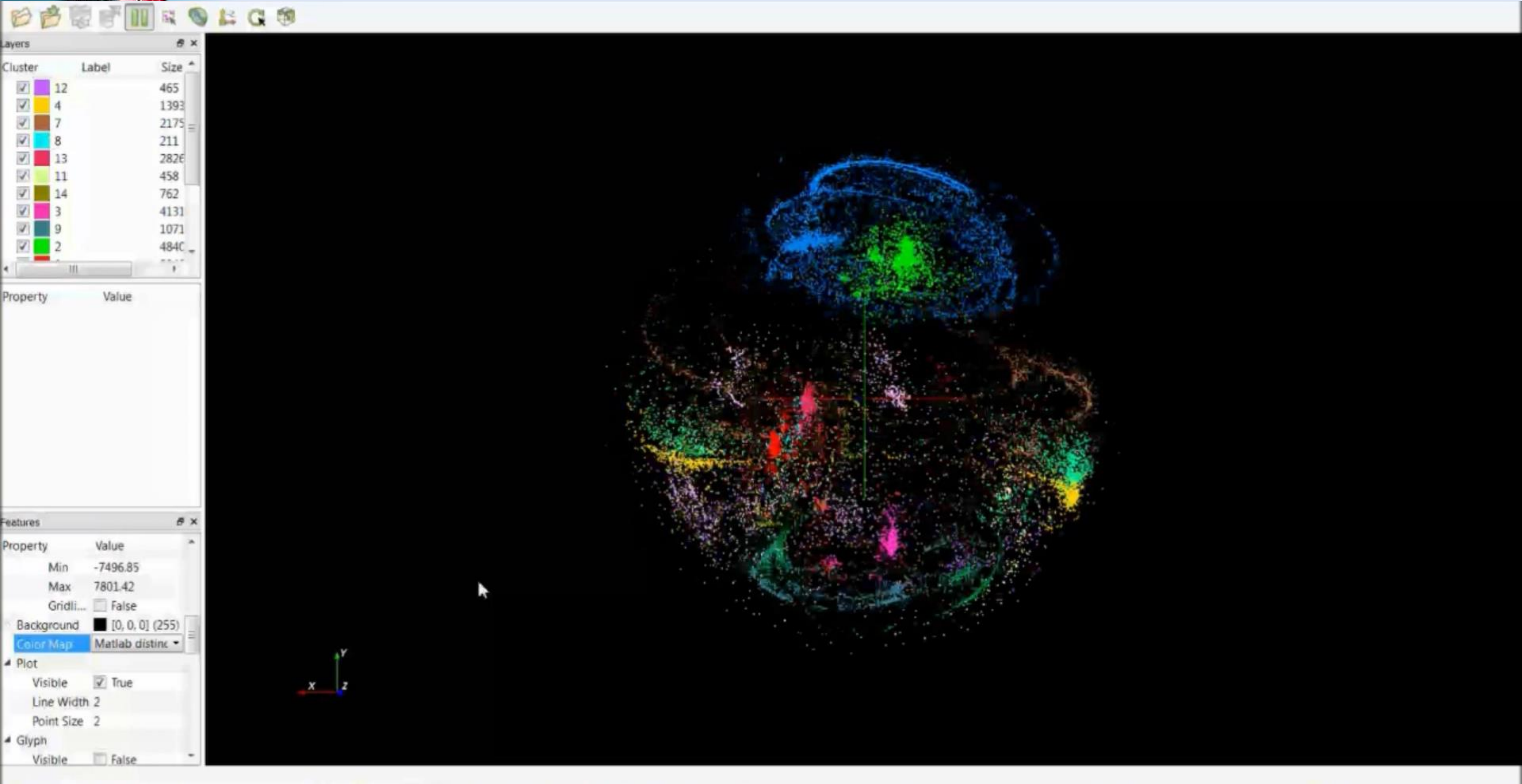
SALSA

Demo of Multi-Dimensional Scaling using Iterative MapReduce



- Input: 30K metagenomics data
- MDS reads pairwise distance matrix of all sequences
- Output: 3D coordinates visualized in PlotViz

Iterative MapReduce - MDS Demo



Outline

1. Introduction: Big Data (Batch and Streaming), interdisciplinary, HPC and Clouds
2. Clouds are important for Big Data Analysis
3. Clouds are important for Education and Training: CloudMOOC
4. Interdisciplinary Applications and Technologies
5. Enhancing Commodity systems (Apache Big Data Stack) to HPC-ABDS
6. Summary and Future

Cloud MOOC

PlotViz

Twister
Cross Platform Iterative MapReduce

Twister4Azure
Iterative MapReduce for Azure Cloud

Education and Training using Cloud

- McKinsey says that there will be up to **190,000 nerds** and **1.5 million extra managers** needed in Data Science by 2018 in USA
- Many more jobs than simulation (third paradigm) where **Computational Science** not very successful as curriculum
- Need curricula to educate people to use (or design) **Clouds** running **Data Analytics** processing **Big Data** to solve problems (e.g. health, social, financial, policy, national security, scientific experiment, environment)

Big Data for Science

Home Tutorials Contact

Big Data for Science Workshop

July 26-30, 2010, NCSA Summer School

300+ Students (200 on sites from 10 institutes; 100 online)
IU MapReduce and UF Virtual Applicance technologies are supported by FutureGrid.

Workshop Schedule
(In Central Time)

July 26

- **10:00AM** - Keynote: Data Intensive Computing
  Alex Szalay
  @The Johns Hopkins University
- **11:30AM** - Break (lunch for Eastern, Central time)
- **12:30PM** - Making the most of the I/O Software Stack
  Rob Latham
  @Argonne National Lab
- **2:00PM** - Break (lunch Mountain, Pacific time)
- **3:00PM** - Data movement & Storage (Data Capacitor WAN Filesystem)
  Justin Miller
  @Indiana University
- **4:00PM** - Scalable and Distributed Visualization using Paraview
  Eric Wernert
  @Indiana University
- **5:30PM** - Local Reception

July 27

Join FutureGrid at Science Cloud Summer School

Get immersive, hands-on training in the use of cloud computing technologies in science. Training will concentrate on application and computer science.

When: **July 30-August 3, 2012**
Participants: Graduate students, post-docs, and professionals
To register, visit: *hub.vscse.org*
For more information, see: *sciencecloudsummer2012.tumblr.com*

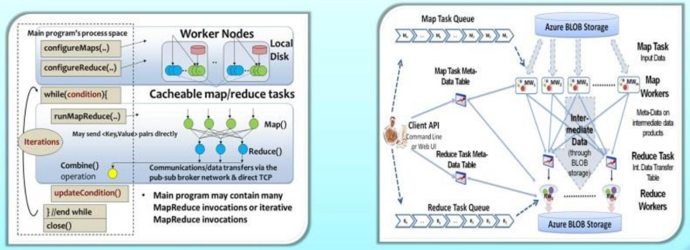
Iterative MapReduce Enabling HPC-Cloud Interoperability

Scientific Applications

Iterative MapReduce Runtime

Twister
Cross Platform Iterative MapReduce

Twister4Azure
Iterative MapReduce for Azure Cloud



Large Scale Infrastructure

HPC Cluster
Windows Azure
amazon web services™

Cloud Computing

The course covers all aspects of the cloud architecture stack, from Software as a Service (large-scale biology and graphics applications), Platform as a Service (MapReduce (Hadoop), Iterative MapReduce (Twister) and NoSQL (HBase)), to Infrastructure as a Service (low-level virtualization technologies).

Class Summary

In this course you will learn basic concepts in Cloud Computing. You will learn how to write your own software using key cloud programming models and tools to support data mining and data analysis applications.

What Should I Know?

General programming experience with Windows or Linux using Java and scripts is required. A background in parallel and cluster computing is a plus, although not necessary.

What Will I Learn?

At the end of this course, you will have learned key concepts in cloud computing and enough programming to be able to solve data analysis problems on your own.

Class Projects

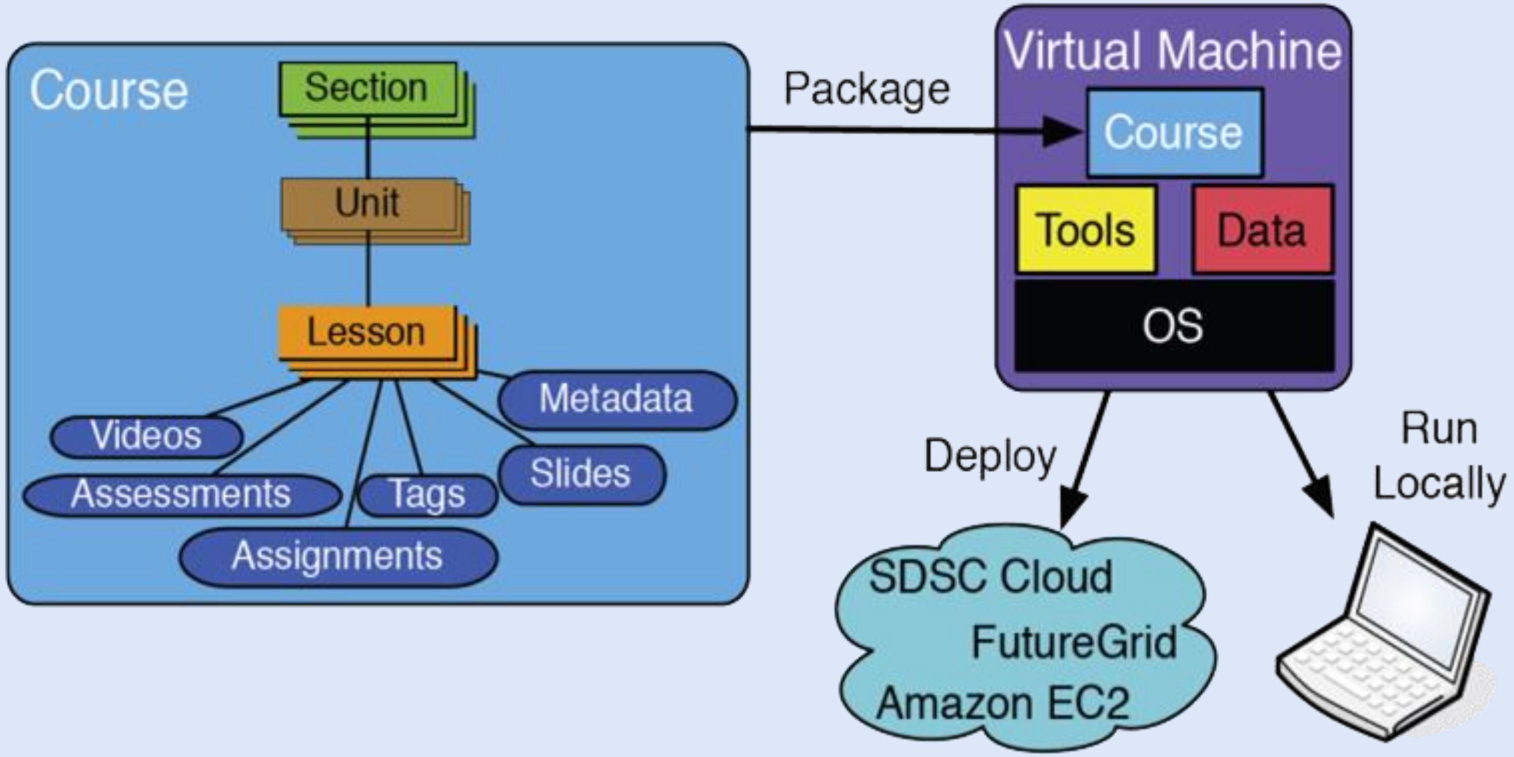
The class has several projects that will allow students to get firsthand experience with the technologies taught here. Projects are performed on VirtualBox Appliances or academic clouds like FutureGrid.

Instructor

Judy Qiu

Judy Qiu is an Assistant Professor in the School of Informatics and Computing at Indiana University. Her research interests focus on data-intensive computing at the intersection of cloud and multicore

Biomedical Big Data Training Collaborative



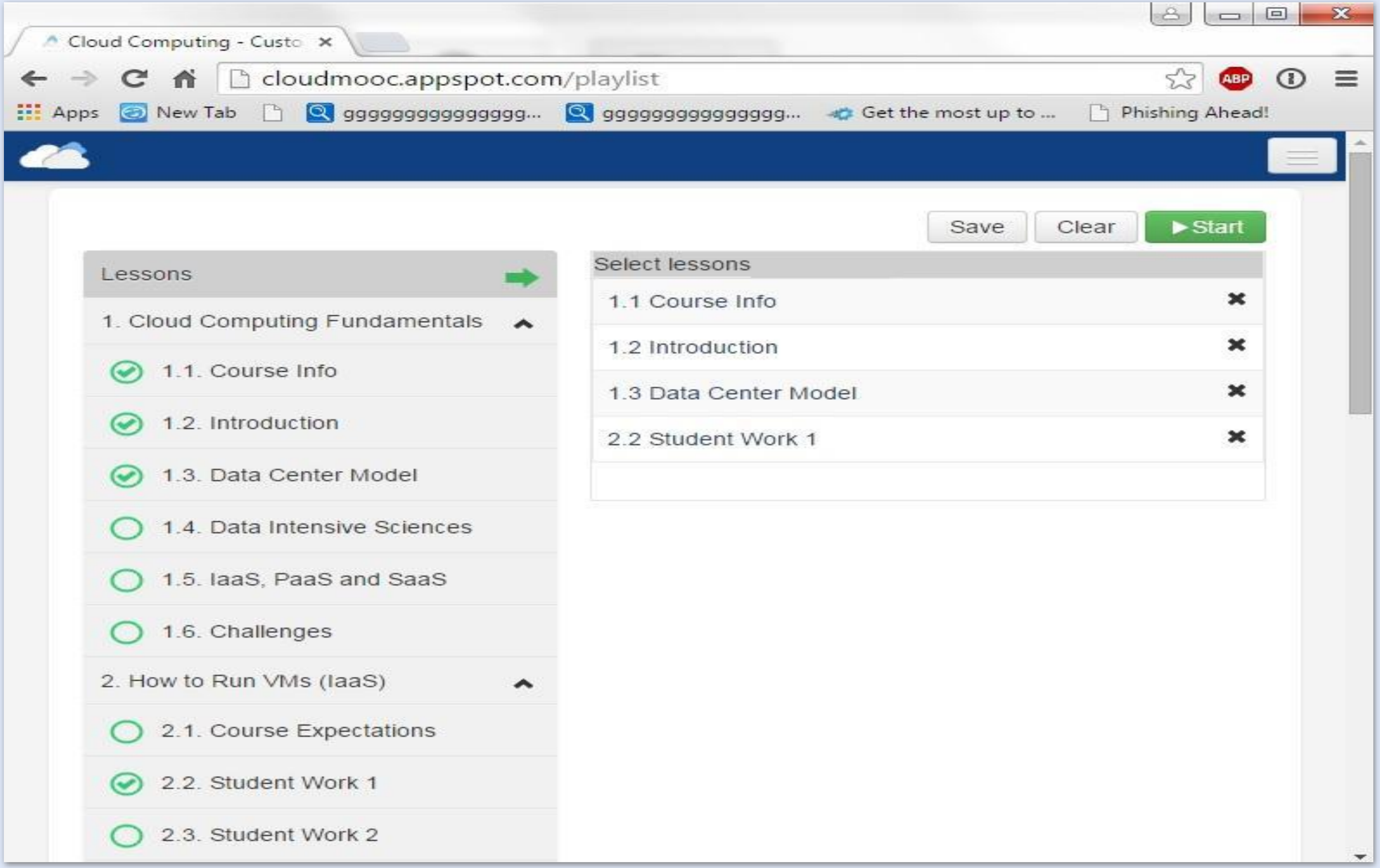
An open online training framework

- No single group or strategy that will be able to cover the full spectrum of educational needs required to comprehensively train biomedical big data researchers

- Building a community repository, and creating lecture content and example courses with hands-on virtual machines for biomedical big data training

SALSA

Customization using Playlist for Cloud Computing MOOC



The playlist feature is demonstrated in our CloudMOOC course, which teaches cloud computing and includes topics like Hadoop, OpenStack and NoSQL databases. This figure shows that a student can simply drag and drop course modules (left) to make a playlist of lessons (right).

SALSA

Curriculum Development

- Data-enabled Science covers Data curation and management, Analytics (Algorithms), Runtime (e.g. MapReduce, Workflow, NoSQL), Visualization for Applications
- Some courses aimed at one aspect of this; our courses cover integration and link to applications
- Look at Massive Open Online Courses (MOOCs) to support online modules that can be used by other universities; initially at ECSU and other HBCU
- 3 funded collaborative curriculum developments using MOOCs
  - Data Science - CloudMOOC (Google Course Builder)
  - Biomedical training community repository - NIH/MOOC (NIH)
  - HBCU-STEM curriculum development - HBCU (NSF) starts Fall 2015

Remote Sensing Curriculum Enhancement using Cloud Computing

ECSU-IU collaboration in environmental applications of Microwave Remote Sensing using Cloud Computing technology.

Demonstrate the concept that Data and Computational Science (remote sensing) curriculum can drive new workforce and research opportunities at Minority Serving Institutions (MSI) by exploiting enhancements using Cloud Computing technology.

We will explore multiple targeted courses built from this repository of shared customizable lessons.

SALSA

Outline

Cloud MOOC

PlotViz

Twister
Cross Platform Iterative MapReduce

Twister4Azure
Iterative MapReduce for Azure Cloud

Harp

IndexedHBase

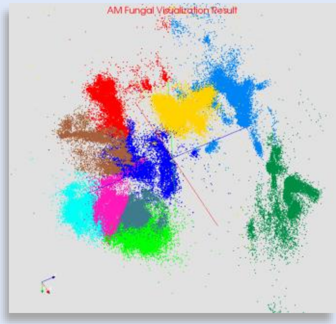
1. Introduction: Big Data (Batch and Streaming), interdisciplinary, HPC and Clouds
2. Clouds are important for Big Data Analysis
3. Clouds are important for education: CloudMOOC
4. **Interdisciplinary Applications and Technologies**
5. Enhancing Commodity systems (Apache Big Data Stack) to HPC-ABDS
6. Summary and Future

SALSA

Large Scale Data Analysis Applications

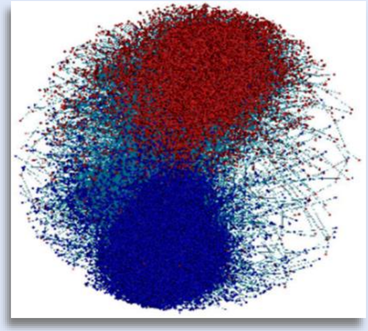
Case Studies

- Bioinformatics: Multi-Dimensional Scaling (MDS) on gene sequence data
- Computer Vision: Kmeans Clustering on image data (high dimensional model data)
- Text Mining: LDA on wikipedia data (dynamic model data due to sampling)
- Complex Network: Online Kmeans (streaming data)
- Deep Learning: Convolutional Neural Networks on image data



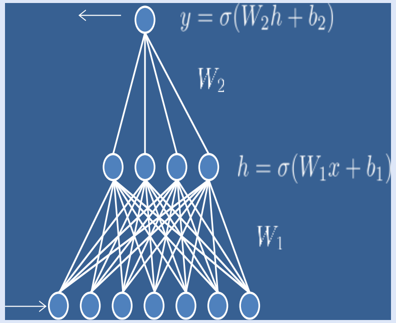
Bioinformatics

Computer Vision



Complex Networks

Text Mining



Deep Learning

4. Interdisciplinary Applications and Technologies

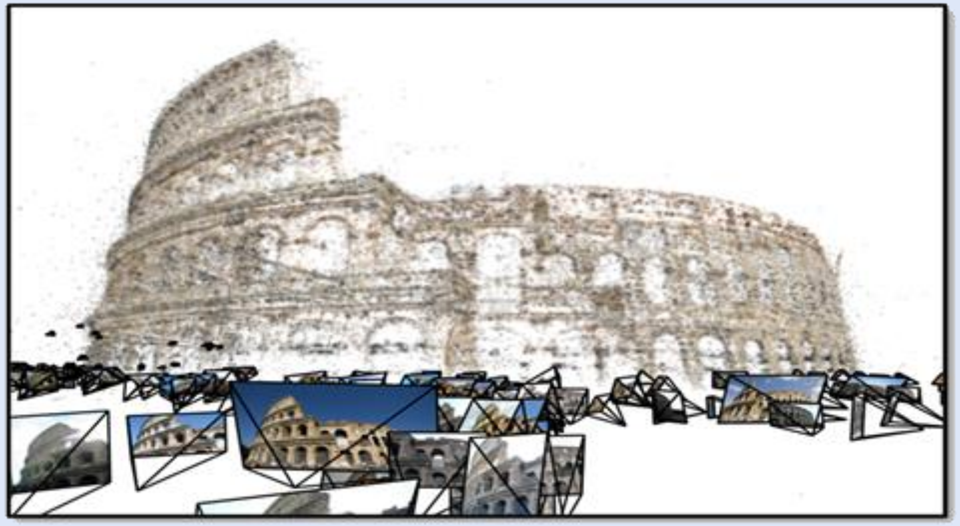
Case Study 1:
High Dimensional Image Data Clustering

Map Collective Computing Paradigm

SALSA

Data Intensive Batch Kmeans Clustering

Image Classification: ***7 million images;*** 512 features per image; 1 million clusters
10K Map tasks; 64G broadcasting data (1GB data transfer per Map task node);
20 TB intermediate data in shuffling.



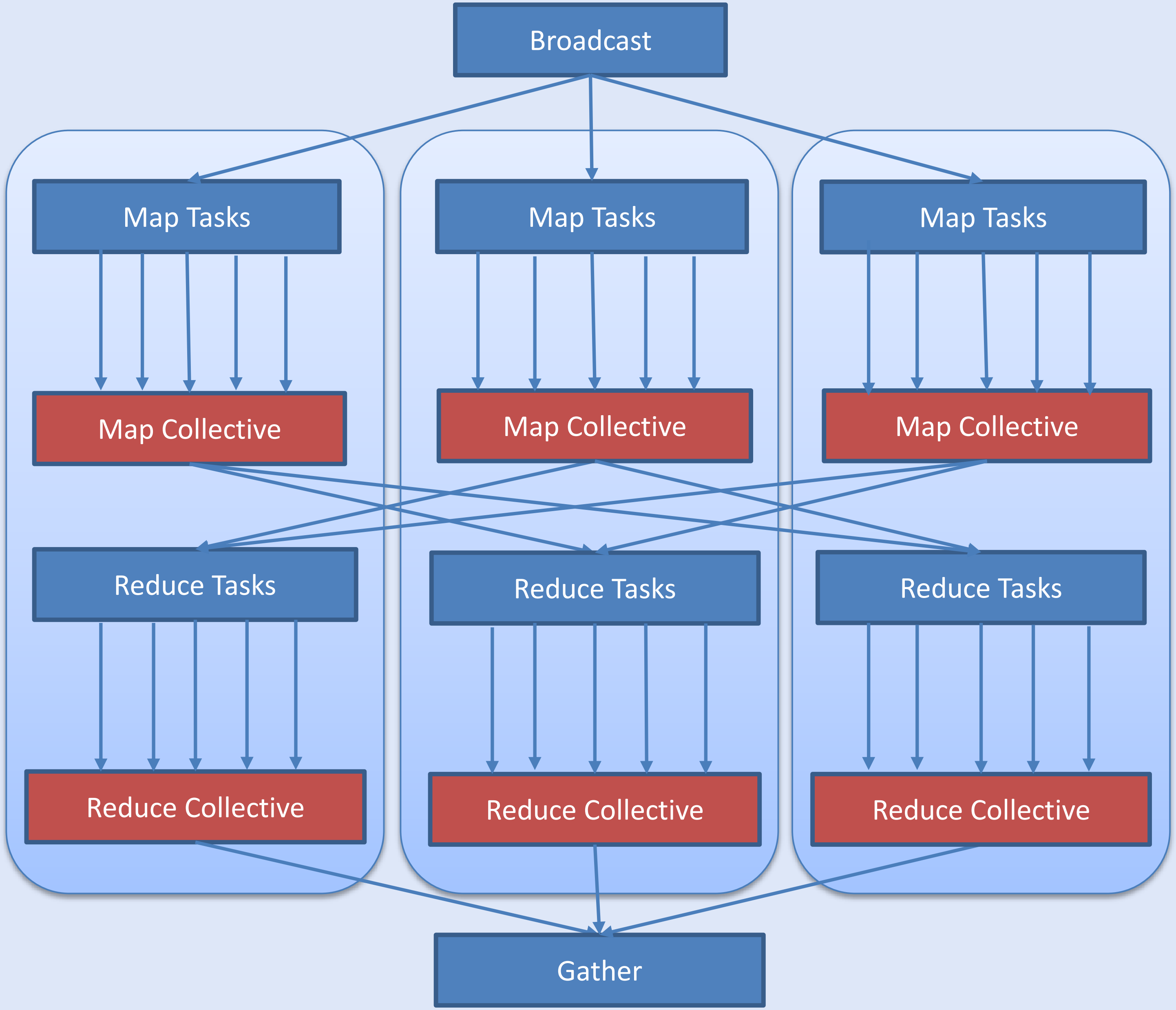
Collaborative work with Prof. David Crandall

High Dimensional Image Data

- K-means Clustering algorithm is used to cluster the images with similar features.
- In image clustering application, each image is characterized as a data point (vector) with dimension in range 512 - 2048. Each value (feature) ranges from 0 to 255.
- Around 180 million vectors in full problem
- Currently, we are able to run K-means Clustering up to 1 million clusters and 7 million data points on 125 computer nodes.
  - 10K Map tasks; 64G broadcast data (1GB data transfer per Map task node);
  - 20 TB intermediate data in shuffling.

Twister Collective Communications

- Broadcasting
  - Data could be large
  - Chain & MST
  - Gather scatter
  - Local global sync
  - Rotation
- Map Collectives
  - Local merge
- Reduce Collectives
  - Collect but no merge
- Combine
  - Direct download or Gather



SALSA

High Performance Data Movement

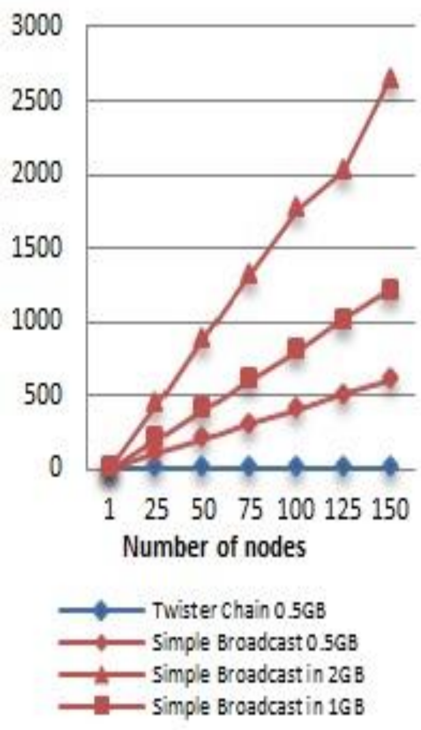


Figure 5. Twister Chain vs. Simple Broadcasting

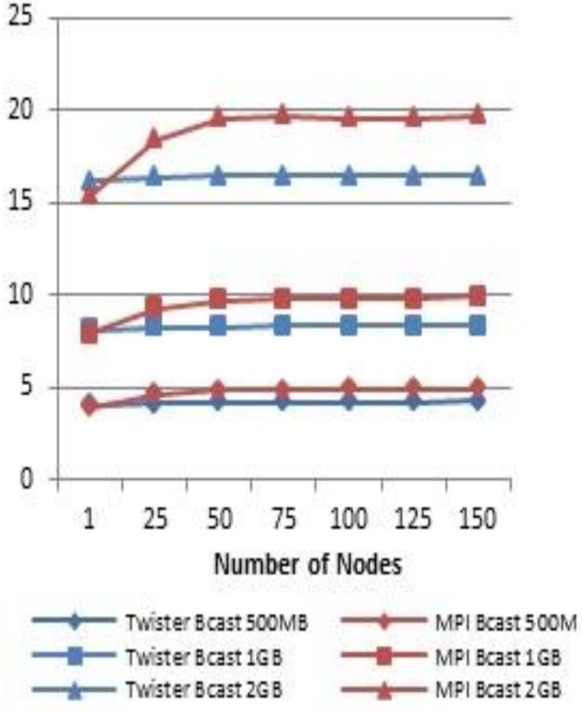


Figure 6. Twister vs. MPI (Broadcasting 0.5~2GB data)

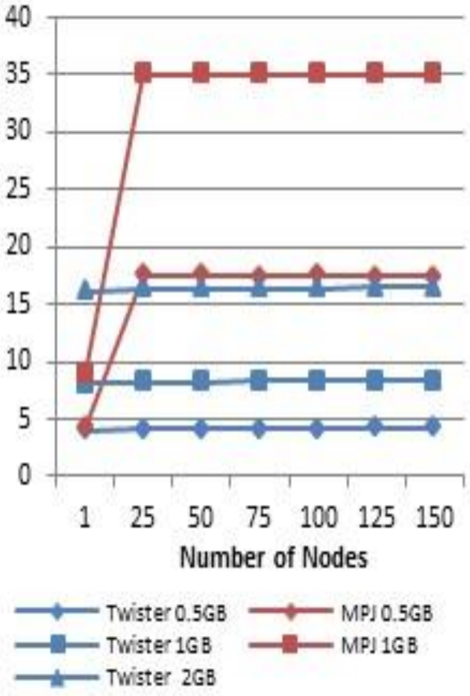


Figure 7. Twister vs. MPJ (Broadcasting 0.5~2GB data)

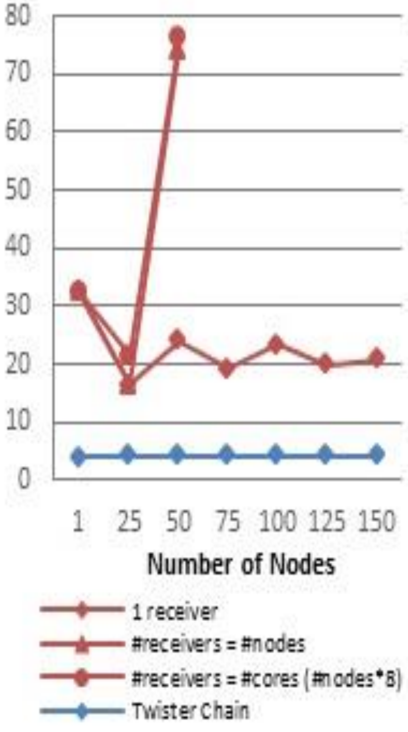


Figure 8. Twister vs. Spark (Broadcasting 0.5GB data)

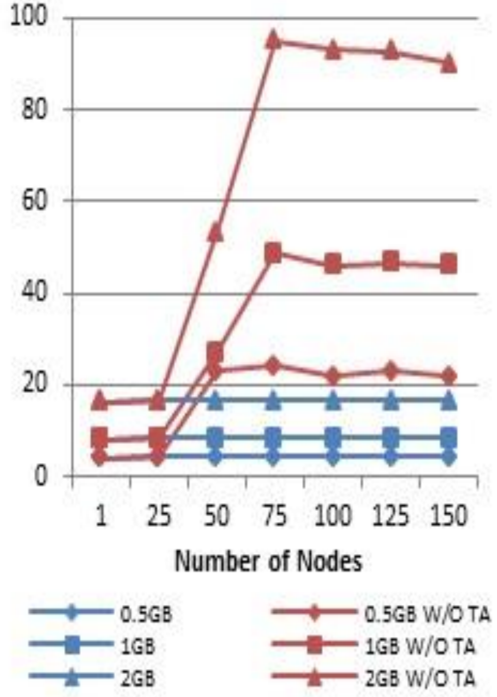


Figure 9. Twister Chain with/without topology-awareness

Tested on IU Polar Grid with 1 Gbps Ethernet connection

- At least a factor of 120 on 125 nodes, compared with the simple broadcast algorithm
- The new topology-aware chain broadcasting algorithm gives 20% better performance than best C/C++ MPI methods (four times faster than Java MPJ)
- A factor of 5 improvement over non-optimized (for topology) pipeline-based method over 150 nodes

K-means Clustering Parallel Efficiency

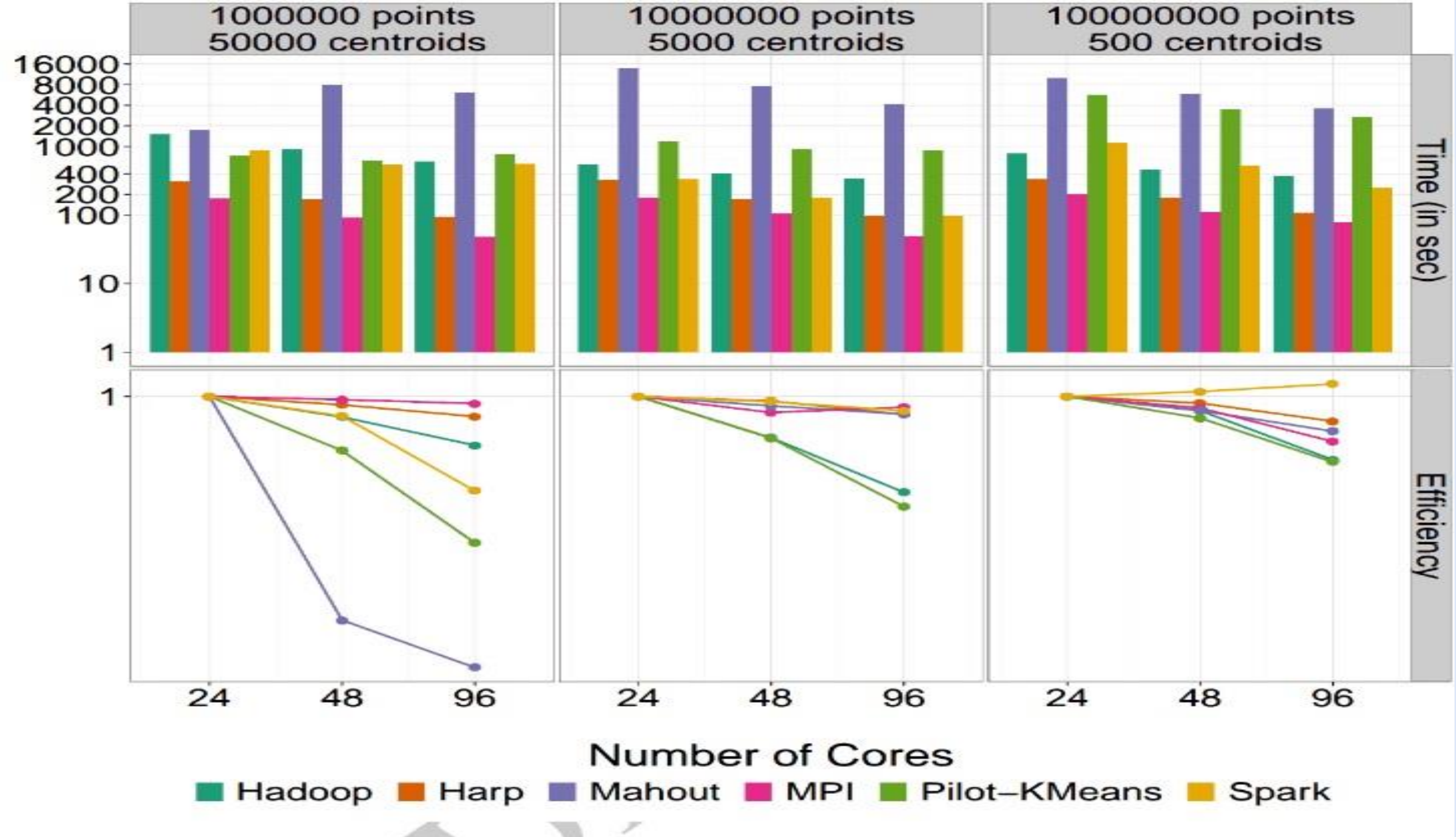


Fig. 5. Time-To-Completion for KMeans on Different Backends

Shantenu Jha et al. A Tale of Two Data-Intensive Paradigms: Applications, Abstractions, and Architectures. 2014.

SALSA

Map Collective Computing Paradigm

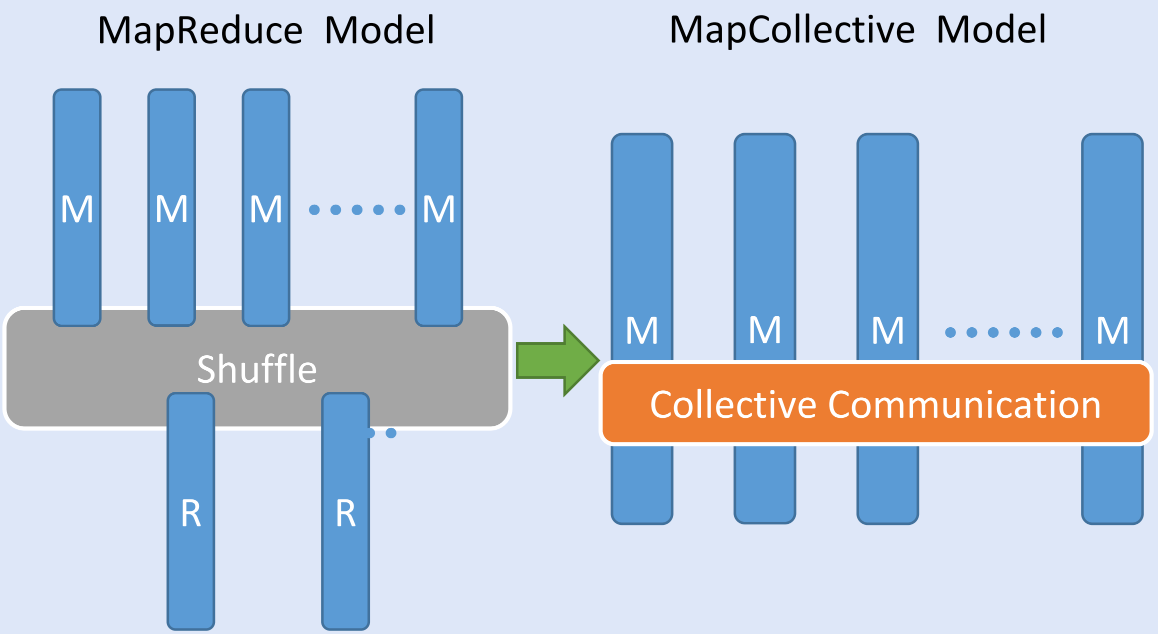
Harp
Spark
Parameter Server

Why Collective Communications for Big Data Processing?

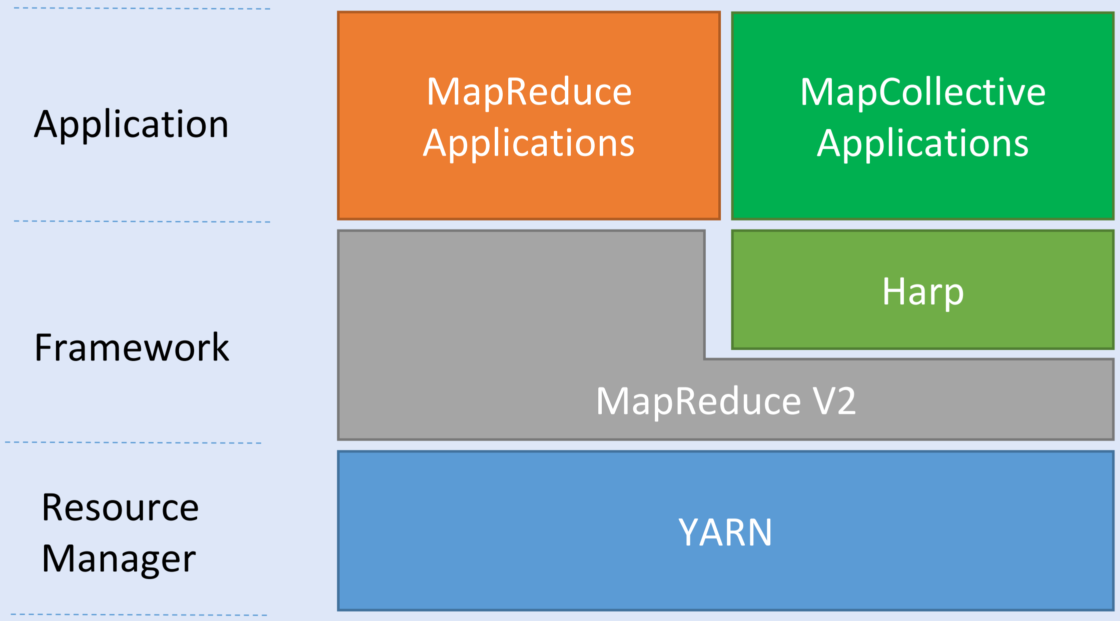
- **Collective Communication and Data Abstractions**
  - Our approach to optimize data movement
  - Hierarchical data abstractions and operations defined on top of them
- **Map-Collective Programming Model**
  - Extended from MapReduce model to support collective communications
  - Two Level of BSP parallelism
- **Harp Implementation**
  - A plug-in to Hadoop
  - Component layers and the job flow

The Concept of Harp Plug-in

Parallelism Model

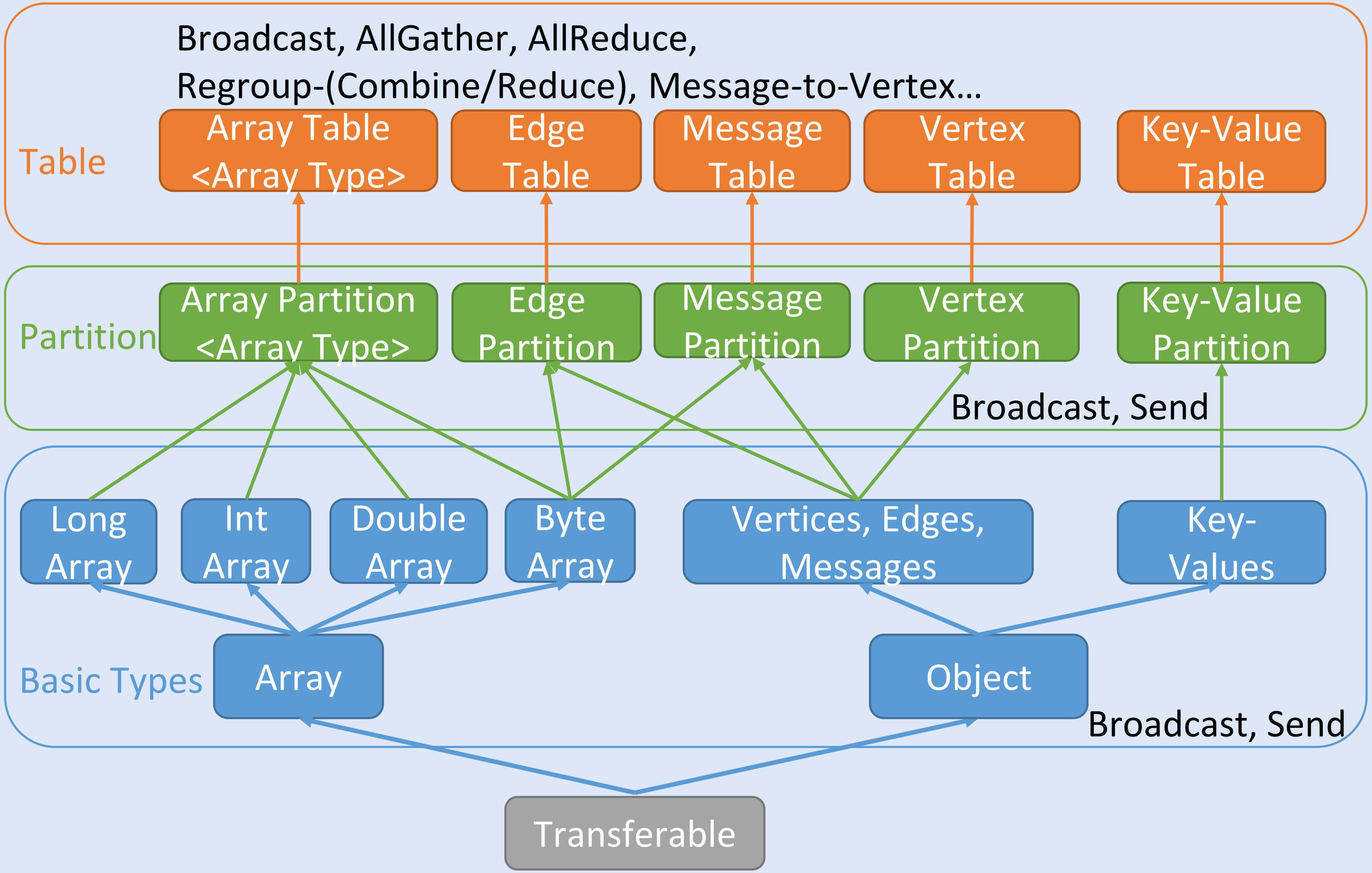


Architecture



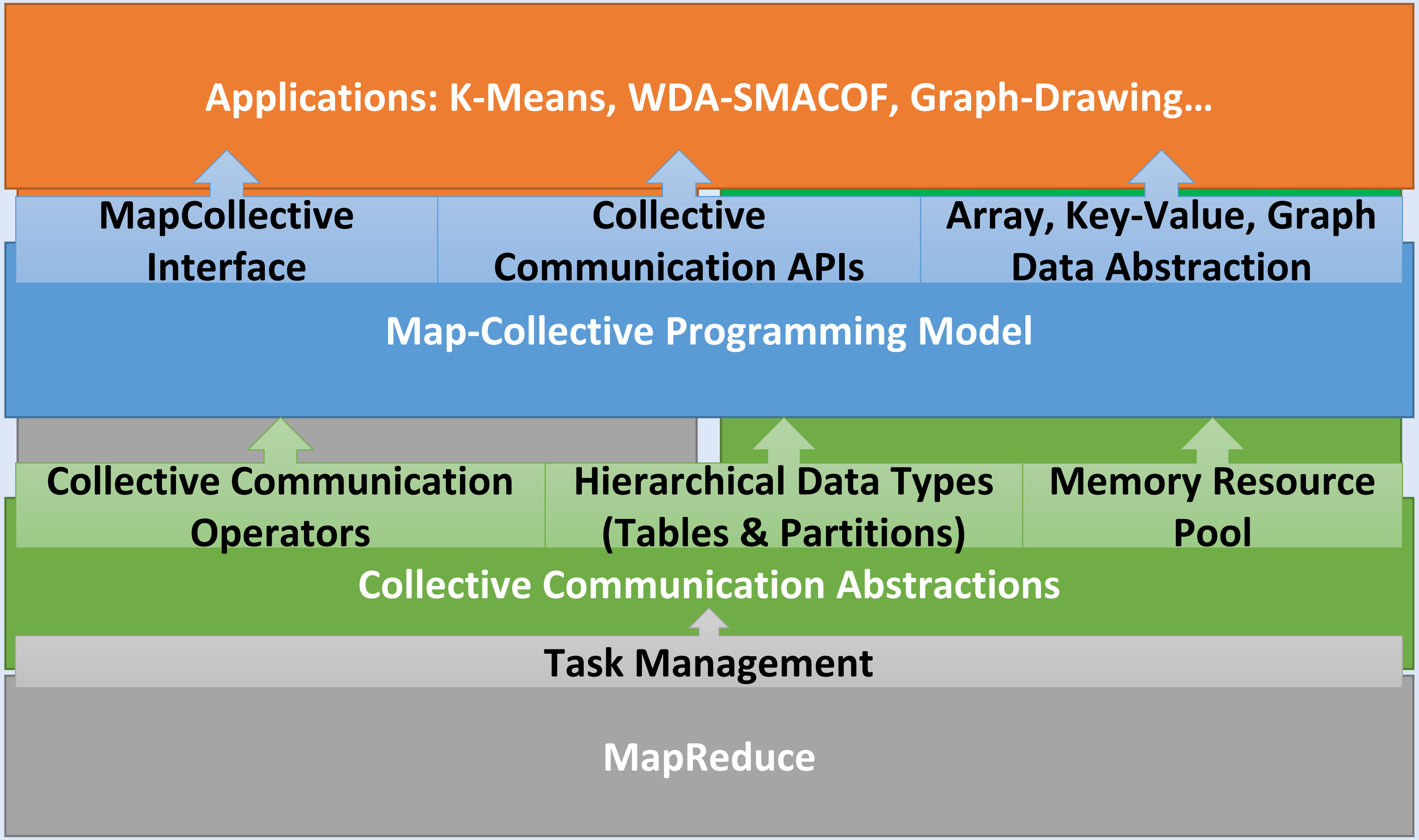
SALSA

Hierarchical Data Abstraction



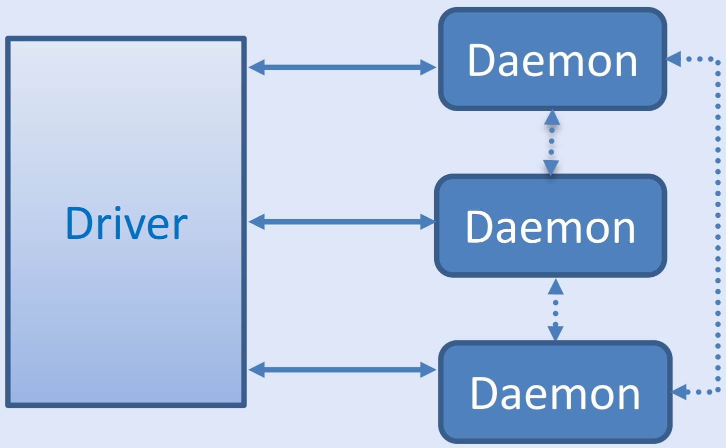
SALSA

Harp Component Layers



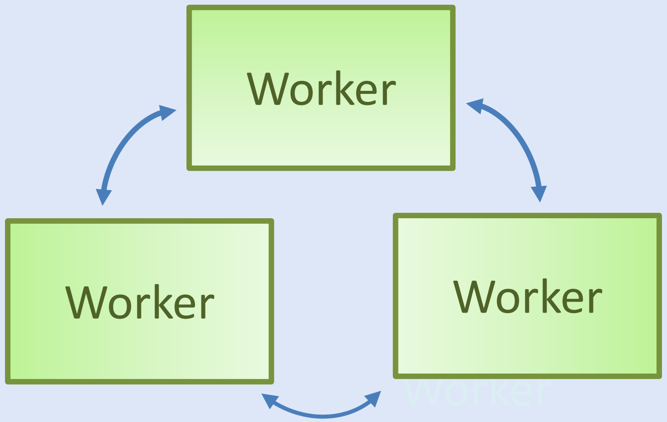
Comparison of Iterative Computation Tools

Spark



- Implicit Data Distribution
- Implicit Communication

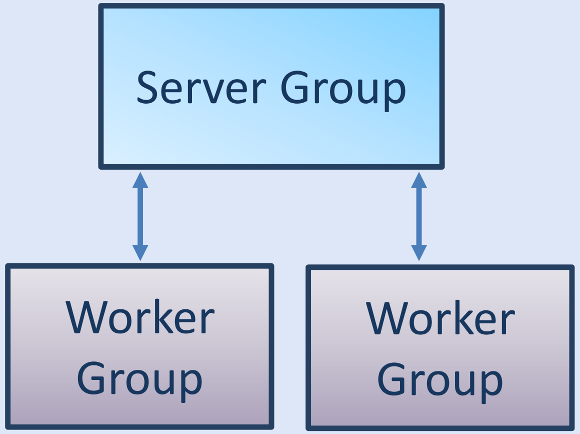
Harp



Various Collective Communication Operations

- Explicit Data Distribution
- Explicit Communication

Parameter Server



Asynchronous Communication Operations

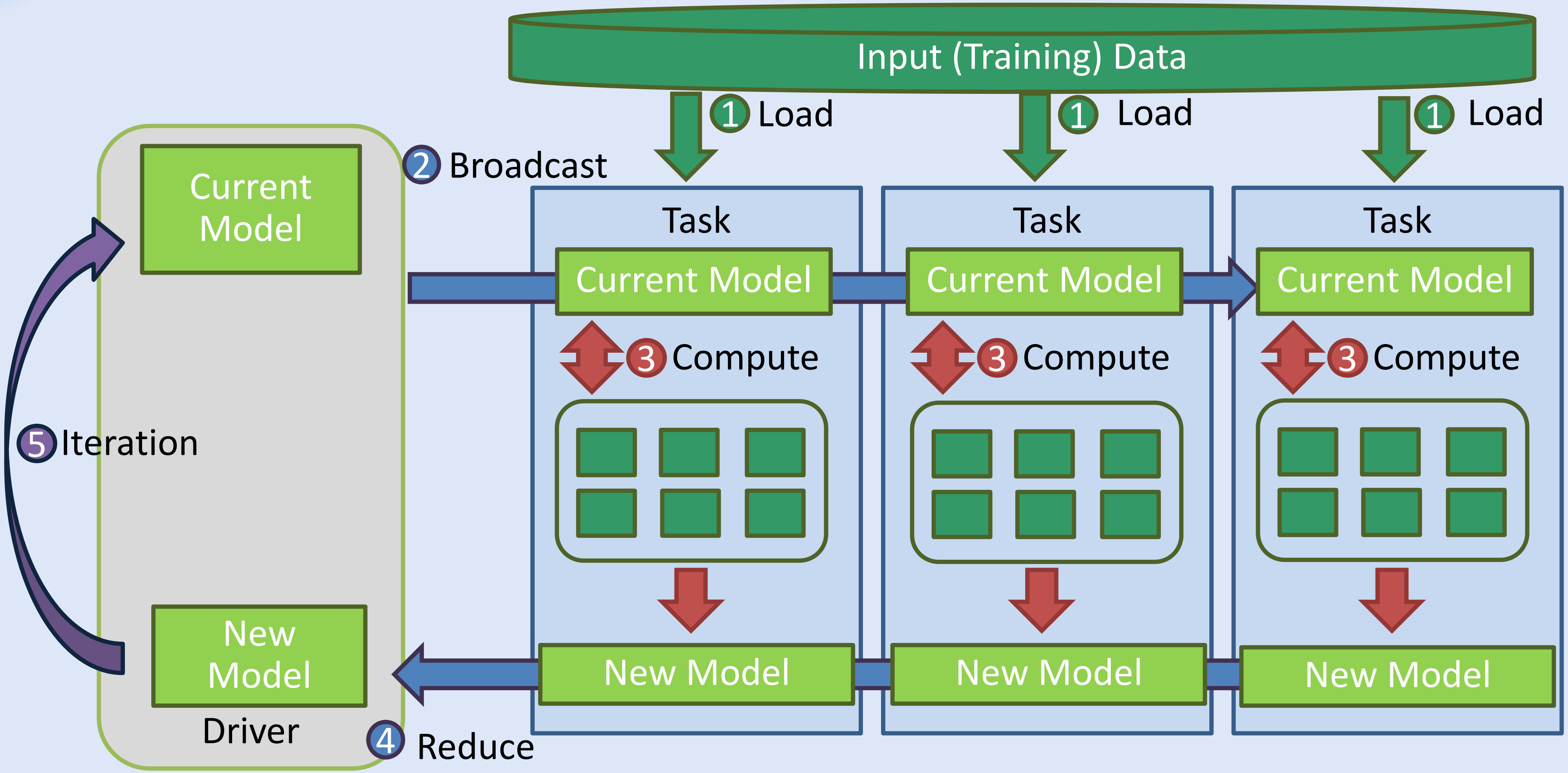
- Explicit Data Distribution
- Implicit Communication

M. Zaharia et al. “Spark: Cluster Computing with Working Sets”. HotCloud, 2010.

B. Zhang, Y. Ruan, J. Qiu. “Harp: Collective Communication on Hadoop”. IC2E, 2015.

M. Li, D. Anderson et al. “Scaling Distributed Machine Learning with the Parameter Server”. OSDI, 2014.

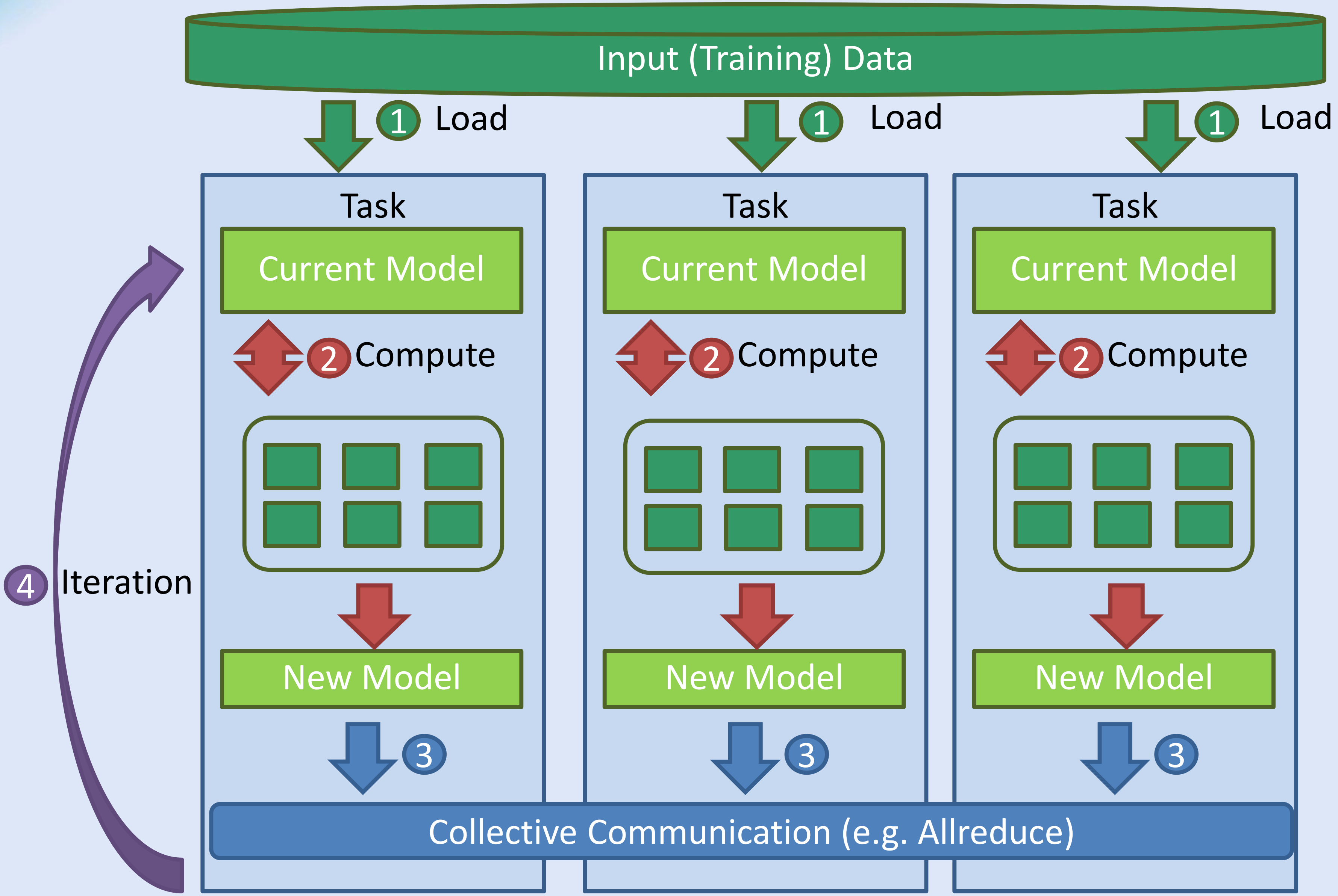
Spark



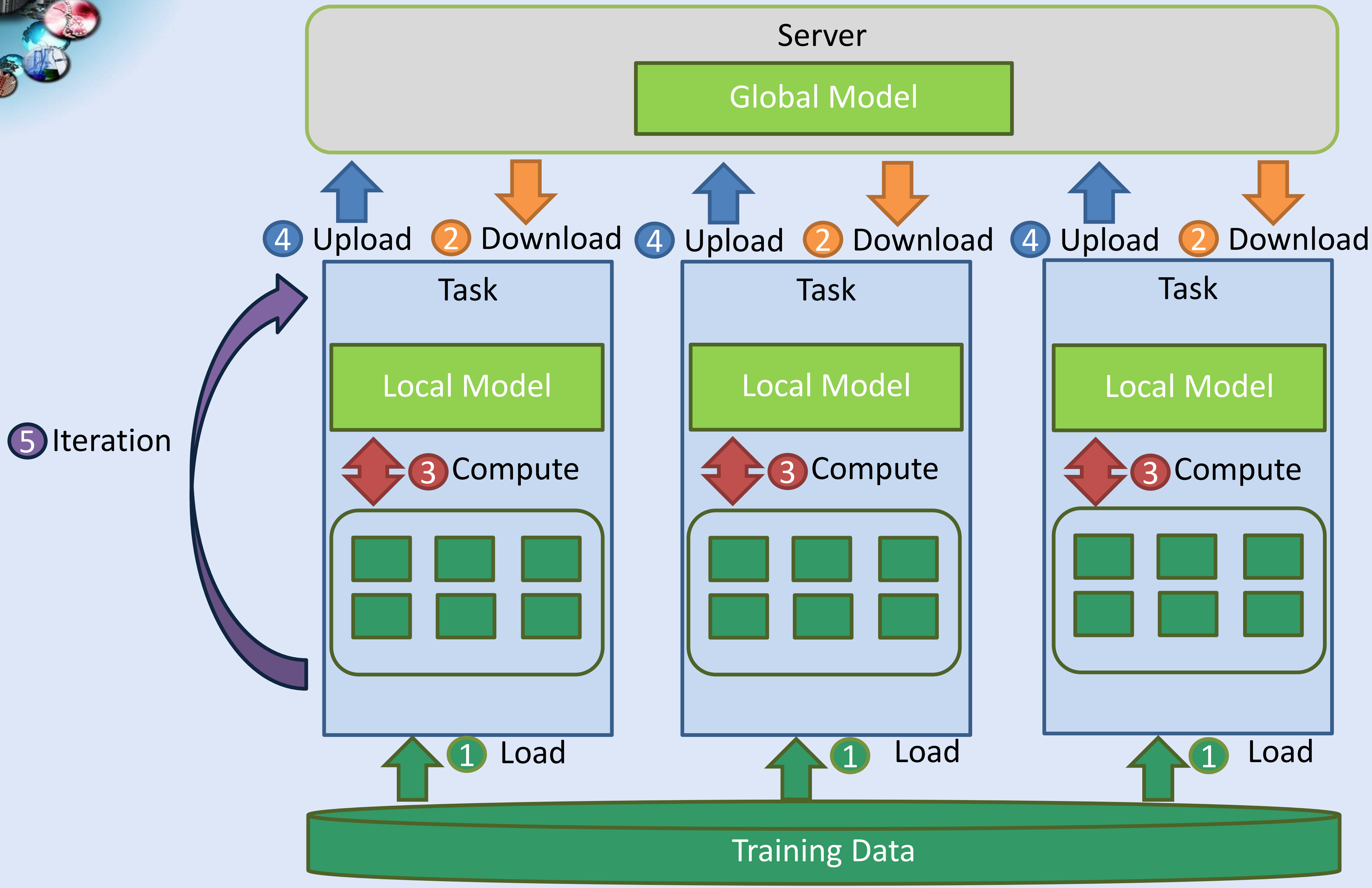
Input (Training) Data
1 Load
1 Load
1 Load
2 Broadcast
Current Model
Task
Current Model
3 Compute
New Model
Task
Current Model
3 Compute
New Model
Task
Current Model
3 Compute
New Model
5 Iteration
New Model
Driver
4 Reduce

SALSA

Harp


Input (Training) Data
1 Load
1 Load
1 Load
Task
Current Model
2 Compute
New Model
3
Task
Current Model
2 Compute
New Model
3
Task
Current Model
2 Compute
New Model
3
Collective Communication (e.g. Allreduce)
4 Iteration

Parameter Server



SALSA

Case Study 2:
Parallel Latent Dirichlet Allocation for Text Mining

Map Collective Computing Paradigm

LDA: mining topics in text collection

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

Figure 8: An example article from the AP corpus. Each color codes a different factor from which the word is putatively generated.

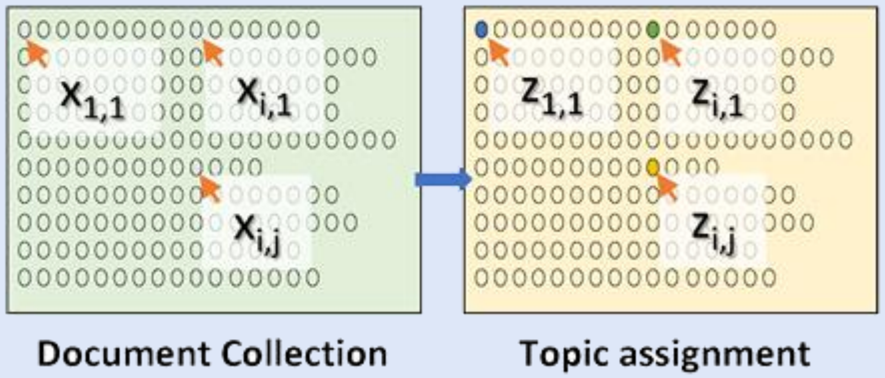
- Huge volume of Text Data
  - information overloading
  - what on earth is inside the TEXT Data?
- Search
  - find the documents relevant to my need (ad hoc query)
- Filtering
  - fixed info needs and dynamic text data
- What's new inside?
  - discover something I don't know

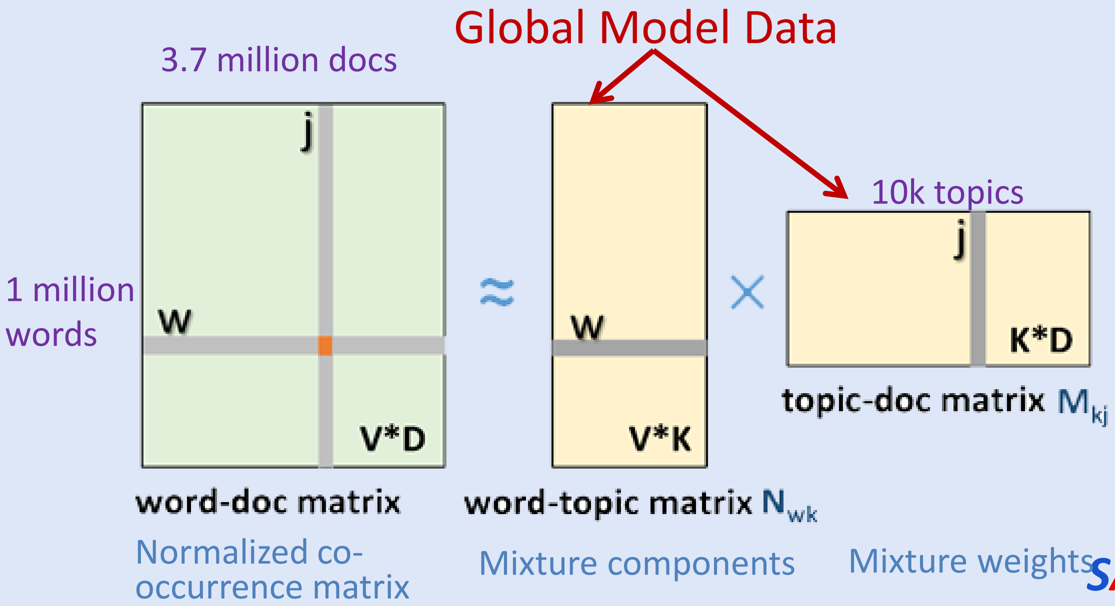
Blei, D. M., Ng, A. Y. & Jordan, M. I. Latent Dirichlet Allocation. J. Mach. Learn. Res. 3, 993–1022 (2003).

LDA and Topic Models

- Topic Models is a modeling technique, modeling the data by probabilistic generative process.
- Latent Dirichlet Allocation (LDA) is one widely used topic model.
- Inference algorithm for LDA is an iterative algorithm using share global model data.

- Document
- Word
- Topic: semantic unit inside the data
- Topic Model
  - documents are mixtures of topics, where a topic is a probability distribution over words





SALSA

Gibbs Sampling in LDA

- Observed data: W_{ij}, word on position i in doc j
- Try to estimate the latent variables (Model Data)
  - Z_{ij}, topic assignment accordingly to W_{ij}
  - N_{wk}, count matrix for word-topic distribution
  - N_{kj}, count matrix for topic-document distribution
- With parameters
  - Concentration Parameters - α, β, control model sparseness
  - D documents, V vocabulary size, K topics

Initialize:
  sample topic index $z_{ij} = k \sim Mult(1/K)$
Repeat until converge:
  for all documents $j \in [1, D]$ **do**
    for all words position $i \in [1, N_m]$ in document j **do**
    // for the current assignment k to a token t of word w_{ij}, decrease counts
    $n_{kj} -= 1; n_{tk} -= 1;$
    // multinomial sampling
    sample new topic index
    $k' \sim p(z_{ij} | z^{\neg ij}, w) \propto \frac{N_{wk}^{\neg ij} + \beta}{\sum_w N_{wk}^{\neg ij} + V\beta} \; N_{kj}^{\neg ij} + \alpha$
    // for the new assignment k' to the token t of word w_{ij}, increase counts
    $n_{k'j} += 1; n_{tk'} += 1;$

Training Datasets used in LDA Experiments

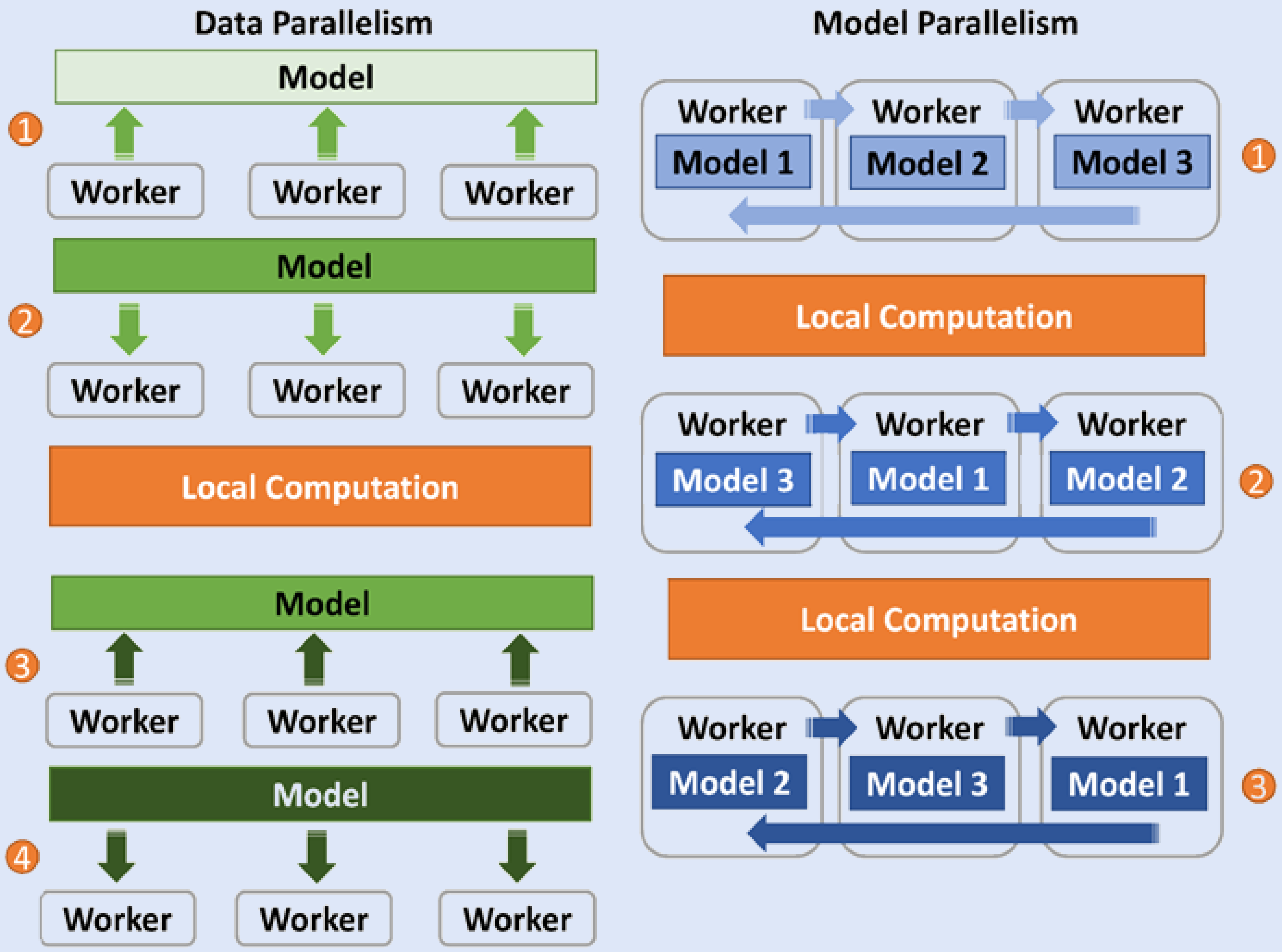
The total number of model parameters is kept as 10 billion on all the datasets.

Dataset	enwiki	clueweb	bi-gram	gutenberg
Num. of Docs	3.8M	50.5M	3.9M	26.2K
Num. of Tokens	1.1B	12.4B	1.7B	836.8M
Vocabulary	1M	1M	20M	1M
Doc Len. Avg/STD	293/523	224/352	434/776	31879/42147
Highest Word Freq.	1714722	3989024	459631	1815049
Lowest Word Freq.	7	285	6	2
Num. of Topics	10K	10K	500	10K
Init. Model Size	2.0GB	14.7GB	5.9GB	1.7GB

Note: Both “enwiki” and “bi-gram” are English articles from Wikipedia [31]. “clueweb is a 10% dataset from ClueWeb09, which is a collection of English web pages [32]. “gutenberg” is comprised of English books from Project Gutenberg [33].

Data Parallelism & Model Parallelism

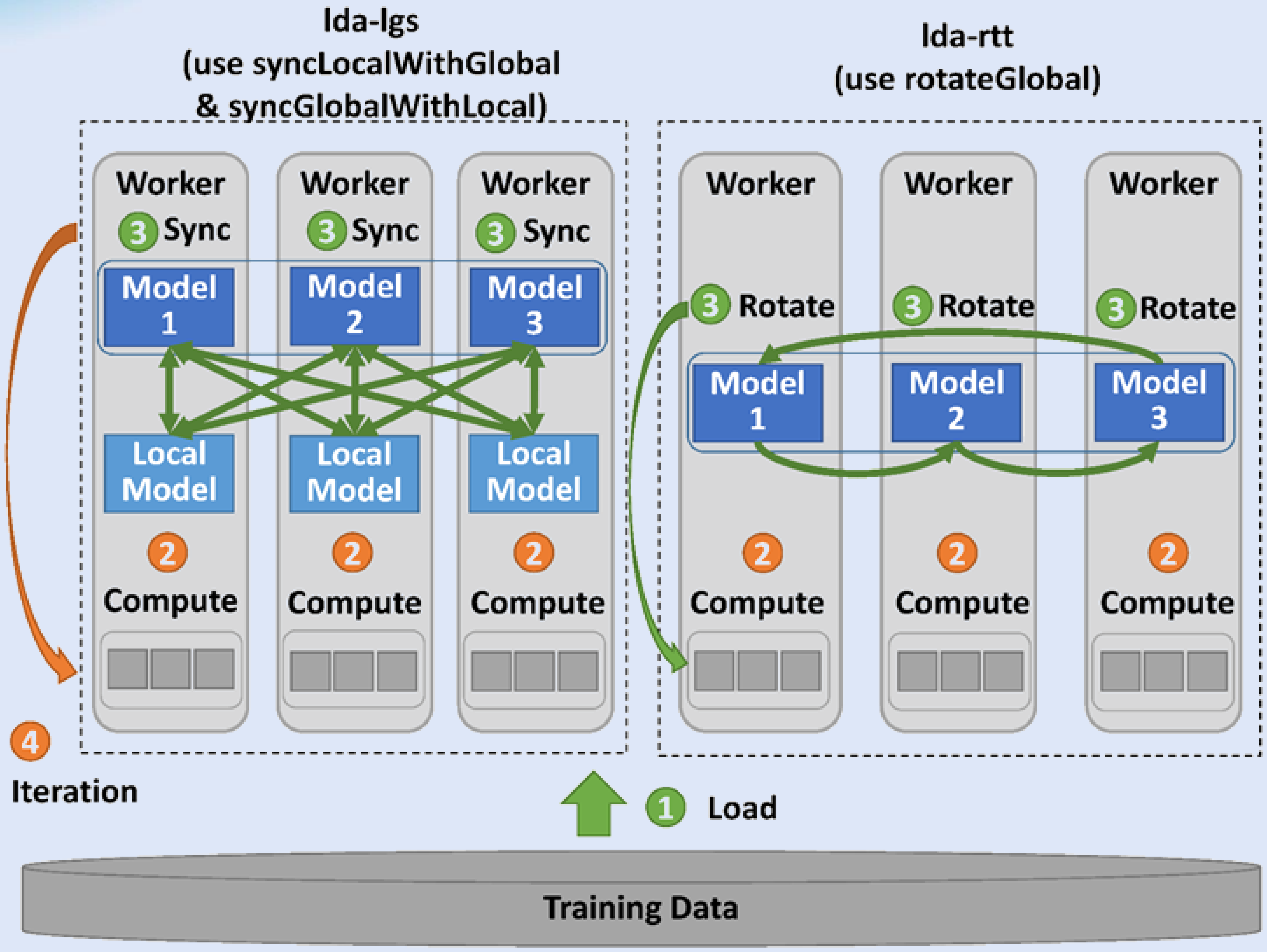
Data Parallelism
While the training data are split among parallel workers, the global model is distributed on a set of servers or existing workers. Each worker computes on a local model and updates it with the synchronization between local models and the global model.



Model Parallelism
In addition to splitting the training data over parallel workers, the global model data is split between workers and rotated between workers

Bingjing Zhang, Bo Peng and Judy Qiu, High Performance LDA through Collective Model Communication Optimization, Proceedings of International Conference on Computational Science (ICCS), June 6-8, 2016.

Harp-LDA Execution Flow



Challenges

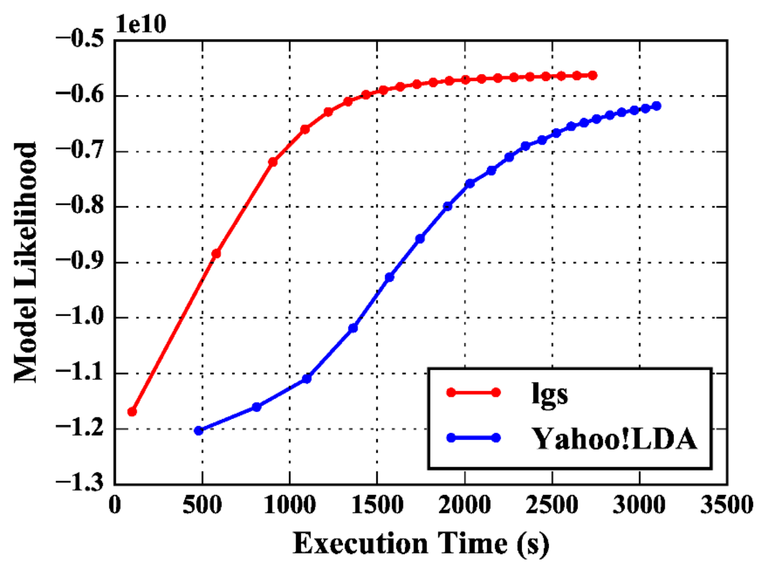
- High memory consumption for model and input data
- High number of iterations (~1000)
- Computation intensive
- Traditional "allreduce" operation in MPI-LDA is not scalable.

- Harp-LDA uses AD-LDA (Approximate Distributed LDA) algorithm (based on Gibbs sampling algorithm)
- Harp-LDA runs LDA in iterations of local computation and collective communication to generate new global model.

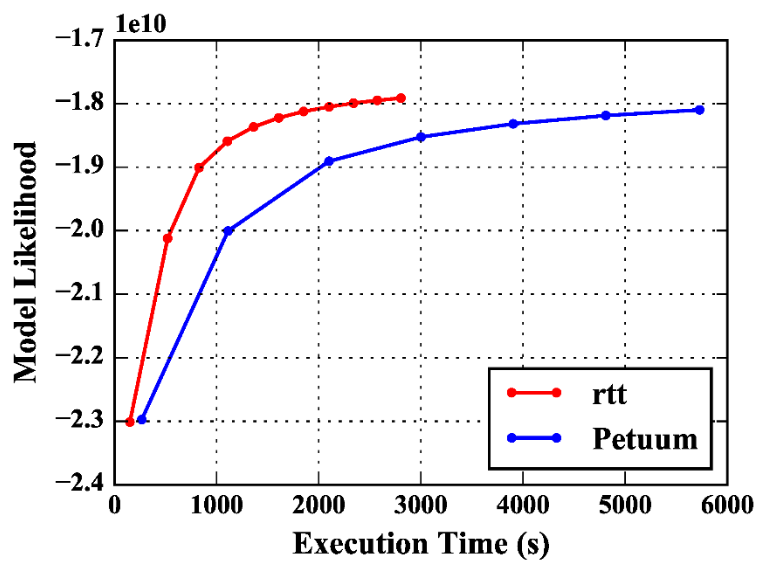
SALSA

Harp-LDA Performance Tests on Intel Haswell Cluster

Data Parallelism
Performance Comparison



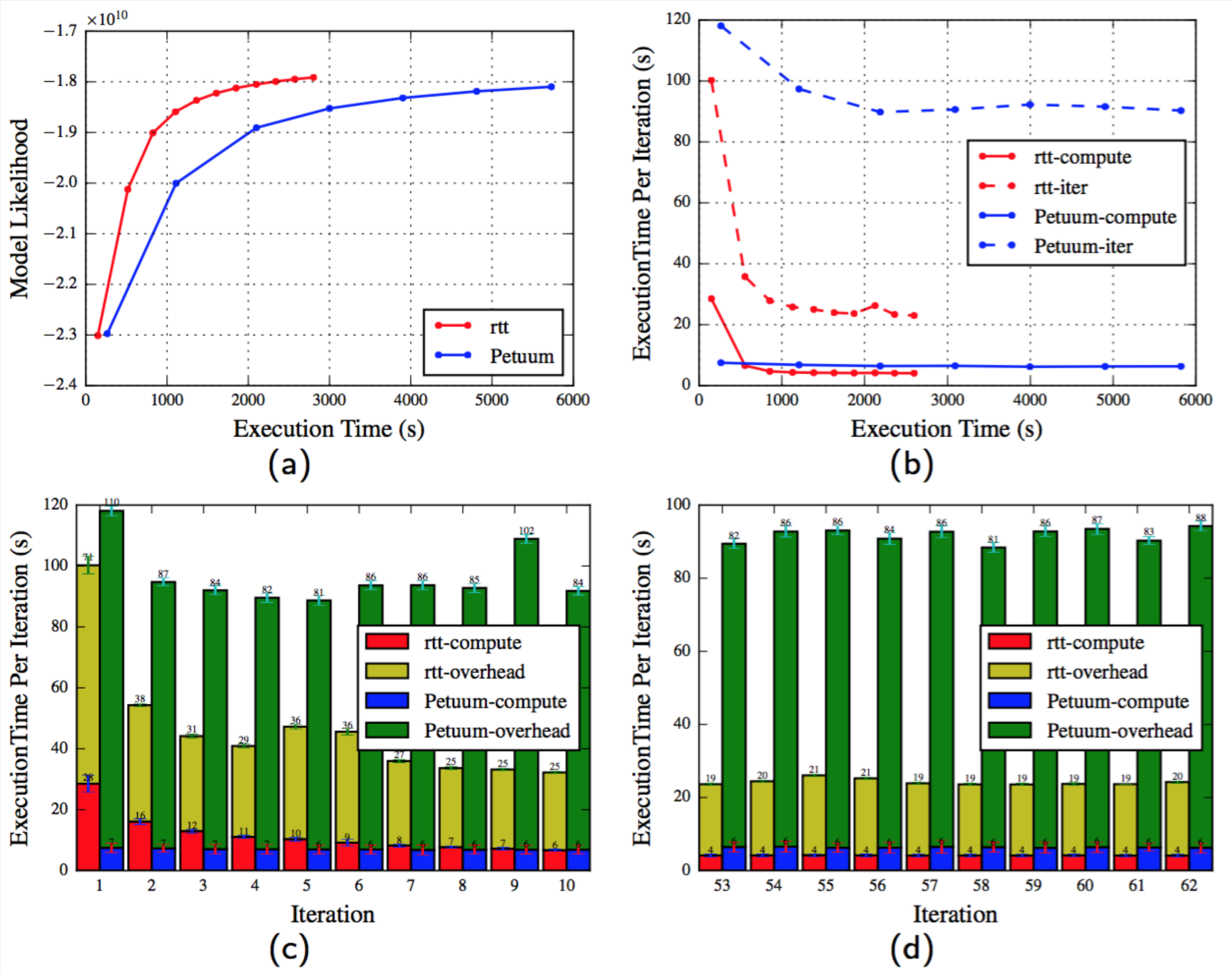
Model Parallelism
Performance Comparison



“enwiki” dataset. 3.8 million Wikipedia documents, Vocabulary: 1 million words; Topics: 10k topics; alpha: 0.01; beta: 0.01; iteration: 200

“bi-gram” dataset. 3.9 Wikipedia documents, Vocabulary: 20 million words; Topics: 500 topics; alpha: 0.01; beta: 0.01; iteration: 200

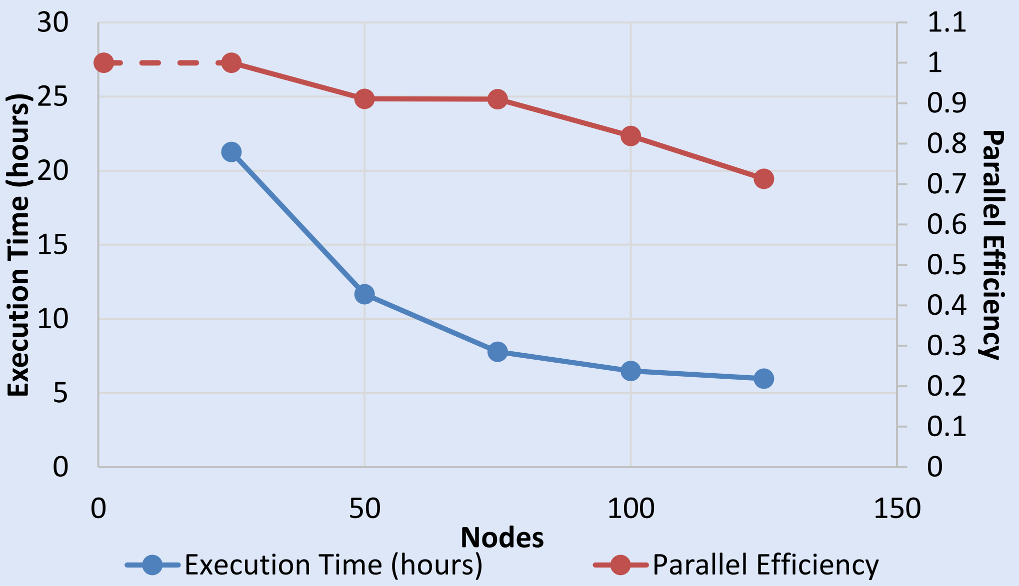
Harp-LDA Model Parallelism on "bi-gram"



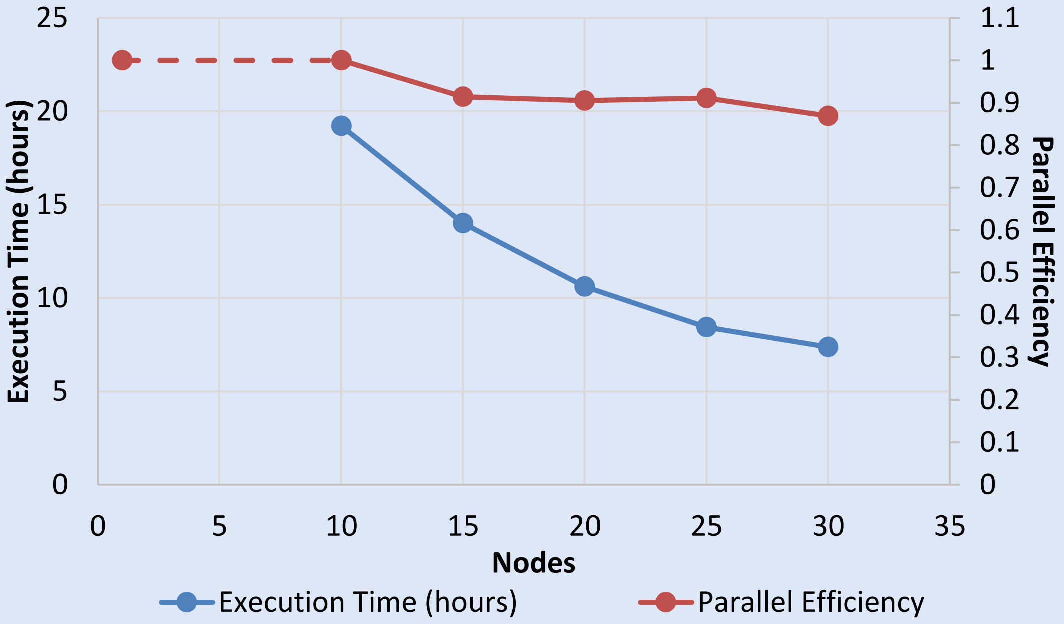
(a) Elapsed Execution Time vs. Model Likelihood (b) Elapsed Execution Time vs. Iteration Execution Time
(c) First 10 Iteration Execution Times (d) Final 10 Iteration Execution Times

Harp LDA Scaling Tests

Harp LDA on Big Red II Supercomputer (Cray)



Harp LDA on Juliet (Intel Haswell)



Corpus: 3,775,554 Wikipedia documents, Vocabulary: 1 million words; Topics: 10k topics; alpha: 0.01; beta: 0.01; iteration: 200

Machine settings

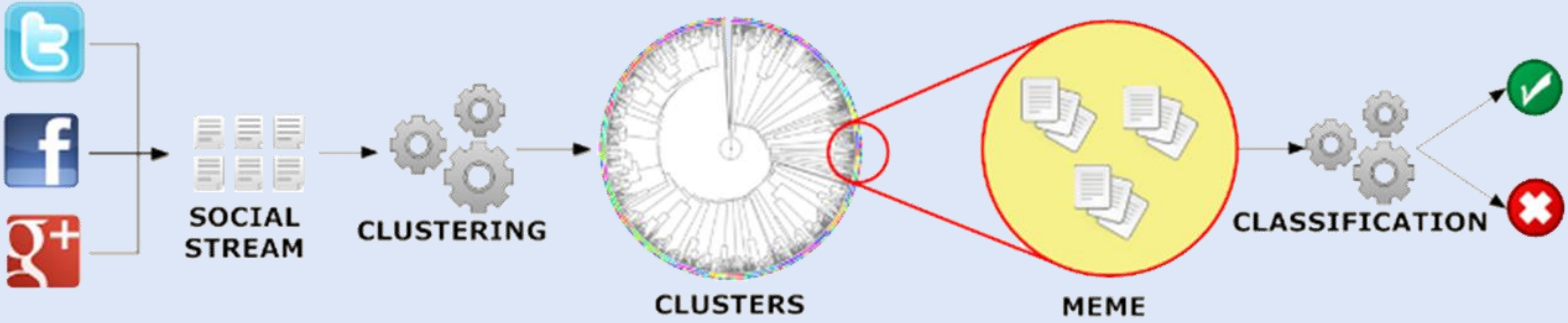
- Big Red II: tested on 25, 50, 75, 100 and 125 nodes, each node uses 32 parallel threads; Gemini interconnect
- Juliet: tested on 10, 15, 20, 25, 30 nodes, each node uses 64 parallel threads on 36 core Intel Haswell node (each with 2 chips); infiniband interconnect

Case Study 3:
Parallel Tweet Online Clustering

Map Streaming Computing Paradigm

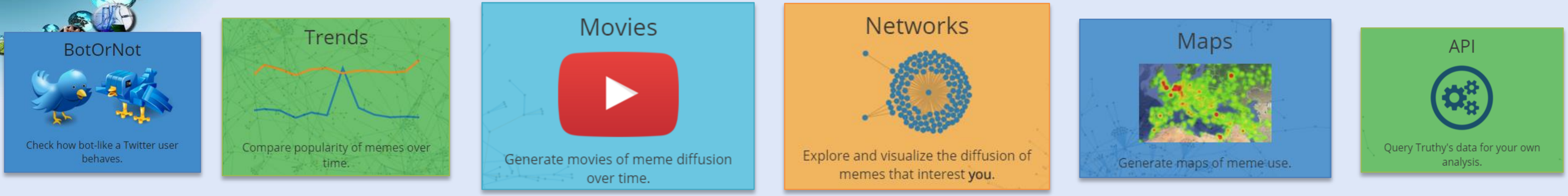
Parallel Tweet Online Clustering with Apache Storm

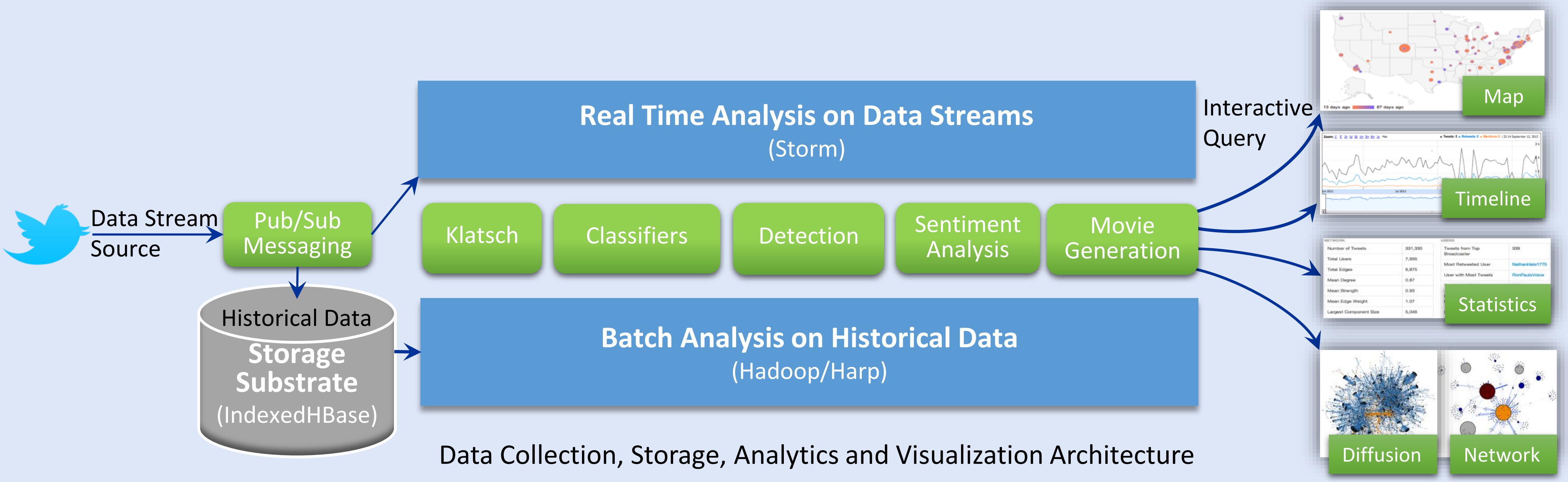
- **IUNI analysis pipeline for meme clustering and classification :** Detecting Early Signatures of Persuasion in Information Cascades
- Implement with HBase + Hadoop (Batch) and HBase + Storm(Streaming) + ActiveMQ
- 2 million streaming tweets processed in 40 minutes; 35,000 clusters
- Storm Bolts coordinated by ActiveMQ to synchronize parallel cluster center updates – add loops/iterations to Apache Storm



Xiaoming Gao, Emilio Ferrara, Judy Qiu, Parallel Clustering of High-Dimensional Social Media Data Streams Proceedings of CCGrid, May 4-7, 2015

Social Media Observatory





Data Collection, Storage, Analytics and Visualization Architecture

- Starting from late 2010, we have collected an ongoing, near uninterrupted sample of 10% public Twitter streaming record (approximate 100 billion tweets to date). The existing collection has 180 TB of historical data and loading rate of 40 million tweets per day.
- IndexedHBase can automatically retrieve data from the 10% Twitter stream (“gardenhose”), split obtained Tweets into partitions, and parse and index such data on a daily base. With multiple parallel partition loaders, one day’s worth of data can be loaded within a few hours.
- We have shown in our recent work to be able to process the Twitter 10% data stream in real-time with 96-way parallelism.

Sequential Algorithm for Clustering Tweet Stream

- Online (streaming) K-Means clustering algorithm with *sliding time window* and *outlier detection*
- Group tweets in a time window as **protomemes**:
  - Label protomemes (points in space to be clustered) by "markers", which are *Hashtags*, *User mentions*, *URLs*, and *phrases*
  - A phrase is defined as the textual content of a tweet that remains after removing the hashtags, mentions, URLs, and after stopping and stemming
    - Number of tweets in a *protomem*e : Min: 1, Max :206, Average 1.33
- Note a given tweet can be in more than one protomeme
  - One tweet on average appears in 2.37 protomemes
  - And number of protomemes is 1.8 times number of tweets

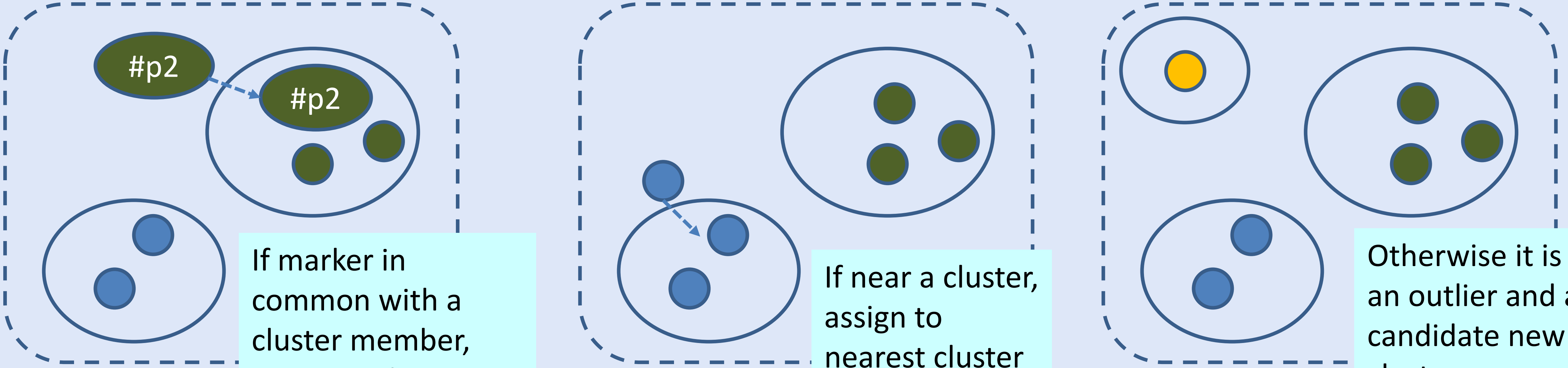
Online K-Means clustering

(1) Slide time window by one time step

(2) Delete old protomemes out of time window from their clusters

(3) Generate protomemes for tweets in this step

(4) For each new protomeme classify in old or new cluster (outlier)



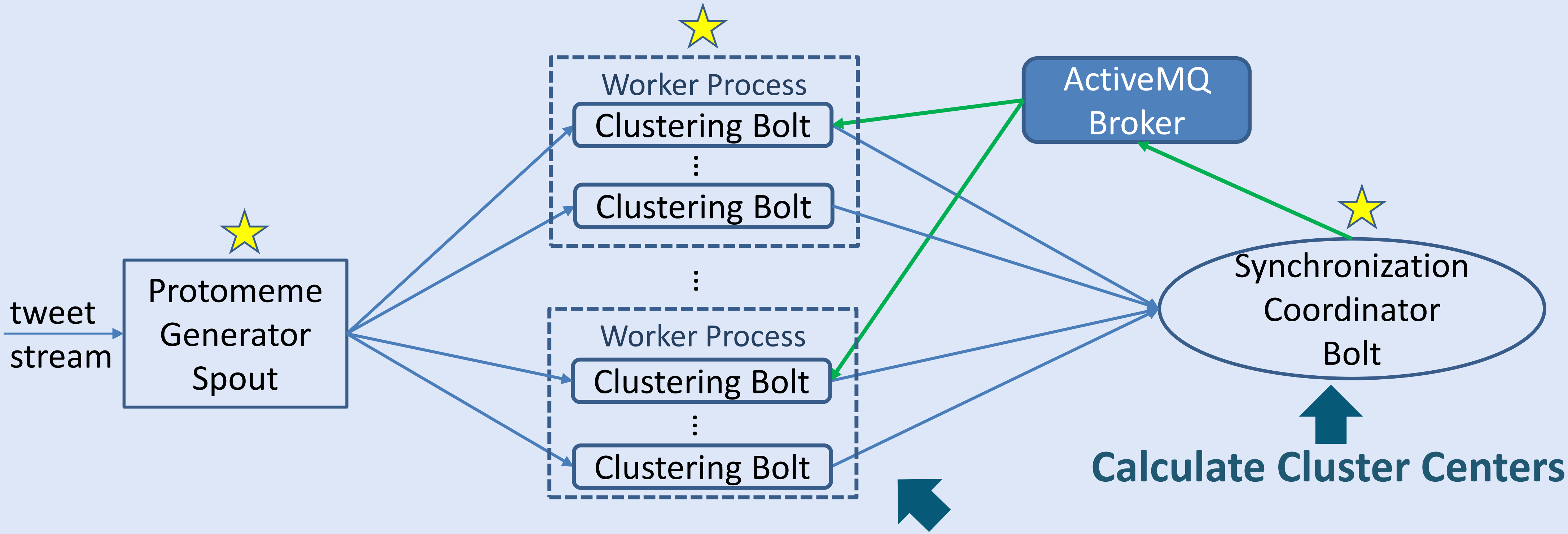
If marker in common with a cluster member, assign to that cluster

If near a cluster, assign to nearest cluster

Otherwise it is an outlier and a candidate new cluster

Parallelization with Storm – Challenges

DAG organization of parallel workers: hard to synchronize cluster information



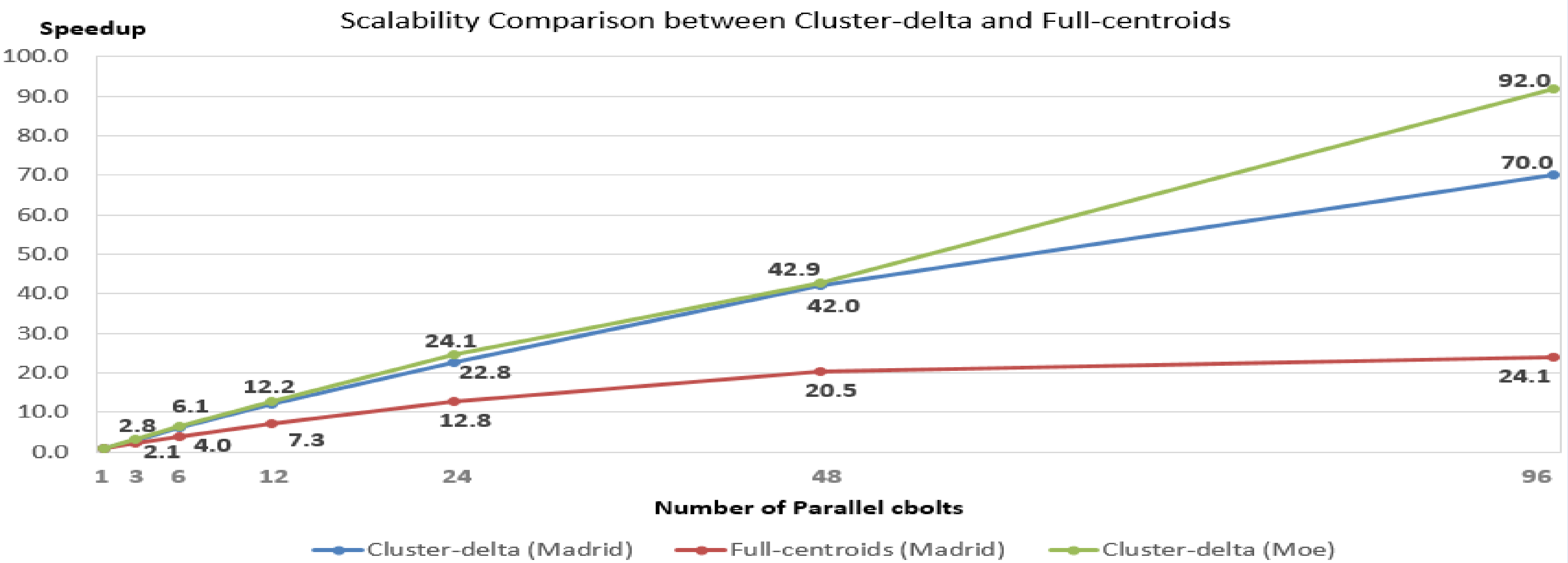
☆ Synchronization initiation methods:

- Spout initiation by broadcasting INIT message
- Clustering bolt initiation by local counting
- Sync coordinator initiation by global counting (of #protomemes)

Suffer from variation of processing speed

SALSA

Parallel Tweet Clustering with Storm



- Speedup on up to 96 bolts on two clusters, Moe and Madrid
- Red curve is old online Kmeans algorithm; green and blue new algorithm
- Full Twitter – 1000 way parallelism (expected)

SALSA

Outline

Cloud Computing

PlotViz

Twister
Cross Platform Iterative MapReduce

Twister4Azure
Iterative MapReduce for Azure Cloud

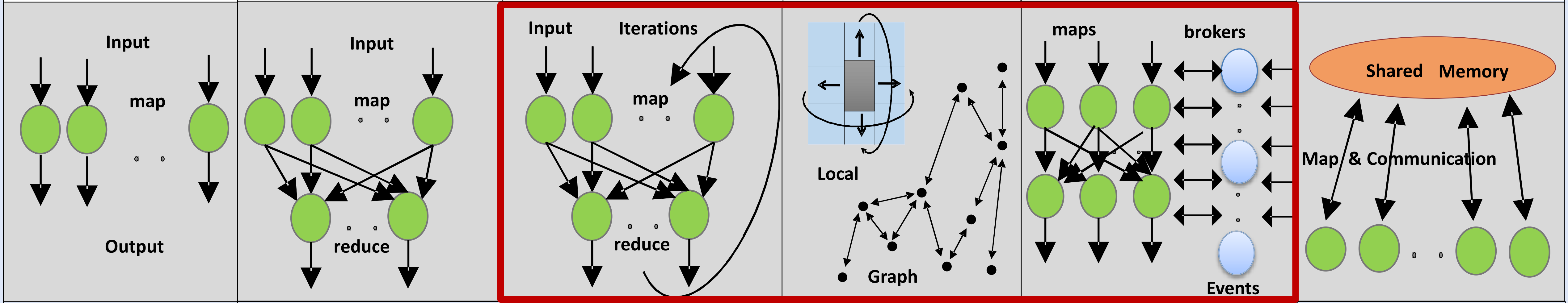
Harp

IndexedHBase

1. Introduction: Big Data (Batch and Streaming), interdisciplinary, HPC and Clouds
2. Clouds are important for Big Data Analysis
3. Clouds are important for Education and Training
4. Interdisciplinary Applications and Technologies
5. Enhancing Commodity systems (Apache Big Data Stack) to HPC-ABDS
6. Summary and Future

Six Computation Paradigms for Data Analytics

(1) Map Only Pleasingly Parallel	(2) Classic Map-Reduce	(3) Iterative Map Reduce or Map-Collective	(4) Point to Point or Map-Communication	(5) Map-Streaming	(6) Shared memory Map-Communication
- BLAST Analysis - Local Machine Learning - Pleasingly Parallel	- High Energy Physics (HEP) Histograms, - Web search - Recommender Engines	- Expectation Maximization - Clustering - Linear Algebra - PageRank	- Classic MPI - PDE Solvers and Particle Dynamics - Graph	- Streaming images from Synchrotron sources, Telescopes, Internet of Things	- Difficult to parallelize - asynchronous parallel Graph



These 3 Paradigms are my Focus

SALSA

Comparison of current Data Analytics stack from Cloud and HPC infrastructure

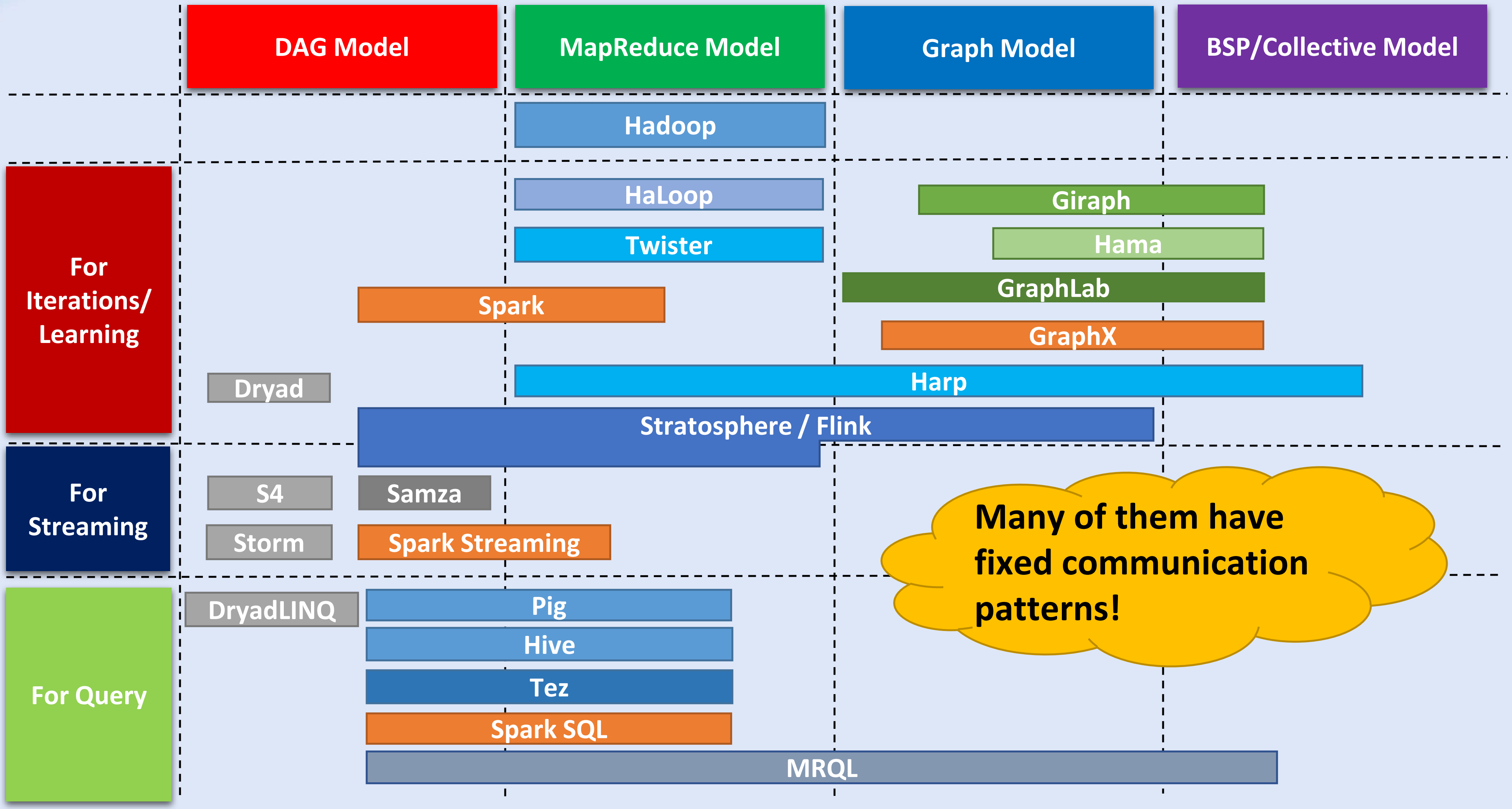
	Big Data ABDS	HPC, Cluster
Orchestration	Crunch, Tez, Cloud Dataflow	Kepler, Pegasus
Libraries	MLlib/Mahout, R, Python	Matlab, Eclipse, Apps
High-Level Programming	Pig, Hive, Drill	Domain-specific Languages
Platform as a Service	App Engine, BlueMix, Elastic Beanstalk	XSEDE Software Stack
Languages	Java, Erlang, SQL, SparQL	Fortran, C/C++
Streaming Parallel Runtime	Storm, Kafka, Kinesis	
Coordination	MapReduce	MPI/OpnMP/OpenCL
Caching	Memcached	
Data Management	Hbase, Neo4J, MySQL	iRODS
Data Transfer	Sqoop	GridFTP
Scheduling	Yarn	Slurm
File Systems	HDFS, Object Stores	Lustre
Formats	Thrift, Protobuf	FITS, HDF
Virualization	Openstack	Docker, SR-IOV
Infrastructure	CLOUDS	SUPERCOMPUTERS

HPC-ABDS Integrated Software

My Focus

J. Qiu, S. Jha, A. Luckow, G. Fox, TowardsHPC-ABDS: An Initial High-Performance Big Data Stack, proceedings of ACM 1st Big Data Interoperability Framework Workshop: Building Robust Big Data ecosystem, NIST special publication, March 13-21, 2014.

The Models of Contemporary Big Data Tools



Computation Characteristics of Big Data Tools

Tool	Computation Model	Data Abstraction	Communication Pattern
MPI [1]	Loosely Synchronous	N/A	Arrays and objects sending/receiving or collective communication operations
Hadoop [2]	(Iterative) MapReduce	Key-Values	Shuffle (disk-based) between Map stage and Reduce stage
Twister [3]			Regroups (in-memory) between Map stage and Reduce stage, “broadcast” and “aggregate”
Spark [4]		RDD	RDD Transformations on RDD, “broadcast and “aggregate”
Dryad [5]	DAG	N/A	Communication is between two connected vertex processes in the execution of DAG
Giraph [6]	Graph/BSP	Graph	Graph-based message communication following Pregel model
Hama [7]			Graph-based message communication following Pregel model or direct message communication between workers
GraphLab (Dato) [8, 9, 10]			Graph-based communication through caching and fetching of ghost vertices and edges or the communication between master vertex and its replicas in PowerGraph (GAS) model
GraphX [11]			Graph-based communication supports Pregel model and PowerGraph model

Outline

1. Introduction: Big Data (Batch and Streaming), interdisciplinary, HPC and Clouds
2. Clouds are important for Big Data Analysis
3. Clouds are important for education: CloudMOOC
4. Interdisciplinary Applications and Technologies
5. Enhancing Commodity systems (Apache Big Data Stack) to HPC-ABDS
6. Summary and Future

Cloud Computing
PlotViz
Twister
Cross Platform Iterative MapReduce
Twister4Azure
Iterative MapReduce for Azure Cloud
Harp
IndexedHBase

Progress in HPC-ABDS Runtime

- **Standalone Twister**: Iterative Execution (caching) and High performance communication extended to first Map-Collective runtime
- **HPC-ABDS Plugin Harp:** adds HPC communication performance and rich data abstractions to Hadoop
- **Online Clustering** with Storm integrates parallel and dataflow computing models
- Development of library of **Collectives** to use at Reduce phase
  - Broadcast and Gather needed by current applications
  - Discover other important ones (e.g. Allgather, Global-local sync, rotation)
  - Implement efficiently on each platform (e.g. Amazon, Azure, Big Red II, Haswell Clusters)
- Clearer application **fault tolerance** model based on implicit synchronizations points at iteration end points
- Runtime for **data parallel languages** with initial work on Apache Pig enhanced with Harp
- Integrate GPU support with Map-Collective model including deep learning

SALSA

Summary and Future

- Identification of **Apache Big Data Software Stack** and integration with **High Performance Computing Stack** to give **HPC-ABDS**
  - ABDS/Many Big Data applications/algorithms need HPC for performance
  - HPC needs ABDS for rich software model productivity/sustainability
- Identification of Six **Computation Models** for HPC and Data Analytics
- Identification and Study of **Map-Collective** and **Map-Streaming** Model
- Integrate streaming and batch workflow as in social observatory – look at Apache Beam and Google Cloud Dataflow
- Implement National Strategic Computing Initiative **HPC-Big Data Convergence** with HPC-ABDS
- Continue **Twister/ Twister4Azure** to **Harp** conversion with more data analytics
  - Apache *Pig*, *Hadoop, Storm,* and *HBase* enhancement in the form of plug-in
- Start HPC incubator project in Apache to bring HPC-ABDS to community

Acknowledgements

Bingjing Zhang　Xiaoming Gao　Stephen Wu

Jaliya Ekanayake　Thilina Gunarathne　Yuan Yang

Prof. Haixu Tang
Bioinformatics

Prof. David Wild
Cheminformatics

Prof. Raquel Hill
Security

Prof. David Crandall
Computer Vision

Prof. Filippo Menczer & CNETS
Complex Networks and Systems

SALSA HPC Group
School of Informatics and Computing
Indiana University

The Harp Library

- Harp is an implementation designed in a pluggable way to bring high performance to the Apache Big Data Stack and bridge the differences between Hadoop ecosystem and HPC system through a clear communication abstraction, which did not exist before in the Hadoop ecosystem.
- Hadoop Plugin that targets Hadoop 2.2.0
- Provides implementation of the collective communication abstractions and MapCollective programming model
- **Project Link:** http://salsaproj.indiana.edu/harp/index.html
- **Source Code Link:** https://github.com/jessezbj/harp-project

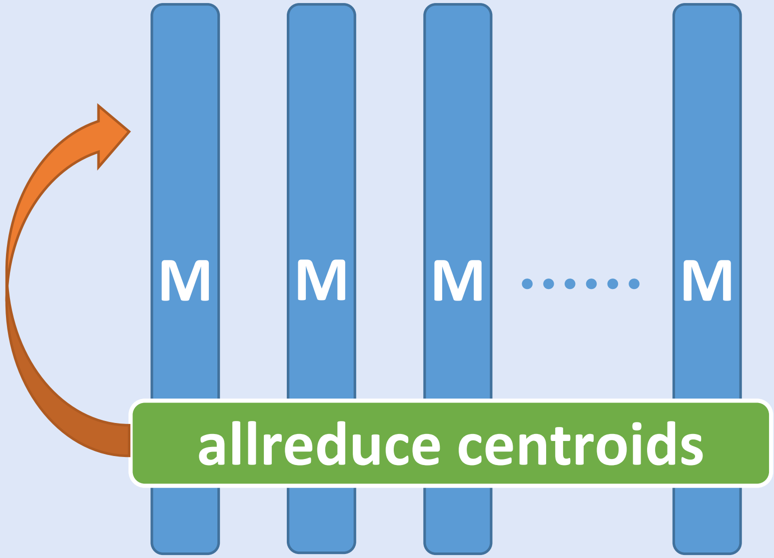
We built Map-Collective as a unified model to improve the performance and expressiveness of Big Data tools. We ran Harp on K-means, Graph Layout, and Multidimensional Scaling algorithms with realistic application datasets over 4096 cores on the IU BigRed II Supercomputer (Cray/Gemini) where we have achieved linear speedup.

Collective Communication Operations

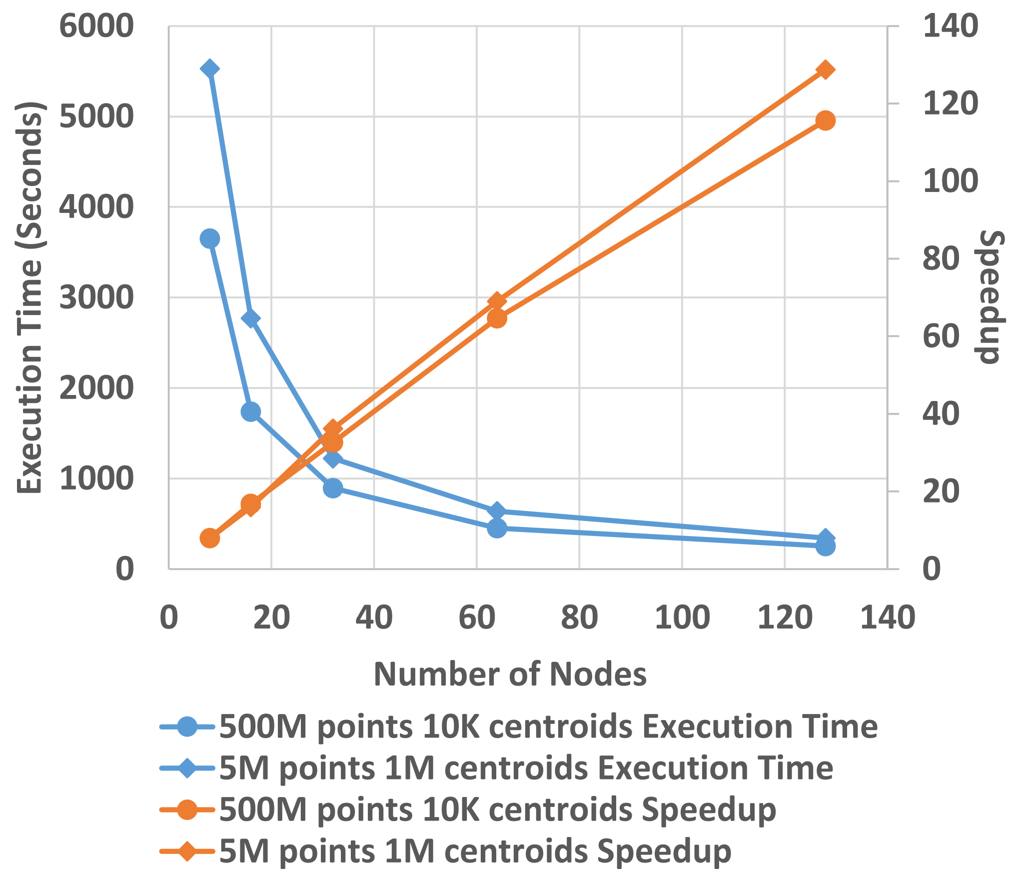
Operation Name	Data Abstraction	Algorithm	Time Complexity
broadcast	arrays, key-value pairs & vertices	chain	$n\beta$
allgather	arrays, key-value pairs & vertices	bucket	$pn\beta$
allreduce	arrays, key-value pairs	bi-directional exchange	$(log_2 p)n\beta$
		regroup-allgather	$2n\beta$
regroup	arrays, key-value pairs & vertices	point-to-point direct sending	$n\beta$
send messages to vertices	messages, vertices	point-to-point direct sending	$n\beta$
send edges to vertices	edges, vertices	point-to-point direct sending	$n\beta$

SALSA

K-means Clustering




```
On each node do
  for t < iteration-num; t←t+1 do
    for each p in points do
      for each c in centroids do
        Calculate the distance between p and c;
      Add point p to the closest centroid c;
    Allreduce the local point sum;
    Compute the new centroids;
```

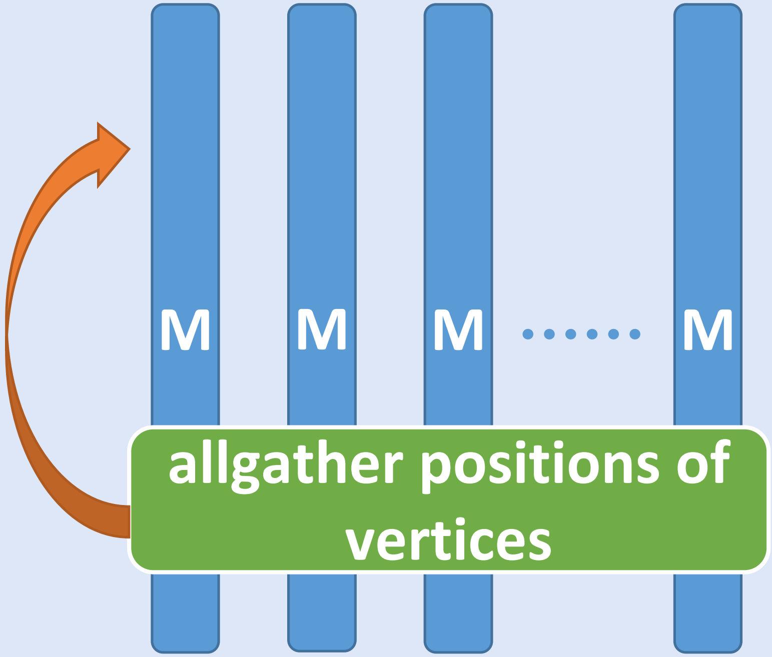


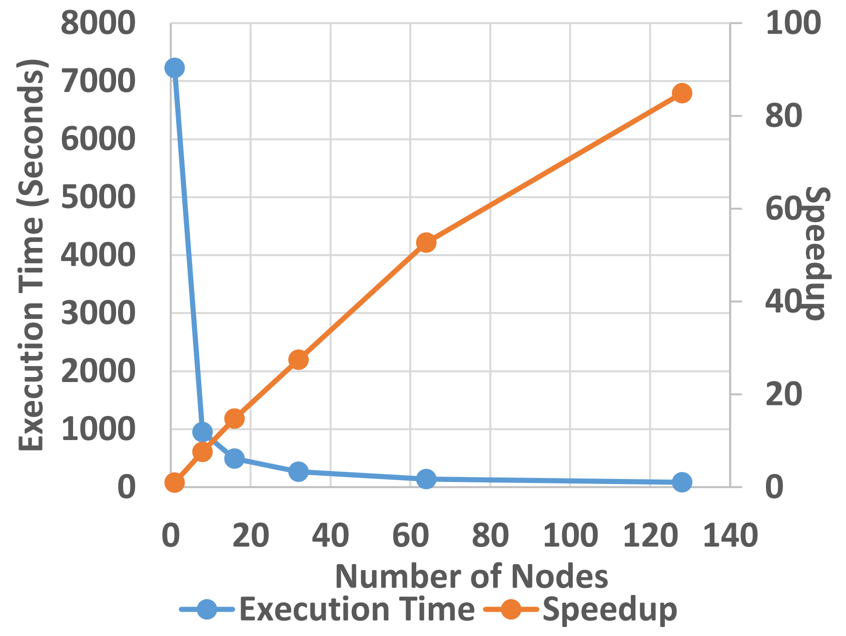
Test Environment: Big Red II

http://kb.iu.edu/data/bcqt.html

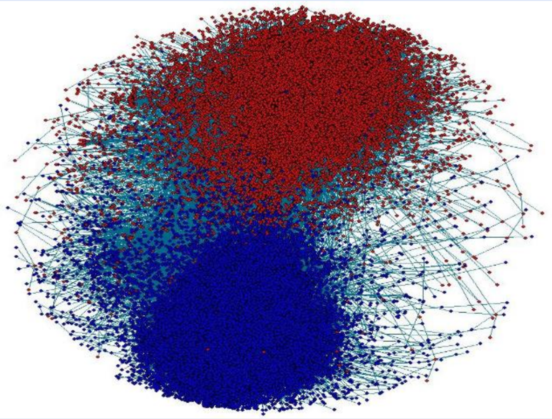
SALSA

Force-directed Graph Drawing Algorithm





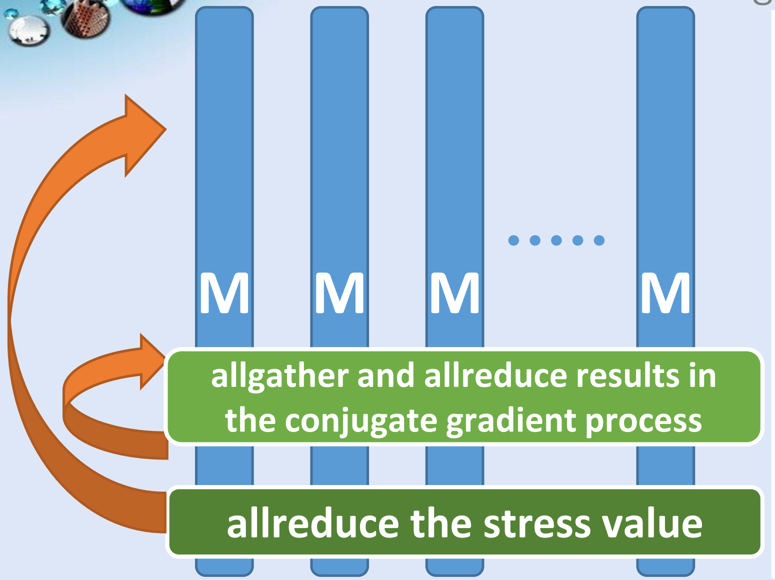

```
On each node do
for t < iteration-num; t←t+1 do
    Calculate repulsive forces and displacements;
    Calculate attractive forces and displacements;
    Move the points with displacements limited by
    temperature;
    Allgather the new coordination values of the
    points;
```



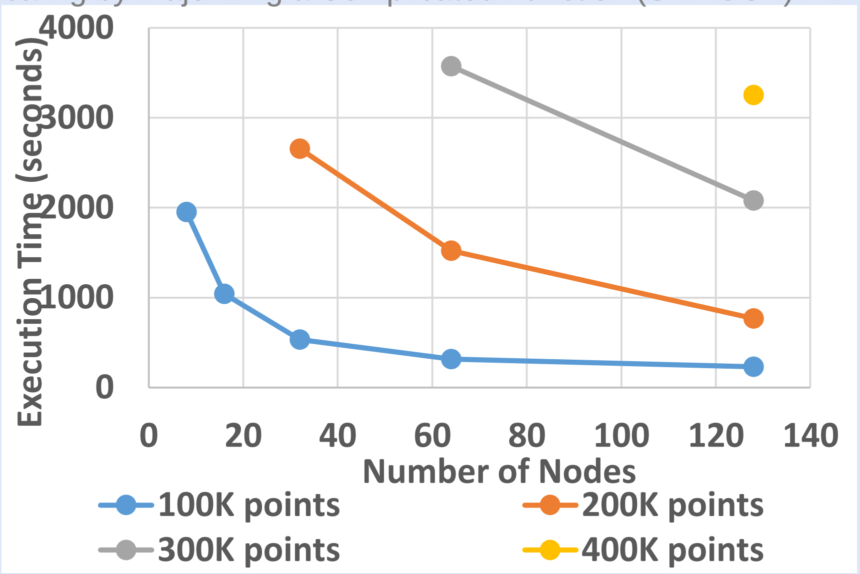
- Near linear scalability Per-iteration on sequential R for 2012 network: 6035 seconds

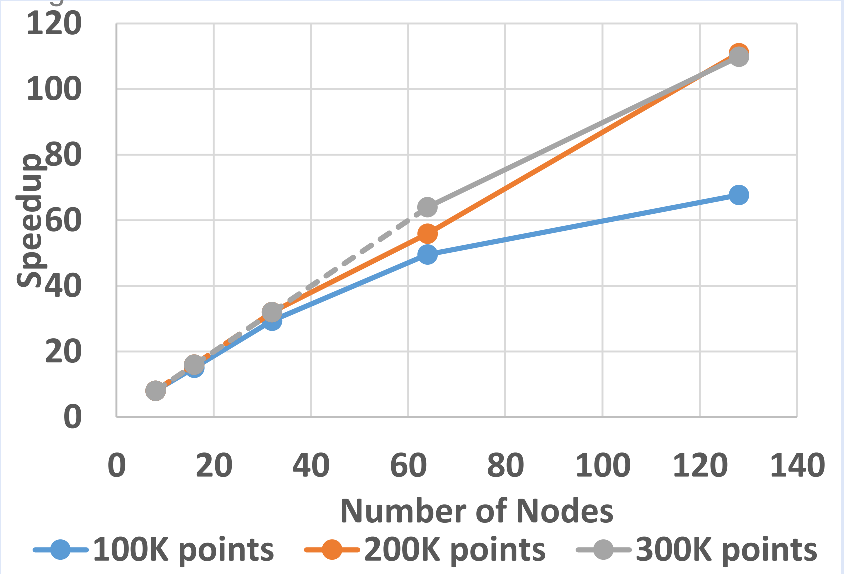
T. Fruchterman, M. Reingold. “Graph Drawing by Force-Directed Placement”, Software Practice & Experience 21 (11), 1991.

WDA-SMACOF

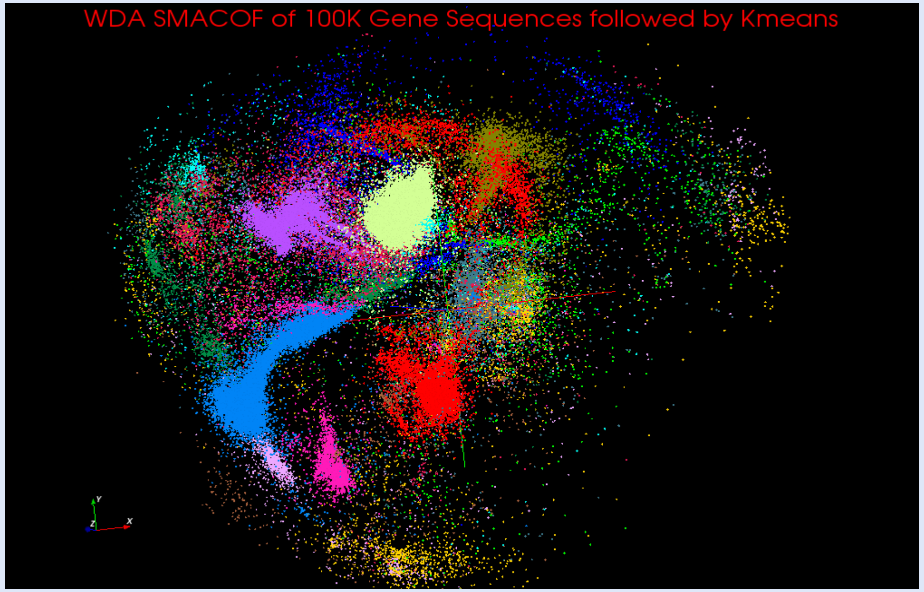


Scaling by Majorizing a Complicated Function (SMACOF) MDS algorithm






```
On each node do
  while current-temperature > min-temperature do
    while stress-difference > threshold do
      Calculate BC matrix;
      Use conjugate gradient process to solve the
      new coordination values;
      (this is an iterative process which contains
       allgather and allreduce operations)
      Compute and allreduce the new stress value;
      Calculate the difference of the stress
      values;
    Adjust the current temperature;
```



Y. Ruan et al. “A Robust and Scalable Solution for Interpolative Multidimensional Scaling With Weighting”. E-Science, 2013.

SALSA