# Introduction to
# Big Data in HPC,
## Hadoop and HDFS – Part Two


Dr. – Ing. Morris Riedel

*Adjunct Associated Professor, University of Iceland*

*Jülich Supercomputing Centre, Germany*

*Head of Research Group High Productivity Data Processing*


***Cy-Tera/LinkSCEEM HPC Administrator Workshop, Nicosia, The Cyprus Institute, 19th January – 21th January 2015***

HELMHOLTZ ASSOCIATION

**Research Field Key Technologies**

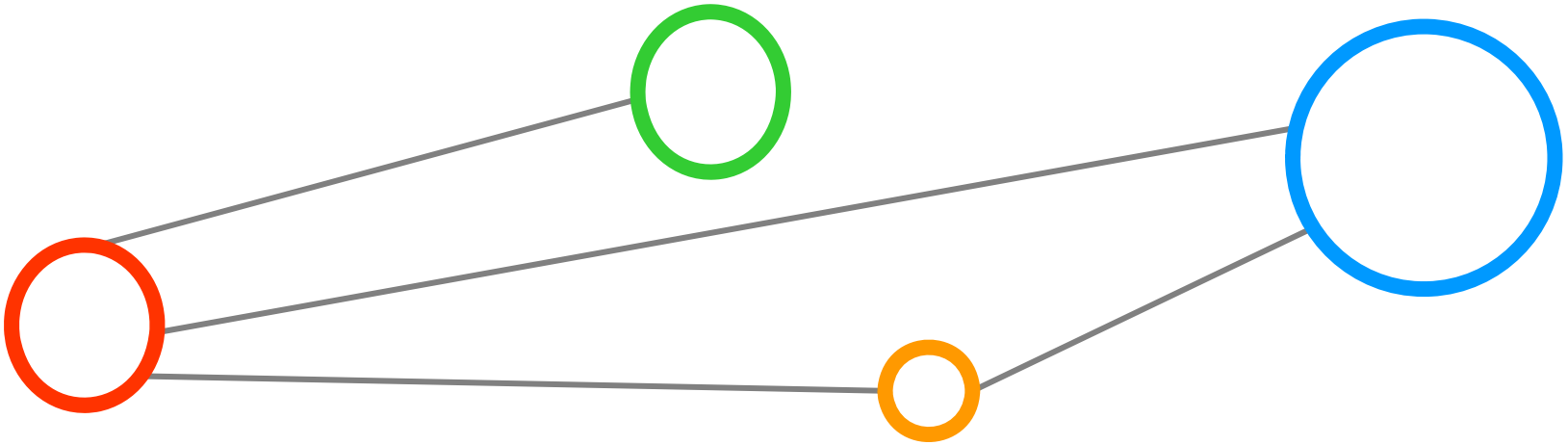
**Jülich Supercomputing Centre**

**Supercomputing & Big Data**

JÜLICH FORSCHUNGSZENTRUM

UNIVERSITY OF ICELAND
SCHOOL OF ENGINEERING AND NATURAL SCIENCES
FACULTY OF INDUSTRIAL ENGINEERING, MECHANICAL ENGINEERING AND COMPUTER SCIENCE

# Overall Course Outline



# Overall Course Outline

## Part One ‘Big Data‘ Challenges & HPC Tools

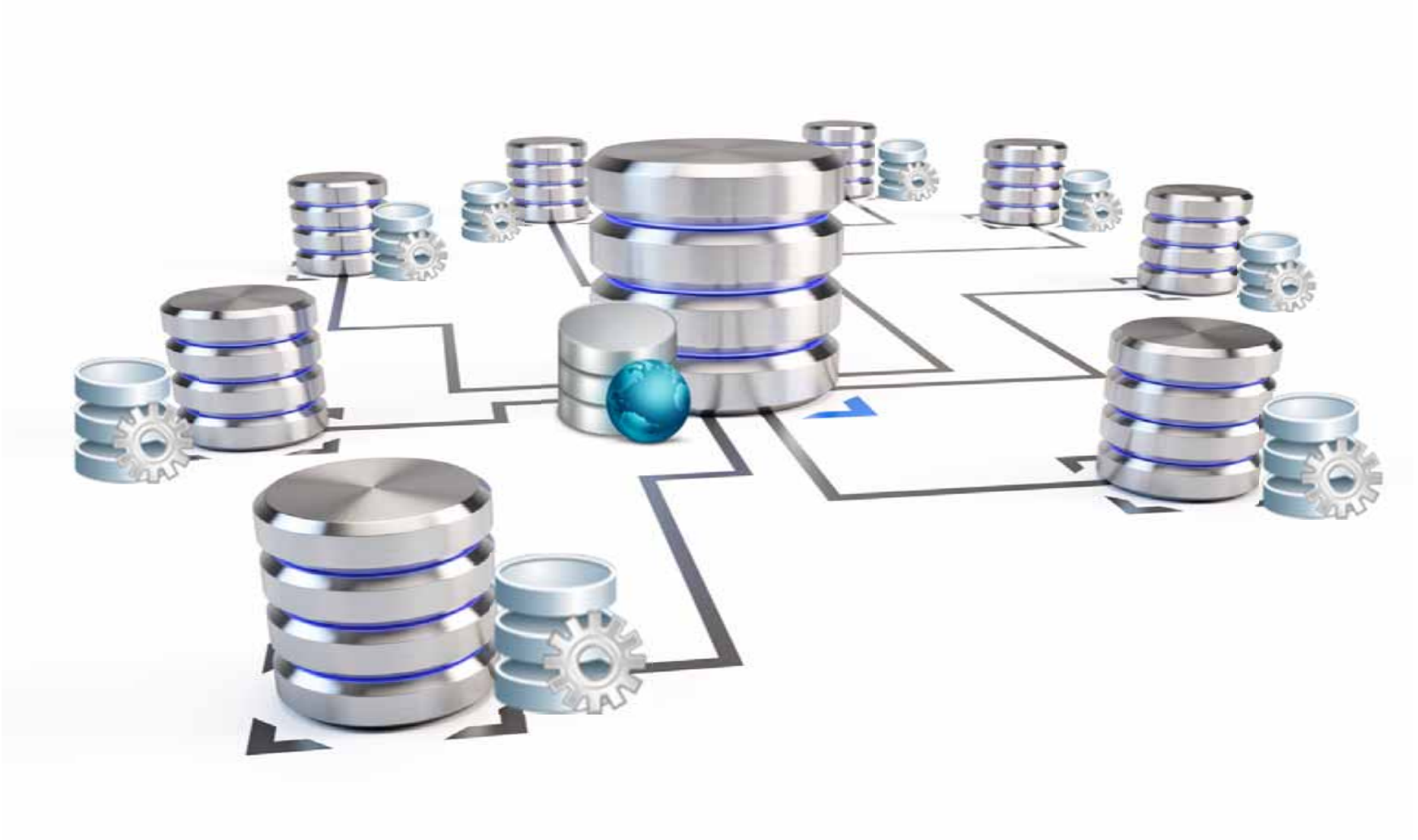
- Understanding ‘Big Data‘ in Science & Engineering
- Statistical Data Mining and Learning from ‘Big Data‘
- OpenMP/MPI Tool Example for Clustering ‘Big Data‘
- MPI Tool Example for Classification of ‘Big Data‘

*coffee break*

## Part Two ‘Big Data‘ & Distributed Computing Tools

- Exploring Parallel & Distributed Computing Approaches
- Examples of Map-Reduce & ‘Big Data‘ Processing with Hadoop
- Tools for handling ‘Big Data‘ storage & replication methods
- Technologied for Large-scale distributed ‘Big Data‘ Management

# Part Two



# Part Two – Outline

## Part One ‘Big Data‘ Challenges & HPC Tools

- Understanding ‘Big Data‘ in Science & Engineering
- Statistical Data Mining and Learning from ‘Big Data‘
- OpenMP/MPI Tool Example for Clustering ‘Big Data‘
- MPI Tool Example for Classification of ‘Big Data‘

*coffee break*

## Part Two ‘Big Data‘ & Distributed Computing Tools

- Exploring Parallel & Distributed Computing Approaches
- Examples of Map-Reduce & ‘Big Data‘ Processing with Hadoop
- Tools for handling ‘Big Data‘ storage & replication methods
- Technologies for Large-scale distributed ‘Big Data‘ Management

# Emerging Big Data Analytics vs. Traditional Data Analysis

- **Data Analytics are powerful techniques to work on large data**
- **Data Analysis is the in-depth interpretation of research data**
- **Both are a complementary technique for understanding datasets**
- **Data analytics may point to interesting ‚events' for data analysis**

## Data Analysis supports the search for 'causality'

- Describing exactly WHY something is happening
- Understanding causality is hard and time-consuming
- Searching it often leads us down the wrong paths


## Big Data Analytics focussed on 'correlation'

- Not focussed on causality – enough THAT it is happening
- Discover novel patterns/events and WHAT is happening more quickly
- Using correlations for invaluable insights – often data speaks for itself

# Google Flu Analytics – Hype vs. (Scientific) Reality

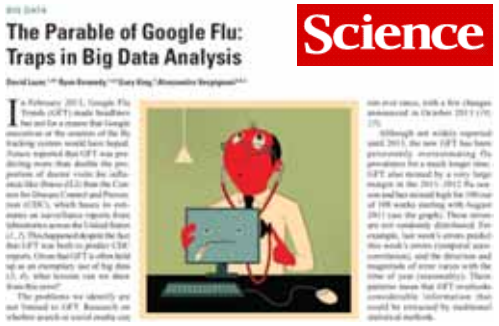
## 2009 – H1N1 Virus made headlines

- Nature paper from Google employees
- Explains how Google is able to predict flus
- Not only national scale, but down to regions even
- Possible via logged big data – 'search queries'


***[1] Jeremy Ginsburg et al., 'Detecting influenza epidemics using search engine query data', Nature 457, 2009***


## 2014 – The Parable of Google Flu

- Large errors in flu prediction were avoidable and offer lessons for the use of big data
- (1) Transparency and Replicability impossible
- (2) Study the Algorithm since they keep changing
- (3) It's Not Just About Size of the Data




***[1]* David Lazer, Ryan Kennedy, Gary King, and Alessandro Vespignani, 'The Parable of Google Flu: Traps in Big Data Analysis', Science Vol (343), 2014**

# Modern Data Mining Applications

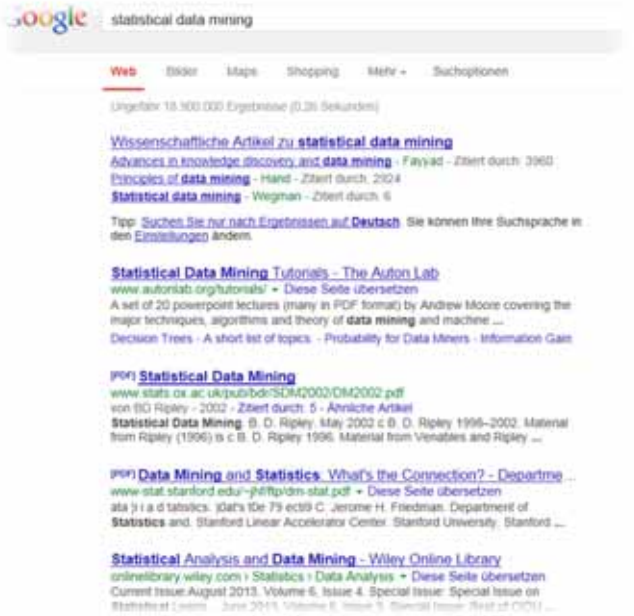
## Properties → opportunity to exploit 'parallelism'

- Require handling of immense amounts of data quickly
- Provide data that is extremely regular and can be independently processed

> - **Many modern data mining applications require computing on compute nodes (i.e. processors/cores) that operate independently from each other**
> - **Independent means there is little or even no communication between tasks**

## Examples from the Web

- Ranking of Web pages by importance
  (includes iterated matrix-vector multiplication)
- Searches in social networking sites
  (includes search in graph with
  hundreds of nodes/billions of edges)




## Major difference to typical HPC workload

- Not simulation of physical phenomena

# Hiding the complexity of computing & data

Many users of parallel data processing machines are not technical savvy users and don't need to know system details

- Scientific domain-scientists (e.g. biology) need to focus on their science & data
- Scientists from statistics/machine-learning need to focus on their algorithms & data

*[3] Science Progress*

Non-technical users raise the following requirements for a 'data processing machinery':

- **The 'data processing machinery' needs to be easy to program**
- **The machinery must hide the complexities of computing (e.g. different networks, various operating systems, etc.)**
- **It needs to take care of the complexities of parallelization (e.g. scheduling, task distribution, synchronization, etc.)**

# ‘Data Processing Machinery’ Available Today

## Specialized Parallel Computers (aka Supercomputers)

- Interconnent between nodes is expensive (i.e. Infiniband/Myrinet)
- Interconnect not always needed in data analysis (independence in datasets)
- Programming is relatively difficult/specific, parallel programming is profession of its own

## Large ‘compute clusters’ dominate data-intensive computing

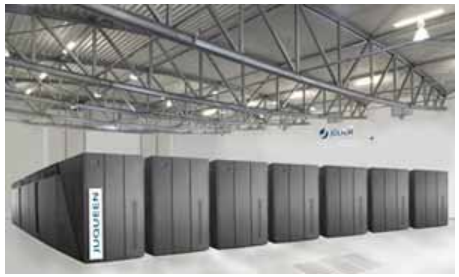
- Offer ‘cheaper’ parallelism (no expensive interconnect & switches between nodes)
- Compute nodes (i.e. processors/cores) interconnected with usual ethernet cables
- Provide large collections of commodity hardware (e.g. normal processors/cores)

# Possible Failures during Operation

- **Rule of thumb: The bigger the cluster is, the more frequent failures happen**

## Reasons for failures

- Loss of a single node within a rack
  (e.g. hard disk crash, memory errors)
- Loss of an entire rack (e.g. network issues)
- Operating software ‘errors/bugs’



## Consequences of failures

- Long-running compute tasks (e.g. hours/days) need to be restarted
- Already written data may become inconsistent and needs to be removed
- Access to unique datasets maybe hindered or even unavailable

# Two Key Requirements for Big Data Analytics

Taking into account the ever-increasing amounts of ‘big data’

- Think: ‘Big Data’ not always denoted by volume, there is velocity, variety, …

1. Fault-tolerant and scalable data analytics processing approach

- Data analytics computations must be divided in small task for easy restart
- Restart of one data analytics task has no affect on other active tasks
- E.g. Hadoop implementation of the map-reduce paradigm

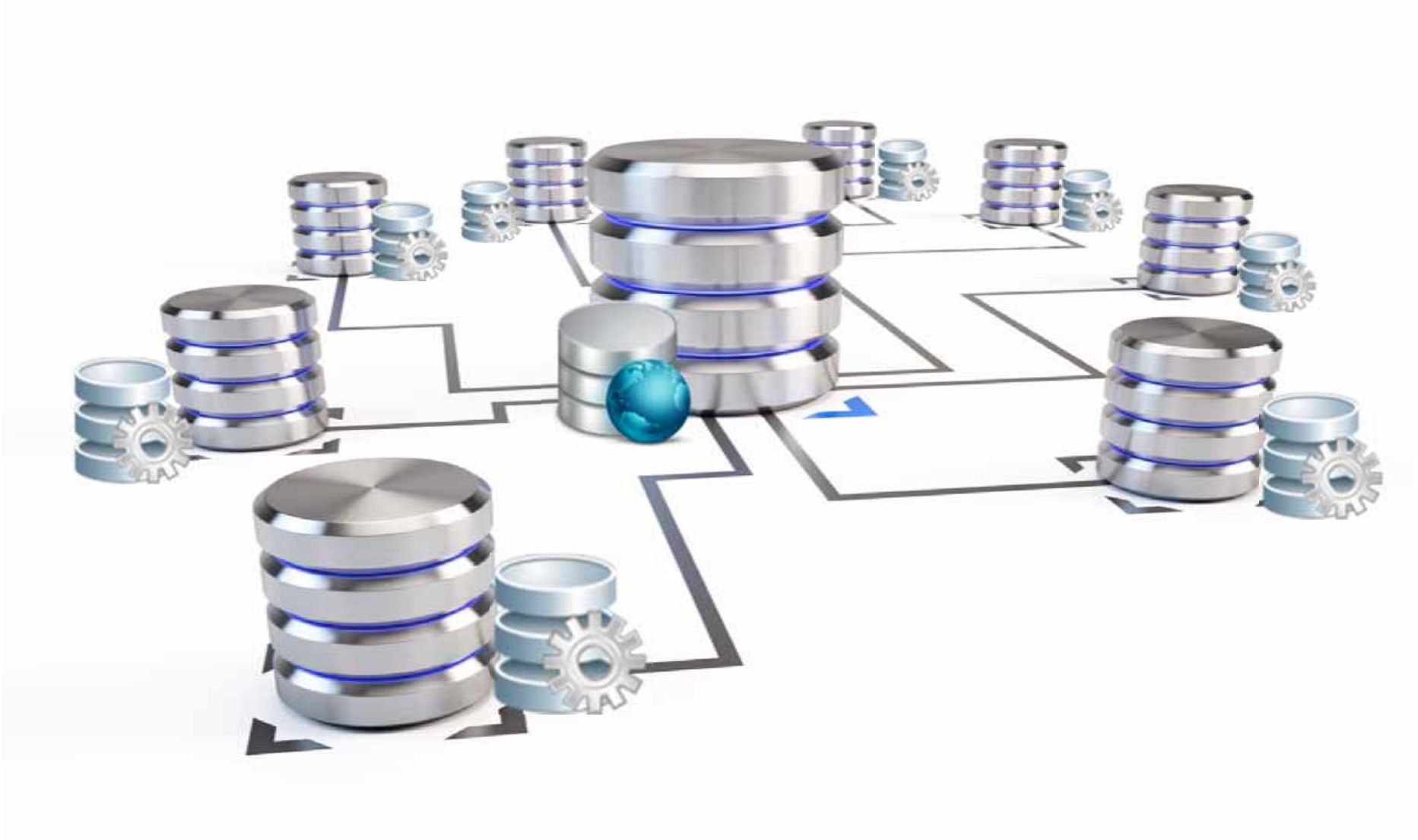
- **A data analytics processing programming model is required that is easy to use and simple to program with fault-tolerance already within its design**

2. Reliable scalable ‘big data’ storage method

- Data is (almost always) accessible even if failures in nodes/disks occur
- Enable the access of large quantities of data with good performance
- E.g. Hadoop Distributed File System (HDFS) implementation

- **A specialized distributed file system is required that assumes failures as default**

# Part Two – Questions



# Part Two – Outline

## Part One ‘Big Data‘ Challenges & HPC Tools

- Understanding ‘Big Data‘ in Science & Engineering
- Statistical Data Mining and Learning from ‘Big Data‘
- OpenMP/MPI Tool Example for Clustering ‘Big Data‘
- MPI Tool Example for Classification of ‘Big Data‘

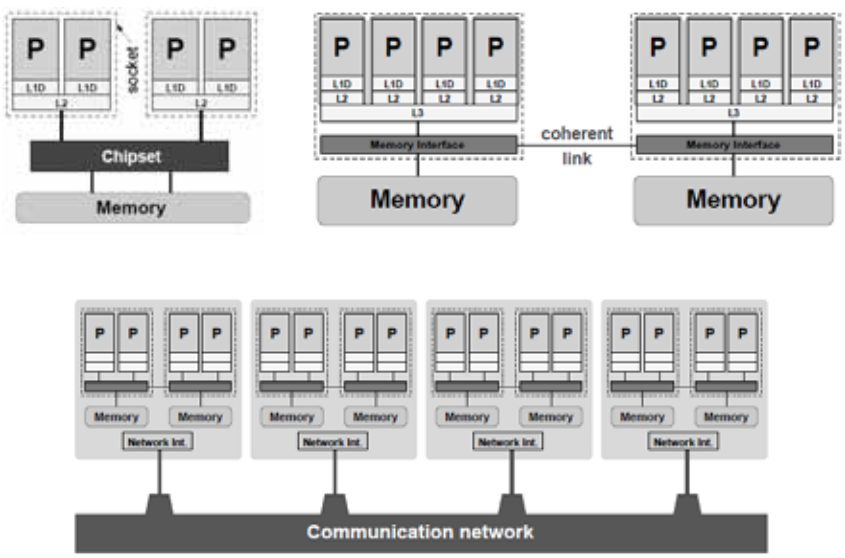
*coffee break*

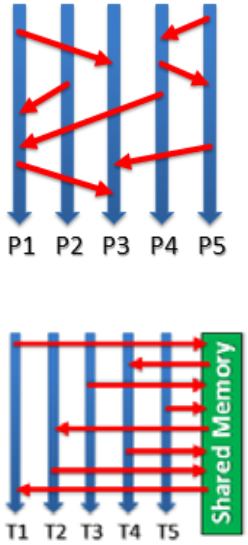
## Part Two ‘Big Data‘ & Distributed Computing Tools

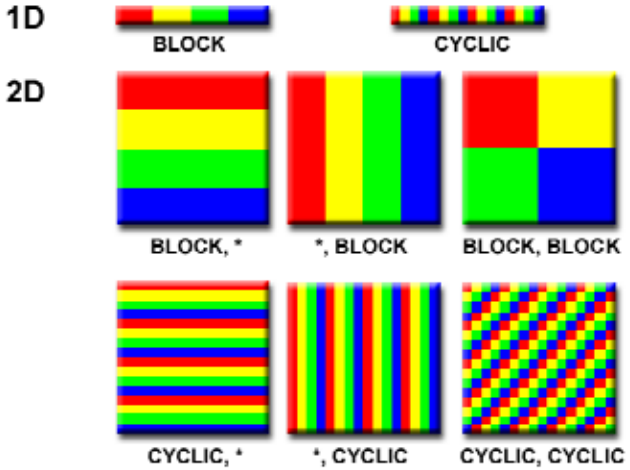
- Exploring Parallel & Distributed Computing Approaches
- Examples of Map-Reduce & ‘Big Data‘ Processing with Hadoop
- Tools for handling ‘Big Data‘ storage & replication methods
- Technologies for Large-scale distributed ‘Big Data‘ Management

# Motivation: Increasing complexities in traditional HPC

- **Different HPC Programming elements (barriers, mutexes, shared-/distributed memory, etc.)**
- **Task distribution issues (scheduling, synchronization, inter-process-communication, etc.)**
- **Complex heterogenous architectures (UMA, NUMA, hybrid, various network topologies, etc.)**
- **Data/Functional parallelism approaches (SMPD, MPMD, domain decomposition, ghosts/halo, etc. )**



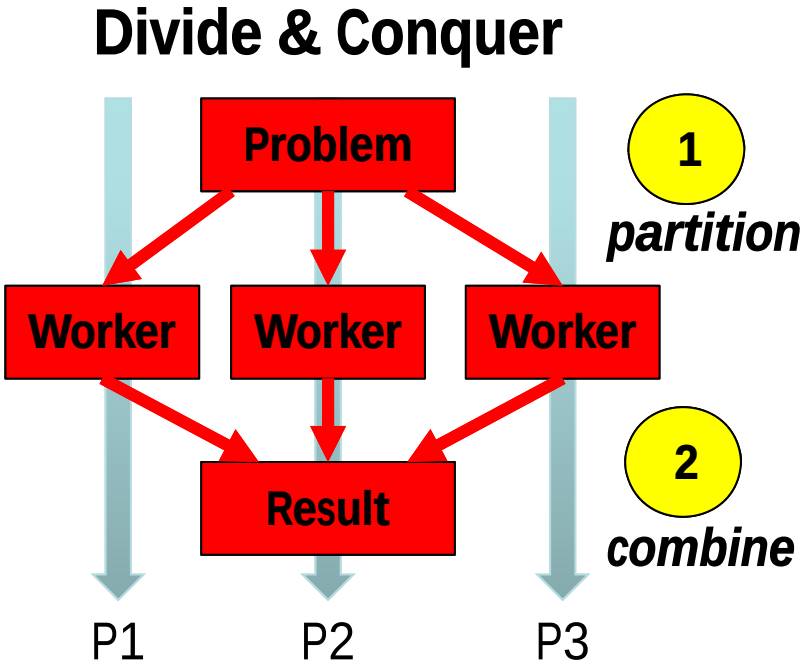

*[5] Parallel Computing Tutorial*

*[4] Introduction to High Performance Computing for Scientists and Engineers*

- **More recently, increasing complexity for scientists working with GPGPU solutions (e.g. CUDA, etc.)**

# Inspired by Traditional Model in Computer Science

Break ‘big tasks’ in many sub-tasks and aggregate/combine results




- (1) Partition the whole problem space
- (2) Combine the partly solutions of each partition to a whole solution of the problem

# Origins of the Map-Reduce Programming Model

## Origin: Invented via the proprietary Google technology by Google technologists


- Drivers: Applications and 'data mining approaches' around the Web
- Foundations go back to functional programming (e.g. LISP)

***[6] MapReduce: Simplified Dataset on Large Clusters, 2004***

## Established 'open source community'


***[7] Apache Hadoop***

- Apache Hadoop in production (mostly business)
- Open Source Implementation of the 'map-reduce' programming model
- Based on Java programming language
- Broadly used – also by commercial vendors within added-value software
- Foundation for many higher-level algorithms, frameworks, and approaches

# Map-Reduce Programming Model

Enables many 'common calculations easily' on large-scale data

- Efficiently performed on computing clusters (but security critics exists)

Offers system that is tolerant of hardware failures in computation

Simple Programming Model

- Users need to provide two functions Map & Reduce with key/value pairs
- Tunings are possible with numerous configurations & combine operation

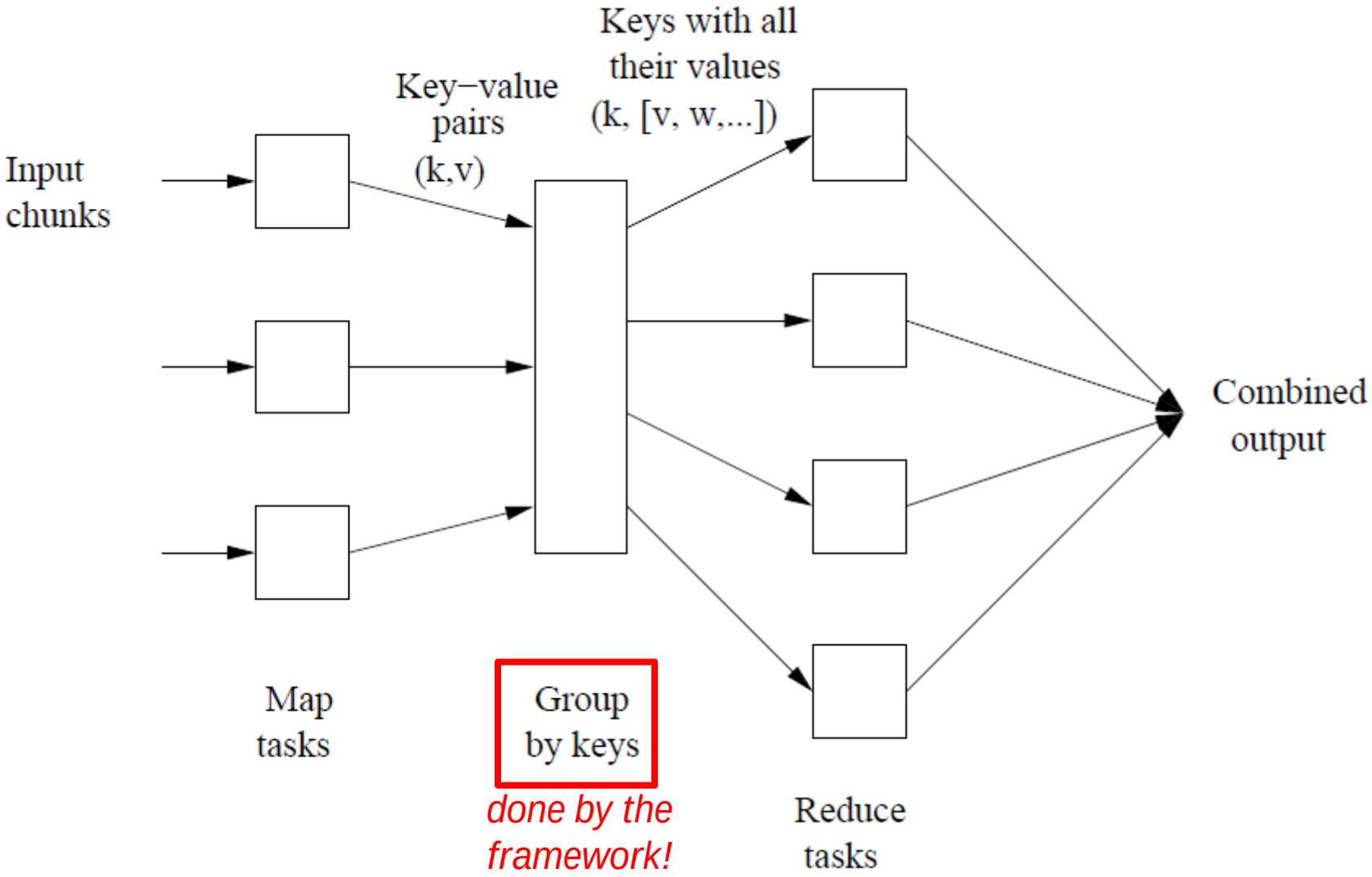
Key to the understanding: The Map-Reduce Run-Time

- Three phases – not just 'map-reduce'

  - **Takes care or the partitioning of input data and the communication**
  - **Manages parallel execution and performs sort/shuffle/grouping**
  - **Coordinates/schedules all tasks that either run Map and Reduce tasks**
  - **Handles faults/errors in execution and re-submit tasks**

  - **Experience from Practice: Talk to your users what they want to do with map-reduce exactly – algorithm implemented, developments?**

# Understanding Map-[Sort/Shuffle/Group]-Reduce




***Modified from [8] Mining of Massive Datasets***

# Key-Value Data Structures

Two key functions to program by user: map and reduce

Third phase 'sort/shuffle' works with keys and sorts/groups them

Input keys and values **(k1,v1)** are drawn from a different domain than the output keys and values **(k2,v2)**

Intermediate keys and values **(k2,v2)** are from the same domain as the output keys and values

- **map (k1,v1) → list(k2,v2)**
- **reduce (k2,list(v2)) → list(v2)**

# Key-Value Data Structures Programming Effort

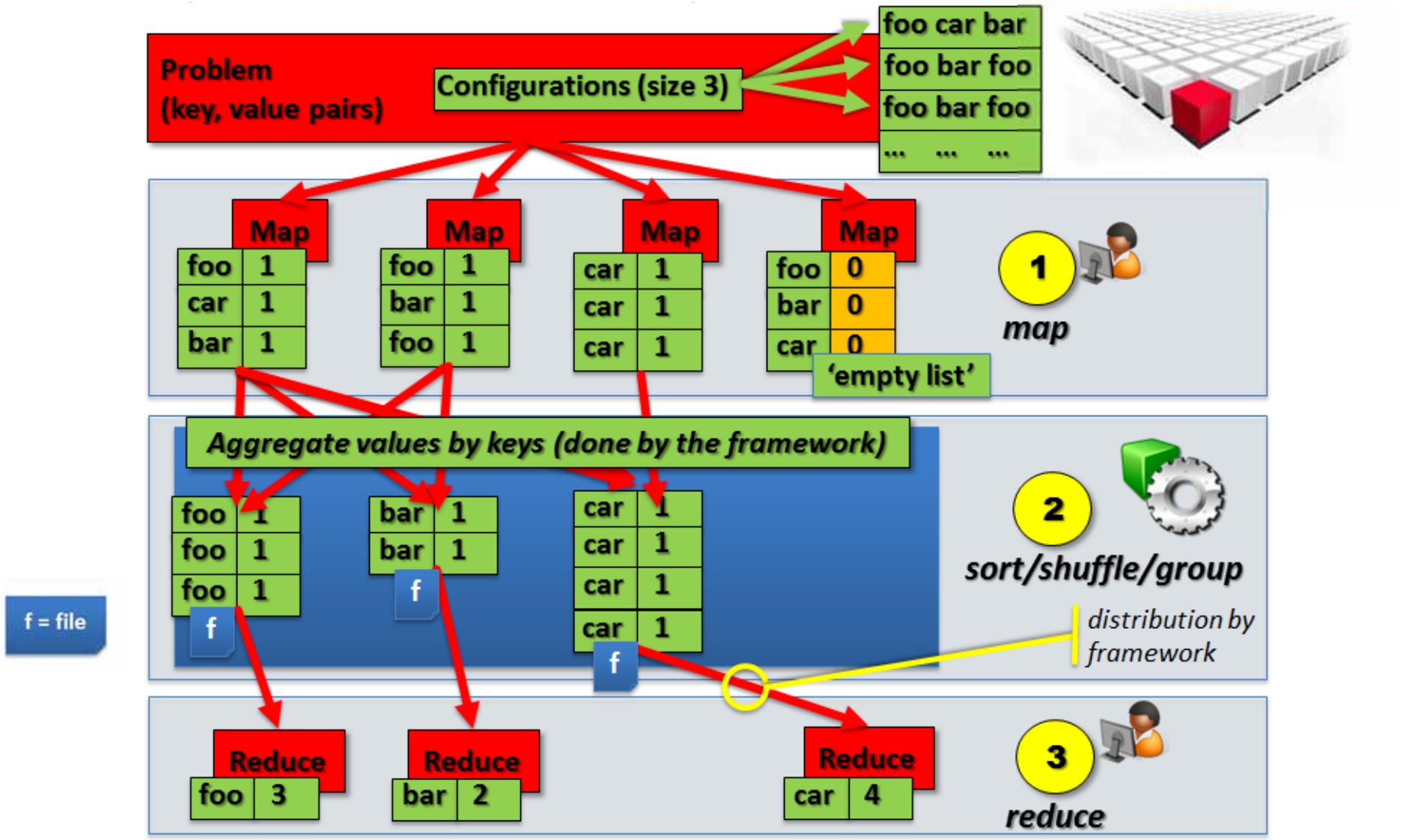
- **map (k1,v1) → list(k2,v2)**
- **reduce (k2,list(v2)) → list(v2)**

```
// counting words example

map(String key, String value):
  // key: document name
  // value: document contents
for each word w in value:
EmitIntermediate(w, "1");

// the framework performs sort/shuffle
// with the specified keys

reduce(String key, Iterator values):
// key: a word
// values: a list of counts
int result = 0;
for each v in values:
result += ParseInt(v);
Emit(AsString(result));
```

- **Goal: Counting the number of each word appearing in a document (or text-stream more general)**
- **Key-Value pairs are implemented as Strings in this text-processing example for each function and as 'Iterator' over a list**
- **Map (docname, doctext) →list (wordkey, 1), …**
- **Reduce (wordkey, list (1, …)) →list (numbercounted)**

***[6] MapReduce: Simplified Dataset on Large Clusters, 2004***

# Map-Reduce Example: WordCount



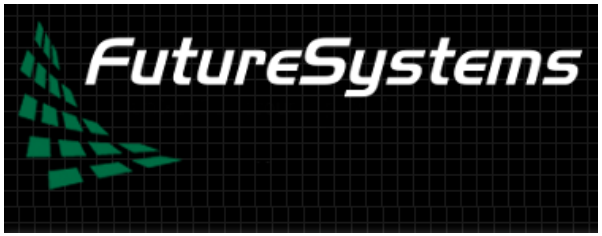
# Map-Reduce on FutureGrid/FutureSystems

## Transition Process FutureGrid/FutureSystems

- Affects straightforward use, but elements remain
- Contact m.riedel@fz-juelich.de if interested


## UoIceland Teaching Project

- Apply for an account
- Upload of SSH is necessary




## Close to real production environment

- Batch system (Torque) for scheduling

## myHadoop → Torque

- Is a set of scripts that configure and instantiate Hadoop as a batch job
- myHadoop is currently installed on four different systems
  - Alamo, Hotel, India, Sierra

***[17] FutureGrid/FutureSystems UoIceland Teaching Project***


***[18] FutureGrid/FutureSystems Submit WordCount Example***

# Further Map-Reduce example: URL Access

Very similar to the WordCount example is the following one:

## Count of URL Access Frequency (‘How often people use a page’)

- Google (and other Web organizations) store billions of logs (‘information’)
- Users independently click on pages or follow links → nice parallelization
- Map function here processes logs of Web page requests → (URL, 1)
- Reduce function adds togeter all values for same URL → (URL, N times)

***[6] MapReduce: Simplified Dataset on Large Clusters***

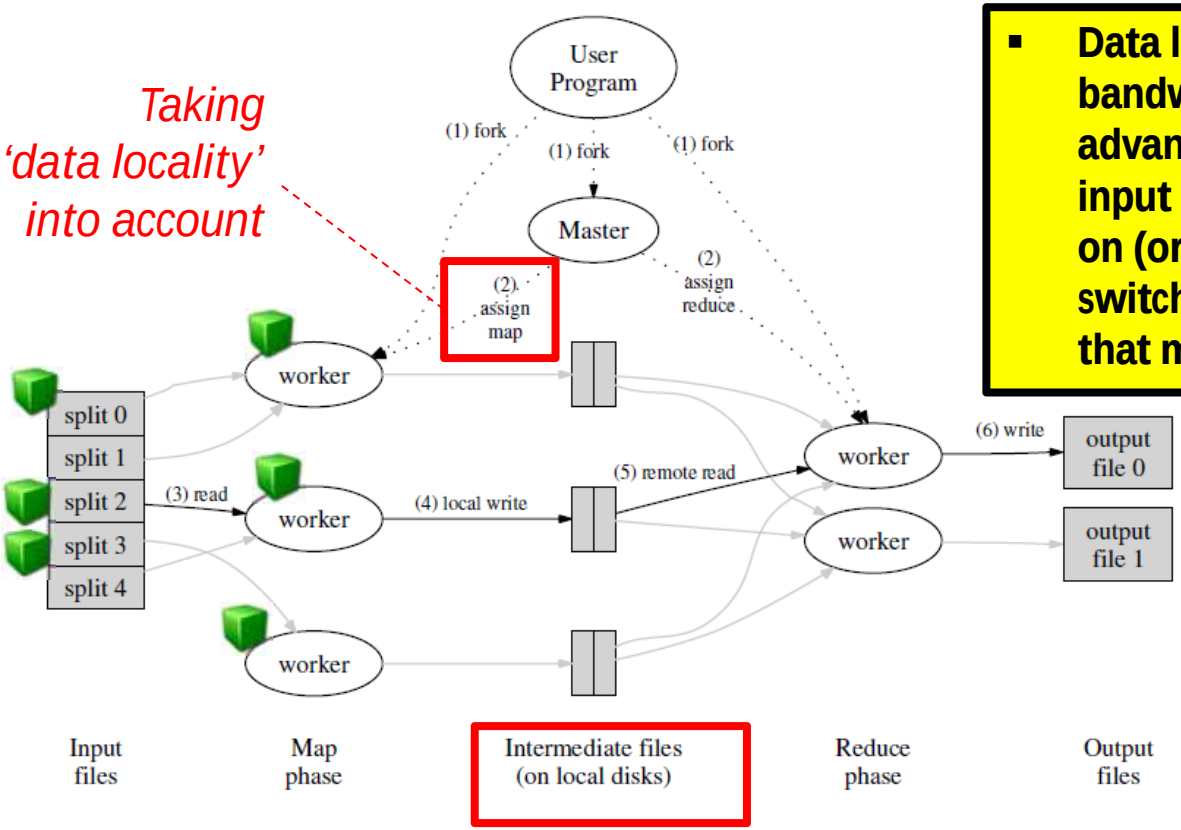
- **Many examples and applications are oriented towards processing large quantities of ‘Web data text’**
- **Examples are typically not scientific datasets or contents of traditional business warehouse databases**

# Communication in Map-Reduce Algorithms

Greatest cost in the communication of a map-reduce algorithm

Algorithms often trade communication cost against degree of parallelism and use principle of 'data locality' (use less network)

*Taking 'data locality' into account*

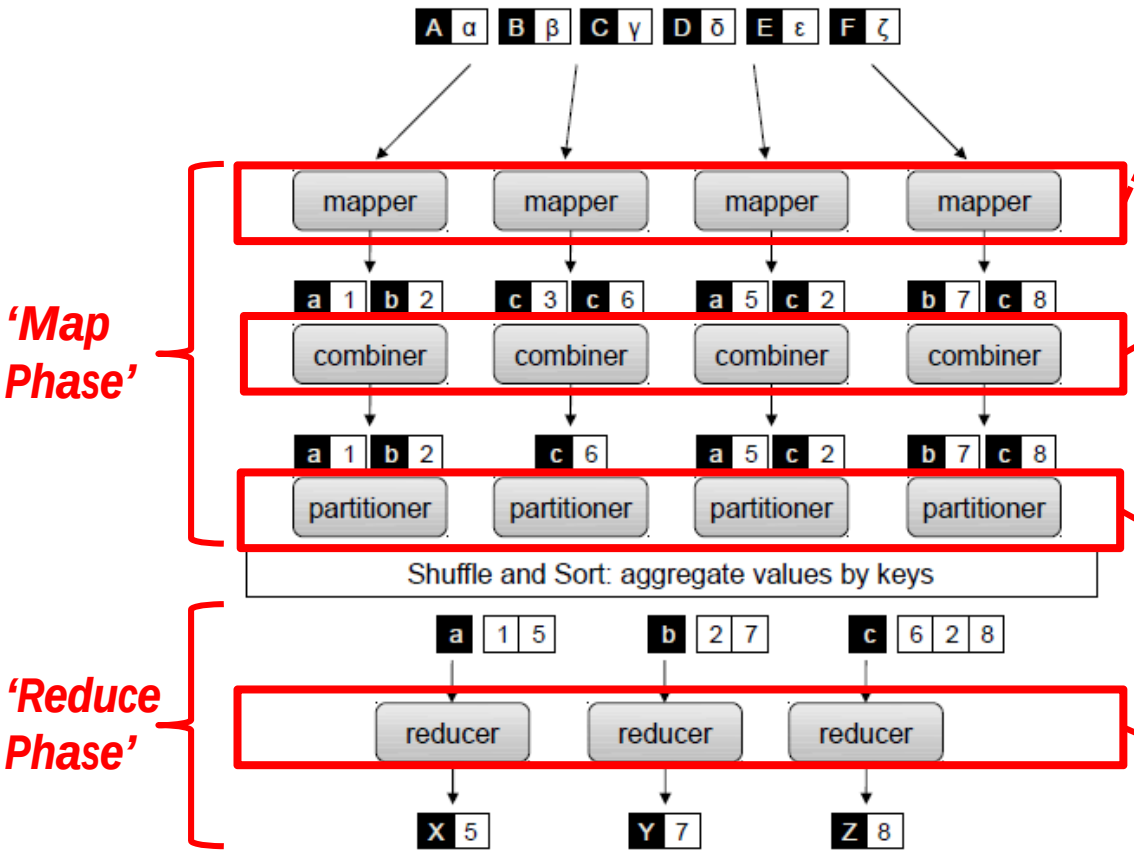



- **Data locality means that network bandwidth is conserved by taking advantage of the approach that the input data (managed by DFS) is stored on (or very near, e.g. same network switch) the local disks of the machines that make up the computing clusters**

***Modified from [6] MapReduce: Simplified Dataset on Large Clusters***

# Map-Reduce Optimizations – Local Combiner

- **The Map-Reduce programming model is based on Mapper, Combiner, Partitioner, and Reducer functionality supported by powerful shuffle/sort/aggregation of keys by the framework**




- **Mapper functionality is applied to input data and computes intermediate results in a distributed fashion**
- **(local) Combiner functionality is applied in-memory to Map outputs and performs local aggregation of its results**
- **Partitioner determines to which reducer intermediate data is shuffled (cf. Computer Vision)**
- **Reduce functionality is applied to intermediate input data from the Map-Phase and aggregates it for results**

***modified from [14] Computer Vision using Map-Reduce***

# Not every problem can be solved with Map-Reduce

- **Map-Reduce is not a solution to every parallelizable problem**
- **Only specific algorithms benefit from the map-reduce approach**
- **No communication-intensive parallel tasks (e.g. PDE solver) with MPI**
- **Applications that require often updates of existing datasets (writes)**
- **Implementations often have severe security limitations (distributed)**

***[8] Mining of Massive Datasets***

## Example: Amazon Online retail sales

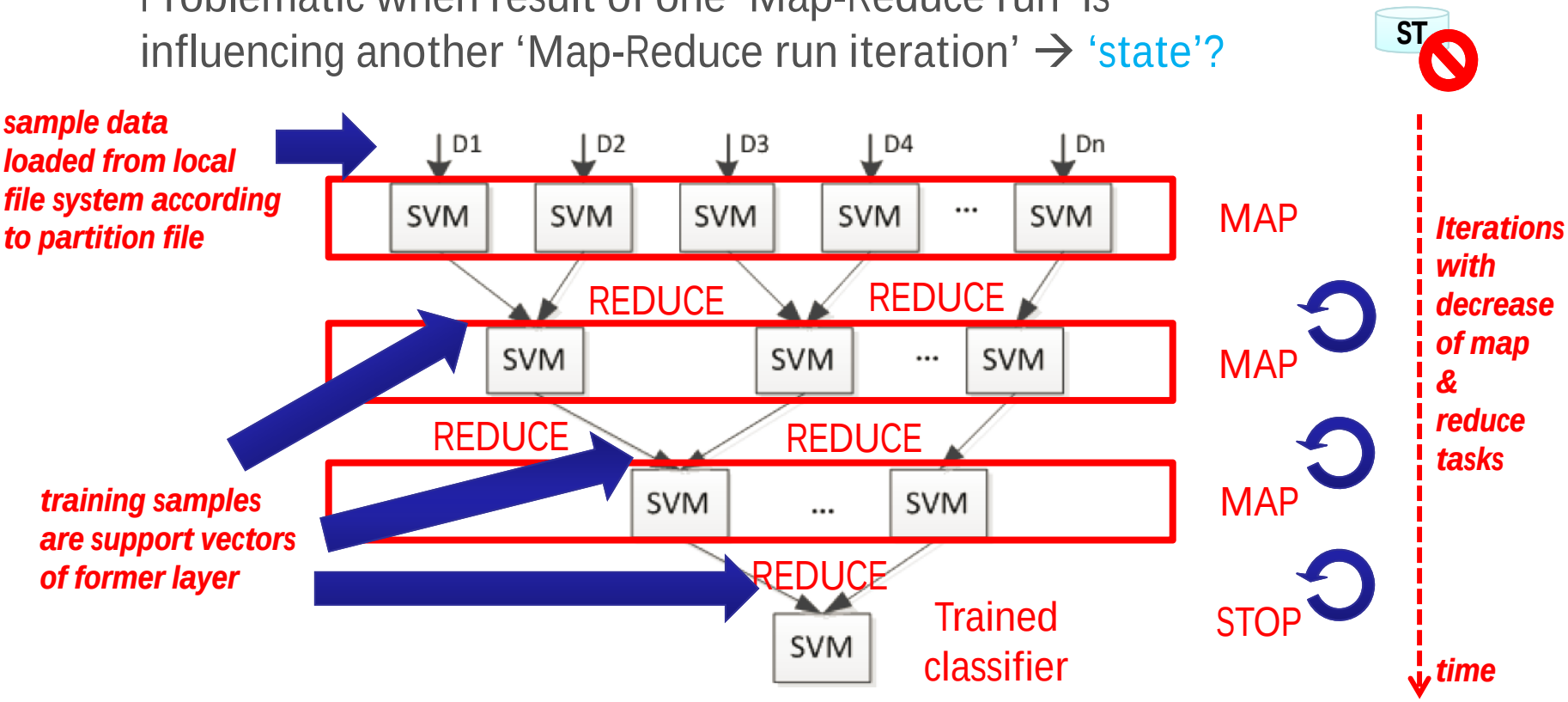
- Requires a lot of updates on their product databases on each user action (a problem for the underlying file system optimization)
- Processes that involve 'little calculation' but still change the database
- Employs thousands of computing nodes and offers them ('Amazon Cloud')
- (maybe they using map-reduce for certain analytics: buying patterns)

# Map-Reduce Limitation: Missing some form of 'State'

## What if 'values depend on previously computed values'?

## 'Map-Reduce runs' map & reduce tasks in parallel and finishes

- Problematic when result of one 'Map-Reduce run' is influencing another 'Map-Reduce run iteration' → 'state'?

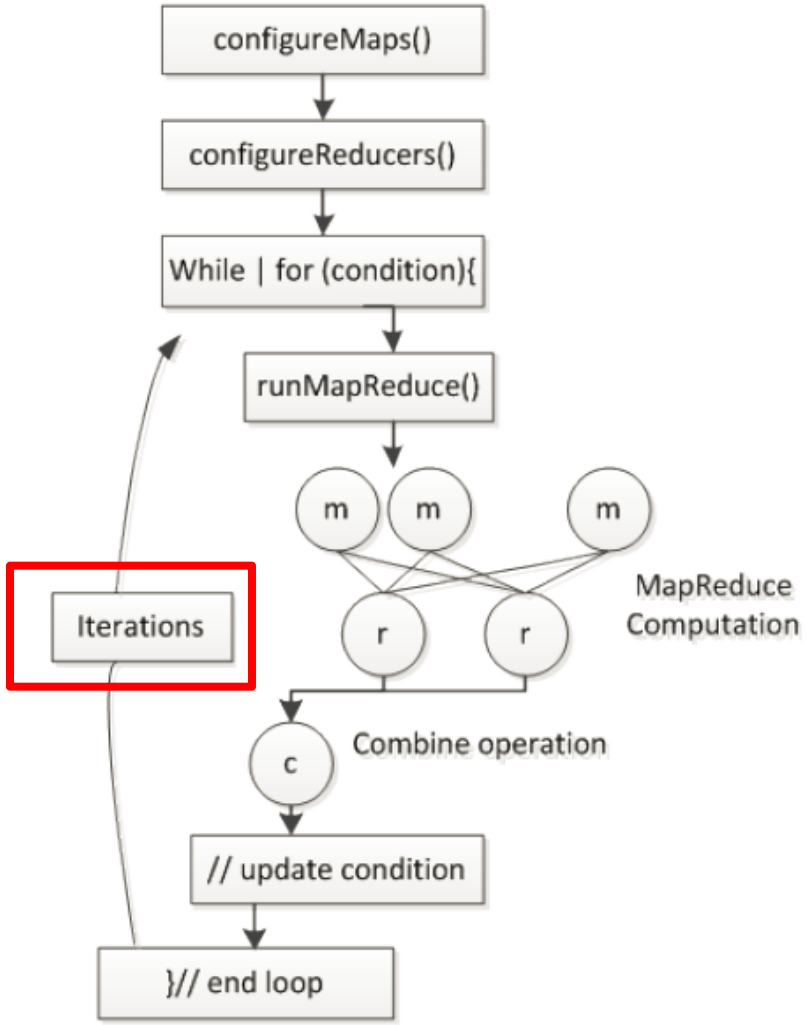



*[15] 'Study on Parallel SVM Based on MapReduce'*

# The Need for Iterative Map-Reduce

## ‘Map-Reduce’ runs map and reduce tasks and finishes

- **Many parallel algorithms are ‘iterative in nature’**
- **Example from many application fields such as data clustering, dimension reduction, link analysis, or machine learning**
- **‘Iterative Map-Reduce’ enables algorithms with same Map-Reduce tasks ideas, but added is a loop to perform iterations until conditions are satisfied**
- **The transfer of ‘states’ from one iteration to another iteration is specifically supported in this approach**




***modified from [12] Z. Sun et al.***

# Iterative Map-Reduce with Twister

## MapReduce jobs are controlled by the 'Client node'

- Via a multi-step process


***[13] Twister Web page***

## Configuration phase: The Client…

- … assigns Map-Reduce methods to the job
- … prepares KeyValue pairs
- … prepares 'static data' for Map-Reduce tasks (through the partition file) if required

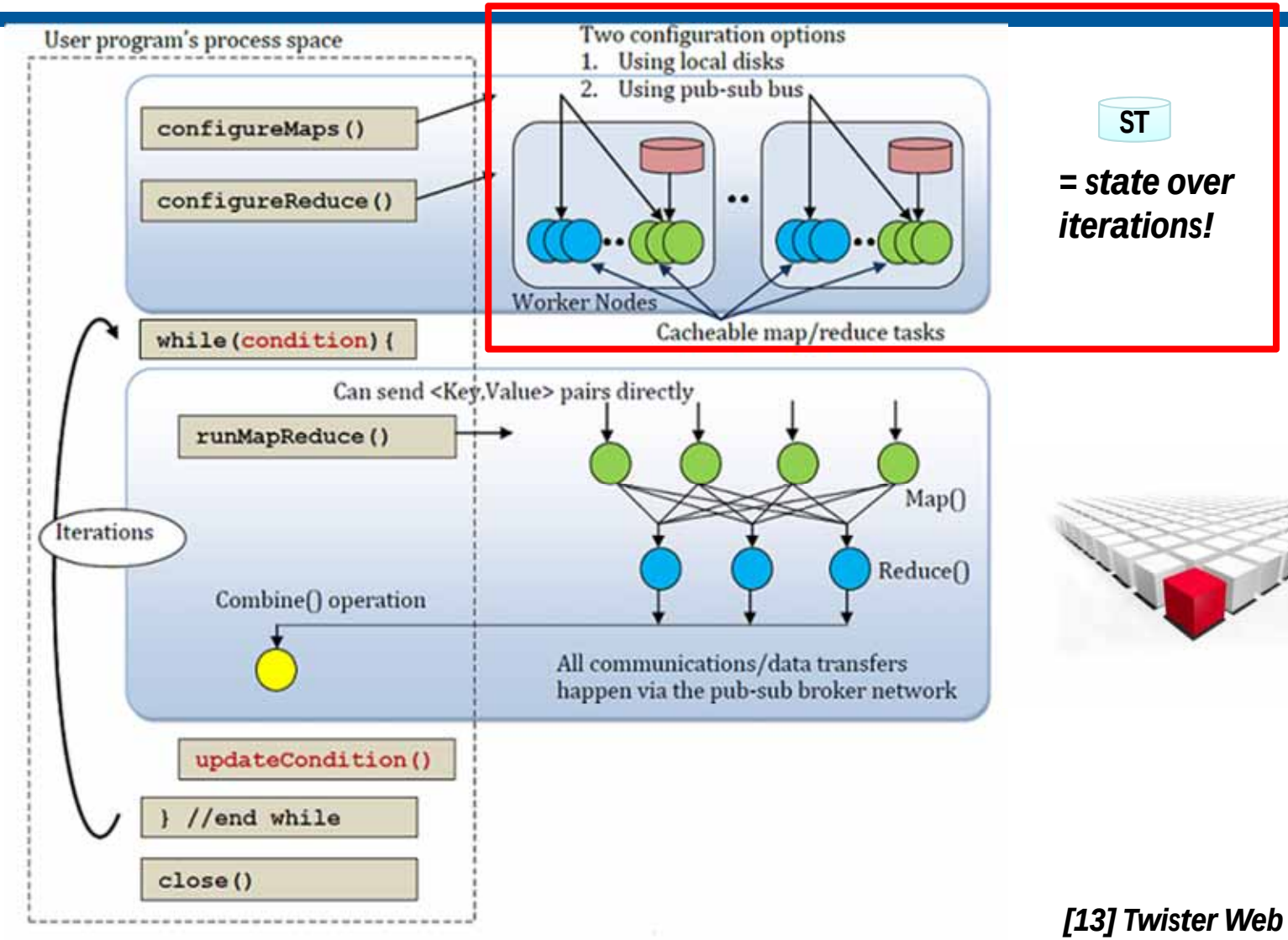
## Running phase: The Client…

- … between iterations receives results collected by the 'combine method'
- … exits gracefully when the job is done (check condition)

## Message communication between jobs

- Realized with 'message brokers', i.e. NaradaBrokering or ActiveMQ

# Overview: Iterative Map-Reduce with Twister






*[13] Twister Web page*

# Distribute ‘Small State’ with Distributed Cache

## Mappers need to read ‘state’ from a single file (vs. data chunks)

## Example: Distributed spell-check application

- Every Mapper reads same copy of the dictionary before processing docs
- Dictionary (‘state’) is small (~ MBs), but all nodes need to reach it

## Solution: Hadoop provides ‘DistributedCache’

- Optimal to contain ‘small files’ needed for initialization (or shared libraries even)
- ‘State’ (or input data= needed on all nodes of the cluster
- Simple use with Java ‘DistributedCache Class’
- Method ‘AddCacheFile()’ add names of files which should be sent to all nodes on the system (need to be in HDFS) by the framework


***[16] Hadoop Distributed Cache***


***= state of data per iteration or small data required by every node***

# Hadoop 2.x vs. Hadoop 1.x Releases

## Apache Hadoop 1.x had several limitations, but is still used a lot

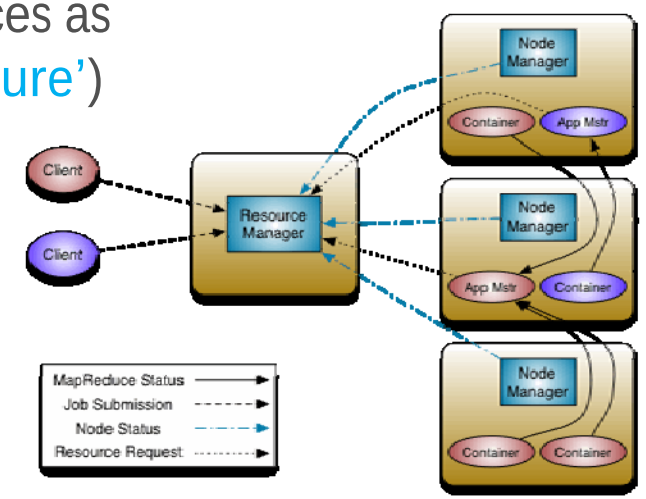
- 'Apache Hadoop 2.x' consisted of significant improvements
- New Scheduler 'Hadoop YARN' with lots of configuation options
  (YARN = Yet Another Resource Negotiator: Map-reduce used beyond its idea)

## HDFS Improvements

- Use multiple independent Namenodes/Namespaces as 'federated system' (addressing 'single point of failure')

## Map-Reduce Improvements

- JobTracker functions are seperated into two new components
  (addressing 'single point of failure')
- New 'ResourceManager' manages the global assignment of compute resources to applications
- New 'ApplicationMaster' manages the application scheduling and coordination




***[11] Hadoop 2.x***

***different scheduling policies can be configured***

# Apache Hadoop Key Configuration Overview

## core-site.xml

```
<configuration>
 <property>
   <name>fs.default.name</name>
   <value>hdfs://localhost:9000</value>
 </property>
</configuration>
```

## hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
   </property>
</configuration>
```

## mapred-site.xml

```
<configuration>
 <property>
   <name>mapred.job.tracker</name>
   <value>hdfs://localhost:9001</value>
 </property>
</configuration>
```


E.g. NameNode, Default status page:
http://localhost:50070/


E.g. JobTracker, Default status page:
http://localhost:50030/

- **core-site.xml contains core properties of Hadoop (fs.default.name is just one example)**
- **hdfs-site.xml contains properties directly related to HDFS (e.g. dfs.replication is just one example)**
- **mapred-site.xml contains properties related to the map-reduce programming environment**

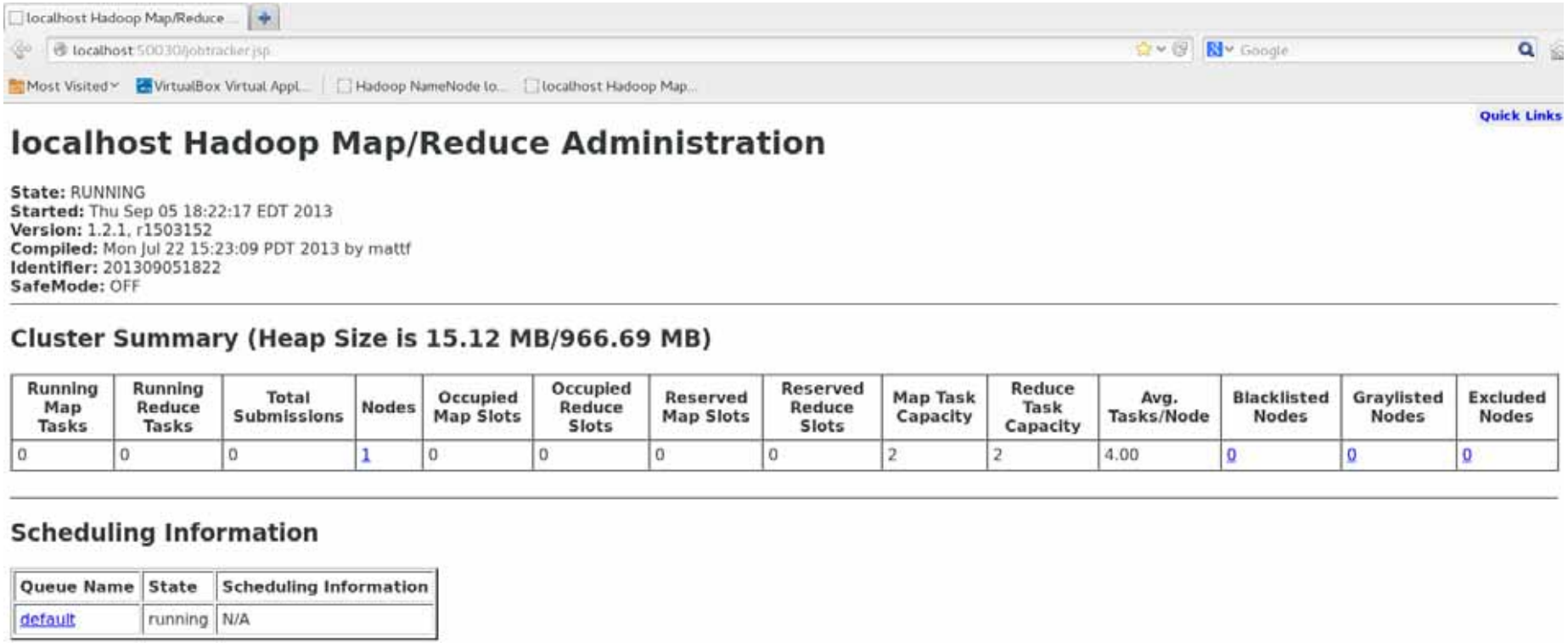
# Hadoop Selected Administration & Configuration (1)

## Release Download

- Well maintained, often new versions
- JobTracker, e.g. max. of map / reduce jobs per node

→ Check Webpage

***[7] Apache Hadoop***



# Hadoop Selected Administration & Configuration (2)

morris@login-0-0:~/bin/hadoop-1.2.1/conf

File Edit View Search Terminal Help

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
<name>mapred.job.tracker</name>
<value>localhost:9001</value>
</property>
</configuration>
~
~
~
~
~
~
~
~
"mapred-site.xml" 11L, 262C
```

**conf/mapred-site.xml**

**Version 2: JobTracker split : resource management and job scheduling/monitoring**

```
<property>
 <name>mapreduce.framework.name</name>
 <value>yarn</value>
</property>
```

**Yarn for resource management**

**conf/yarn-site.xml**

| | | |
|---|---|---|
| yarn.nodemanager.aux-services | mapreduce.shuffle | Shuffle service that needs to be set for Map Reduce applications. |
| yarn.nodemanager.aux-services.<br>mapreduce.shuffle.class | org.apache.hadoop.mapred.<br>ShuffleHandler | The exact name of the class for shuffle service |
| yarn.resourcemanager.address | resourcemanager.company.com:8032 | The address of the applications manager interface in the RM. |
| yarn.resourcemanager.<br>scheduler.address | resourcemanager.company.com:8030 | The address of the scheduler interface. |

# Hadoop 1.2.1 Usage Examples

```
[morris@login-0-0 hadoop-1.2.1]$ ./bin/hadoop
Usage: hadoop [--config confdir] COMMAND
where COMMAND is one of:
  namenode -format     format the DFS filesystem
  secondarynamenode    run the DFS secondary namenode
  namenode             run the DFS namenode
  datanode             run a DFS datanode
  dfsadmin             run a DFS admin client
  mradmin              run a Map-Reduce admin client
  fsck                 run a DFS filesystem checking utility
  fs                   run a generic filesystem user client
  balancer             run a cluster balancing utility
  oiv                  apply the offline fsimage viewer to an fsimage
  fetchdt              fetch a delegation token from the NameNode
  jobtracker           run the MapReduce job Tracker node
  pipes                run a Pipes job
  tasktracker          run a MapReduce task Tracker node
  historyserver        run job history servers as a standalone daemon
  job                  manipulate MapReduce jobs
  queue                get information regarding JobQueues
  version              print the version
  jar <jar>            run a jar file
  distcp <srcurl> <desturl> copy file or directories recursively
  distcp2 <srcurl> <desturl> DistCp version 2
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archive
  classpath            prints the class path needed to get the
                       Hadoop jar and the required libraries
  daemonlog            get/set the log level for each daemon
 or
  CLASSNAME            run the class named CLASSNAME
Most commands print help when invoked w/o parameters.
```

```
./bin/hadoop jar hadoop-examples-1.2.1.jar wordcount DaVinciExample DaVinciExampleOutput
```

# Apache Hadoop Architecture Elements

## NameNode (and Secondary NameNode)

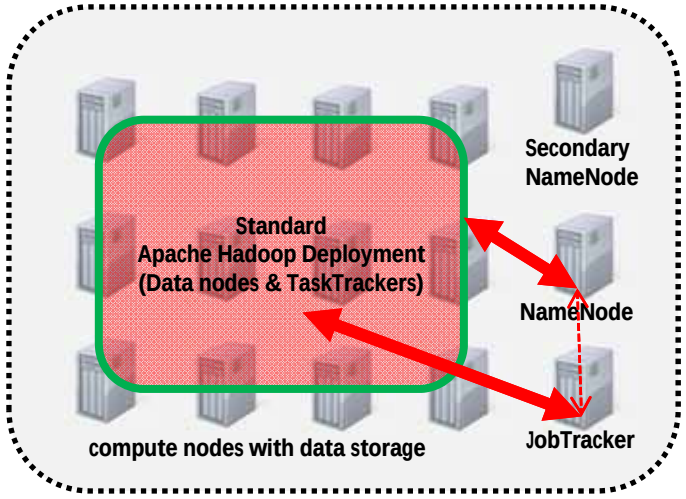
- 'Service' that has all information about the HDFS file system ('data nodes')

## JobTracker (point of failure → no secondary instance!)

- 'Service' that 'farms out' map-reduce tasks to specific nodes in the cluster

## TaskTracker (close to DataNodes, offering 'job slots' to submit to)

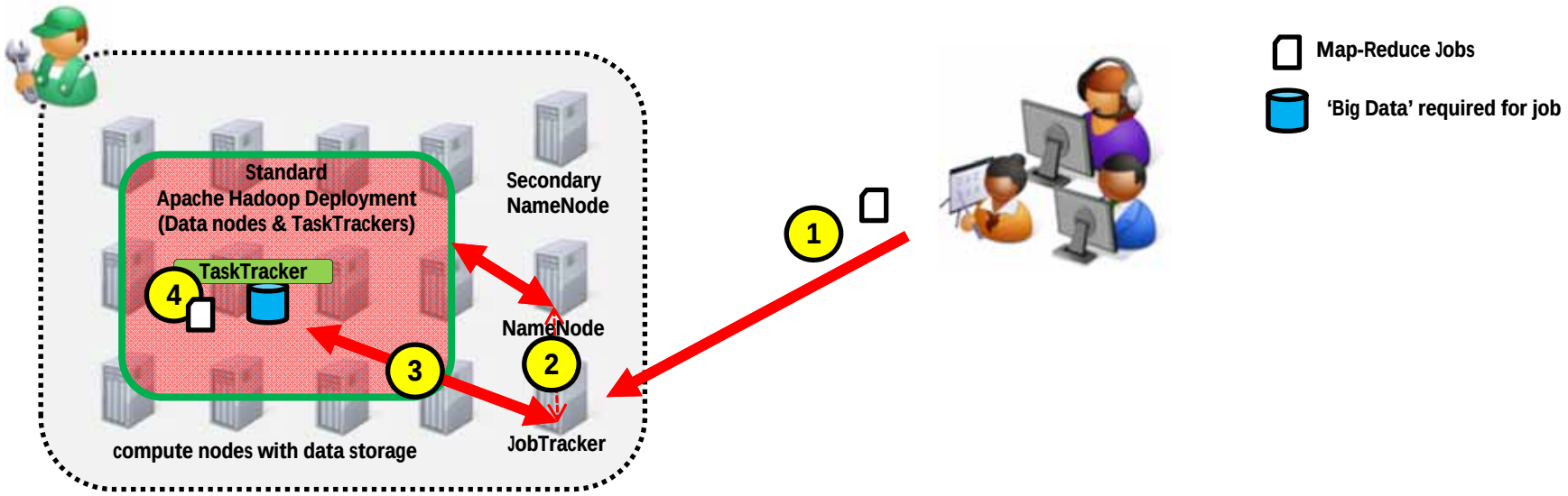
- Entity in a node in the cluster that 'accepts/performs map-reduce tasks'




- DataNode
  - Part of the HDFS filesystem
  - Responds to requests from the NameNode for 'filesystem operations'

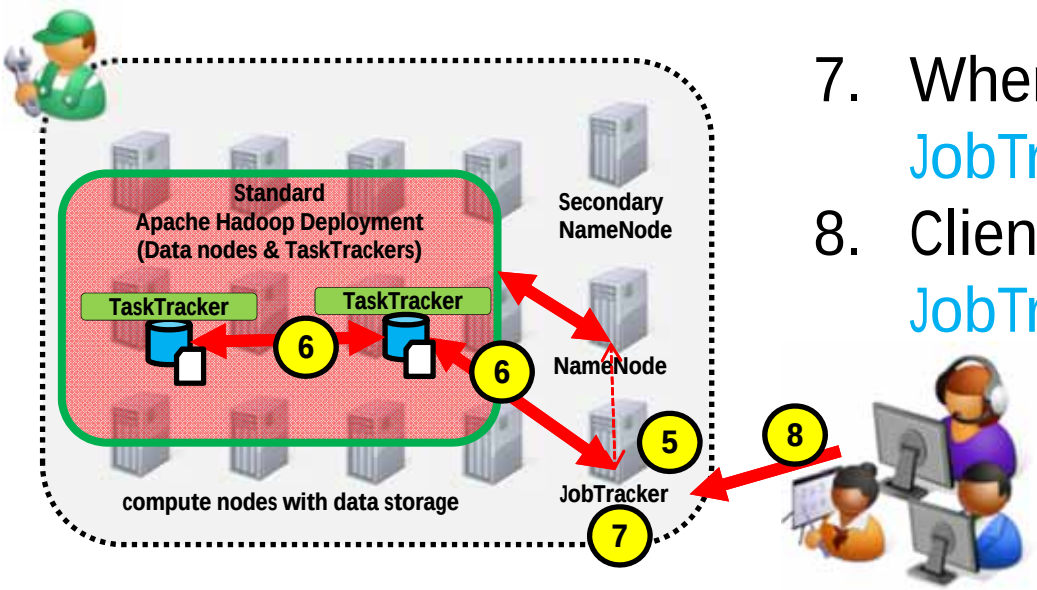
# Reliability & Fault-Tolerance in Map-Reduce (1)

1. Users use the client application to submit jobs to the JobTracker
2. A JobTracker interacts with the NameNode to get data location
3. The JobTracker locates TaskTracker nodes with available slots at or near the data ('data locality principle')
4. The JobTracker submits the work to the chosen TaskTracker nodes



# Reliability & Fault-Tolerance in Map-Reduce (2)

5. The TaskTracker nodes are monitored (ensure 'reliability')
   - 'Fault tolerance': If they do not submit heartbeat signals often enough, they are deemed to have failed & the work is given to different TaskTracker
6. The TaskTracker notifies the JobTracker when a task fails
   - The JobTracker decides next action: it may resubmit the job elsewhere or it may mark that specific record as something to avoid
   - The Jobtracker may may even 'blacklist the TaskTracker as unreliable'




7. When the work is completed, the JobTracker updates its status
8. Client applications can poll the JobTracker for information

# Cluster Setups with Hadoop-On-Demand (HOD)

- **Hadoop On Demand (HOD) is a specific Hadoop distribution for provisioning virtual Hadoop cluster deployments over a large physical cluster that is managed by a scheduler (i.e. Torque).**

## When to use?

- A given physical cluster exists with nodes managed by scheduling system

## 'Semi-Automatic Deployment' approach

- HOD provisions and maintains Hadoop Map-Reduce and HDFS instances through interaction with several HOD components on given physical nodes

## Performs cluster node allocation

- Starts Hadoop Map/Reduce and HDFS daemons on allocated nodes
- Makes it easy for administrators to quickly setup and use Hadoop

## Includes automatic configurations

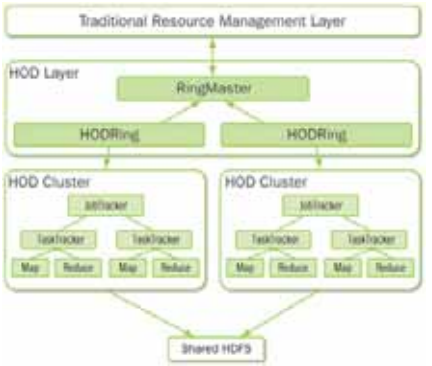
- Generates configuration files for the Hadoop daemons and client

# HOD Deployment on Different Cluster Node Types

## Submit nodes

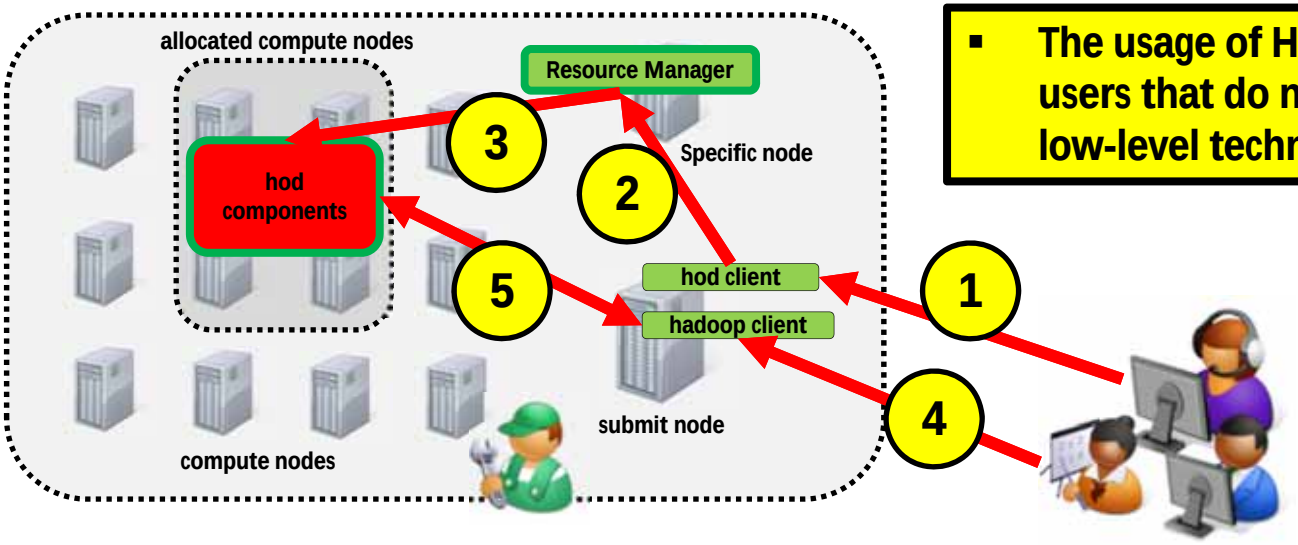
- Users use the HOD client on these nodes to allocate 'cluster nodes' and use the Hadoop client to submit Map-Reduce jobs

***[27] HOD Architecture***




## Compute nodes

- The resource manager runs HOD components on these nodes to provision the Hadoop daemons that enable Map-Reduce jobs




- **The usage of HOD is optimized for users that do not want to know the low-level technical details**

# HOD Detailed Architecture Elements

## Basic System Architecture of HOD includes:

- A Resource Manager & Scheduling system (i.e. Torque)
- Hadoop Map/Reduce and HDFS daemons need to run
- Various HOD components (HOD RingMaster, HOD Rings)

Torque

Map-Reduce Jobs

## HOD RingMaster

hod ringmaster

- Starts as a process of the compute nodes (mother superior, in Torque)
- Uses a resource manager interface (pbsdsh, in Torque)
- Runs the HodRing as distributed tasks on the allocated compute nodes

## HOD Rings

- Communicate with the HOD RingMaster to get Hadoop commands (e.g. new map-reduce jobs) and run them accordingly
- Once the Hadoop commands are started they register with the RingMaster, giving information about the daemons of the HOD Rings

hodring

- **Since map-reduce version 2 HOD is deprected and YARN is the scheduler to be used instead**

# Hadoop Adoptions – In Industry

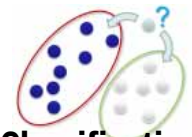

YAHOO! IBM

Smart Data Innovation Lab

***[19] IBM System @ Smart Data Innovation Lab***



***Closed Source Algorithms in Business Solutions (e.g. also IBM SPSS)***




**Classification**

- **Uptake of Hadoop in many different business environments, SMEs, etc.**

# Map-Reduce Deployment Models

- **Map-Reduce deployments are particularly well suited for cloud computing deployments**
- **Deployments need still some useful map-reduce codes (cf. to MPI/OpenMP w/o their codes)**


**Options to move 'data to strong computing power' ...**

**... or move 'compute tasks close to data'**


***High Trust?*** ***Data Privacy*** ***Low Trust?***

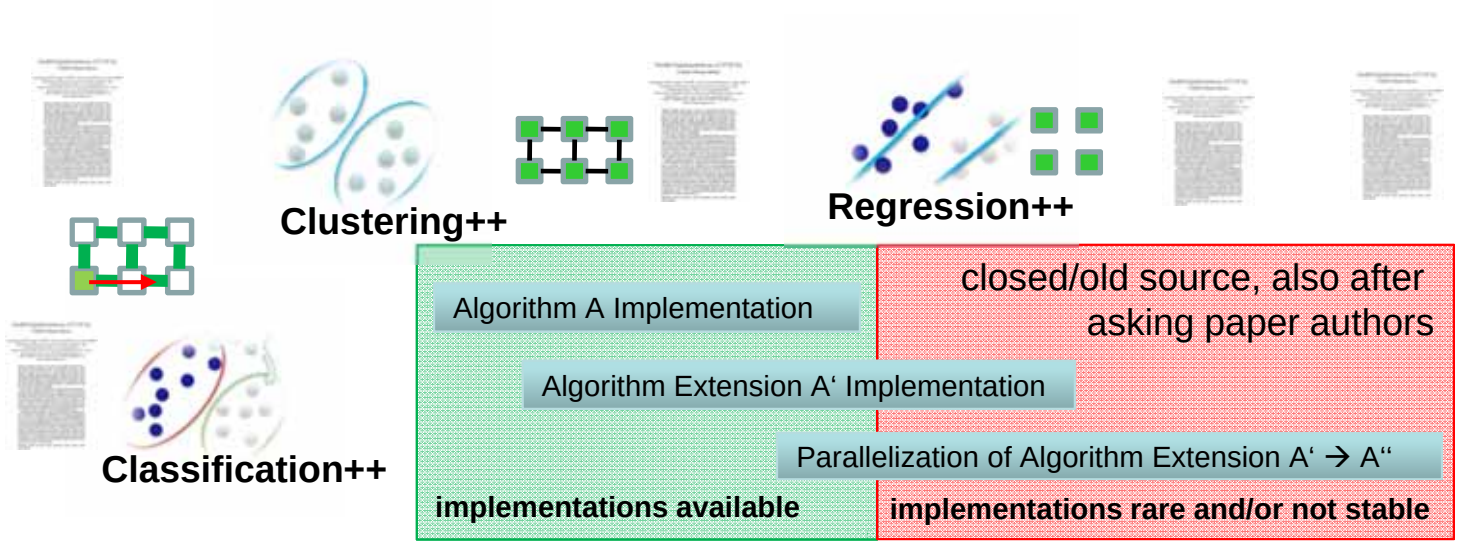
**On-premise full custom** | **Map-Reduce Appliance** | **Map-Reduce Hosting** | **Map-Reduce As-A-Service**

***Bare-metal*** ***Virtualized*** ***Clouds***

***[23] Inspired by a study on Hadoop by Accenture***

# Data Analytics in the view of Open Data Science

## Experience in investigation of available parallel data mining algorithms









- **Stable open source algorithms are still rather rare (Map-reduce, MPI/OpenMP, and GPGPUs)**

# Data Analytics with SVM – Algorithm Availability

| Tool | Platform Approach | Parallel Support Vector Machine |
|---|---|---|
| Apache Mahout | Java; Apache **Hadoop 1.0** (map-reduce); HTC | No strategy for implementation (Website), serial SVM in code |
| Apache Spark/MLlib | Apache Spark; HTC | Only linear SVM; no multi-class implementation |
| Twister/ParallelSVM | Java; Apache **Hadoop 1.0** (map-reduce); Twister (iterations), HTC | Much dependencies on other software: Hadoop, Messaging, etc. Version 0.9 development |
| Scikit-Learn | Python; HPC/HTC | Multi-class Implementations of SVM, but not fully parallelized |
| piSVM | C code; Message Passing Interface (MPI); HPC | Simple multi-class parallel SVM implementation outdated (~2011) |
| GPU accelerated LIBSVM | CUDA language | Multi-class parallel SVM, relatively hard to program, no std. (CUDA) |
| pSVM | C code; Message Passing Interface (MPI); HPC | Unstable beta, SVM implementation outdated (~2011) |

# Lessons Learned – Hadoop & Map-Reduce

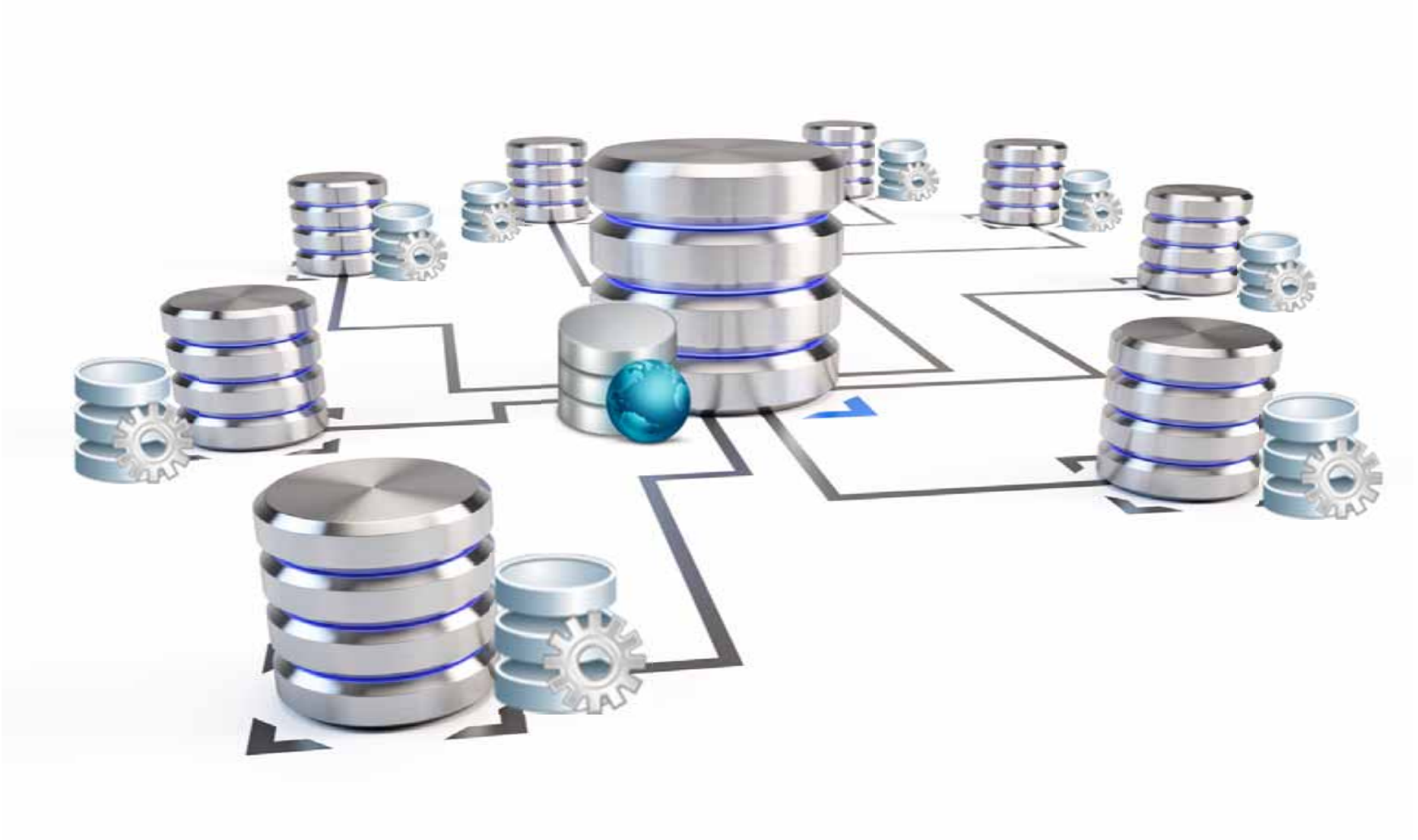
## Be careful of investing time & efforts

- Frameworks keep significantly fast changing, no peer-reviews as in HPC (e.g. Hadoop 1.0 → Hadoop 2.0, new move of community to Spark)
- Map-reduce basically standard, but not as stable as established MPI or OpenMP
- Hadoop 2.0 improvements with YARN to work in 'HPC scheduling environments'
- Consider and observe developments around Apache Spark

## Solutions on-top-of Hadoop keep changing

- Many different frameworks are available on top of Hadoop
- Often business-driven developments (e.g. to be used in recommender systems)
- Data Analytics with Mahout have only a limited number of algorithms (E.g. Decision trees, collaborative filtering, no SVMs, no artifical neural networks)
- Data Analytics with Twister works, but limited algorithms (E.g. SVM v.0.9 works, but old development/research version, unmaintained)

# Part Two – Questions



# Part Two – Outline

## Part One ‘Big Data‘ Challenges & HPC Tools

- Understanding ‘Big Data‘ in Science & Engineering
- Statistical Data Mining and Learning from ‘Big Data‘
- OpenMP/MPI Tool Example for Clustering ‘Big Data‘
- MPI Tool Example for Classification of ‘Big Data‘

*coffee break*

## Part Two ‘Big Data‘ & Distributed Computing Tools

- Exploring Parallel & Distributed Computing Approaches
- Examples of Map-Reduce & ‘Big Data‘ Processing with Hadoop
- Tools for handling ‘Big Data‘ storage & replication methods
- Technologies for Large-scale distributed ‘Big Data‘ Management

# Distributed File Systems vs. Parallel File Systems

## Distributed File Systems

- Clients, servers, and storage devices are geographically dispersed among machines of a distributed system (often appear as ‘single system’)
- Manage access to files from multiple processes
- But generally treat ‘concurrent access’ as an unusual event
- E.g. Hadoop Distributed File System (HDFS) implementation

## Parallel File Systems

- Deal with many problematic questions arising during ‘parallel programming’
- E.g. How can hundreds or thousands of processes access the same file concurrently and efficiently?
- E.g. How should file pointers work?
- E.g. Can the UNIX sequential consistency semantics be preserved?
- E.g. How should file blocks be cached and buffered?

# Hadoop Distributed File System (HDFS)

## A ‘specialized filesystem’ designed for reliability and scalability

- Designed to run on ‘comodity hardware’
- Many similarities with existing ‘distributed file systems’
- Takes advantage of ‘file and data replication concept’

## Differences to traditional distributed file systems are significant

- Designed to be ‘highly fault-tolerant’ → improves dependability!
- Enables use with applications that have ‘extremely large data sets’

## Provides ‘high throughput access’ to application data

- HDFS relaxes a few POSIX requirements
- Enables ‘streaming access’ to file system data

## Origin

- HDFS was originally built as infrastructure for the ‘Apache Nutch web search engine project’

***[9] The Hadoop Distributed File System***

# HDFS Key Feature: File Replication Concept

- **File replication is a useful redundancy for improving availability and performance**

## Ideal: Replicas reside on ‘failure-independent machines’

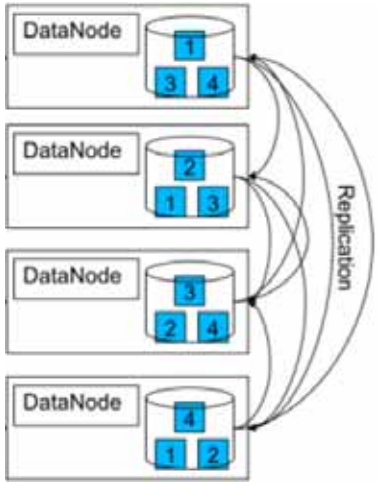
- Availability of one replica should not depend on availability of others
- Requires ability to place replica on particular machine
- ‘Failure-independent machines’ hard to find, but in ‘system design’ easier

## Replication should be hidden from users

- But replicas must be distinguishable at lower level
- Different DataNode’s are not visible to end-users

## Replication control at higher level

- Degree of replication must be adjustable
  (e.g. Hadoop configuration files)



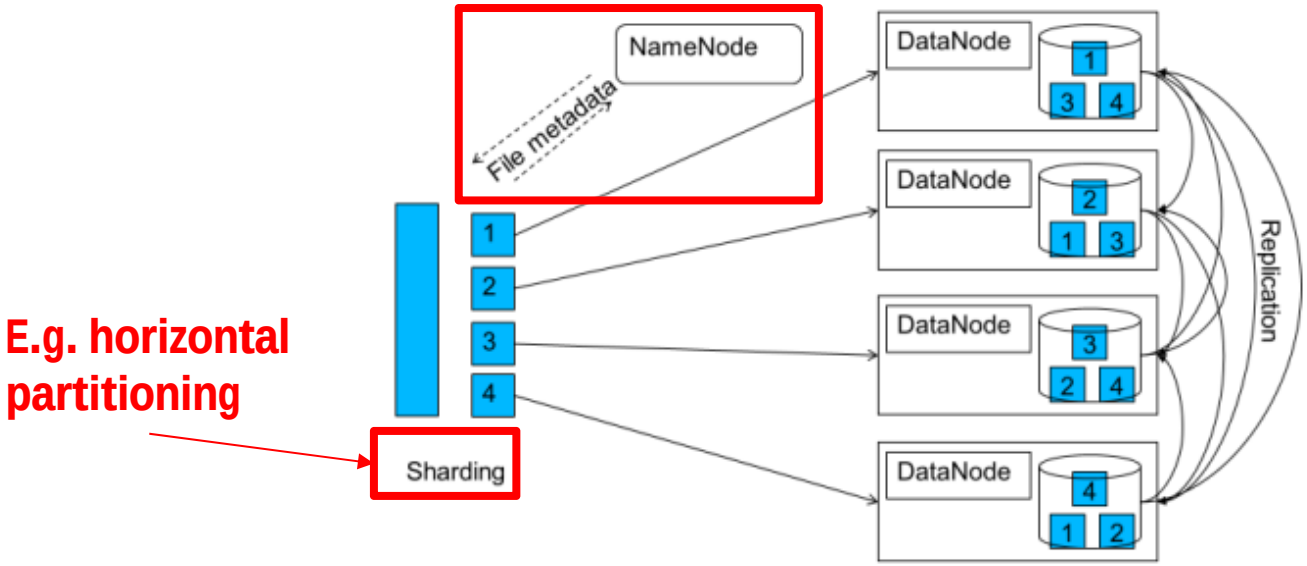

***Modified from [10] Virtual Workshop***

# HDFS Master/NameNode

## The master/name node is replicated

- A directory for the file system as a whole knows where to find the copies
- All participants using the DFS know where the directory copies are

- **The Master/Name node keeps metadata (e.g. node knows about the blocks of files)**




***Modified from [10] Virtual Workshop***

# HDFS File Operations

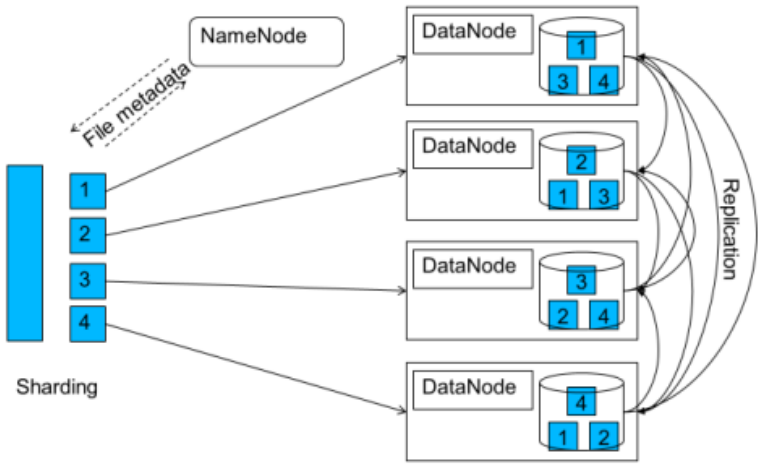
## Different optimized operations on different ‘nodes’

### NameNode

- Determines the mapping of ‘pure data blocks’ to DataNodes (→ metadata)
- Executes file system namespace operations
- E.g. opening, closing, and renaming files and directories

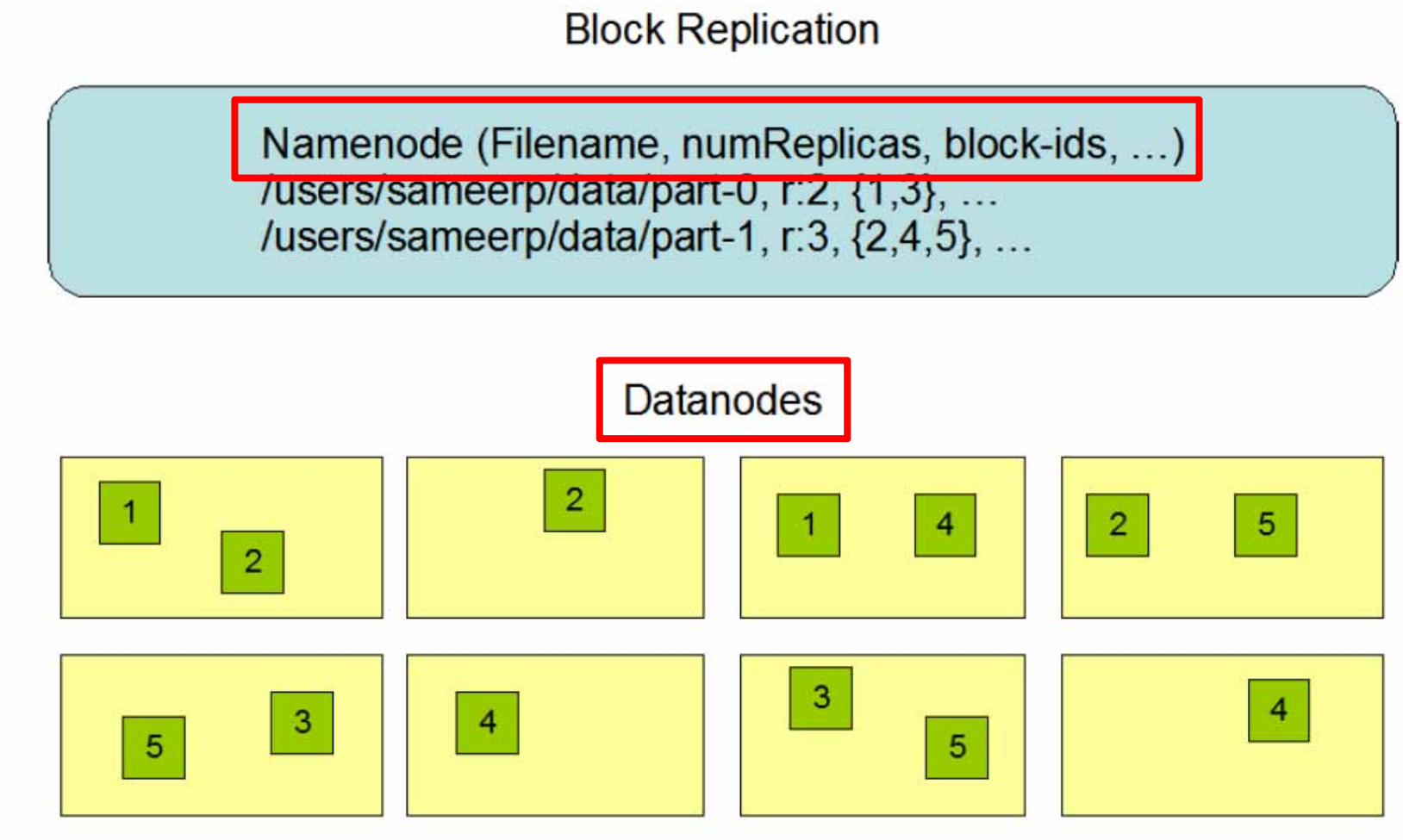
### DataNode

- Serving ‘read and write requests’ from HDFS filesystem clients
- Performs block creation, deletion, and replication (upon instruction from the NameNode)




***[10] Virtual Workshop***

# HDFS File/Block(s) Distribution Example




***[9] The Hadoop Distributed File System***

# Working with HDFS

## HDFS can be deployed in conjunction with another filesystem

- But HDFS files are not directly accessible via the 'normal file system'

## 'Normal User' Commands

- Create a directory named /foodir
  e.g. bin/hadoop dfs -mkdir /foodir
- View the contents of a file named /foodir/myfile.txt
  e.g. bin/hadoop dfs –cat /foodir/myfile.txt

## Administrator Commands

- Generate a list of DataNodes
  e.g. bin/hadoop dfsadmin -report
- Decommission DataNode datanodename (e.g. maintenance/check reasons)
  e.g. bin/hadoop dfsadmin –decommission datanodename

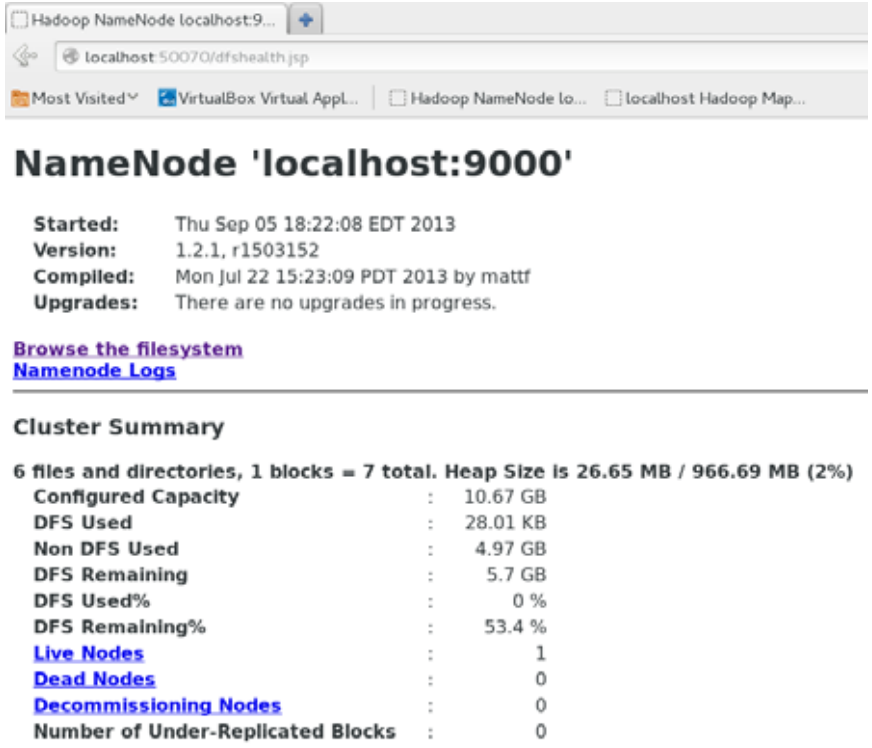
***[9] The Hadoop Distributed File System***

# HDFS Selected Administration & Configuration (1)

## Release Download

- Well maintained, often new versions
- Add/remove nodes dynamically
- Namenode

→ Check Webpage

# HDFS Selected Administration & Configuration (2)

```
morris@login-0-0:~/bin/hadoop-1.2.1/conf
File  Edit  View  Search  Terminal  Help
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>fs.default.name</name>
  <value>hdfs:login-0-0.local:9000</value>
</property>
</configuration>
~
~
~
~
~
~
~
~
"core-site.xml" 11L, 274C
```

**conf/core-site.xml**

```
morris@login-0-0:~/bin/hadoop-1.2.1/conf
File  Edit  View  Search  Terminal  Help
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>
~
~
~
~
~
~
~
~
"hdfs-site.xml" 11L, 246C
```

**conf/hdfs-site.xml**

# HDFS Usage Examples

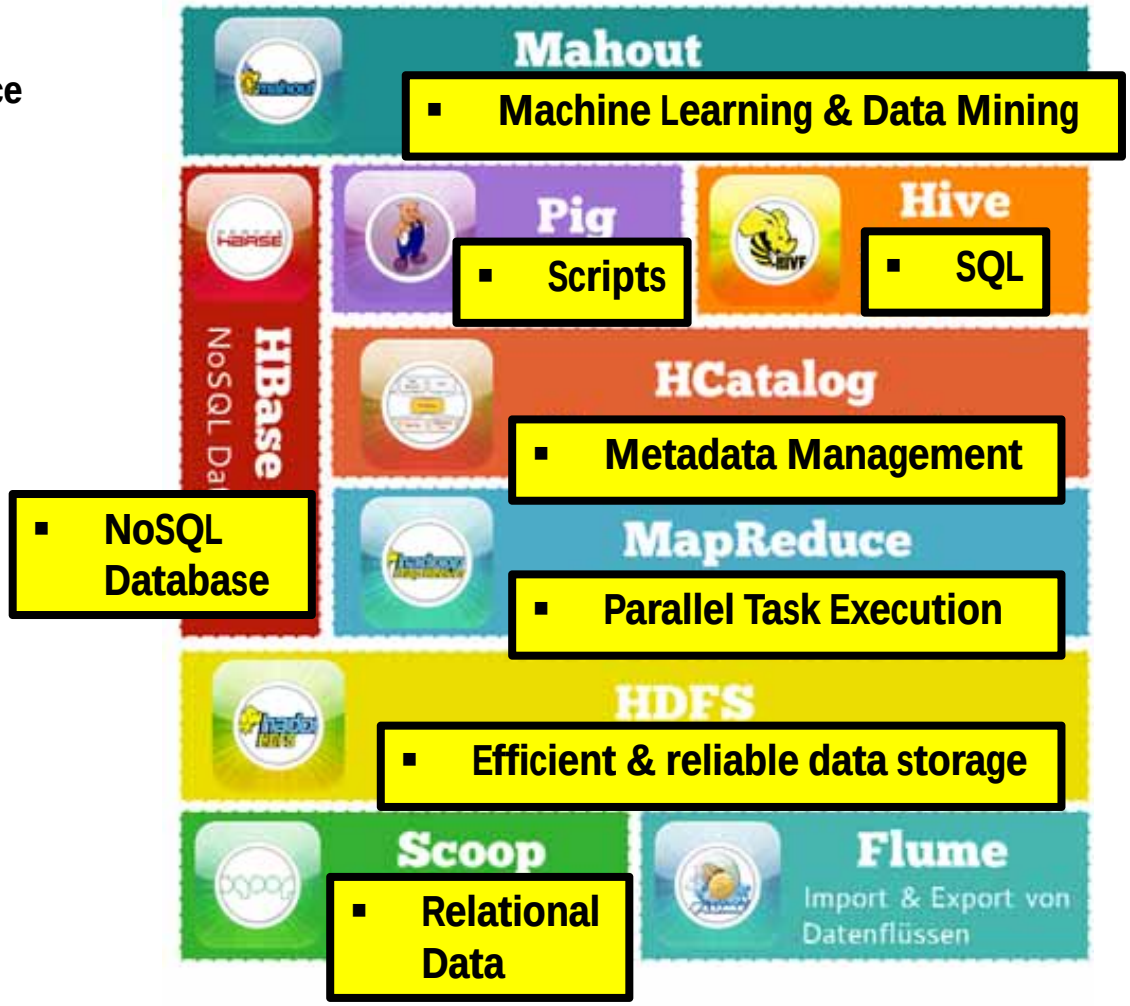
```
./bin/hadoop dfs -copyFromLocal ~/BIGDATA/DaVinciExample DaVinciExample

[morris@login-0-0 hadoop-1.2.1]$ vi dfsworkminingcourse.sh
[morris@login-0-0 hadoop-1.2.1]$ ls ~/BIGDATA/DaVinciExample
eclipseoutput  part-r-00000  pg5000.txt
[morris@login-0-0 hadoop-1.2.1]$ ./dfsworkminingcourse.sh
[morris@login-0-0 hadoop-1.2.1]$ ./bin/hadoop dfs -ls
Found 1 items
drwxr-xr-x   - morris supergroup          0 2013-09-06 10:53 /user/morris/DaVinciExample
```

```
linux-8djg:/home/adminuser/bin/hadoop-1.2.1 # ./bin/hadoop namenode -format
13/09/05 18:15:00 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = linux-8djg.site/127.0.0.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 1.2.1
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/branch-1.2 -r 1503152; compiled by 'mattf' on Mon Jul 22 15:23:09 PDT 2013
STARTUP_MSG:   java = 1.7.0_15
************************************************************/
Re-format filesystem in /tmp/hadoop-root/dfs/name ? (Y or N) Y
13/09/05 18:15:04 INFO util.GSet: Computing capacity for map BlocksMap
13/09/05 18:15:04 INFO util.GSet: VM type       = 64-bit
13/09/05 18:15:04 INFO util.GSet: 2.0% max memory = 1013645312
13/09/05 18:15:04 INFO util.GSet: capacity      = 2^21 = 2097152 entries
13/09/05 18:15:04 INFO util.GSet: recommended=2097152, actual=2097152
13/09/05 18:15:05 INFO namenode.FSNamesystem: fsOwner=root
13/09/05 18:15:05 INFO namenode.FSNamesystem: supergroup=supergroup
13/09/05 18:15:05 INFO namenode.FSNamesystem: isPermissionEnabled=true
13/09/05 18:15:05 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
13/09/05 18:15:05 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessKeyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
13/09/05 18:15:05 INFO namenode.FSEditLog: dfs.namenode.edits.toleration.length = 0
13/09/05 18:15:05 INFO namenode.NameNode: Caching file names occuring more than 10 times
13/09/05 18:15:05 INFO common.Storage: Image file /tmp/hadoop-root/dfs/name/current/fsimage of size 110 bytes saved in 0 seconds.
13/09/05 18:15:05 INFO namenode.FSEditLog: closing edit log: position=4, editlog=/tmp/hadoop-root/dfs/name/current/edits
13/09/05 18:15:05 INFO namenode.FSEditLog: close success: truncate to 4, editlog=/tmp/hadoop-root/dfs/name/current/edits
13/09/05 18:15:05 INFO common.Storage: Storage directory /tmp/hadoop-root/dfs/name has been successfully formatted.
13/09/05 18:15:05 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at linux-8djg.site/127.0.0.1
************************************************************/
```

# HDFS & Hadoop Map-Reduce Ecosystem



**Modified from [26] Map-Reduce**




**What your users Want to do with map-reduce?**

- **Administration & deployment experience – hard to get versions in sync and maintain updates**

# Lessons Learned – HDFS

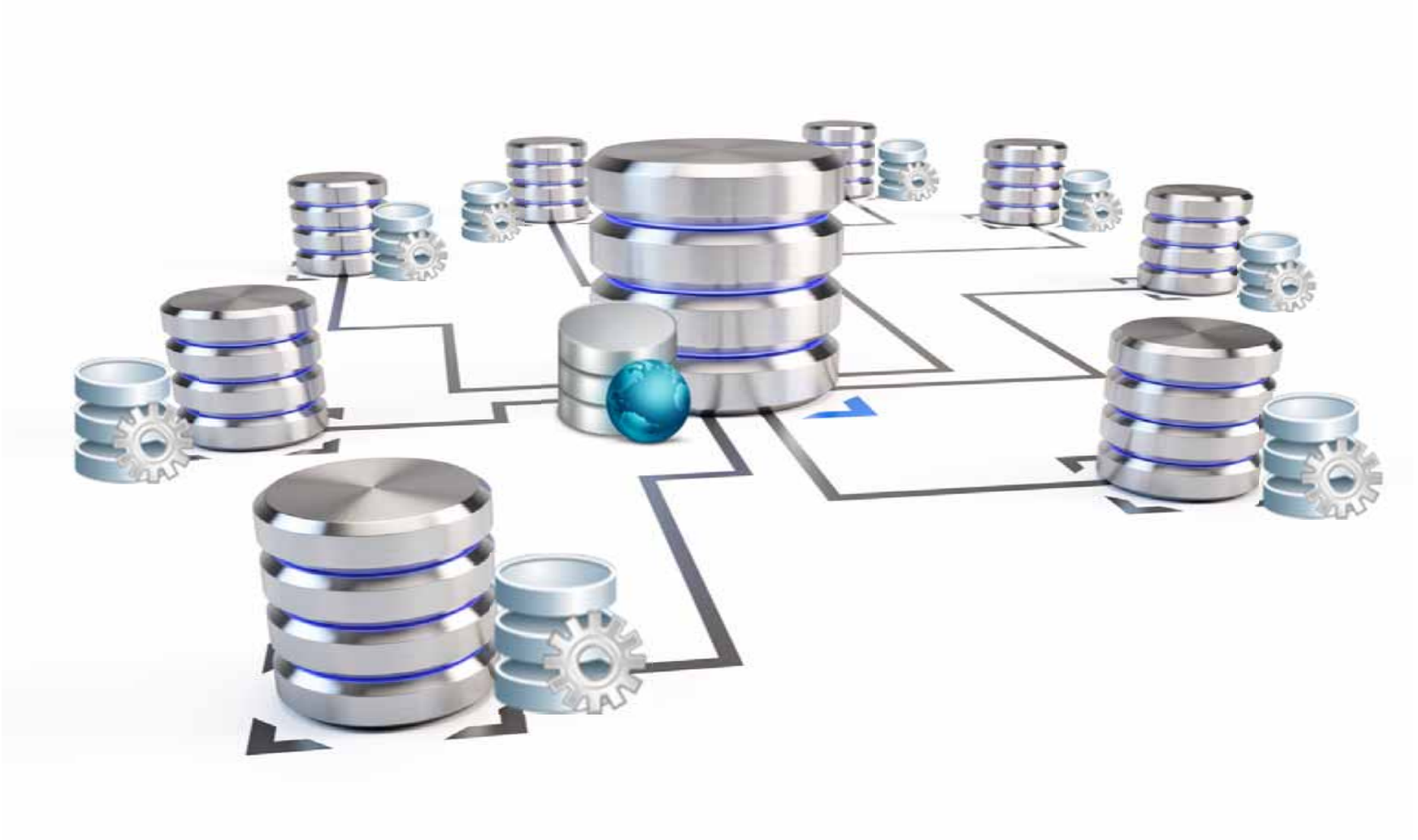
## Be careful of investing time & efforts

- Frameworks keep significantly fast changing, no peer-reviews as in HPC (e.g. HDFS is part of the changing Hadoop series)
- Large-scale deployments of HDFS rather in business than in scientific production (e.g. 2-3 times replication of scientific data on active disks usually not affordable)
- Consider and observe developments around Apache Spark

## Comparisons with HPC Centers & usual parallel filesystems

- Deployments together with parallel filesystems are not straightforward (e.g. IBM works still on Watson and GPFS integration for its Hadoop stacks)
- HPC centers with parallel filesystems often have large backend storage too (e.g. single site vs. map-reduce HDFS idea of different sites)
- Parallel I/O and high-level libraries like HDF5 or pNetCDF are very scalable (e.g. tool support and integration straightforward, de-facto-standards, etc.)

# Part Two – Questions



# Part Two – Outline

## Part One ‘Big Data‘ Challenges & HPC Tools

- Understanding ‘Big Data‘ in Science & Engineering
- Statistical Data Mining and Learning from ‘Big Data‘
- OpenMP/MPI Tool Example for Clustering ‘Big Data‘
- MPI Tool Example for Classification of ‘Big Data‘

*coffee break*

## Part Two ‘Big Data‘ & Distributed Computing Tools

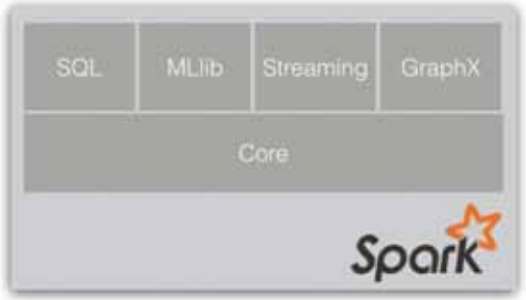
- Exploring Parallel & Distributed Computing Approaches
- Examples of Map-Reduce & ‘Big Data‘ Processing with Hadoop
- Tools for handling ‘Big Data‘ storage & replication methods
- Technologies for Large-scale distributed ‘Big Data‘ Management

# Emerging Spark Platform

- **Apache Spark offers scalable machine learning and data mining algorithms**
- **Supports Scala, Java and Python and runs in parallel using map-reduce**

## Selected Facts

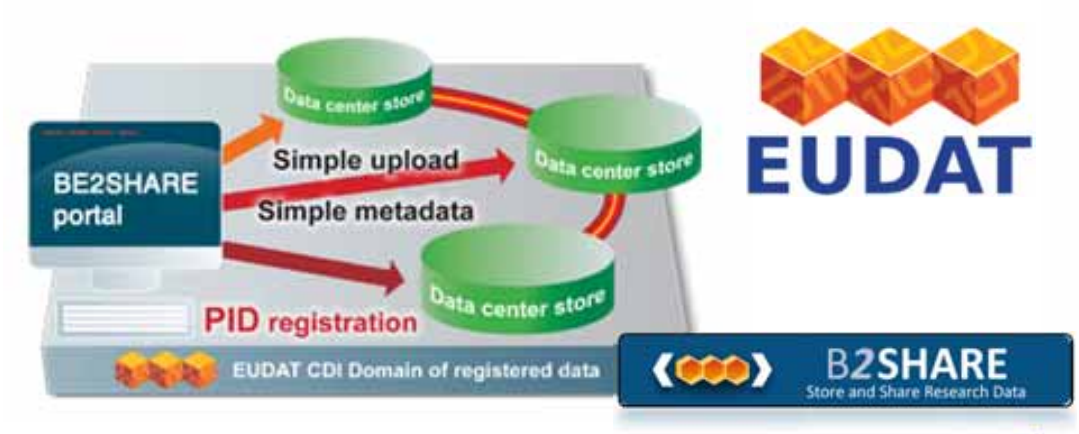
- Machine learning library (in development, v.0.9)
- Offers flexible execution platform that can take advantage of Apache Hadoop using map&reduce


***[20] Apache Spark Webpage***

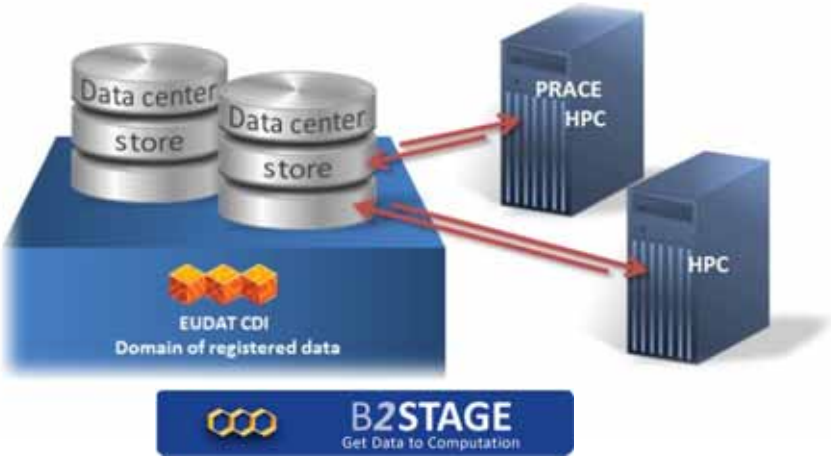
- Basic statistics for summary statistics
- Classification and regression with linear models with Support Vector Machines, logistic regression, linear regresssion
- Classification with decision trees and naive Bayes
- Collaborative filtering using alternating least squares
- Clustering using K-Means
- Dimensionality reduction with Principle Component Analysis (PCA) and Singular Value Decomposition (SVD)

# Tools for Large-scale Distributed Data Management

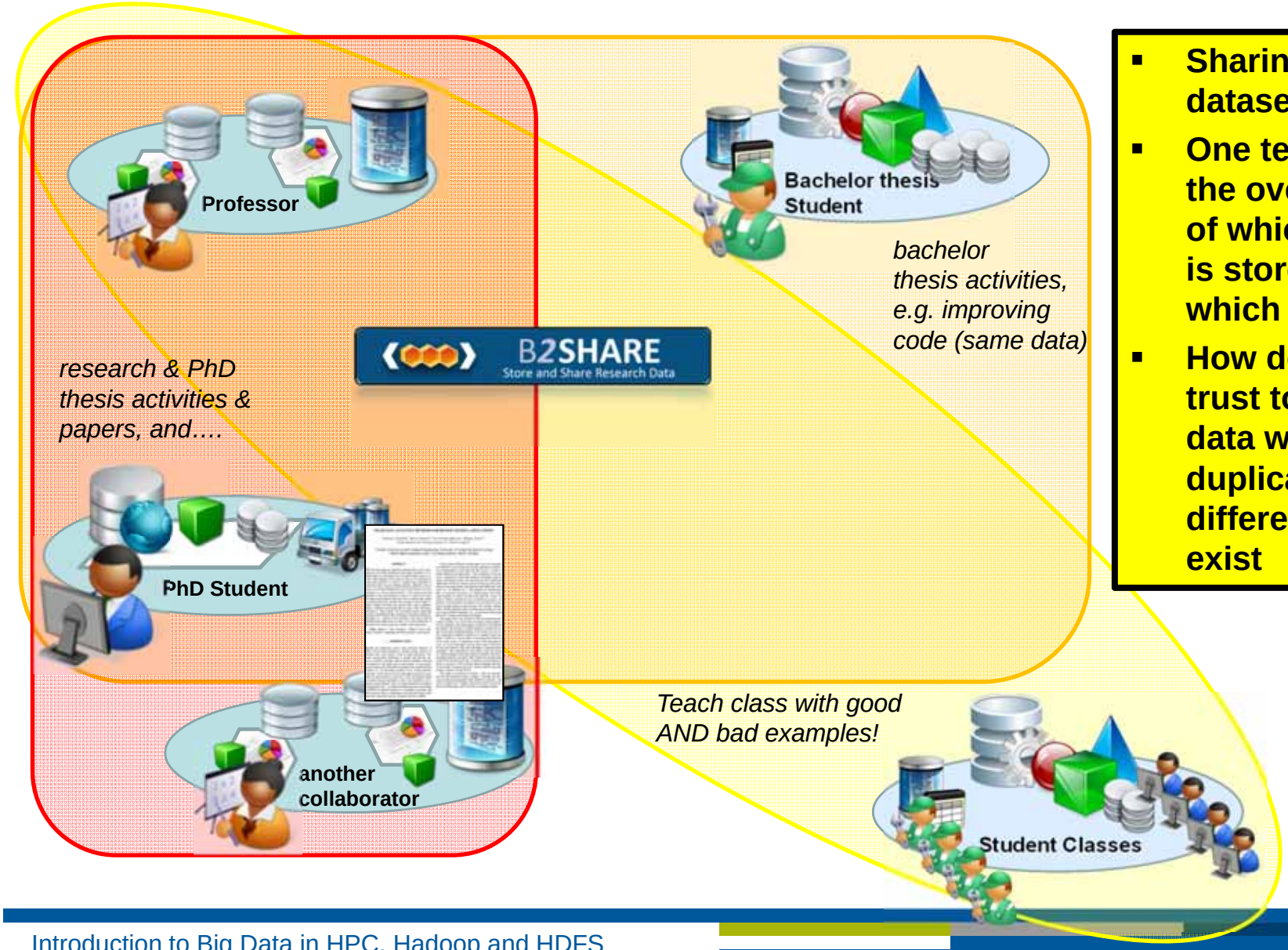



*[25] M. Riedel & P. Wittenburg et al.*

- **Useful tools for data-driven scientists & HPC users**

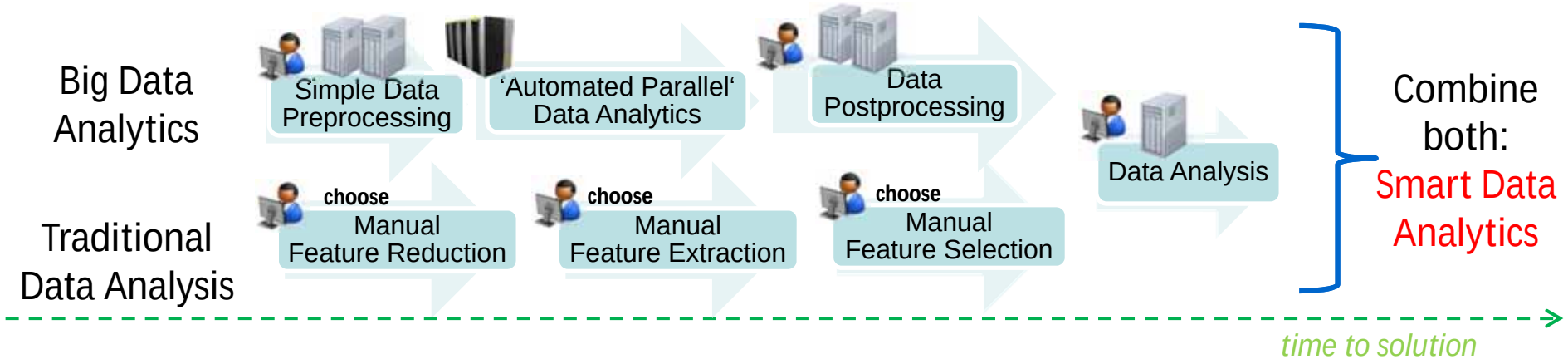



# Need for Sharing & Reproducability in HPC – Example

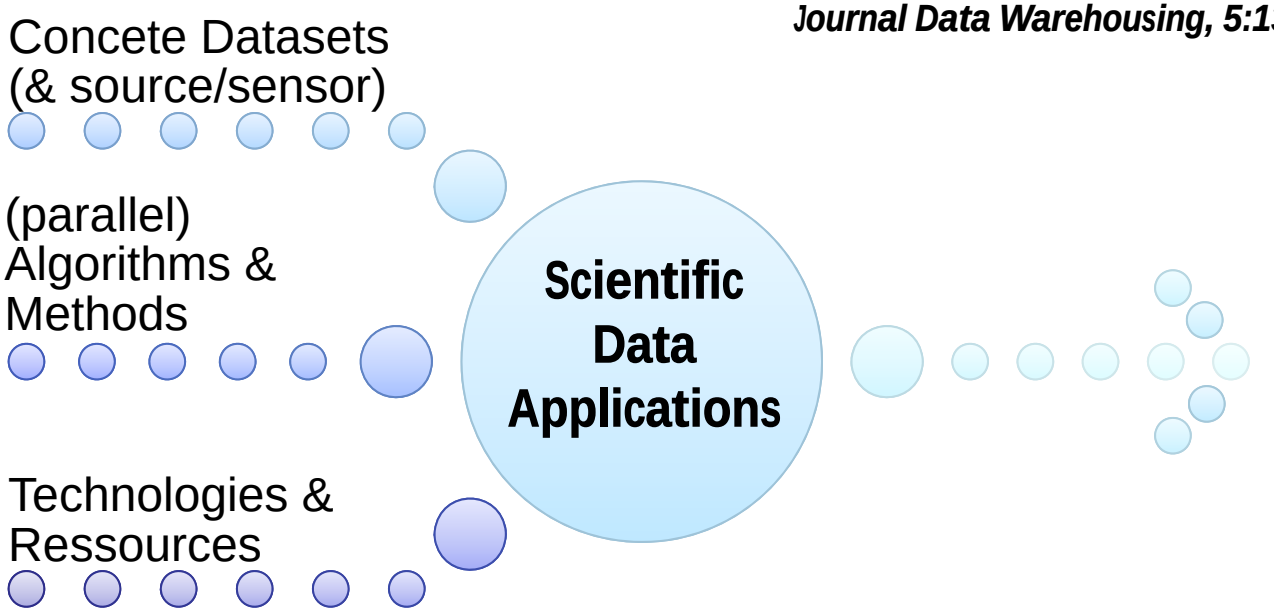


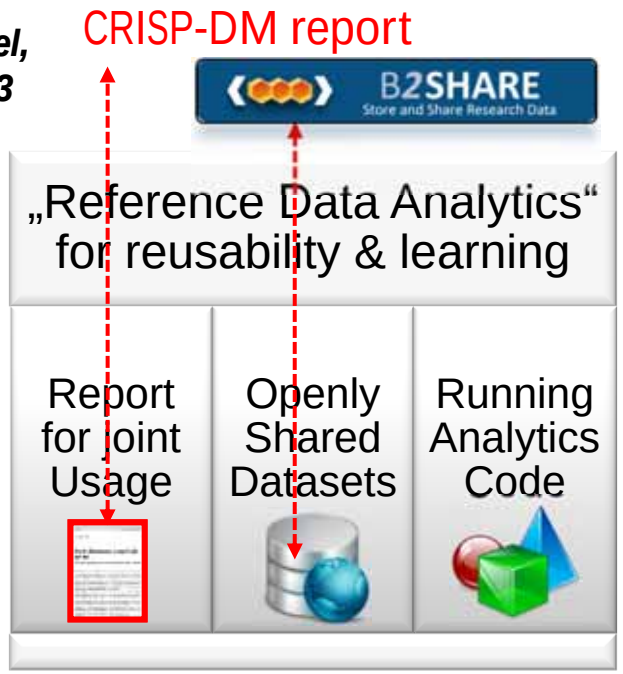

- **Sharing different datasets is key**
- **One tend to loose the overview of which data is stored on which platform**
- **How do we gain trust to delete data when duplicates on different systems exist**

# Smart Data Analytics Process

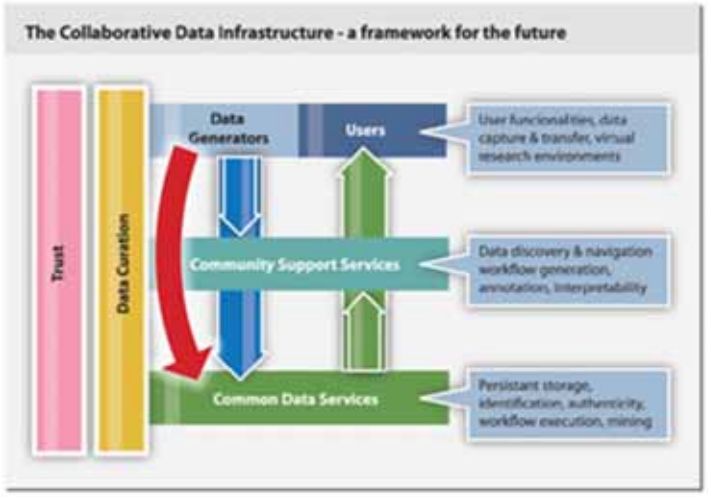


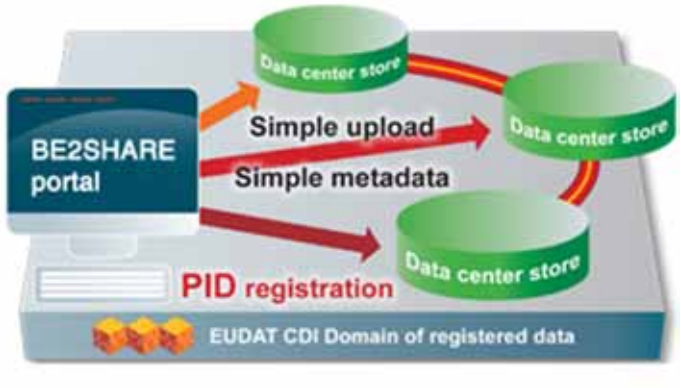

***[24] C. Shearer, CRISP-DM model, Journal Data Warehousing, 5:13***



# Reproducability Example in Data-driven Science (1)

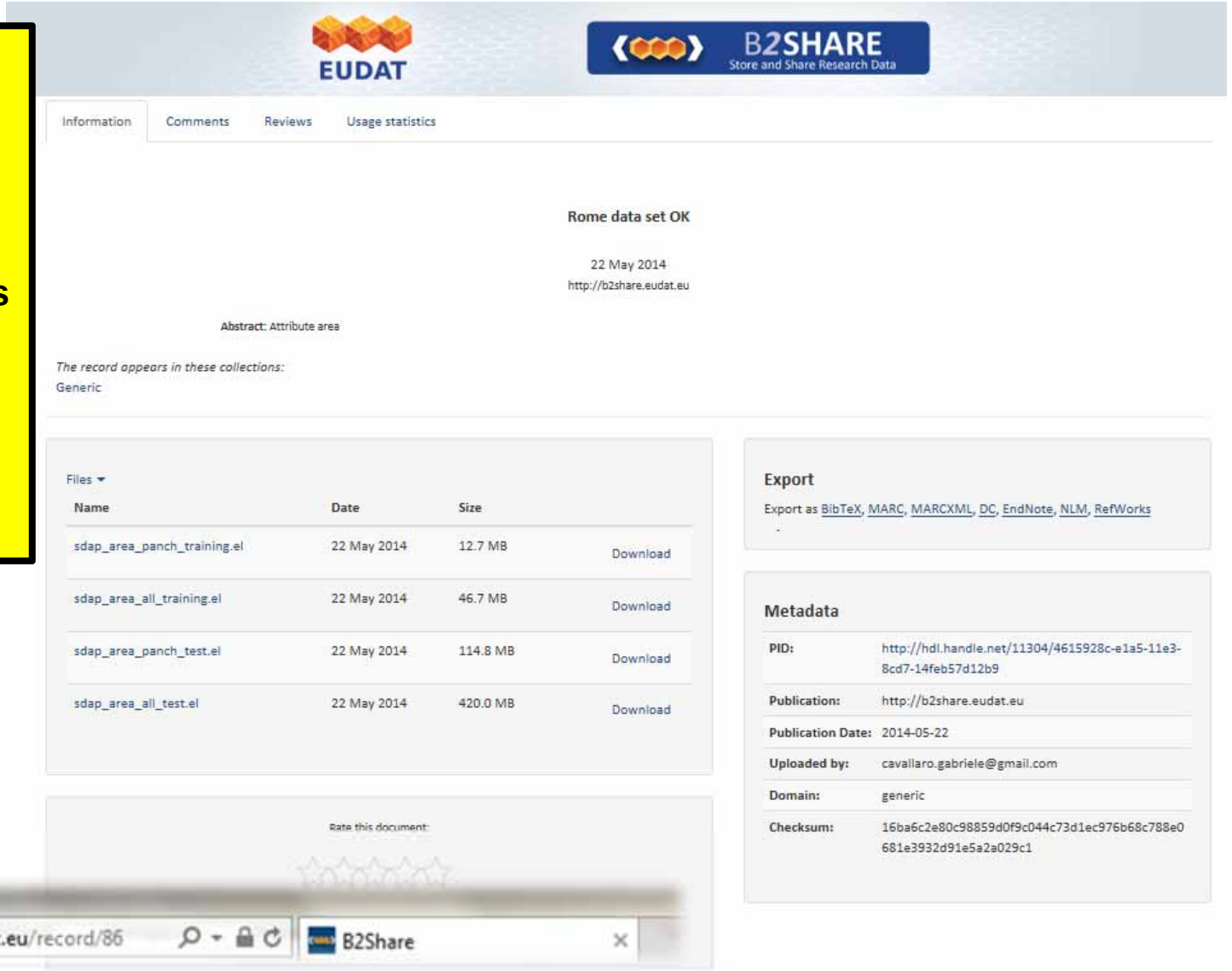



- **Having this tool available on the Web helps tremendously to save time for no research tasks**
- **Using the tool enables to focus better on the research tasks**

# Reproducability Example in Data-driven Science (2)

- **Sharing pre-processed data**
- **LibSVM format**
- **Training and Testing Datasets**
- **Different setups for analysis (SDAP on All or SDAP on Panchromatic)**



# Reproducability Example in Data-driven Science (3)

## Simple download from http using the wget command

```
mriedel@judge:~/bigdata> ls -al
total 640
drwxrwxrwx 21 mriedel  zam  32768 2014-09-17 22:20 .
drwxr-xr-x 19 mriedel  zam  32768 2014-09-18 11:49 ..
drwxr-xr-x  2 mriedel  zam  32768 2014-06-19 07:17 102-salinasindian
drwxr-xr-x  2 mriedel  zam    512 2014-06-19 20:11 107-salinasrescaled
drwxr-xr-x  2 mriedel  zam    512 2014-07-08 17:14 111-romemultispectral
drwxr-xr-x  2 mriedel  zam    512 2014-07-10 11:46 112-romeoriginalbands
drwxr-xr-x  2 mriedel  zam    512 2014-09-17 22:31 120-indianpine
drwxr-xr-x  2 mriedel  zam    512 2014-09-17 22:14 121-salinas
drwxr-xr-x  2 mriedel  zam    512 2014-09-17 22:19 122-salinas2
drwxr-xr-x  2 mriedel  zam    512 2014-09-17 22:24 123-indianpine2
drwxr-xr-x  2 mriedel  zam    512 2014-07-09 11:03 86-romeok
drwxr-xr-x  2 mriedel  zam  32768 2014-06-10 18:51 bigindianpines
            mriedel    zam  32768 2014-05-28 10:59 indian
            mriedel    zam  32768 2014-06-10 20:48 indianpinesreduced
            mriedel    zam    512 2014-07-28 17:53 mnist-576-rbf
            skoehnen   inm1   512 2014-07-29 16:35 pli-blockface
            mriedel    zam  32768 2014-06-25 11:09 rome-ok
            mriedel    zam    512 2014-07-08 13:29 rome-ok-copy
            mriedel    zam  32768 2014-06-03 14:24 salinas
            mriedel    zam  32768 2014-06-16 16:50 salinasindianrev
            mriedel    zam  32768 2014-06-10 15:47 salinas-new
```

*…other open B2SHARE datasets*

*…before adopting B2SHARE regularly*

- **Simple Download from http using wget**
- **Well defined directory structures**

# Reproducability Example in Data-driven Science (4)

## Make a short note in your directory linking back to B2SHARE

```
mriedel@judge:~/bigdata> cd 86-romeok/
mriedel@judge:~/bigdata/86-romeok> ls -al
total 580320
drwxr-xr-x  2 mriedel zam       512 2014-07-09 11:03 .
drwxrwxrwx 21 mriedel zam     32768 2014-09-17 22:20 ..
-rw-r--r--  1 mriedel zam        35 2014-07-09 11:01 b2share.txt
-rw-r--r--  1 mriedel zam 419974873 2014-05-22 13:36 sdap_area_all_test.el
-rw-r--r--  1 mriedel zam  46652874 2014-05-22 13:36 sdap_area_all_training.el
-rw-r--r--  1 mriedel zam 114763982 2014-05-22 13:36 sdap_area_panch_test.el
-rw-r--r--  1 mriedel zam  12745692 2014-05-22 13:36 sdap_area_panch_training.el
mriedel@judge:~/bigdata/86-romeok> more b2share.txt
https://b2share.eudat.eu/record/86
mriedel@judge:~/bigdata/86-romeok>
```

- **Enables the trust to delete data if necessary (working against big data)**
- **Link back to B2SHARE for quick checks and file that links back fosters trust**

# Reproducability Example in Data-driven Science (5)

```
mriedel@judge:~> ls -al
total 111840
drwxr-xr-x  19 mriedel zam     32768 2014-09-18 11:49 .
drwxr-xr-x 214 root    sys     32768 2014-09-12 09:02 ..
-rw-r--r--   1 mriedel zam 113233920 2014-08-08 10:35 115-RunsMatthiasStable.tar
drwxr-xr-x   3 mriedel zam     32768 2014-06-03 16:24 ann-0.1
drwxr-xr-x   3 mriedel zam     32768 2014-06-03 17:02 ann-0.2
drwxr-xr-x   3 mriedel zam     32768 2014-06-04 14:42 ann-0.3
drwxr-xr-x   2 mriedel zam     32768 2014-06-16 19:12 ann-0.4
drwxr-xr-x   2 mriedel zam     32768 2014-06-16 19:24 ann-0.4-orig
drwxr-xr-x   2 mriedel zam     32768 2014-06-19 08:38 ann-0.5
drwxr-xr-x   6 mriedel zam     32768 2014-06-25 00:52 ann-0.6
drwxr-xr-x   4 mriedel zam     32768 2014-06-19 16:31 ann-0.6-scal
drwxr-xr-x   2 mriedel zam     32768 2014-06-24 17:02 ann-0.7
-rw-------   1 mriedel zam      1797 2014-05-12 13:51 .bashrc
drwxrwxrwx  21 mriedel zam     32768 2014-09-17 22:20 bigdata
drwxr-xr-x   3 mriedel zam       512 2014-06-19 09:34 .config
drwxr-xr-x   3 mriedel zam     32768 2014-06-03 14:38 .emacs.d
-rw-------   1 mriedel zam      1864 2014-05-12 13:51 .kshrc
drwxr-xr-x   3 mriedel zam     32768 2014-05-12 14:56 pisvm-1.2
drwxr-xr-x   5 mriedel zam     32768 2014-09-18 11:49 pisvm-1.2.1
drwxr-xr-x   3 mriedel zam       512 2014-07-09 14:51 pisvm-1.2-refactored
-rw-------   1 mriedel zam      2686 2014-05-12 13:51 .profile
-rw-------   1 mriedel zam     22490 2014-09-18 11:51 .sh_history
drwx------   2 mriedel zam     32768 2014-05-12 14:38 .ssh
drwxr-xr-x   2 mriedel zam     32768 2014-05-12 14:39 transfers
-rw-------   1 mriedel zam     19526 2014-09-18
-rw-------   1 mriedel zam       204 2014-09-17
mriedel@judge:~>
```

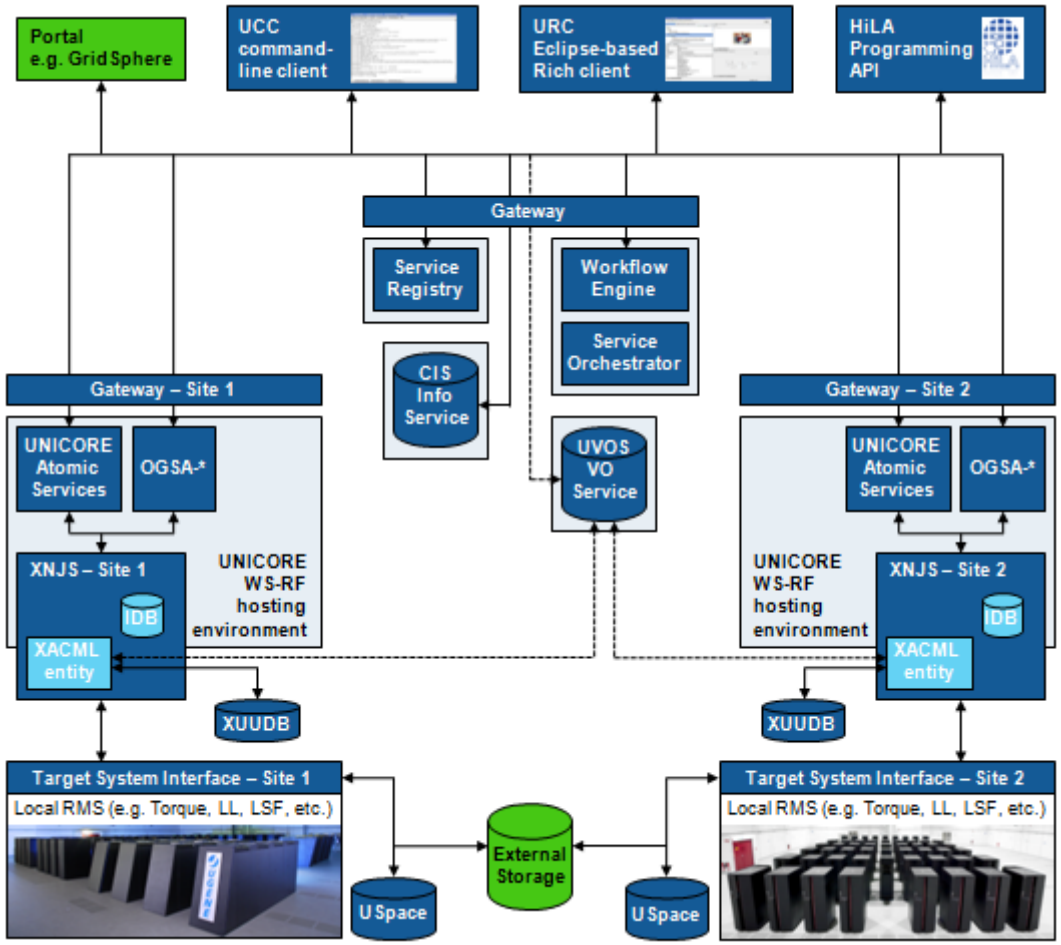
*… a bachelor project*

*… different versions of a parallel neural network code (another classification technique)*

*… different versions of a parallel support vector machine code*

- **True reproducability needs: (1) datasets; (2) technique parameters (here for SVM); and (3) correct versions of algorithm code**

# Distributed Large-scale Data Management & Execution




UNICORE

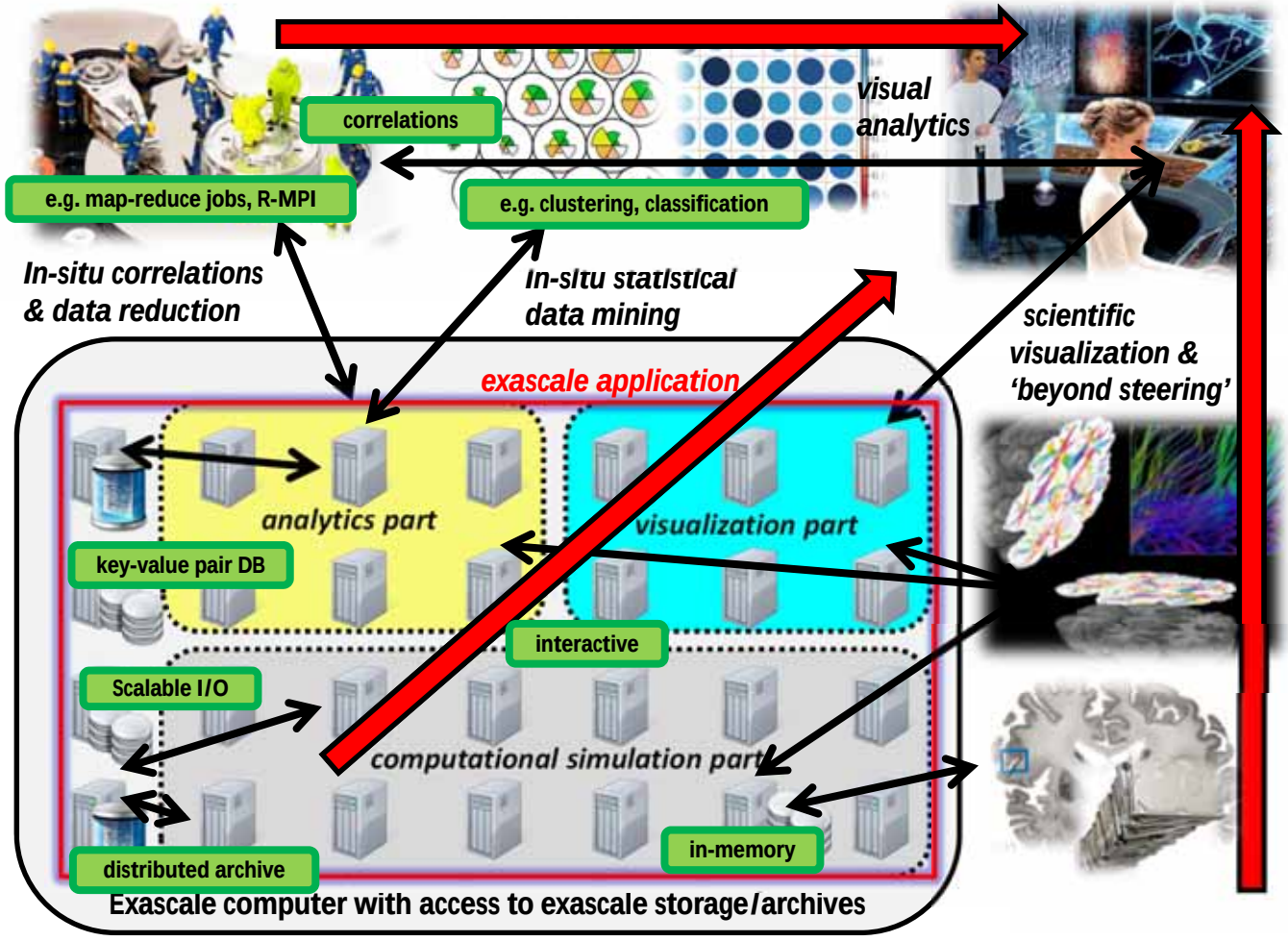
*[21] UNICORE.eu*

PRACE

XSEDE

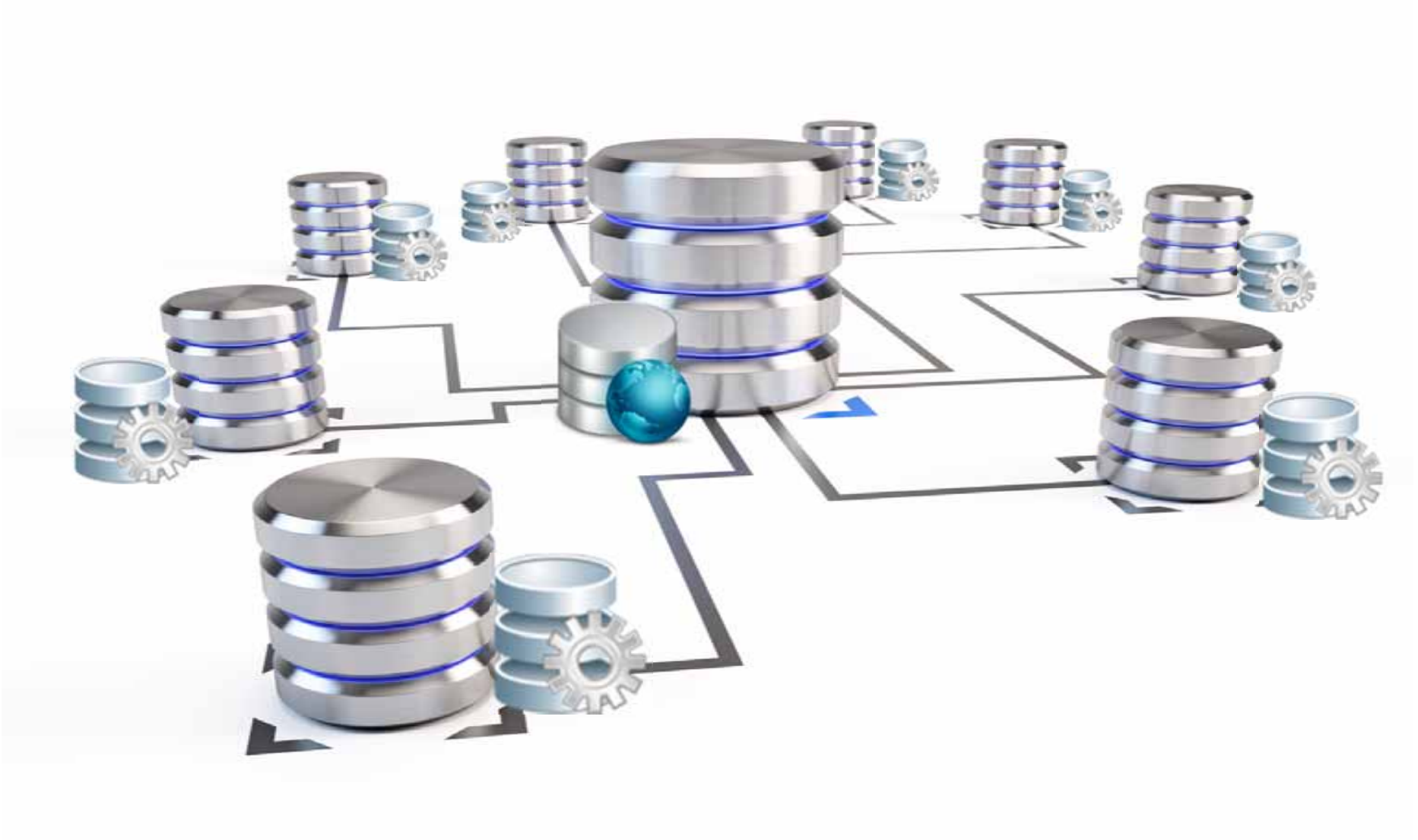
Extreme Science and Engineering Discovery Environment

# In-Situ Analytics for HPC & Exascale

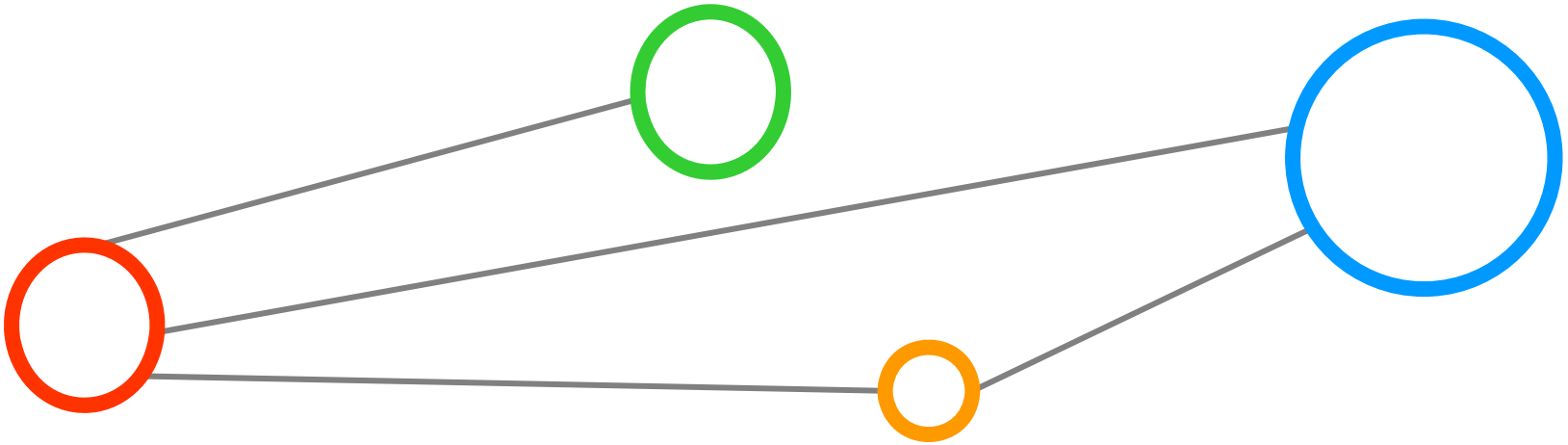



*[21] Inspired by ASCAC DOE report*

# Part Two – Questions



# References



# References (1)


[1] Jeremy Ginsburg et al., 'Detecting influenza epidemics using search engine query data', Nature 457, 2009

[2] David Lazer, Ryan Kennedy, Gary King, and Alessandro Vespignani, 'The Parable of Google Flu: Traps in Big Data Analysis', Science Vol (343), 2014

[3] Science Progress, Online: http://scienceprogress.org/2009/11/dna-confidential/

[4] Introduction to High Performance Computing for Scientists and Engineers, Georg Hager & Gerhard Wellein, Chapman & Hall/CRC Computational Science, ISBN 143981192X

[5] Introduction to Parallel Computing Tutorial, Online: https://computing.llnl.gov/tutorials/parallel_comp/

[6] MapReduce: Simplified Dataset on Large Clusters, J. Dean and S. Ghemawat, 6th Symposium on Operating Systems Design and Implementation, Online: https://www.usenix.org/legacy/event/osdi04/tech/full_papers/dean/dean.pdf

[7] Apache Hadoop, Online: http://hadoop.apache.org/

[8] Mining of Massive Datasets, Online: http://infolab.stanford.edu/~ullman/mmds/book.pdf

[9] Dhruba Borthakur, 'The Hadoop Distributed File System: Architecture and Design', Online: http://hadoop.apache.org/docs/r0.18.0/hdfs_design.pdf

[10] Stampede Virtual Workshop, Online: http://www.cac.cornell.edu/Ranger/MapReduce/dfs.aspx

[11] Hadoop Release 2.2.0, Online: http://hadoop.apache.org/docs/r2.2.0/

[12] 'Study on Parallel SVM Based on MapReduce', Zhanquan Sun and Geoffrey Fox, Online: http://grids.ucs.indiana.edu/ptliupages/publications/Study%20on%20Parallel%20SVM%20Based%20on%20MapReduce.pdf

# References (2)


[13] Twister – Iterative Map-Reduce, Online: http://www.iterativemapreduce.org/

[14] Brandyn White et al., 'Web-Scale Computer Vision using MapReduce for Multimedia Data Mining', Online: http://dl.acm.org/citation.cfm?doid=1814245.1814254

[15] 'Study on Parallel SVM Based on MapReduce', Zhanquan Sun and Geoffrey Fox, Online: http://grids.ucs.indiana.edu/ptliupages/publications/Study%20on%20Parallel%20SVM%20Based%20on%20MapReduce.pdf

[16] Hadoop Distributed Cache, Online: https://hadoop.apache.org/docs/r1.0.4/api/org/apache/hadoop/filecache/DistributedCache.html

[17] FutureGrid/FutureSystems UoIceland Teaching Project, online: https://portal.futuregrid.org/projects/358

[18] FutureGrid/FutureSystems WordCount Example, online: https://portal.futuregrid.org/manual/hadoop-wordcount

[19] Smart Data Innovation Lab (SDIL), online: http://www.sdil.de/de/

[20] Apache Spark MLlib Webpage, online: https://spark.apache.org/mllib/

[21] DoE ASCAC Report, 2013

[22] UNICORE middleware, online: http://www.unicore.eu/

[23] Study on Hadoop, Accenture

[24] Shearer C., 'The CRISP-DM model: the new blueprint for data mining', J Data Warehousing (2000); 5:13—22.

[25] M. Riedel and P. Wittenburg et al. 'A Data Infrastructure Reference Model with Applications: Towards Realization of a ScienceTube Vision with a Data Replication Service', 2013

[26] Einführung in Hadoop (German language, sorry), online: http://blog.codecentric.de/2013/08/einfuhrung-in-hadoop-die-wichtigsten-komponenten-von-hadoop-teil-3-von-5/

[27] HOD Architecture, online: http://www.informit.com/articles/article.aspx?p=2190193&seqNum=4

# Acknowledgements

PhD Student Gabriele Cavallaro, University of Iceland
Tómas Philipp Runarsson, University of Iceland
Kristján Jonasson, University of Iceland

Markus Axer, Stefan Köhnen, Tim Hütz, Institute of Neuroscience & Medicine, Juelich

**Selected Members of the Research Group on High Productivity Data Processing**

**Ahmed Shiraz Memon**
**Mohammad Shahbaz Memon**
**Markus Goetz**
**Christian Bodenstein**
**Philipp Glock**
**Matthias Richerzhagen**

# Thanks


*Slides available at http://www.morrisriedel.de/talks*