2012 London Summer Games

BIG DATA BY THE NUMBERS

The Age of Big Data Is Dawning

For two weeks this summer, when the world comes together for the 2012 Summer Games, an unprecedented spike in the sheer volume of big data is expected to be generated on a global scale. Are today's businesses and IT systems ready to support the record-setting amount of big data that the world is on pace to create during the 2012 Summer Games?

60GB
OF INFORMATION PER SECOND
expected to flow across British Telecom's networks
(the equivalent of all of Wikipedia every 5 seconds)

30%
MORE RESULTS DATA
will be processed during the 2012 London Games than during the 2008 Beijing Games

200,000
HOURS
of big data will be generated while testing the IT systems before the Games even start
(the equivalent of 8,333 days of work)

845
MILLION
monthly active Facebook users resulting in an average of
15TB+
of data predicted to be collected each day during the Games

expected to visit the official Website of the 2012 Summer Games

2000
HOURS
of live sports media coverage (covering every single sport each day of the Games)

will be digitally broadcasted to more than...

14,000+
TV and broadcast stations

with...
4B
PEOPLE
worldwide tuning in to watch the opening ceremonies

13,000+
TWEETS PER SECOND
expected to be posted to Twitter during the Summer Games

8.5B
DEVICES

expected to be connected to the Internet in 2012

Keystrokes from many of these devices will add to the amount of big data generated during the Summer Games.

NetApp

Sources:
1. BT London 2012. At A Glance. Retrieved June, 15, 2012, from http://www.btplc.com/BTLondon2012/AtaglanceIndex.htm
2. Kirk, Jeremy (2011, November 25). London Wires Up for 2012 Olympics. IDG News Service/Computerworld UK
3. The London Organising Committee of the Olympic and Paralympic Games (LOCOG). Media Centre. Retrieved June 15, 2012, from www.London2012.com
4. Mains, Richard (2012, March 29.) Game On! London 2012's Technology Hub Ready For Action. London Press Service
5. The Next Web, DAZEINFO
6. Rowady, John (2012, May 1). The Social Olympics: The Effect on London 2012. MediaPost

Large scale processing using Hadoop

Ján Vaňo

What is Hadoop?

- Software platform that lets one easily write and run applications that process vast amounts of data

Includes:

- MapReduce – offline computing engine
- HDFS – Hadoop distributed file system
- …

Brief history

- Created by Doug Cutting
- Named after his son‘s toy - elephant

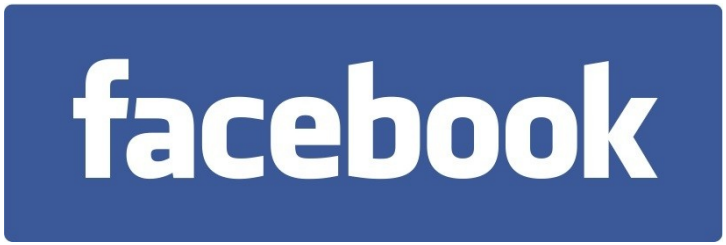
Brief history

- 2002 - Project Nutch started (open source web search engine)
- 2003 - GFS (Google File System) paper published
- 2004 - Implementation of GFS started
- 2004 - Google published MapReduce paper
- 2005 - Working implementations of MapReduce and GFS (NDFS)
- 2006 - System applicable beyond realm of search
- 2006 - Nutch moved to Hadoop project, Doug Cutting joins Yahoo!
- 2008 - Yahoo!s production index generated by 10,000 core Hadoop cluster
- 2008 - Hadoop moved under Apache Foundation
- April 2008 - Hadoop broke world record - fastest sorting of 1 TB of data (209 seconds, previously 297)
- November 2008 - Google's implementation sorted 1 TB in 68 seconds
- May 2009 - Yahoo! team sort 1 TB in 62 seconds

Why Hadoop?

- Scalable: It can reliably store and process petabytes
- Economical: It distributes the data and processing across clusters of commonly available computers (in thousands)
- Efficient: By distributing the data, it can process it in parallel on the nodes where the data is located
- Reliable: It automatically maintains multiple copies of data and automatically redeploys computing tasks based on failures

Who uses Hadoop?

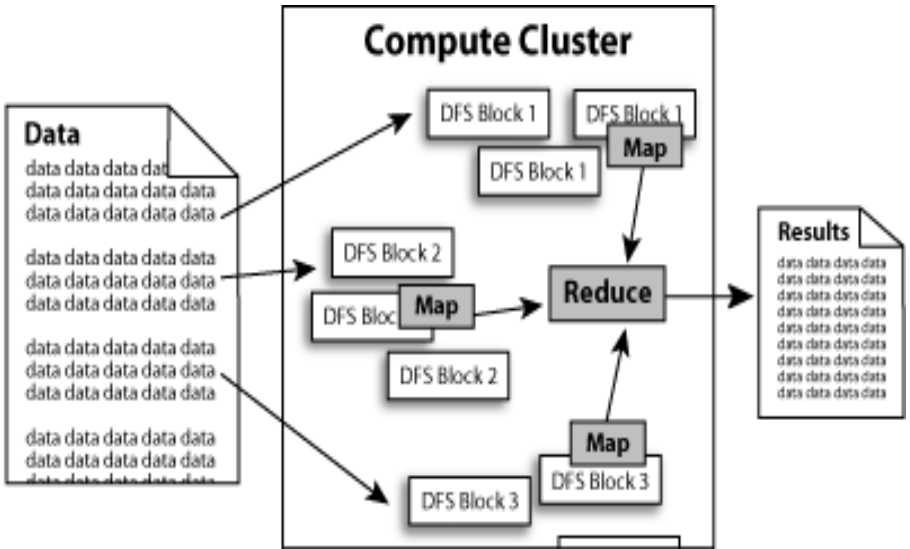


Hadoop modules

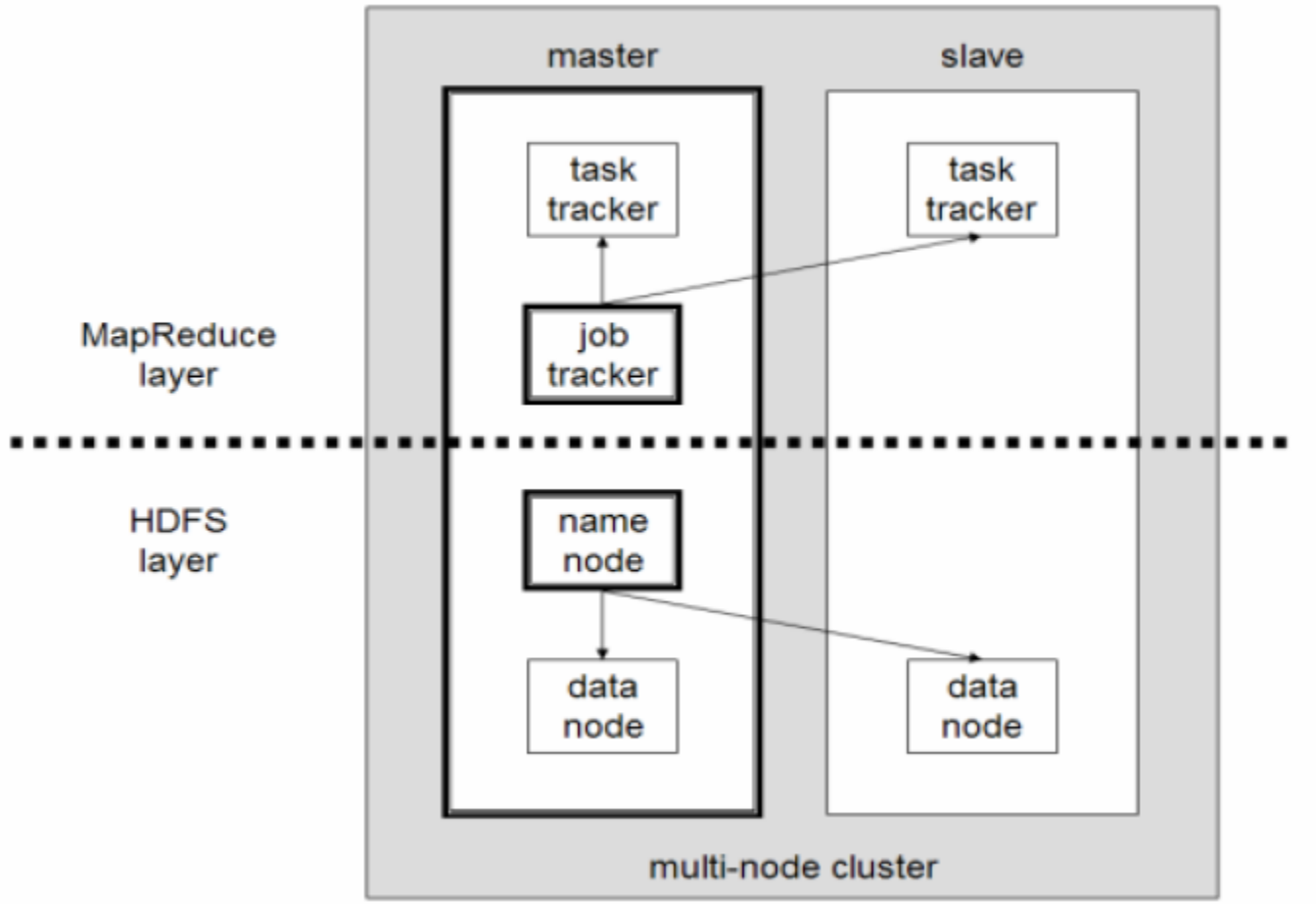
- Hadoop Common - contains libraries and utilities needed by other Hadoop modules
- Hadoop MapReduce - a programming model for large scale data processing
- Hadoop Distributed File System (HDFS) - a distributed file-system that stores data on the commodity machines, providing very high aggregate bandwidth across the cluster
- Hadoop YARN - a resource-management platform responsible for managing compute resources in clusters and using them for scheduling of users' applications

What does it do?

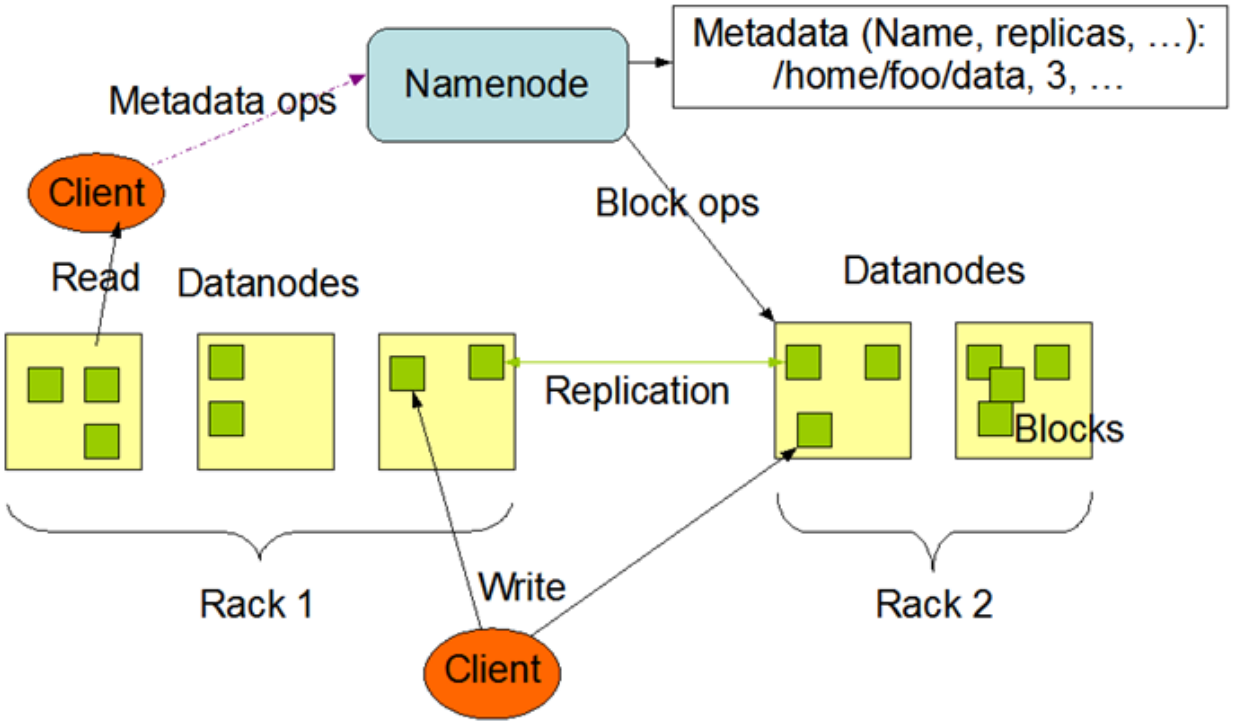
- Hadoop implements Google's MapReduce, using HDFS
- MapReduce divides applications into many small blocks of work
- HDFS creates multiple replicas of data blocks for reliability, placing them on compute nodes around the cluster
- MapReduce can then process the data where it is located
- Hadoop 's target is to run on clusters of the order of 10,000-nodes



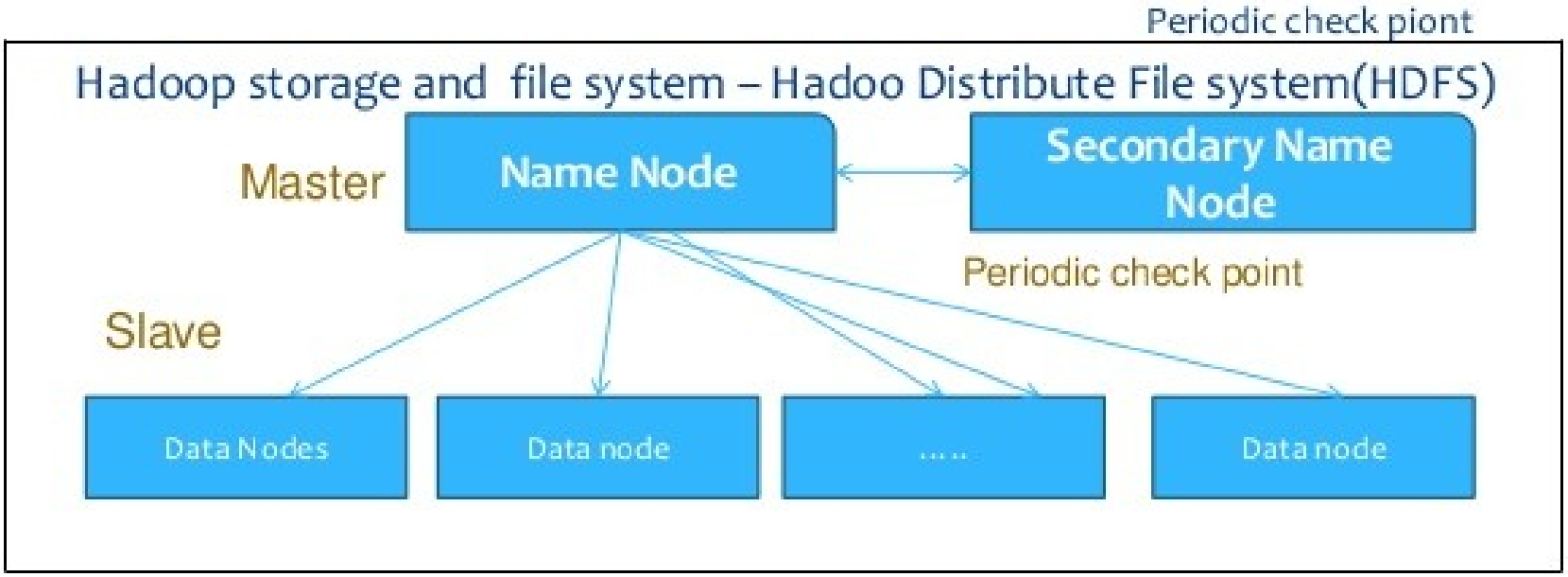
Hadoop architecture



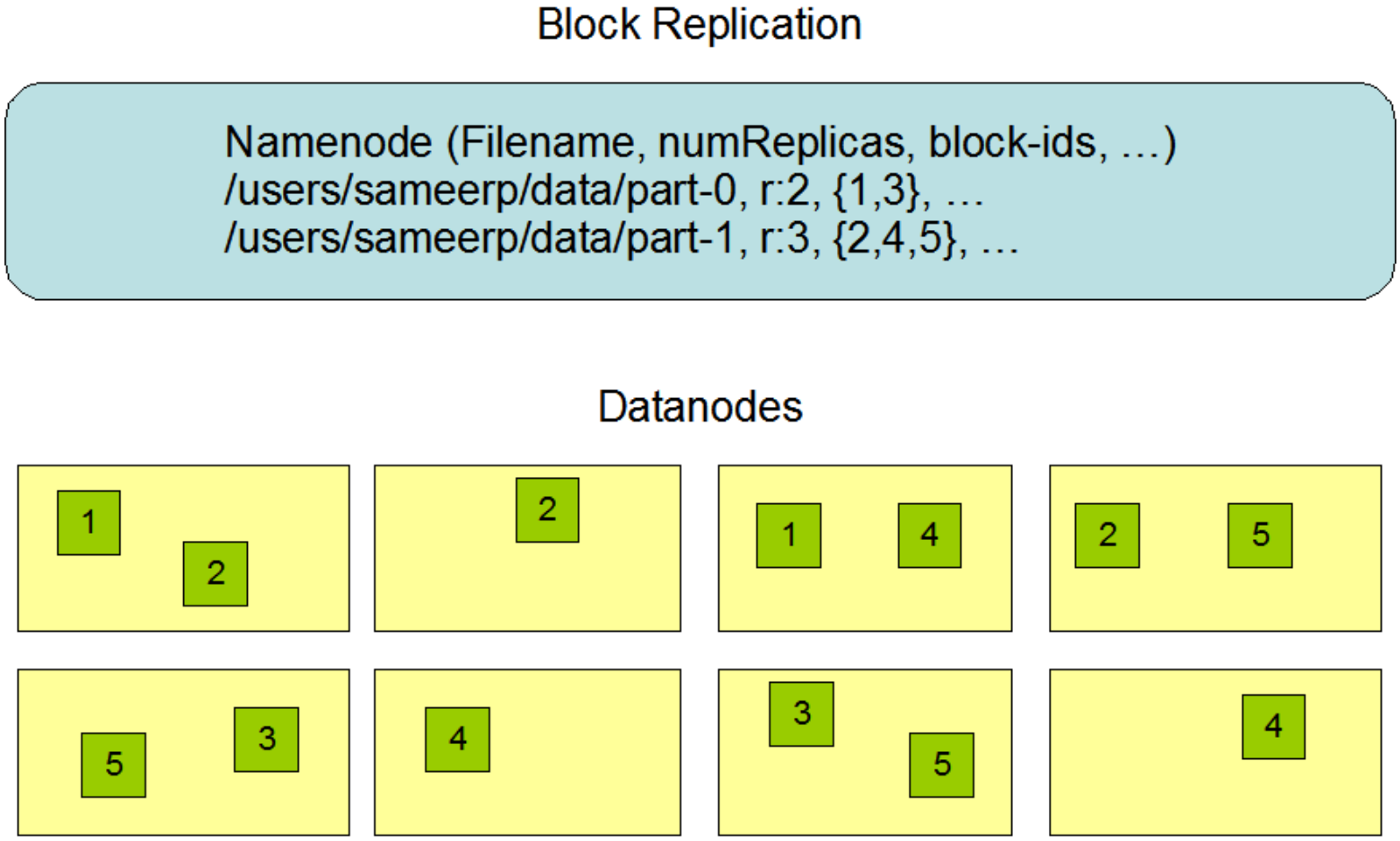
HDFS Architecture



HDFS Architecture



Data replication in HDFS



MapReduce vs. RDBMS

	Traditional RDBMS	MapReduce
Data Size	Gigabytes	Petabytes
Access	Interactive and batch	Batch
Updates	Read and write many times	Write once, read many times
Structure	Static schema	Dynamic schema
Integrity	High	Low
Scaling	Nonlinear	Linear

Data Structure

- Structured Data – data organized into entities that have a defined format.
  - Realm of RDBMS
- Semi-Structured Data – there may be a schema, but often ignored; schema is used as a guide to the structure of the data.
- Unstructured Data – doesn’t have any particular internal structure.
- MapReduce works well with semi-structured and unstructured data.

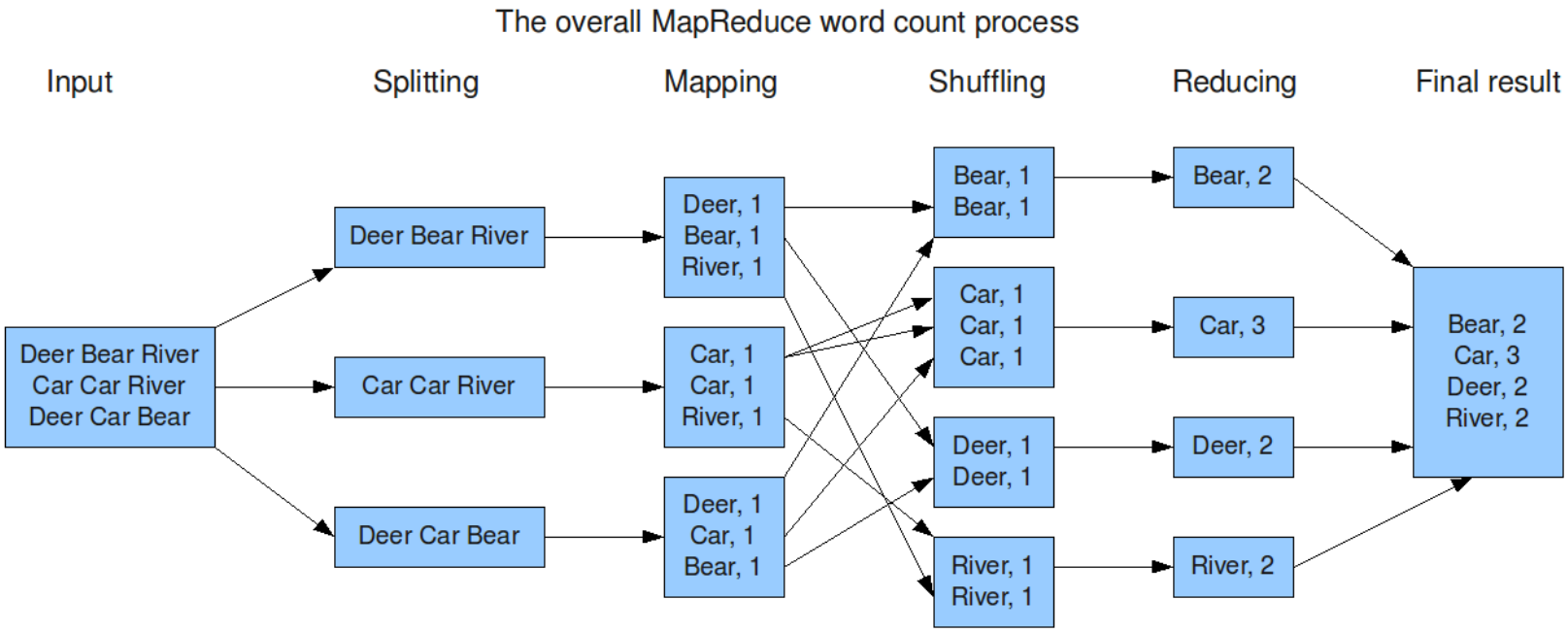
Assumptions

- Hardware will fail
- Processing will be run in batches. Thus there is an emphasis on high throughput as opposed to low latency
- Applications that run on HDFS have large data sets. A typical file in HDFS is gigabytes to terabytes in size
- It should provide high aggregate data bandwidth and scale to hundreds of nodes in a single cluster. It should support tens of millions of files in a single instance
- Applications need a write-once-read-many access model
- Moving Computation is Cheaper than Moving Data
- Portability is important

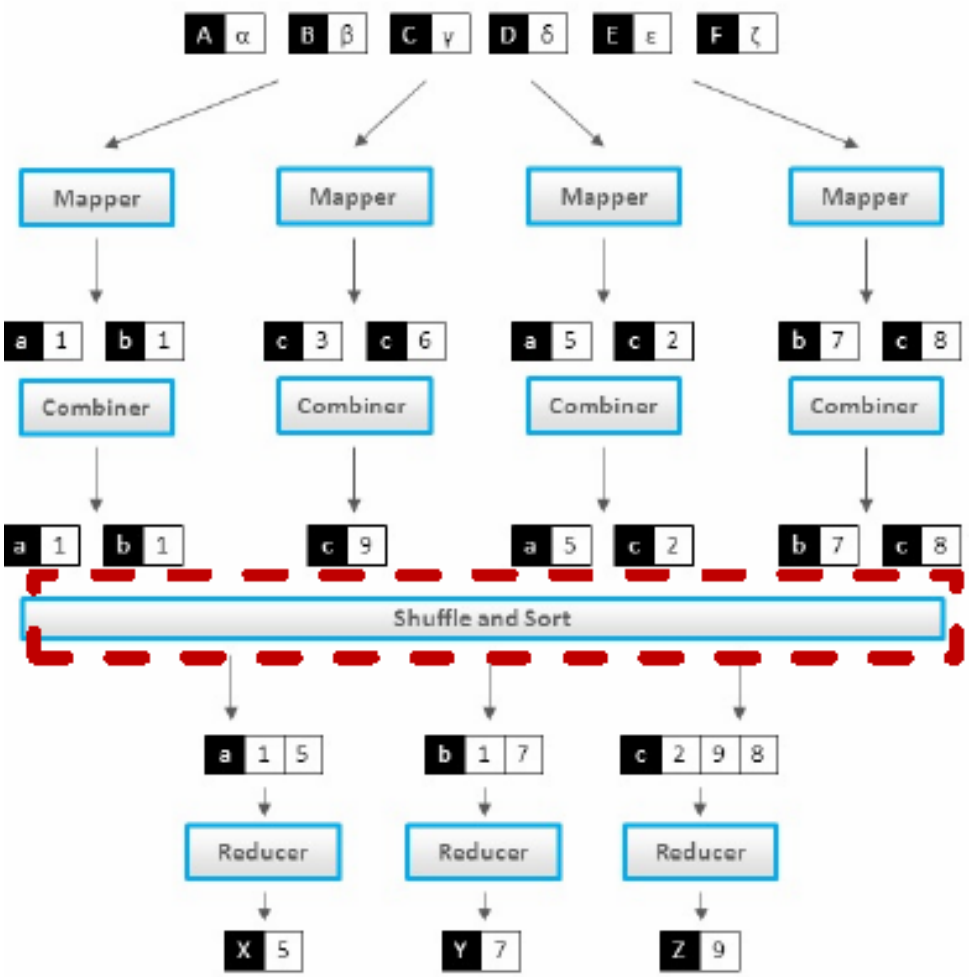
How to use Hadoop?

- Implement 2 basic functions:
  - Map
  - Reduce

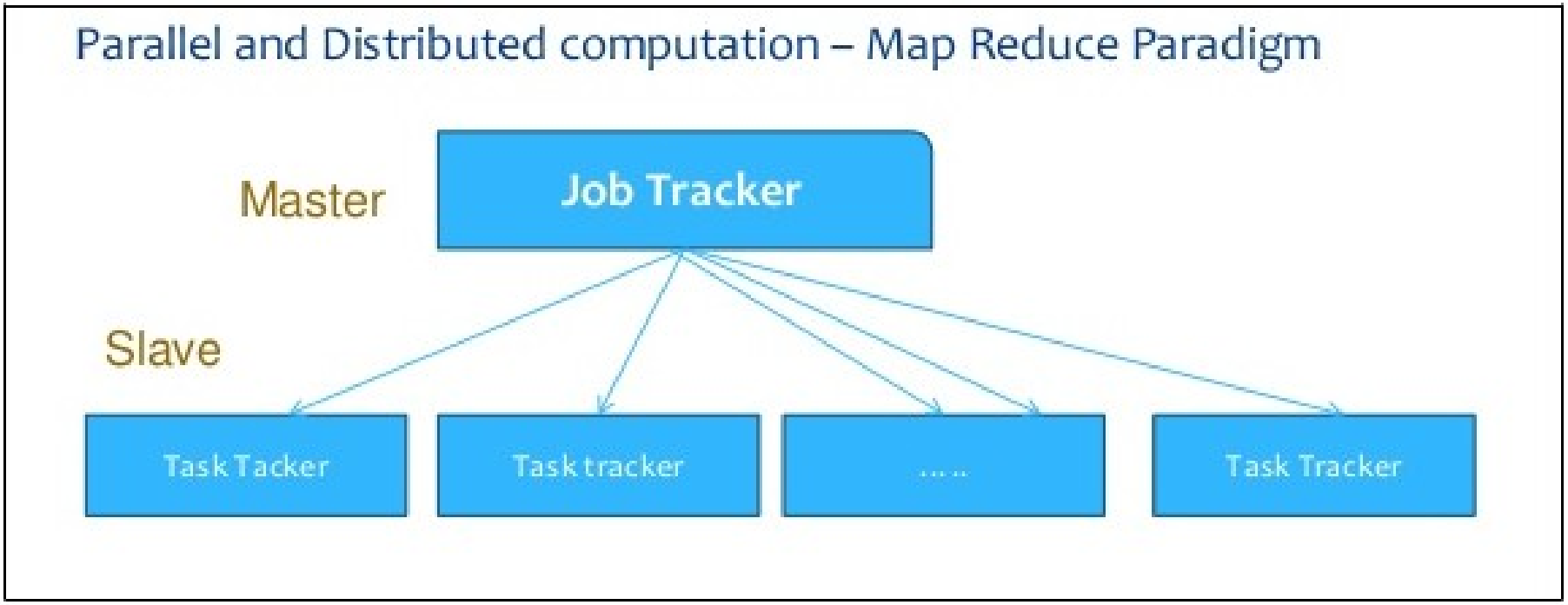
MapReduce


The overall MapReduce word count process
Input
Splitting
Mapping
Shuffling
Reducing
Final result
Deer Bear River
Car Car River
Deer Car Bear
Deer Bear River
Car Car River
Deer Car Bear
Deer, 1
Bear, 1
River, 1
Car, 1
Car, 1
River, 1
Deer, 1
Car, 1
Bear, 1
Bear, 1
Bear, 1
Car, 1
Car, 1
Car, 1
Deer, 1
Deer, 1
River, 1
River, 1
Bear, 2
Car, 3
Deer, 2
River, 2
Bear, 2
Car, 3
Deer, 2
River, 2

MapReduce structure



Job submission



Job submission

- Client applications submit jobs to the Job tracker
- The JobTracker talks to the NameNode to determine the location of the data
- The JobTracker locates TaskTracker nodes with available slots at or near the data
- The JobTracker submits the work to the chosen TaskTracker nodes
- The TaskTracker nodes are monitored. If they do not submit heartbeat signals often enough, they are deemed to have failed and the work is scheduled on a different TaskTracker

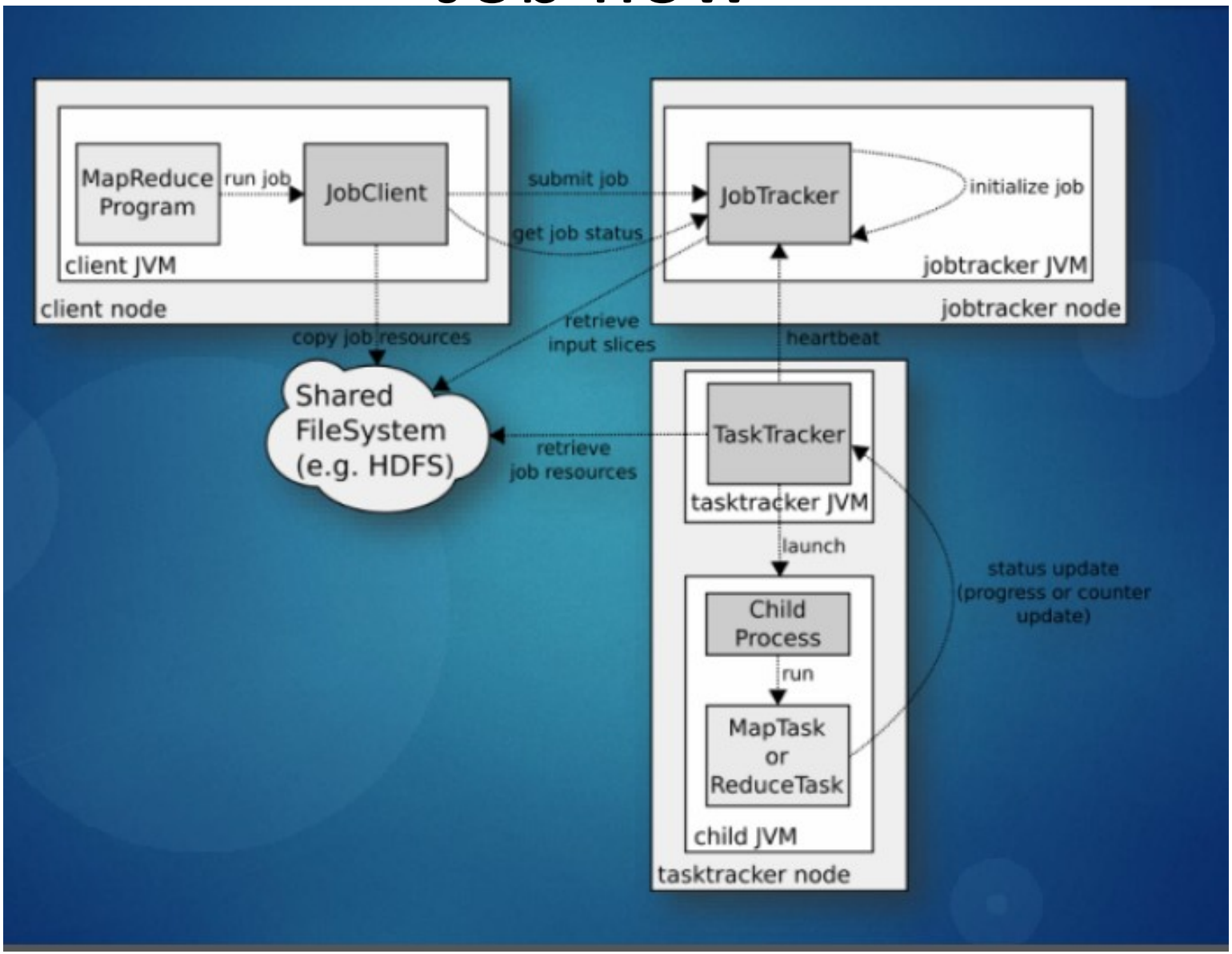
Job submission

- A TaskTracker will notify the JobTracker when a task fails. The JobTracker decides what to do then: it may resubmit the job elsewhere, it may mark that specific record as something to avoid, and it may may even blacklist the TaskTracker as unreliable

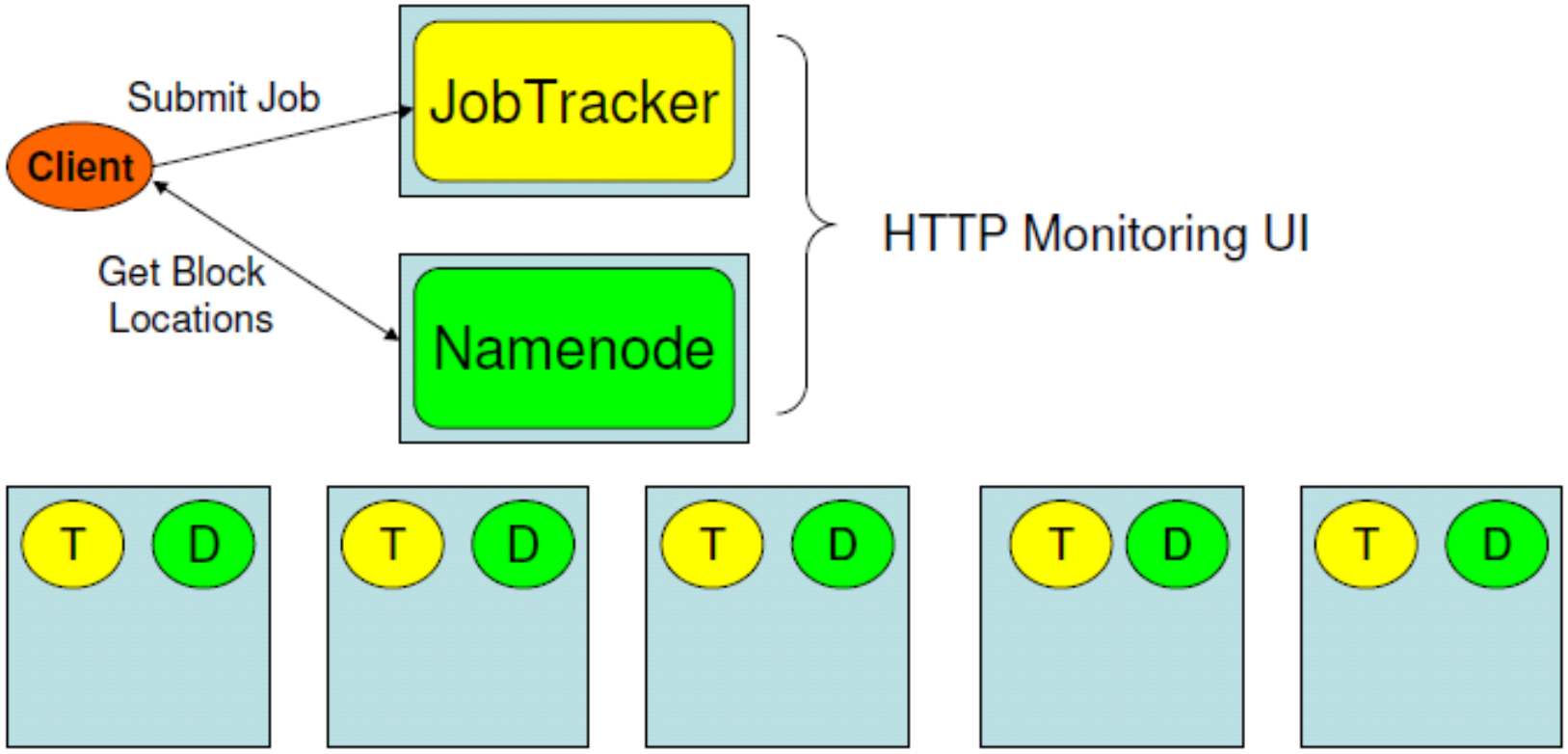
- When the work is completed, the JobTracker updates its status

- Client applications can poll the JobTracker for information

Job flow


MapReduce Program
run job
JobClient
client JVM
client node
submit job
get job status
JobTracker
initialize job
jobtracker JVM
jobtracker node
copy job resources
retrieve input slices
heartbeat
Shared FileSystem (e.g. HDFS)
retrieve job resources
TaskTracker
tasktracker JVM
launch
Child Process
run
MapTask or ReduceTask
child JVM
tasktracker node
status update (progress or counter update)

Hadoop HDFS + MR cluster



Machines with Datanodes and Tasktrackers

What is Hadoop not

- A replacement for existing data warehouse systems
- A database

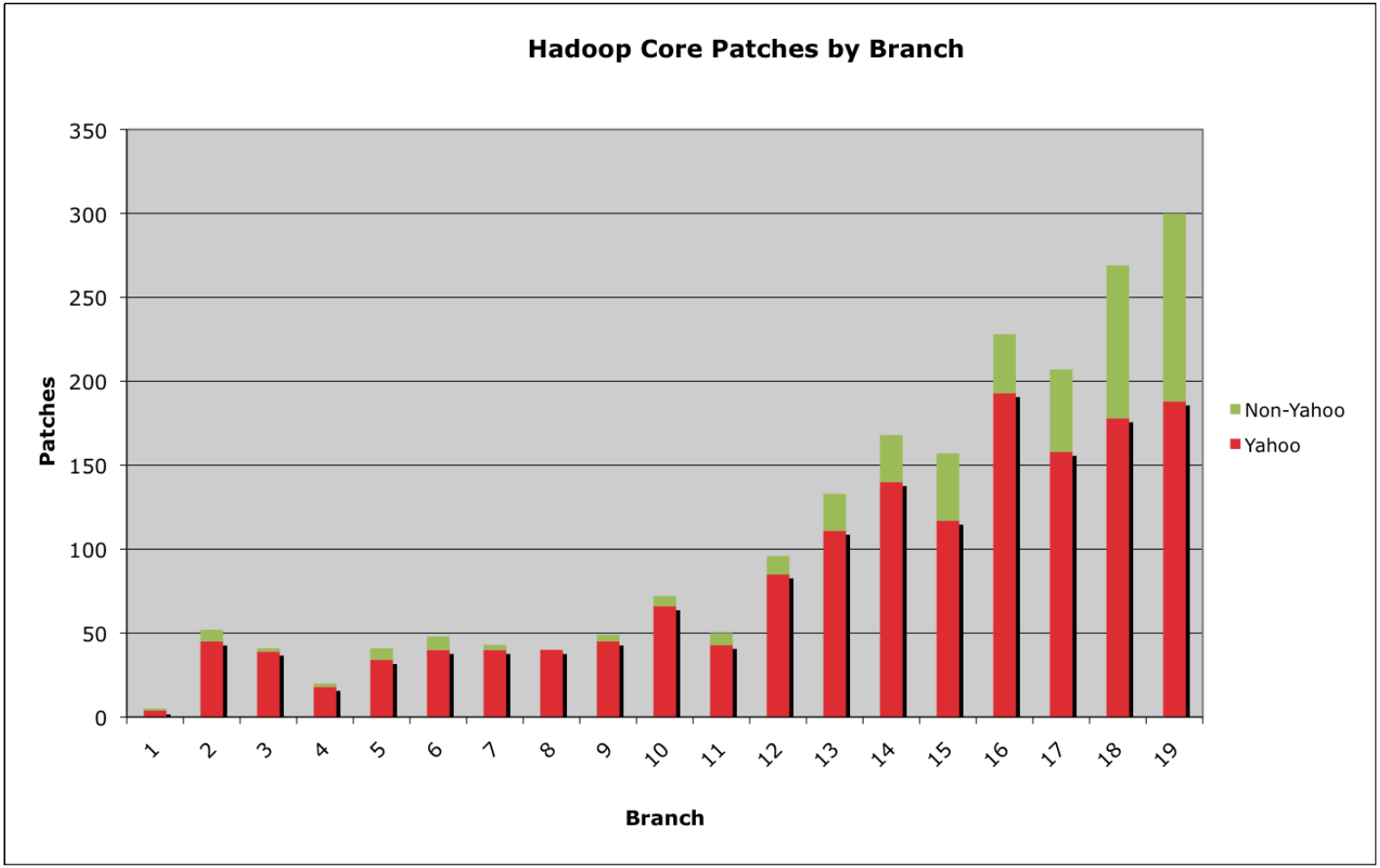
Hadoop-related projects

- Apache Pig [high-level platform for querying]
- Apache Hive [data warehouse infrastructure]
- Apache Hbase [database for real-time access]
- Apache Sqoop [CLI for SQL to/from Hadoop]
- …

Examples of production use

- Yahoo! : More than 100,000 CPUs in ~20,000 computers running Hadoop; biggest cluster: 2000 nodes (2*4cpu boxes with 4TB disk each); used to support research for Ad Systems and Web Search
- Facebook: To store copies of internal log and dimension data sources and use it as a source for reporting/analytics and machine learning; 320 machine cluster with 2,560 cores and about 1.3 PB raw storage

Size of releases



Hadoop

+ Framework for applications on large clusters
+ Built for commodity hardware
+ Provides reliability and data motion
+ Implements a computational paradigm named Map/Reduce
+ Very own distributed file system (HDFS) (very high aggregate bandwidth across the cluster)
+ Failures handles automatically

Hadoop

- Time consuming development
- Documentation sufficient, but not the most helpful
- HDFS is complicated and has plenty issues of its own
- Debugging a failure is a "nightmare"
- Large clusters require a dedicated team to keep it running properly
- Writing a Hadoop job becomes a software engineering task rather than a data analysis task

Alternatives

- BashReduce - MapReduce for std. Unix commands (no task coordination, lack of fault tolerance)
- Disco Project - MapReduce by Nokia Research for Python written in Erlang (no own file system, no persistent fault tolerance)
- Spark - UC Berkeley implementation in Scala with in-memory querying even distributed across machines (depends on cluster manager Mesos)