北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

Advanced Software Engineering

Lecture 9: Cloud Computing and Big Data

by

Prof. Harold Liu

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Content

- Cloud Computing
- Hadoop
- HDFS
- MapReduce

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

A Cloud is …

- A data center hardware and software that the vendors use to offer the computing resources and services

Cloud Computing

"Cloud Computing is the transformation of IT from a product to a service"

Innovation → Product → Service

Cloud Computing

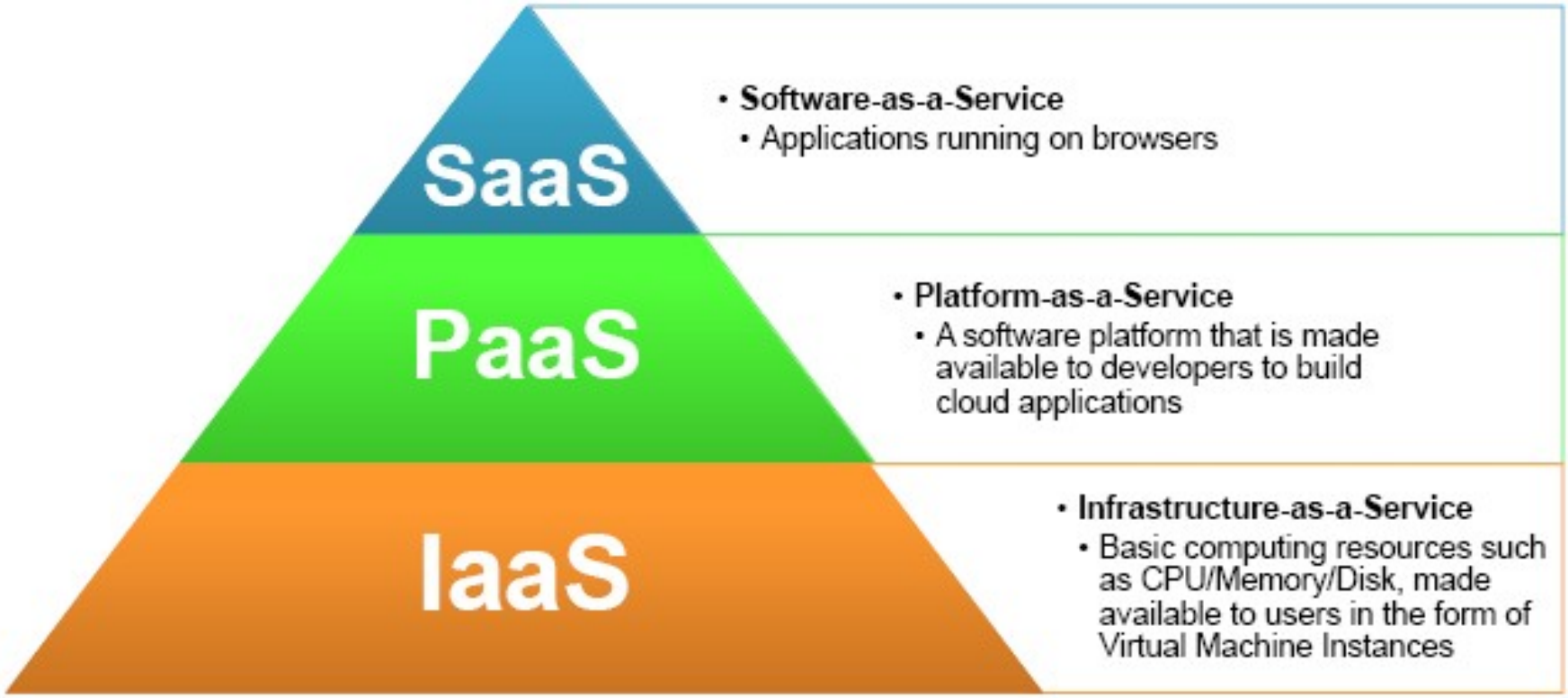
北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Cloud Computing is the delivery of computing as a **service** rather than a **product**,

whereby **shared resources, software**, and **information** are provided to computers and other devices,

as a **metered service** over a **network.**

Cloud Service Models

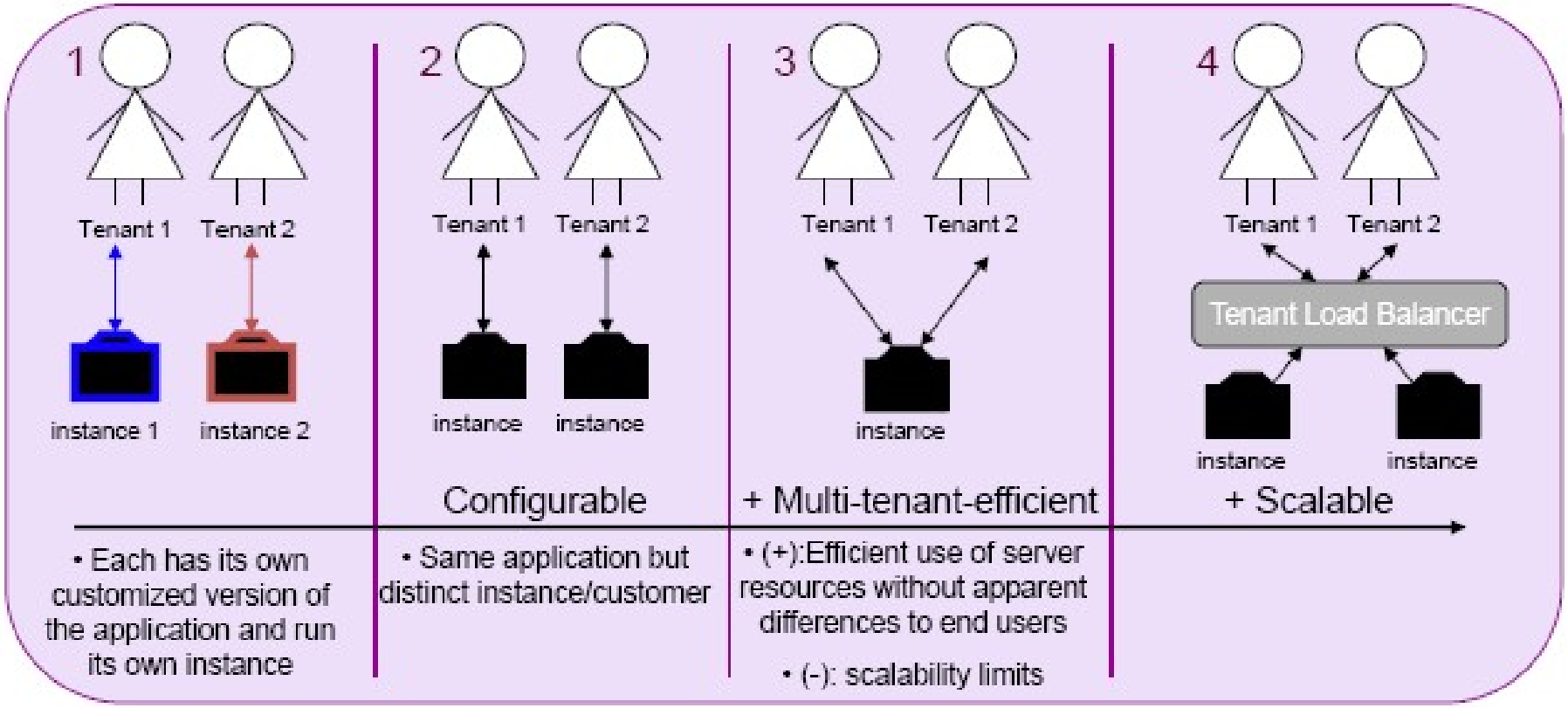


SaaS

- You are most familiar with this!
- Software is delivered as a service over the Internet, eliminating the need to install and run the application on the customer's own computer
- This simplifies maintenance and support
- Examples: Gmail, YouTube, and Google Docs, among others

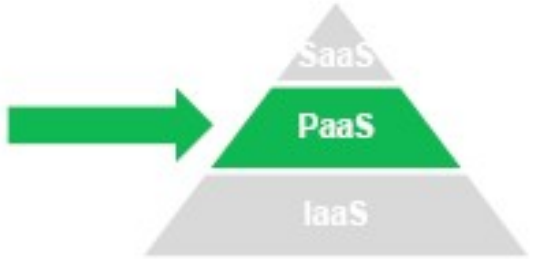
SaaS Maturity Levels

- **Distinguishing attributes:** configurability, multi-tenant efficiency, scalability

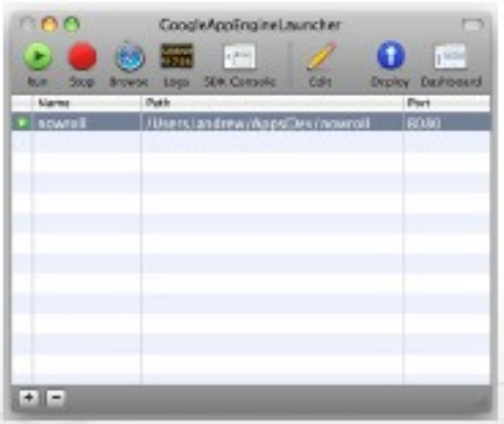


PaaS

- The Cloud provider exposes a set of tools (a platform) which allows users to create SaaS applications



- The SaaS application runs on the provider's infrastructure



- The cloud provider manages the underlying hardware and requirements

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

PaaS Example I

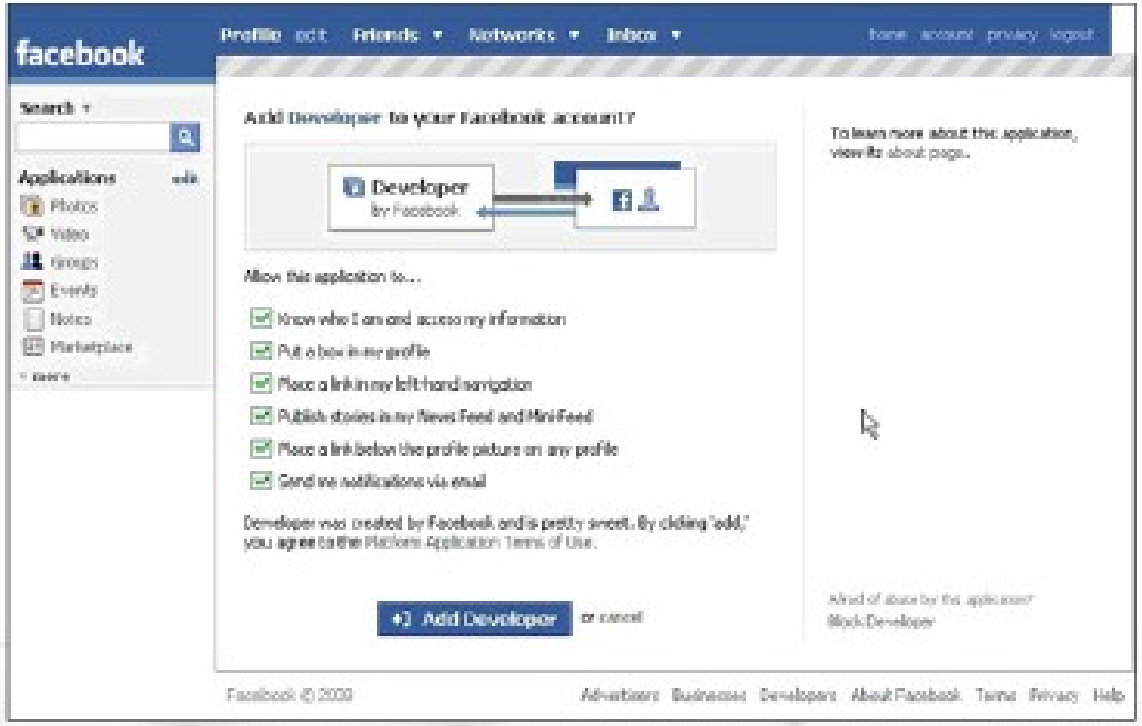
- Google App Engine

Build web applications on Google's Infrastructure

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

PaaS Example II

- The Facebook Developer Platform

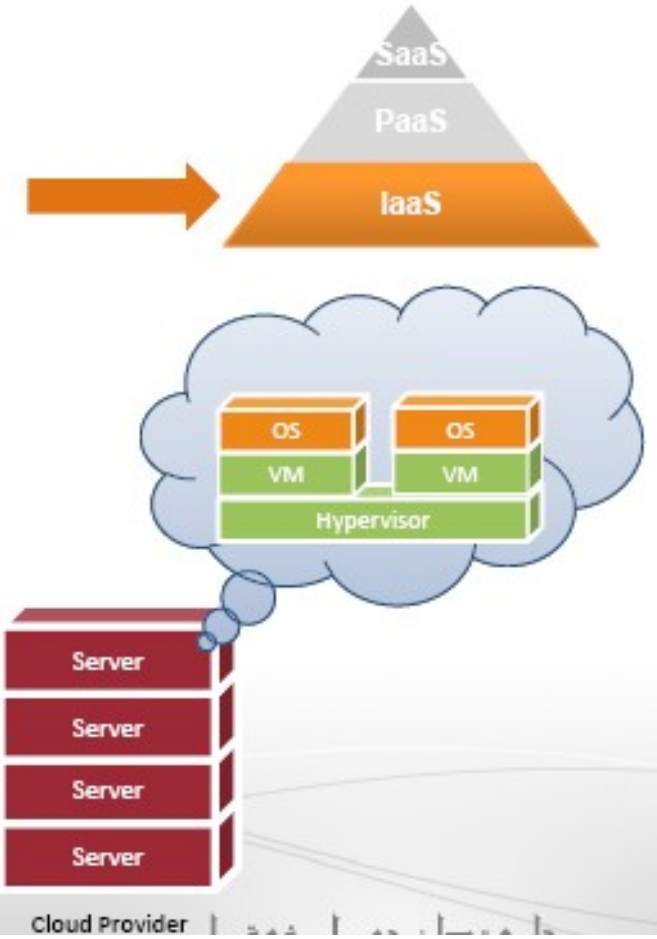


Set of APIs that allow you to create Facebook Applications

IaaS (1/3)

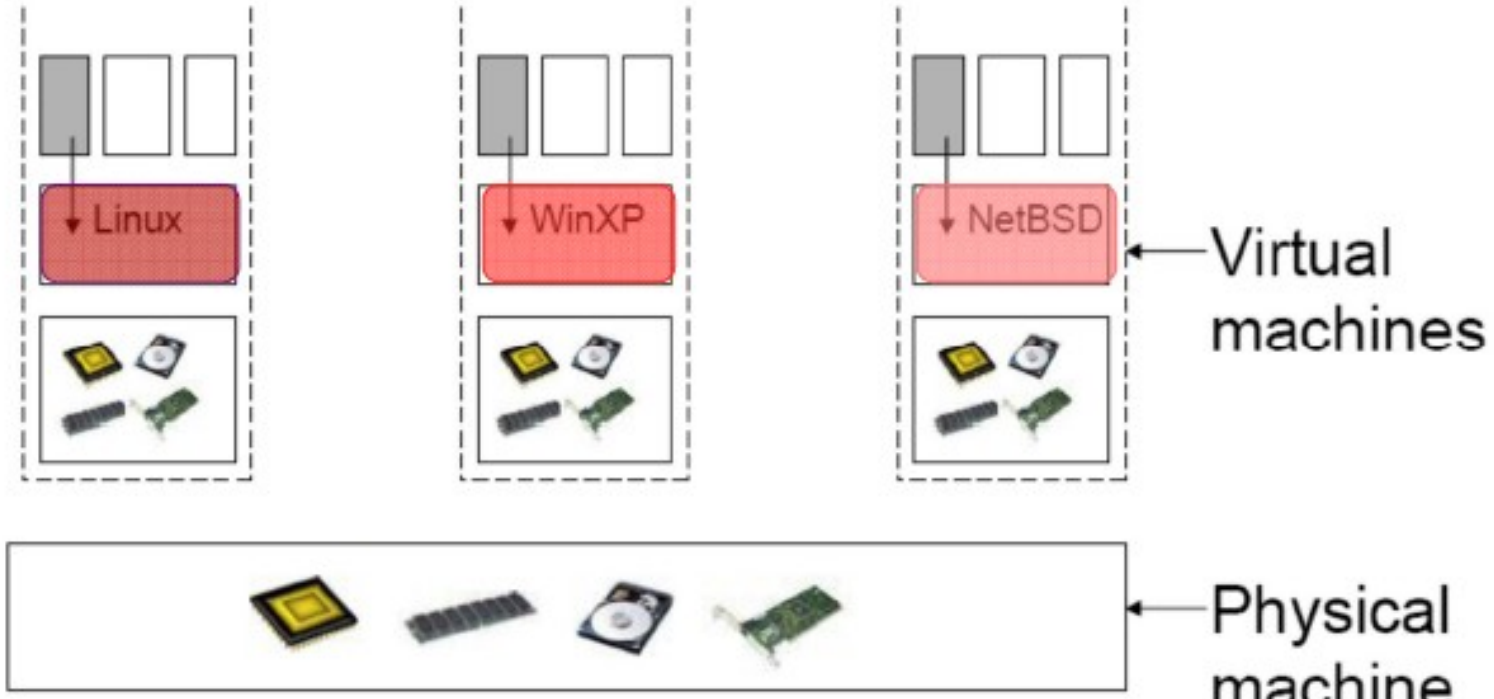
- The cloud provider leases to users Virtual Machine Instances (i.e., computer infrastructure) using the ***virtualization*** technology

- The user has access to a standard Operating System environment and can install and configure all the layers above it

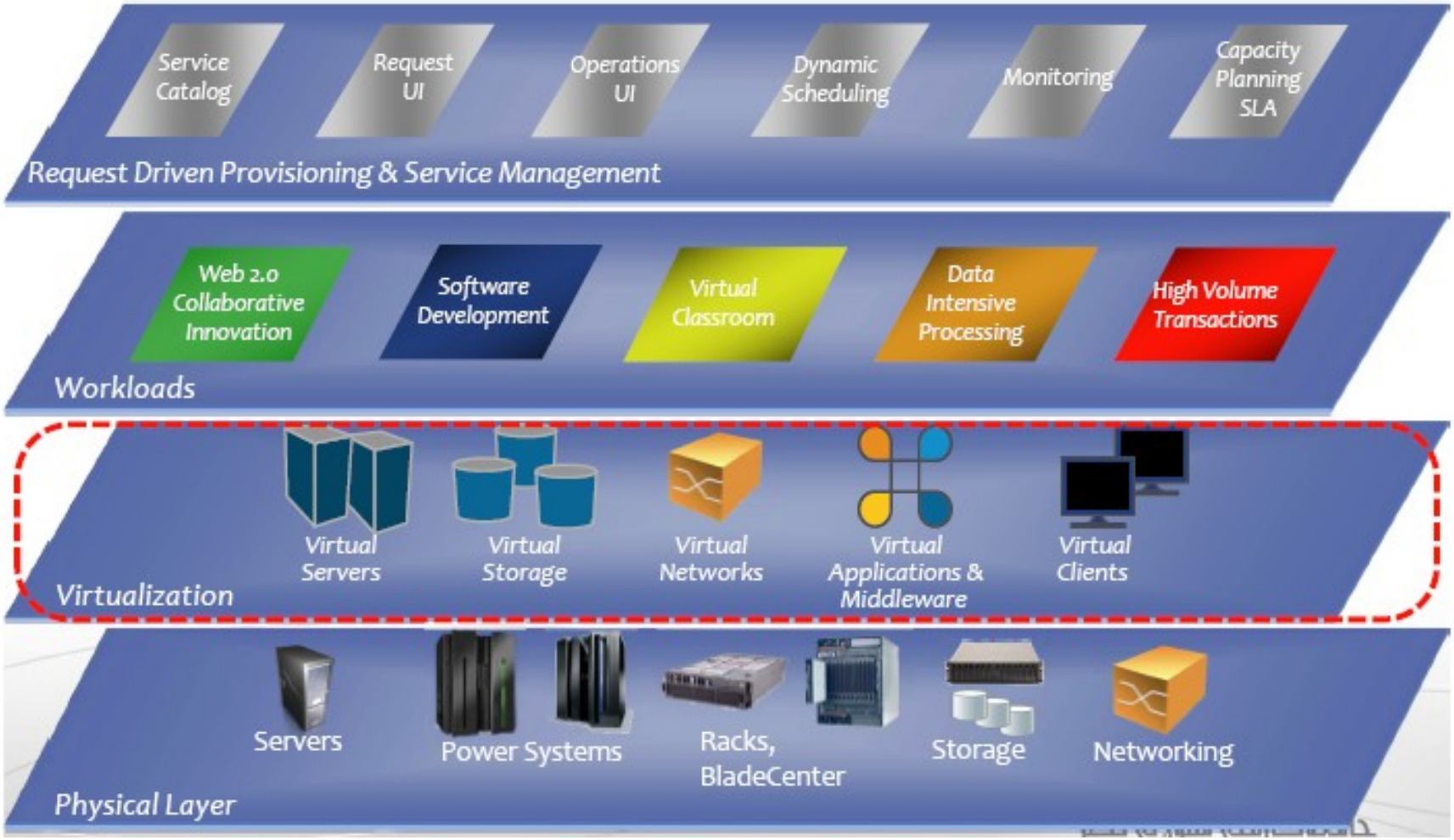


IaaS (2/3)

- The virtualization technology is a major enabler of IaaS



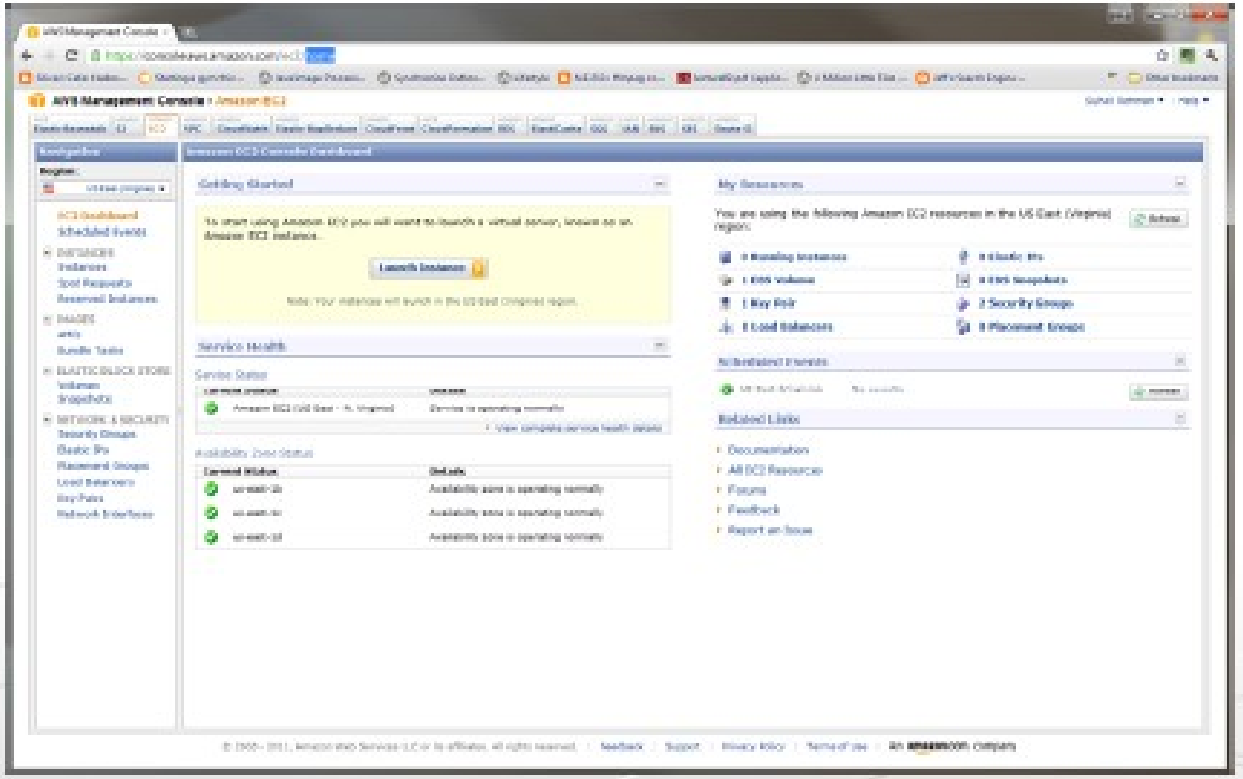
IaaS (3/3)


Service Catalog
Request UI
Operations UI
Dynamic Scheduling
Monitoring
Capacity Planning SLA
Request Driven Provisioning & Service Management
Web 2.0 Collaborative Innovation
Software Development
Virtual Classroom
Data Intensive Processing
High Volume Transactions
Workloads
Virtual Servers
Virtual Storage
Virtual Networks
Virtual Applications & Middleware
Virtual Clients
Virtualization
Servers
Power Systems
Racks, BladeCenter
Storage
Networking
Physical Layer

北京理工大學
BEIJING INSTITUTE OF TECHNOLOGY

IaaS Example

- Amazon Web Service Elastic Compute Cloud (EC2)



北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

The Cloud Stack

Applications

- Cloud applications can range from Web applications to scientific computational jobs

Data

- Data Management
- New generation cloud-specific databases and management systems
- E.g., Hbase, Cassandra, Hive, Pig etc.

Runtime Environment

- Runtime platforms to support cloud programming models
- E.g., MPI, MapReduce, Pregel etc.

Middleware for Clouds

- Management platforms that enable:
  - Resource Management
  - Monitoring
  - Provisioning
  - Identity Management and Security

Operating Systems

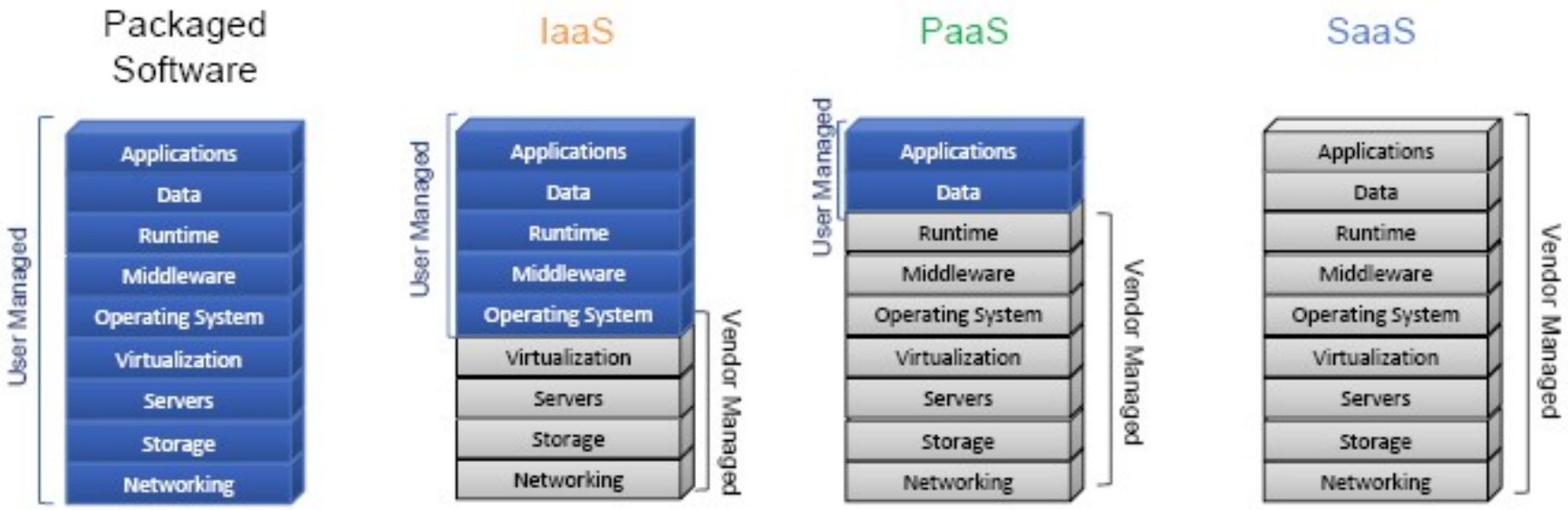
- Standard Operating Systems used in Personal Computing
- Packaged with libraries and software for quick deployment and provisioning
- E.g., Amazon Machine Images (AMI) contain OS as well as required software packages as a “snapshot” for instant deployment

Virtualization

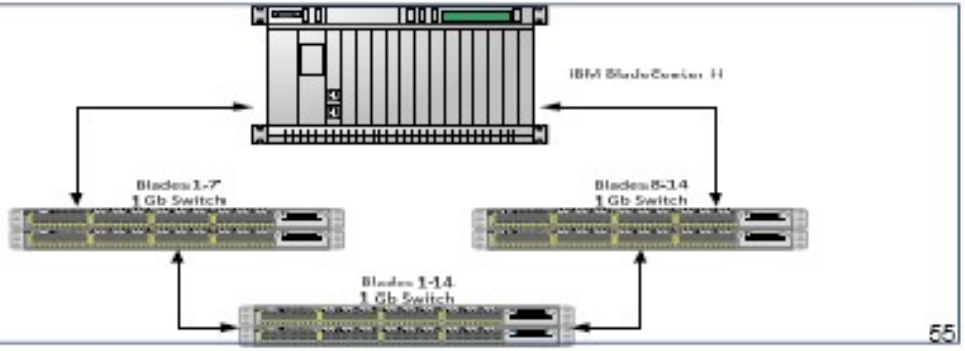
- Key Component
- Resource Virtualization
- Amazon EC2 is based on the Xen virtualization platform

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Cloud Service Layers in the Service Levels



Qloud Stack

Applications	Bayesian Classification, K-Means, etc.
Data	HDFS
Runtime	Apache Hadoop 0.20.1
Middleware	Ganglia cluster monitoring system and the VMware vSphere Client
Operating System	64-Bit Fedora 13
Virtualization	Vmware vSphere 4.1/ESXi 4.1
Servers	14 IBM Quad Core (E5420) Blades
Storage	Storage/Blade = 2 x 300 GB SAS & RAM/Blade = 8 GB RAM
Networking	

An Ecosystem for Cloud Computing

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Problem

- Batch (offline) processing of huge data set using commodity hardware is not enough for real-time applications

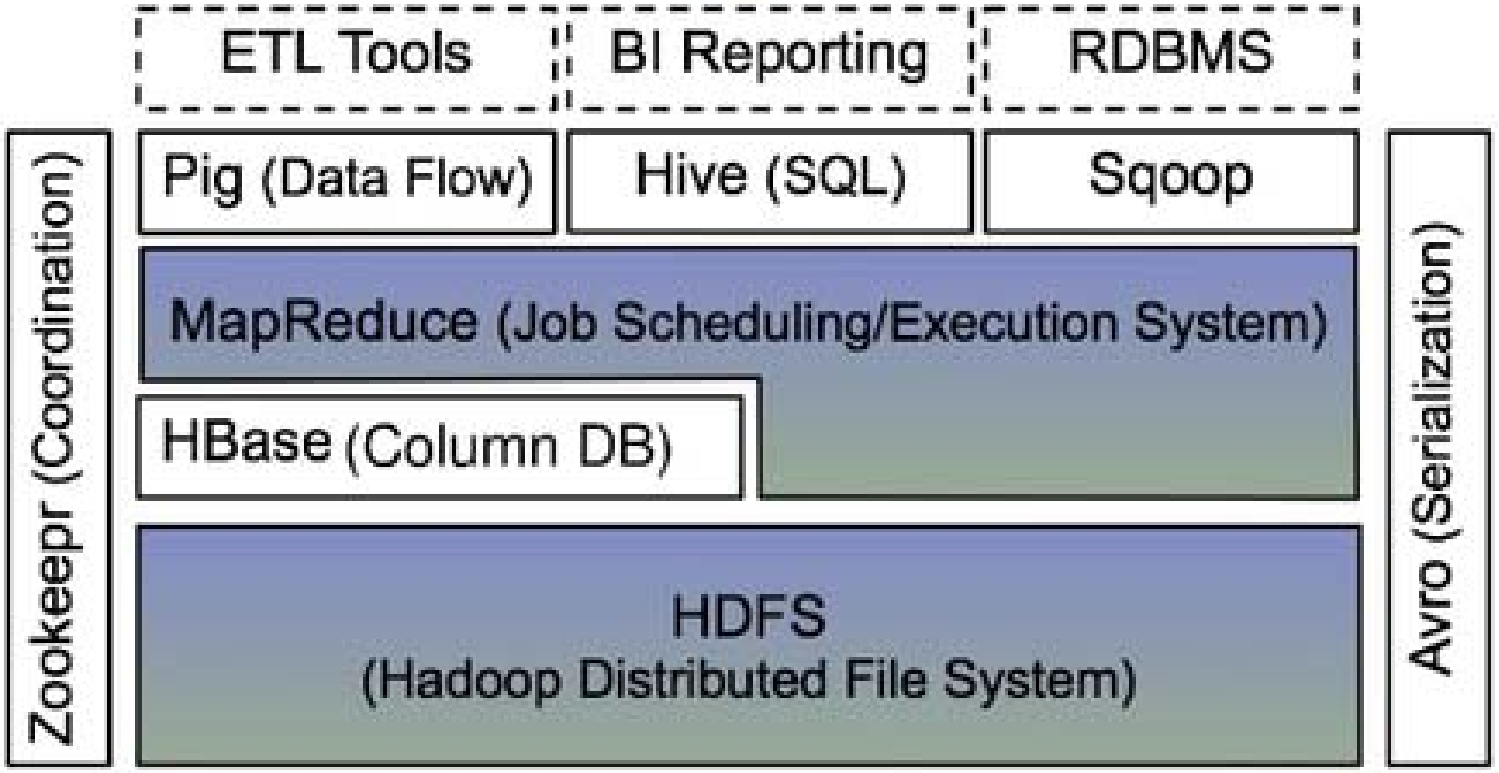
- Strong desire for linear scalability

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Explosive Data! – Storage

- New York Stock Exchange: 1 TB data per day
- Facebook: 100 billion photos, 1 PB (1000 TB)
- Internet Archive: 2 PB data, growing by 20 TB per month
- Can’t put data on a SINGLE node
- Strong needs for distributed file systems

The Hadoop Ecosystem



Java/Python/C interfaces

cloudera

Naming it Hadoop

- The name Hadoop is not an acronym; it's a made-up name. The project's creator, Doug Cutting, explains how the name came about:
- The name my kid gave a stuffed yellow elephant. Short, relatively easy to spell and pronounce, meaningless, and not used elsewhere: those are my naming criteria. Kids are good at generating such. Googol is a kid's term.

- Building a web search engine from scratch was an ambitious goal
- Mike Cafarella and Doug Cutting estimated a system supporting a 1-billion-page index would cost around half a million dollars in hardware, with a monthly running cost of $30,000.11
- Nutch was started in 2002, however not scale to the billions of pages on the Web.
- Help was at hand with Google's distributed filesystem, GFS, in 2003
- In 2004, they set about writing an open source implementation, Nutch Distributed Filesystem (NDFS).
- In 2004, Google published a paper that introduced MapReduce
- Early in 2005, the Nutch developers had a working MapReduce implementation in Nutch, and by the middle of that year all the major Nutch algorithms had been ported to run using MapReduce and NDFS.
- February 2006 they moved out of Nutch to form Hadoop. At around the same time, Doug Cutting joined Yahoo!, which provided a dedicated team and the resources to turn Hadoop into a system that ran at web scale
- demonstrated in February 2008 when Yahoo! announced that its production search index was being generated by a 10,000-core Hadoop cluster.

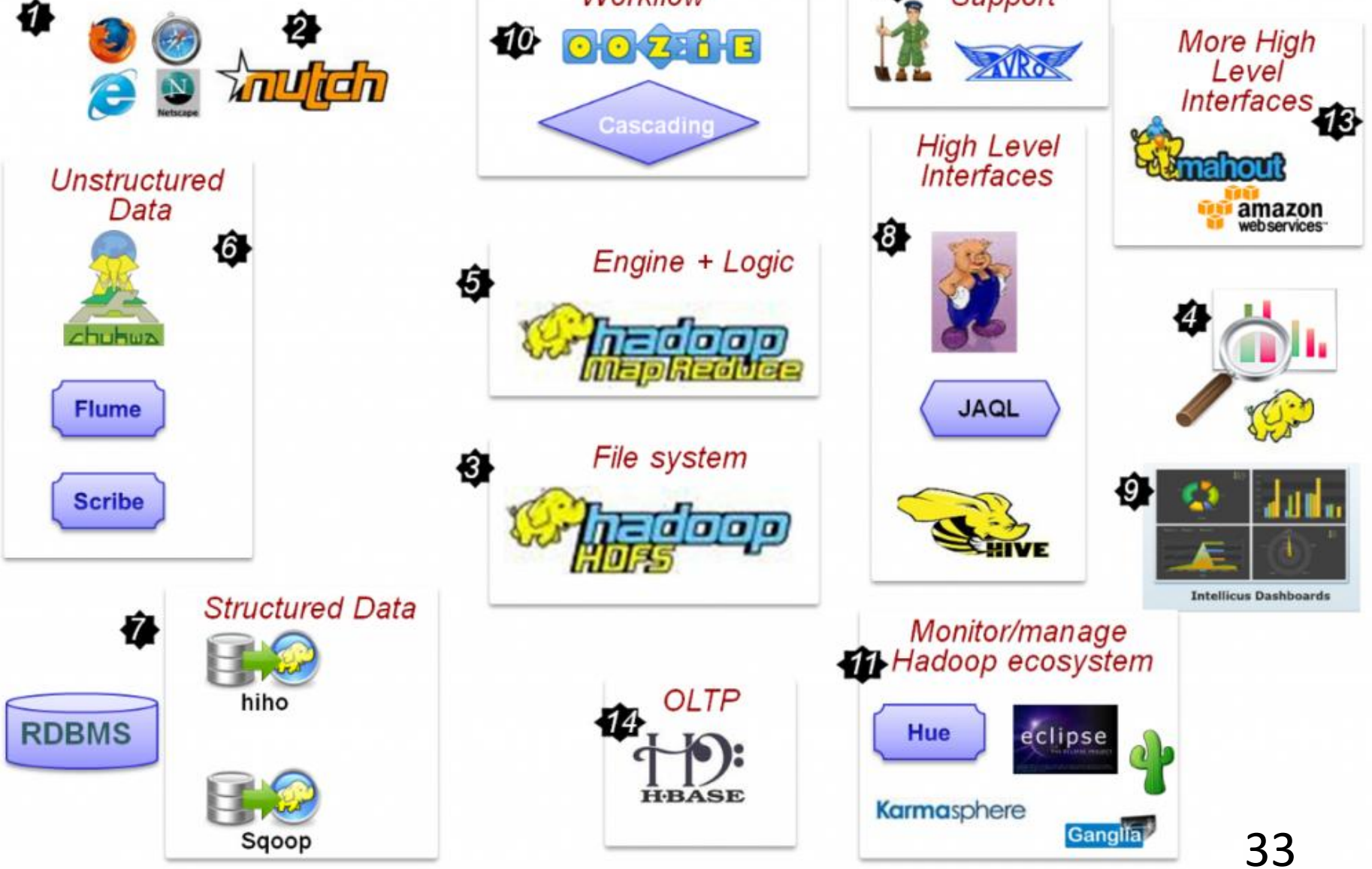
- In January 2008, Hadoop was made its own top-level project at Apache
- *New York Times* used Amazon's EC2 compute cloud to crunch through 4TB of scanned archives from the paper converting them to PDFs for the Web.
- The processing took less than 24 hours to run using 100 machines, and the project probably wouldn't have been embarked on without the combination of Amazon's pay-by-the-hour model (which allowed the NYT to access a large number of machines for a short period) and Hadoop's easy-to-use parallel programming model.
- In April 2008, Hadoop broke a world record to become the fastest system to sort a terabyte of data. Running on a 910-node cluster, Hadoop sorted one terabyte in 209 seconds
- In November of the same year, Google reported that its MapReduce implementation sorted one terabyte in 68 seconds.
- As the first edition of this book was going to press (May 2009), it was announced that a team at Yahoo! used Hadoop to sort one terabyte in 62 seconds.

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Hadoop History

- Dec 2004 – Google GFS paper is published
- July 2005 – Nutch project uses MapReduce
- Feb 2006 – becomes a subproject of Lucene
- Apr 2007 – Yahoo! established a 1000 node Hadoop cluster
- Jan 2008 – becomes Apache top-tier project
- Jul 2008 – established 4000 node cluster
- Sept 2008 – Hive becomes Hadoop subproject
- ……

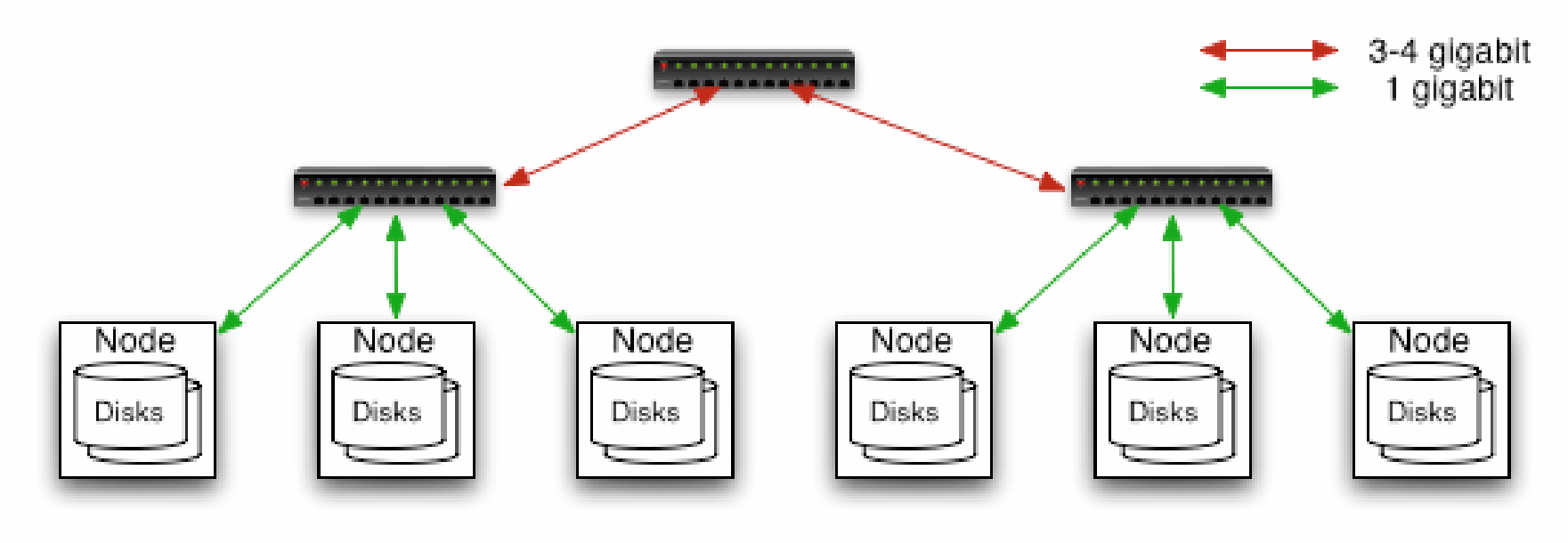
Hadoop Ecosystem Map



北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Who is Using Hadoop?

Commercial Hardware



Two-tier architecture

– commodity servers (cheap)

– 30-40 servers/Rack

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

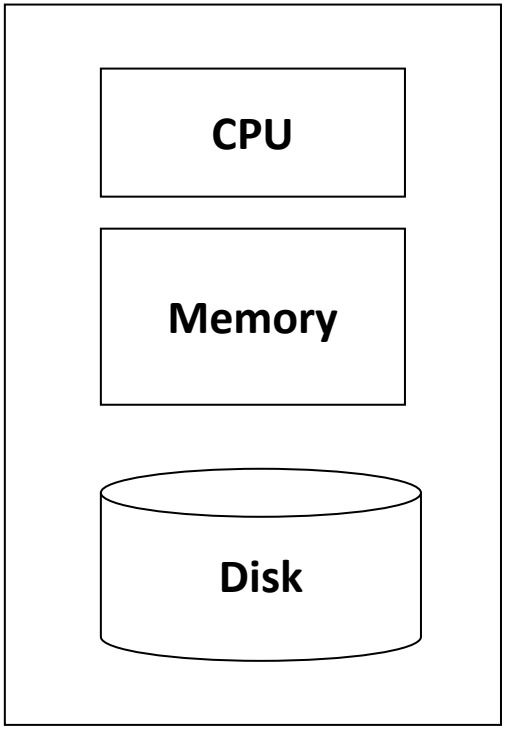
Example: Facebook Hadoop Cluster

- **Product Cluster**
  - 4800 core, 600 servers, each 16GB RAM — April 2009
  - 8000个 core, 1000 servers, each 32GB RAM — July 2009
  - Each has four 1TB SATA disk
  - Two layer architecture
  - Each rack has 40 servers
  - In total 2PB storage

- **Testing Cluster**
  - 800 core, each 16GB RAM

A Distributed File System

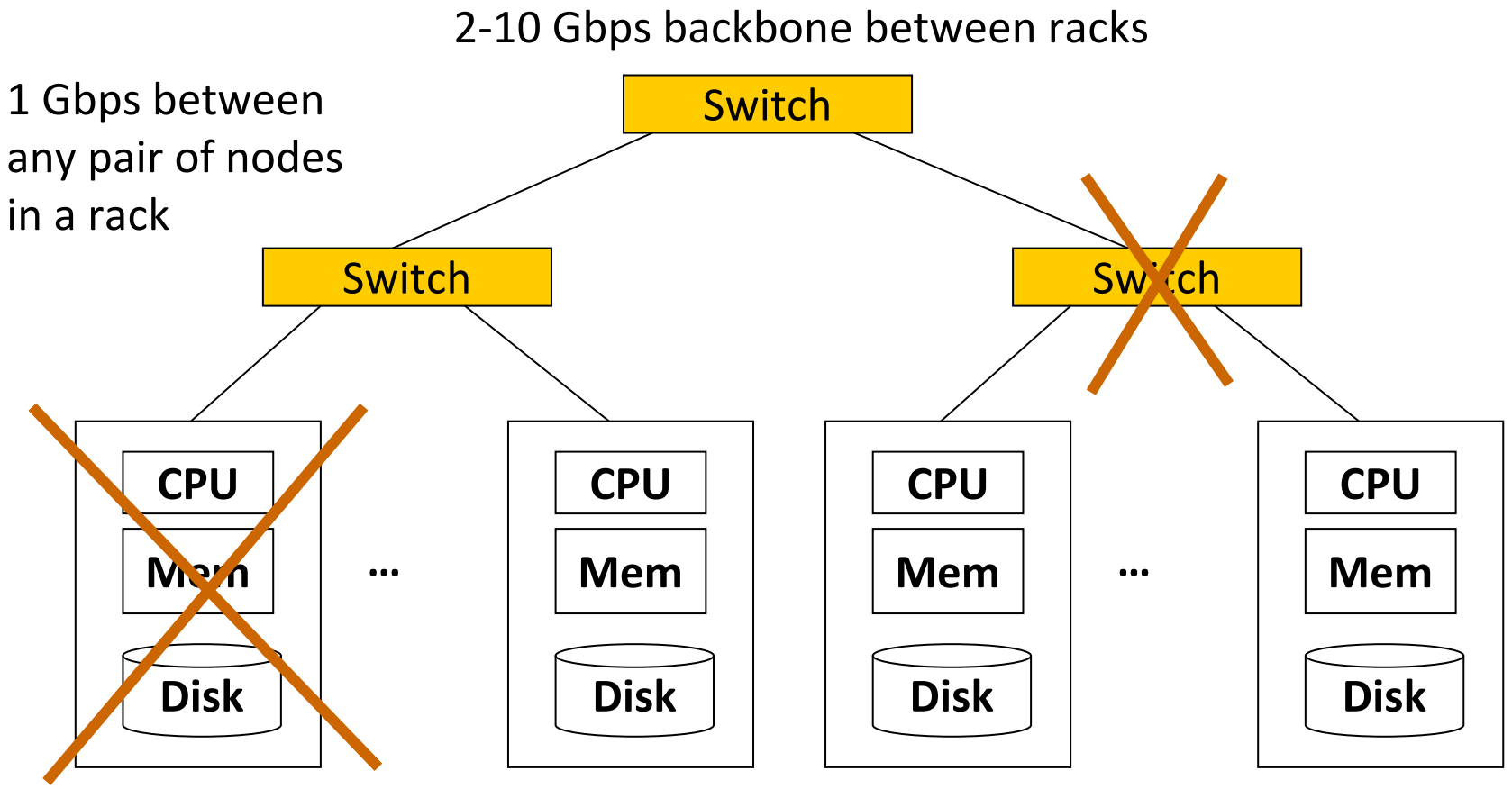
Single-Node Architecture



Machine Learning, Statistics

"Classical" Data Mining

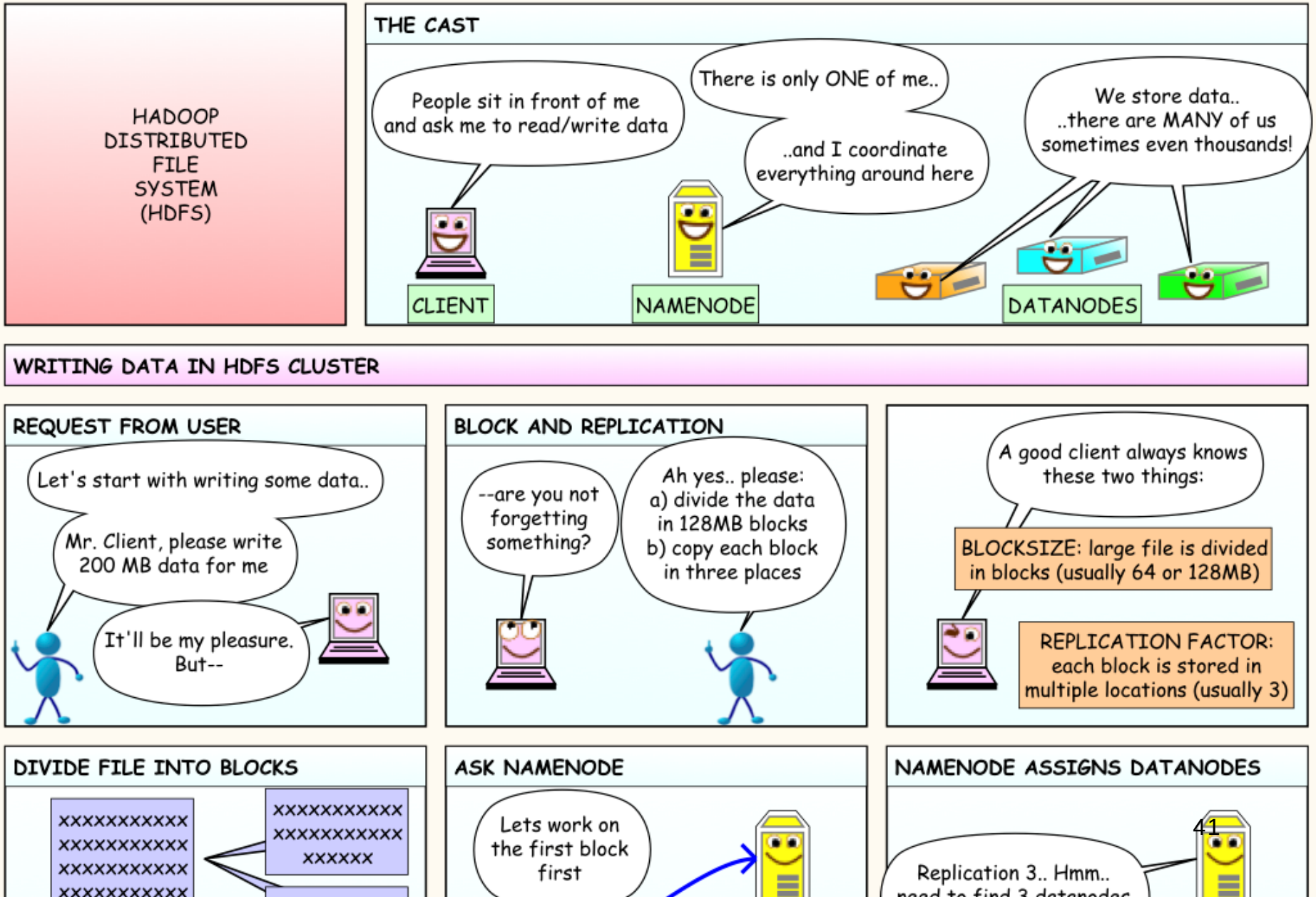
Cluster Architecture

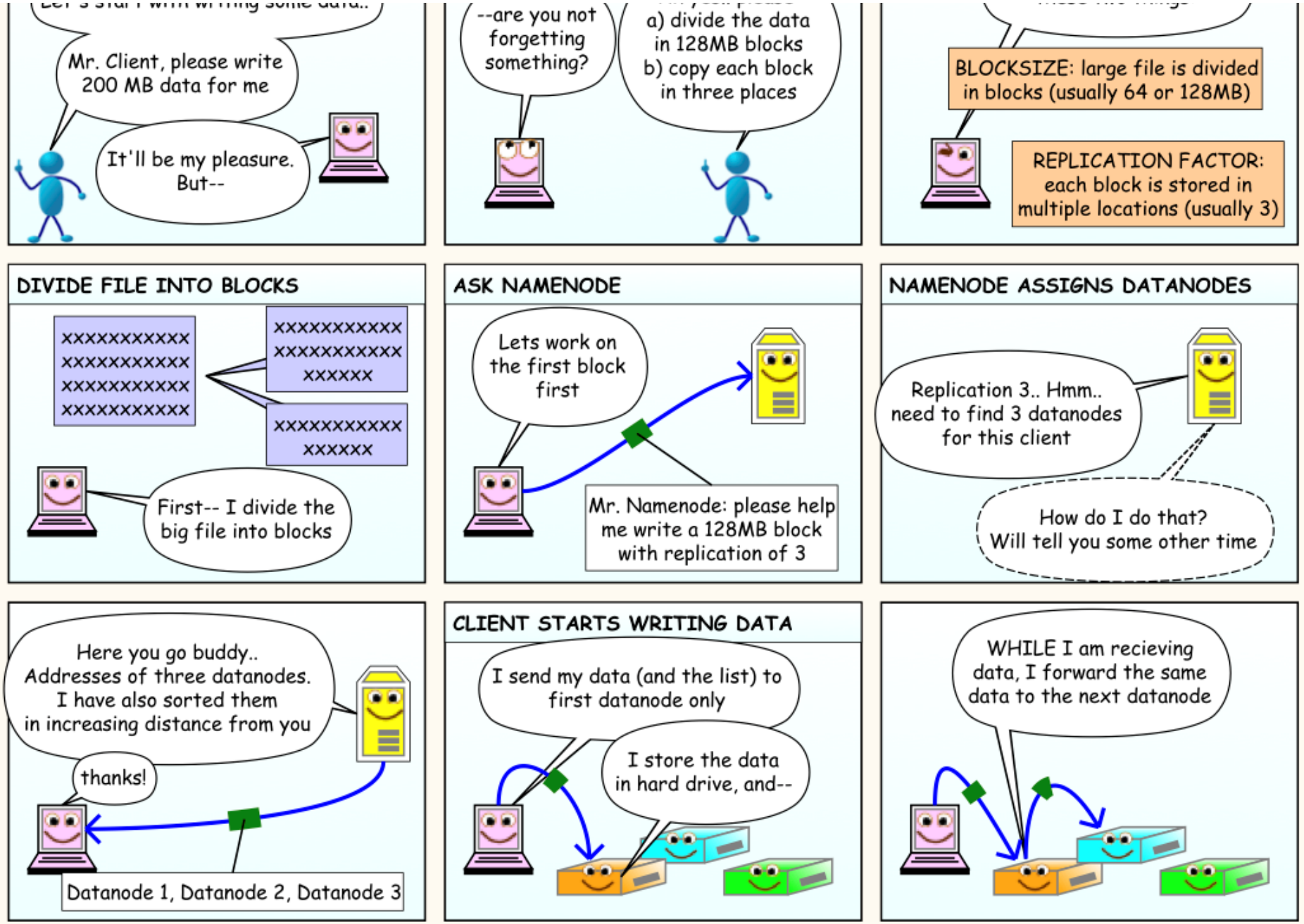


北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

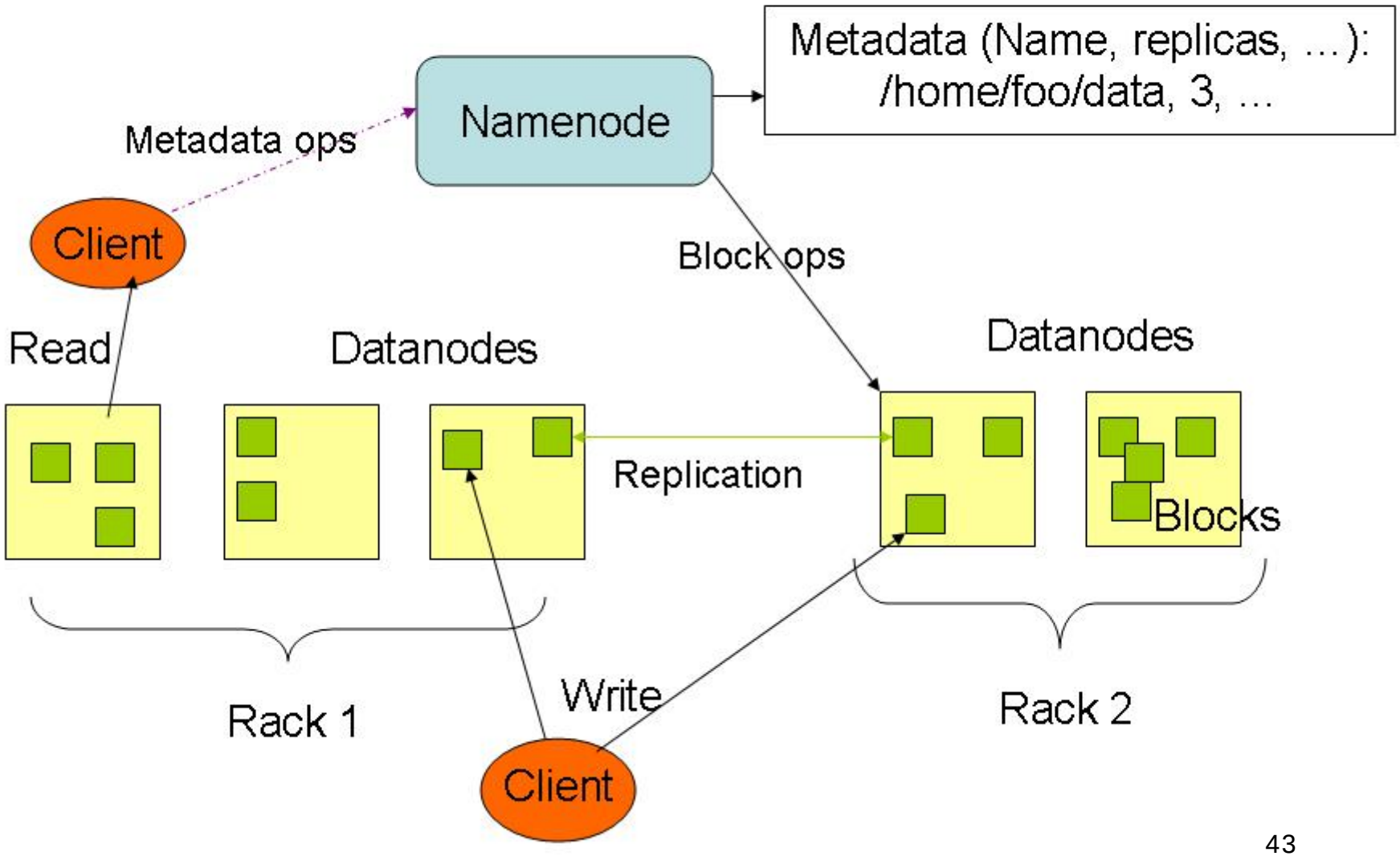
Stable Storage with Fault Tolerance

- If nodes can fail, how can we store data persistently?
- Answer: Distributed File System
  - Provides global file namespace
  - Google GFS
  - Hadoop HDFS
  - Kosmix KFS


HADOOP DISTRIBUTED FILE SYSTEM (HDFS)
THE CAST
People sit in front of me and ask me to read/write data
There is only ONE of me..
..and I coordinate everything around here
We store data.. ..there are MANY of us sometimes even thousands!
CLIENT
NAMENODE
DATANODES
WRITING DATA IN HDFS CLUSTER
REQUEST FROM USER
Let's start with writing some data..
Mr. Client, please write 200 MB data for me
It'll be my pleasure. But--
BLOCK AND REPLICATION
--are you not forgetting something?
Ah yes.. please: a) divide the data in 128MB blocks b) copy each block in three places
A good client always knows these two things:
BLOCKSIZE: large file is divided in blocks (usually 64 or 128MB)
REPLICATION FACTOR: each block is stored in multiple locations (usually 3)
DIVIDE FILE INTO BLOCKS
ASK NAMENODE
Lets work on the first block first
NAMENODE ASSIGNS DATANODES
Replication 3.. Hmm..



HDFS Architecture



北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Namenode and Datanodes

- Master/slave architecture
- 1 Namenode, a master server that manages the file system namespace and regulates access to files by clients.
- many DataNodes usually one per node in a cluster.
  - manage storage
  - serves read, write requests, performs block creation, deletion, and replication upon instruction from Namenode.
- HDFS exposes a file system namespace and allows user data to be stored in files.
- A file is split into one or more blocks and set of blocks are stored in DataNodes.

Namespace

- Hierarchical file system with directories and files
- Create, remove, move, rename etc.
- Namenode maintains the file system
- Any meta information changes to the file system recorded by Namenode.
- An application can specify the number of replicas of the file needed: replication factor of the file. This information is stored in the Namenode.

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Data Replication

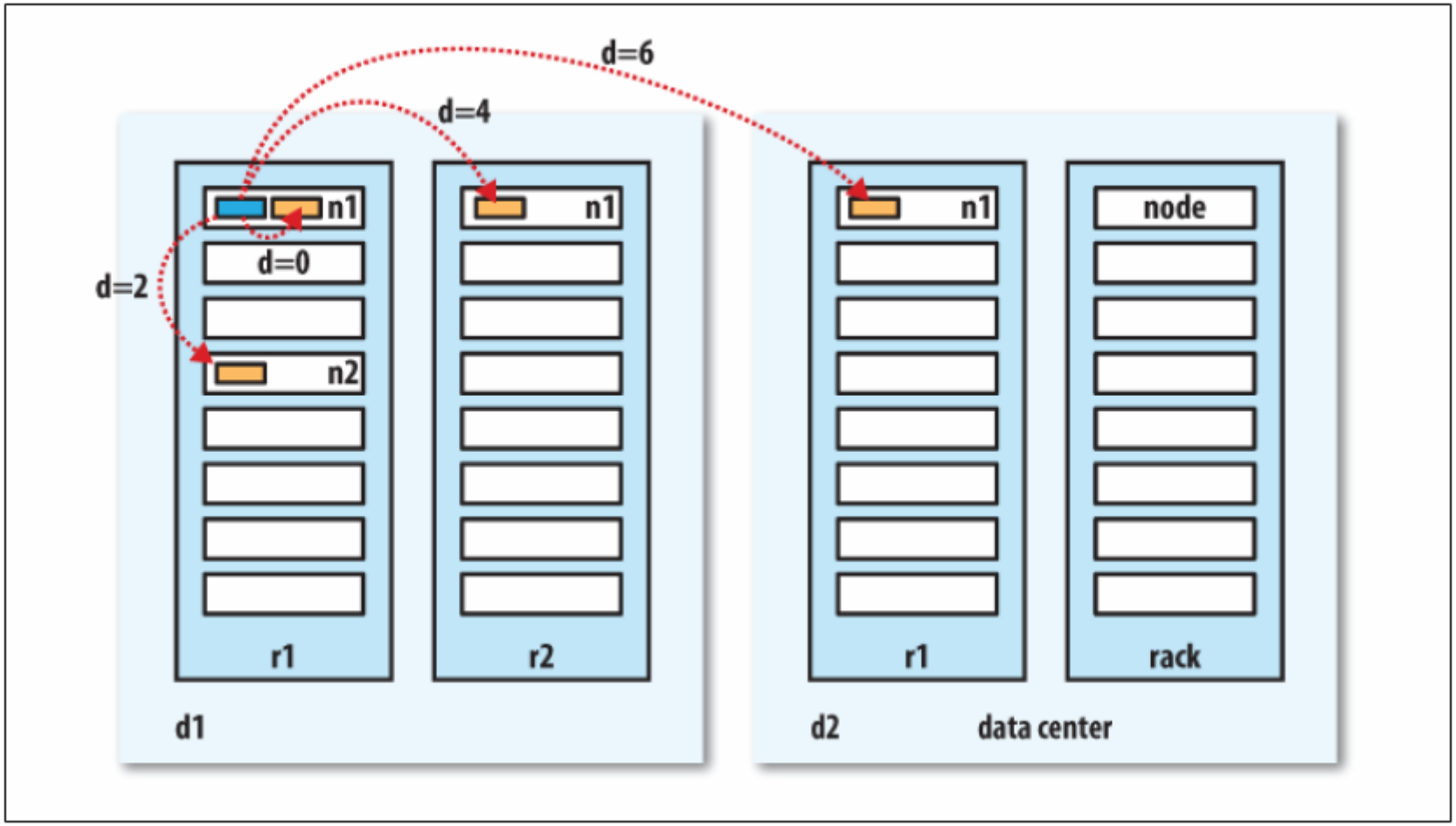
- Store very large files across machines in a large cluster.
- Each file is a sequence of blocks of same size.
- Blocks are replicated 2-3 times.
- Block size and replicas are configurable per file.
- Namenode receives a Heartbeat and a BlockReport from each DataNode in the cluster.
- BlockReport contains all the blocks on a Datanode.

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Replica Placement

- Rack-aware:
  - Goal: improve reliability, availability and network bandwidth utilization
  - Research topic
- Namenode determines the rack id for each DataNode.
- Replicas are placed:
  - 1 in a local rack, 1 on a different node in the local rack and 1 on a node in a different rack.
  - 1/3 of the replica on a node, 2/3 on a rack and 1/3 distributed evenly across remaining racks.

Datanode Distance



Replication Pipelining

- When the client receives response from Namenode
- It flushes its block in small pieces (4K) to the first replica
- that in turn copies it to the next replica and so on.
- Thus data is pipelined from Datanode to the next.

Replica Selection

- Replica selection for READ operation: HDFS tries to minimize the bandwidth consumption and latency.
- If there is a replica on the Reader node then that is preferred.
- HDFS cluster may span multiple data centers: replica in the local data center is preferred over the remote one.

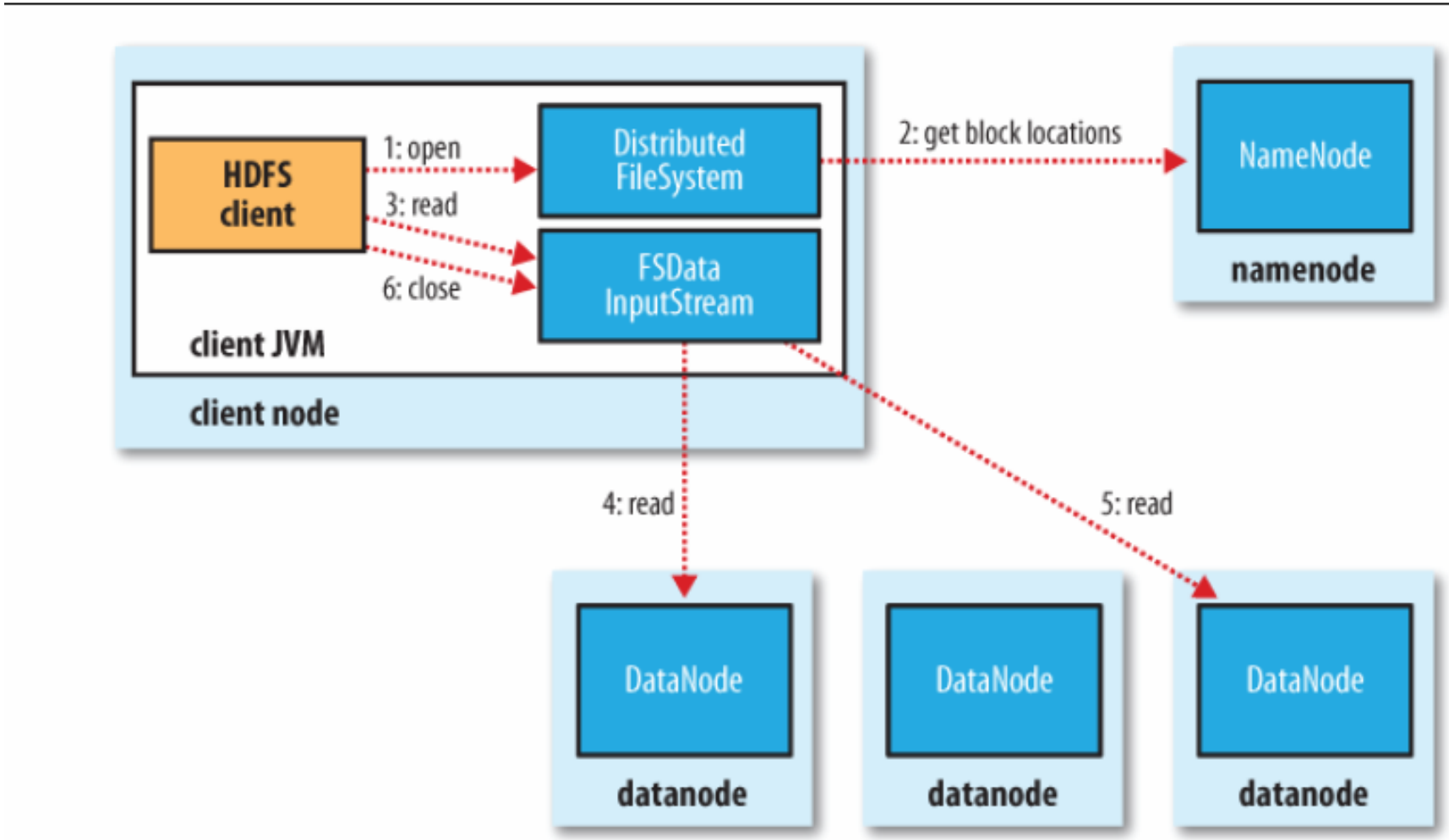
北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Datanode

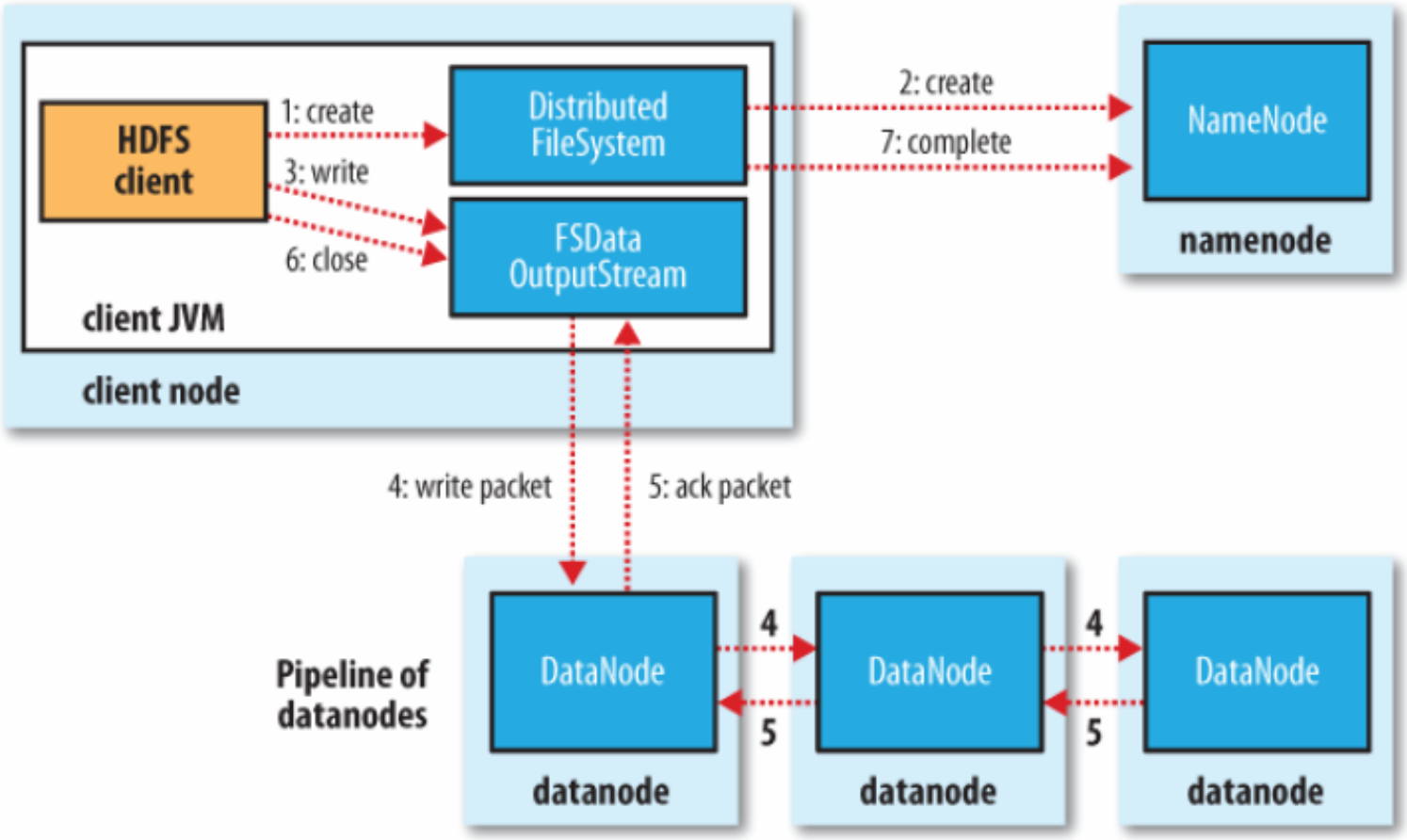
- A Datanode stores data in files in its local file system.
- Datanode has no knowledge about HDFS filesystem
- It stores each block of HDFS data in a separate file.
- Datanode does not create all files in the same directory.
- It uses heuristics to determine optimal number of files per directory and creates directories appropriately:
  - Research issue?
- When the filesystem starts up it generates a list of all HDFS blocks and send this report to Namenode: Blockreport.

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

HDFS: File Read



HDFS: File Write



4

Communication Protocol

- All protocols are layered on top of the TCP/IP protocol
- A client establishes a connection to a configurable TCP port on the Namenode machine. It talks ClientProtocol with the Namenode.
- Datanodes talk to the Namenode using Datanode protocol.
- RPC abstraction wraps both ClientProtocol and Datanode protocol.
- Namenode is simply a server and never initiates a request; it only responds to RPC requests issued by DataNodes or clients.

DataNode Failure and Heartbeat

- Datanodes lose connectivity with Namenode.
- Namenode detects this condition by the absence of a Heartbeat message.
- Namenode marks Datanodes without Hearbeat and does not send any IO requests to them.
- Any data registered to the failed Datanode is not available to the HDFS.

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Cluster Rebalancing

- HDFS architecture is compatible with data rebalancing schemes.
- A scheme might move data from one Datanode to another if the free space on a Datanode falls below a certain threshold.
- In the event of a sudden high demand for a particular file, a scheme might dynamically create additional replicas and rebalance other data in the cluster.
- These types of data rebalancing are not yet implemented: research issue.

APIs

- HDFS provides Java API for application to use.
- Python access is also used in many applications.
- A C language wrapper for Java API is also available.
- A HTTP browser can be used to browse the files of a HDFS instance.

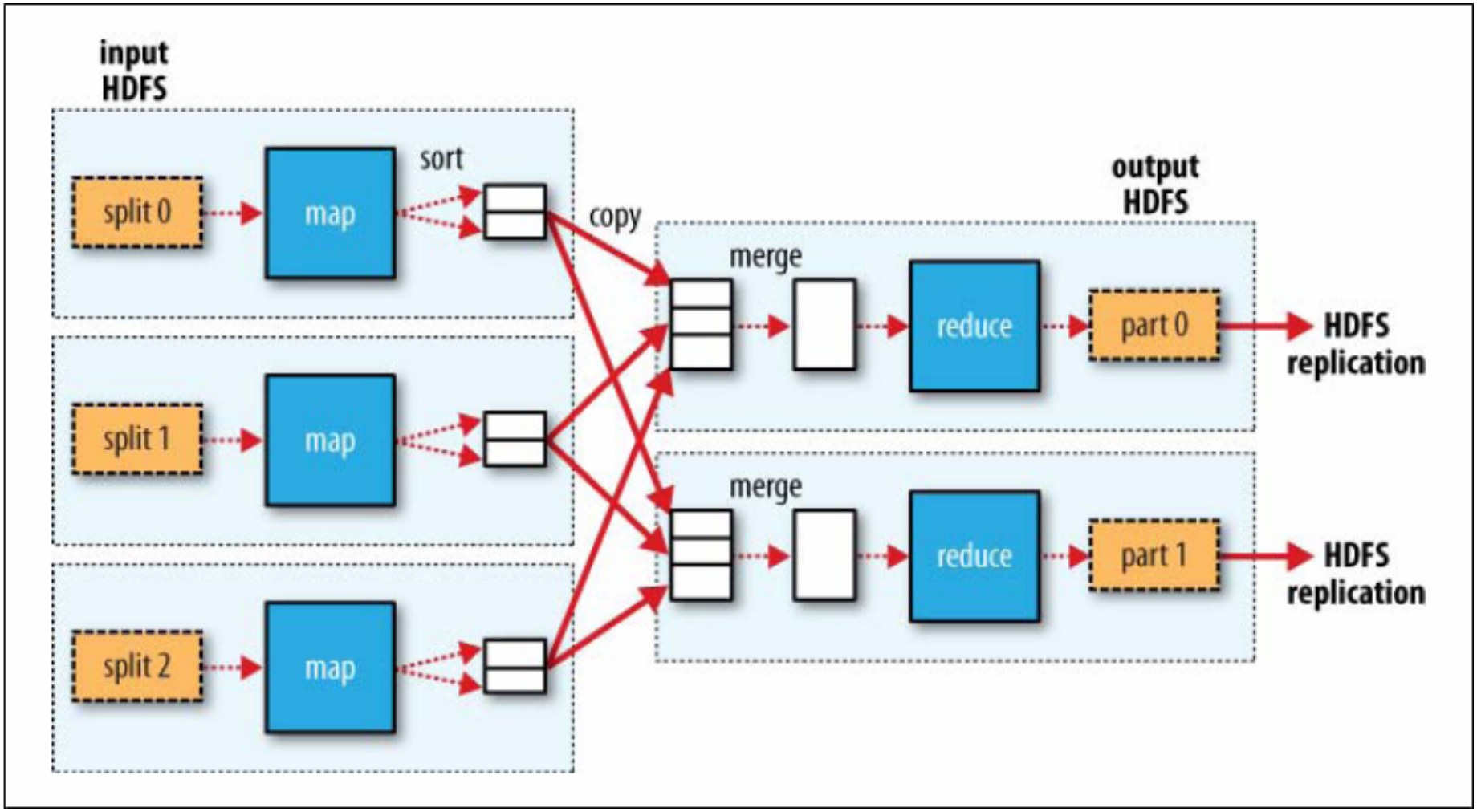
A Distributed Computation Framework for Large Data Set

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

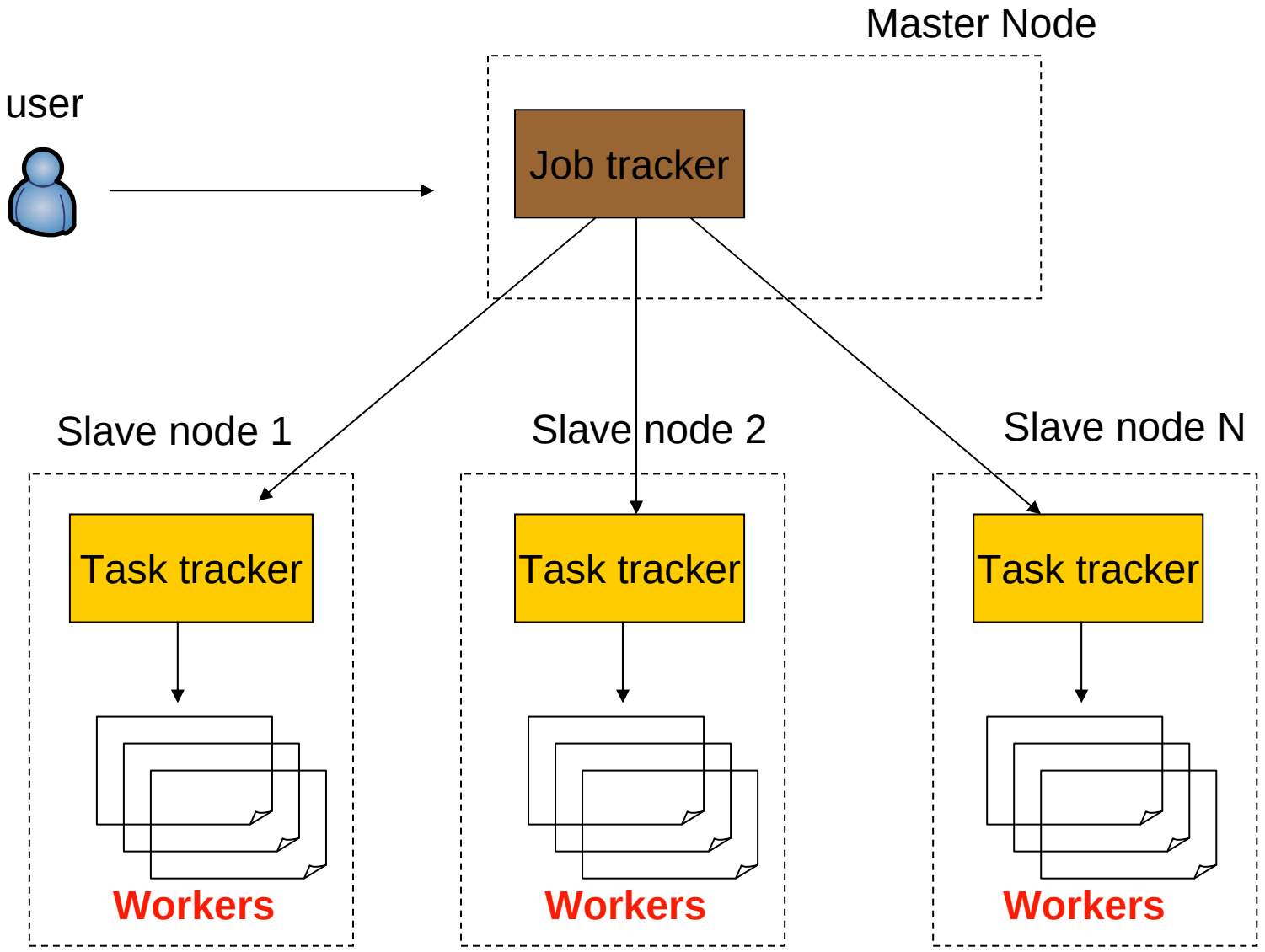
What is Map/Reduce?

- A Programming Model
- Decompose a processing job into Map and Reduce stages
- Developer need to
  - provide code for Map and Reduce functions
  - configure the job
  - let Hadoop handle the rest

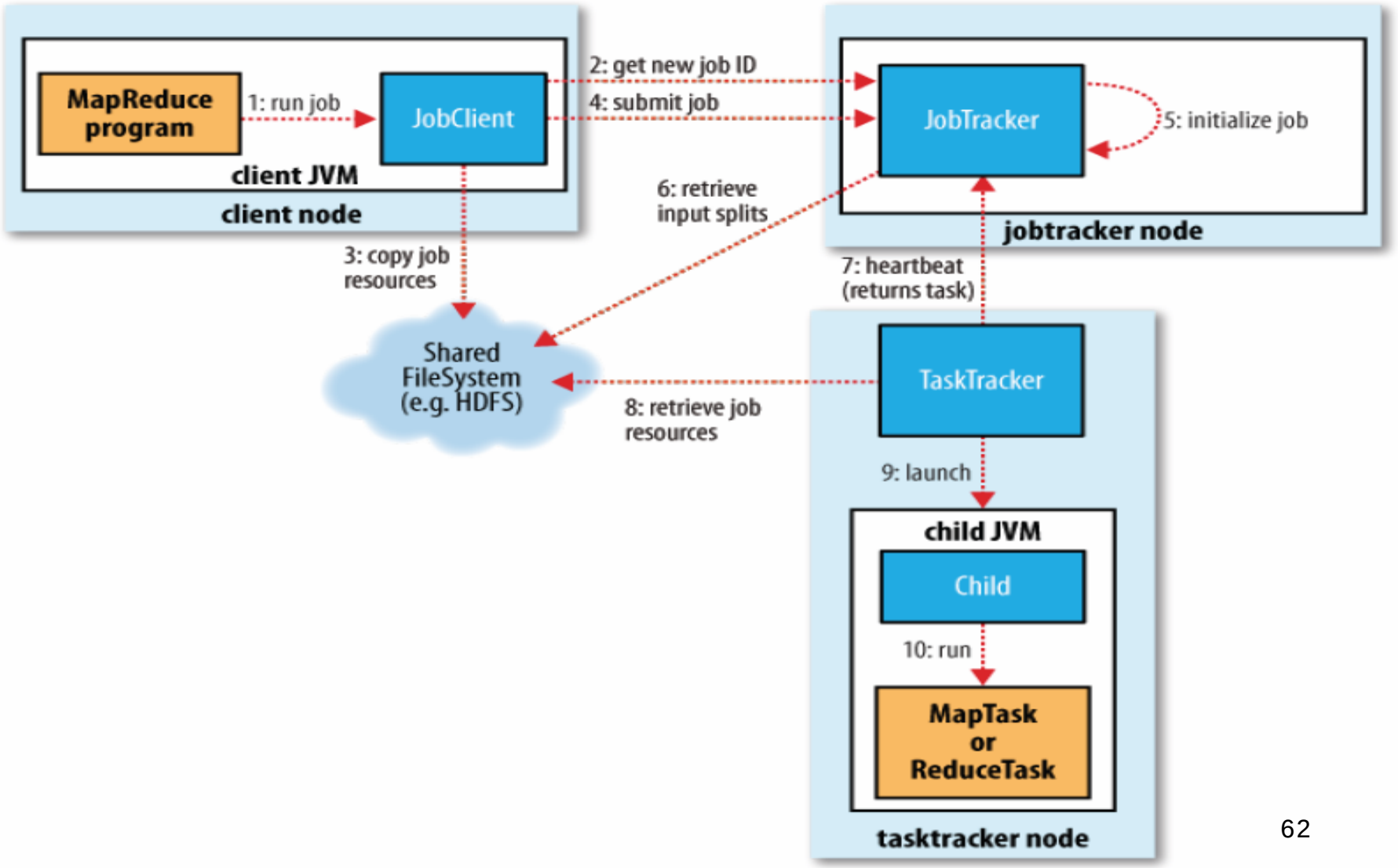
MapReduce Model



Architecture Overview



Inside Hadoop



Example: Word Count

- We have a large file of words, one word to a line
- Count the number of appearances for each distinct word

- *Sample application*: analyze web server logs to find popular URLs

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

MapReduce

- Input: a set of key/value pairs
- User supplies two functions:
  - map(k,v) → list(k1,v1)
  - reduce(k1, list(v1)) → v2
- (k1,v1) is an intermediate key/value pair
- Output is the set of (k1,v2) pairs

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

What is MAP?

- Map each data entry into a pair
  - <key, value>
- Examples
  - Map each log file entry into <URL,1>
  - Map day stock trading record into <STOCK, Price>

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

What is Shuffle/Merge phase?

- Hadoop merges(shuffles) output of the MAP stage into
  - <key, valulue1, value2, value3>
- Examples
  - <URL, 1 ,1 ,1 ,1 ,1 1>
  - <STOCK, Price On day 1, Price On day 2..>

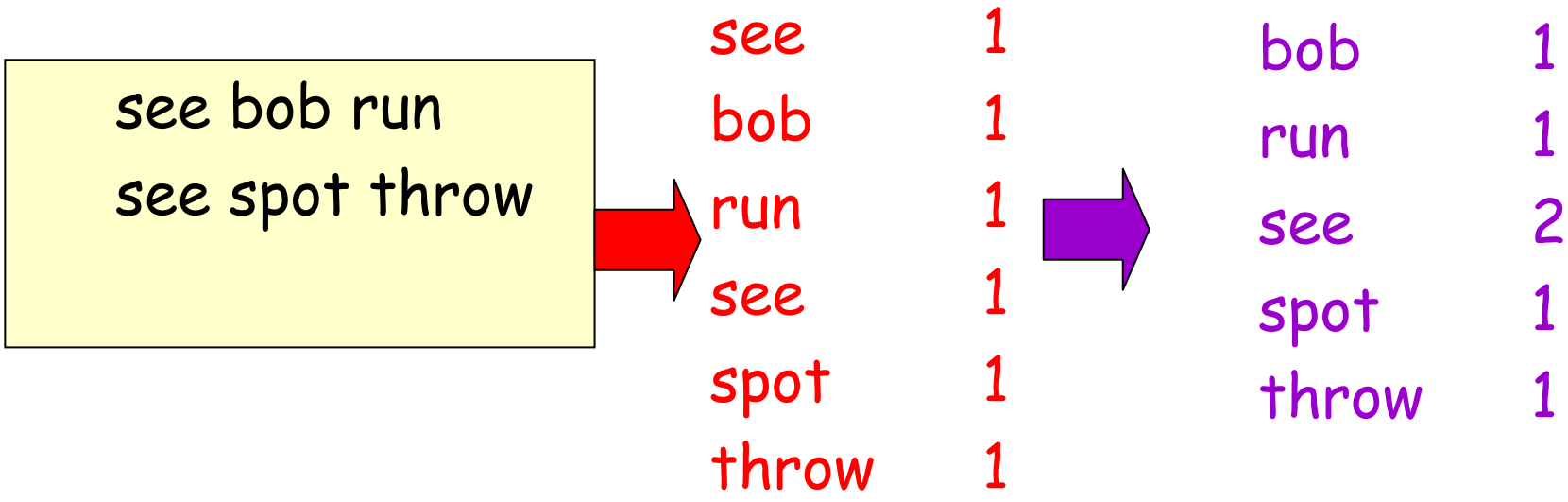
What is Reduce?

- Reduce entries produces by Hadoop merging processing into <key, value> pair

- Examples
  - Map <URL, 1,1,1> into <URL, 3>
  - Map <Stock, 3,2,10> into <Stock, 10>

Word Count

map(key=url, val=contents):
For each word *w* in contents, emit (w, "1")
reduce(key=word, values=uniq_counts):
Sum all "1"s in values list
Emit result "(word, sum)"



北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

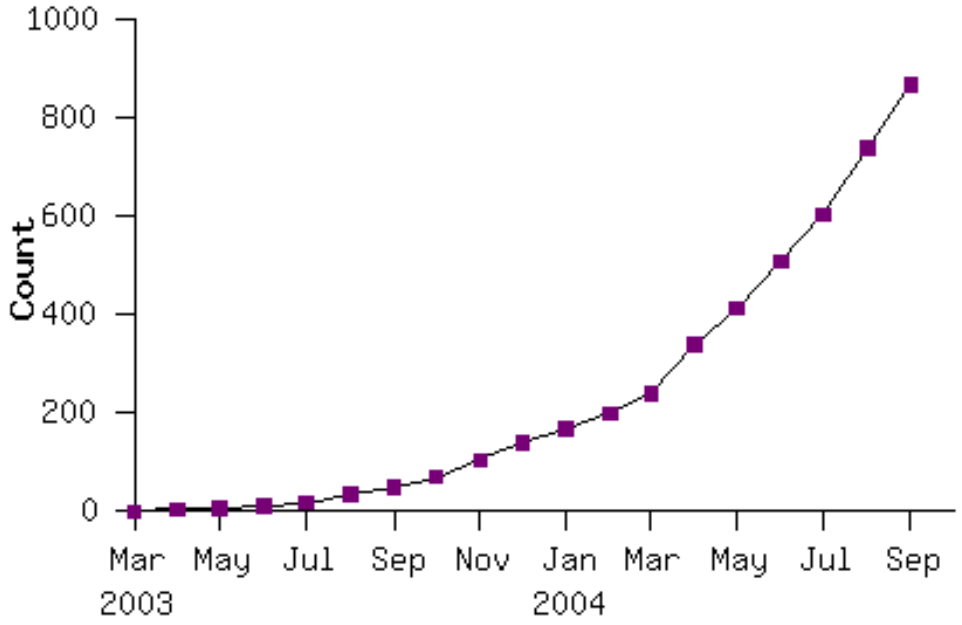
Pseudo-Code: Word Count

```
map(key, value):
// key: document name; value: text of document
    for each word w in value:
        emit(w, 1)

reduce(key, values):
// key: a word; values: an iterator over counts
        result = 0
        for each count v in values:
            result += v
        emit(key,result)
```

Widely Applicable

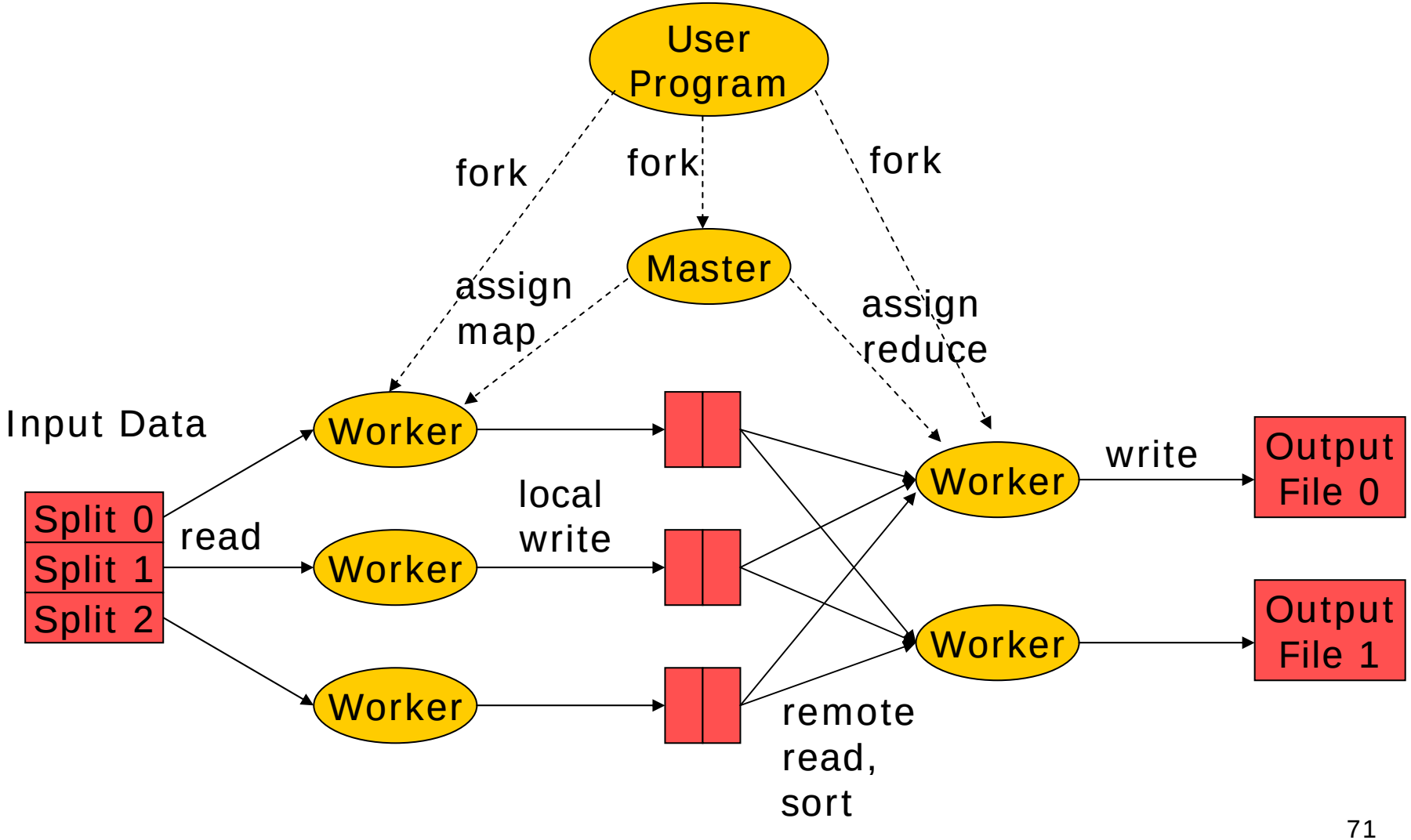
MapReduce Programs in Google Source Tree



Example uses:

distributed grep	distributed sort	web link-graph reversal
term-vector / host	web access log stats	inverted index construction
document clustering	machine learning	statistical machine translation
...	...	...

Distributed Execution Overview



北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Data Flow

- Input, final output are stored on HDFS
  - Scheduler tries to schedule map tasks "close" to physical storage location of input data
- Intermediate results are stored on local FS of map and reduce workers
- Output is often input to another map reduce task

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Coordination

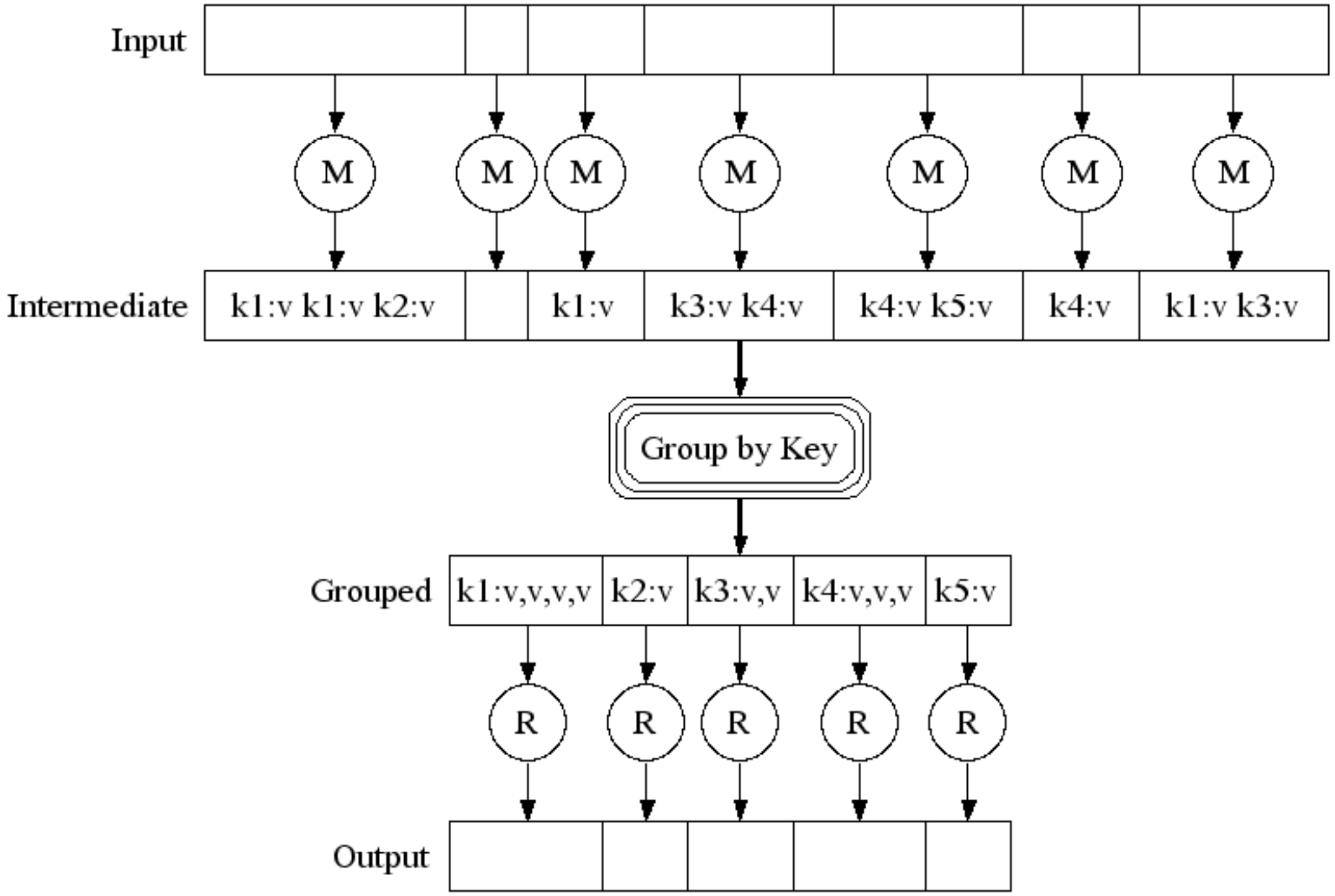
- Master data structures
  - Task status: (idle, in-progress, completed)
  - Idle tasks get scheduled as workers become available
  - When a map task completes, it sends the master the location and sizes of its R intermediate files, one for each reducer
  - Master pushes this info to reducers
- Master pings workers periodically to detect failures

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

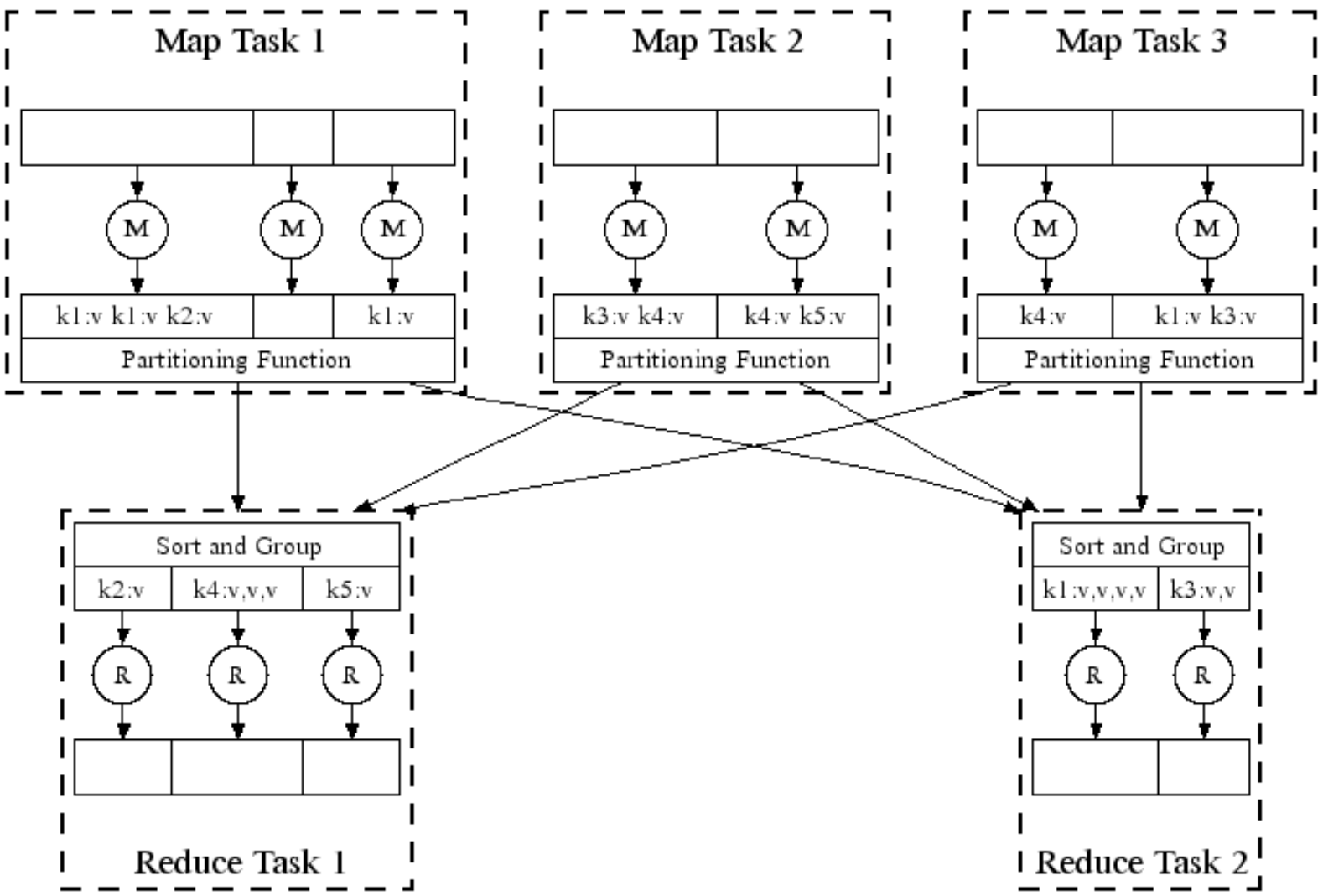
Failures

- Map worker failure
  - Map tasks completed or in-progress at worker are reset to idle
  - Reduce workers are notified when task is rescheduled on another worker
- Reduce worker failure
  - Only in-progress tasks are reset to idle
- Master failure
  - MapReduce task is aborted and client is notified

Execution


Input
M
M
M
M
M
M
M
Intermediate
k1:v k1:v k2:v
k1:v
k3:v k4:v
k4:v k5:v
k4:v
k1:v k3:v
Group by Key
Grouped
k1:v,v,v,v
k2:v
k3:v,v
k4:v,v,v
k5:v
R
R
R
R
R
Output

Parallel Execution



How Many Map and Reduce Jobs?

- M map tasks, R reduce tasks
- Rule of thumb:
  - M, R >> (# of nodes) in cluster
  - One DFS chunk per map is common
  - Improves dynamic load balancing and speeds recovery from worker failure
- Usually R is smaller than M, because output is spread across R files

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Combiners

- Often a map task will produce many pairs of the form (k,v1), (k,v2), … for the same key k
  - e.g., popular words in Word Count
- Can save network time by pre-aggregating at mapper
  - combine(k1, list(v1)) → v2
  - same as reduce function

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Partition Function

- Inputs to map tasks are created by *contiguous* splits of input file
- For reduce, we need to ensure that records with the same intermediate key end up at the same worker
- System can use a default partition function e.g., hash(key) mod R
- Sometimes useful to override
  - e.g., hash(hostname(URL)) mod R ensures URLs from a host end up in the same output file

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Execution Summary

- How is this distributed?
  1. Partition input key/value pairs into chunks, run map() tasks in parallel
  2. After all map()s are complete, consolidate all emitted values for each unique emitted key
  3. Now partition space of output map keys, and run reduce() in parallel
- If map() or reduce() fails, re-execute!

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Example: Trading Data Processing

- Input:
  - Historical Stock Data
  - Records are CSV (comma separated values) text file
  - Each line : stock_symbol, low_price, high_price
  - 1987-2009 data for all stocks one record per stock per day
- Output:
  - Maximum interday delta for each stock

北京理工大學 BEIJING INSTITUTE OF TECHNOLOGY

Map Function: Part I

```
public class StockAnalyzerMapper extends MapReduceBase
    implements Mapper<LongWritable, Text, Text,
        FloatWritable> {

   @Override
   public void map(LongWritable key, Text value,
         OutputCollector<Text, FloatWritable> output,
             Reporter reporter)
         throws IOException {

      String record = value.toString();

      if (record.startsWith("Symbol")) {
         // ignore header row
         return;
      }
```

Map Function: Part II

```
    String[] fields = record.split(",");
    String symbol = fields[0];
    String high = fields[3];
    String low = fields[4];

    float lowValue = Float.parseFloat(low);
    float highValue = Float.parseFloat(high);
    float delta = highValue - lowValue;

    output.collect(new Text(symbol),
                   new FloatWritable(delta));
  }
}
```

Reduce Function

```
public class StockAnalyzerReducer extends MapReduceBase
        implements Reducer<Text, FloatWritable, Text,
            FloatWritable> {

    @Override
    public void reduce(Text key, Iterator<FloatWritable>
        values,
         OutputCollector<Text, FloatWritable> output,
             Reporter reporter)
         throws IOException {

        float maxValue = Float.MIN_VALUE;
        while (values.hasNext()) {
            maxValue = Math.max(maxValue, values.next().get());
        }

        output.collect(key, new FloatWritable(maxValue));
    }
}
```

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Running the Job : Part I

```
public class StockAnalyzerConfig {

   public static void main(String[] args) throws Exception {
      if (args.length != 2) {
         System.err.println("Usage: StockAnalyzerConfig <
             input> <output>");
         System.exit(1);
      }

      JobConf conf = new JobConf(StockAnalyzerConfig.class);
      conf.setJobName("Stock analyzer");

      Path inputPath = new Path(args[0]);
      FileInputFormat.addInputPath(conf, inputPath);

      Path outputPath = new Path(args[1]);
      FileOutputFormat.setOutputPath(conf, outputPath);
```

北京理工大学 BEIJING INSTITUTE OF TECHNOLOGY

Running the Job: Part II

```
        conf.setMapperClass(StockAnalyzerMapper.class);
        conf.setReducerClass(StockAnalyzerReducer.class);

        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(FloatWritable.class);

        JobClient.runJob(conf);
    }
}
```