Enterprise Computing: Hadoop and MapReduce

Markus Klems, Stefan Tai

Wirtschaftsinformatik –
Information Systems Engineering

Agenda

- Hadoop: a framework for distributed computing
- MapReduce programming model
- Hadoop MapReduce implementation
- A quick glance at selected Hadoop components

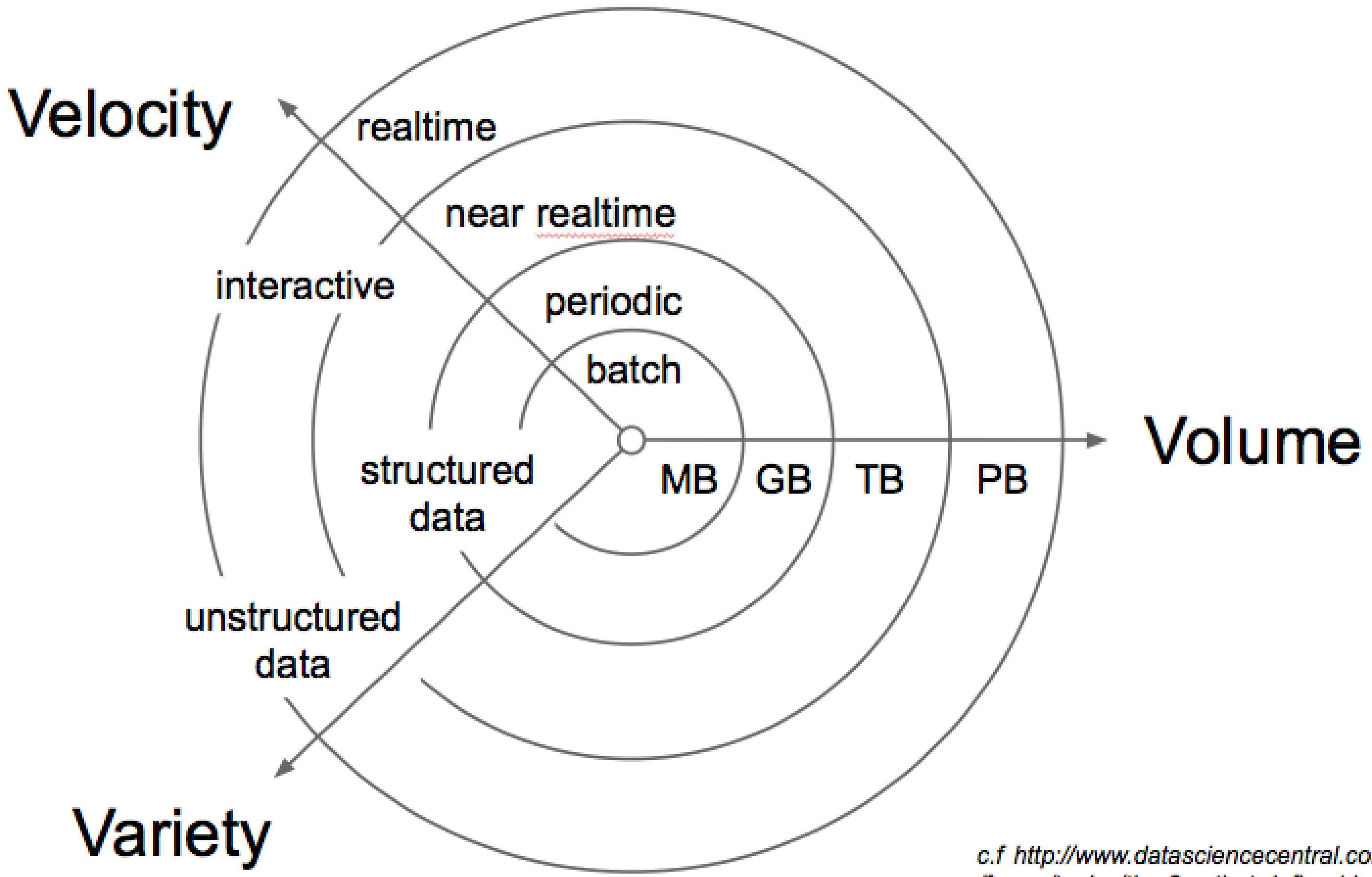
Hadoop

- Hadoop is an open source framework that consists of many components for distributed computing, such as:
  - Hadoop core utilities
  - HBase: A scalable, distributed database
  - HDFS: A distributed file system.
  - Hive: data summarization and ad hoc querying.
  - MapReduce: distributed processing on compute clusters.
  - Pig: A high-level data-flow language for parallel computation.
  - ZooKeeper: coordination service for distributed applications

Hadoop

	Google technology (proprietary)	Apache Hadoop (open source project)
Language	**C++**	**Java**
File System	**GFS**	**HDFS**
Database	**BigTable**	**HBase**
Distributed Coordination Service	**Chubby**	**Zookeeper**

Big Data



c.f http://www.datasciencecentral.com /forum/topics/the-3vs-that-define-big-data

Use Hadoop to manage and process Big Data

- **Store** and **manage** large amounts of unstructured, semi-structured, or structured data
- **Analyze, aggregate, and transform** large amounts of data
  - Server administration: log files
  - Social Media: user engagement (e.g., posts & tweets)
  - Website analytics: logs of clickstreams
  - Data analysis: sensor data, geolocation data, …

Selected Hadoop use cases

Facebook

- 1100 node and 800 node cluster with total of 15PB raw storage
- Internal logs, analytics, and machine learning

Netflix

- Built own hadoop-as-a-service for internal use
- Based on AWS EC2 with web GUIs for scheduling

Rackspace

- 30 node cluster with total of 45TB
- Parse and index logs from email hosting system

Spotify

- 690 node cluster with total of 28 PB, 7500+ jobs daily
- Content generation, data aggregation, and analytics

Hadoop at American Express

Some quotes from Sastry Durvasula and Kevin Murray, VP Technology:

„American Express customers expect us to know them, to understand and anticipate their preferences and personalize our offerings to meet their specific needs. [...] In addition, merchants, small businesses and corporations also want increased value, insights and relevance from our global network.“

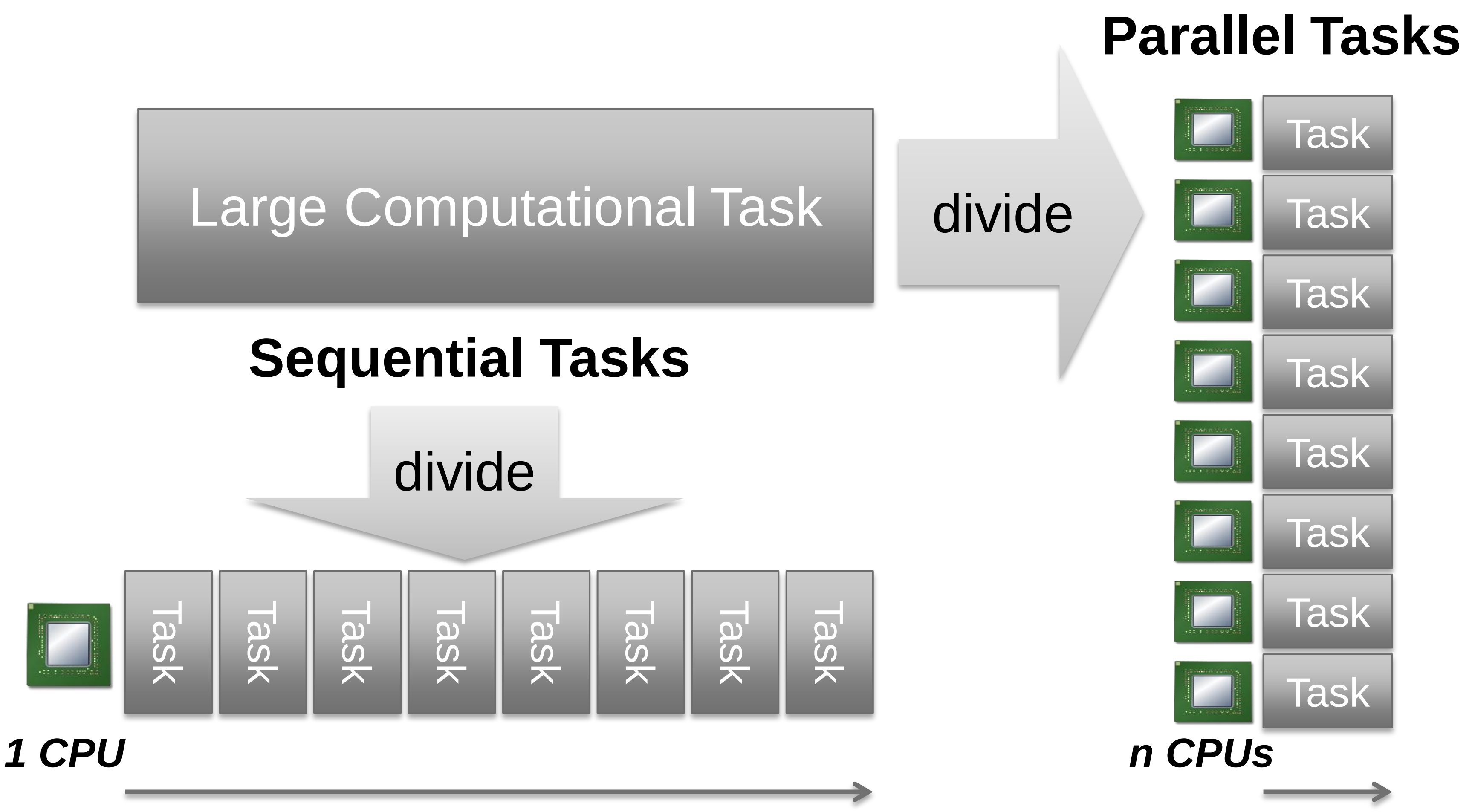
„From a technical perspective, we are advancing an enterprise-wide big data platform that leverages open source technologies like Hadoop, integrating it with our analytical and operational capabilities across the various business lines. This platform also powers strategic partnerships and real-time experiences through emerging digital channels. Examples include Amex Offers, which connects our Card Members and merchants through relevant and personalized digital offers; an innovative partnership with Trip Advisor to unlock exclusive benefits; insights and tools for our B2B partners and small businesses; and advanced credit and fraud risk management.“

“The Hadoop platform indeed provides the ability to efficiently process large-scale data at a price point we haven’t been able to justify with traditional technology. That said, not every technology process requires Hadoop; therefore, we have to be smart about which processes we deploy on Hadoop and which are a better fit for traditional technology (for example, RDBMS).”

Hadoop

- Hadoop is an open source framework that consists of many components:
  - Hadoop core utilities
  - HBase: A scalable, distributed database
  - HDFS: A distributed file system.
  - Hive: data summarization and ad hoc querying.
  - **MapReduce: distributed processing on compute clusters.**
  - Pig: A high-level data-flow language for parallel computation.
  - ZooKeeper: coordination service for distributed applications

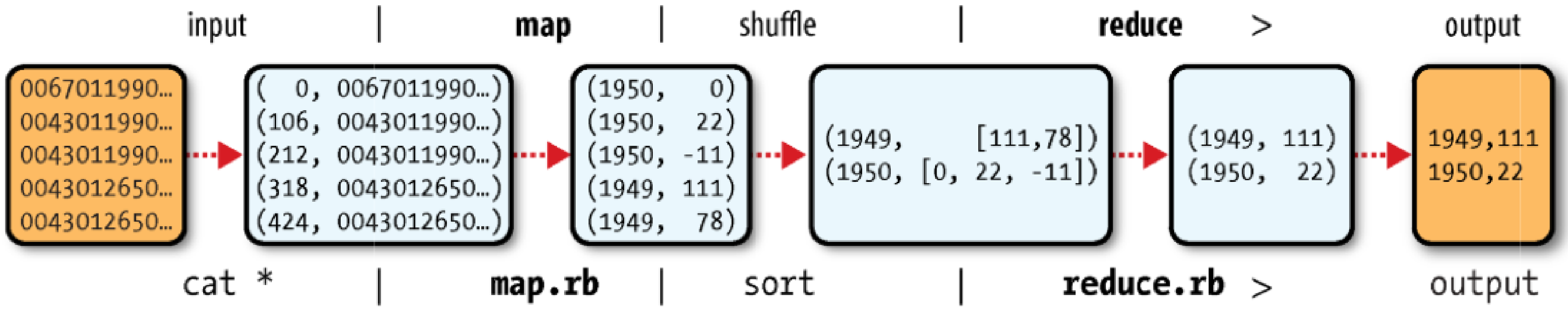
Why Parallel and Distributed Computing?



Map Reduce

- MapReduce is a **Programming Model** to compute large jobs in parallel
- Structure of parallel computations is often similar
  - **Map:** divide problem into subproblems and solve in parallel
  - **Reduce:** combine solutions of subproblems to aggregated solution
- The MapReduce programming model provides a way to express large problems in parallel computations

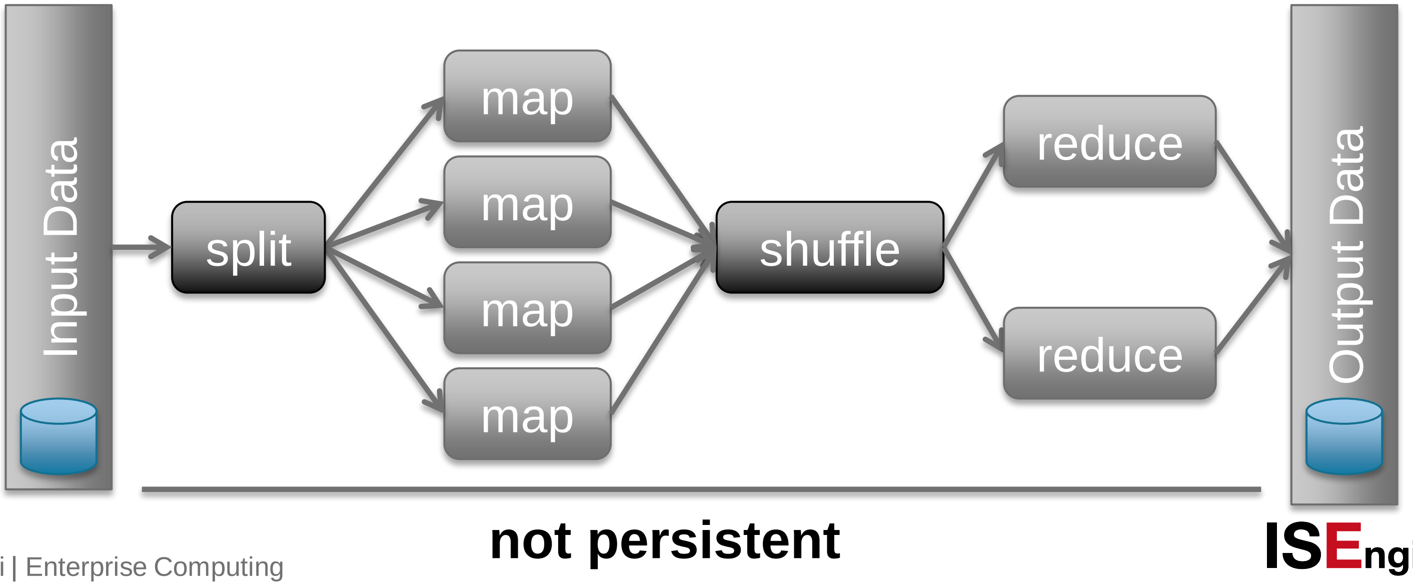
Weather Data Example: Max Temperature



Source: Hadoop – The Definitive Guide, O'Reilly

Map Reduce

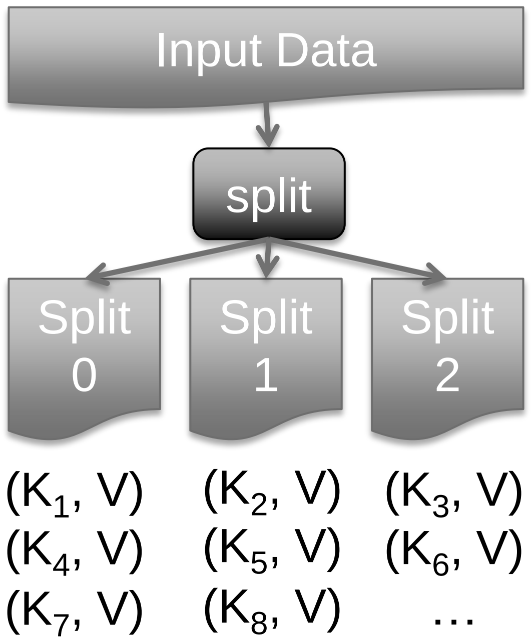
- Many large computation problems can be divided into parallel subproblems
  - Find a **map function** that solves subproblems
  - Find a **reduce function** that aggregates solutions

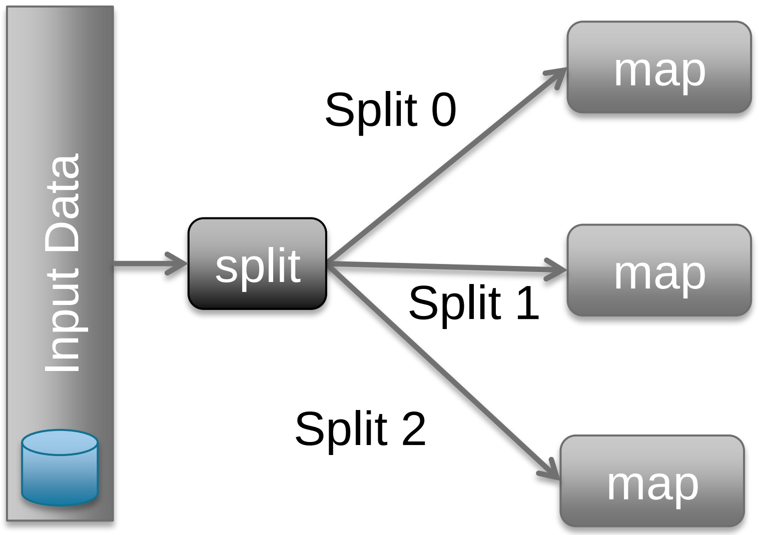


ISEngineering
Wirtschaftsinformatik –
Information Systems Engineering

Map Reduce: Split Function

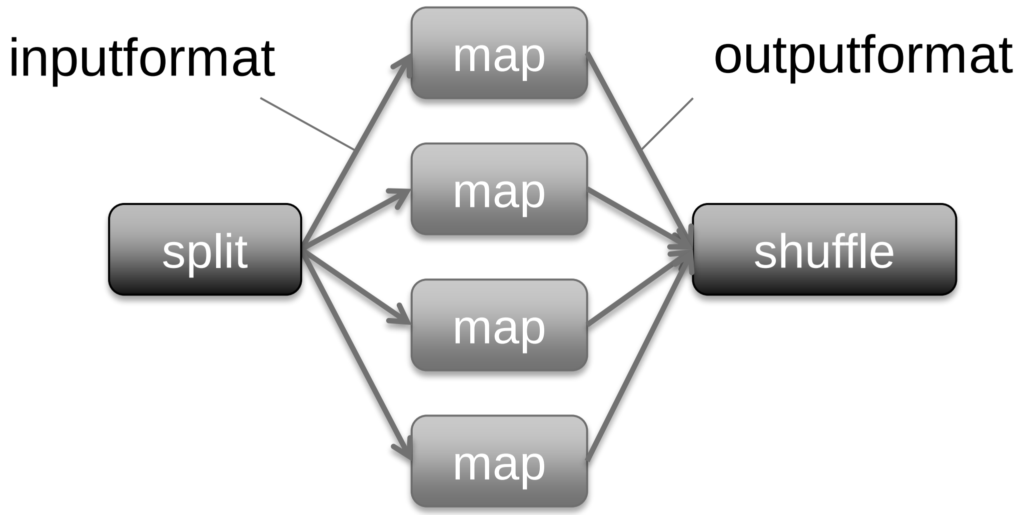
- Read data from storage and transfrom to (key, value) pairs
- Divide set of (key, value) pairs into splits
- Assign splits to map functions





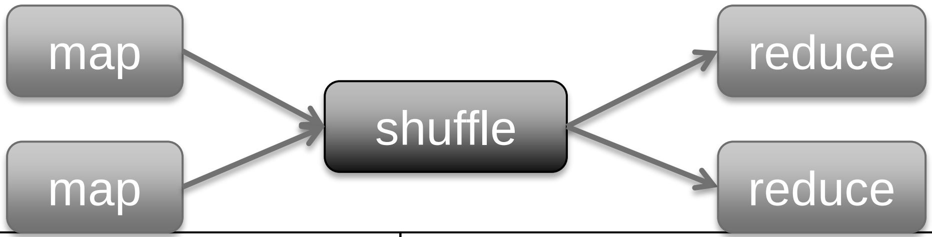
Map Reduce: Map Function

map(key$_i$, value$_i$) -> (key$_j$, value$_j$)



Map Reduce: Shuffle and Sort

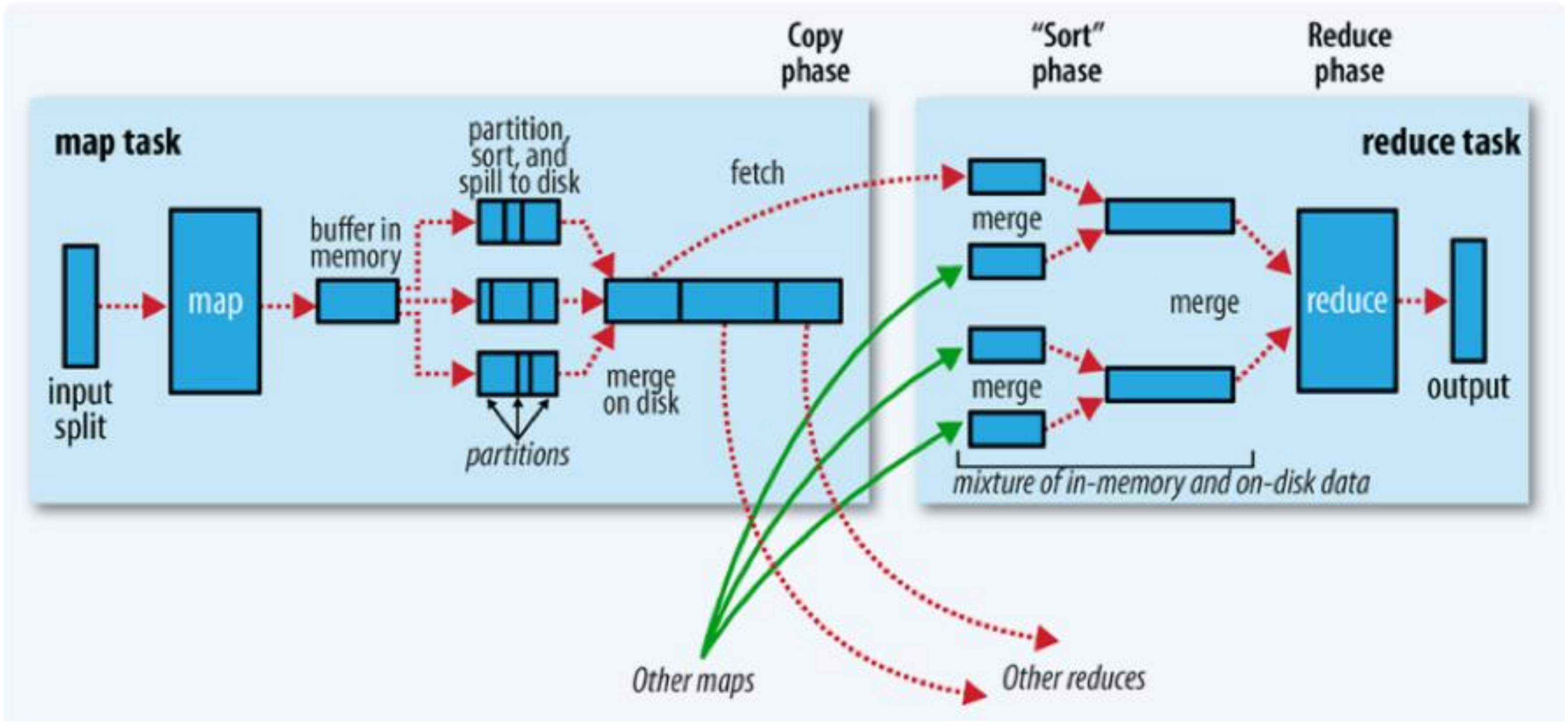
- MapReduce makes the guarantee that the input to every reducer is sorted by key



• Write map output to circular memory buffer (100 MB by default) and perform in-memory sort by key • At 80% buffer capacity, a background thread starts spilling data on local disk • If the buffer is full, the map tasks block until spilling is complete	• The reduce tasks need map output from different map tasks, likely located on other servers • The reduce tasks start copying the map output files as soon as they become available • The map outputs are buffered in memory and merged to disk-resident files

Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

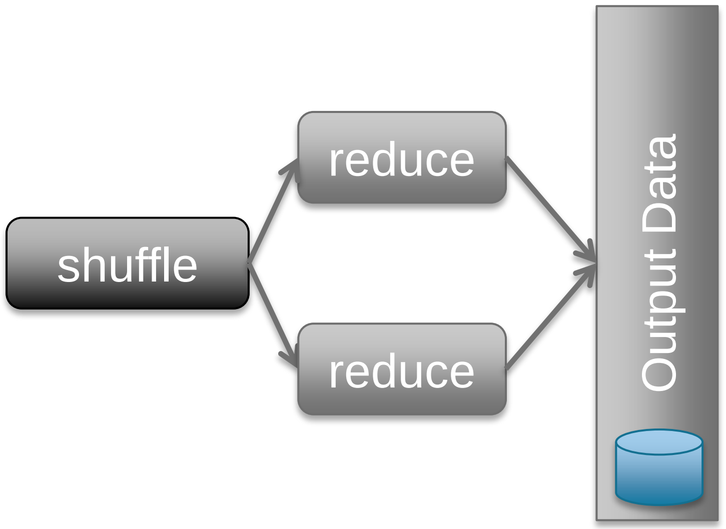
Map Reduce: Shuffle and Sort



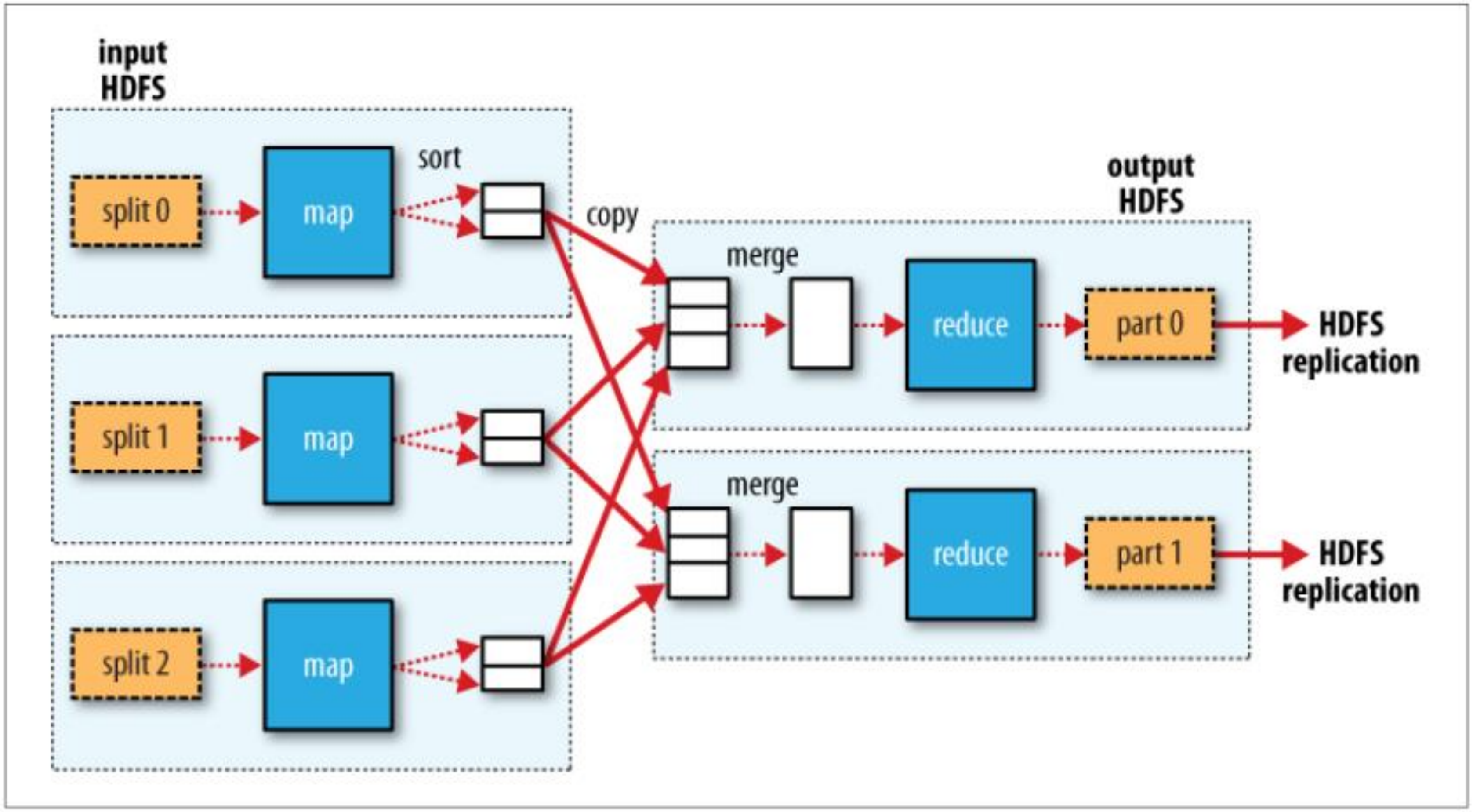
Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

Map Reduce: Reduce Functions

- Function to aggregate the sorted map phase output
- Write reduce output to a distributed file system (persistent)

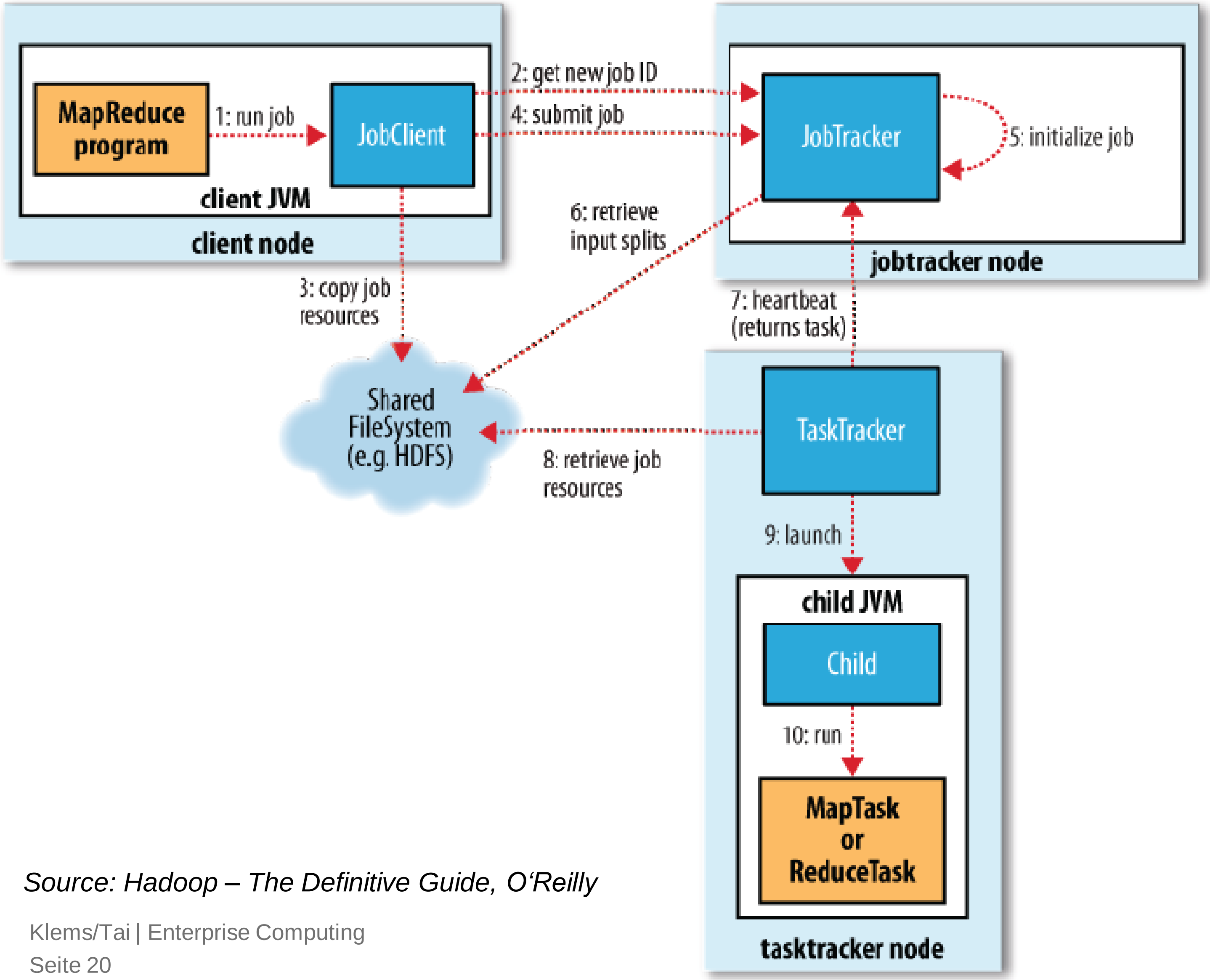


Map Reduce data flow with multiple reducers



Source: Hadoop – The Definitive Guide, O‘Reilly

Apache Hadoop implementation of Map Reduce



Source: Hadoop – The Definitive Guide, O'Reilly

Apache Hadoop implementation of Map Reduce

- The **JobClient** submits the MapReduce job
- The **jobtracker** schedules and coordinates the job run
- The **tasktrackers** launch the tasks that the job has been split into
- Individual worker tasks run **Map** and **Reduce** tasks
- The **distributed file system** (typically HDFS) is used for sharing job files and data locality
  - HDFS blocks are usually very large (64MB or more)
  - The optimal split size is equal to the HDFS block size since map tasks get their input from their local server or from a single server close to it in terms of network connection

Anatomy of a Map Reduce run: Job submission

- The `runJob()` method on `JobClient` creates a new `JobClient` instance and calls `submitJob()` on it:
  - Ask the jobtracker for a new job ID
  - Check if output path exists
  - Check input path and create input splits
  - Copy the resources needed by the job, including the jar file
- After job submission, `runJob()` polls the job's progress and reports progress changes on the console until the job has either succeeded or failed

Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

Anatomy of a Map Reduce run: Job initialization

- When the JobTracker receives a call to its `submitJob()` method, it puts it into an internal queue from where the job scheduler will pick it up and initialize it
- Initialization involves creating an object to represent the job being run, which encapsulates tasks, and other information to track the tasks' status and progress
- The job scheduler creates a Map task for each input split and a configurable number of Reduce tasks

Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

Anatomy of a Map Reduce run: Task assignment

- Tasktrackers run a loop that periodically sends heartbeat messages to the jobtracker
  - Tell jobtracker that tasktracker is alive
  - Piggyback messages that indicate whether it is ready for a new task
- If a tasktracker reports that it is ready for a new task, the jobtracker selects a job (according to the job scheduler) and then selects a task from the job
- Each tasktracker has a limited slot for Map and Reduce tasks which should be configured to reflect the resource capacity of its server (CPU cores and memory)

Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

Anatomy of a Map Reduce run: Task execution

- When the tasktracker has been assigned a task, it then executes the task
  - First, it localizes the job JAR file by copying it from HDFS to the taskstracker's filesystem
  - Second, the JAR is extracted into a local directory
  - Third, it creates a `TaskRunner` instance to run the task. TaskRunner creates a new Java Virtual Machine (JVM) to run each task in (for fault isolation)

Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

Anatomy of a Map Reduce run: Progress and Status Updates

- MapReduce jobs are long-running batch jobs, taking a few minutes or even hours
- Therefore, it is important that the user gets periodic progress and status updates
  - Status: e.g., “running”, “successfully completed”, “failed”
  - Progress: the proportion of tasks completed
  - Map task progress: proportion of input that has been processed
  - Reduce task progress: more complex calculation of the shuffle phases’ progress, i.e., “copy”, “sort”, and finally “reduce”

Source: Hadoop – The Definitive Guide, O‘Reilly, Chapter 6

Anatomy of a Map Reduce run: Job completion and failures

- When a jobtracker receives a notification that the last task for a job is complete, it changes the job to "successful"
- Failures
  - **Task failure**: e.g., a runtime exception in a map or reduce task. If this happens, the task's child JVM reports the error back to its parent tasktracker, before it exits.
  - **Tasktracker failure**: If the tasktracker runs very slowly or crashes, it does not send any more heartbeat messages to the jobtracker (or too infrequently or late). The jobscheduler will stop sending tasks to the failed tasktracker.
  - **Jobtracker failure**: single point of failure

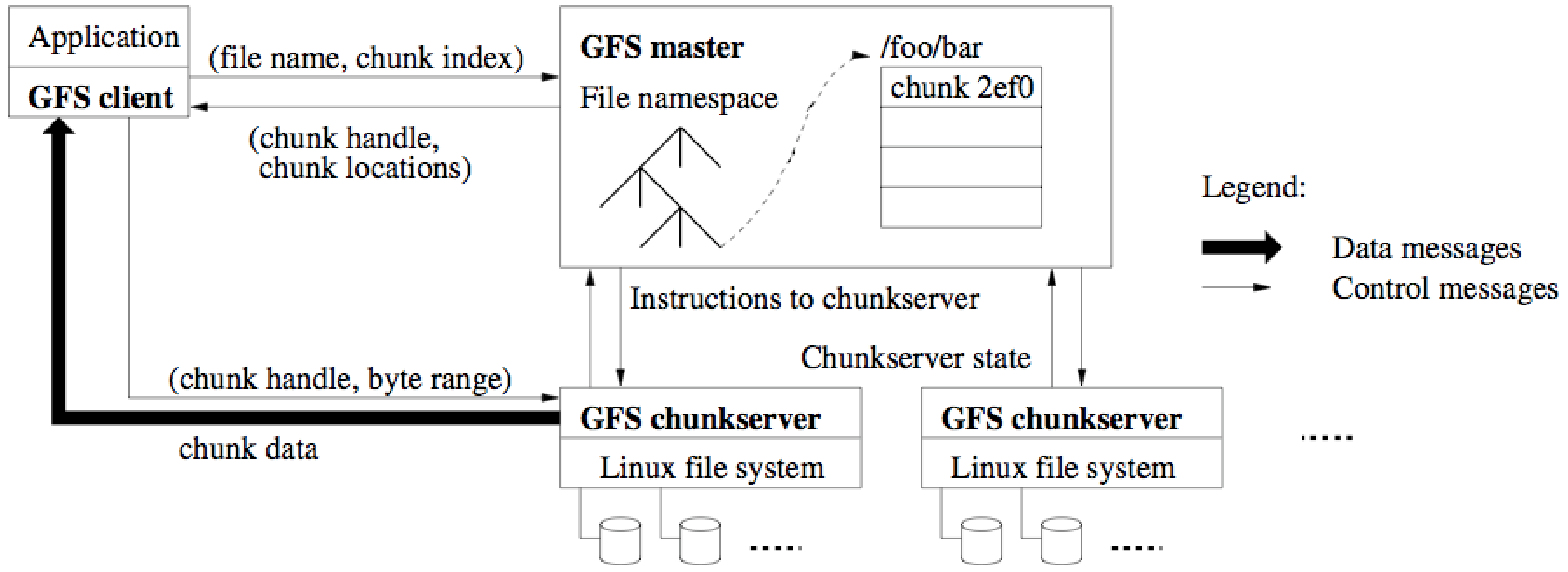
Source: Hadoop – The Definitive Guide, O'Reilly, Chapter 6

A quick glance at: Hadoop Distributed File System (HDFS)

- Hadoop is an open source framework that consists of many components:
  - – Hadoop core utilities
  - – HBase: A scalable, distributed database
  - – **HDFS: A distributed file system.**
  - – Hive: data summarization and ad hoc querying.
  - – MapReduce: distributed processing on compute clusters.
  - – Pig: A high-level data-flow language for parallel computation.
  - – ZooKeeper: coordination service for distributed applications

A quick glance at: Hadoop Distributed File System (HDFS)

- HDFS is the open source implementation of Google File System (GFS)



… more details in the lecture on Dec 8th “GFS/BigTable”

Source: Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung: “The Google File System“. SOSP 2003.

A quick glance at:
Hadoop Distributed File System (HDFS)

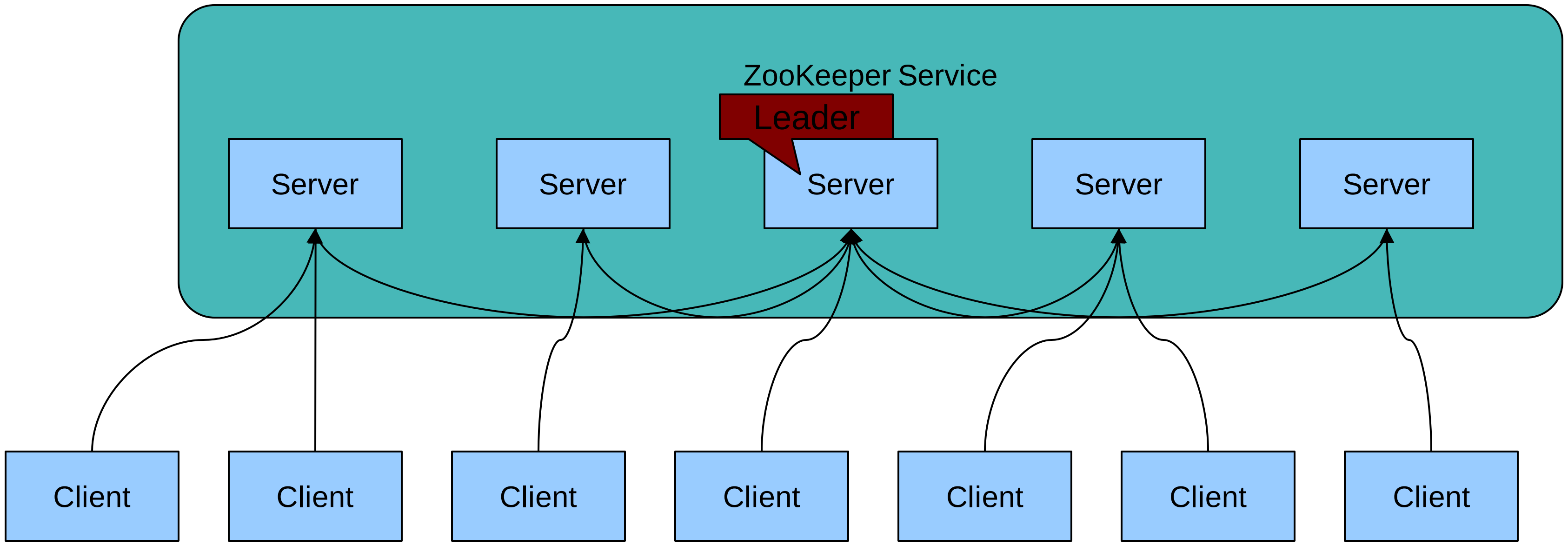
- HDFS **Namenode** (=~ GFS Master)
  - Records all global meta data (File names, block-ids etc.)
  - Datanodes register at Namenode

- Data is stored on HDFS **Datanodes** (=~ GFS chunkservers)
  - Default block size: 64 MB
  - Default replication factor: 3

A quick glance at: ZooKeeper

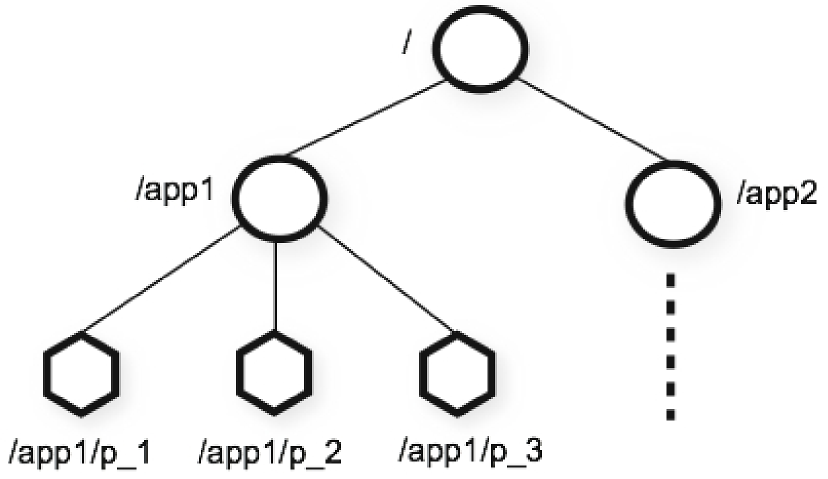
- Hadoop is an open source framework that consists of many components:
  - – Hadoop core utilities
  - – HBase: A scalable, distributed database
  - – HDFS: A distributed file system.
  - – Hive: data summarization and ad hoc querying.
  - – MapReduce: distributed processing on compute clusters.
  - – Pig: A high-level data-flow language for parallel computation.
  - – **ZooKeeper: coordination service for distributed applications**

A quick glance at: ZooKeeper - System



- All servers store a copy of the data (in memory)
- A leader is elected at startup
- Followers service clients, all updates go through leader
- Update responses are sent when a majority of servers have persisted the change

A quick glance at: ZooKeeper – Data Model



- Data hierarchy for cluster coordination, status info, configs, etc.
- Access to ZNodes (zookeeper nodes) via paths, e.g., */app1/p_2*
- ZNodes can have *watches*, and *sequential numbers* and can be *ephemeral*

A quick glance at: ZooKeeper – Lock example

1. Add child node to a lock ZNode in data hierarchy
    - Sequential #
    - Ephemeral
2. Read all child ZNodes
3. Cluster node that created first child (lowest #) has lock
4. Other cluster nodes (higher #)
    - Watch existence of created # -1
    - If not exists (anymore) go to 2.

Unlock: Delete child ZNode

A quick glance at: ZooKeeper – Barrier example

1. Create barrier ZNode when cluster nodes starts their process

2. Waiting cluster nodes set watch on barrier Znode

3. Upon trigger and barrier ZNode does not exist, proceed!

A quick glance at: Pig

- Hadoop is an open source framework that consists of many components:
  - – Hadoop core utilities
  - – HBase: A scalable, distributed database
  - – HDFS: A distributed file system.
  - – Hive: data summarization and ad hoc querying.
  - – MapReduce: distributed processing on compute clusters.
  - – **Pig: A high-level data-flow language for parallel computation.**
  - – ZooKeeper: coordination service for distributed applications

A quick glance at: Pig

- Goal: A high level language to express data analysis programs
  - Simple **programming language “Pig latin”**
  - Reduce **maintenance overhead** for MapReduce programs
  - Exploitation of **optimization** opportunities
  - Extensibility through **user functions**
- At Pig’s core is a parser + compiler converting scripts to MapReduce jobs
- Interactive shell or batch mode (.pig scripts)

A quick glance at: Pig

- Important Expressions in Pig Latin
  - LOAD: load data from files
  - FOREACH: iterate over data items
  - GROUP: aggregate with a rule
  - FILTER: extract certain information
  - STORE/DUMP: output data to file/console
- Lines can be assignments or commands
  - A = LOAD ...
  - STORE ...
- Pig Latin includes arithemtic and boolean expressions

A quick glance at: Hive

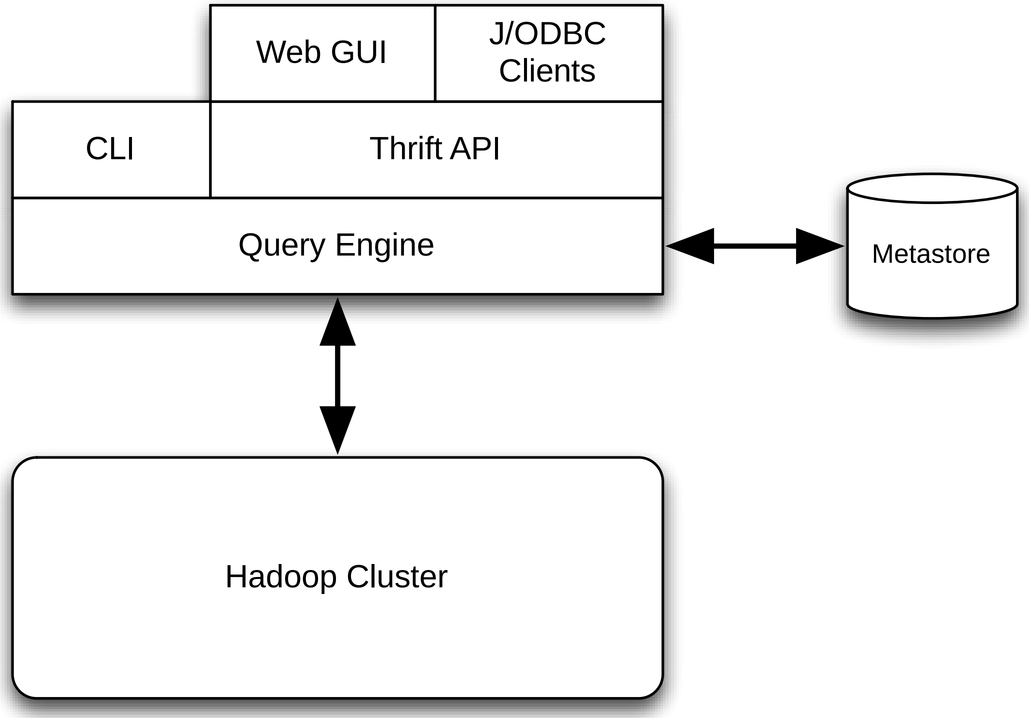
- Hadoop is an open source framework that consists of many components:
  - Hadoop core utilities
  - HBase: A scalable, distributed database
  - HDFS: A distributed file system.
  - **Hive: data summarization and ad hoc querying.**
  - MapReduce: distributed processing on compute clusters.
  - Pig: A high-level data-flow language for parallel computation.
  - ZooKeeper: coordination service for distributed applications

A quick glance at: Hive

- Built on top of Apache Hadoop, Hive provides
  - Tools to enable easy data extract/transform/load (ETL)
  - A mechanism to impose structure on a variety of data formats
  - Access to files stored either directly in Apache HDFS or in other data storage systems such as Apache HBase
  - Query execution via MapReduce
- In a nutshell: “SQL for Hadoop”, or “Hadoop with structured Data”
- Command-line, web (Port 9999), and ODBC interfaces
- Facebook developed Hive for data warehousing

A quick glance at: Hive - Architecture

- At the core of Hive is a query engine that ...
  - maps tables to HDFS files
  - transforms SQL statements into MapReduce Jobs



A quick glance at: Hive – DDL/DML/SQL

- **DDL:**
  CREATE TABLE, DROP, ALTER, DESCRIBE
  SHOW TABLES

- **DML:**
  LOAD DATA (LOCAL) INPATH ... (OVERWRITE) INTO TABLE ... PARTITION

- INSERT, UPDATE, DELETE

- SELECT ... FROM ...

A quick glance at: HBase

- Hadoop is an open source framework that consists of many components:
  - Hadoop core utilities
  - **HBase: A scalable, distributed database**
  - HDFS: A distributed file system.
  - Hive: data summarization and ad hoc querying.
  - MapReduce: distributed processing on compute clusters.
  - Pig: A high-level data-flow language for parallel computation.
  - ZooKeeper: coordination service for distributed applications

A quick glance at: HBase

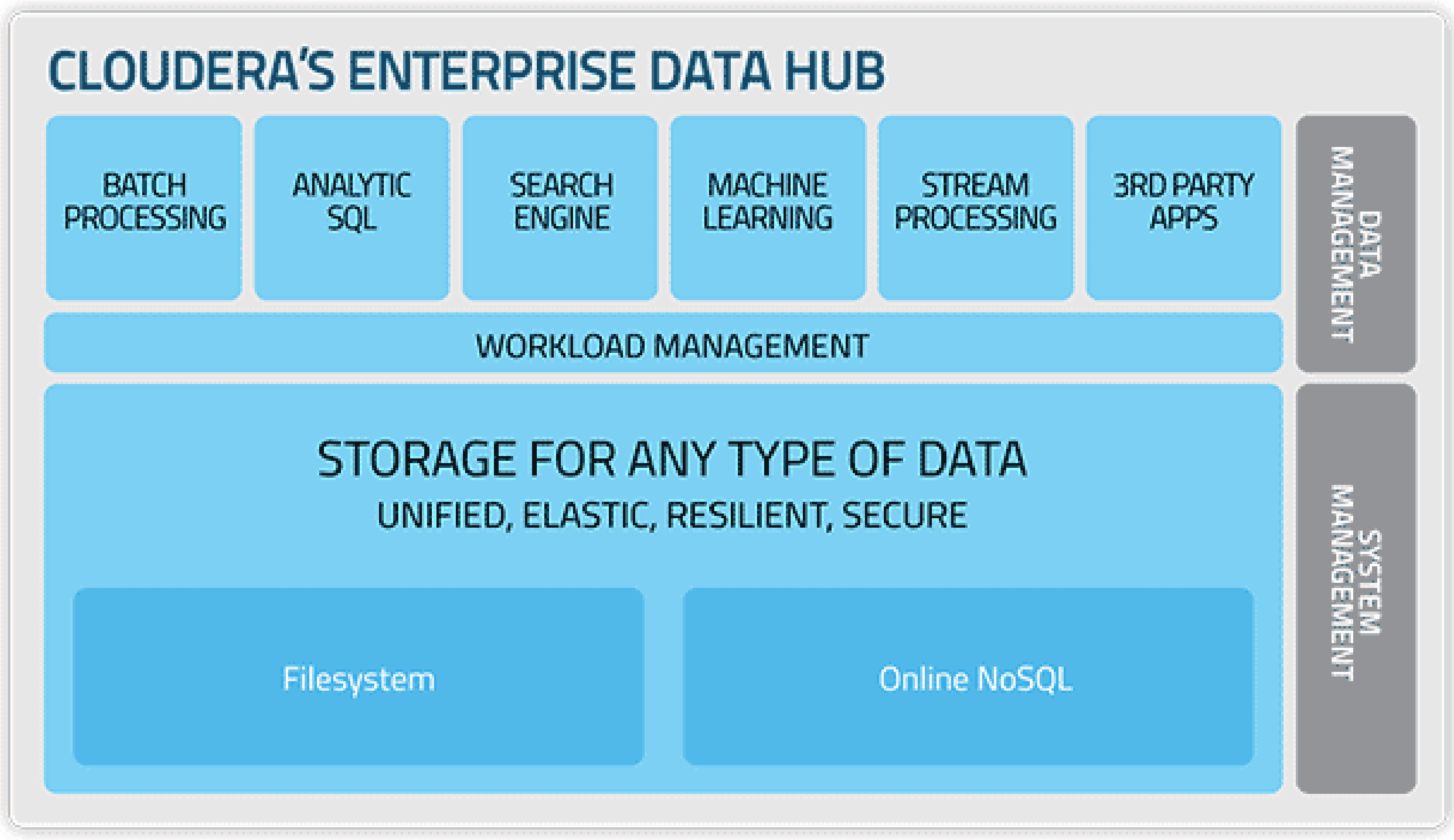
- HBase is a distributed NoSQL database that can scale horizontally to 1,000s of commodity servers and petabytes of indexed storage
- Designed to operate on top of the Hadoop distributed file system (HDFS)
- Implements the architecture and data model of Google BigTable

… more details in the lecture on Dec 8th "GFS/BigTable"

Commercial Hadoop Offerings

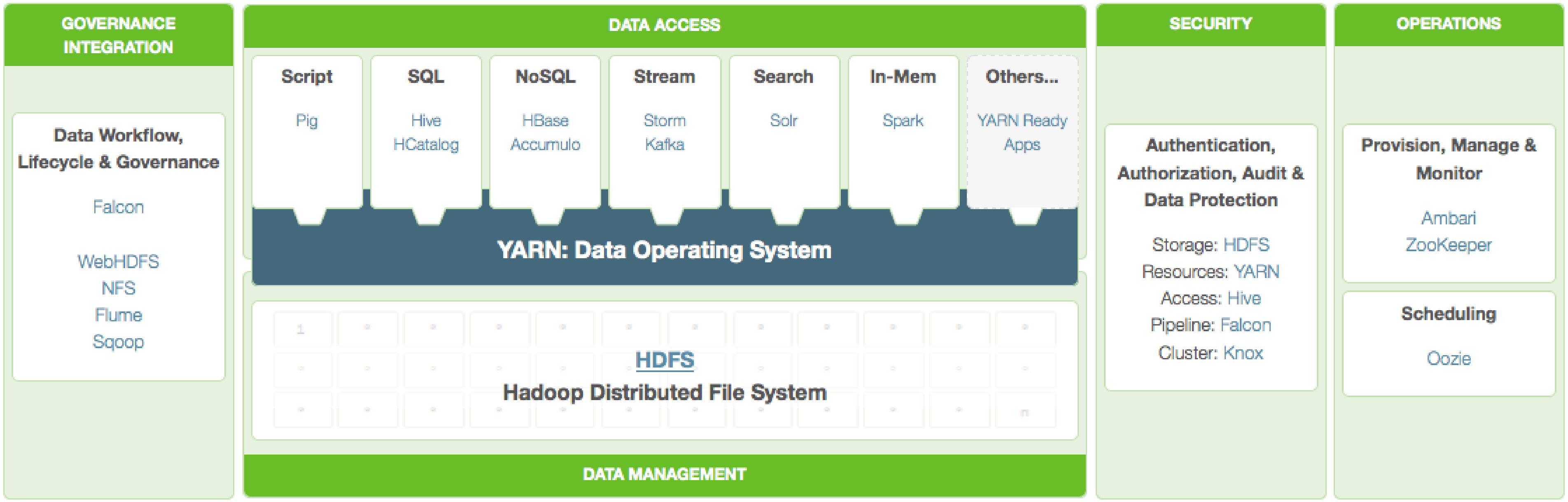
- Cloudera Enterprise
  - Major contributor to open source Apache Hadoop project
- MapR
  - Hadoop solutions implemented in C++ instead of Java with altered architecture
- Hortonworks Enterprise Hadoop
  - Founded in 2011 by Yahoo and Benchmark Capital

Cloudera‘s Enterprise Data Hub (CDH)



Source: http://www.cloudera.com

Hortonworks Enterprise Hadoop Data Platform



Source: http://hortonworks.com

Cloud-based Hadoop Offerings

- Apache Hadoop on Google Cloud Platform
  - MapReduce, Pig, Hive, etc. are supported
  - Direct access to Google Cloud Storage, Google BigQuery, Google Cloud Datastore

- AWS MapReduce Cloud Service
  - Supports multiple programming languages
  - Deploys on Amazon EC2 instances

the lecture is over

enjoy the exercise

the 1
lecture 1
is 1
over 1
enjoy 1
the 1
exercise 1

the 2
lecture 1
is 1
over 1
enjoy 1
exercise 1