EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Advanced Database Systems

MapReduce

Strategic project of TBU in Zlín, reg. no. CZ.02.2.69/0.0/0.0/16_015/0002204

Zdenka Prokopova
TBU in Zlín

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Content

- Distributed data processing
- The basic principle of MapReduce
  - Map and Reduce functions
  - Google File System (GFS)
  - MapReduce: Schema, Example, MapReduce Framework
- Apache Hadoop
  - Hadoop Modules and Related Projects
  - Hadoop Distributed File System (HDFS)
  - Hadoop MapReduce
- MapReduce on other systems

Zdenka Prokopova
TBU in Zlín

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Distributed data processing

- **MapReduce** - the basic principle for parallel distributed processing of Big Data
- **Programming model** – introduced by Google in 2004

Definition:

MapReduce is "a simple and powerful interface that enables the automatic distribution and parallelization of computations over large-scale data, and the corresponding implementation of this interface that allows achieving high performance using a large cluster of commonly available computers" (Dean, 2008).

Zdenka Prokopova
TBU in Zlín

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Distributed data processing

⇒ the implementation is called the **MapReduce framework**

- Hadoop (Apache) - the best known and most widespread, it follows on from it
  - Hive - data warehouses
  - Mahout - data mining
  - ZooKeeper - distributed coordination of services
  - Spark - a unified analytical tool for large-scale data processing
- It also occurs as a direct part of NoSQL databases
  - MongoDB
  - CouchDB
  - Riak

Zdenka Prokopova
TBU in Zlín

Ghazi, M. R., & Gangodkar, D. (2015). Hadoop, MapReduce and HDFS: a developers perspective. *Procedia Computer Science, 48*, 45-50.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

The basic principle of MapReduce

- MapReduce is a programming model that sits on top of the Distributed File System (DFS)
  - Originally: no data model - data stored directly in files
- A distributed computing task contains **three phases**:
  1. **Mapping phase**: data transformation
  2. **Grouping phase**
     - It is done automatically by the MapReduce Framework
  3. **Reduction phase**: merging data

- The user only needs to define the mapping and reduction functions

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., . . . Gruber, R. E. (2006). *A distributed storage system for structured data.* Paper presented at the Proceedings OSDI.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Mapping phase - Map

- **Map** is used to process each object from a set of input data, and its output is pairs (key, value)
  - **Input**: one data item (e.g. a line of text) from a data file
  - **Output**: zero or more pairs (key, value) - multiple pairs can be created with the same key (non-unique)

- The mapping phase applies the Map function to all items

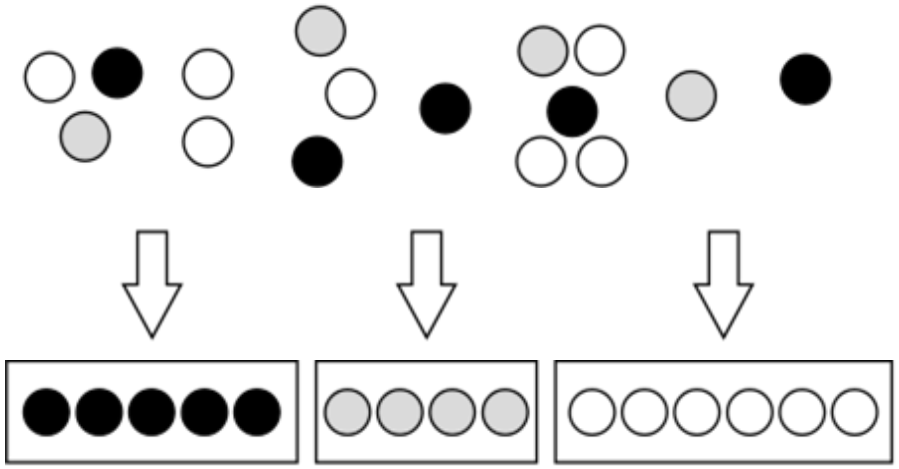
Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Shuffling phase

- **Grouping** (Shuffling): The key value **outputs** from the mapping phase **are grouped by key**
  - Values sharing the same key are sent to the same reducer
  - These values are consolidated into a single list (key, list of values)



intermediate step – mapping output

shuffle phase

Zdenka Prokopova
TBU in Zlín

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Reduction phase - Reduce

- **Reduce** combines the **values for each key**
  - To achieve the final result of the computational task
- **Input**: (key, list of values)
  - The list of values contains all values generated for a given key in the Map phase
- **Output**: (key, reduced list of values)

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

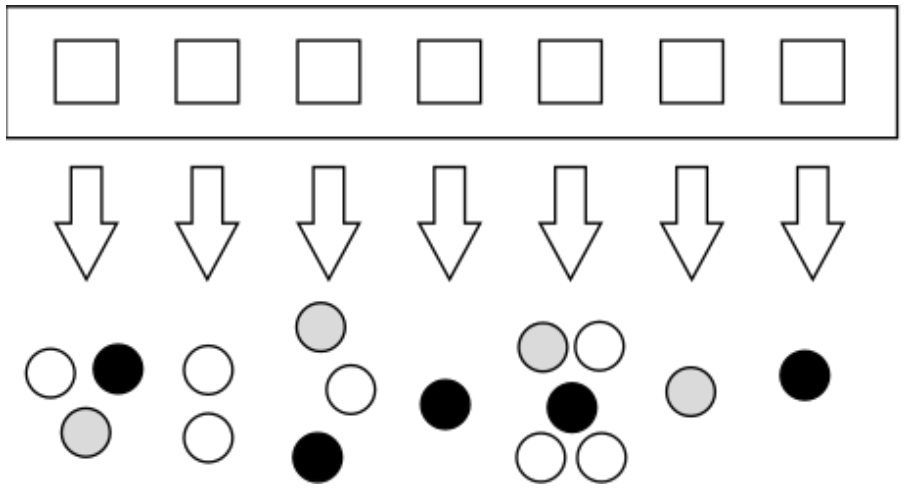
EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Example: Sum of words

Task: Find the number of words occurrences in a given set of documents

```
Map(String key, String value):
   // key: the document name
   // value: the document content(words)
 foreach word in value:
     return(word, 1);
```



input data (words)

Map function

output data (*colour indicates key*)

The Map function loops through individual document words and returns for each of them pair (word,1)

Zdenka Prokopova
TBU in Zlín

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

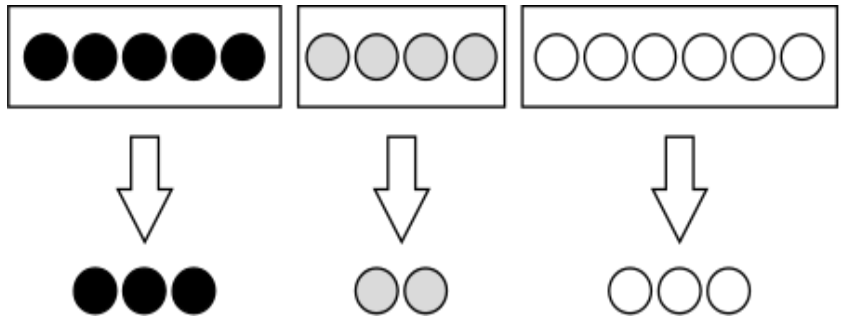
EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Example: Sum of words

```
Reduce(String key, Iterator values):
   // key: word
   // values: list of occurrences
int result = 0;
foreach v in values:
    result += v;
return(key, result);
```

The Reduce function sums the occurrence values of words in documents



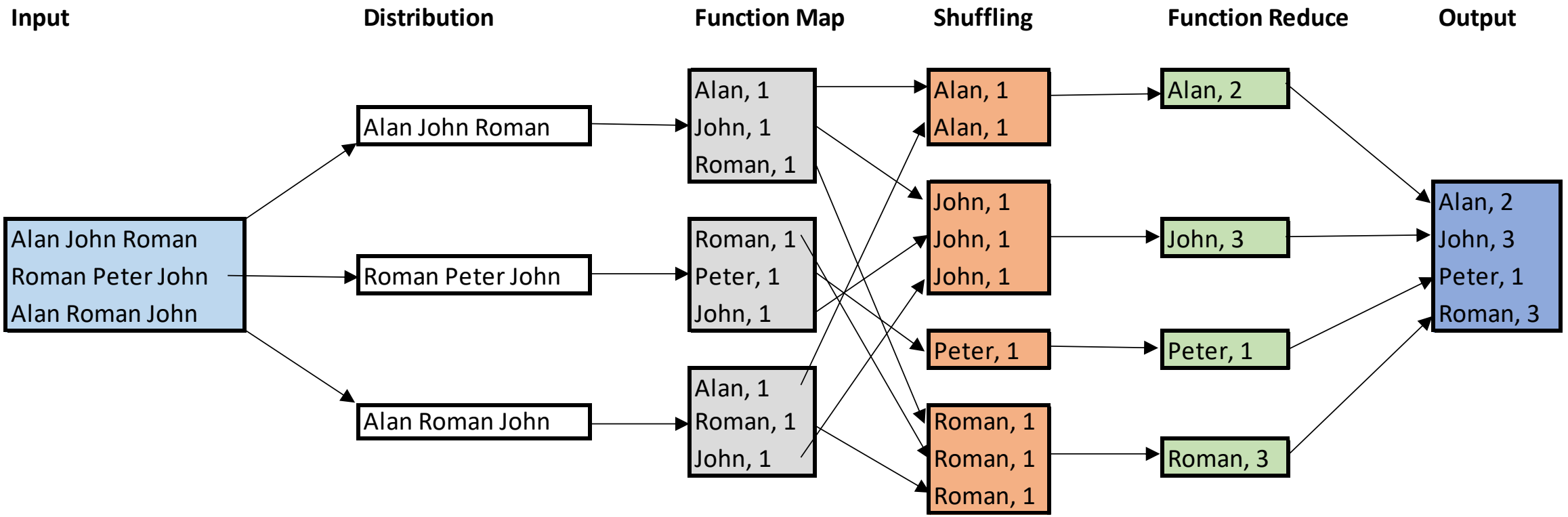
Zdenka Prokopova
TBU in Zlín

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Example: Sum of names



Zdenka Prokopova
TBU in Zlín

Holubová, I., Kosek, J., Minařík, K., & Novák, D. (2015). *Big Data a NoSQL databáze*: Grada.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

MapReduce Framework

- MapReduce framework takes care of:
  - Distribution and parallelization of calculation
  - Monitoring the entire distributed task
  - Grouping phase (compilation of interim results)
  - Recovery from all failures

- The user **only needs to define** the Map and Reduce functions
  - Defines the number of parts to process the Map functions - the M parameter
  - Defines the number of parts to process the Reduce functions - the R parameter
  - The parameters M and R may not exactly correspond to the number of nodes in the cluster (they tend to be higher)

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

MapReduce Framework - work phase

1. **Input Read** (function)
   - Defines how to read data from storage
2. **Map** (stage)
   - The master node prepares M parts of data and M Map of tasks (waiting)
   - Passes the individual split parts to the Map function, these tasks run on workers
   - After the task is completed, its interim results are saved
3. **Combinator** (function) - optional
   - Combines local intermediate results from the Map Phase

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

MapReduce Framework - work phase

4. **Partition** (function)
   - Specifies the distribution of data to the R part of the disk, corresponding to the R Reduce jobs
5. **Comparator** (function)
   - Sorts and groups the inputs for each reducer
6. **Reduce** (phase)
   - The master node prepares R tasks (waiting) for workers (workers)
   - The Partition function defines a data batch for each reducer
   - Each Reduce job uses a comparator to create (key, value) pairs
7. **Output Write** (Function)
   - Defines how output pairs (key, value) are printed

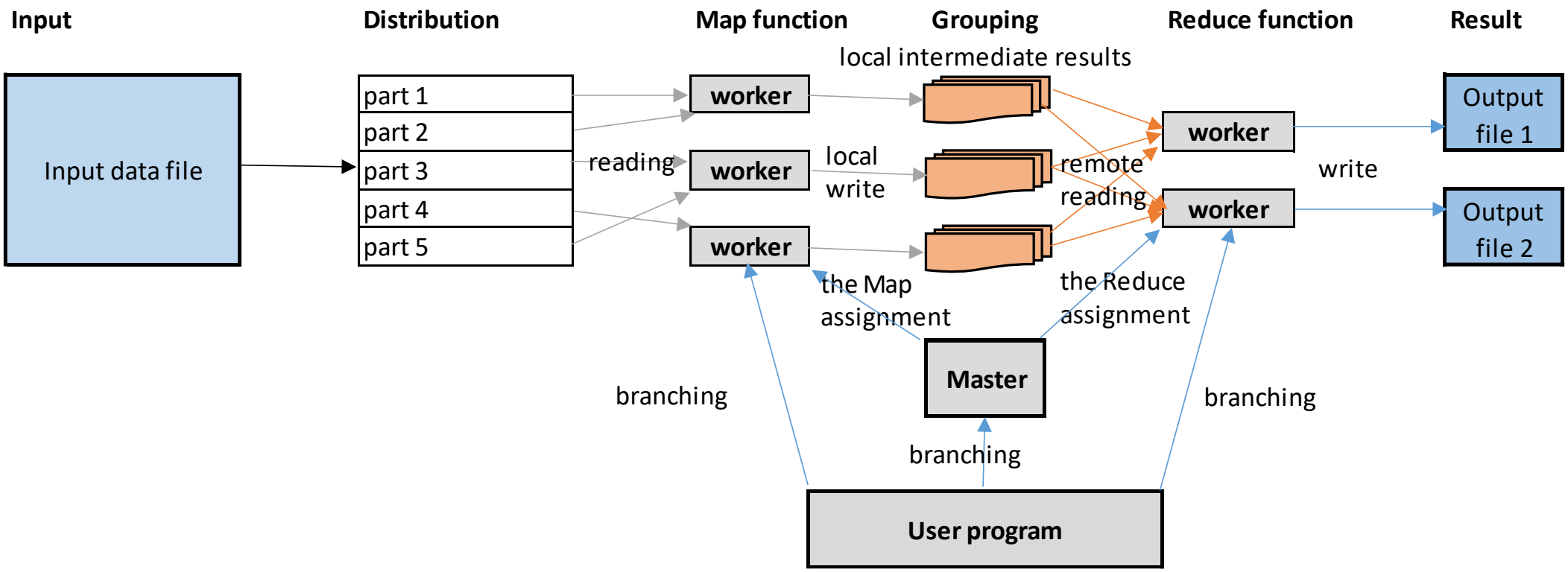
Zdenka Prokopova
TBU in Zlín

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

MapReduce Framework



Zdenka Prokopova
TBU in Zlín

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

MapReduce - properties

- MapReduce uses a "**shared nothing**" architecture
  - Nodes work independently, without memory/disk sharing
  - A common feature of many NoSQL databases

- Data is partitioned and replicated across multiple nodes
  - **Advantage**: a large number of read / write operations per second
  - **Disadvantage**: coordination problem - which nodes have my data, and when…

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Apache Hadoop

- **Hadoop modules** and related projects
- **Hadoop Distributed File System** (HDFS)
- **Hadoop MapReduce**

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Apache Hadoop

- Open-source Framework (created as part of the Nutch project - a web search engine)
  - 1st edition 2006
  - Implemented in Java
- The basic ideas are based on the MapReduce framework of Google

- Capable of running applications on large clusters of commodity hardware
  - Petabytes (10^{15}), Exabytes (10^{18}), Zettabytes (10^{21}) …data
  - Thousands of nodes

Apache Hadoop. (2019). Retrieved from http://hadoop.apache.org/

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Hadoop - Core Modules

- Hadoop Common
  - Common support functions for other Hadoop modules
- Hadoop Distributed File System (HDFS)
  - Distributed file system
  - It allows you to load and store large data efficiently
- Hadoop YARN
  - Task scheduling and resource management of the entire cluster
- Hadoop MapReduce
  - Implementation of MapReduce using previous modules

Apache Hadoop. (2019). Retrieved from http://hadoop.apache.org/

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Hadoop Distributed File System (HDFS)

- Open-source, platform independent (Java)
- Highly scalable
- Fault tolerant - able to:
  - Detect errors
  - Recover quickly and automatically
- Low efficiency
  - Optimized for batch processing

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

HDFS - data characteristics

- It assumes:
  - Access to streaming data
    - Read files from start to finish
  - Batch processing rather than interactive user access
- Large data sets
- Write-once / read-many
  - The created file does not need to be changed often
- Optimal applications for this model: MapReduce, web crawlers, data warehouses,…

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

HDFS – basic elements

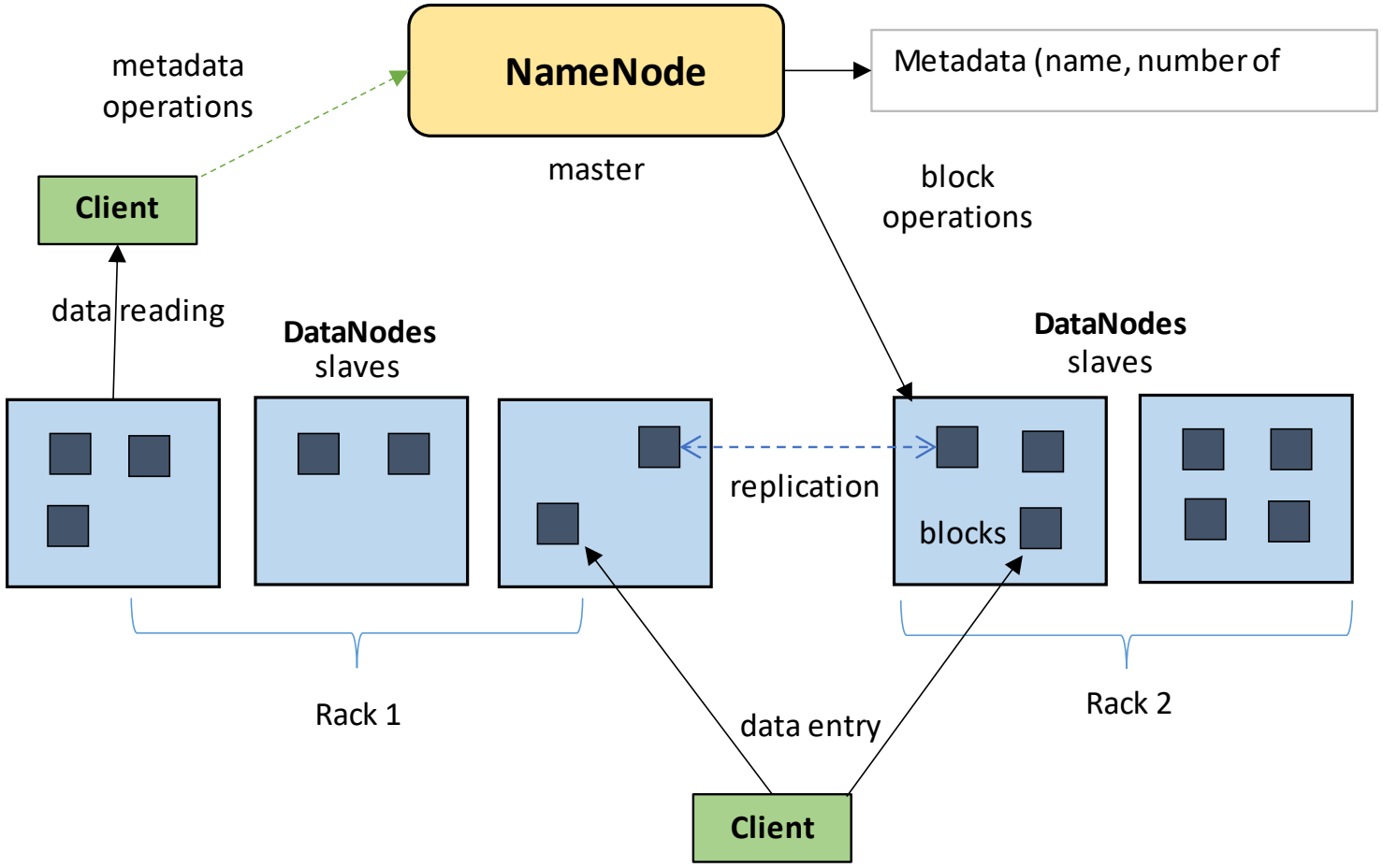
- **Master/Slave** architecture
- **NameNode** – Master type node
  - Manages the file system namespace
  - Open / close / rename files and directories
  - Regulates access to files
  - Specifies the mapping of blocks to DataNodes
- **DataNode** – (slave type data nodes) data in local file systems
  - Block read/write/create/delete/replicate
  - Typically, one data node corresponds to one physical node

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

HDFS schema



Zdenka Prokopova
TBU in Zlín

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Blocks and replication

- HDFS can store very large files across the cluster - each file is a sequence of blocks
  - All blocks in the file are the same size - except for the last one
  - Block size is configurable to one file (default 128 MB)
- **Blocks are replicated** for fault tolerance
  - The number of replicas is configurable for each file (the replication factor is typically set to 3)
- The **NameNode** receives HeartBeat and BlockReport messages from each DataNode
  - HeartBeat – a message about the functionality of the connection (if it does not come, the blocks on the DataNode are marked as non-functional)
  - BlockReport - list of all blocks in the DataNode

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Reliability

- The main goal is to **reliably store data** in case of:
  - NameNode failures
  - DataNode failures
  - Network fragmentation - a subset of DataNodes may lose connection to the NameNode

- In the absence of a HeartBeat message
  - NameNode marks DataNodes without HeartBeat and does not send any I/O requests to them
  - The death of a DataNode usually results in repeated replication

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Hadoop - MapReduce

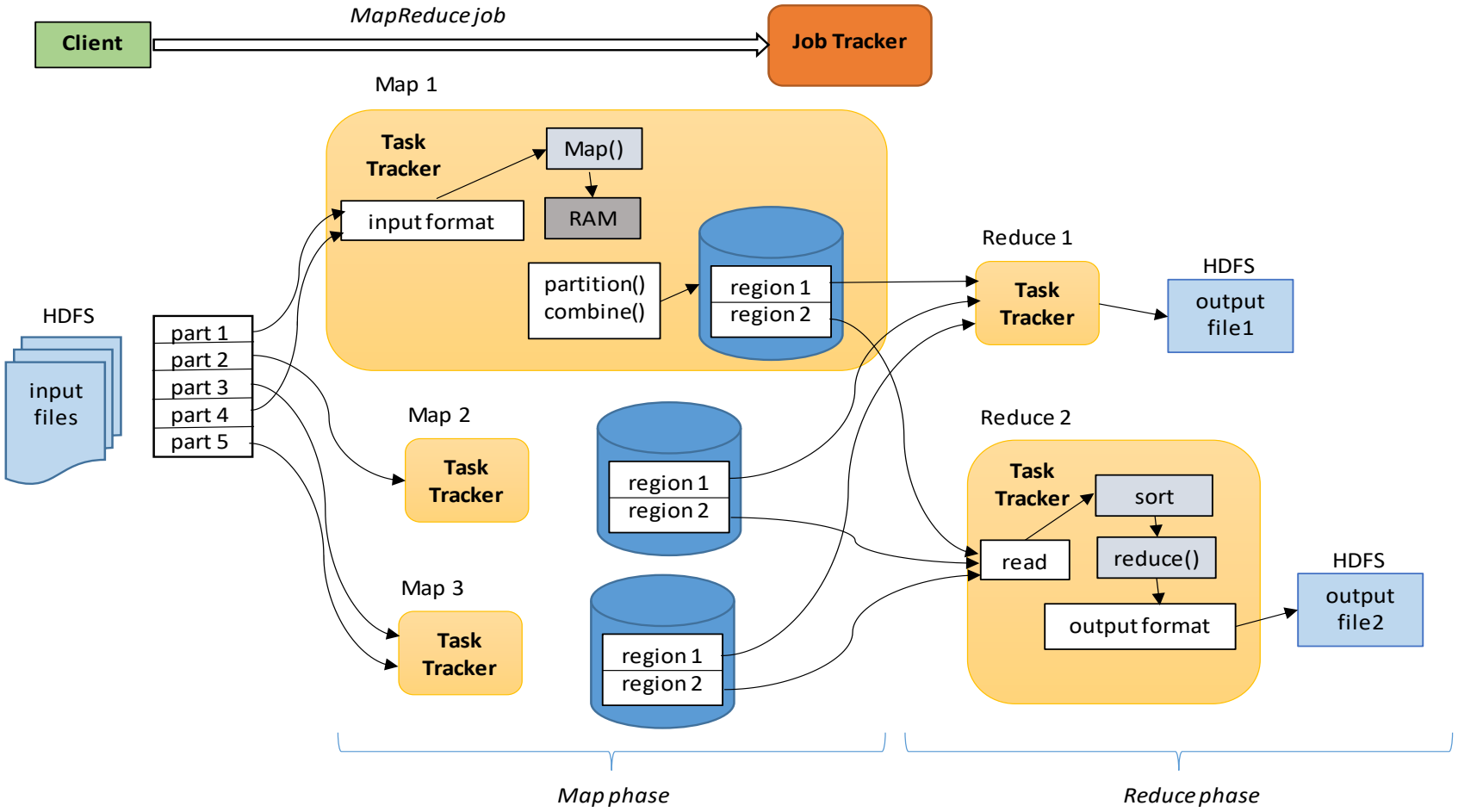
- **Hadoop MapReduce** needs:
  - Distributed file system (usually HDFS) – for storing data
  - A tool that can distribute, coordinate, track, and aggregate results (usually YARN)
- It contains **two main components**:
  - JobTracker (master) – provides control of the calculation (scheduler)
    - Monitors the entire MapReduce job
    - Communicates with the HDFS NameNode to run a job near the data
  - TaskTracker (slave, worker) - performs calculations - assigned Map or Reduce task (or other operations)
- A separate JVM (Java Virtual Machine) is started for each task

Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Schema - Hadoop MapReduce



Holubová, I., Kosek, J., Minařík, K., & Novák, D. (2015). *Big Data a NoSQL databáze*: Grada.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Hadoop - system superstructures

- Mahout: a scalable library for machine learning and data mining
- Hive: data warehouse - ad hoc querying and summarization of data
- Pig: using MapReduce principles for BigData analysis
- ZooKeeper: a high-performance coordination service for distributed applications

Lakhe, B. (2016). *Practical Hadoop migration : how to integrate your RDBMS with the Hadoop Ecosystem and re-architect relational applications to NoSQL*. Berkeley, California: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

MapReduce on other systems

- MongoDB - scalable distributed document database
- Cassandra - Scalable distributed columnar database
- Riak - Scalable distributed key-value database
- Spark - a universal distributed computing system for processing large volumes of data
- Amazon Elastic MapReduce - Hadoop-based data processing service

Lakhe, B. (2016). *Practical Hadoop migration : how to integrate your RDBMS with the Hadoop Ecosystem and re-architect relational applications to NoSQL*. Berkeley, California: Apress.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

The negatives of MapReduce

- The MapReduce principle is suitable for a specific set of tasks
- The efficiency of processing tasks using MapReduce tends to be very low
- MapReduce lacks important features that are part of DBS (integrity constraints, indexes, transactions, etc.)
- MapReduce is not compatible with tools that are implemented in DBS (business intelligence, data mining, etc.)
- A new approach - cloud computing

Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

References

- Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.
- RNDr. Irena Holubova, Ph.D. MMF UK course PA195: NoSQL Databases.
- Ghazi, M. R., & Gangodkar, D. (2015). Hadoop, MapReduce and HDFS: a developers perspective. *Procedia Computer Science, 48*, 45-50.
- Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., . . . Gruber, R. E. (2006). *A distributed storage system for structured data.* Paper presented at the Proceedings OSDI.

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

References

- Koitzsch, K. (2016). *Pro Hadoop Data Analytics: Designing and Building Big Data Systems Using the Hadoop Ecosystem*: Apress.
- Lakhe, B. (2016). *Practical Hadoop migration : how to integrate your RDBMS with the Hadoop Ecosystem and re-architect relational applications to NoSQL*. Berkeley, California: Apress.
- Paz, J. R. G. (2018). Working with a Globally Distributed Database. In *Microsoft Azure Cosmos DB Revealed* (pp. 203-218): Springer.
- Apache Hadoop. (2019). Retrieved from http://hadoop.apache.org/

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research,
Development and Education

MINISTRY OF EDUCATION,
YOUTH AND SPORTS

Questions?

Strategic project of TBU in Zlín, reg. no. CZ.02.2.69/0.0/0.0/16_015/0002204

Zdenka Prokopova
TBU in Zlín