BIG DATA ANALYTICS

By : Prof Mohammed Tanzeem Agra

BIG DATA ANALYTICS

DOUGLAS EADLINE " Hadoop 2 Quick – Start Guide"

MODULE 1 - HDFS

- Hadoop Distributed File System Basic
- Running Example Programs and Benchmark
- Hadoop Map Reduce Frame Work
- MapReduce Programming

Defining Apache Hadoop

- Hadoop was started at Yahoo! To provide a data processing infrastructure to the Apache web search engine in 2005.
- Single Software to support a web search engine
- It is a system tool used to analyse large amount of data (block range from 64MB to 128MB)
- Key : move the computation to the data because it is more faster than moving data from one server to another server.

Some area where Hadoop is used and prominent users

- Social Media
- Retail web commerce
- Financial Services
- Web Search
- Government
- R&D

- Yahoo
- Facebook
- Amazon
- Ebay
- American Airlines
- The New York Times
- IBM
- Chevron

The Features that shape Hadoop data processing are as follows

- Core parts are open source under the Apache License.
- Support for structure, unstructured and mixed form of data.
- Analysis usually involves large unstructured data sets sometimes in the petabyte (10^{15} bytes) range.
- Data is stored across multiple servers using the scalable hadoop distributed file system, iSCSI.
- Many applications and tools are based on the Hadoop version 1 MapReduce Programming model.

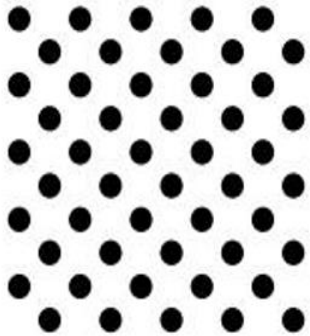
Features

- Hadoop MapReduce job can scale from a single server to thousand of machine and tens of thousand of processor cores.
- YARN (Yet Another Resource Negotiator) are supported in Hadoop v2.
- Hadoop core components were designed to run on commodity hardware and cloud.
- Many projects and application are built on the top of the Hadoop Infrastructure .
- Core components are written in java Hadoop application can use almost any programming language.

Defining Big Data

- Big Data, as the name implies, suggests large – volume data processing – often measured in petabytes (10^{15} bytes), several characteristics that define Big Data
  - Volume : the sheer size of the data makes it impossible to evaluate by more conventional means.
  - Variety : Data may come from a variety of source and not necessarily be related to other data sources.
  - Velocity : The term velocity in this context refers to how fast the data can be generated and processed
  - Variability : Data may be highly variable, incomplete and inconsistent.
  - Complexity : Relationship between data source may no be entirely clear and not amenable to traditional relational method.

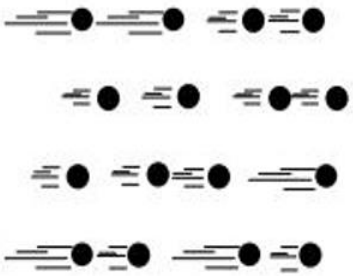
Volume



Data at Rest

Terabytes to exabytes of existing data to process

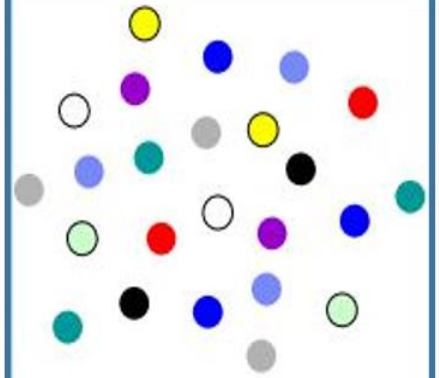
Velocity



Data in Motion

Streaming data, milliseconds to seconds to respond

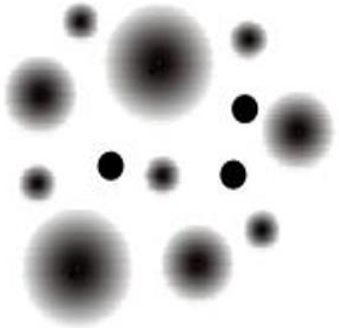
Variety



Data in Many Forms

Structured, unstructured, text, multimedia

Veracity



Data in Doubt

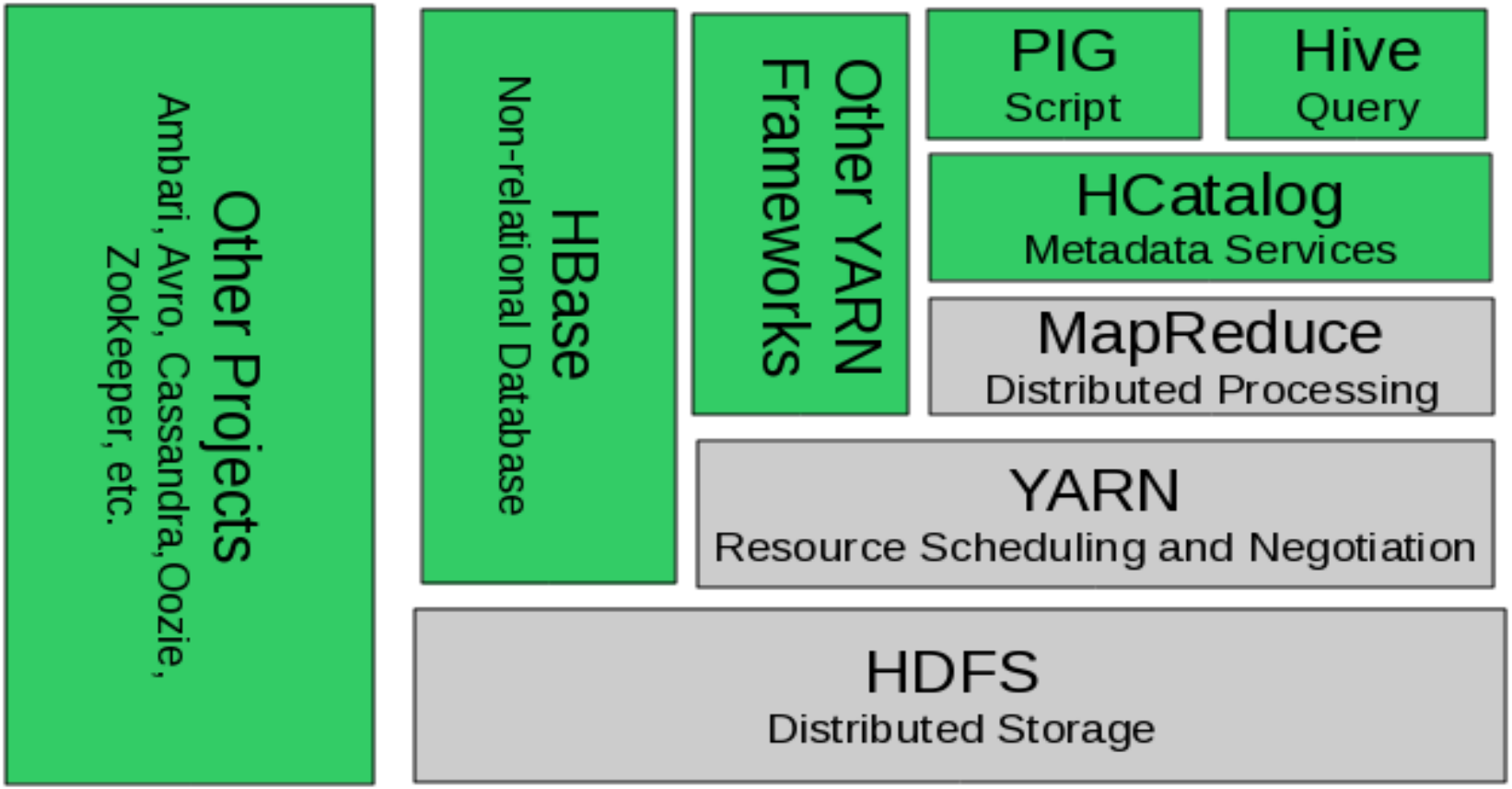
Uncertainty due to data inconsistency & incompleteness, ambiguities, latency, deception, model approximations

Characteristics of Big Data

Type of Data Processing

- Media including video, audio, and photos
- Web data including system/web logs, click trails, and text messages/email
- Written Documents and books
- Scientific Data including simulation results, human genome
- Stock transaction, customer data and retail purchase
- Telecommunications including phone record.
- IoT (Data from all connected device)
- Real time sensor data including traffic and transportation logistic.

Example of Hadoop v2



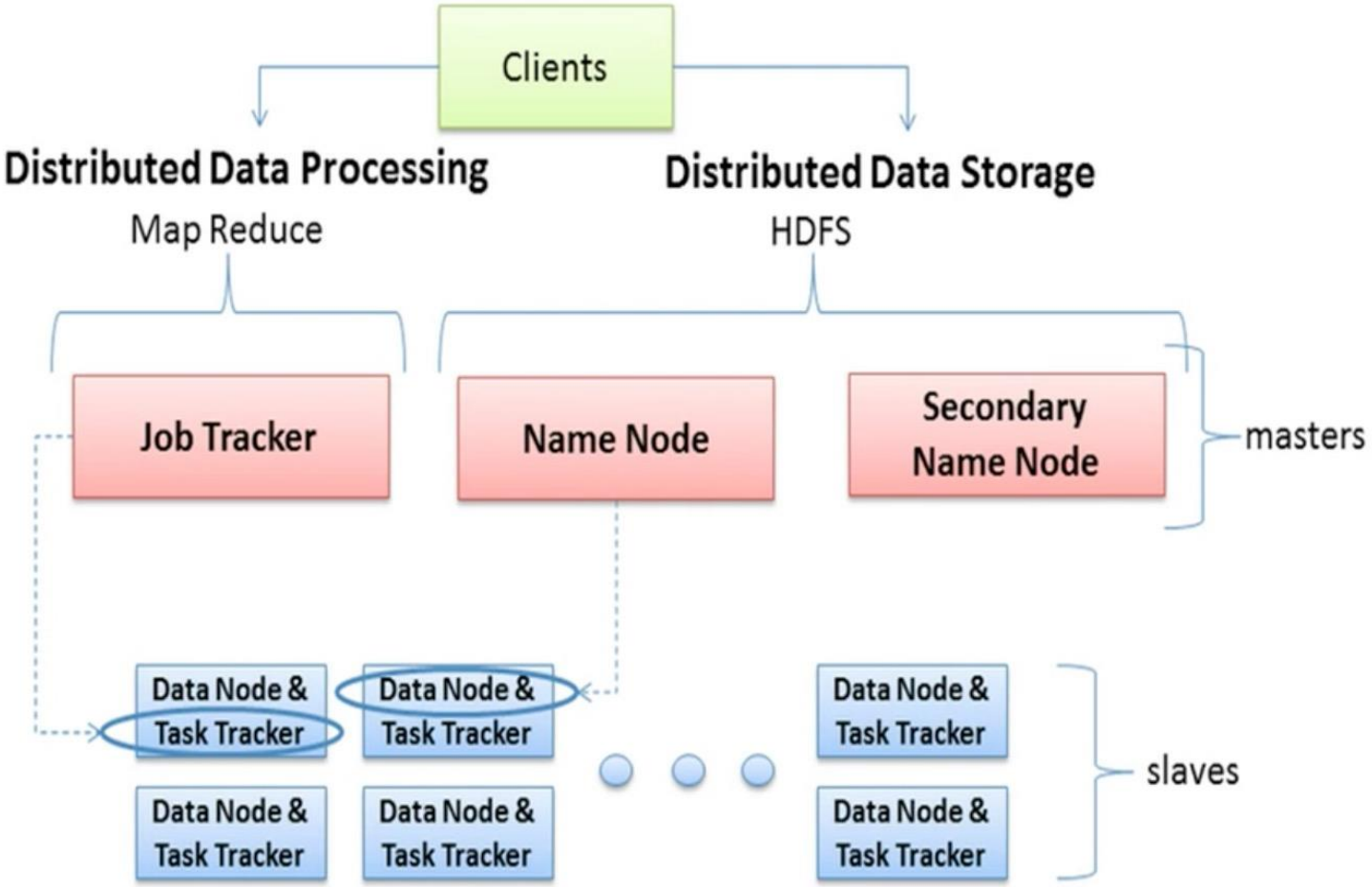
1.1 Hadoop Distributed File System Basics

- HDFS architecture
- Block Replication
- Safe Mode
- Rack Awareness
- High Availability
- Back Up

HDFS FEATURES

- HDFS was designed for Big Data Processing, it support for many users simultaneosuly, but hdfs is not designed as a true parallel system.
- Designed as a write once / read many model to solve coherence and concurrency problem in parallel programming language.
- Data storage and processing happens on the same server node.
- Moving computation cheaper the moving data
- File system maintaince multiple copies of data across the cluster.

HADOOP ARCHITECTURE



COMPONENTS

- There are two types of HDFS installation
  - Single Node Installation.
  - Multiple Node Installation.
- The designed of HDFS is based on three types of nodes
  - Name Node : Only one node per cluster
  - Date Node : Multiple Nodes
  - Secondary Node : Checkpoint or Backup Node

NAME NODE (MASTER NODE)

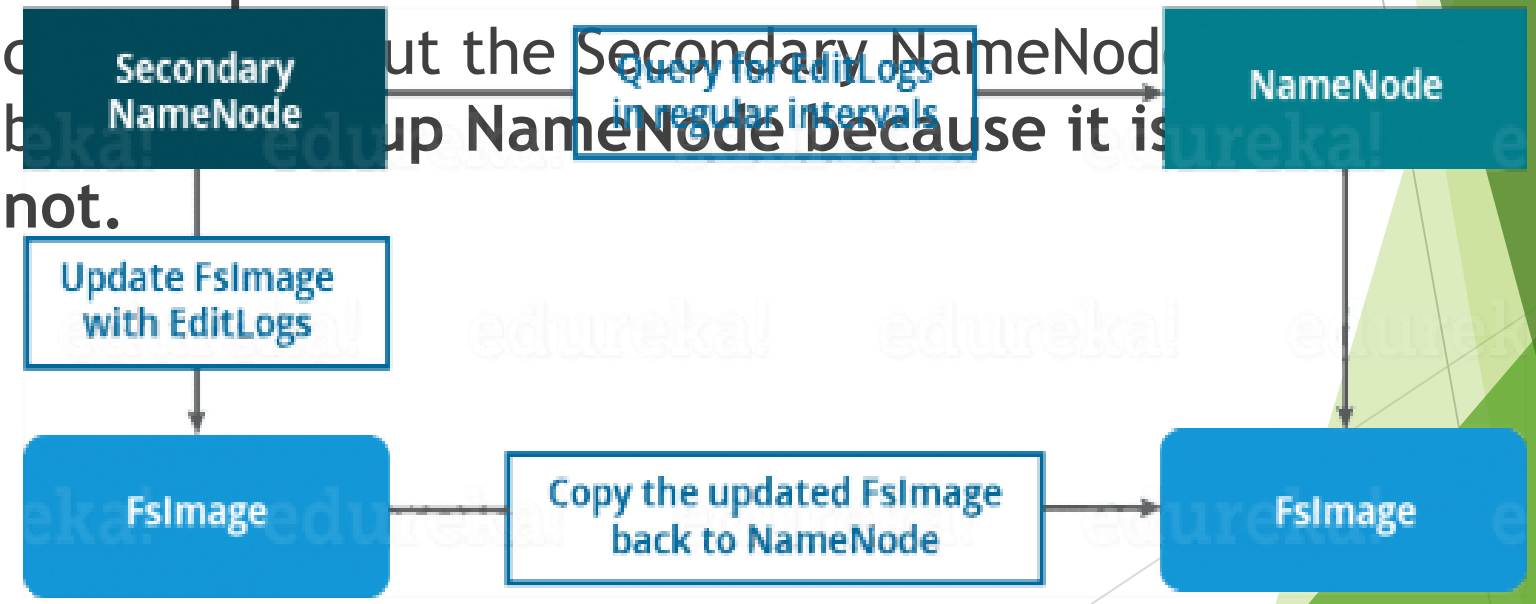
- Name Node Manages all the meta data needed to store and retrieve actual data from the Data Nodes.
- Data is not stored in Namenode.
- Master Node manages the file system namespace and regulates access to files by clients.
- Name Node perform operation such as opening, closing, delete, renaming files and directories.
- NameNode also handles mapping of blocks to DataNodes and Handle Data Node Failure.
- Name Node Manages block creation,mapping of blocks, deletion and replication.
- It maintains two file **FsImage (New File Entry) and EditLogs (Deletion and Modification)**

DATA NODE (SLAVE NODE)

- DataNodes are the slave nodes in HDFS. Unlike NameNode, DataNode is a commodity hardware, that is, a non-expensive system which is not of high quality or high-availability. The DataNode is a block server that stores the data in the local file ext3 or ext4.
- The actual data is stored on DataNodes and process which runs on each slave machins.
- The DataNodes perform the low-level read and write requests from the file system's clients.
- They send heartbeats to the NameNode periodically to report the overall health of HDFS, by default, this frequency is set to 3 seconds.

SECONDARY NAMENODE

- The Secondary NameNode works concurrently with the primary NameNode as a **helper daemon.** And don't be c ut the Secondary NameNod b up NameNode **because it is not.**

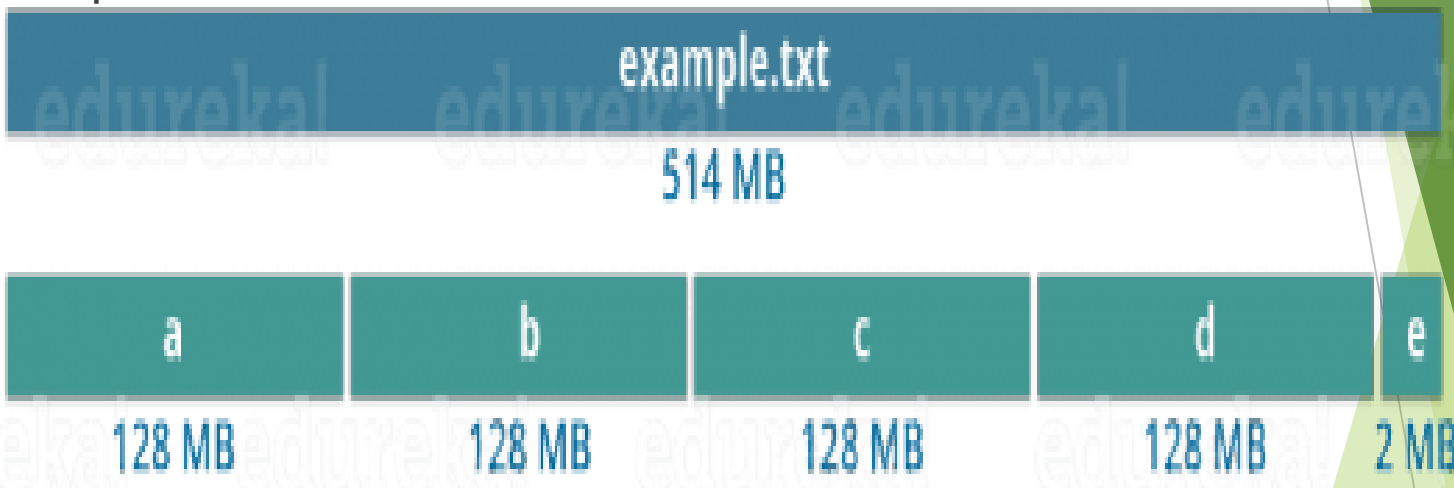


SECONDARY NAMENODE

- The Secondary NameNode is one which constantly reads all the file systems and metadata from the RAM of the NameNode and writes it into the hard disk or the file system.
- It is responsible for combining the EditLogs with FsImage from the NameNode.
- It downloads the EditLogs from the NameNode at regular intervals and applies to FsImage. The new FsImage is copied back to the NameNode, which is used whenever the NameNode is started the next time

HDFS BLOCK REPLICATION

- Example



HDFS Safe Mode

- When the NameNode starts, it enters a read-only safe mode, where blocks cannot be replicated or deleted.
- Safe Mode enables the NameNode to perform two important things

1) The previous file system state is reconstructed by loading fsimage file into memory and replaying the edit log.
2) Mapping between data node and block is created to register. Atleast one datanode should be available to exit from safe mode.

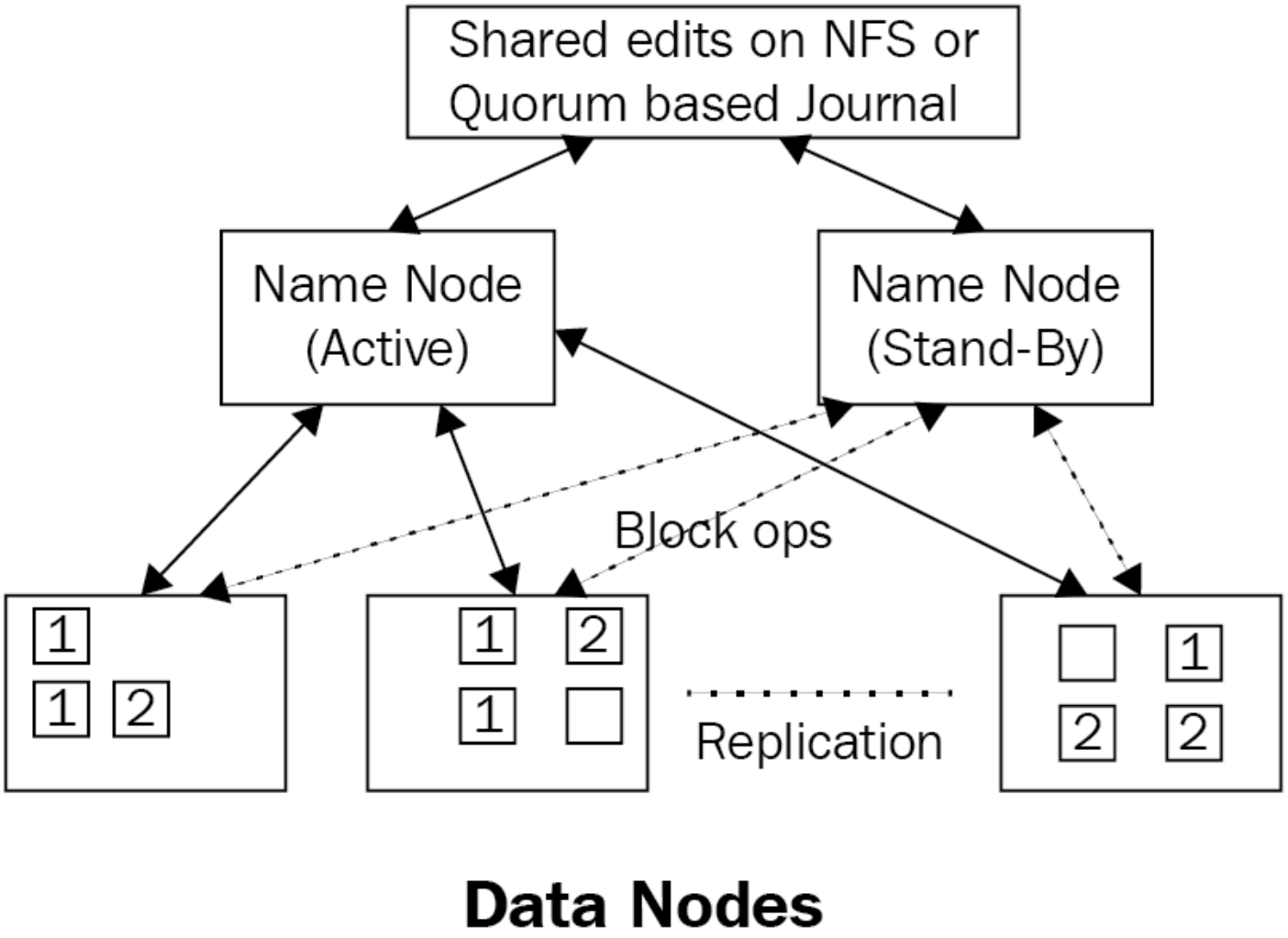
Command : hdfs dfsadmin-safemode (Admin Side)

NameNode High Availability

- NameNode is a single point of failure that could bring down the entire Hadoop Cluster.
- NameNode Hardware often employed redundant power supplies and storage to guard against such problem, but it was still susceptible to other failure.
- The solution is to implement NameNode High Availability (HA) to provide true failure service.

- Hadoop cluster has two separate NameNode Machine. Each Machine is configured with exactly the same software.
- NameNode : Active State and Standby State
- Active State : The active NameNode is responsible for all client HDFS operations in the cluster.
- Standby Node : To back up NodeNode
- Journal Node : Durring Failure, It write one at a time in standby node
- ZooKeeper is used to monitor the NameNode Health. ZooKeeper is highly available service for maintaining small amount of coordination data, notifying clients of changes in that data.

High Availability



Data Nodes

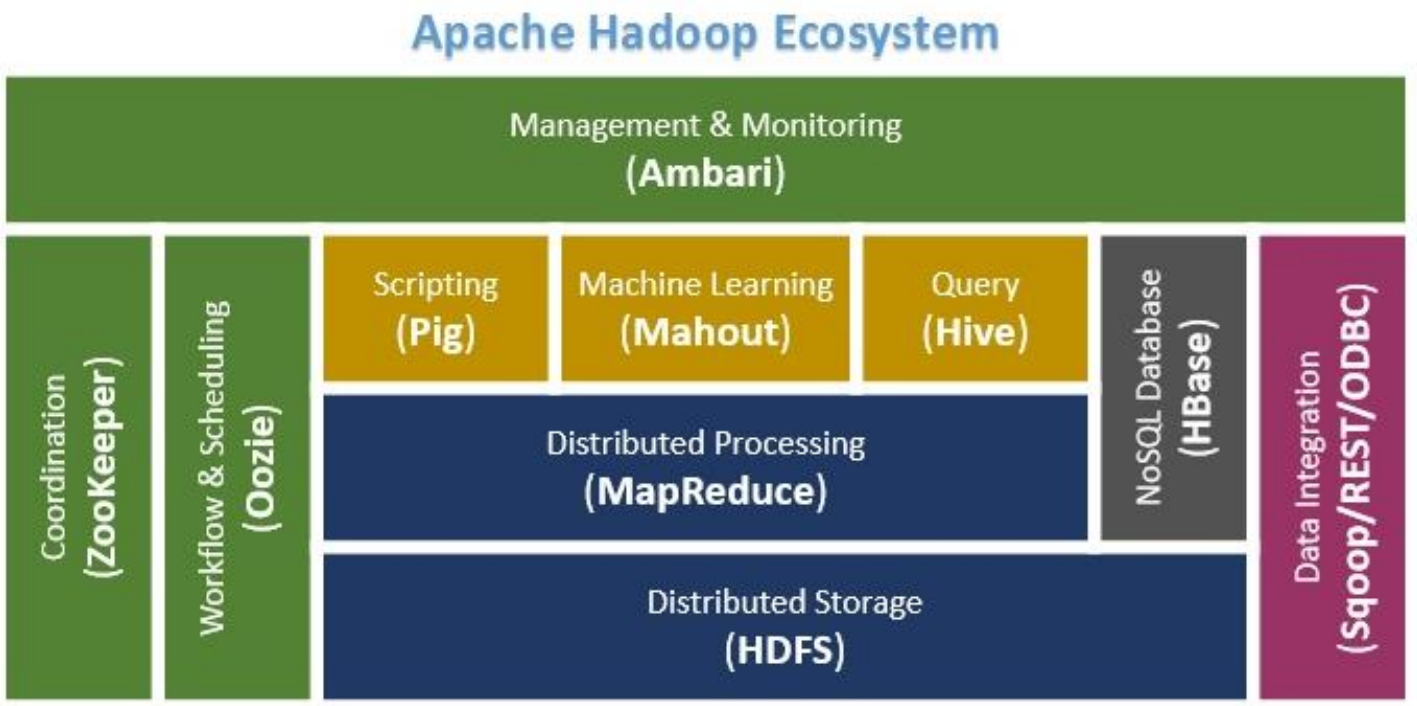
HDFS Snapshots (Backup)

- Created by administrator using the hdfs dfs -snapshot command.
- They offer the following feature
  - ✓Snapshot can be taken for directory or entire file system.
  - ✓Can be used for data backup, protection against user error, and disaster recovery.
  - ✓Creation is instantaneous.
  - ✓Block are on the data node are not copied, There is no data copying or creating duplicate file.
  - ✓It will not effect regular HDFS operations.

Hadoop MapReduce Framework

- The MapReduce Model
- MapReduce Parallal Data Flow
- Fault Tolerance and Speculative Execution
- Hadoop MapReduce Hardware
- Example

MapReduce Model



MapReduce Model

- MapReduce is a programming framework that allow us to perform distributed and parallel processing on large data sets in a distributed environment.
- There are two stages
- ✓ Mapping Stage : a mapping procedure is applied to the input; its some kind of filter or sorting process.
- ✓ Reducing Stage : the reduce function takes the input value,sum them and generates a single output of the data and final sum.
- ✓

Example

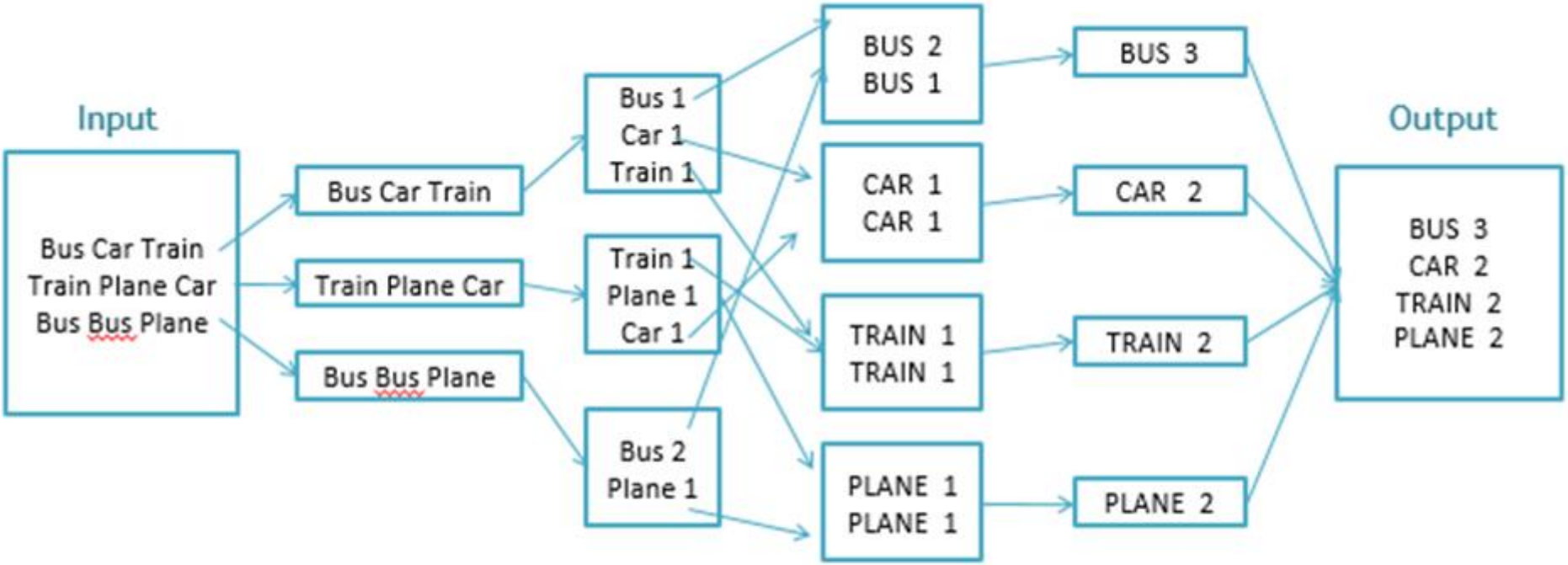


Fig. WorkFlow of MapReducing

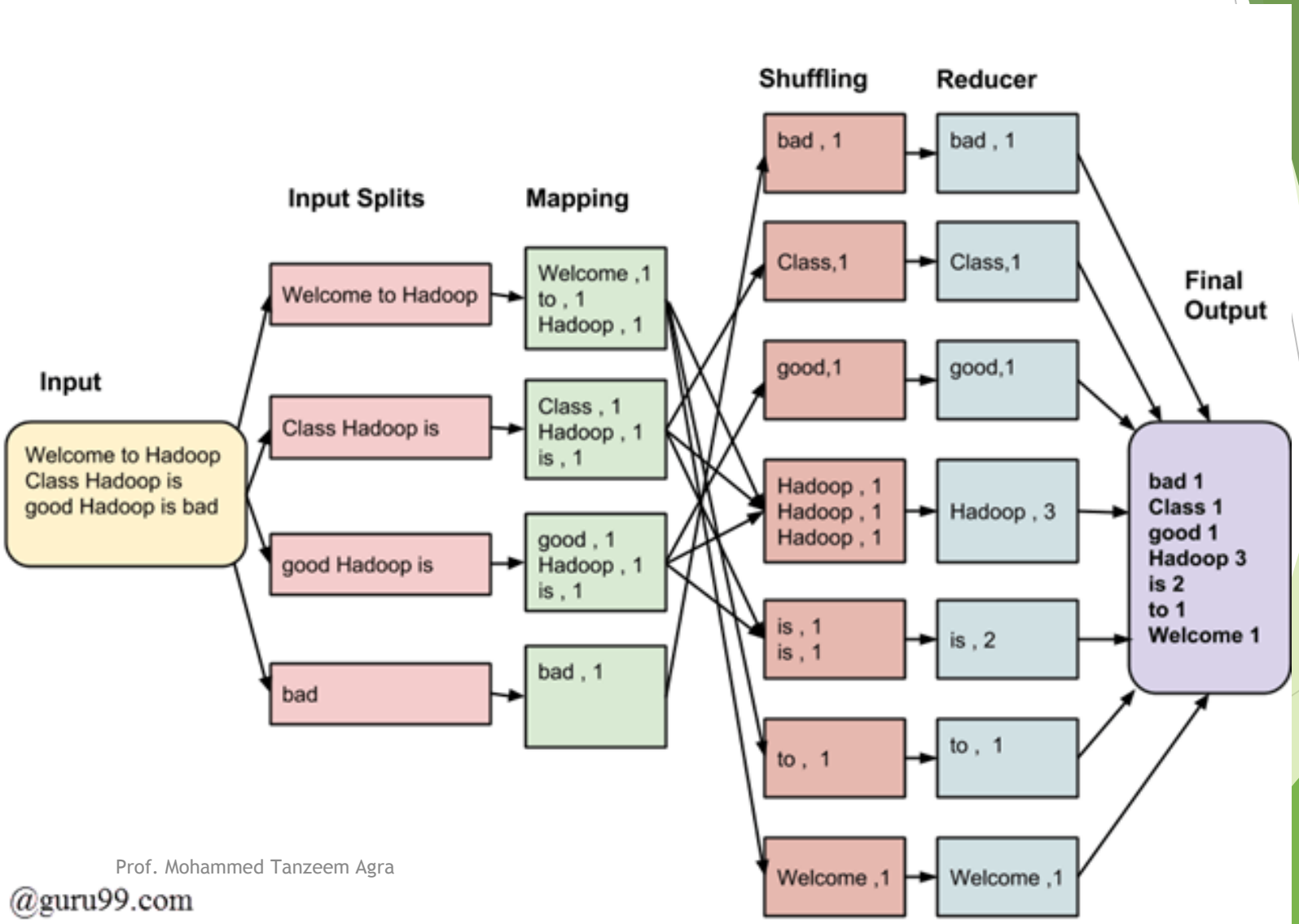
MapReduce Important Properties

- Data Flow is in one direction (map to reduce). It is possible to use output of a reduce step as the input to another MapReduce process.
- The input data are not changed but by applying the mapping and reduction functions to the input data, new data is produce.
- There is no dependency on how the mapping and reducing function are applied to the data, the mapper and reducer data flow can be implemented in any number of ways to provide better performance.
- MapReduce enable large anount of data to be analyzed quickly. In general the mapper process is fully scalable and can be applied to any subset of the input data.
- Data Locality and Parallel Processing

2.2 : MapReduce Parallel Data Flow

- The Basic Steps are as follows
- ✓ Input Split
- ✓ Map Step
- ✓ Combiner Step
- ✓ Shuffle step
- ✓ Reduce Step

Data Flow Diagram



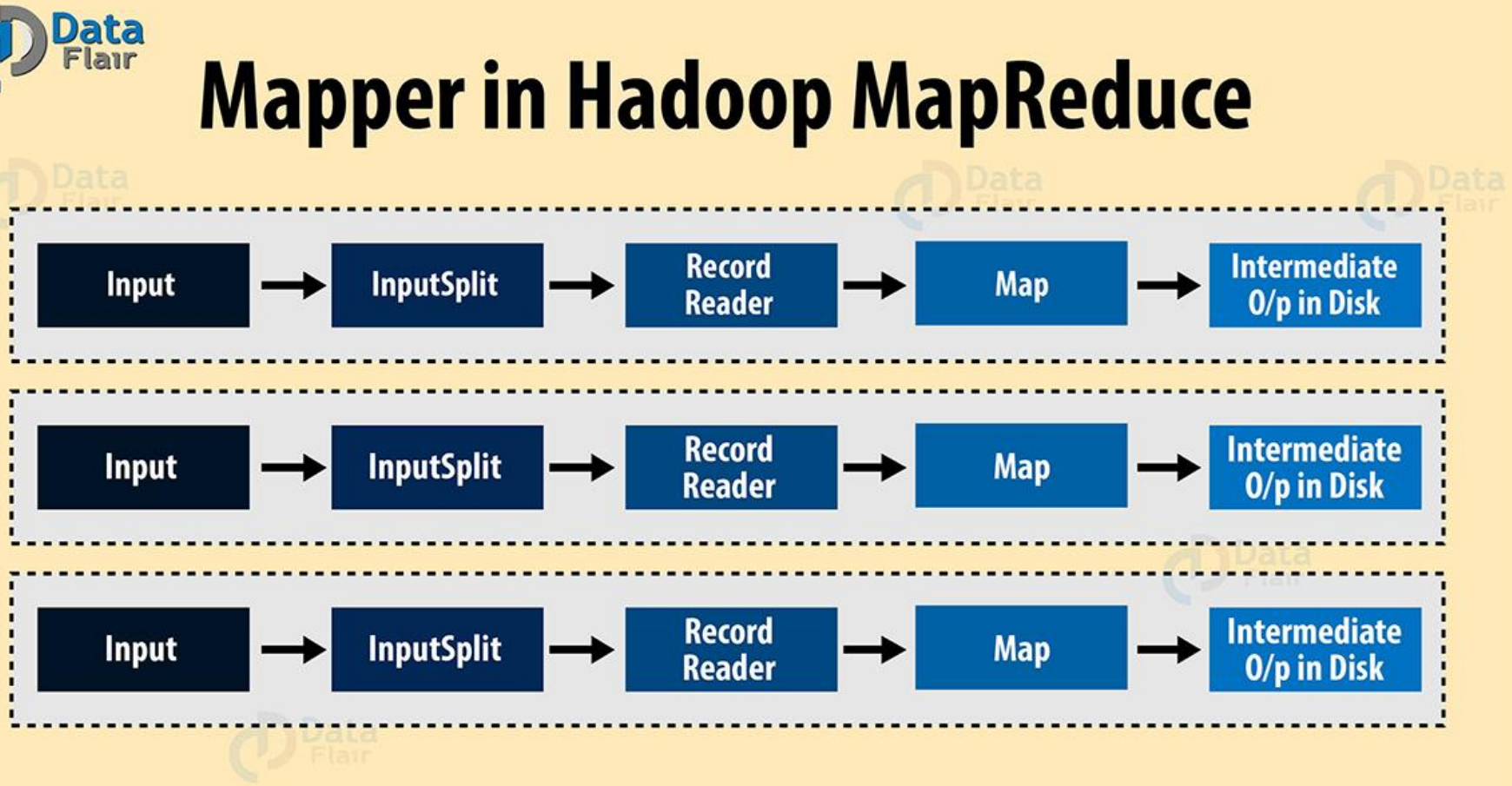
Split Input

- HDFS distribute and replicates data over multiple servers. The default data chunk or block size is 64MB/128MB.
- Thus 500MB file would be broken down into 8 blocks and written to the different machine in the cluster.
- The data also replicated on multiple machine.
- The time required to load and distribute data throught out HDFS server can be considered part of the total processing time.
- Splits are always and almost smaller than the HDFS block size. The number of splits corresponds to the number of mapping processes used in the map stage.

MAP Step

- Mapper is the first phase of processing that process each input record and generates an intermediet output on local machine.
- For large amount of data, many mappers can be operating at the same time.
- MapReduce will try to execute the mapper on the machine where the block resides.
- Hadoop Mapper task processes each input record and it generates a new <key, value> pairs.
- In mapper task the output is full collection of <value,key> pair.

MAP Diagram


Data Flair
Mapper in Hadoop MapReduce
Input
InputSplit
Record Reader
Map
Intermediate O/p in Disk
Input
InputSplit
Record Reader
Map
Intermediate O/p in Disk
Input
InputSplit
Record Reader
Map
Intermediate O/p in Disk

Combiner Step (Mini-Recorder)

- Key – value pair are combined prior to the next stage.
- The primary job of Combiner is to process the output data from the Mapper, before passing it to Reducer.
- It runs after the mapper and before the Reducer and its use is optional.

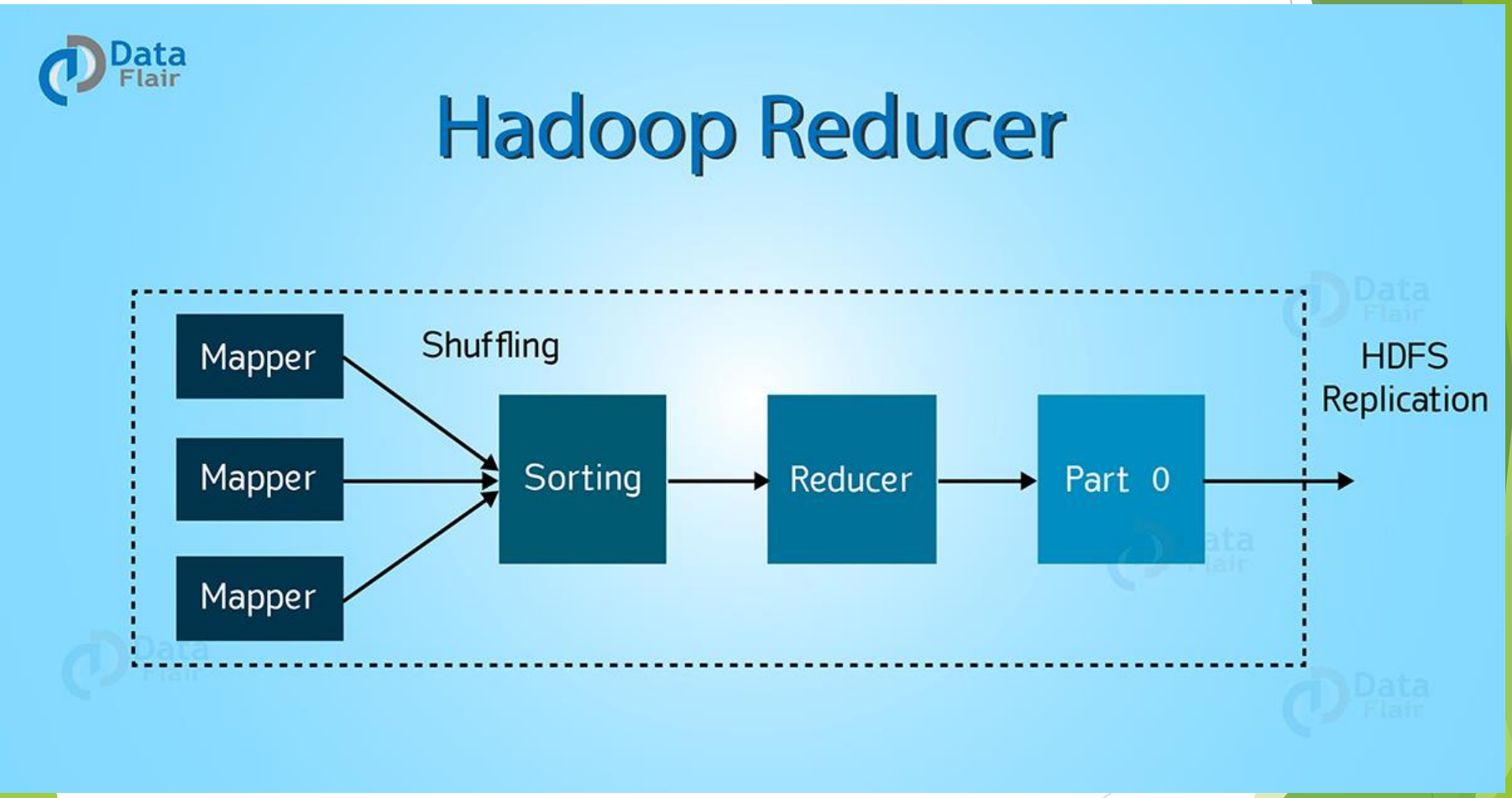
Shuffle Stage

- All the key is combined and counted by the same reducer process.
- The results of the map stage must be collected by key-value pair and shuffled to the same reducer process.
- If only a single reducer process is used, the shuffle stage is not needed.

Reduce Step

- Reducer takes the output of the Mapper (intermediate key-value pair) process each of them to generate the output. The output of the reducer is the final output, which is stored in HDFS.
- One-one mapping takes place between keys and reducers. Reducers run in parallel since they are independent of one another. The user decides the number of reducers. By default number of reducers is 1.
- Example : a MapReduce job running four reducers will create files called part-0000,part-0001,part-0002,part-0003

Reducer Diagram



Fault Tolerance

- Fault Tolrence is a property that enables a system to continue operating properlly in the event of failure of some of its components.
- Strict Control of Data Flow through out the execution : mapper processes do not exchange data with other mapper processes and data can only go from mapper to reducer.
- Recovery From failure of one or many map process : if server fail, the map tasks that were running on that machine could easily be restarted on another working server.
- Failed Reducers can be restarted but addition work has to be redone in such case.

Speculative Execution

- Execution of program is challenging because of large cluster but control and monitoring resourcing is easy.
- When one part of MapReduce process runs slowly; it ultimately slow down everything such in parallel computing
- In Sepculative Execution.

Hadoop MapReduce Hardware

- Server
- Storage (Hard disk)
- Processing (Processor)
- Old and New Hardware ?