Big Data

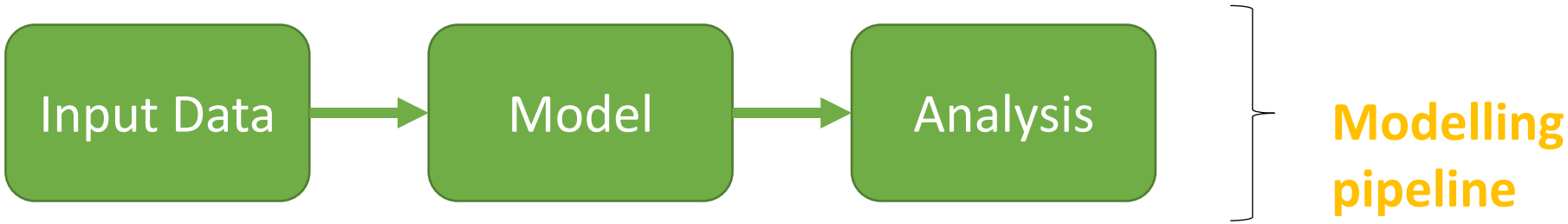
Introduction

Rachana B S, Usha Devi B G, Resma K S, Prafullata K A, Subramaniam K V
Department of Computer Science and Engineering
{rachanabs,ushadevibg, resma, prafullatak, subramaniamkv}@pes.edu
+91 80 6666 3333 Extn 877

What is Big Data?

There is no one standard single definition.

Big Data is data whose scale, diversity, and complexity require new architecture, techniques, algorithms, and analytics to manage it and extract value and hidden knowledge from it…



Model – is a human construct that better helps us understand real-world systems/phenomena.

With Big Data, this means….

BIG DATA

Big Data themes

- How to manage very large amounts of data (*data management)*
- Google: store index to WWW and search
- and extract value and knowledge from them (*analytics)*
- Amazon: store user purchases and make recommendations

Large-Scale Data Management

Big Data Analytics

Data Science and Analytics

Big Data: Motivating Example

BIG DATA

Big Data themes

- How to manage very large amounts of data (*data management)*
- and extract value and knowledge from them (*analytics)*

- Google: store index to WWW and search
- Amazon: store user purchases and make recommendations

Large-Scale Data Management

Big Data Analytics

Data Science and Analytics

High level approach: motivating example

- Machine Translation
- Translating a sentence from English → Hindi
- What would be the traditional approach?
- How will it differ from the Big Data approach?

English	Hindi
Can you teach me?	क्या तुम मुझे सिखा सकते हो?
You make mistakes if you do things in a hurry.	जल्दबाज़ी में काम करोगे तो ग़लतियाँ तो होंगी ही।

https://towardsdatascience.com/intuitive-explanation-of-neural-machine-translation-129789e3c59f

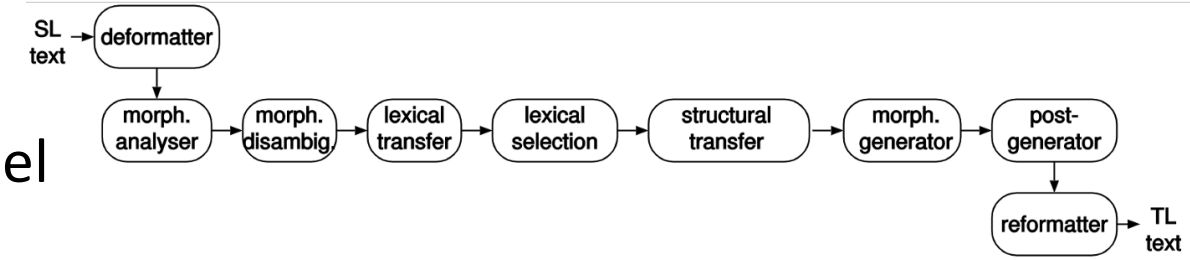
BIG DATA

Traditional Approach

- Understand the system – linguistic approach – rule based

- Requires a linguistic expert to build a model

- Model should include
  - Language structure → morphology, grammar
  - Meaning of the words
  - Mapping words from one language to another

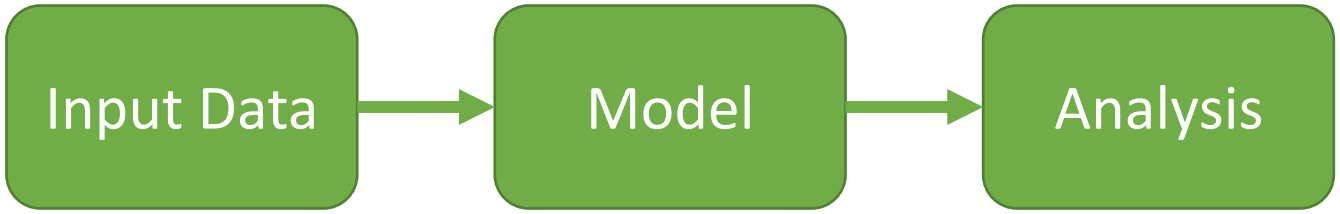


https://towardsdatascience.com/machine-translation-a-short-overview-91343ff39c9f

Big Data Approach

- No attempt to understand language
- Gather data about different sentences and translations
  - Requires a parallel corpus
  - Millions of sentences and their translations
- Build a statistical model
- For example:
  - Every time the word cat appears in the English sentence
  - The hindi equivalent has *billi*
  - So infer that cat can be translated as *billi*

https://techmediahub.com/machine-translation-complete-useful-guide/



Traditional Approach

The model is human generated

Big Data Approach

The model is machine generated

BIG DATA

What about domain knowledge?

- Correlation is enough?
- Gene sequencing of DNA fragments found in ocean by J. Craig Venter
  - 1000s of new species
  - No idea of what species looks like or any other info

- All models are wrong, and increasingly you can succeed without them
  - Peter Norvig, Google's research director
  - “The unreasonable effectiveness of data”


The End of Theory: The Data Deluge Makes the Scientific Method Obsolete

By Chris Anderson 06.23.08

Illustration: Marian Bantjes

http://www.wired.com/science/discoveries/magazine/16-07/pb_theory

Conclusions from Peter Norvig's talk

- Algorithms are not important, data is
  - Domain knowledge (e.g., physics/grammar) is not important

- Demonstrates how images can be merged together using just data

- And translation of text giving examples of issues in segmentation

- Peter Norvig, Head Google Research, *The Unreasonable Effectiveness of Data*
  https://www.youtube.com/watch?v=yvDCzhbjYWs

What about domain knowledge?

- Can we rely only on data alone?

- Does this mean that domain knowledge is obsolete?

Big Data: Pitfalls in Analysis

Issues in machine translation

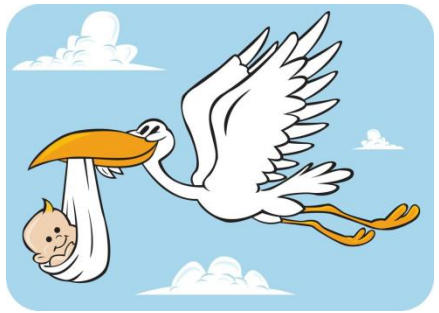
- What about *let the cat out of the bag*?
  - Naïve translation - *billi ko bag ke bahar chhod diya*
  - English meaning: reveal a secret

- To be able to solve this, we need information about the language → domain knowledge and some experimentation

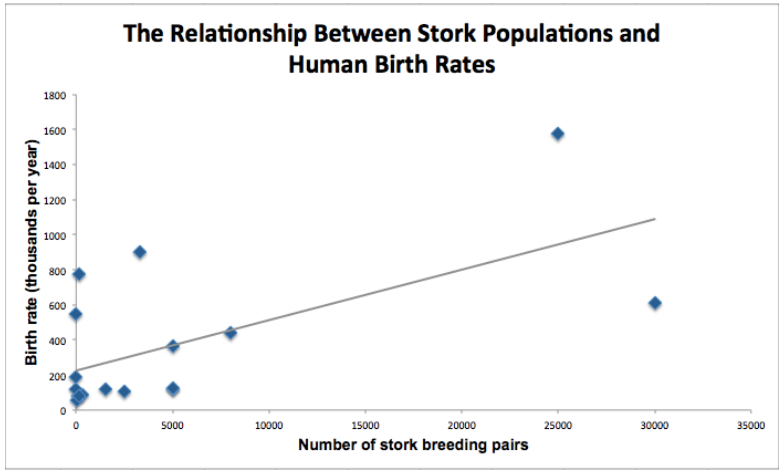
- Peter Norvig, Head Google Research, *The Unreasonable Effectiveness of Data*
  https://www.youtube.com/watch?v=yvDCzhbjYWs

BIG DATA

Pitfall : Spurious correlation



- C->A, C->B
  - *Does A->B?*
- Example:
  - Do storks deliver babies?
- Chart shows positive correlation between
  - Stork population and human birth rates in European countries
  - What it does not show is a hidden variable
    - Available nesting area?



- http://en.wikipedia.org/wiki/Spurious_relationship
- http://www.cut-the-knot.org/do_you_know/misuse.shtml

BIG DATA

Pitfall : Gaps in the data

- Selection bias
- Convenience
- Example
  - Rutgers University study
  - Examine decision-making process in emergency
  - Study tweets during Hurricane Sandy
  - Most tweets from Manhattan!
  - If studying impact of Sandy: *Manhattan most impacted!*

- *More Data, More Problems: Is Big Data Always Right?* ARI ZOLDAN
  http://www.wired.com/insights/2013/05/more-data-more-problems-is-big-data-always-right/

BIG DATA

Pitfall : Gaps in the data

- Another example: medicine

- Missing data is always a challenge
  - but we also know that “negative results” are more likely to go missing.
  - This means we have a *biased sample*, overestimating the benefits of treatments.

- *The Information Architecture of Medicine is Broken* Ben Goldacre
  http://strataconf.com/strata2012/public/schedule/detail/22941
- https://www.youtube.com/watch?v=AK_EUKJyusg

Big Data: How to address the issues?

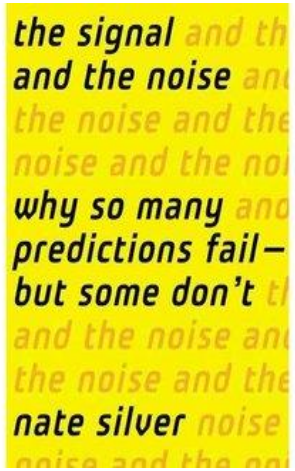
Summary of the methods

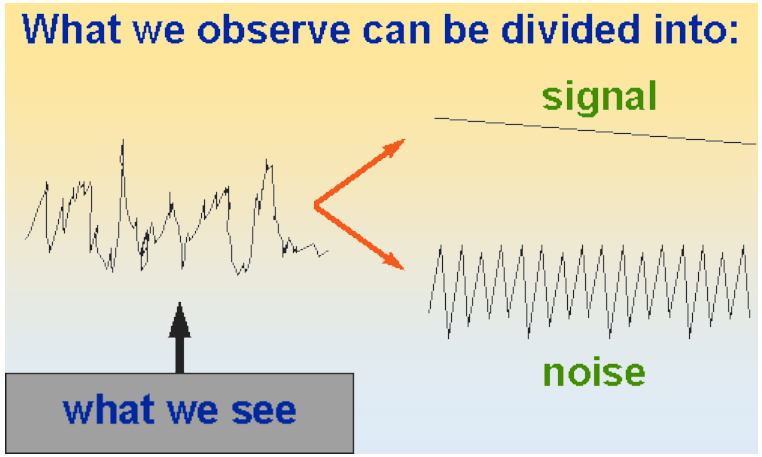
- Use domain knowledge to check model for validity
- Estimate errors

BIG DATA

Let's look to some experts

- Nate Silver book
  - The Signal and the noise
- On Time Magazine 2009 – 100 most influential people
- Correctly predict US 2008/2012 elections





BIG DATA

Example: Weather Forecasting

Why is weather forecasting very successful?

Chaotic (dynamic, non-linear system)

- Lorenz: 29.5168 instead of 29.517

Adjustment by humans

- Compute probabilities: how often predict rain, didn't rain?
- On ground reality

The effect of marketing/customer satisfaction in commercial weather forecasting.

- More sensitive about errors in predicting no rain than rain

Big Data Error Estimation

- Purely empirical: cannot be analysed by theory
- Divide data into *training set* and *testing set*
- Develop algorithm using training set; estimate error from testing set
  - Can be used to compare analytics algorithms
- Examples
  - Nate Silver: weather prediction: human adjustment
  - Amazon recommendations
    - Derive model using historical data; make recommendations
    - Get statistics on how many people look at or buy recommendations

Big Data: Summary and architecture

BIG DATA

Big Data themes

- How to manage very large amounts of data (*data management)*

- and extract value and knowledge from them (*analytics)*

- Google: store index to WWW and search

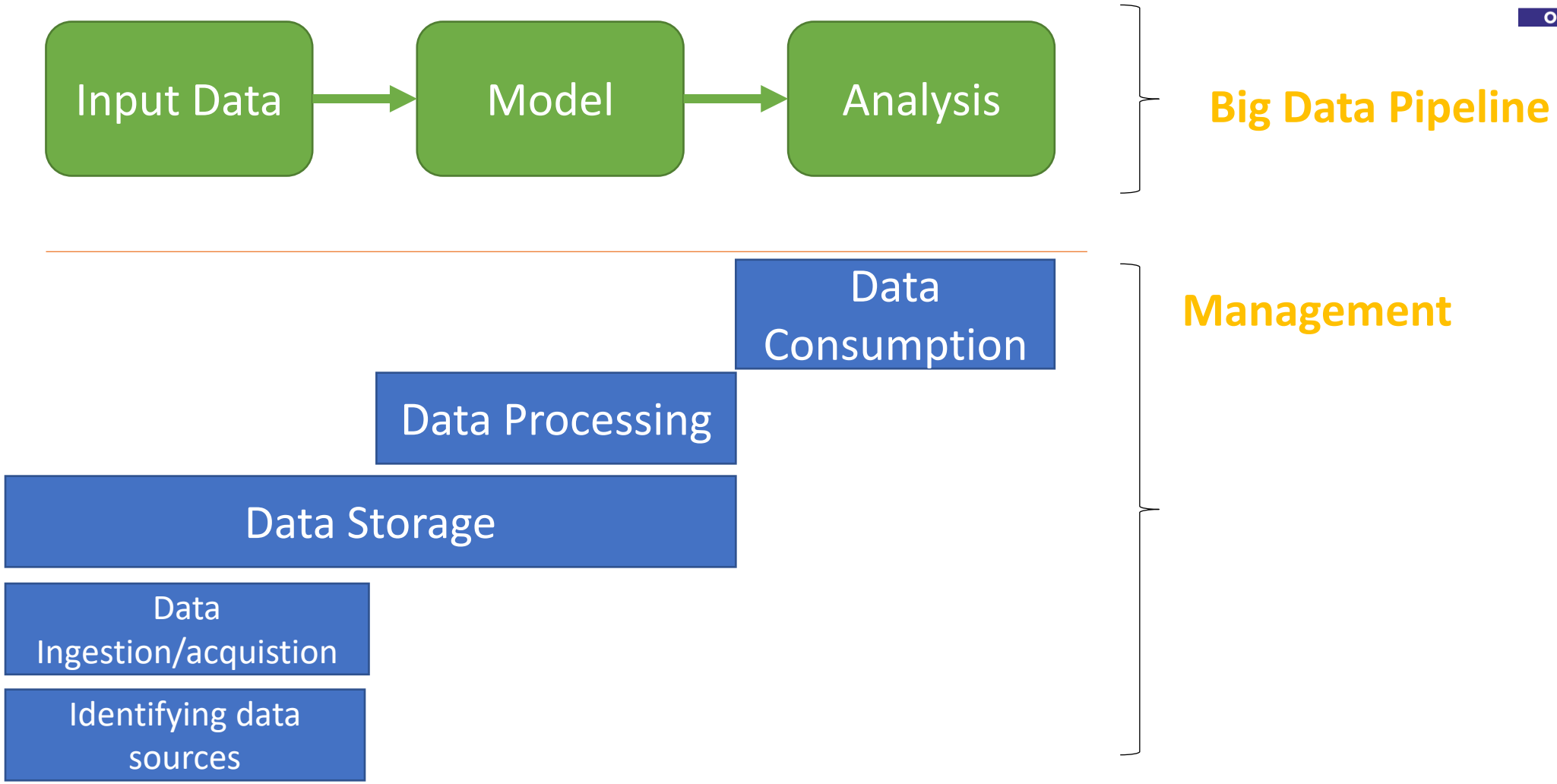
- Amazon: store user purchases and make recommendations

Large-Scale Data Management

Big Data Analytics

Data Science and Analytics

Big Data and Analytics



THANK YOU

K V Subramaniam

Department of Computer Science and Engineering

subramaniamkv@pes.edu

+91 80 6666 3333 Extn 877

Content Creators

Prof Prafullata K A

Prof Usha Devi

Prof Rachana

Prof Resma

BIG DATA

Introduction : Characteristics and Architecture

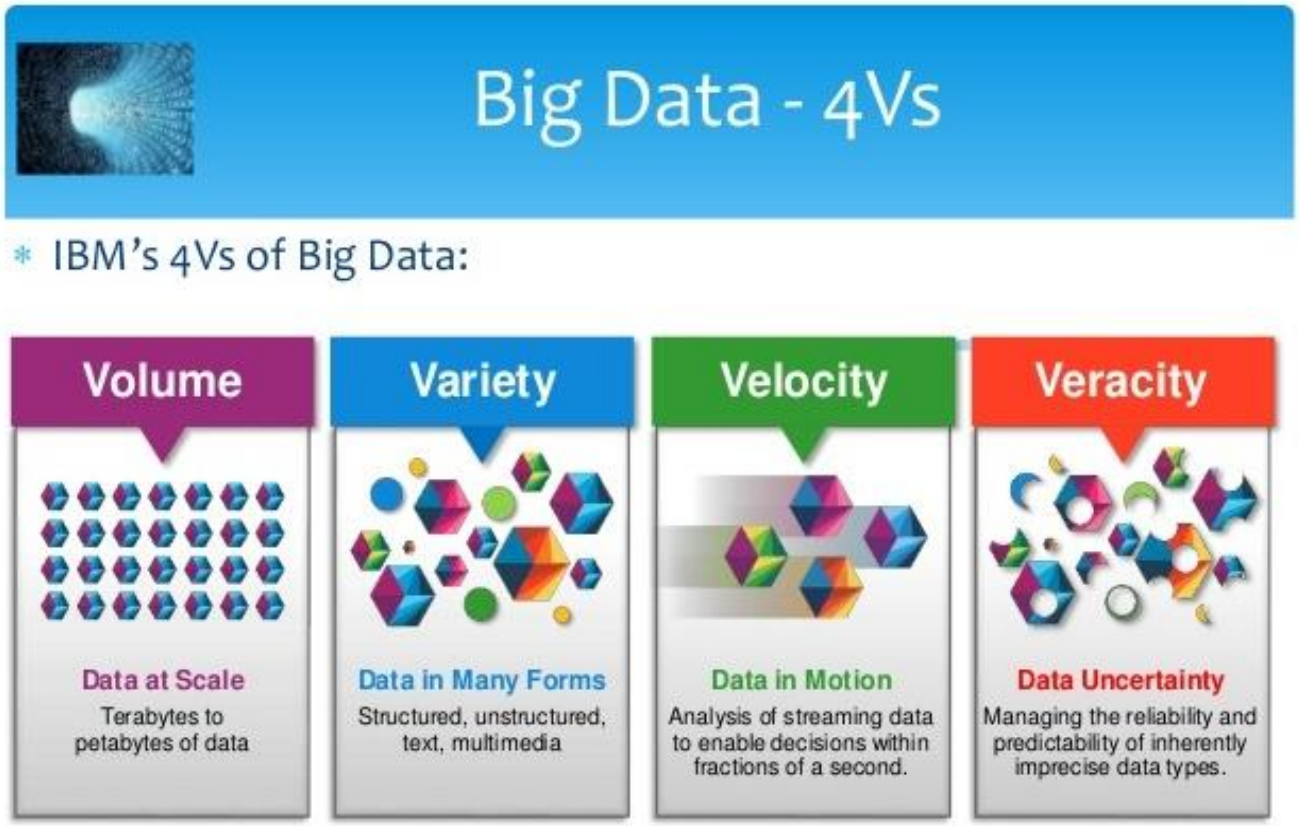
K V Subramaniam
Computer Science and Engineering
subramaniamkv@pes.edu

BIG DATA

Course Introduction Contd...

1. Big Data – Characteristics
2. Data Architecture Design
3. Data Format/Types
4. Big Data Platforms
5. Case Study - Google

BIG DATA
Characteristics – Quick Summary

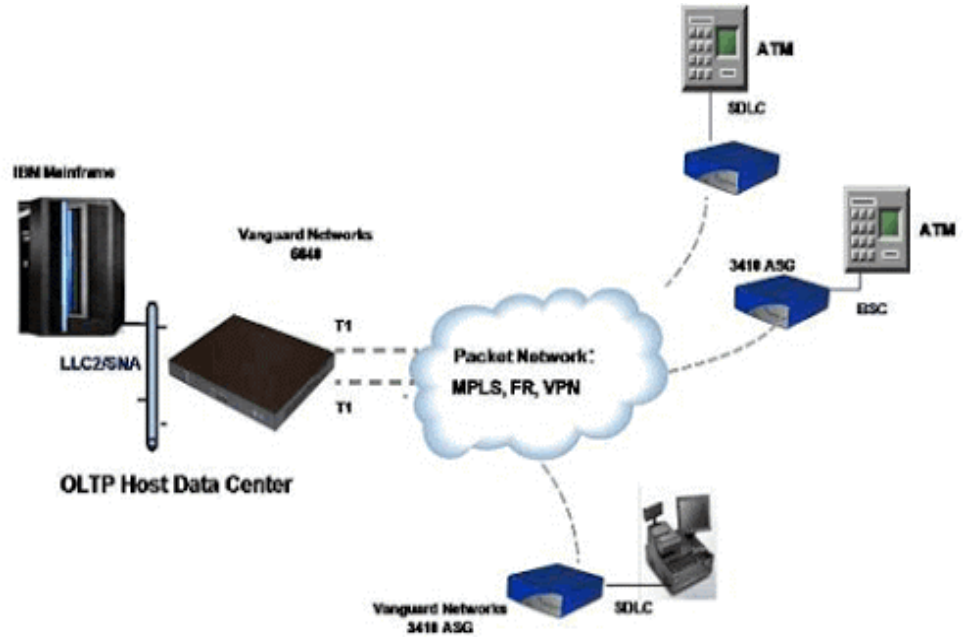


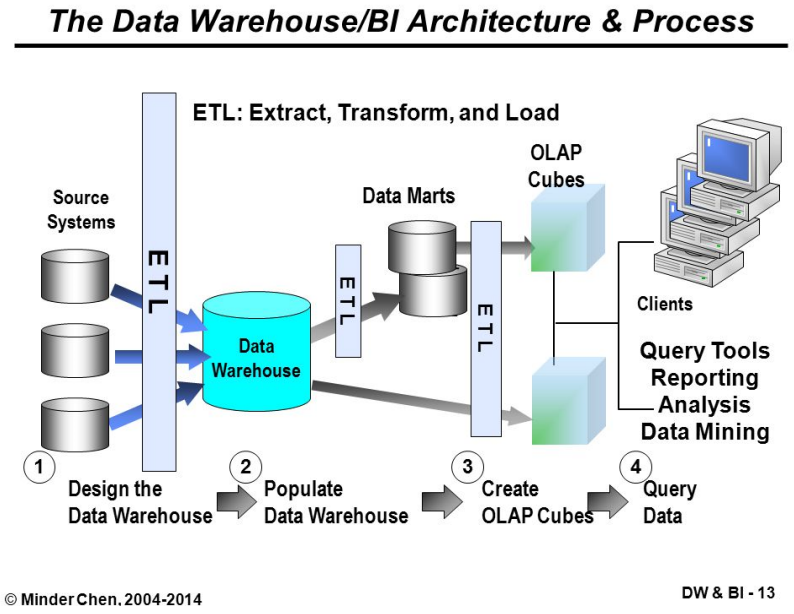
http://www.sliceofbi.com/2015/09/basics-of-big-data.html

Big Data: Characteristics - Volume

BIG DATA

Old Style Data





http://www.techbridge.solutions/home/solutions/solutions-ibm-applications/oltp-host-concentrator-solution/

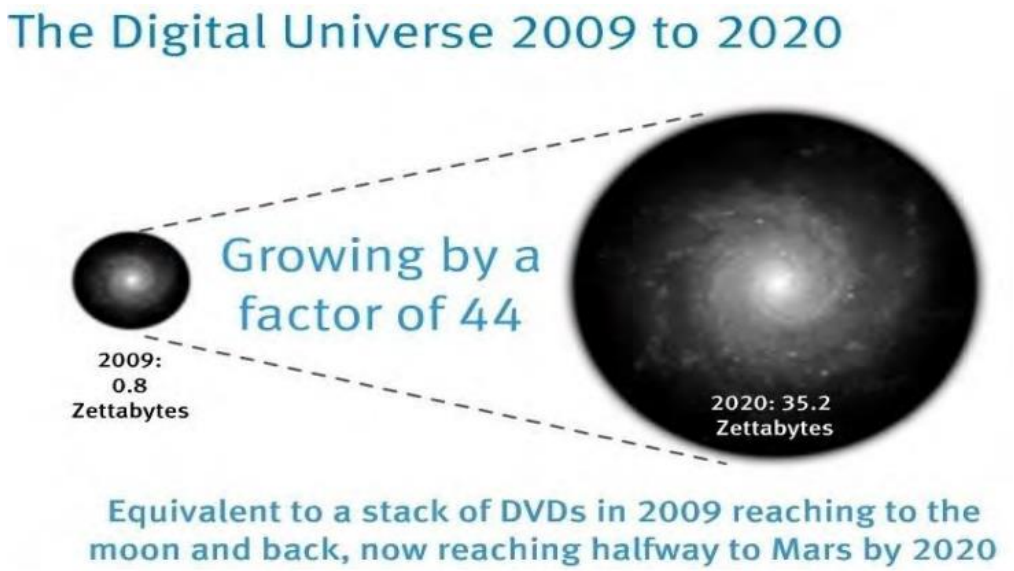
https://data-flair.training/blogs/business-intelligence-and-data-warehousing/

- Fixed schema and format
- Clean data
- Consistent
- Predictable data rates

- Elaborate ETL procedure (Not real time; once a day at best)
- Output mostly reports (Queries run in batches; take hours)

BIG DATA
Characteristics - Volume

- Old Style Data vs Big Data
  1-Scale (Volume)
  - 44x increase from 2009 2020
  - From 0.8 zetta bytes to 35zb
  - Data volume is increasing exponentially

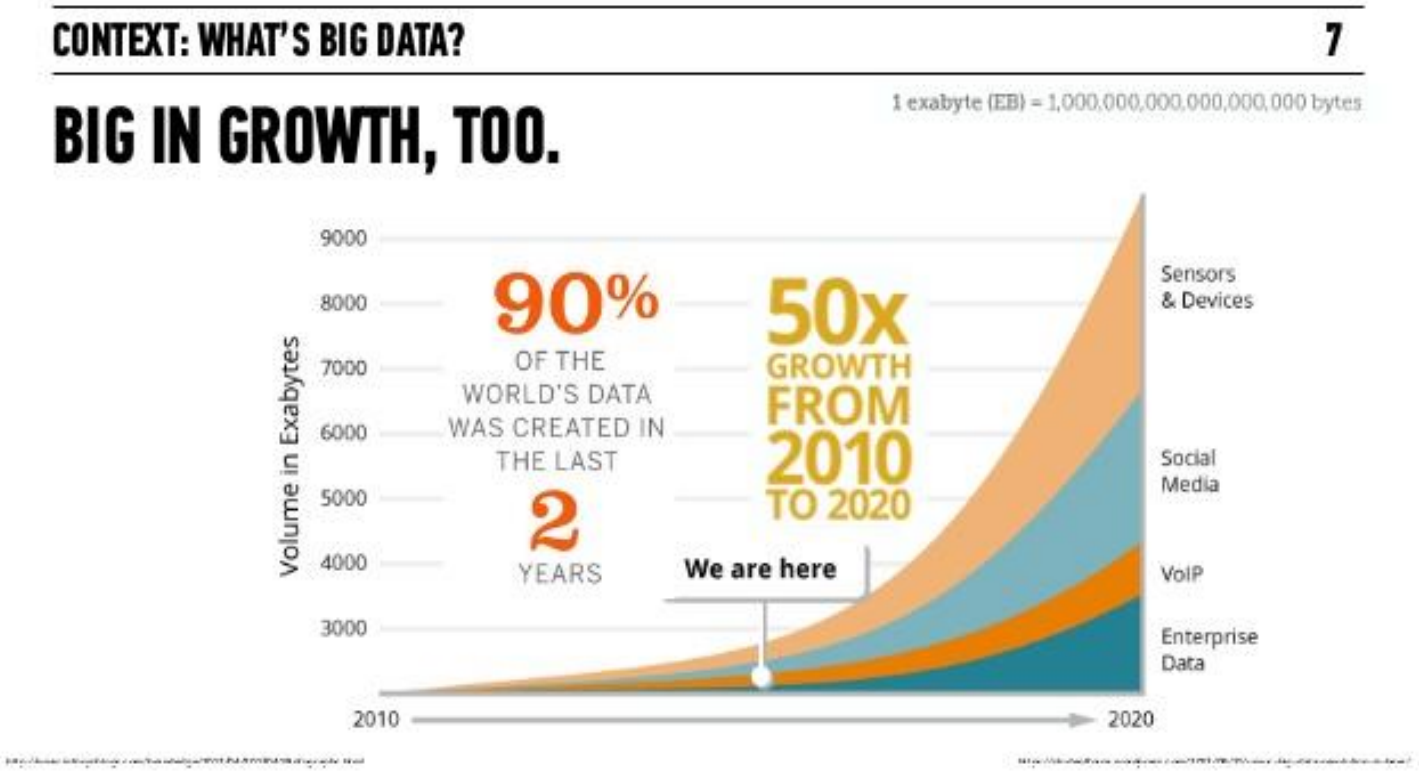


http://www.egsa.org/Portals/7/Documents/CommitteeDocs/Misc/F15_Data_Center_Market.pdf

BIG DATA

Characteristics - Volume

- Why is the volume so high? Who's Generating it?



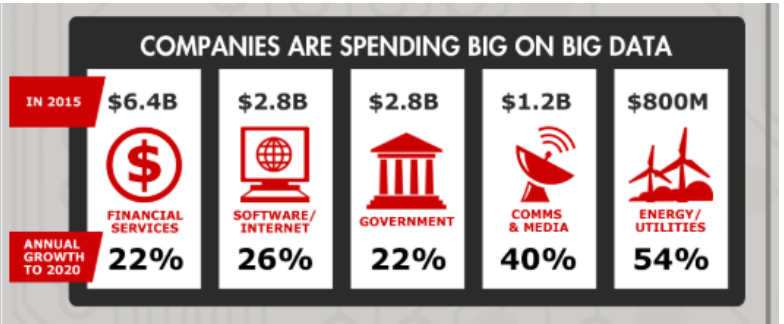
Old Style Data	Big Data
Enterprises only	Everybody

https://www.promptcloud.com/blog/want-to-ensure-business-growth-via-big-data-augment-enterprise-data-with-web-data/

BIG DATA
Characteristics - Volume

Old Style Data	Big Data
Manual entry	Automated

- How this VOLUME gets collected?
  - Financial Services
    - Stock trading, banking, financial services are all computerized (In India – no paper shares any more)
    - Every transaction is recorded
  - Energy / Utilities
    - Can put sensors in every home
    - Smart grid
  - Comms & media
    - Internet services



https://www.forbes.com/sites/louiscolumbus/2014/06/24/roundup-of-analytics-big-data-business-intelligence-forecasts-and-market-estimates-2014/#26c06d11388e

Big Data: Characteristics - Variety

BIG DATA
Characteristics - Variety

- Why is analyzing Big Data *complex*?
- Various formats, types, and structures
- Text, numerical, images, audio, video, sequences, time series, social media data, multi-dim arrays, etc…
- Static data vs. streaming data
- A single application can be generating/collecting many types of data

Old Style Data	Big Data
Fixed format / schema	Integrate Twitter, Maps, Facebook…

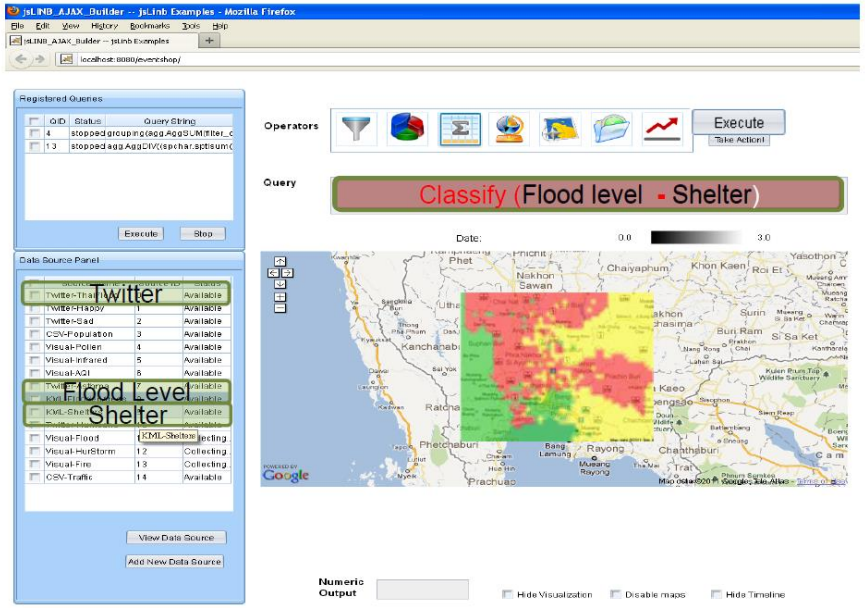
To extract knowledge ➔ all these types of data need to linked together

https://twitter.com/maritime_uk/status/1072159378931879937

BIG DATA

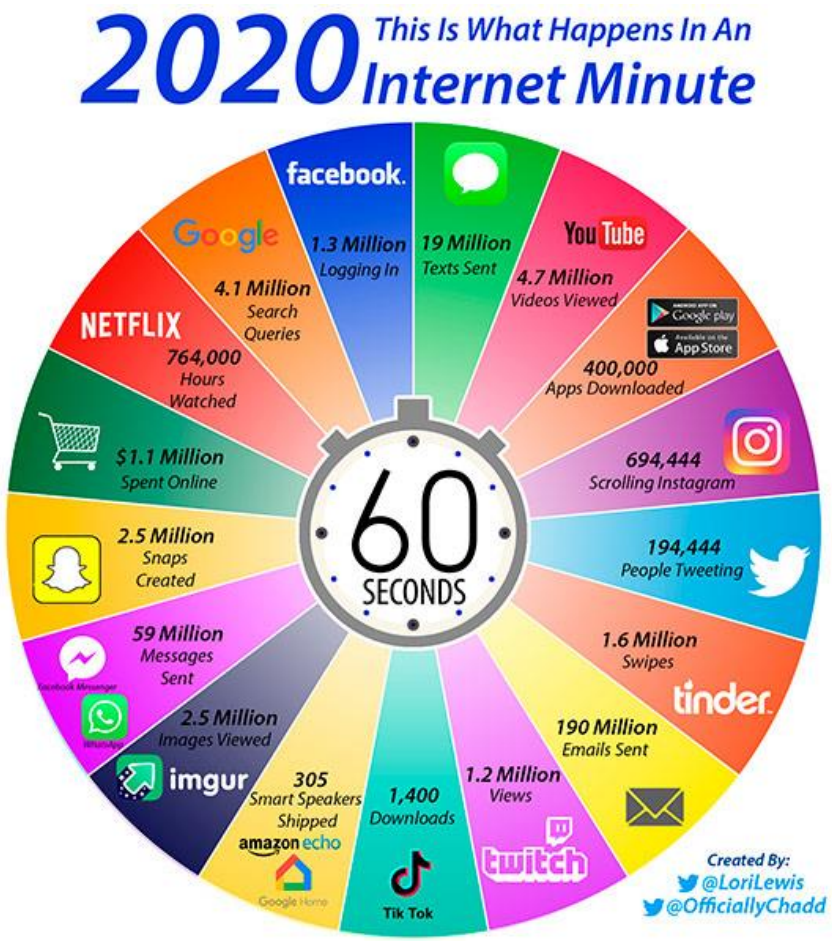
Characteristics - Variety

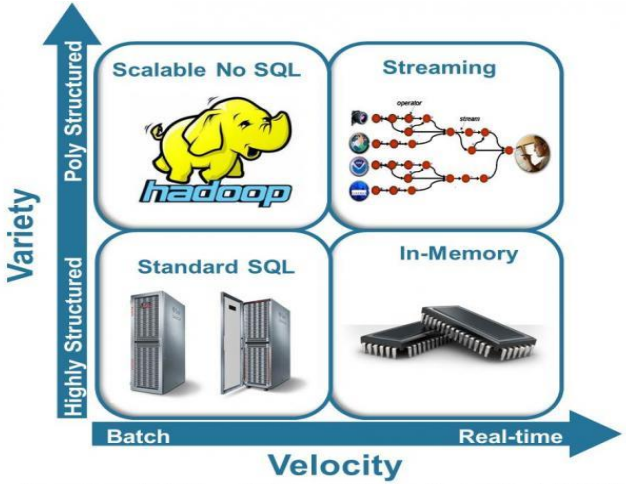
EventShop: Thai Floods



Big Data: Characteristics - Velocity

BIG DATA
Characteristics - Velocity





Source: Forrester Webinar: Big Data: Gold Rush or Illusion?, Sept 19, 2013

https://go.forrester.com/blogs/14-05-27-boost_your_digital_intelligence_with_big_data/

https://www.allaccess.com/merge/archive/31294/infographic-what-happens-in-an-internet-minute

BIG DATA
Characteristics - Velocity

Nokia launches real-time mobile network analytics platform

Staff writer | July 28, 2016 | Telco Analytics Asia

https://www.nokia.com/about-us/news/releases/2016/06/30/nokia-real-time-mobile-network-analytics-provides-instant-correlation-of-network-wide-data-and-its-impact-on-customer-experience/

The analytics link the performance of applications and devices to network issues in real-time, effectively making every mobile device part of a network test bed.

This enables operators to pinpoint potential causes of service degradation much more rapidly than they can today since they no longer need to consult a myriad of tools and correlate the data from them manually. It also offers engineering teams a proactive way to understand over the top (OTT) application impacts on the network and optimize opportunities.

Old Style Data	Big Data
Input data at night	Real-time input
Output daily report	Real-time response

Big Data: Characteristics - Veracity

Characteristics - Veracity

Veracity

refers to the messiness or trustworthiness of the data.

Accuracy cannot be controlled due to

- Hashtags
- Typos
- Abbreviations

Need to work with such data

Traditional: manually entered, fixed fields, less chance of error

Old Style Data	Big Data
Clean data	Inconsistent data

BIG DATA
Characteristics - Veracity

- How people get multiple PAN cards

- Slight changes in name and address
  PES University, 100 ft Road…
  PES university, BSK III stage...
  Pes University, Dwaraka Nagar...

- Names and addresses don't match

- But postman / courier will deliver to correct place!!!

Big Data: Data Architecture Design

BIG DATA
Data Formats / Types

Structured Data

0.103	0.176	0.387	0.300	0.379
0.333	0.384	0.564	0.587	0.857
0.421	0.309	0.654	0.729	0.228
0.266	0.750	1.056	0.936	0.911
0.225	0.326	0.643	0.337	0.721
0.187	0.586	0.529	0.340	0.829
0.153	0.485	0.560	0.428	0.628

Databases

Unstructured Data

Documents, Tweets, Videos

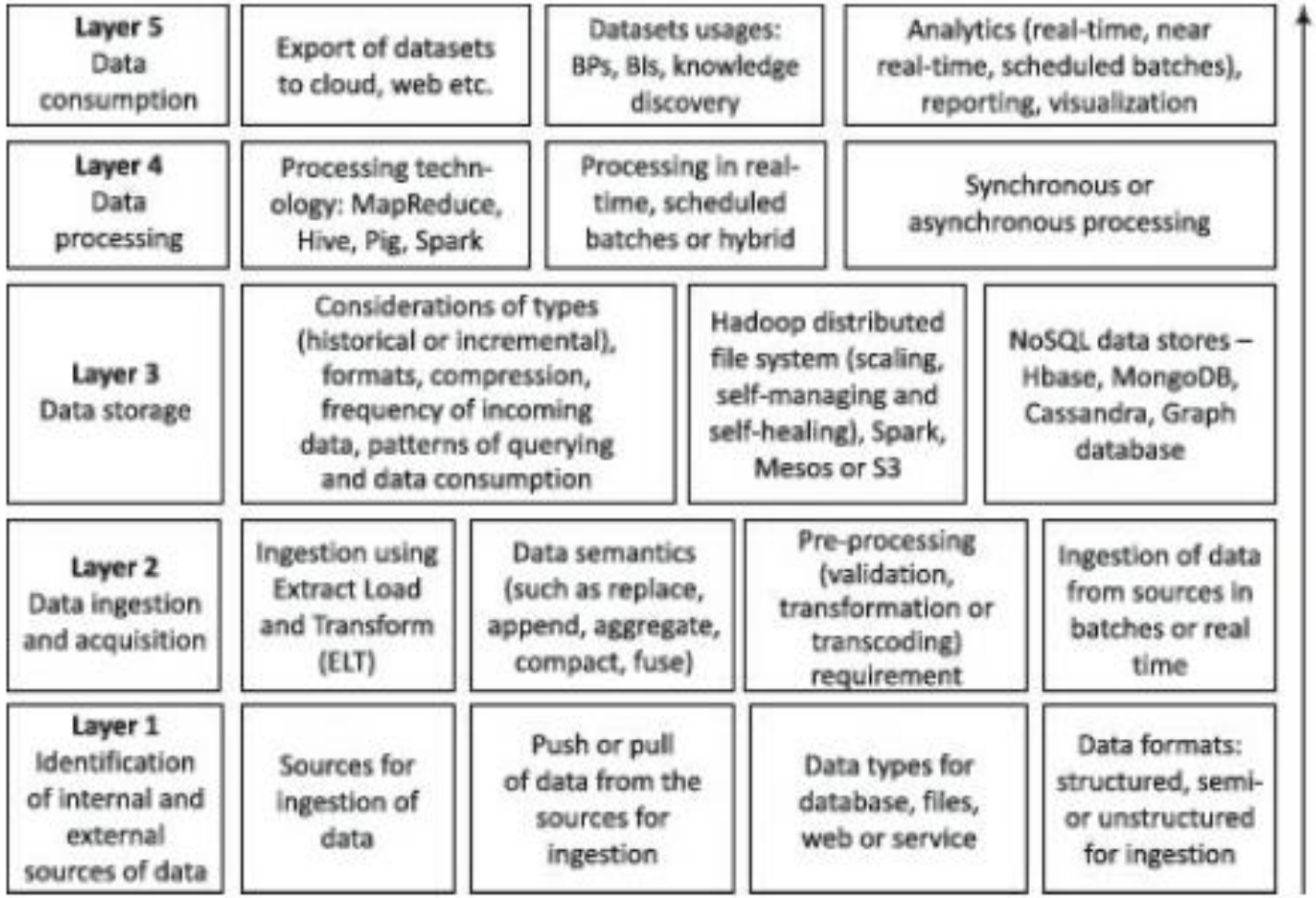
Semi Structured Data

Emails, XML

http://biisfun.blogspot.com/2016/03/big-unstructured-data-vs-structured.html

BIG DATA

Data Architecture Design

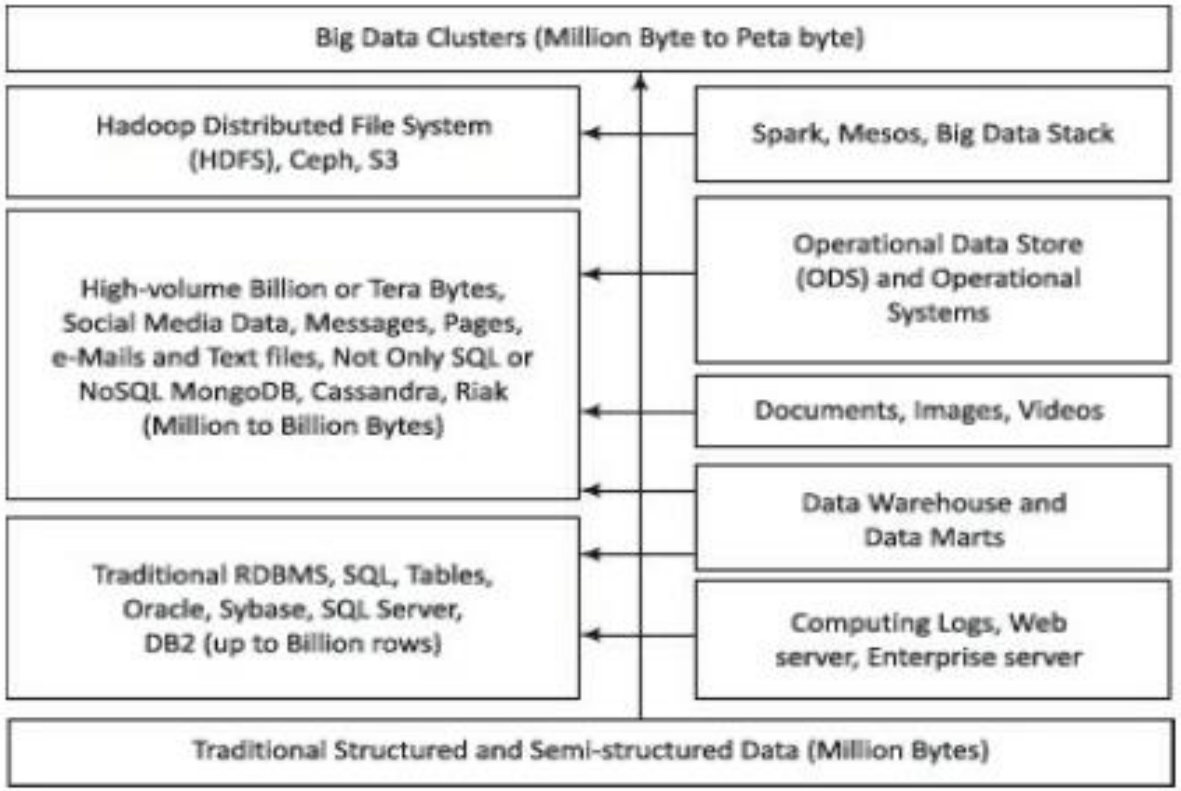


T1: Fig 1.2 – Logical layers in data processing architecture and their functions.

BIG DATA

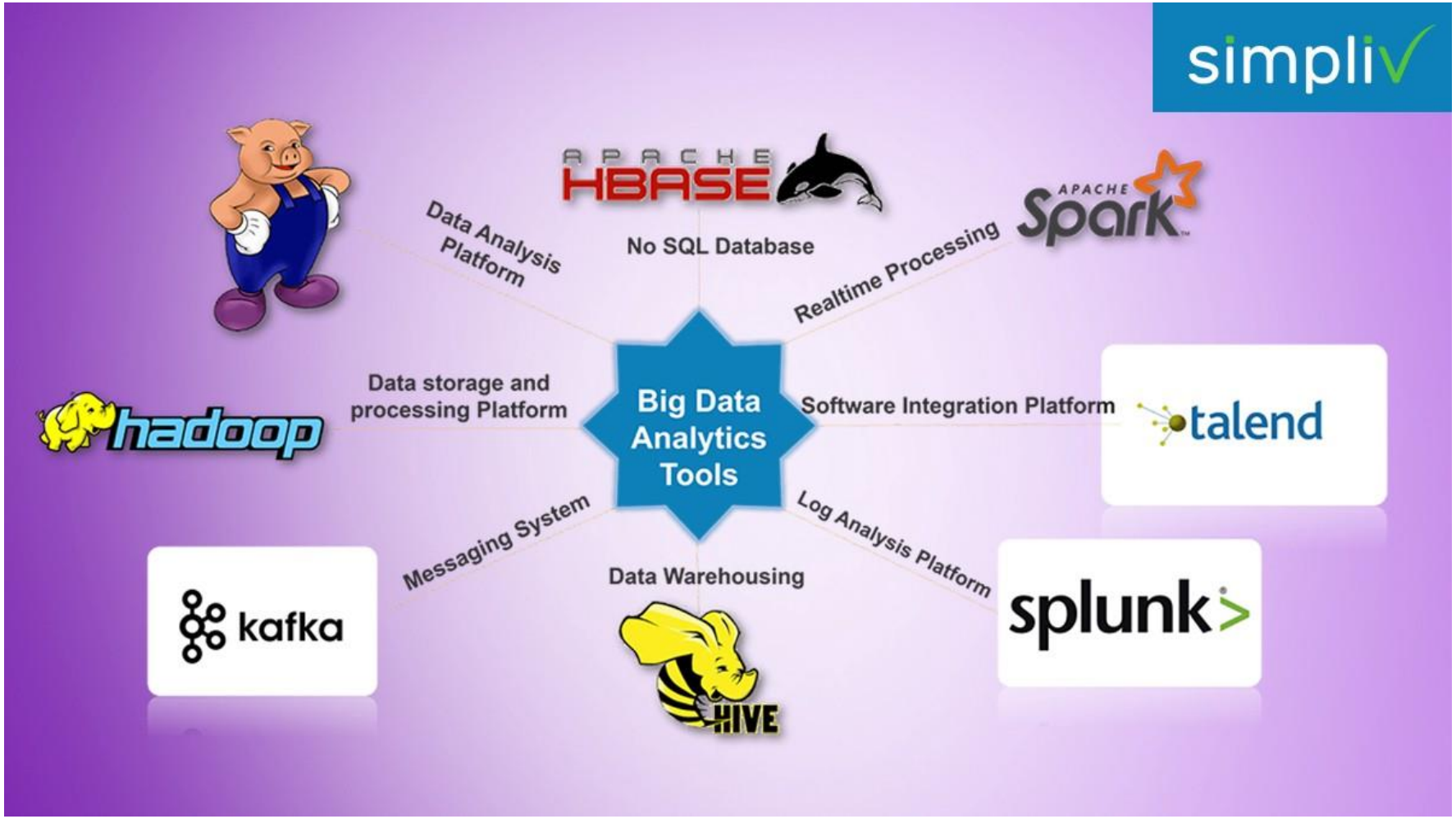
Data Formats / Types : Big Data Storage

- Data storage for traditional and Big Data



T1: Fig 1.7 – Big data storage plan – RDBMS and NoSQL together

BIG DATA
Big Data Platforms

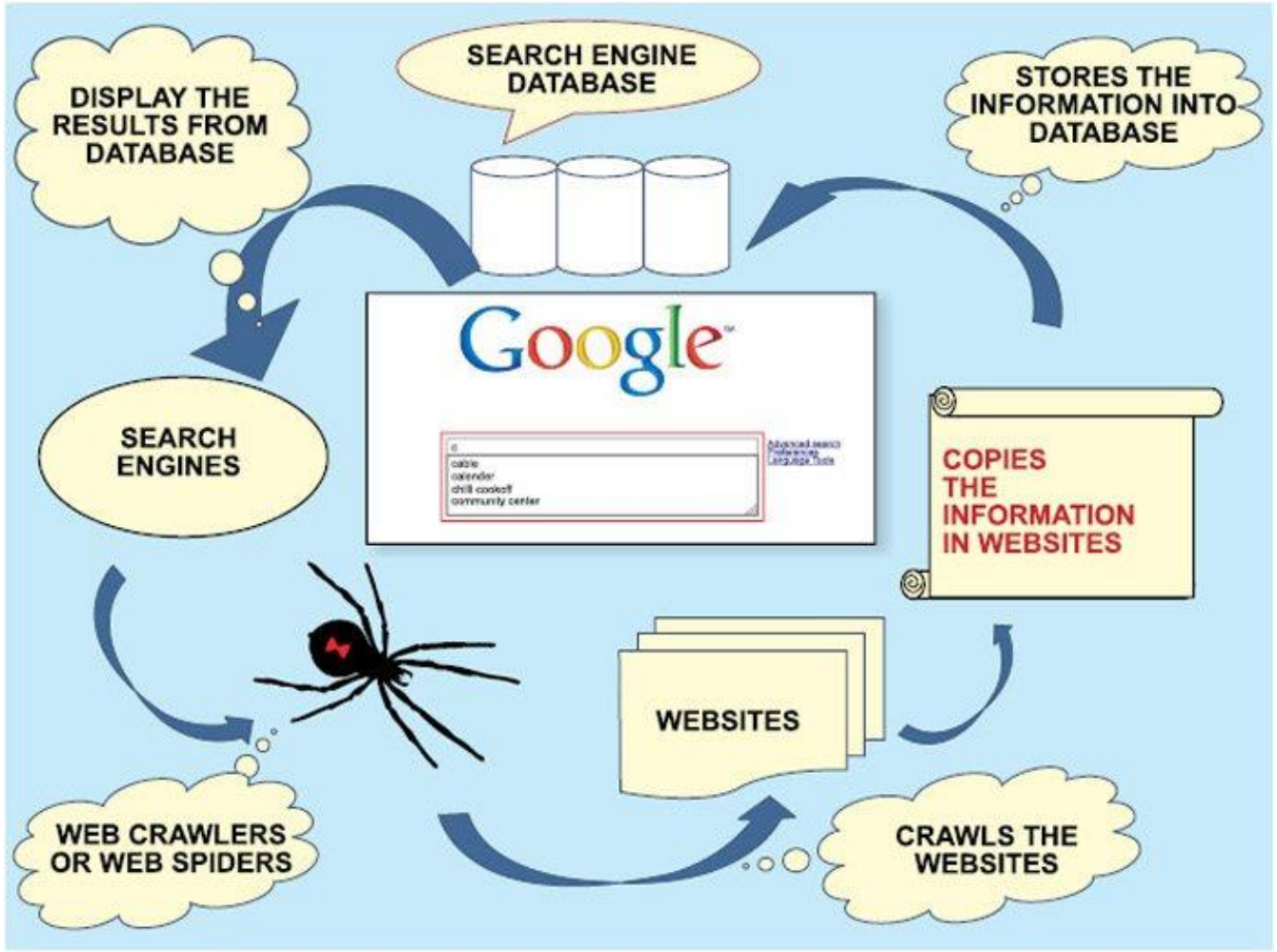


https://medium.com/@satyanageshan8/5-prominent-big-data-analytics-tools-to-learn-in-2019-ee1d4f5b98ce

Big Data: Case Study – Google Search

BIG DATA

Case Study: How Google Search works?



https://www.electronicsforu.com/resources/how-google-search-engine-works

BIG DATA

Interesting facts about Google Search

https://en.wikipedia.org/wiki/PageRank

- Google initial implementation
  - 24 M web pages, 322 M links
  - 5 days to compute Page Rank
    - Page rank is proportional to the popularity of the page
    - If more links point to a page, that page will be more popular
- Page Rank – Treats the web as a graph
  - User -
    - Starts at a random page
    - Takes a random link
  - Page Rank Assumptions –
    - The more popular a page - The better is its quality

THANK YOU

K V Subramaniam
Computer Science and Engineering

subramaniamkv@pes.edu

+91 80 6666 3333 Extn 877

Big Data
HDFS

K V Subramaniam
Computer Science and Engineering

BIG DATA

Lecture overview - HDFS

- Need for Distributed file systems
- HDFS Introduction
- Architecture
- Operations
- Internals
- Fault Tolerance and replication
- Blocks

Big Data: File systems and distributed file systems

BIG DATA

Data Growth – Why the need for HDFS?

As per RBI in May 2019,

#credit/debit card transactions~ 1.3 Billion
(https://rbidocs.rbi.org.in/rdocs/ATM/PDFs/ATM052019E96EC259708C4ED9AD9E0C6B5E8B6DD5.PDF)

If each transaction requires about 10K of data

13 TB of data

Lot of data and this is only for credit/debit card transactions

There are other transactions also

Suppose you want to look for fraudulent transactions

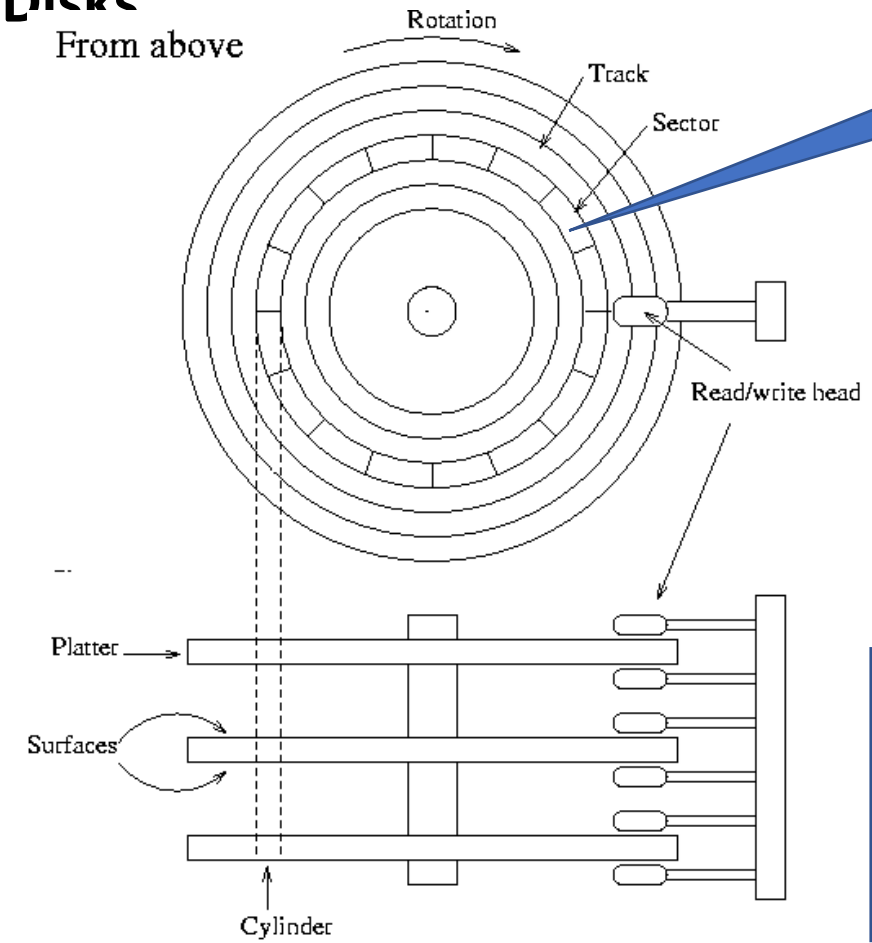
How to store and process this data?

This class

Image Courtesy: https://www.indiamart.com/proddetail/big-data-analytics-service-6764825662.html

Disk Storage – persistent storage

Persistent Storage - Disks



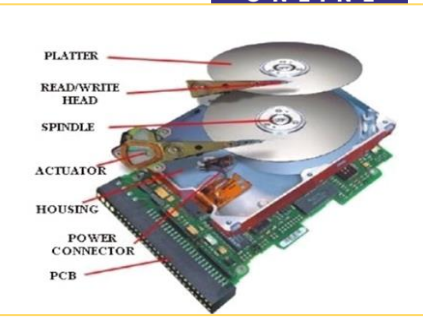
Block oriented device : storage divided into fixed size blocks

Can we store the persistent data directly on these blocks?

Who will maintain the meta-data?

Concerns: Which block contains the data as files are not necessarily multiples of blocks size

A filesystem layer on top of disk manages blocks and maps data/metadata to blocks



BIG DATA

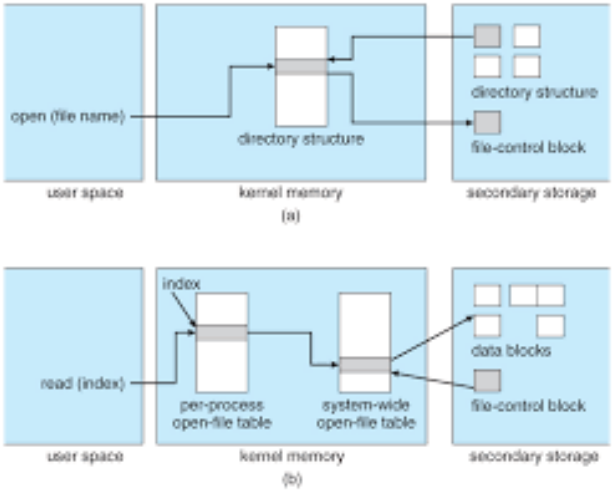
HDFS – Hadoop distributed File system

FILES

- Named collection of related information
- on disks

Desirable properties of files

- Long-term existence
- Sharable between processes
- Access permissions



https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/12_FileSystemImplementation.html

BIG DATA

Distributed File System

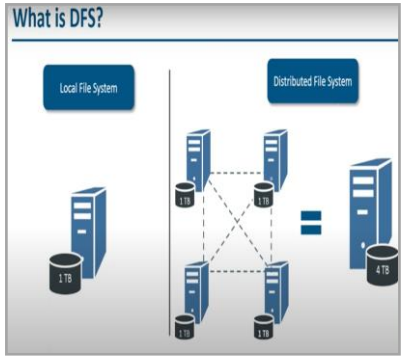
Consider case when data is so large that it cannot fit on a single disk.

- A DFS manages files and folders across multiple computers.

DFS can organize and display the files as if they are stored on one computer.

- It serves the same purpose as a traditional file system.

Designed to provide file storage and controlled access to files over local and wide area networks.



BIG DATA

Exercise

Consider that you have 1TB of data?
Compare the time taken to read data in both the cases below

- single machine (*4 I/O channels each channel 100mb/s)*
- 10 machines (*each having 4 I/O channels each channel 100mb/s*).

Big Data: HDFS Introduction

BIG DATA

Exercise Solution

Consider that you have 1TB of data?

Compare the time taken in both the cases below

- single machine (*4 I/O channels each channel 100mb/s)*
- 10 machines (*each having 4 I/O channels each channel 100mb/s*).

HDFS – History

HDFS – Inspired by GFS

GFS – Google File System (2003)

Distributed File system on a cluster of machines

Developed by Doug Cutting and Mike Cafarella

Origin - Apache Nutch

- Goal : web search engine on 1 Billion Pages

Open source

HDFS – Hadoop distributed File system

"HDFS is a filesystem designed for storing very large files with streaming data access patterns, running on clusters of commodity hardware."

Very large

- Files can be MB/GB/TB in size
- Hadoop clusters that are PB are currently operational

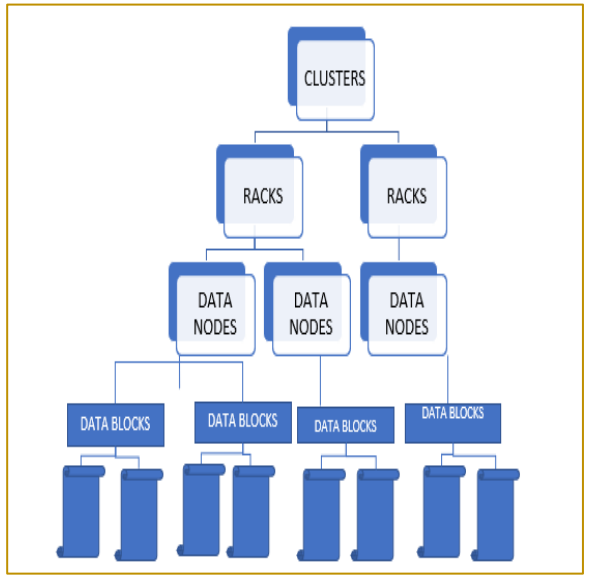
Read Mostly data

- most efficient data processing pattern is a write-once, read-many-times pattern.
- Each analysis will involve a large proportion of the dataset
- time to read the whole dataset is more important than the latency in reading the first record.

Commodity hardware

- Hadoop doesn't require expensive, highly reliable hardware
- Designed to run on clusters of commodity hardware

HDFS – Hadoop distributed File system



https://medium.com/@Meela349588204/rack-server-vs-blade-server-d1b5aee58ca2

Exercise

- If you want to store a file on disk what constitutes
  - Data
  - Metadata
- What are their access patterns?
  - How often do you think each one will be accessed during a normal file read
- How large are they (comparatively)? Why is this important?

Big Data: HDFS Architecture

Exercise Solution

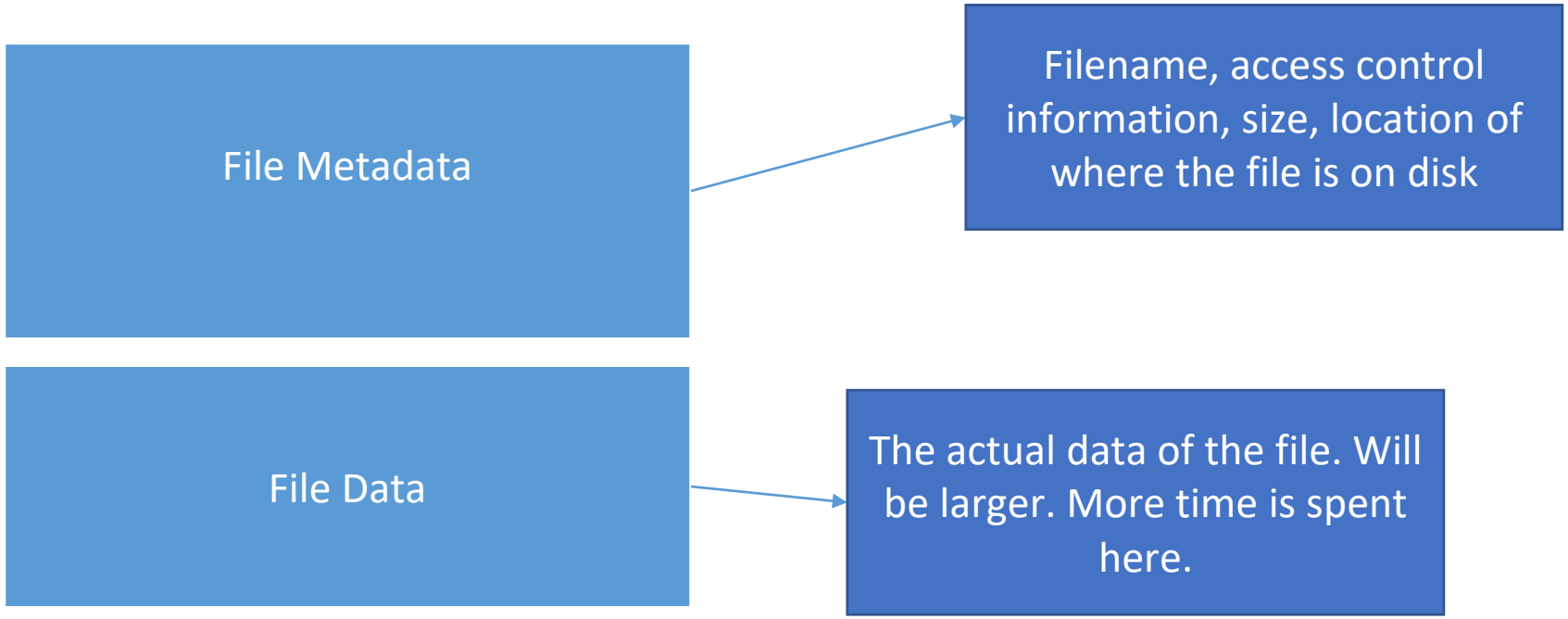
- If you want to store a file on disk what constitutes
  - Data
  - Metadata
- What are their access patterns?
  - How often do you think each one will be accessed during a normal file read
- How large are they (comparatively)? Why is this important?

Data: much larger in size. Occupies multiple blocks

Metadata – smaller compared to data
Only information on filenames and blocks it occupies

Since data is much larger. Most time spent in fetching data.

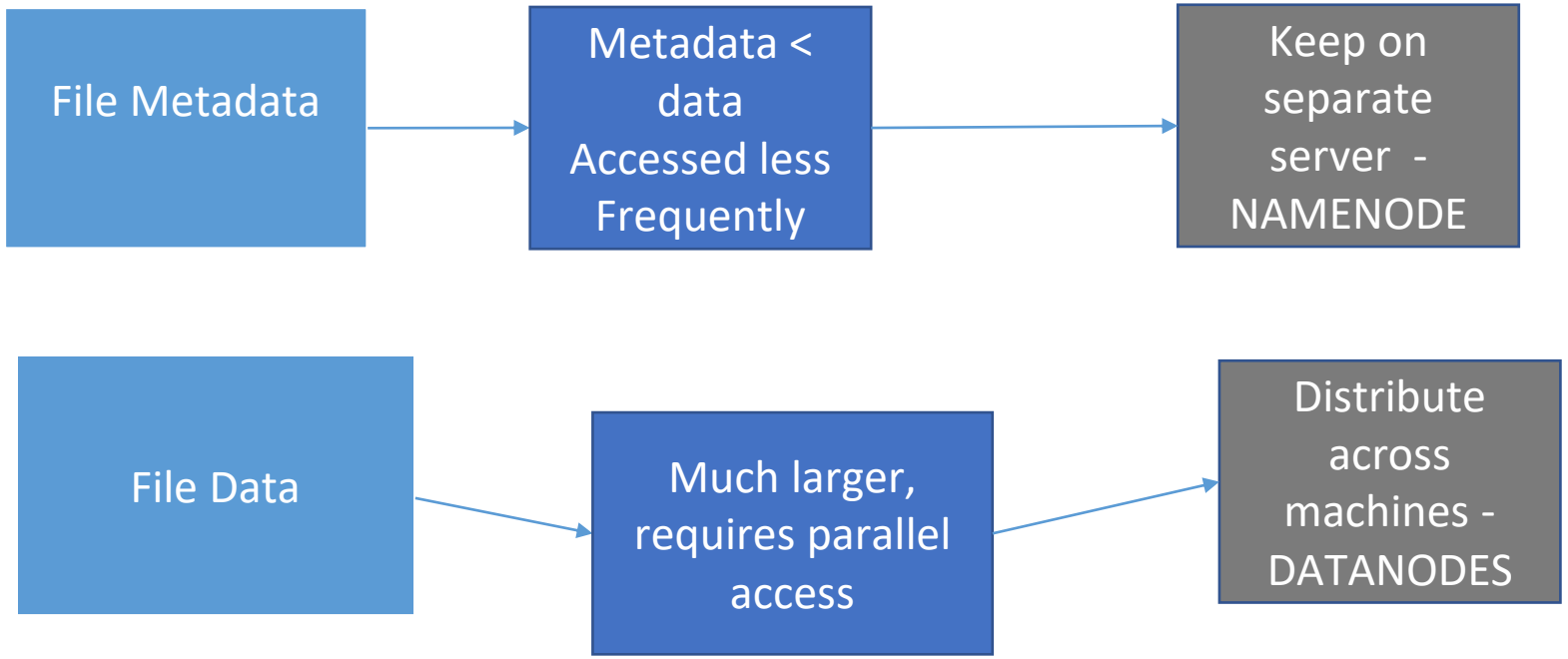
HDFS – Architecture Motivation



BIG DATA

HDFS – Architecture Motivation

Solution



BIG DATA

HDFS – Master Slave Architecture

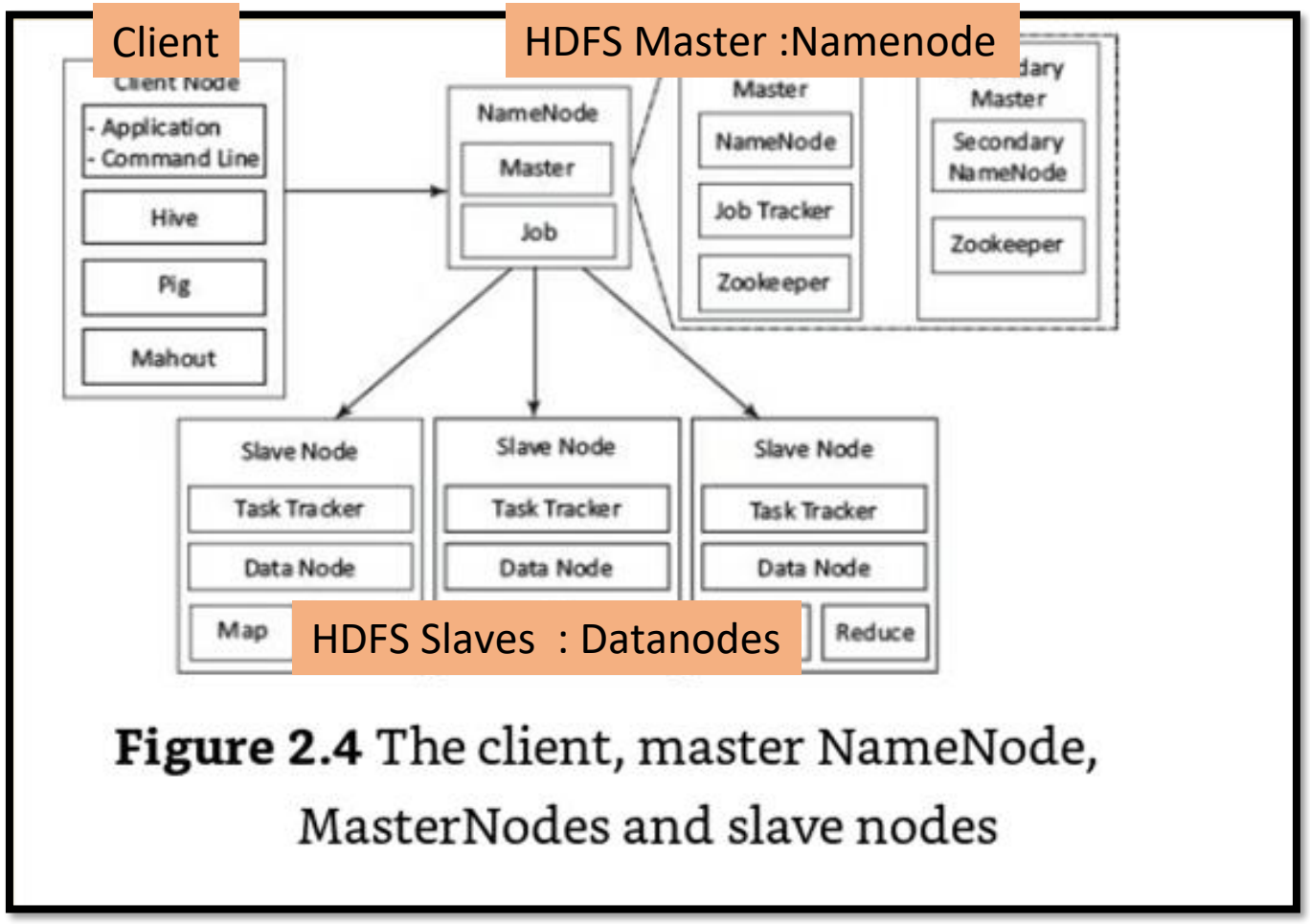


Figure 2.4 The client, master NameNode, MasterNodes and slave nodes

Diagram Courtesy: Big Data Analytics (Kindle Location 2502). McGraw Hill Education. Kindle Edition.

Big Data: HDFS Operations

BIG DATA

HDFS – Writing a file

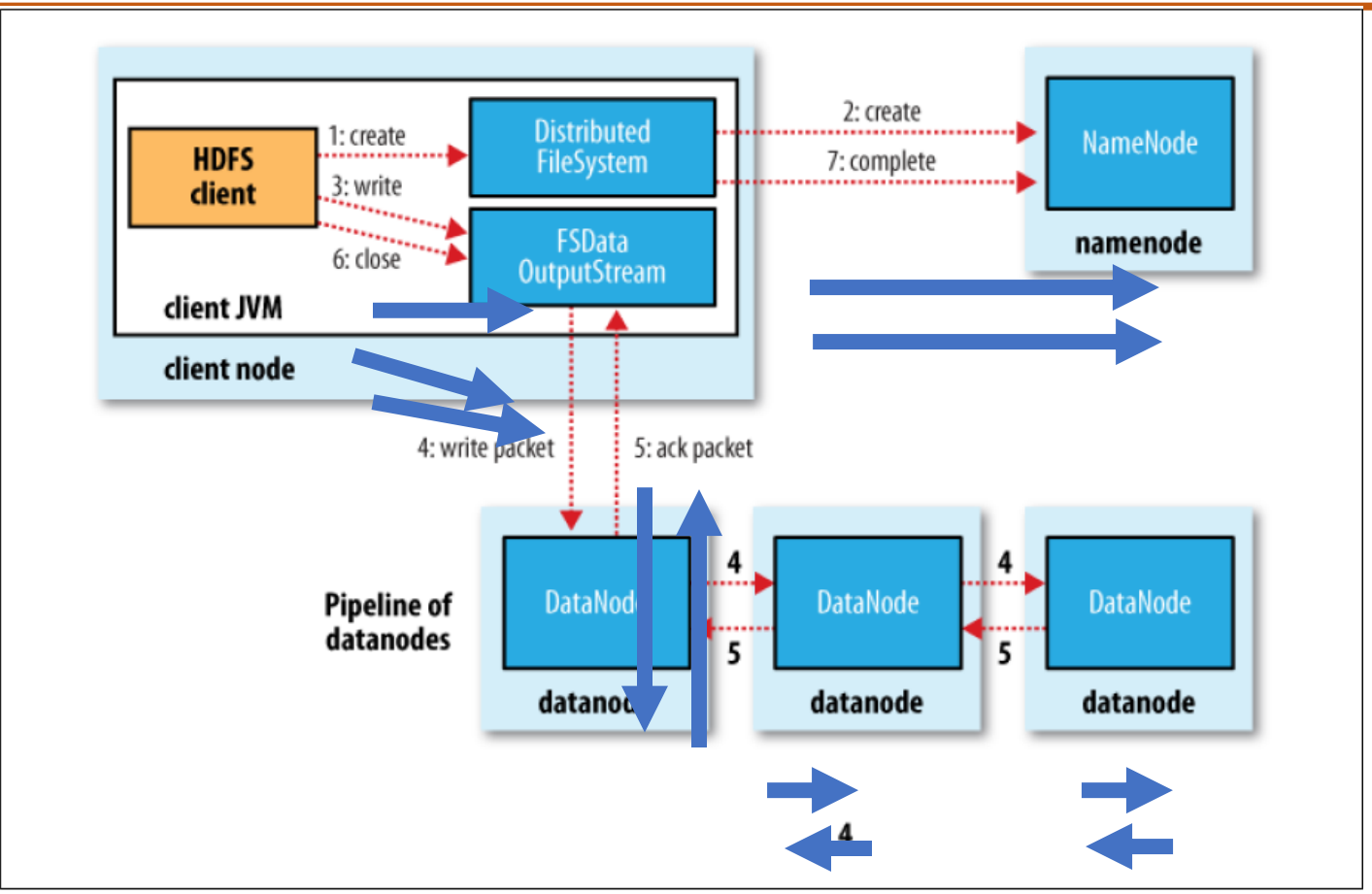
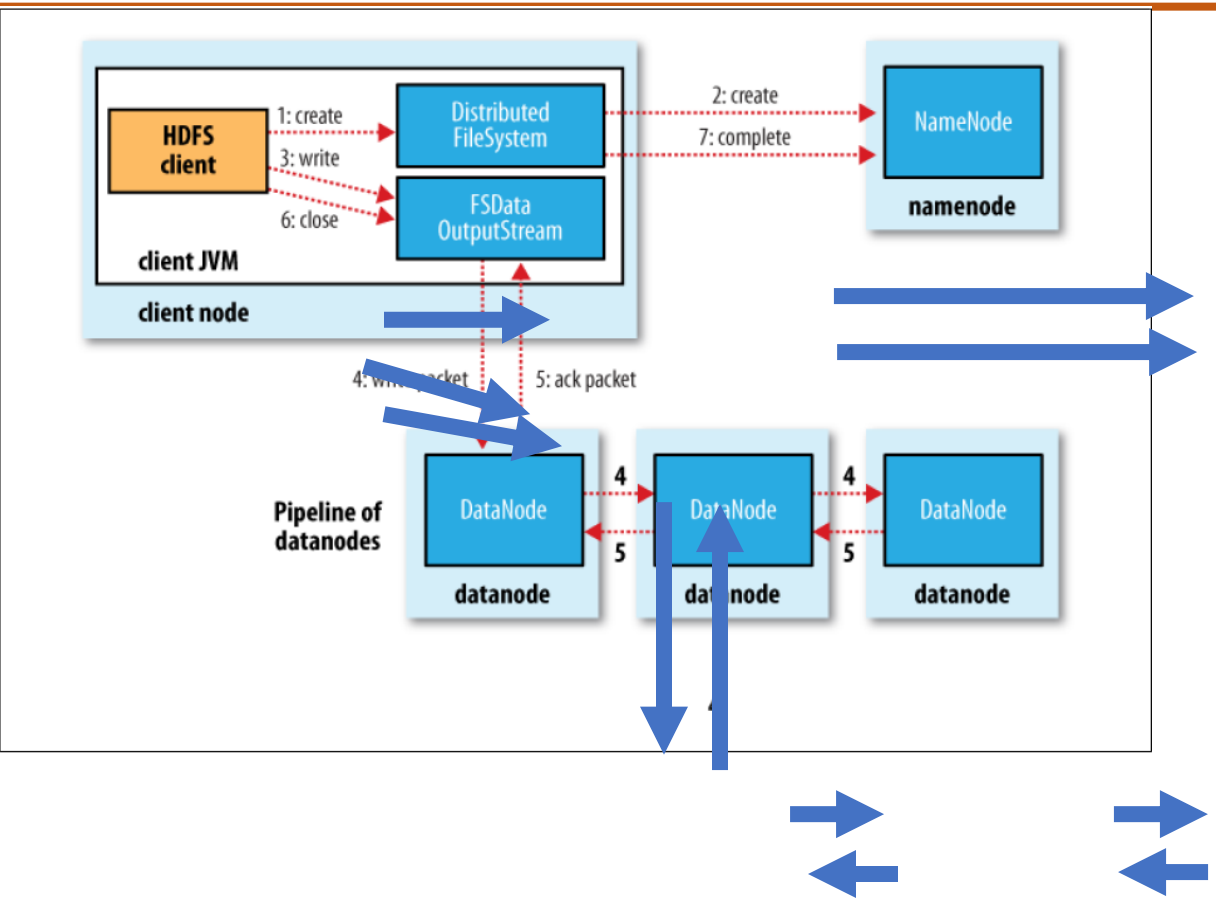


Image source: Hadoop Definitive Guide, Tom White, O'Reilly

BIG DATA

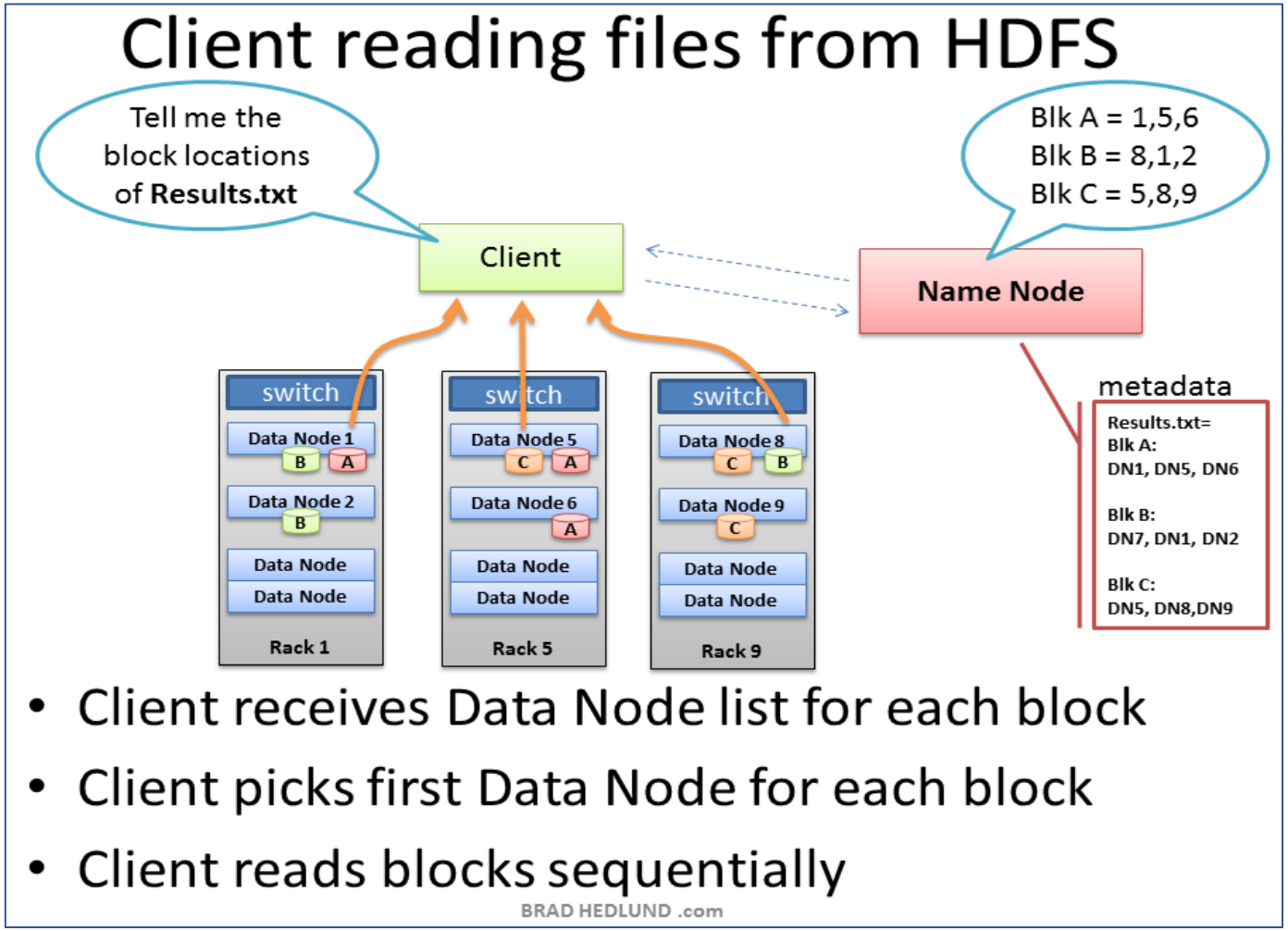
HDFS – Writing a file



- The write operation is outlined in the figure
- Can you outline the steps in the read operation?

Image source: Hadoop Definitive Guide, Tom White, O'Reilly

HDFS: File reading

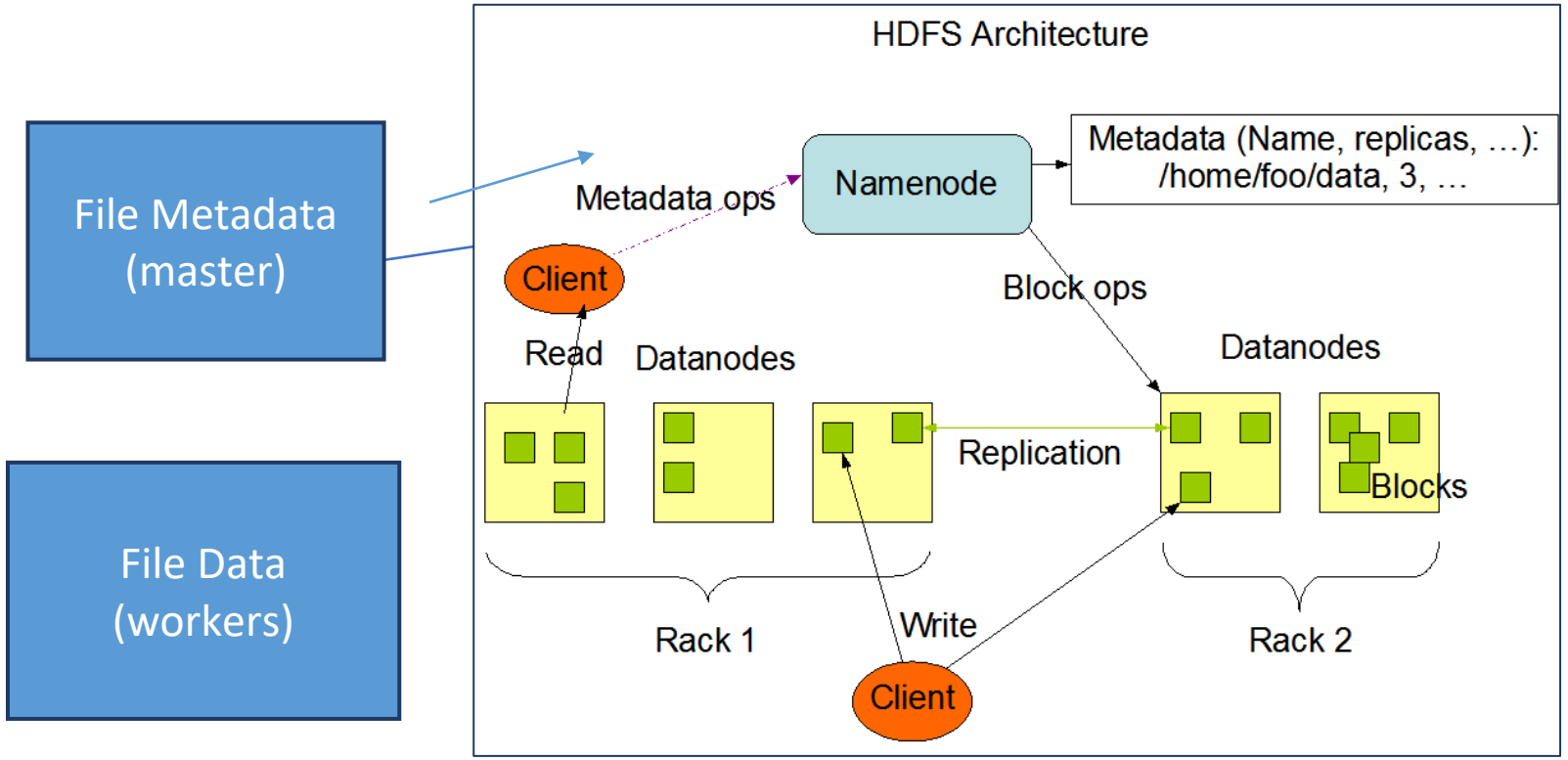


http://bradhedlund.com/2011/09/10/understanding-hadoop-clusters-and-the-network/

Big Data: HDFS Internals

HDFS – Hadoop distributed File system

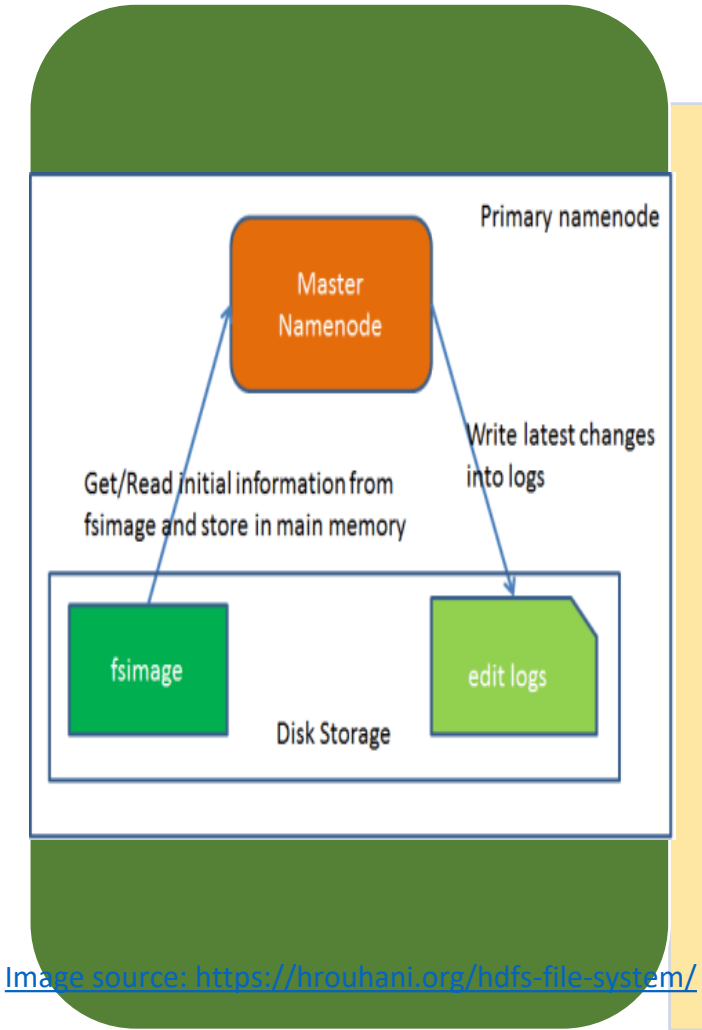
HDFS Architecture



https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

BIG DATA

Namenode features



- Manages file system namespace
- Regulate access to files by client
- Opening closing and renaming files
- Manages block creation, deletion and replication
- Determines mapping of blocks to data nodes
- Handles block failure
- Transaction Log
- Contains the metadata in memory

Image source: https://hrouhani.org/hdfs-file-system/

Namenode organization

- FSImage
  - Serialized form of filesystem tree
  - Not updated on every write
    - To avoid recopy of data
  - Stores
    - Filename
    - access time
    - #blocks
    - blocks consisting the file

- Edit Log
  - Every write first writes to edit log
  - Flushed and synced after every transaction
  - Append only operation

Since no modify operation is done on either file, it can be done really fast.

Namenode memory requirements

- Stored in memory
- What takes up space?
  - #blocks/file
  - Filename length
  - #directories
- Limited amount of memory
- Rule of thumb – 1000MB per million blocks
- Solution:
  - Limit the responsibility of each node

- Example calculation
  - 200 node cluster
  - 24TB/node
  - 128MB block size
  - Replication factor = 3
- How much space is required?
  - #blocks = 200*24*2^20/(128*3)
  - ~12Million blocks
  - ~12,000 MB memory.

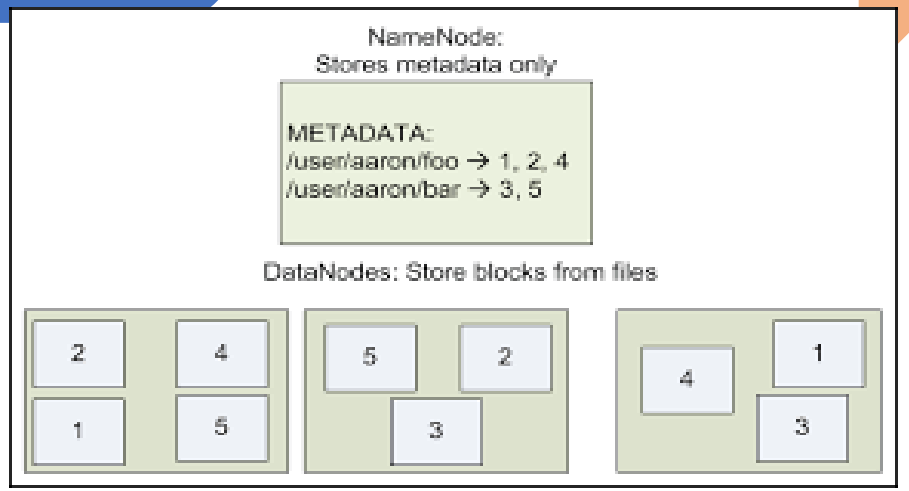
BIG DATA

HDFS – Data Nodes

Have lots of disk storage and moderate amounts of processing capabilities and DRAM

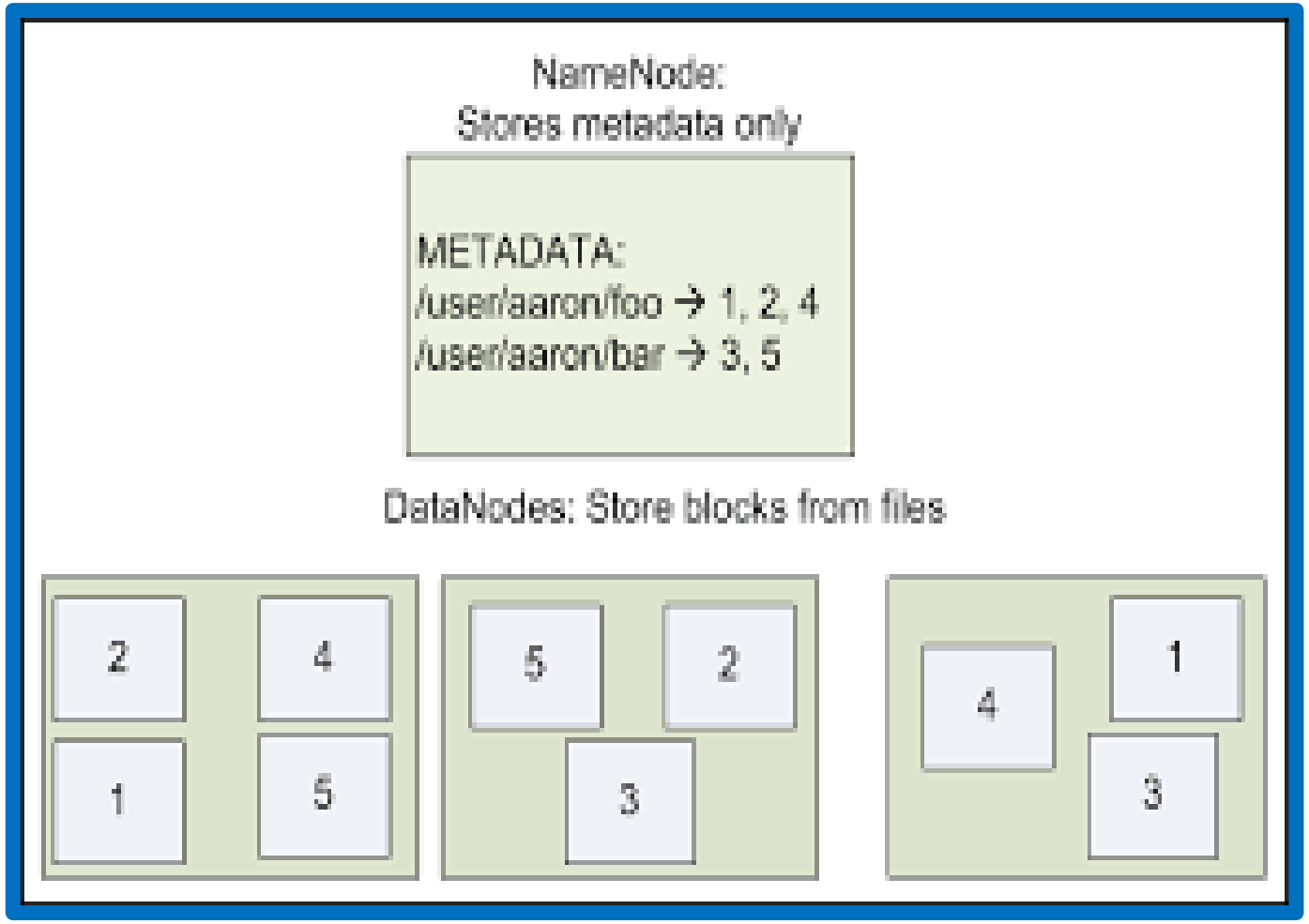
Responsible to store the data and process the computation tasks

Periodically sends a report to the name node.



BIG DATA

HDFS – Data Nodes



BIG DATA

HDFS – Hadoop distributed File system

Data Blocks

Each file split into datablocks – 128MB for HDFS v2, 64MB for HDFS v1

Each block is stored on one or more nodes

Each copy of the block is called replica

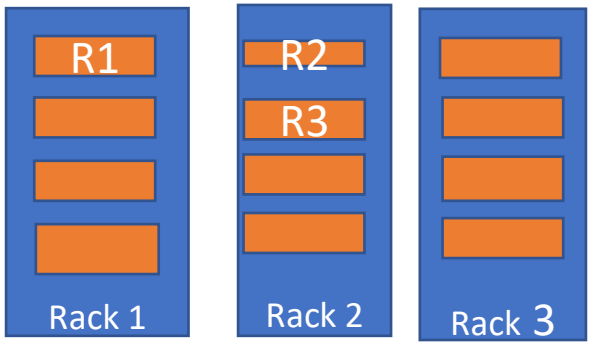
Block placement policy

First replica is placed on the local node

Second replica is placed in a different rack

Third replica is placed in the same rack as the second replica

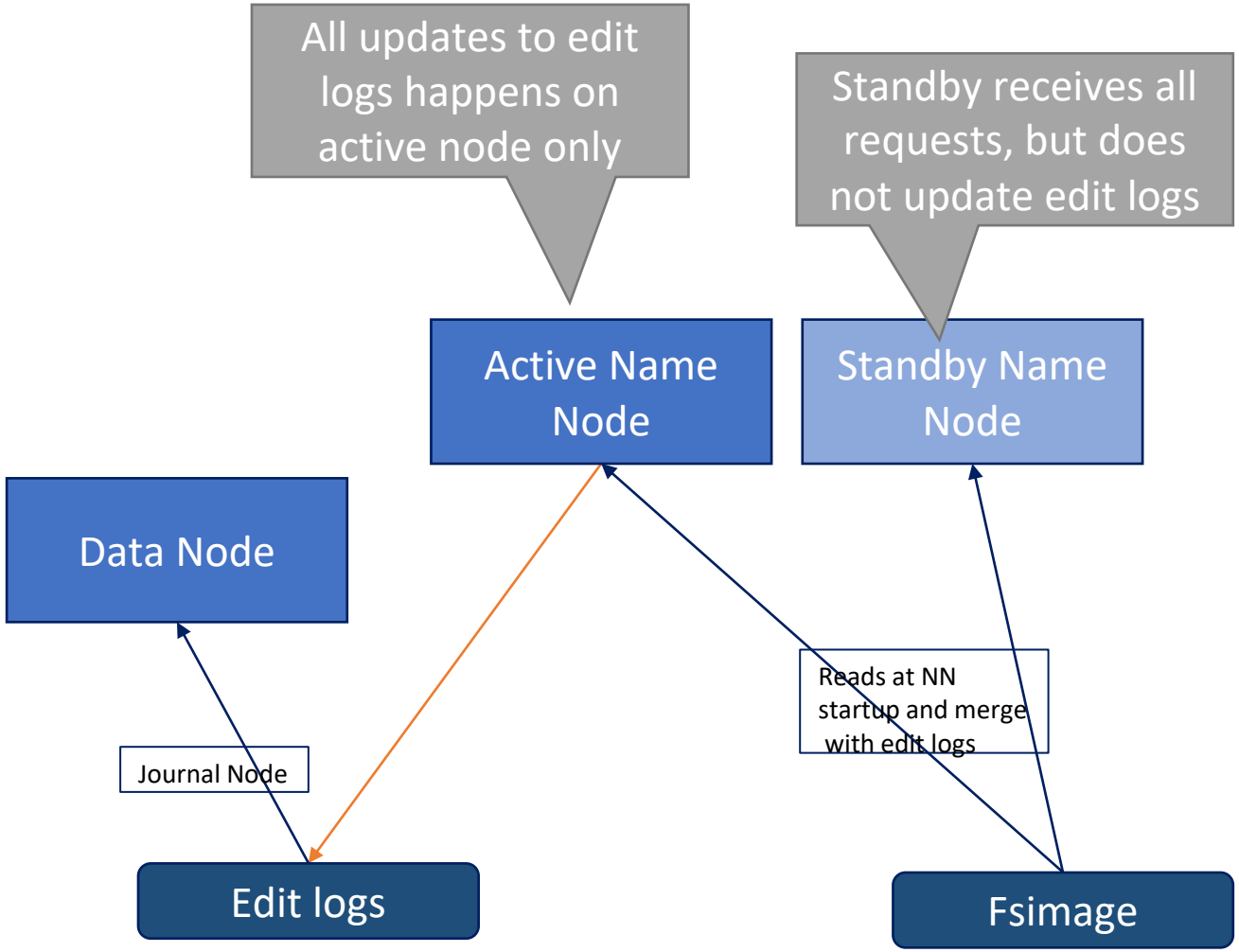
Block placement policy



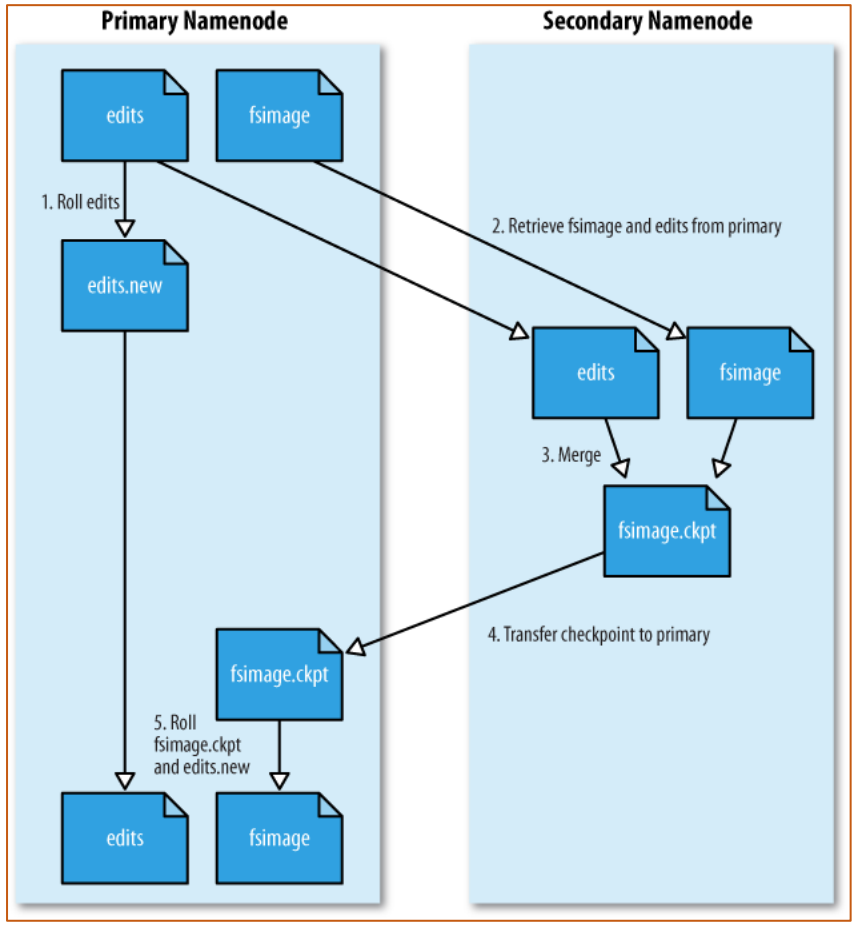
Big Data: HDFS Fault Tolerance and Replication

BIG DATA

Namenode fault tolerance



HDFS : Secondary Namenodes - Operation



Used to combine fsimage with edits

Offload the operation to secondary

Primary can focus on serving request

On completion of merging, new fsimage will be used.

BIG DATA

Secondary Namenodes

Secondary node	• keeps a copy of NameNode meta data.
Stored meta data	• merges the metadata files to obtain an updated Fsimage
Checkpointing	• Primary and secondary sync up the data
Why?	• To free namenode from task of merging

Big Data: HDFS Blocks

BIG DATA

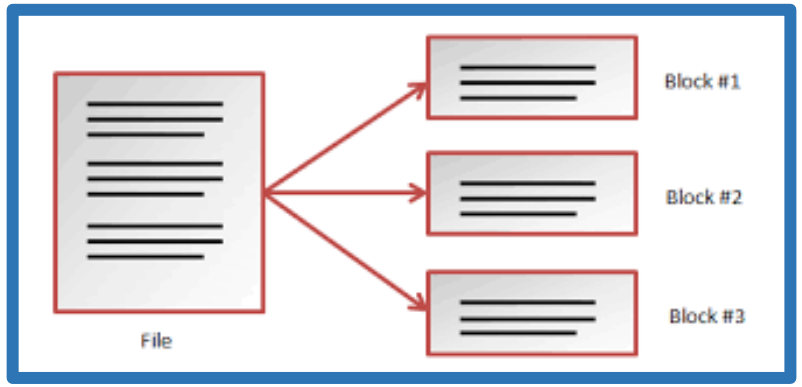
HDFS blocks - why?

Benefits of block abstraction.

- A file can be larger than any single disk in the network.
- Simplifies the storage subsystem
- Blocks fit well with replication for providing fault tolerance and availability.

% hadoop fsck -files –blocks

- will list the blocks that make up each file in the filesystem



HDFS block size – why 128MB?

- Time to read data from disk
  - = seek time + rotational latency + transfer time
  - Seek time – time to position head onto track (variable) - mechanical
  - Rotational latency – time to spin disk to get sector under head (fixed)
  - Transfer time – time to read data (fixed)
- Improving performance of read reduce seek times.
- Large block size transfer time >> seek time

Example

If the seek time is around 10 ms, and the transfer rate is 100 MB/s, then to make the seek time 1% of the transfer time, we need to make the block size around 100 MB.

Hadoop v1 default – 64MB
Hadoop v2 default – 128MB
Review: why has this increased?

THANK YOU

K V Subramaniam

Department of Computer Science and Engineering

subramaniamkv@pes.edu

+91 80 6666 3333 Extn 877

Content Creators

Prof Resma K S

Prof Prafullata K A

Prof Usha Devi

Prof Rachana

BIG DATA

Map Reduce Programming model and Architecture

K V Subramaniam
Computer Science and Engineering

What we have learnt so far..

- Large amounts of data to be processed.
- We have HDFS as a distributed store.
- We need to distribute the processing also.

What is Map Reduce ?

BIG DATA

Map Reduce Programming model and Architecture

Why Map-Reduce ?

- A new fundamental way to process extremely large data (?)
- We are going to
  - Study Map-Reduce paradigm
  - Study Hadoop architecture
    - Open Source implementation of Map-Reduce

What is MapReduce ?

- Origin from Google, [OSDI'04]
- A simple programming model
- Functional model
- For large-scale data processing
  - Exploits large set of commodity computers
  - Executes process in distributed manner
  - Offers high availability

Big Data: Map Reduce Motivating Example

Map Reduce - Motivation

- Lots of demands for very large scale data processing

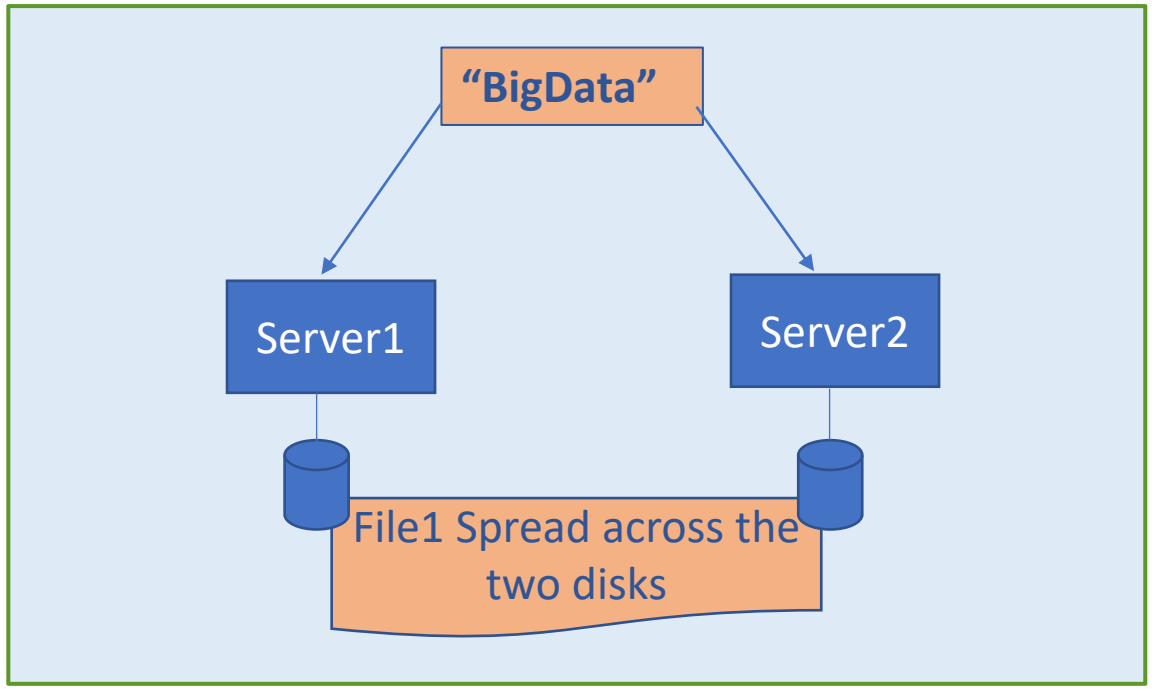
- Lots of machines needed (scaling)
- Two basic operations on the input
  - Map
  - Reduce

- To understand what Map/Reduce really are..

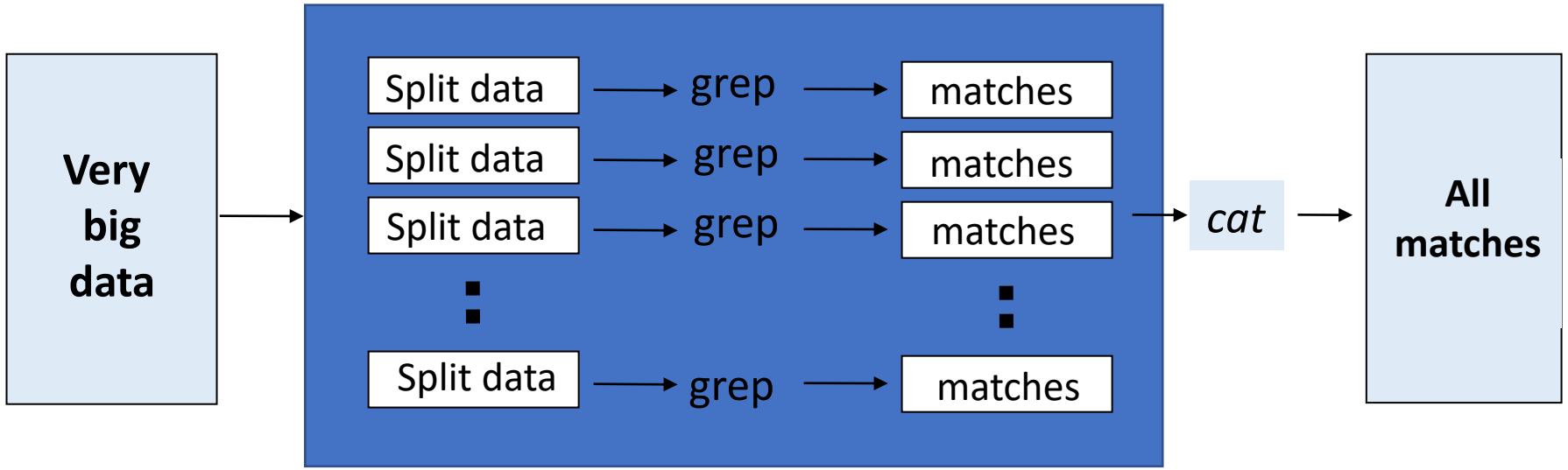
BIG DATA

A MapReduce Example

- Consider a very large text file and you want to determine if a word exists in this file?
  - The file is stored in HDFS across two machines.
  - How will you search to check if the word "BigData" is present in the file?



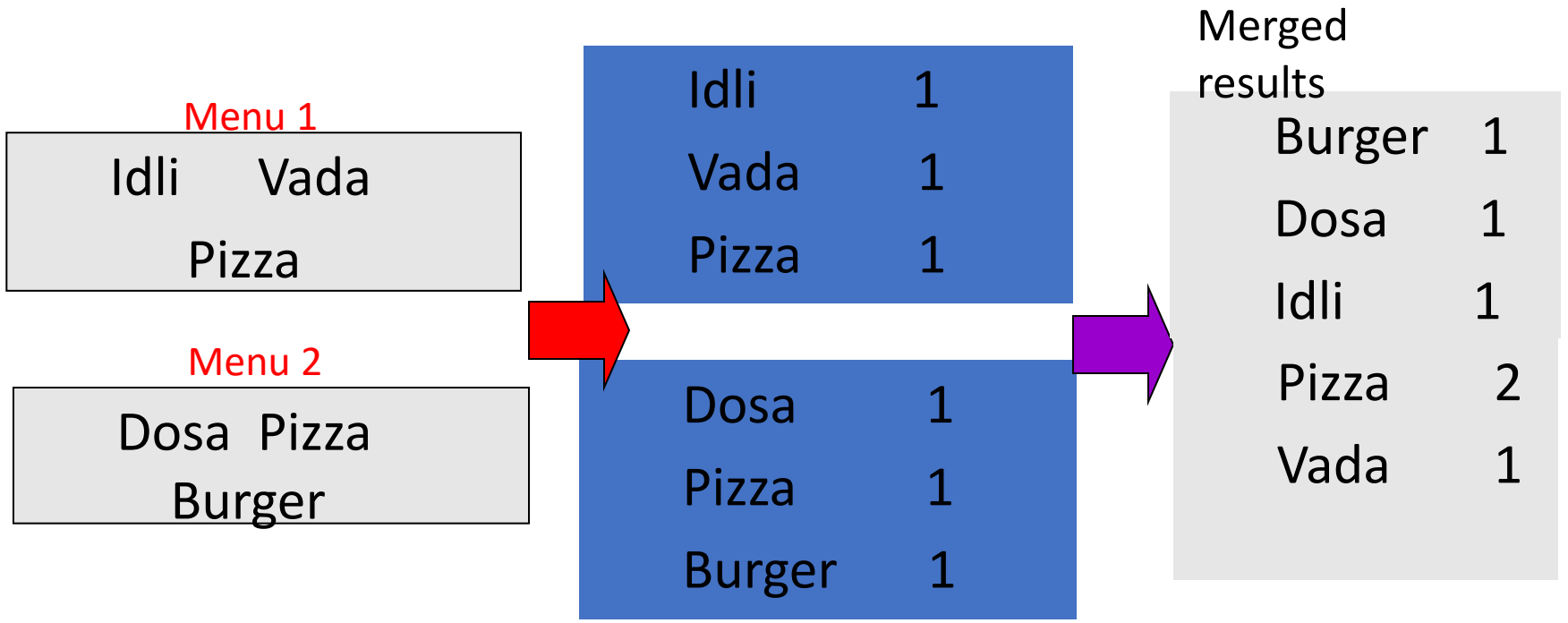
Map Reduce: Distributed Grep-Solution



Distributed Word Count

BIG DATA

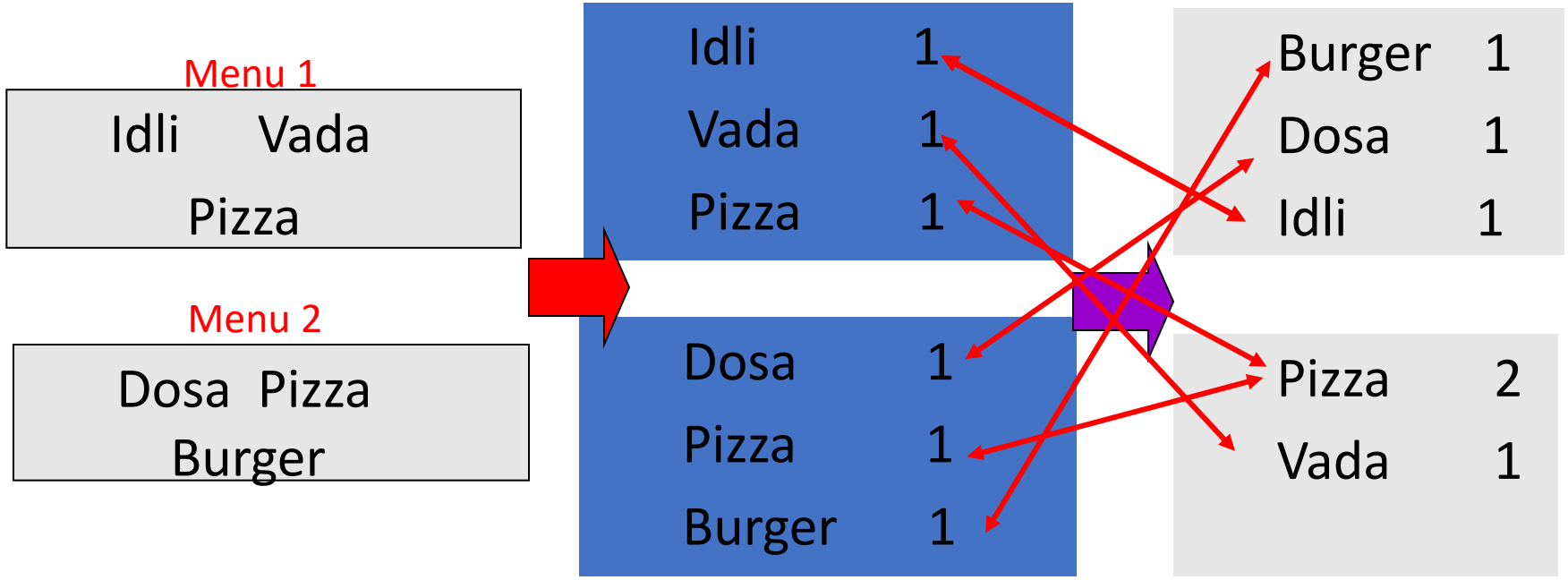
Example: find the number of restaurants offering each item?



- Suppose we need parallelism of the merge program.
- Would the earlier approach work?
- What do we need to do?

BIG DATA

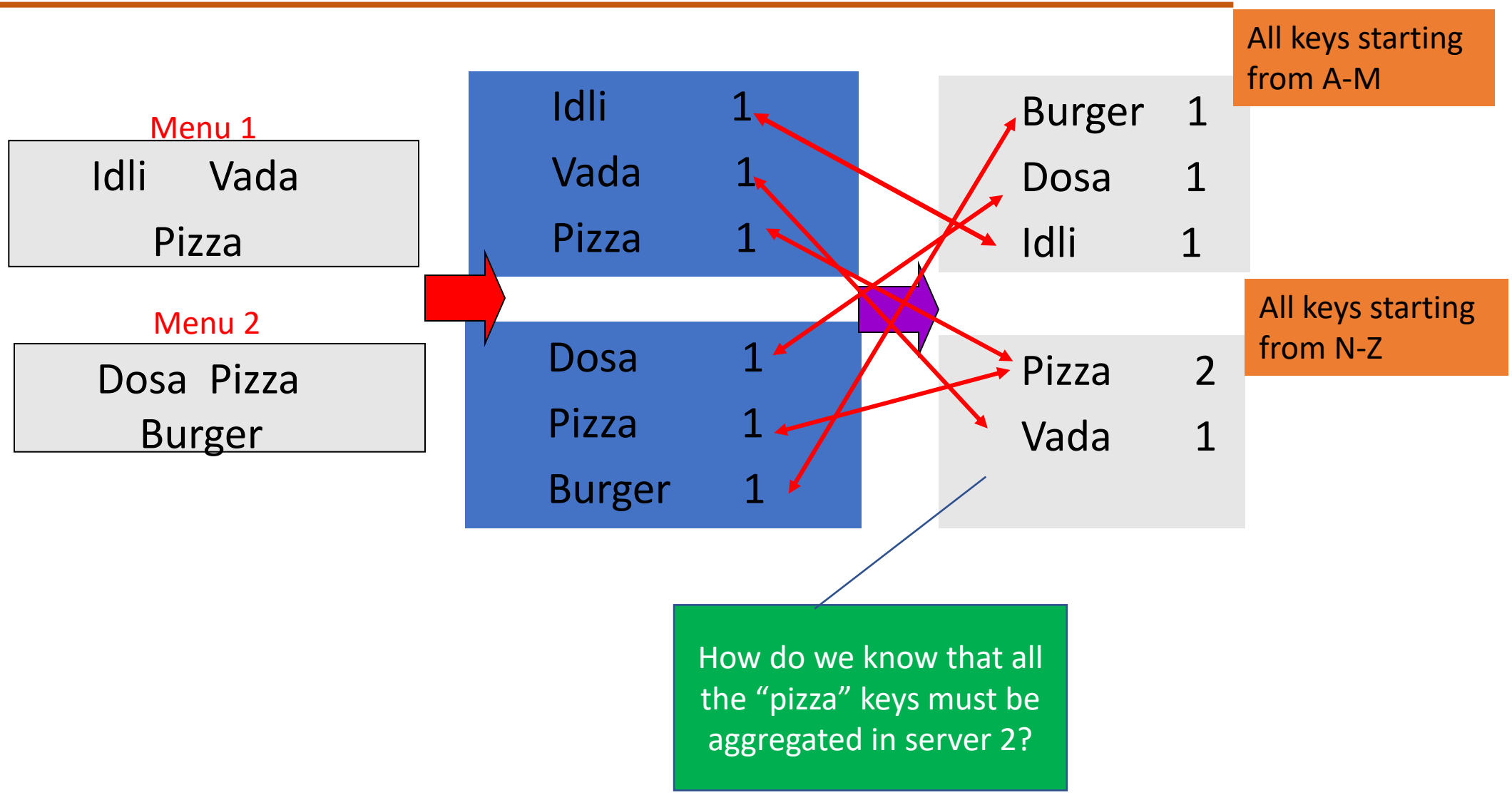
Example: find the number of restaurants offering each item?



- Treat output of first program as a key value pair
- Partition the keys between the second program

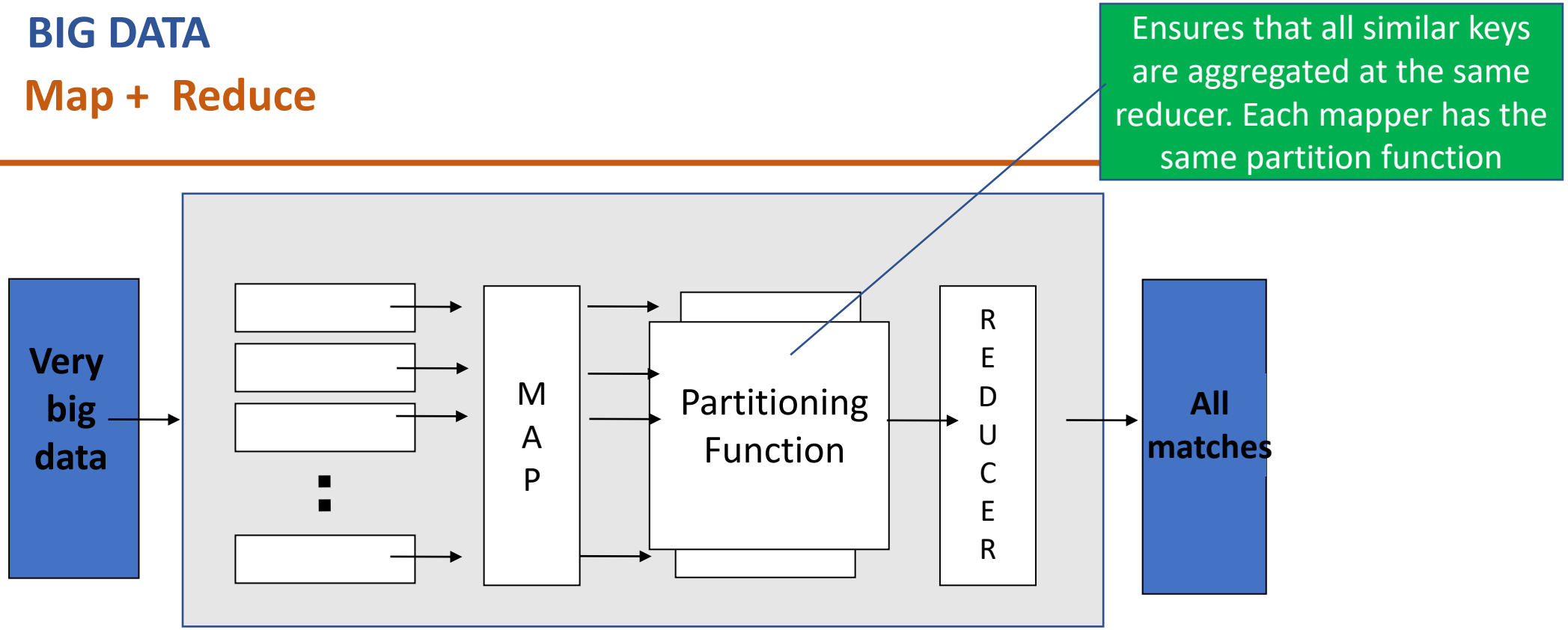
BIG DATA

Distributed Word Count



BIG DATA

Map + Reduce



- Map:
  - Accepts *input* key/value pair
  - Emits *intermediate* key/value pair

- Reduce :
  - Accepts *intermediate* key/value* pair
  - Emits *output* key/value pair

Map Reduce: A look at the code

Map Reduce Programming model

- **Data type:** key-value *records*
- **Map function:**

$$(K_{in}, V_{in}) \rightarrow list(K_{inter}, V_{inter})$$

- **Reduce function:**

$$(K_{inter}, list(V_{inter})) \rightarrow list(K_{out}, V_{out})$$

Map Reduce- Mapper for Word Count

```
public static class TokenizerMapper
      extends Mapper<Object, Text, Text, IntWritable>{

   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();

   public void map(Object key, Text value, Context context
                   ) throws IOException, InterruptedException {
     StringTokenizer itr = new StringTokenizer(value.toString());
     while (itr.hasMoreTokens()) {
       word.set(itr.nextToken());
       context.write(word, one);
     }
   }
 }
```

(Key , value)

List (Key ,value)

$(K_{in}, V_{in}) \rightarrow list(K_{inter}, V_{inter})$

BIG DATA

Map Reduce -Reducer for Word Count

```
public static class IntSumReducer
      extends Reducer<Text,IntWritable,Text,IntWritable> {
   private IntWritable result = new IntWritable();

   public void reduce(Text key, Iterable<IntWritable> values,
                      Context context
                      ) throws IOException, InterruptedException {
     int sum = 0;
     for (IntWritable val : values) {
       sum += val.get();
     }
     result.set(sum);
     context.write(key, result);
   }
 }
```

Key ,List (value)

List (Key ,value)

$(K_{inter}, list(V_{inter})) \rightarrow list(K_{out}, V_{out})$

Map Reduce - Main Program for Word Count

```
public static void main(String[] args) throws Exception {
 Configuration conf = new Configuration();
 String[] otherArgs = new GenericOptionsParser(conf, args).
                           getRemainingArgs();
 if (otherArgs.length < 2) {
   System.err.println("Usage: wordcount <in> [<in>...] <out>");
   System.exit(2);
 }
 Job job = new Job(conf, "word count");
 job.setJarByClass(WordCount.class);
 job.setMapperClass(TokenizerMapper.class);

 job.setReducerClass(IntSumReducer.class);
 job.setOutputKeyClass(Text.class);
 job.setOutputValueClass(IntWritable.class);
 for (int i = 0; i < otherArgs.length - 1; ++i) {
   FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
 }
 FileOutputFormat.setOutputPath(job,
   new Path(otherArgs[otherArgs.length - 1]));
 System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

Set Mapper and Reducer class

Map Reduce: Sample Exercise

Input: file(lineNumber, line) records and pattern

Output: lines matching a given pattern

What will be the mapper and reducer? What will be the keys?

Map:

Reduce:

Map Reduce , Search - Solution

Input: file (lineNumber, line) records and pattern

Output: lines matching a given pattern

Map:

```
if(line matches pattern):
        output(line)
```

Reduce: identity function

–Alternative: no reducer (map-only job)

Map Reduce , Functions in the Model

- **Map**
  - Process a key/value pair to generate intermediate key/value pairs
  - Sorts all key/value pairs before sending to reducer
- **Reduce**
  - Merge all intermediate values associated with the same key
  - Runs after all Map tasks are finished (why?)
- **Partition**
  - By default : **`hash(key) mod R`** （Well balanced)
  - There are cases where this can be more complex

Map Reduce: Revision exercise

Map Reduce, Sort

Input: (key, value) records

Output: same records, sorted by key

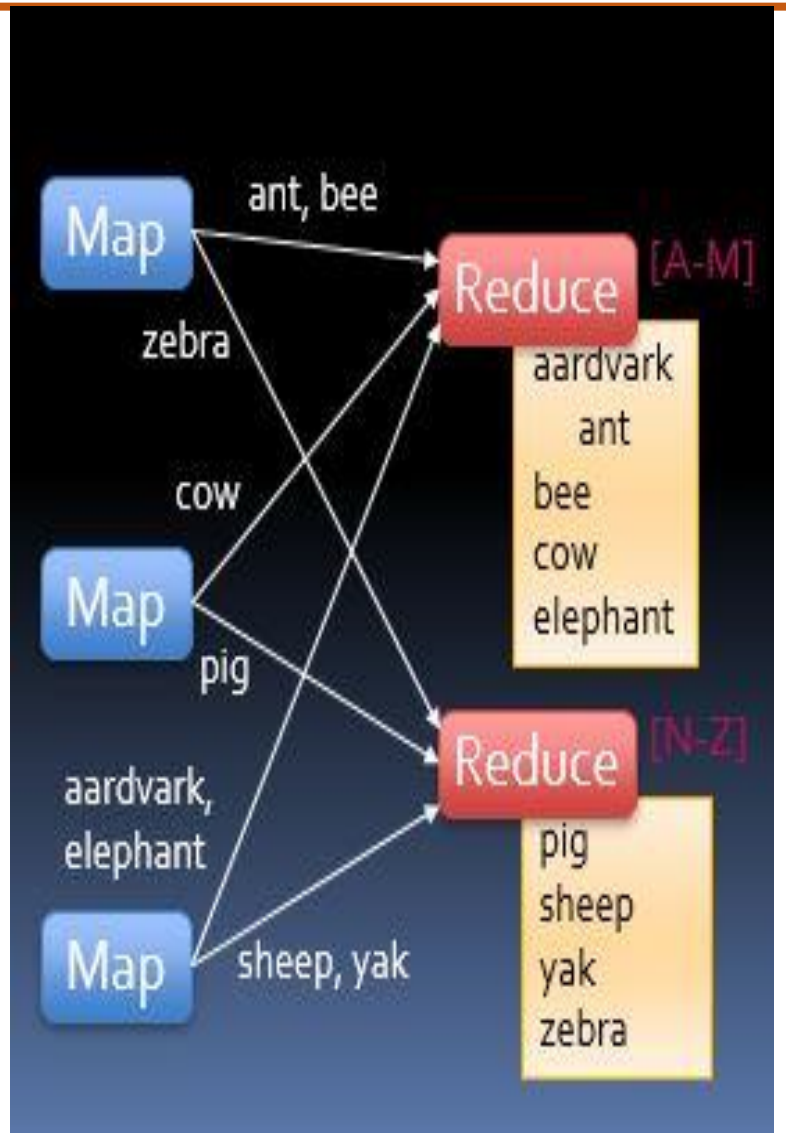
What will be the mapper and reducer?
What will be the partition function?

Map:

Reduce:

Map Reduce, Sort - Solution

- **Input**: (key, value) records
- **Output**: same records, sorted by key
- **Map**: identity function
- **Reduce**: identity function
- **Trick**: Pick partitioning function *p* so that $k_1 < k_2 \Rightarrow p(k_1) < p(k_2)$
- Works because map sorts output keys.

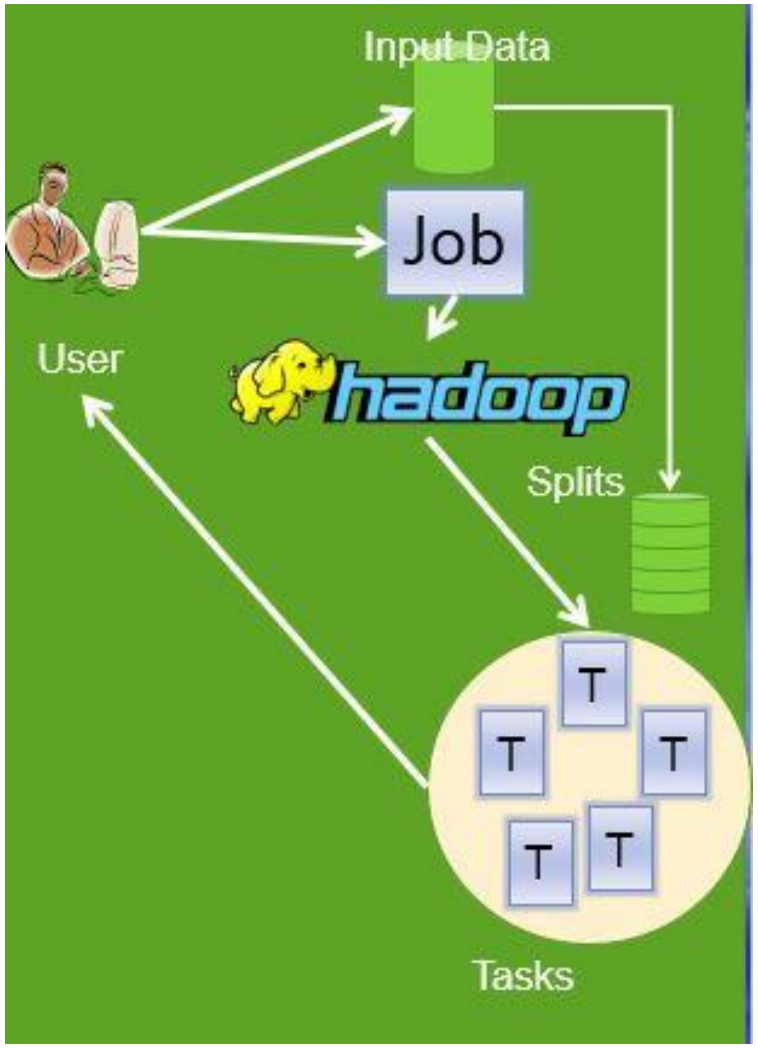


Map Reduce: Job Submission Flow

BIG DATA

Map Reduce , Hadoop Flow.

- User submits job
  - input data, MapReduce program, and configuration information
- How is parallelization achieved?
  - Divide input into smaller chunks – called **input splits**
- Hadoop divides jobs into tasks
  - Map tasks, reduce tasks
  - One map task per split
  - Tasks run in parallel



BIG DATA

MapReduce Flow for A SINGLE REDUCE Task

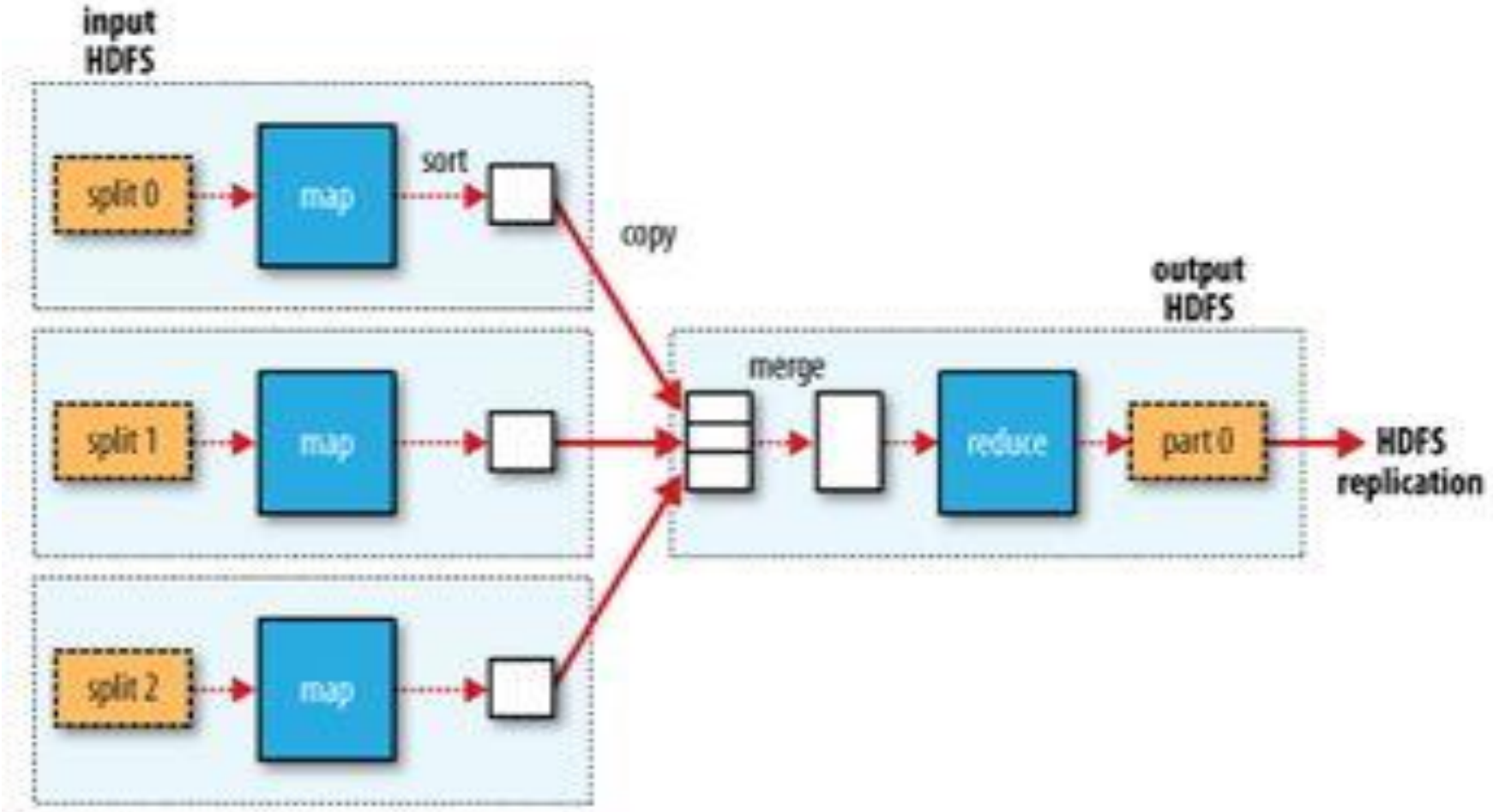


Figure: Map reduce data flow for single reducer task.

Map Reduce:
Exercise : Job Submission Flow with two Reducers

MapReduce Flow

- How will it work when there are two reducers?
- Where will the outputs be?

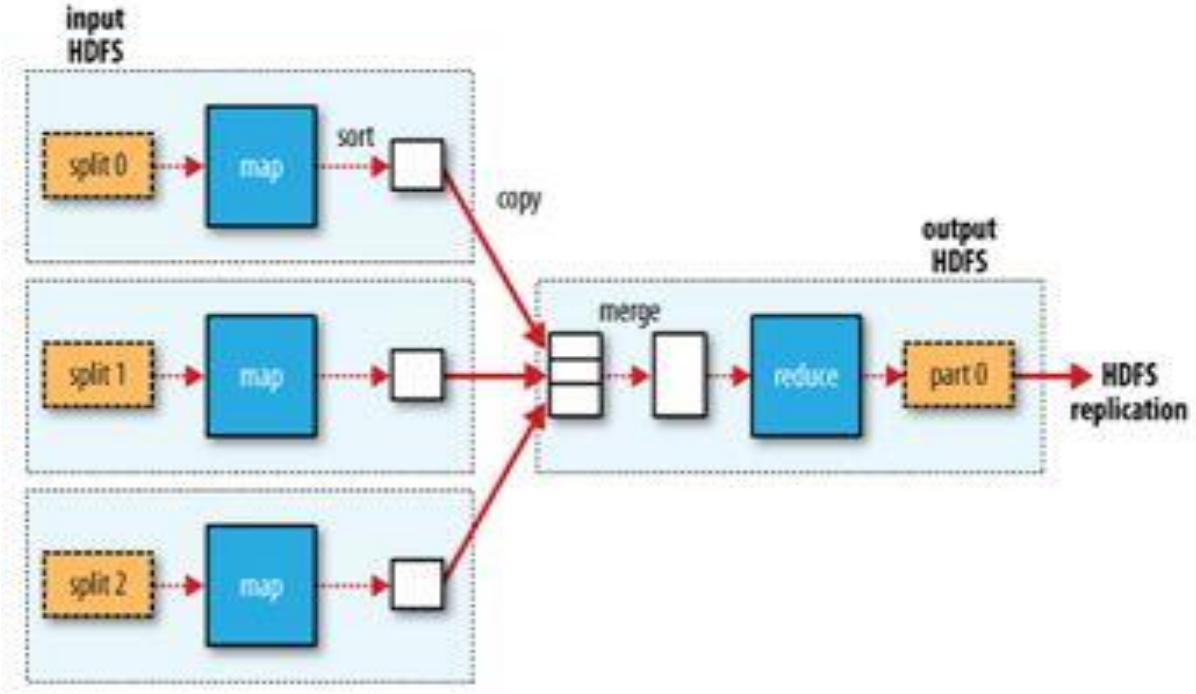
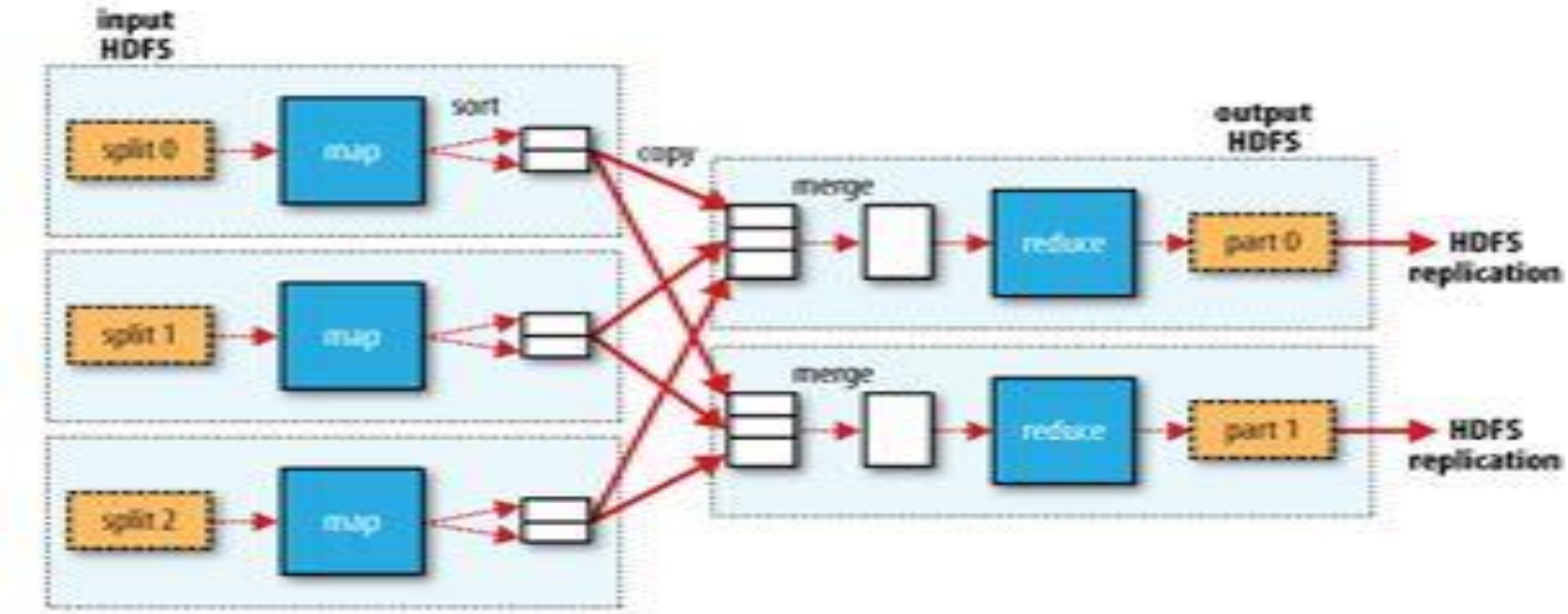


Figure: Map reduce data flow for single reducer task.

BIG DATA

MapReduce Flow for MULTIPLE REDUCE Tasks



- Outputs are created on two different nodes and have to be merged.
- But available through HDFS on any node

Map Reduce: Splits

Map Reduce Split size considerations

- Split size proportional to parallelism
- Small split size
  - Advantages
    - Large #splits
    - Increased parallelism
    - Increased load balancing
  - Disadvantages
    - the overhead of managing the splits and of map task creation
    - begins to dominate the total job execution time.

Optimal split size == size of HDFS block (128MB) (default) on Hadoop v2

Map Output

Split == Block size

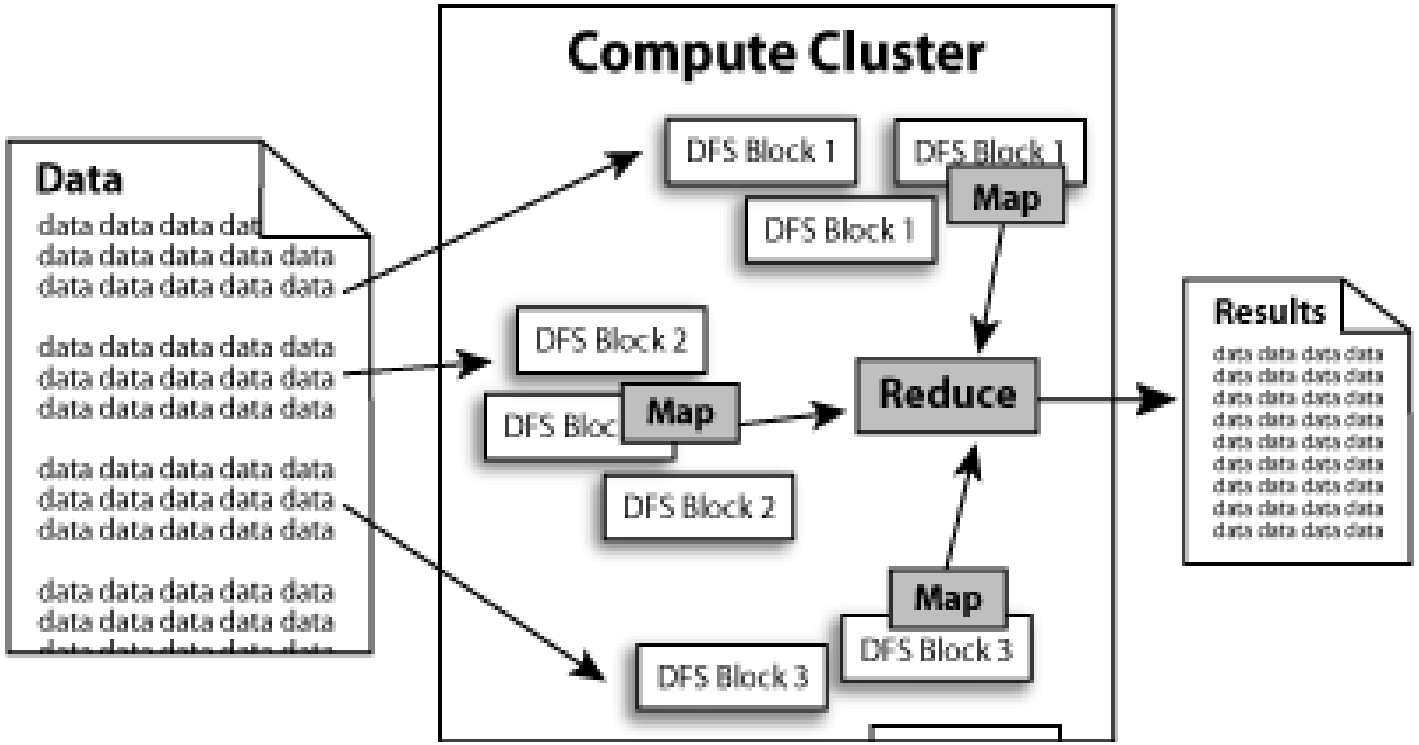
- All data required for Map
  - In the same node
  - No inter-node data transfer is required

Split != block size

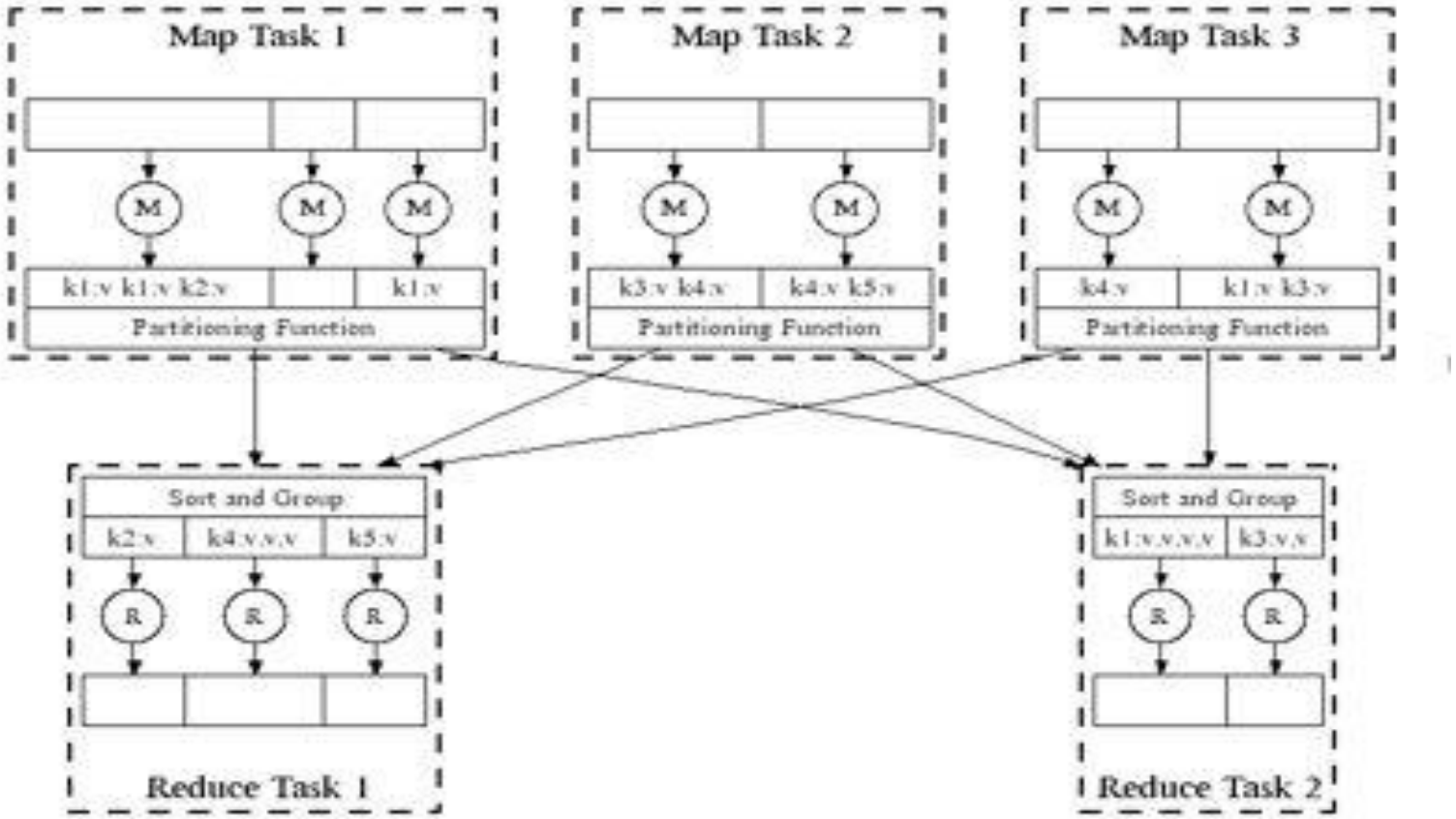
- Data transfer across multiple nodes
- Impacts performance

BIG DATA

Traditional: Move Data to Compute



Big Data: Move Compute to Data



Parallel Execution

Map Output

- **Where is Map output written to?**
  - Local disk and not HDFS
  - Why? Temporary output to be discarded after reduce.

- **Failure**
  - If the node running the map task fails
    - before the output has been consumed by reducer
  - Automatically rerun map task on another node

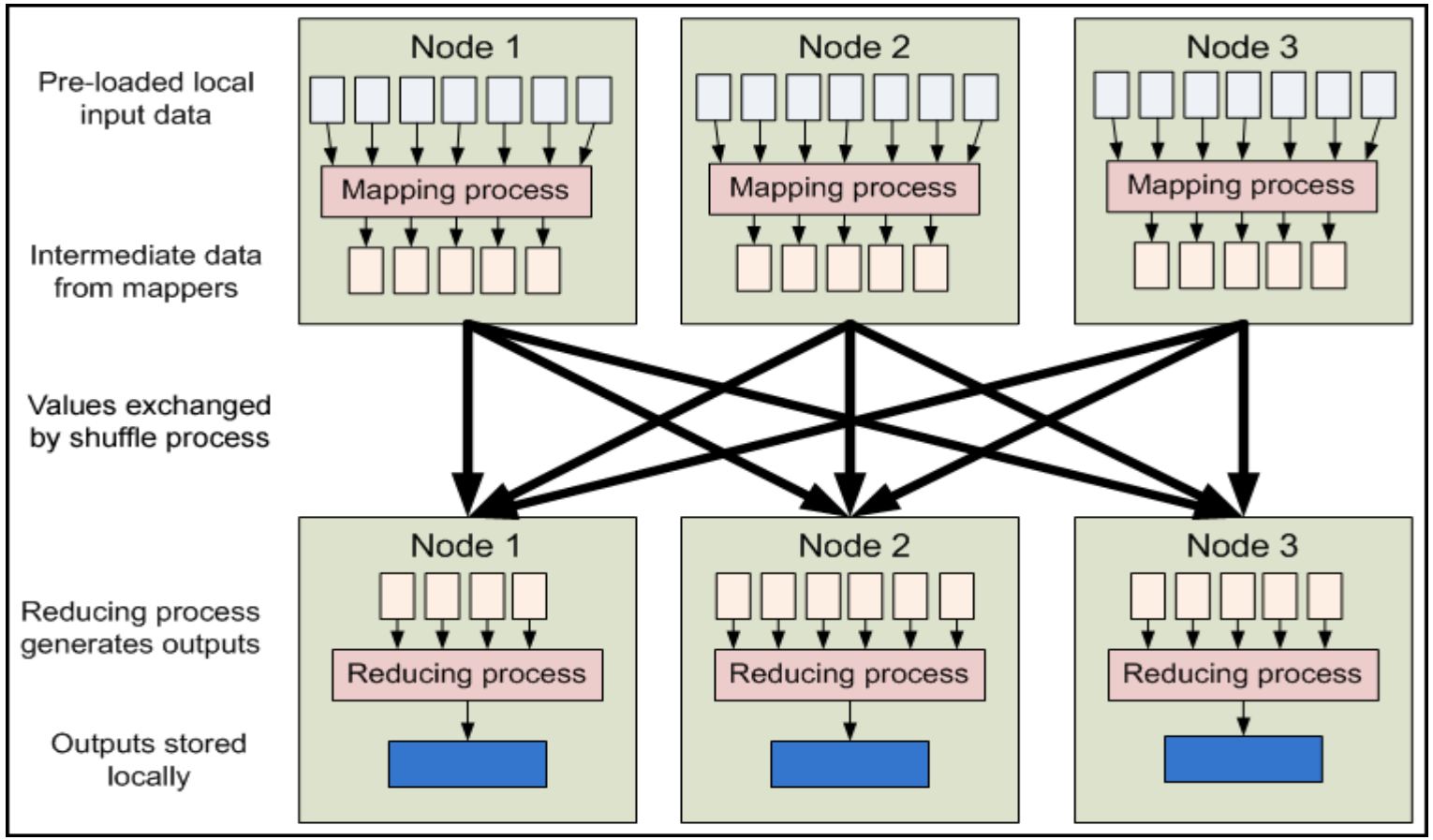
Reduce Tasks

- Reduce tasks don't have the advantage of data locality
  - the input to a single reduce task is normally the output from all mappers.
- Sorted map outputs
  - have to be transferred across the network
  - Where to?
    - To the node where the reduce task is running
    - Merge data from different mappers
    - Then passed to the user-defined reduce function.
- The output of the reduce is normally stored in HDFS for reliability.
- Where is the reduce output stored
  - 1 on the local node where the reduce happens
  - Other replicas on off-rack nodes.
  - Consumes network bandwidth

Map Reduce: Working

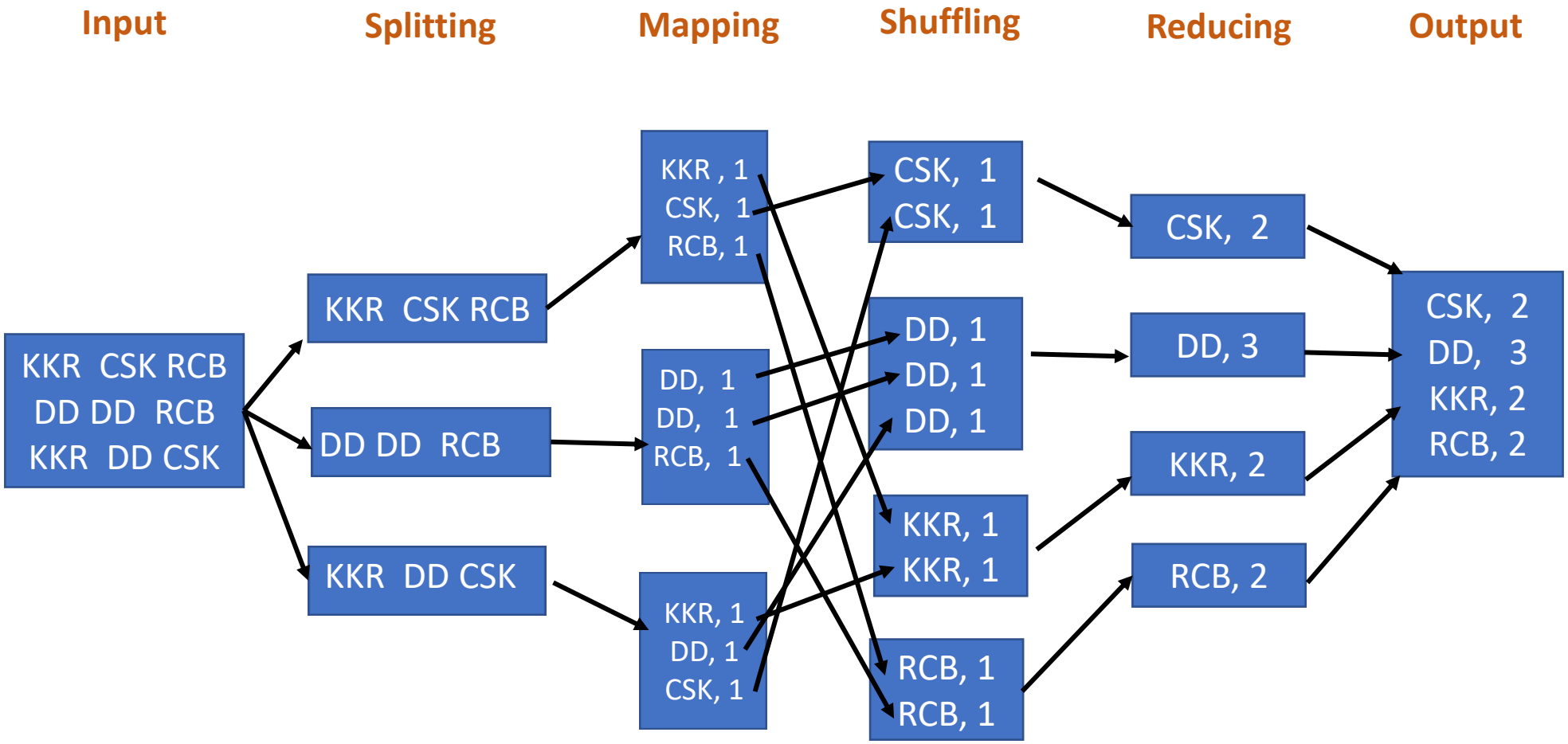
BIG DATA

High level view



BIG DATA

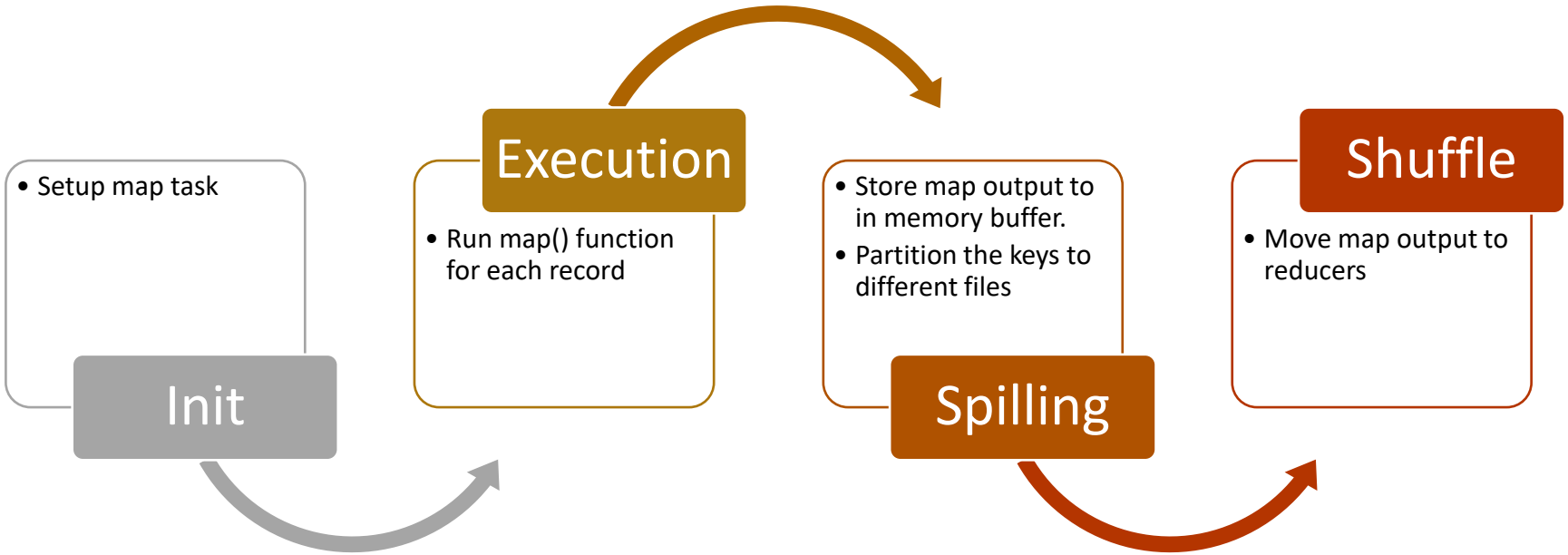
Shuffling - example

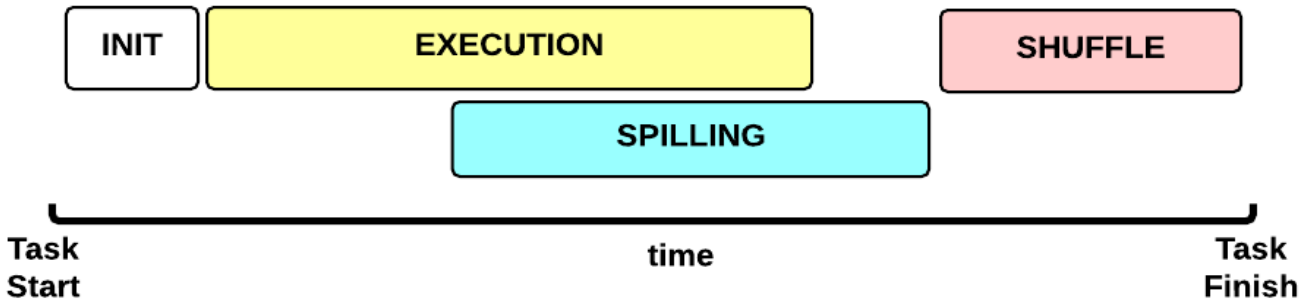


The overall Map-Reduce word count Process

BIG DATA

Closer look - Map

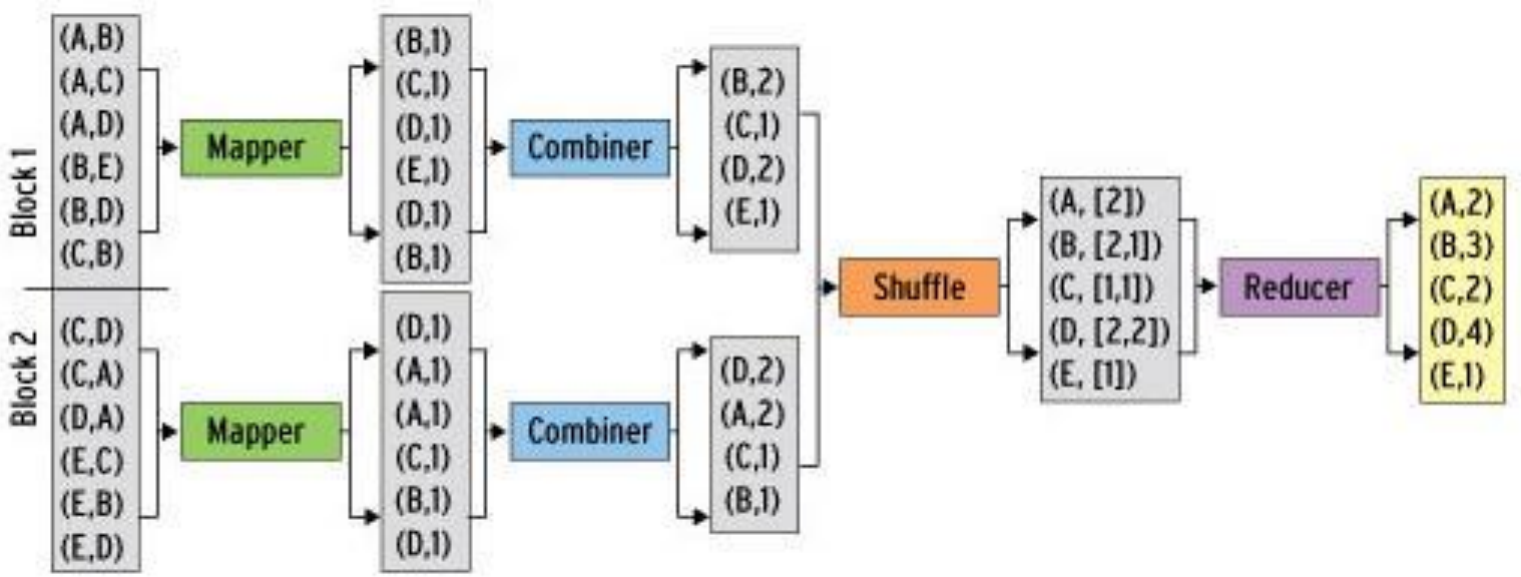




Map Reduce: Combiners

BIG DATA

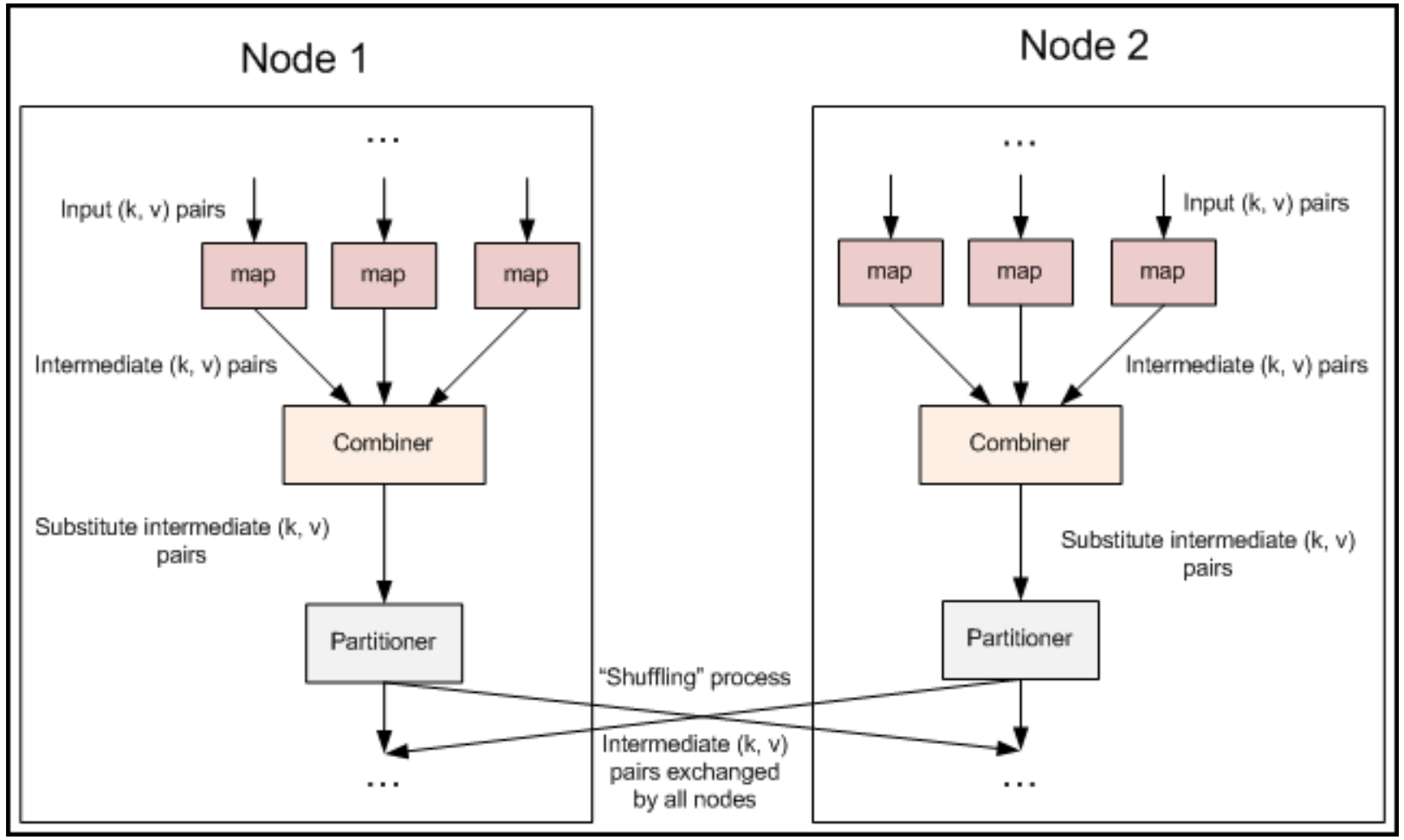
MapReduce – Combiners - motivation



- Combine multiple map outputs before doing a reduce
- Can write a combiner function in program
  - Combiner will be run before reduce
- Mini-reducer

BIG DATA

Combiner – when does it run?



BIG DATA

Map Reduce - Main Program for Word Count

```
public static void main(String[] args) throws Exception {
 Configuration conf = new Configuration();
 String[] otherArgs = new GenericOptionsParser(conf, args).
                           getRemainingArgs();
 if (otherArgs.length < 2) {
   System.err.println("Usage: wordcount <in> [<in>...] <out>");
   System.exit(2);
 }
 Job job = new Job(conf, "word count");
 job.setJarByClass(WordCount.class);
 job.setMapperClass(TokenizerMapper.class);
 job.setCombinerClass(IntSumReducer.class);
 job.setReducerClass(IntSumReducer.class);
 job.setOutputKeyClass(Text.class);
 job.setOutputValueClass(IntWritable.class);
 for (int i = 0; i < otherArgs.length - 1; ++i) {
   FileInputFormat.addInputPath(job, new Path(otherArgs[i]));
 }
 FileOutputFormat.setOutputPath(job,
   new Path(otherArgs[otherArgs.length - 1]));
 System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

Combiner is set here

Review Questions

Review Questions

- Questions from T1, LOR 2.4

Sample questions

- How many mappers and reducers will get started when trying to process a 230 MB file with Hadoop v2?
  - Ans: Block size = 128MB, so there will be two blocks. Assuming one block per split there will be 2 mappers
    - #reducers is configurable.
- Where is a combiner executed?
  - On the mapper.
- Write mappers and reducer pseudo code showing keys for counting #unique words in a file?
  - Similar to word count. Just that reducer does not have to write the count.

Additional Notes, References and Videos

Notes

- Chapter 2.4 from T1 – Rajkamal
- Tom whites book is an excellent reference for the programming component.

THANK YOU

K V Subramaniam
Dept. of Computer Science and Engineering

subramaniamkv@pes.edu

BIG DATA

Hands On Session - 1

MapReduce

K V Subramaniam
Computer Science and Engineering

Overview

BIG DATA

MapReduce Handson Setup

- In order to execute the wordcount application, we need
- Ubuntu – virtualbox (example Ubuntu 20.04)
- Java version 8
- Hadoop v3.2.1
- Wordcount application

BIG DATA

Overview

- MapReduce is a programming model.
- Implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.
- Mapper method performs filtering and sorting, and a Reduce method, performs an aggregate operation.
- We aim to solve a real world problem using MapReduce.

Problem Statement

- Find the number of cars in every city which use gas as a mode of fuel using MapReduce.

BIG DATA

MapReduce Hands-on Exercise

SPECIFICATIONS

1. Ubuntu 16.04+
2. Hadoop: 3.2
3. Python: 2.x/3.x
4. Java: 1.8
5. Dataset: You will be using the modified “Craigslist Used Cars Dataset” for this session. Please download the dataset from the below Google Drive link: https://drive.google.com/open?id=1GxEaY_aAlkMHfJN2Z1Cvt1O1yNtCp1gN

- **Columns of the Dataset :** The columns are indexed from [0-25] (Ex. Transmission is the 11th index)

- **Sample output :**

City	Number of Cars that use Gas
Bangalore	10
Chennai	12

- Actual output to be in a text file with each line of the answer having the pair <cityname> <number> .

Pointers & Clues

- **Python Coders:**
  - start the code with the python shebang:
    #!/usr/bin/python
  - use sys module to read from stdin

- **Java Coders -** A sample word count program can be found here:
  - **https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html**

BIG DATA

Job Management and YARN

K V Subramaniam
Computer Science and Engineering

What we have learnt so far..

- Data processing distributed over a cluster – Map Reduce
- Job Submission Flow
- How does job management actually happen?
- How is failure management addressed?

… handled by **YARN**

Map Reduce Programming model and Architecture

Lecture Overview

- Need for YARN - history
- YARN Architecture
- Job submission lifecycle – YARN
- Scheduling
- Failure Handling
- Benefits of YARN

Big Data: The need for YARN

Motivation

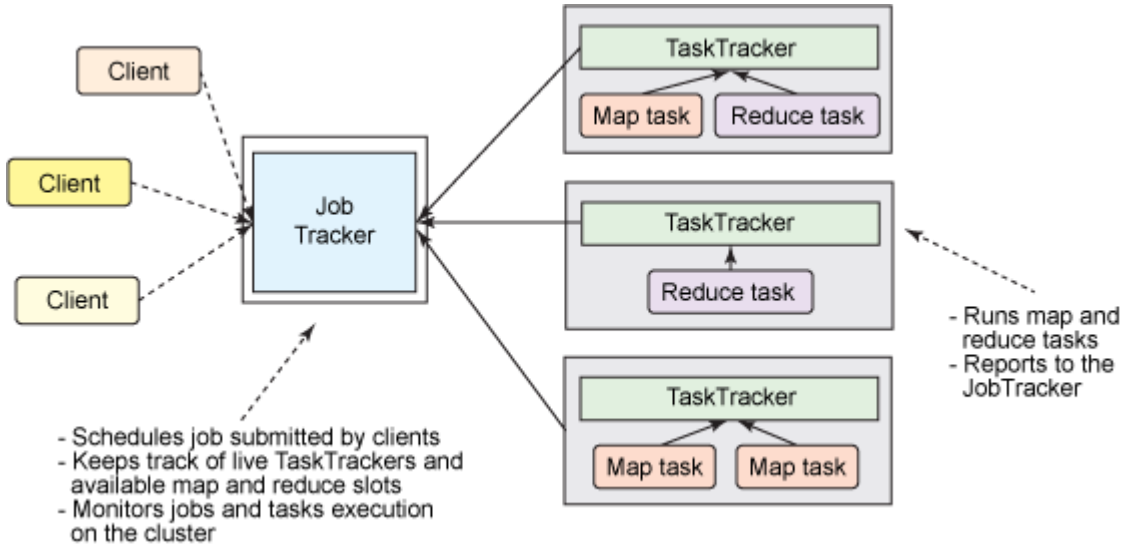
- Recall
  - Job – the entire map reduce application
  - Task – Individual mappers/reducers

- How do we
  - Allocate resources – determine which nodes will run the jobs
  - Monitor the tasks – start new tasks or restart failed/slow tasks
  - Monitor the overall state of the job?

BIG DATA

Hadoop 1.0 Job Management

- Job Tracker
  - Manage Cluster resources
  - Job scheduling
- Task Tracker
  - One per task
  - Manage the task
- Fault Tolerance, Cluster resource management and scheduling handled by JobTracker



Hadoop 1.0 Issues

Limits scalability

- Job tracker runs on a single machine and is responsible for cluster management, scheduling and monitoring

Availability

- JobTracker is the single point of availability/failure

Resource utilization problems

- Predefined #map/reduce slots. Utilization issues because map slots may be full but reduce slots are free.

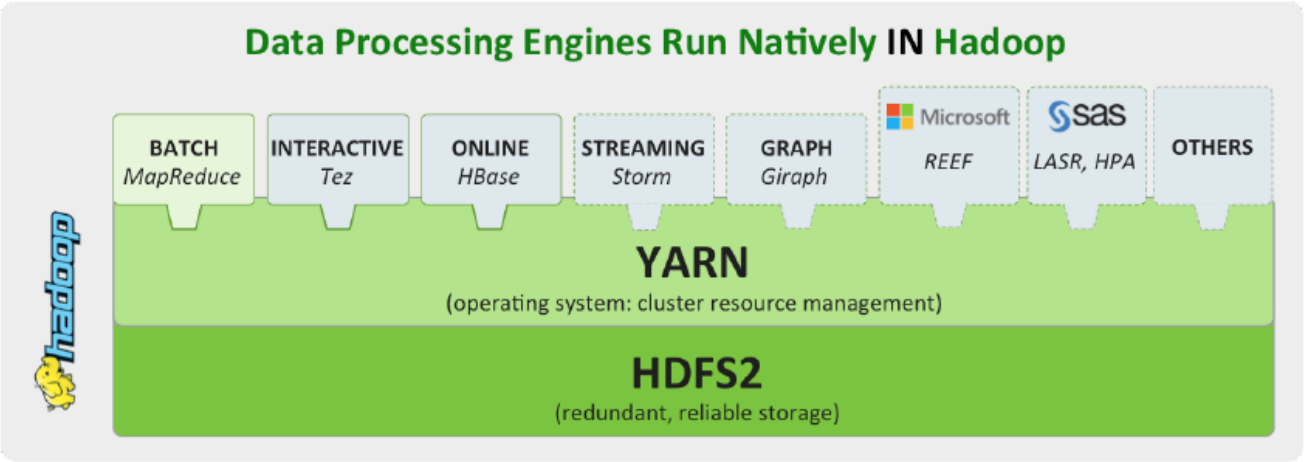
Limitation in running MR applications

- Tightly integrated with Hadoop. Only MR apps can run. Can’t coexist with other applications.

Big Data: YARN Architecture

BIG DATA

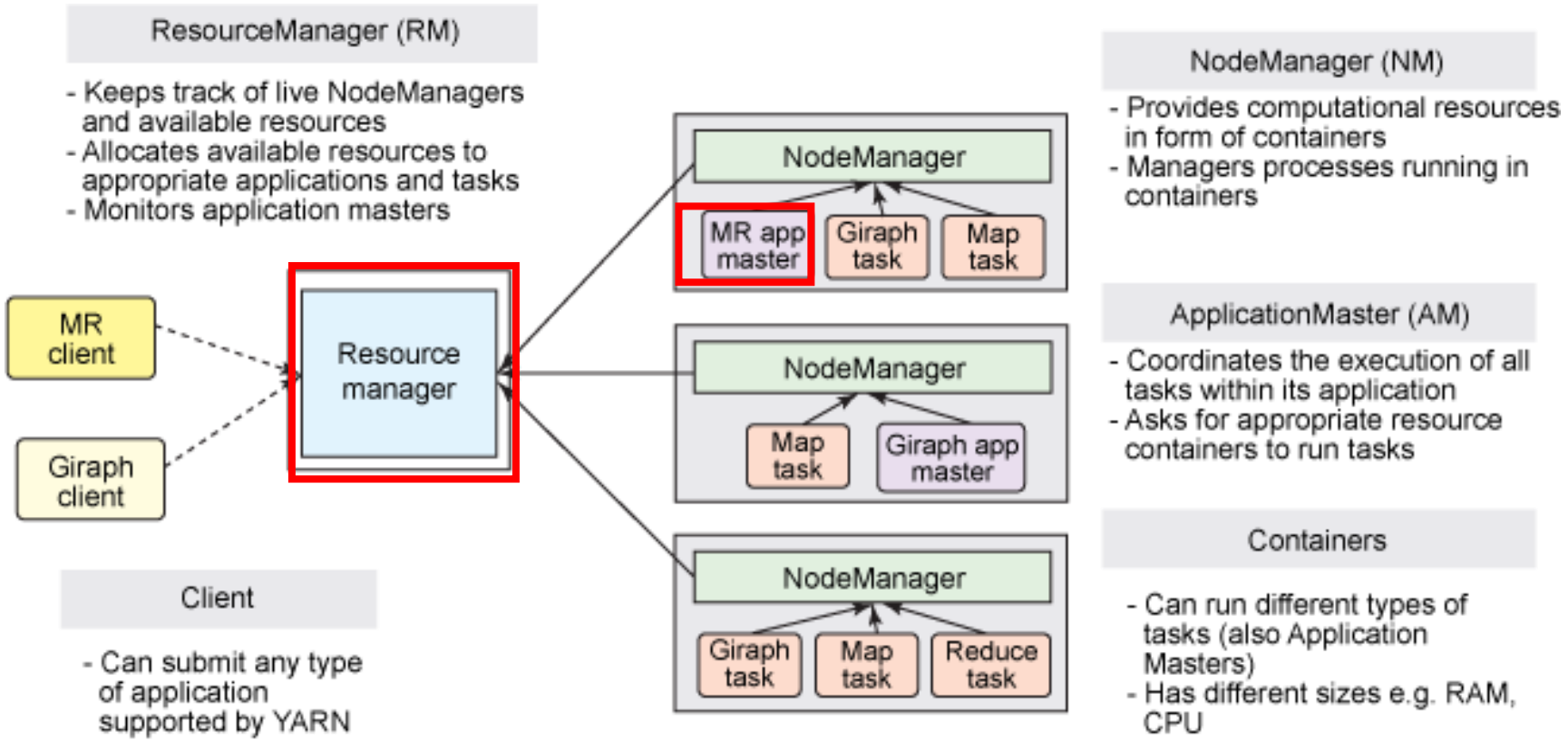
Map Reduce - Motivation



- Issues in managing clusters > 4000 nodes
- 2010 – MapReduce v2 with YARN
  - Yet Another Resource Negotiator
  - YARN Application Resource Negotiator!!

BIG DATA

YARN Architecture



- Split responsibility of Job Tracker
- Resource Manager – manage cluster wide resources
- Application Master – manage lifecycle of application

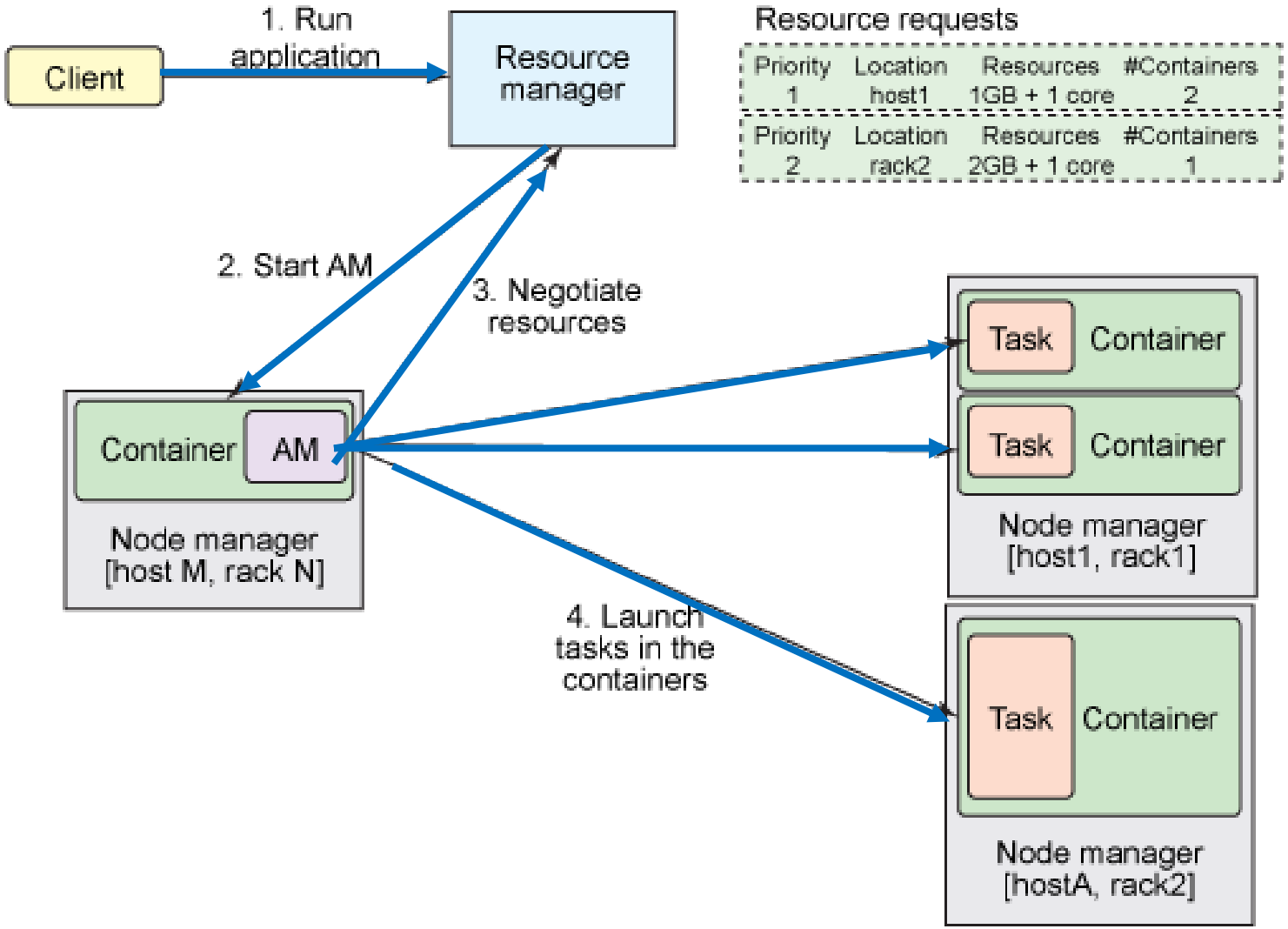
YARN Components

Resource Manager	• Arbitrates resources amongst all applications of the system
Node Manager	• Per machine slave • Responsible for launching application containers • Monitors resource usage
Application Master	• Negotiate appropriate resource containers from the scheduler • Track and monitor the progress of the containers
Container	• Unit of allocation incorporating resources such as memory, CPU, disk

Big Data: Job Submission - YARN

BIG DATA

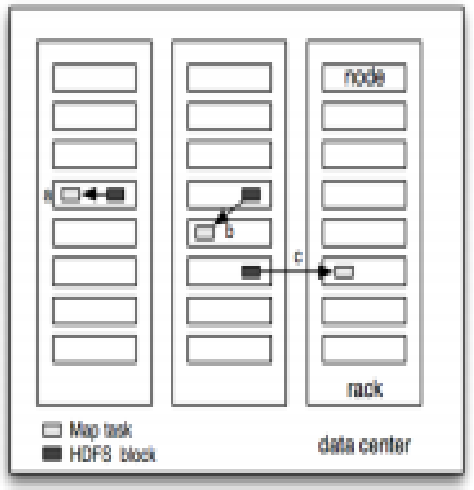
YARN Working



Data Locality in Map Reduce

- Attempts to run the map task on a node where the input data resides in HDFS.
  - *data locality optimization* - it doesn't use valuable cluster bandwidth.
- What happens when all nodes hosting the block replicas are busy?
  - look for a free map slot on a node in the same rack as one of the blocks.
- Very occasionally even this is not possible, so an off-rack node is used, which results in an inter-rack network transfer.

.



Scheduling in YARN

Schedulers in Hadoop

- Early Hadoop versions → simplistic FIFO scheduler
  - In order of submission
  - each job would use the whole cluster
  - so jobs had to wait their turn.
- How to share resources fairly?
- Balance between
  - Production jobs
  - Ad-hoc jobs

Fair Scheduler

- Aims to give every user a fair share of the cluster capacity over time.
- Jobs are placed in pools,
  - Default → each user gets their own pool.
- If a single job is running, it gets all of the cluster.
- As more jobs are submitted,
  - free task slots are given to the jobs in such a way as to give each user a fair share of the cluster.
- Short job – completes in reasonable time
- Long job – can continue making progress.

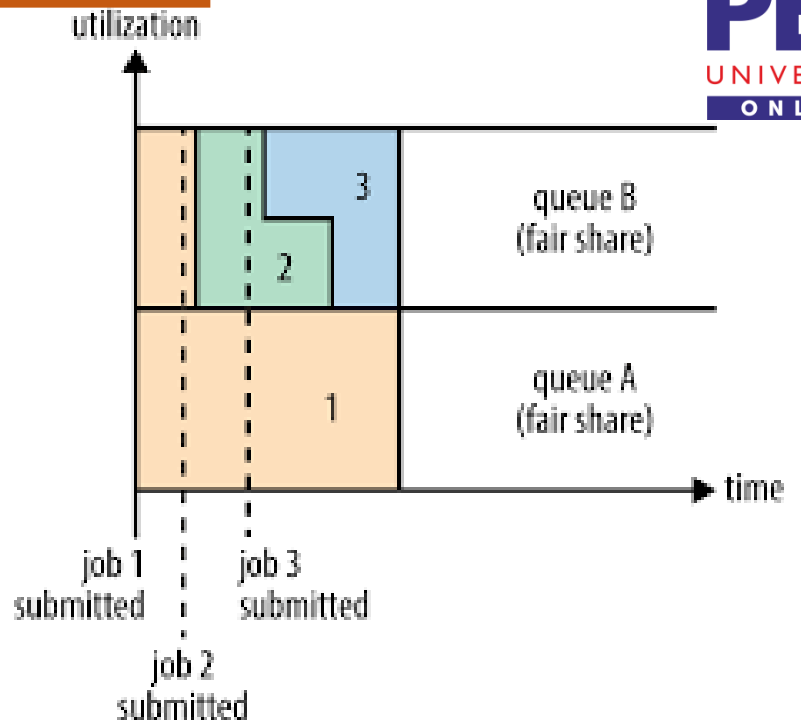


Image courtesy: Tom White, "Hadoop the definitive guide"

Fair Scheduler

- Consider a user who submits more jobs
  - Scheduler ensures → that user does not hog the cluster
- Custom pools
  - Guaranteed minimum capacities with map/reduce slots
  - It is also possible to define custom pools with guaranteed minimum capacities defined in terms of the number of map
- The Fair Scheduler supports preemption
  - If pool not received its fair share over certain time
  - scheduler will kill tasks in pools running over capacity

BIG DATA

Capacity Scheduler

- Different approach
- number of queues (like the Fair Scheduler's pools),
  - Has an allocated capacity
  - Can be hierarchical
  - Within each queue → scheduled using FIFO (with priorities)
- Cannot use free spare capacity even if it exists
- Like breaking up cluster into smaller clusters

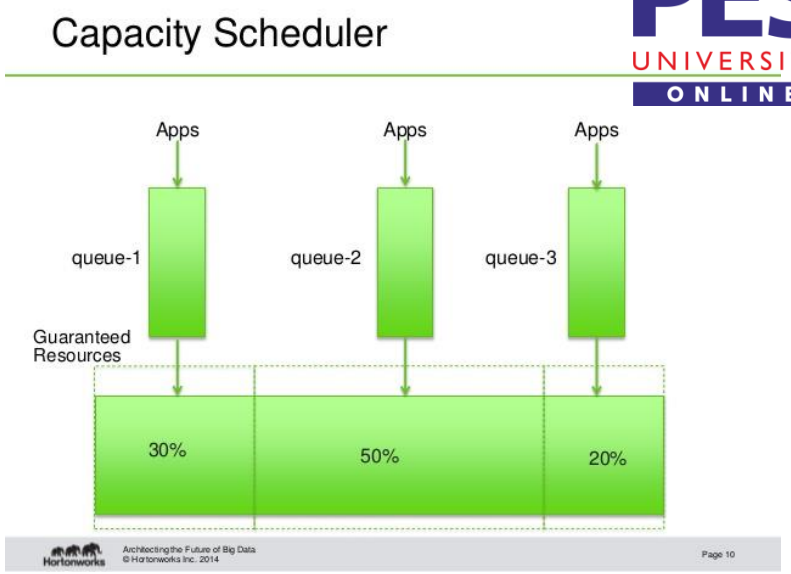


Image: https://www.slideshare.net/Hadoop_Summit/w-525hall1shenv2

Handling Failures

What can fail?

- Task
- Application Manager
- Resource Manager
- Node Manager

Task Failure

Due to runtime exceptions	• JVM reports error back to parent application master
Hanging tasks	• Progress updates not happening for 10 mins • Timeout value can be set.
Killed tasks	• Speculative duplicates can be killed
Recovery	• AM tries restarting task on a different node

Application Master Failure

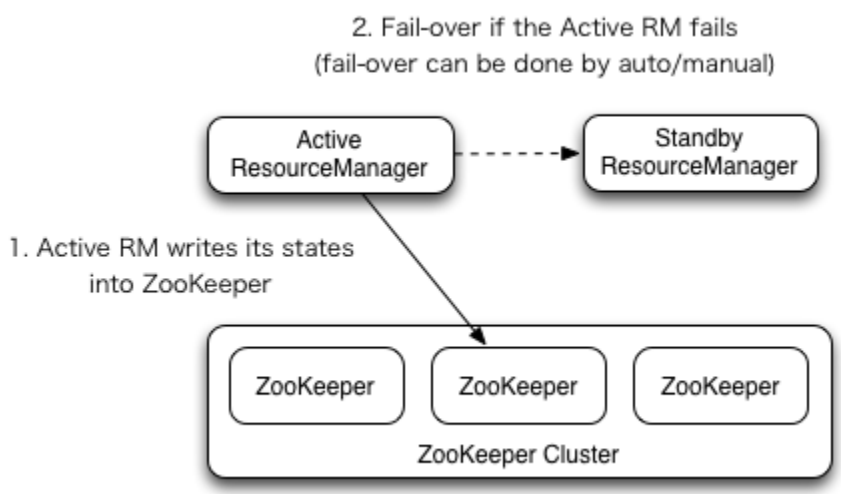
When can failure occur?	• Due to hardware or network failures
How to detect for failures?	• AM sends periodic heartbeats to Resource Manager
Restart	• Max-attempts to restart application • Default = 2

Node Manager Failure

When can failure occur?	• Hardware, crashing, slow network
How to detect for failures?	• When a heartbeat is not received by RM for 10mins
Restart	• Tasks of incomplete jobs will be rerun – maybe on different node

Resource Manager Failure

How is failure handled?	• Active Standby configuration
Impact	• More serious as all tasks fail
Restart	• Handled by failover controller



https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/ResourceManagerHA.html

Benefits of YARN

YARN Benefits – Case Study @Yahoo

- YARN manages a very large cluster at Yahoo
  - Scalability – to over 40,000 servers with 100,000 CPUs, 455 PB of data
    - Runs over 850,000 jobs per day
  - Flexibility
    - Same cluster has Hadoop, Storm and Spark (100 node cluster) sharing resources using YARN

https://www.techrepublic.com/article/why-the-worlds-largest-hadoop-installation-may-soon-become-the-norm/

Review Exercises

Review Exercises

- All problems listed in T1 as part of LO2.5

Solved Exercises

- A 1000 node YARN cluster has no jobs running. Two pools are configured with max of 50% of the resources. A new job requiring 600 nodes is submitted and on starting consumes all 600 nodes. Which YARN scheduler is active?
  - Either FIFO or Fair because they will use the entire cluster if there is no other job.
- Will the failure of task result in failure of the entire job?
  - No. Task will be restarted
- What are speculative duplicates?
  - Tasks that are started when AM determines that there is a slow running task.

Additional Notes, Reference Material and Notes

YARN Further Reading

- Chapter 2.5 of T1
- Chapter 4 in T2
- https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html
- There is a good description of YARN in the Tom White book.
- Also follow links from slides given before

THANK YOU

K V Subramaniam
Dept. of Computer Science and Engineering

subramaniamkv@pes.edu