CACS402: Cloud computing
BCA 7th Semester

Mechi Multiple Campus

Bhadrapur

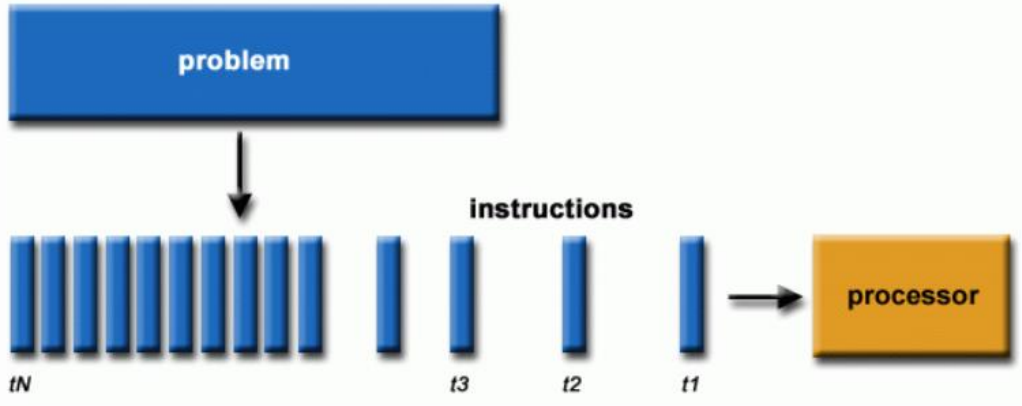
Assistant Professor Krishna Pd. Acharya

MapReduce

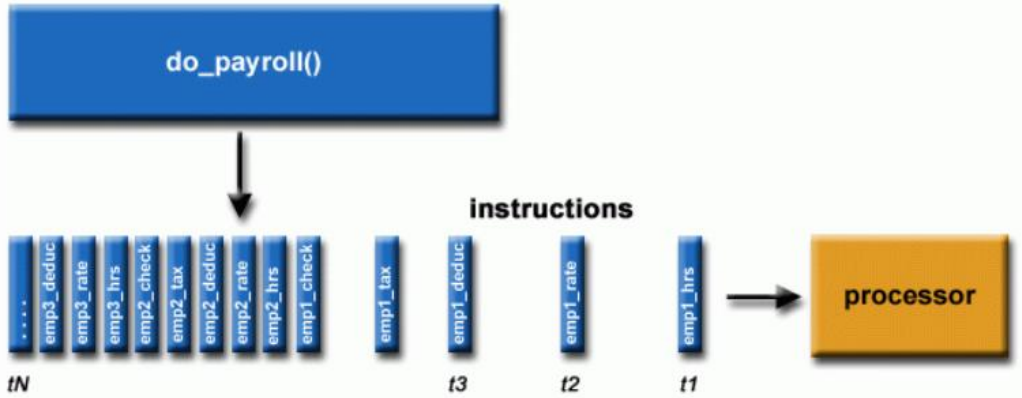
- Introduction to parallel computing
- Map-reduce model
- Applications of map reduce
- Parallel efficiency of Map-Reduced
- MapReduce infrastructure

Serial Computing

- Traditionally, software has been written for ***serial*** computation:
- A problem is broken into a discrete series of instructions.
- Instructions are executed sequentially one after another.
- Executed on a single processor.
- Only one instruction may execute at any moment in time



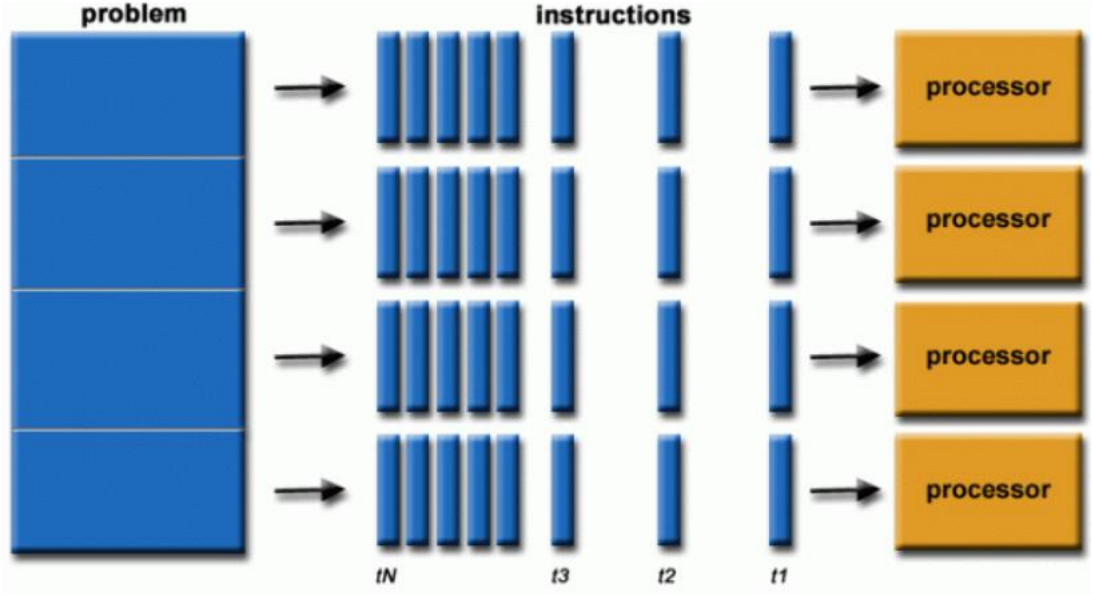
Serial computing generic example



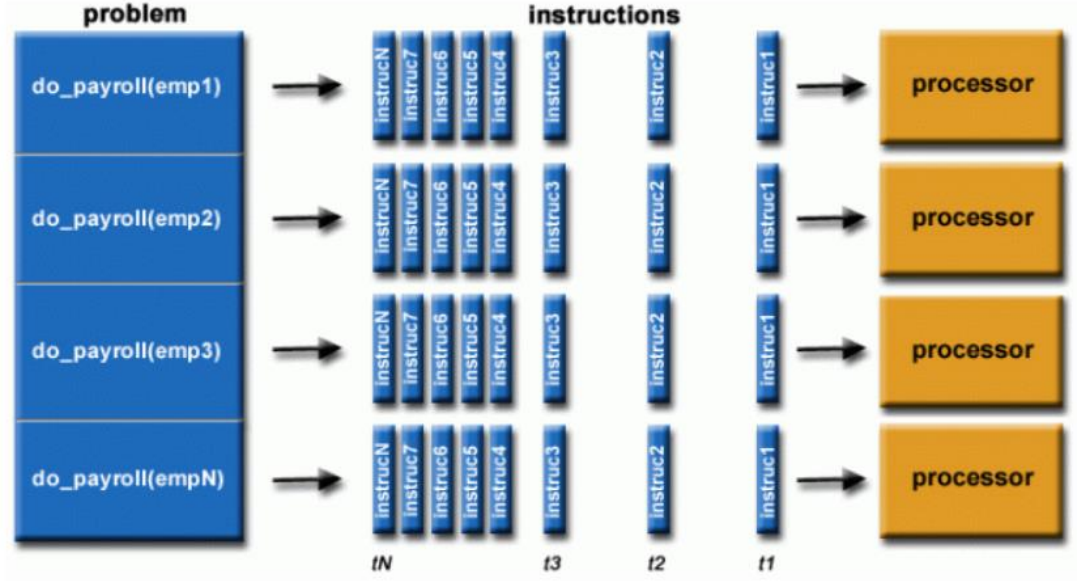
Serial computing example of processing payroll

Parallel Computing

- In the simplest sense, ***parallel computing*** is the simultaneous use of multiple compute resources to solve a computational problem:
- A problem is broken into discrete parts that can be solved concurrently.
- Each part is further broken down to a series of instructions.
- Instructions from each part execute simultaneously on different processors.
- An overall control/coordination mechanism is employed.



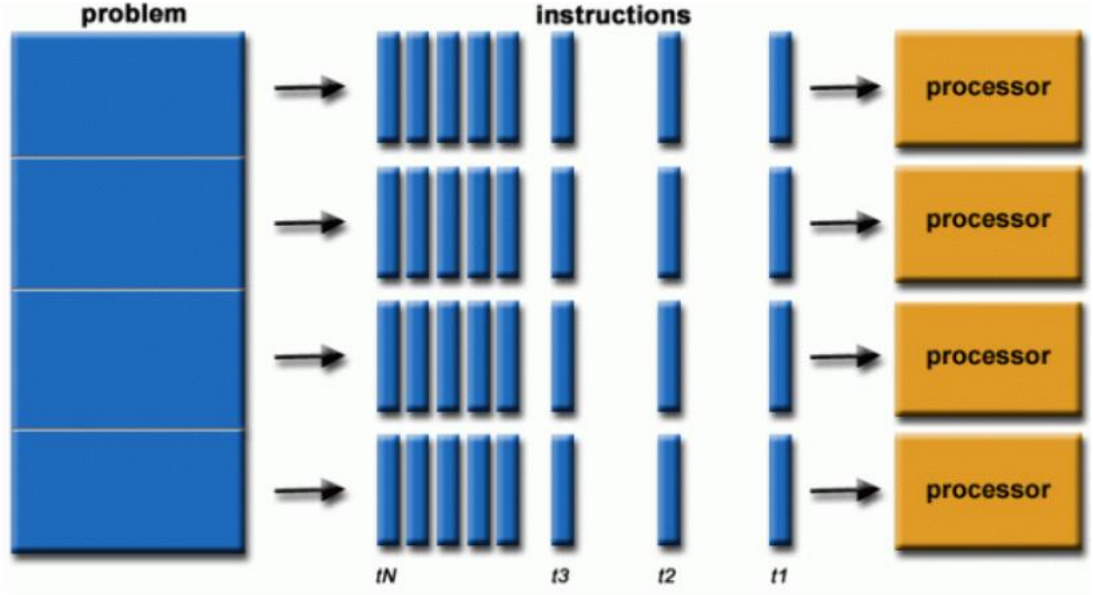
Parallel computing generic example



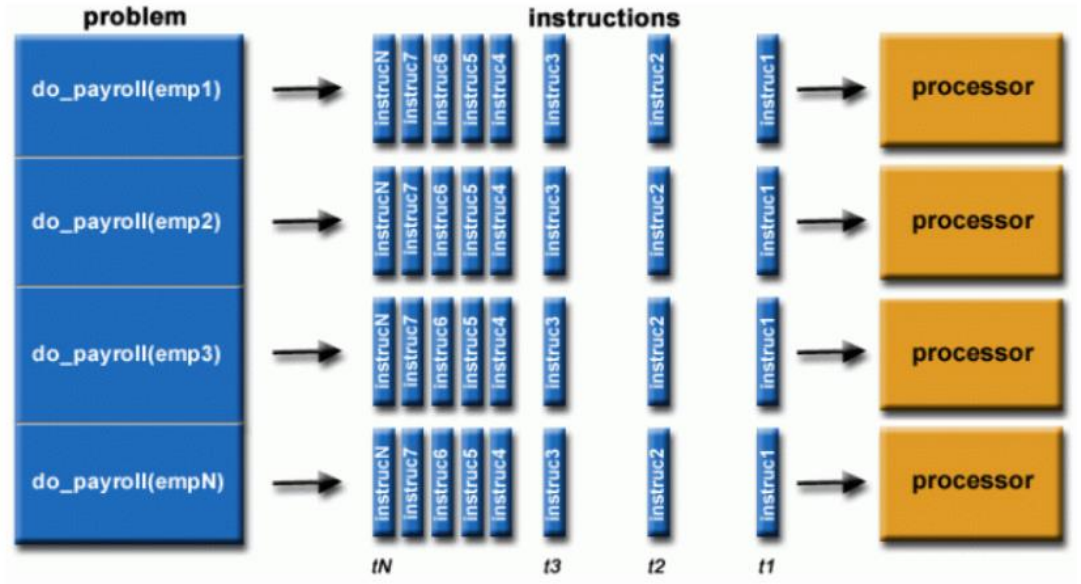
Parallel computing example of processing payroll

Parallel Computing

- The computational problem should be able to:
  - Be broken apart into discrete pieces of work that can be solved simultaneously;
  - Execute multiple program instructions at any moment in time;
  - Be solved in less time with multiple compute resources than with a single compute resource.
- The compute resources are typically:
  - A single computer with multiple processors/cores
  - An arbitrary number of such computers connected by a network



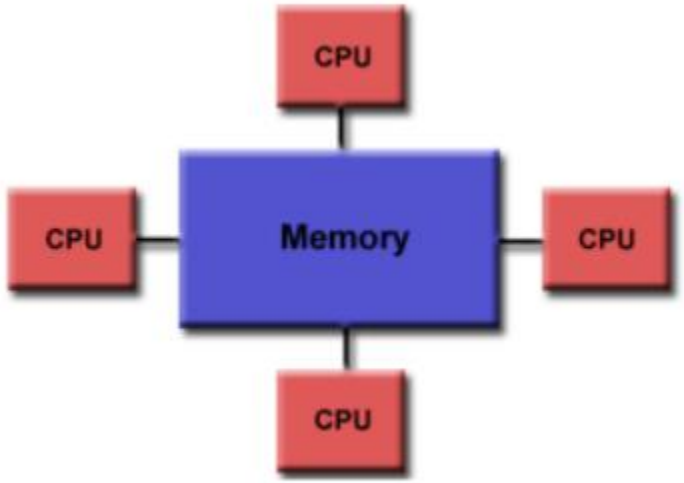
Parallel computing generic example



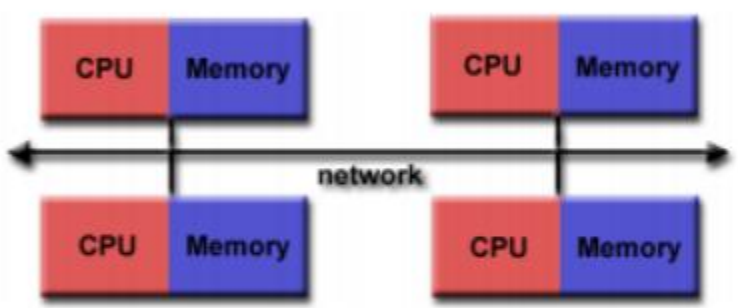
Parallel computing example of processing payroll

Unit 4: MapReduce

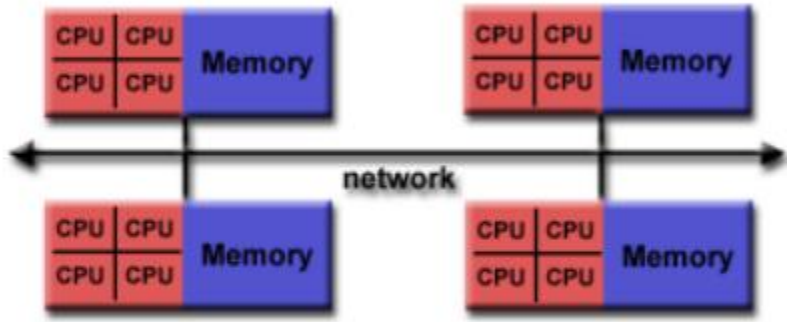
- Introduction to parallel computing
-
-
-
-



Shared Memory



Distributed Memory



Hybrid Distributed-Shared Memory

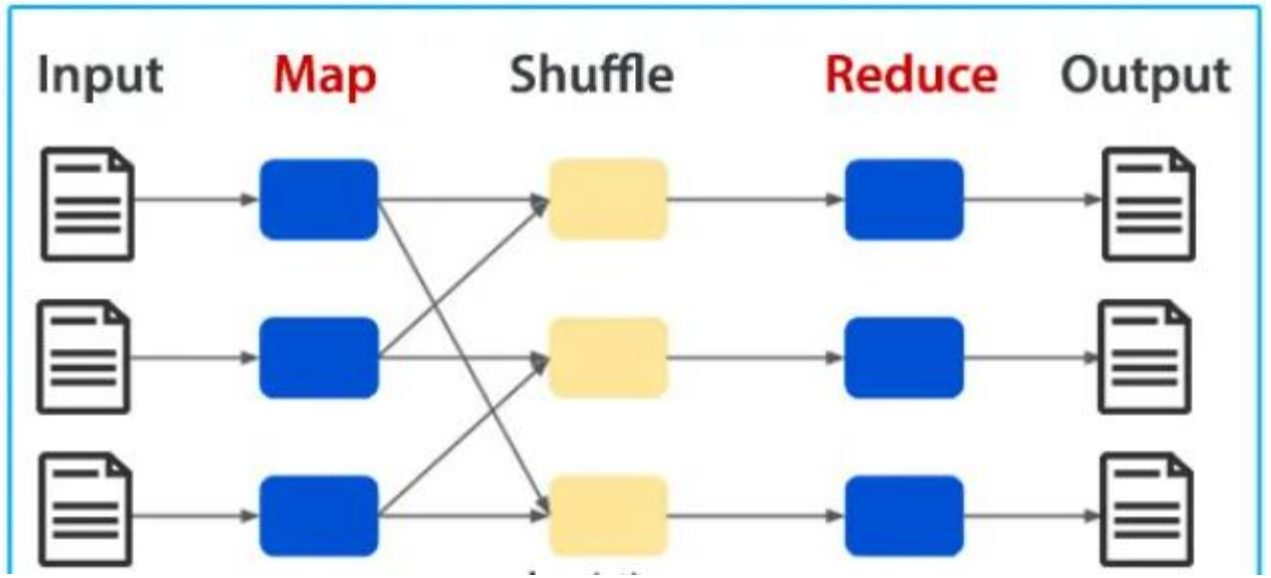
Why Parallel Computing ?

- The real world is massively complex
- Save time and/or money
- Solve larger / more complex problems
- Provide concurrency
- Take advantage of non-local resources
- Make better use of underlying parallel hardware

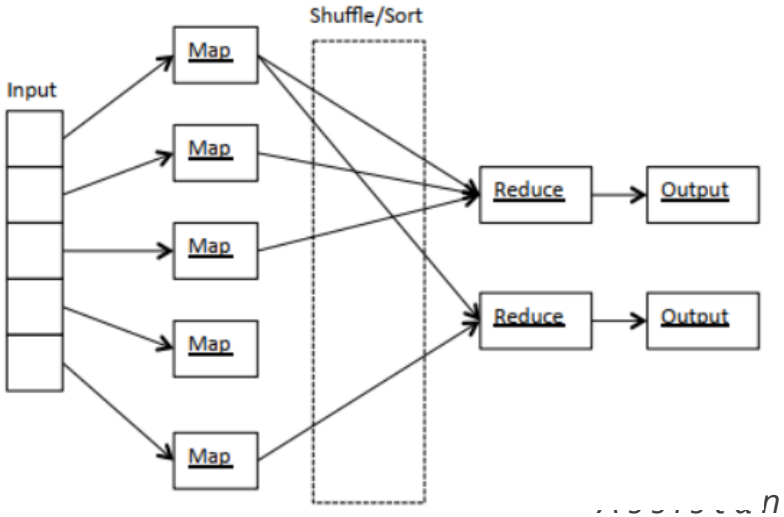
MapReduce

- Big Data can be defined as a huge dataset or collection of such huge datasets that cannot be processed by traditional systems.
- Big Data has become a whole subject in itself which consists of a study of different tools, techniques and frameworks rather than just data. MapReduce is a framework which is used for making applications that help us with processing of huge volume of data on a large cluster of commodity hardware.
- **Why MapReduce?**
- Traditional systems tend to use a centralized server for storing and retrieving data. Such huge amount of data cannot be accommodated by standard database servers. Also, centralized systems create too much of a bottleneck while processing multiple files simultaneously.
- Google, came up with MapReduce to solve such bottleneck issues. MapReduce will divide the task into small parts and process each part independently by assigning them to different systems. After all the parts are processed and analyzed, the output of each computer is collected in one single location and then an output dataset is prepared for the given problem.

How does MapReduce works?



1. Application data is partitioned into smaller homogeneous pieces (chunks)
2. Performs computations on these smaller pieces
3. Aggregated the results in parallel fashion



How does MapReduce works?

- MapReduce is a programming paradigm or model used to process large datasets with a parallel distributed algorithm . In Big Data Analytics, MapReduce plays a crucial role. When it is combined with HDFS. we can use MapReduce to handle Big Data.
- Hadoop Distributed File System (HDFS) is the primary data storage system used by Hadoop applications. HDFS employs a NameNode and DataNode architecture to implement a distributed file system that provides high-performance access to data across highly scalable Hadoop clusters.
- Hadoop itself is an open source distributed processing framework that manages data processing and storage for big data applications.
- The basic unit of information used by MapReduce is a key-value pair. All the data whether structured or unstructured needs to be translated to the key-value pair before it is passed through the MapReduce model.
- MapReduce model as the name suggests has two different functions; Map-function and Reduce-function. The order of operation is always **Map->Shuffle->Reduce**

How does MapReduce works?

- **Map stage**: Map stage is the crucial step in the MapReduce framework. Mapper will give a structure to the unstructured data. For example, if you have to count the songs and music files in my laptop as per genre in my playlist, I will have to analyze the unstructured data.
- The mapper makes key-value pairs from this dataset. So, in this case, the key is genre and value is the music file. Once all this data is given to the mapper, we have a whole dataset ready with the key-value pair structure.
- So the **mapper** will work on one key-value pair at a time. One input may produce any number of outputs. Basically, Map-function will process the data and make several small chunks of data.
- **Reduce stage**: The Shuffle stage and the Reduce stage together are called the Reduce stage. Reducer will take the output from the mapper as an input and make the final output as specified by the programmer.
- This new output will be saved to the HDFS. The Reducer will take all the key-value pairs from the mapper and check the association of all keys with value. All the values associated with a single key will be taken and it will provide an output of any number of key-value pairs.

example

Suppose we have 4 sentences for processing:

1. Red, Green, Blue, Red, Blue
2. Green, Brown, Red, Yellow
3. Yellow, Blue, Green, Orange
4. Yellow, Orange, Red, Blue

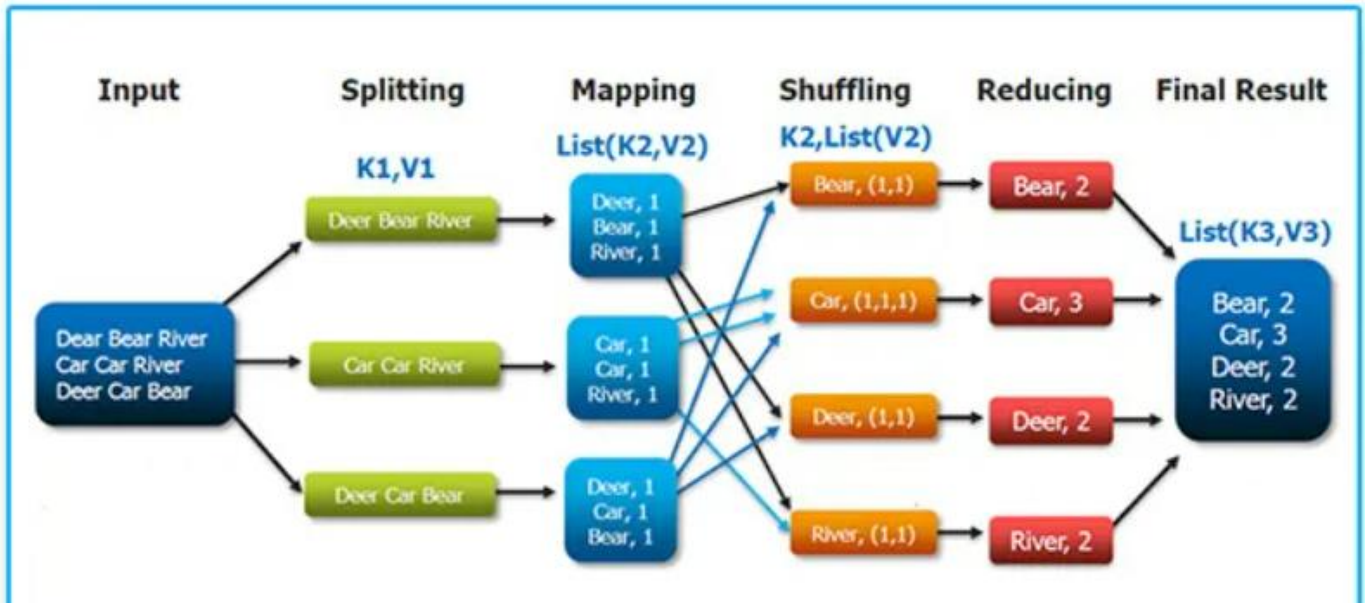
When such an input is passed into the mapper, mapper will divide those into two different subsets.

First will be the subset of the first two sentences and second, the subset of remaining two sentences. Now,

Mapper has:

Subset 1: Red, Green, Blue, Red, Blue and Green, Brown, Red, Yellow

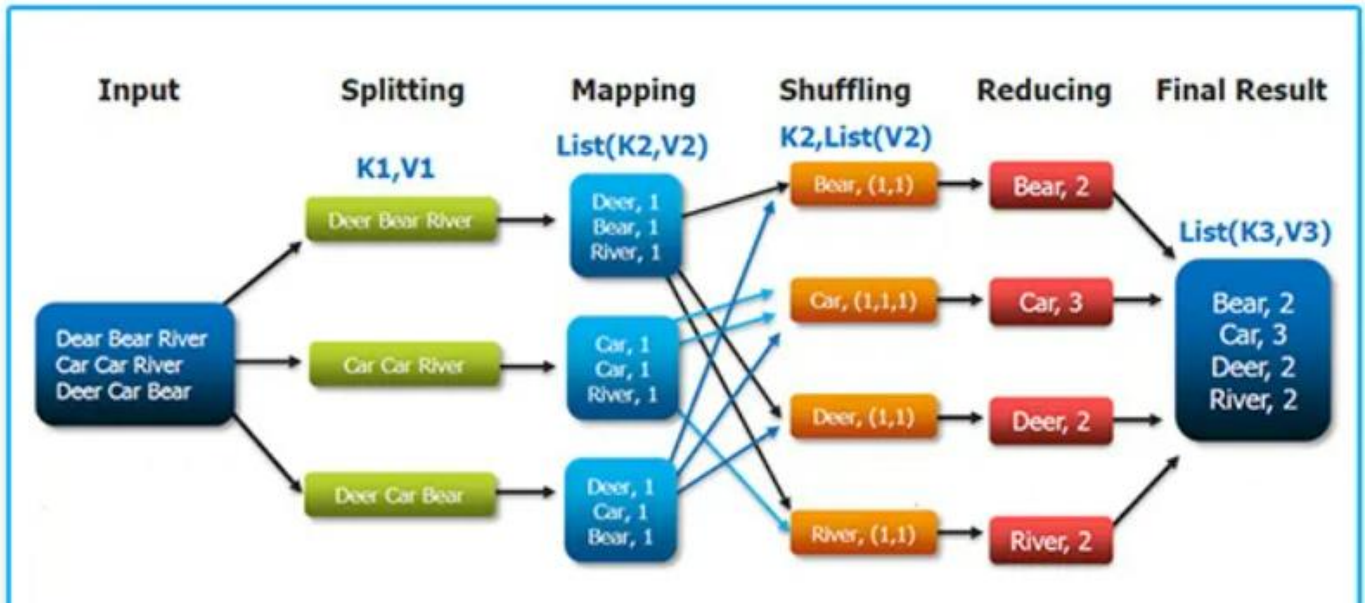
Subset 2: Yellow, Blue, Green, Orange and Yellow, Orange, Red, Blue



example

Mapper will make the key-value pair for each subset. For our example, key is the colour and value is the number of times they have appeared. So, we will have key-value pairs for subset 1 as (Red, 1), (Green, 1), (Blue, 1) and so on. Similarly for subset 2.

Once this is done, the key-value pairs are given to the reducer as input. So, reducer will give us the final count of all the colours in our input subsets and then combine the two outputs. Reducer output will be, (Red, 4), (Green, 3), (Blue, 4), (Brown, 1), (Yellow, 3), (Orange, 2).



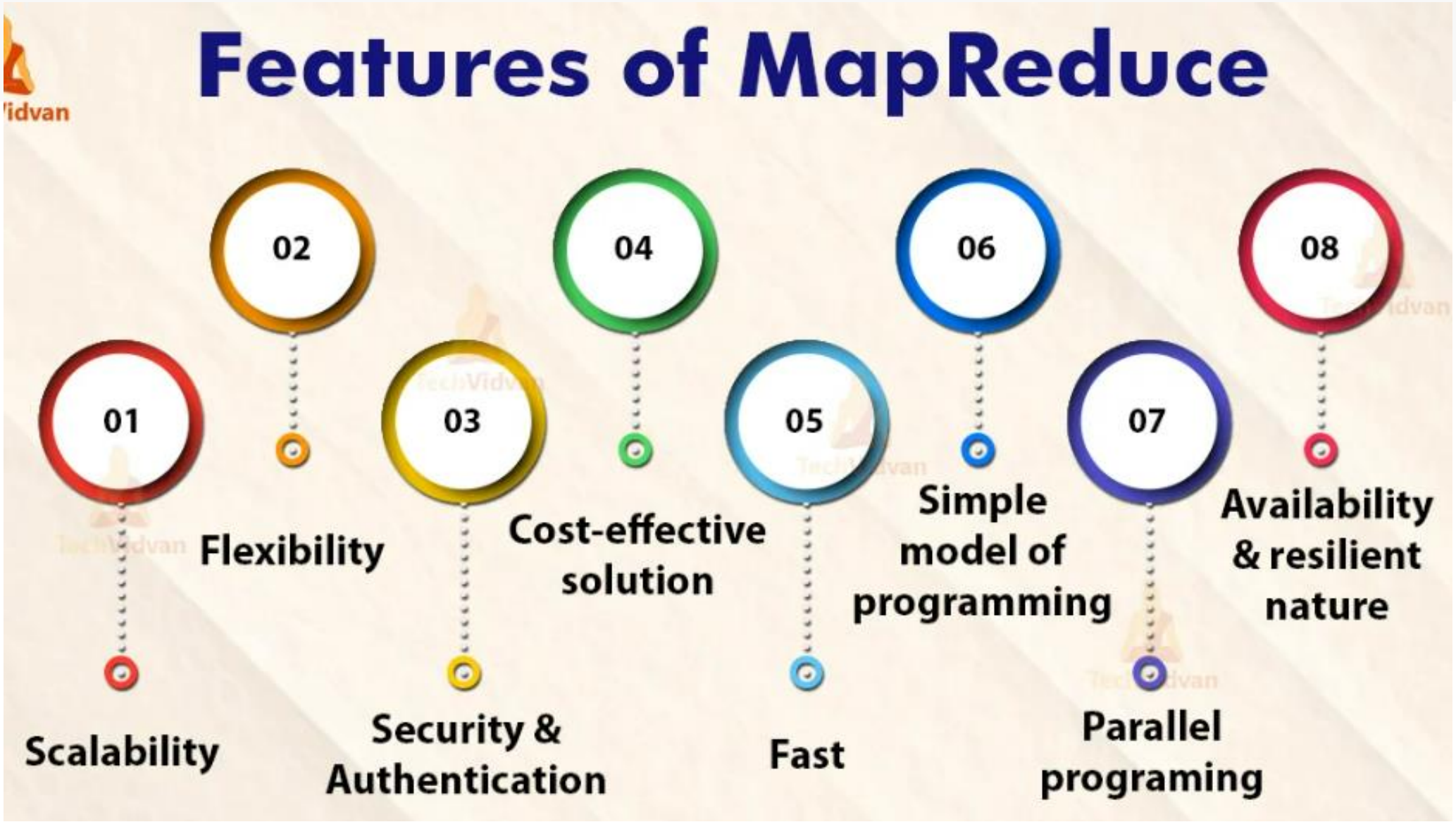
MapReduce Key Points

- The computation takes a set of input key/value pairs, and produces a set of output key/value pairs.
- The user of the MapReduce library expresses the computation as two functions: Map and Reduce.

```
map      (k1,v1)          → list(k2,v2)
reduce   (k2,list(v2))    → list(v2)
```

- Map, written by the user, takes an input pair and produces a set of intermediate key/value pairs.
- The MapReduce library groups together all intermediate values associated with the same intermediate key I and passes them to the Reduce function.
- The Reduce function, also written by the user, accepts an intermediate key I and a set of values for that key.
- It merges together these values to form a possibly smaller set of values.
- Typically just zero or one output value is produced per Reduce invocation.
- The intermediate values are supplied to the user's reduce function via an iterator.
- This allows us to handle lists of values that are too large to fit in memory .

Features of MapReduce



Features of MapReduce

Scalability

- Apache Hadoop is a highly scalable framework. This is because of its ability to store and distribute huge data across plenty of servers. All these servers were inexpensive and can operate in parallel. We can easily scale the storage and computation power by adding servers to the cluster.
- Hadoop MapReduce programming enables organizations to run applications from large sets of nodes which could involve the use of thousands of terabytes of data.
- Hadoop MapReduce programming enables business organizations to run applications from large sets of nodes. This can use thousands of terabytes of data.

Flexibility

- MapReduce programming enables companies to access new sources of data. It enables companies to operate on different types of data. It allows enterprises to access structured as well as unstructured data, and derive significant value by gaining insights from the multiple sources of data.
- Additionally, the MapReduce framework also provides support for the multiple languages and data from sources ranging from email, social media, to clickstream.
- The MapReduce processes data in simple key-value pairs thus supports data type including meta-data, images, and large files. Hence, MapReduce is flexible to deal with data rather than traditional DBMS.

Security and Authentication

- The MapReduce programming model uses HBase and HDFS security platform that allows access only to the authenticated users to operate on the data. Thus, it protects unauthorized access to system data and enhances system security.

Features of MapReduce

Cost-effective solution

- Hadoop's scalable architecture with the MapReduce programming framework allows the storage and processing of large data sets in a very affordable manner.

Fast

- Hadoop uses a distributed storage method called as a Hadoop Distributed File System that basically implements a mapping system for locating data in a cluster.
- The tools that are used for data processing, such as MapReduce programming, are generally located on the very same servers that allow for the faster processing of data.
- So, Even if we are dealing with large volumes of unstructured data, Hadoop MapReduce just takes minutes to process terabytes of data. It can process petabytes of data in just an hour.

Simple model of programming

- Amongst the various features of Hadoop MapReduce, one of the most important features is that it is based on a simple programming model. Basically, this allows programmers to develop the MapReduce programs which can handle tasks easily and efficiently.
- The MapReduce programs can be written in Java, which is not very hard to pick up and is also used widely. So, anyone can easily learn and write MapReduce programs and meet their data processing needs.

Parallel Programming

- One of the major aspects of the working of MapReduce programming is its parallel processing. It divides the tasks in a manner that allows their execution in parallel.The parallel processing allows multiple processors to execute these divided tasks. So the entire program is run in less time.

Features of MapReduce

Availability and resilient nature

- Whenever the data is sent to an individual node, the same set of data is forwarded to some other nodes in a cluster. So, if any particular node suffers from a failure, then there are always other copies present on other nodes that can still be accessed whenever needed. This assures high availability of data.
- One of the major features offered by Apache Hadoop is its fault tolerance. The Hadoop MapReduce framework has the ability to quickly recognizing faults that occur.
- It then applies a quick and automatic recovery solution. This feature makes it a game-changer in the world of big data processing.

MapReduce Architecture

Client: The MapReduce client is the one who brings the Job to the MapReduce for processing. There can be multiple clients available that continuously send jobs for processing to the Hadoop MapReduce Manager.

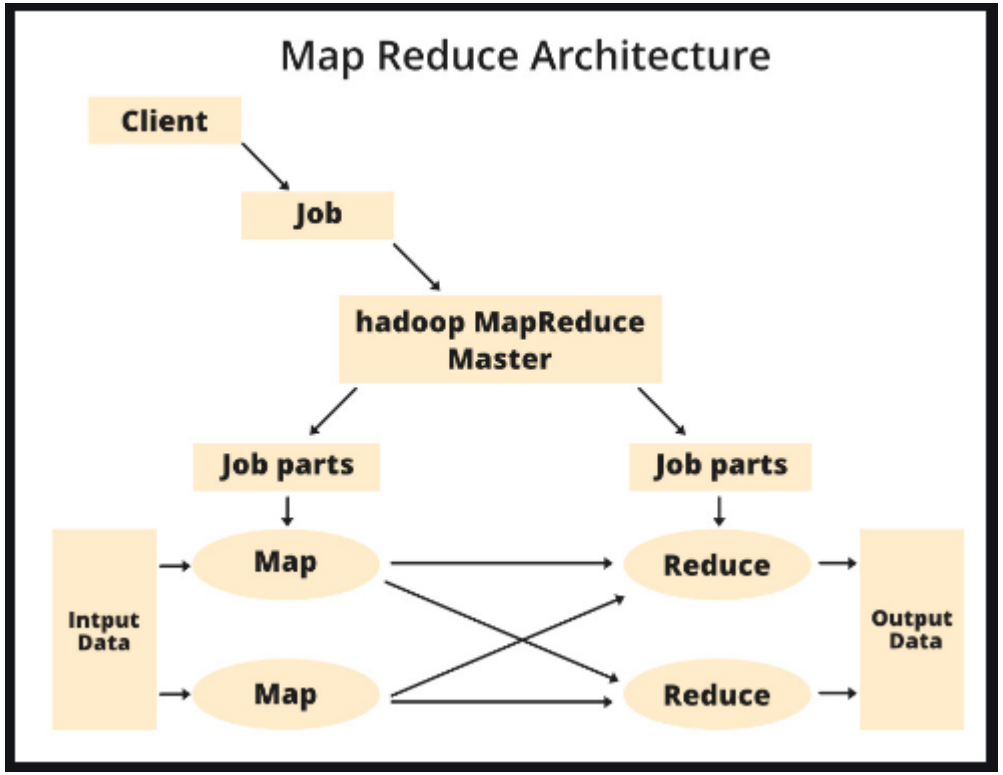
Job: The MapReduce Job is the actual work that the client wanted to do which is comprised of so many smaller tasks that the client wants to process or execute.

Hadoop MapReduce Master: It divides the particular job into subsequent job-parts.

Job-Parts: The task or sub-jobs that are obtained after dividing the main job. The result of all the job-parts combined to produce the final output.

Input Data: The data set that is fed to the MapReduce for processing.

Output Data: The final result is obtained after the processing.



Application of MapReduce

- The functioning of MapReduce like we just went through is a sequential flow and the example was a very small and basic example to understand MapReduce at beginners level.
- The reason why it is admired so much is its capability of parallelism and getting the output based on key-value pair analysis. It is capable of doing big wonders when it comes to Big Data.
- The example was a very small thing but when it comes to real world problems, MapReduce does make it a great choice for easy processing of any volume of data. If Big Data is what you are looking forward to, MapReduce should be the first thing that comes to your mind.
- Large-scale machine learning problems,
- Clustering problems for the Google News and Google products
- Extraction of data used to produce reports of popular queries (e.g. Google Zeitgeist)
- Extraction of properties of web pages for new experiments and products (e.g. extraction of geographical locations from a large corpus of web pages for localized search)
- Large-scale graph computations.