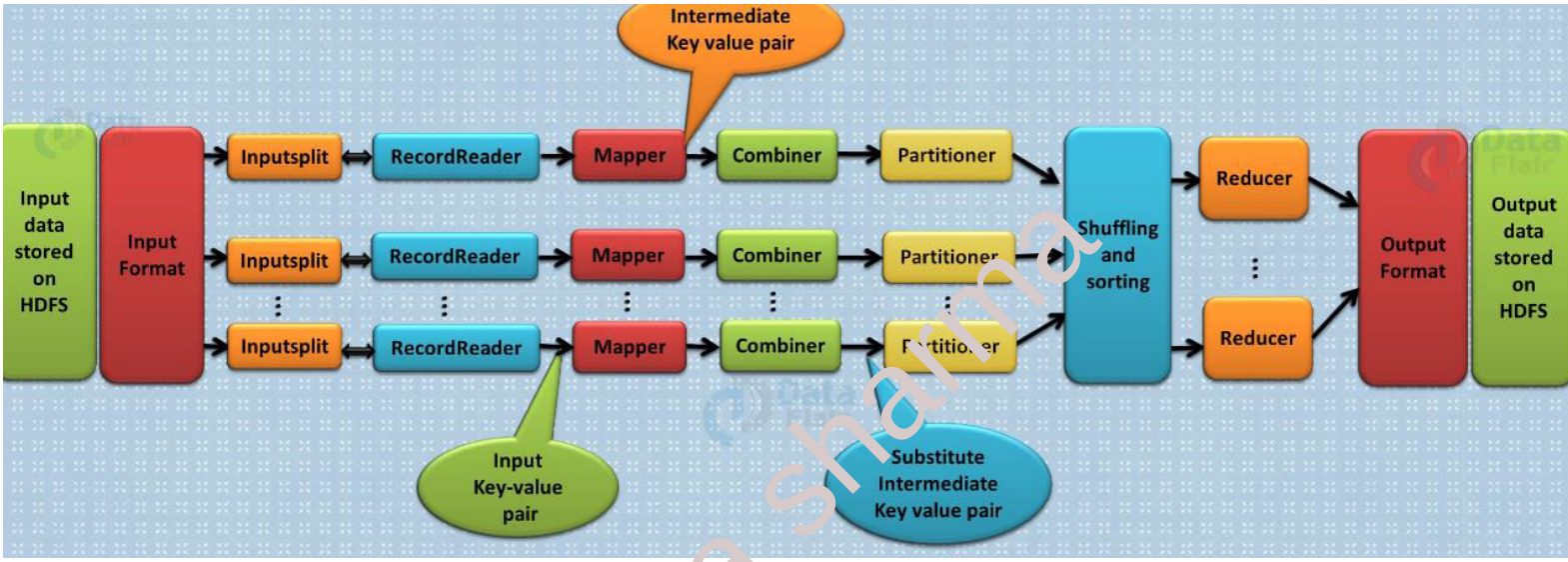
UNIT -2

Hadoop MapReduce

1. **MapReduce** is the core component of **Hadoop** that process huge amount of data in parallel by dividing the work into a set of independent tasks. In MapReduce data flow in step by step from Mapper to Reducer.



Kratika Sharma

2. What is MapReduce?

- **MapReduce** is the data processing layer of Hadoop. It is a software framework for easily writing applications that process the vast amount of structured and unstructured data stored in the Hadoop Distributed Filesystem (HDFS).
- It processes the huge amount of data in parallel by dividing the job (submitted job) into a set of independent tasks (sub-job). By this parallel processing speed and reliability of cluster is improved. We just need to put the custom code (business logic) in the way map reduce works and rest things will be taken care by the engine.

3. How Hadoop MapReduce Works?

- In Hadoop, MapReduce works by breaking the data processing into two phases: Map phase and Reduce phase.
- The map is the first phase of processing, where we specify all the complex logic/business rules/costly code.

- Reduce is the second phase of processing, where we specify light-weight processing like aggregation/summation.

4. MapReduce Flow Chart

Now let us see How Hadoop MapReduce works by understanding the end to end Hadoop MapReduce job execution flow with components in detail:

4.1. Input Files

- The data for a MapReduce task is stored in **input files**, and input files typically lives in **HDFS**.
- The format of these files is arbitrary, while line-based log files and binary format can also be used.

4.2. InputFormat

- Now, **InputFormat** defines how these input files are split and read. It selects the files or other objects that are used for input.
- InputFormat creates InputSplit.

4.3. InputSplits

- It is created by InputFormat, logically represent the data which will be processed by an individual **Mapper** .
- One map task is created for each split; thus the number of map tasks will be equal to the number of InputSplits.
- The split is divided into records and each record will be processed by the mapper.

kratika sharma

4.4. RecordReader

- It communicates with the **InputSplit** in Hadoop MapReduce and converts the data into key-value pairs suitable for reading by the mapper.
- By default, it uses TextInputFormat for converting data into a key-value pair.
- RecordReader communicates with the InputSplit until the file reading is not completed. It assigns byte offset (unique number) to each line present in the file. Further, these key-value pairs are sent to the mapper for further processing.

4.5. Mapper

- It processes each input record (from RecordReader) and generates new key-value pair, and this key-value pair generated by Mapper is completely different from the input pair.
- The output of Mapper is also known as intermediate output which is written to the local disk.

- The output of the Mapper is not stored on HDFS as this is temporary data and writing on HDFS will create unnecessary copies (also HDFS is a high latency system).
- Mappers output is passed to the combiner for further process.

4.6. Combiner

- The combiner is also known **as 'Mini-reducer'**.
- Hadoop MapReduce Combiner performs local aggregation on the mappers' output, which helps to minimize the data transfer between mapper and **reducer** (we will see reducer below).
- Once the combiner functionality is executed, the output is then passed to the partitioner for further work.

4.7. Partitioner

- Hadoop MapReduce, Partitioner comes into the picture if we are working on **more than one reducer (for one reducer partitioner is not used).**
- Partitioner takes the output from combiners and performs partitioning. Partitioning of output takes place on the basis of the key and then sorted. By hash function, key (or a subset of the key) is used to derive the partition.

- According to the key value in MapReduce, each combiner output is partitioned, and a record having the same key value goes into the same partition, and then each partition is sent to a reducer. Partitioning allows even distribution of the map output over the reducer.

kratika sharma

4.8. Shuffling and Sorting

- Now, the output is Shuffled to the reduce node (which is a normal slave node but reduce phase will run here hence called as reducer node).
- The shuffling is the physical movement of the data which is done over the network.
- Once all the mappers are finished and their output is shuffled on the reducer nodes, then this intermediate output is merged and sorted, which is then provided as input to reduce phase.

4.9. Reducer

- It takes the set of intermediate key-value pairs produced by the mappers as the input and then runs a reducer function on each of them to generate the output.
- The output of the reducer is the final output, which is stored in HDFS.

4.10. RecordWriter

- It writes these output key-value pair from the Reducer phase to the output files.

4.11. OutputFormat

- The way these output key-value pairs are written in output files by RecordWriter is determined by the OutputFormat.
-
- OutputFormat instances provided by the Hadoop are used to write files in HDFS or on the local disk. Thus the final output of reducer is written on HDFS by OutputFormat instances.

Hence, in this manner, a Hadoop MapReduce works over the cluster.

Hadoop Combiner

1. Hadoop Combiner / MapReduce Combiner

- Hadoop Combiner is also known as “Mini-Reducer” that summarizes the Mapper output record with the same Key before passing to the Reducer.

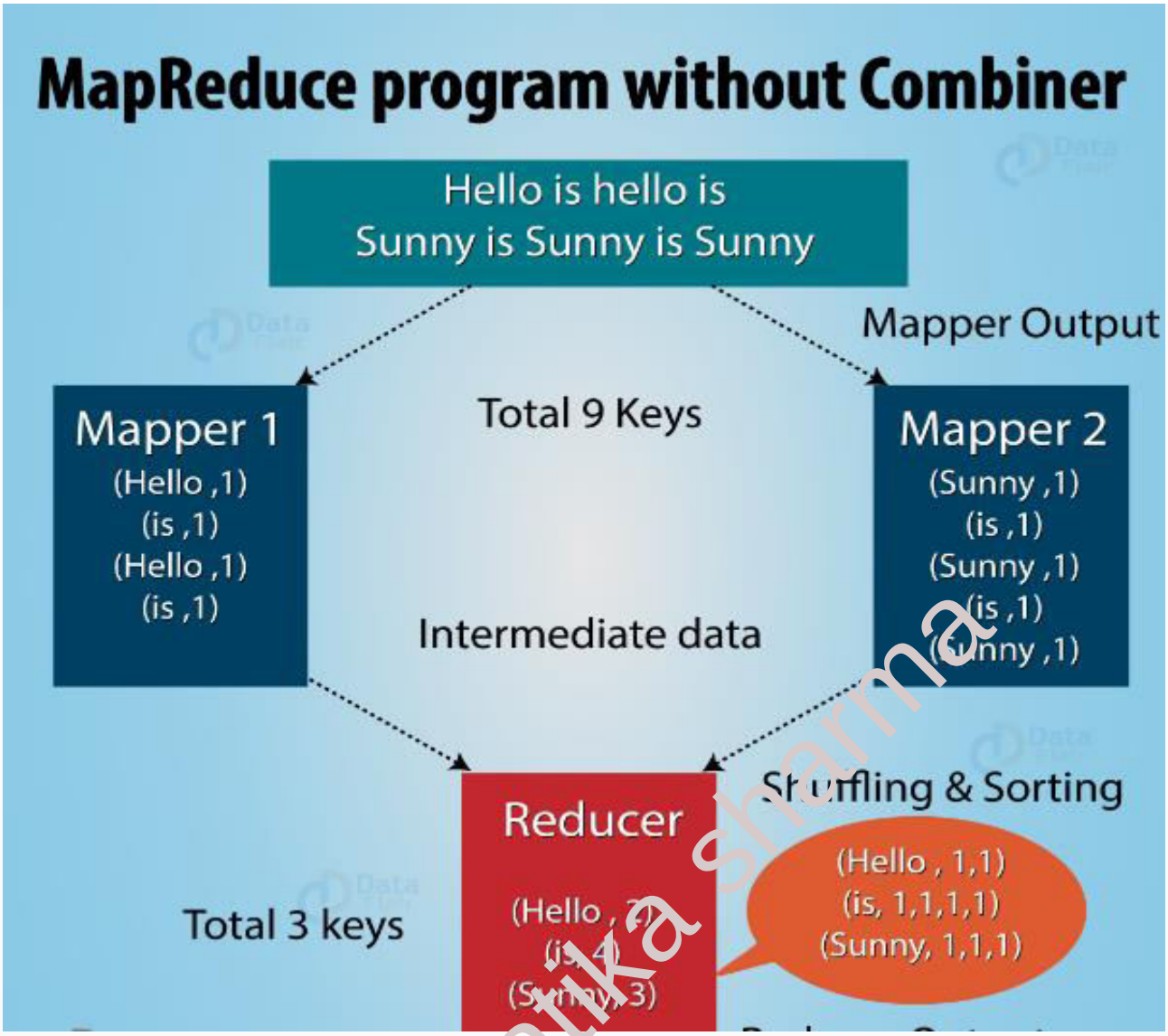
2. What is Hadoop Combiner?

- On a large dataset when we run MapReduce job, large chunks of intermediate data is generated by the Mapper and this intermediate data is passed on the Reducer for further processing, which leads to enormous network congestion.
- MapReduce framework provides a function known as Hadoop Combiner that plays a key role in reducing network congestion.
- The combiner in MapReduce is also known as ‘Mini-reducer’. 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.

Kratika sharma

3. How does MapReduce Combiner works

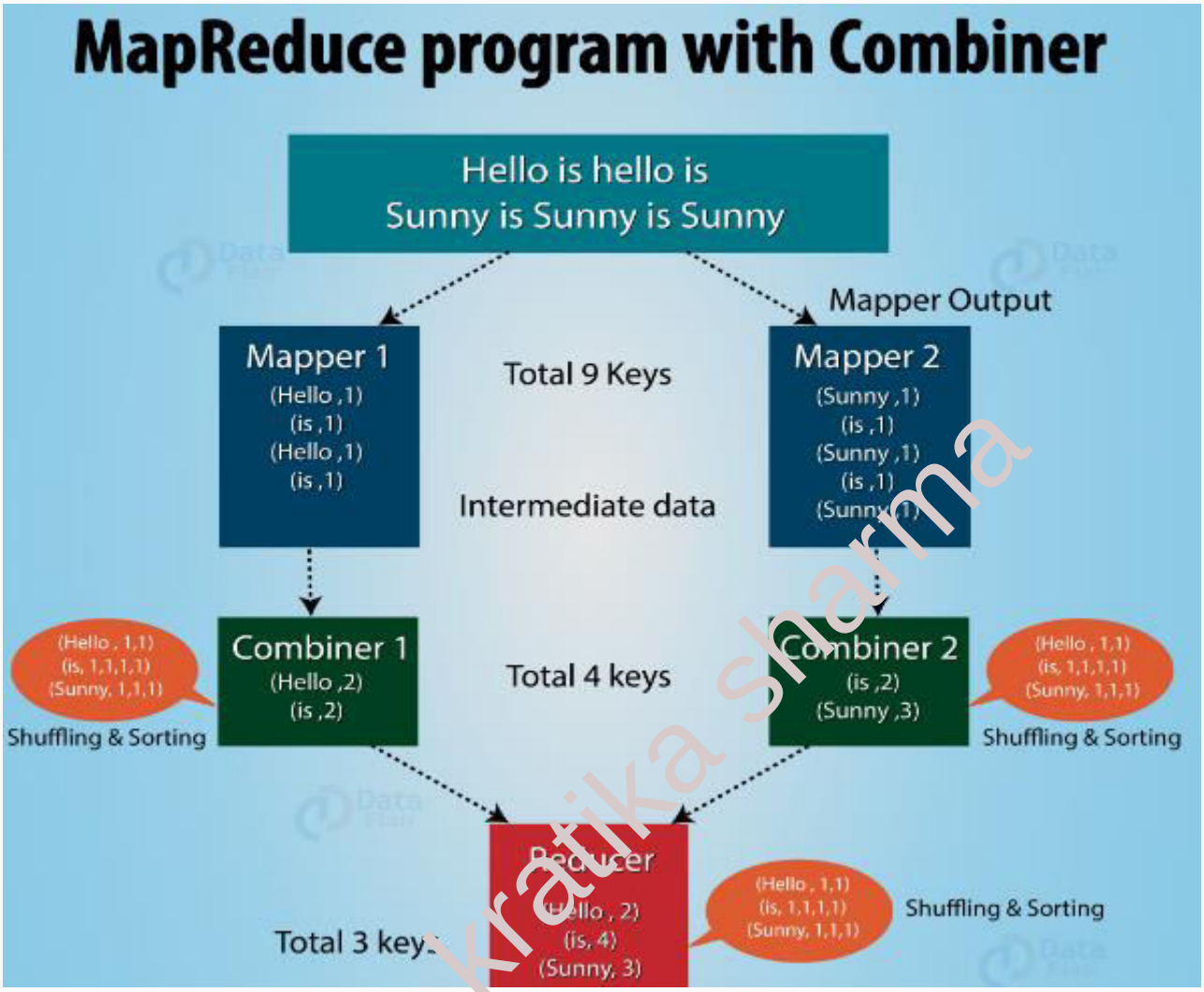
3.1. MapReduce program without Combiner



kratika sharma

- In the above diagram, no combiner is used. Input is split into two mappers and 9 keys are generated from the mappers. Now we have (9 key/value) intermediate data, the further mapper will send directly this data to reducer and while sending data to the reducer, it consumes some network bandwidth (bandwidth means time taken to transfer data between 2 machines). It will take more time to transfer data to reducer if the size of data is big.
- Now in between mapper and reducer if we use a hadoop combiner, then combiner shuffles intermediate data (9 key/value) before sending it to the reducer and generates 4 key/value pair as an output.

3.2. MapReduce program with Combiner in between Mapper and Reducer



- Reducer now needs to process only 4 key/value pair data which is generated from 2 combiners. Thus reducer gets executed only 4 times to produce final output, which increases the overall performance.
- In conclusion, we can say that MapReduce Combiner plays a key role in reducing network congestion. MapReduce combiner improves the overall performance of the reducer by summarizing the output of Mapper.