MapReduce

Group No: B19

Group members:

Name	Student ID
Harshal Thakor	202001169
Milind Patel	202001246
Gaurang Ganvit	202001247
Poorav Patel	202001250
Krunal Patel	202001442

Introduction of mapReduce:

MapReduce is a programming model and algorithm designed to process large amounts of data in a distributed and parallel way. It was first presented by Google and has since gained popularity.

➔ Two primary phases of the MapReduce paradigm.

- ◆ The map stage
  - The input data is partitioned into chunks and processed individually by several worker nodes during the map step. To the supplied data chunk, each worker node executes a map function, producing intermediate key-value pairs.
- ◆ The reduce stage
  - The reduction stage starts when the map stage is finished.The intermediate key-value pairs are now sorted and shuffled according to their keys.

Here's an overview of how these functions work together in MapReduce:

Splitting:

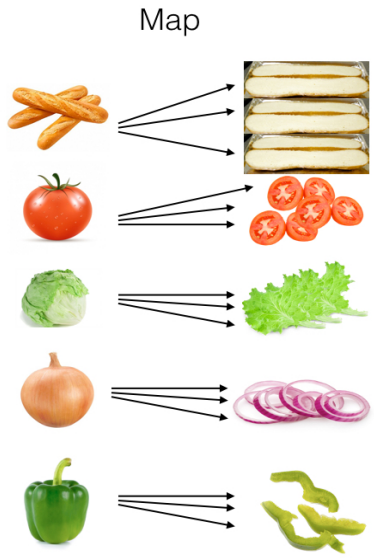
- The input data is divided into smaller chunks, called input splits, which can be processed independently in parallel. The splitting function is responsible for breaking the input data into these input splits. In some cases, the input data may already be split into smaller chunks.
- All input data that our system has to process is divided by some algorithm into approximately equal portions.

Mapping:

- In this phase, the input data is processed in parallel by a set of mapper functions. Each mapper function takes an input split and produces a set of intermediate key-value pairs. The mapping function is responsible for

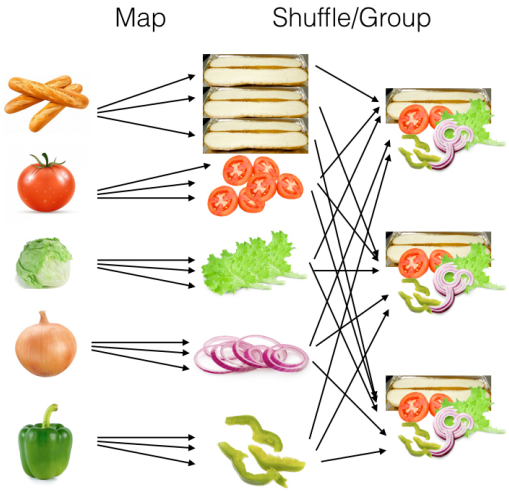
applying a transformation to each input split to produce the intermediate key-value pairs. The intermediate key-value pairs are not yet aggregated, and may have duplicate keys.

- Worker nodes performs all this computations. Master node just sends the tasks to the worker nodes and receives a responses from them.



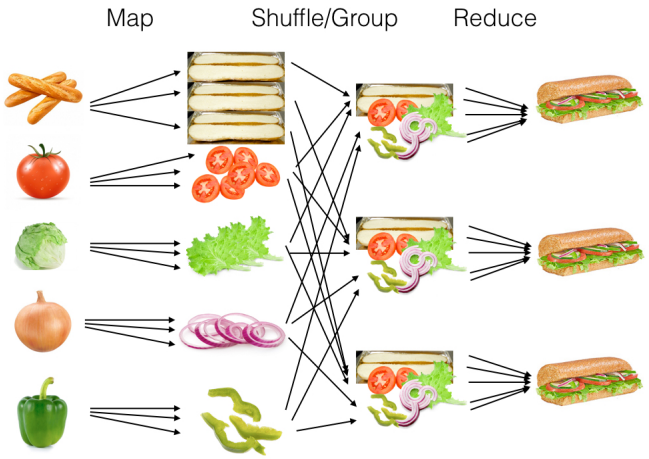
Shuffling:

- The intermediate key-value pairs produced by the mapping phase are then sorted and grouped by key. This is done to prepare the data for the next phase, the reducing phase. The shuffling function is responsible for grouping the intermediate key-value pairs by key, and sending them to the appropriate reducer function.
- This step is optional and may be omitted if this data does not require any conversion. Also, if the computations implied at this stage are laborious, then they can be distributed between worker nodes in the same way as in the previous step.



Reducing:

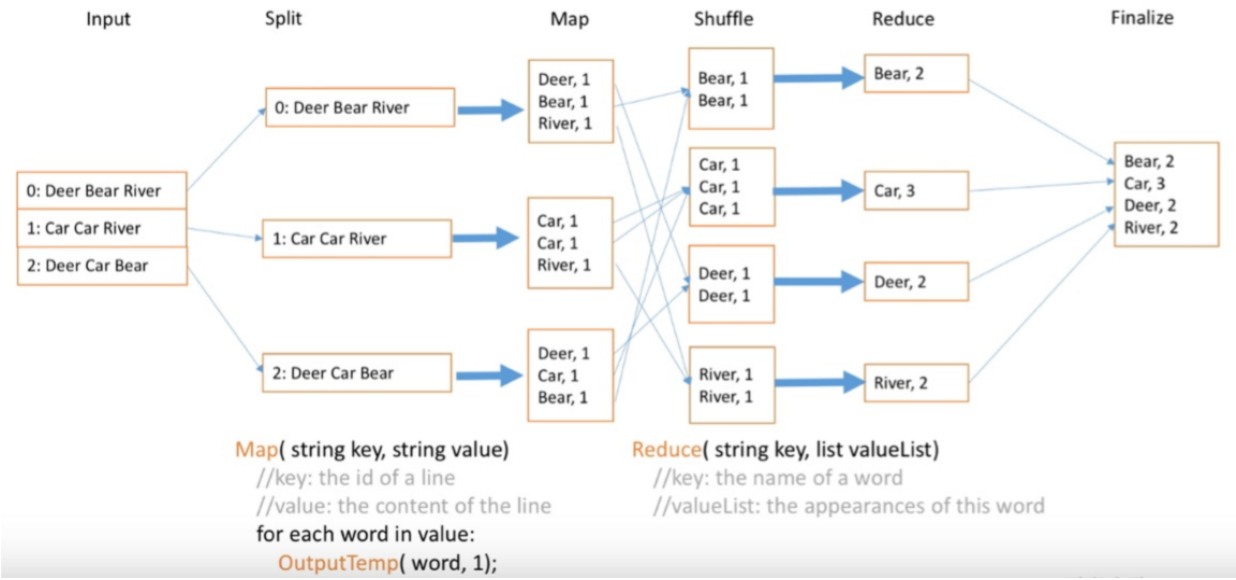
- In this phase, the intermediate key-value pairs are processed in parallel by a set of reducer functions. Each reducer function takes a set of intermediate key-value pairs with the same key, and produces a set of output key-value pairs. The reducing function is responsible for applying an aggregation function to the values associated with each key, and producing the final output key-value pairs.



Advantages of MapReduce:

1. **Scalability:** MapReduce is designed to scale horizontally, meaning that it can handle larger amounts of data by adding more nodes to the cluster. The MapReduce framework automatically distributes data and computation across these nodes, allowing for efficient processing of large data sets.
2. **Flexibility:** MapReduce provides flexibility in terms of programming languages and data sources. It supports a variety of programming languages, including Java, Python, and C++, and can process data from a wide range of sources, such as Hadoop Distributed File System (HDFS), Amazon S3, and Google Cloud Storage.
3. **Faster processing of data:** MapReduce processes data in parallel, which allows for faster processing of large data sets. It divides the processing into two main phases: the map phase, where data is divided into smaller chunks and processed in parallel across multiple nodes, and the reduce phase, where the output of the map phase is combined and processed to produce the final result. This parallel processing can significantly reduce the time it takes to process large amounts of data.
4. **Efficiency**: MapReduce is highly efficient and can process data in parallel, which helps to reduce the time it takes to complete processing tasks. This makes it possible to process large amounts of data in a relatively short amount of time.

Here's an example to illustrate how these functions work together in MapReduce:



Assume we have a large file containing a list of words. We want to count the frequency of each word in the file using MapReduce.

Splitting: The file is divided into smaller input splits, which can be processed independently in parallel. For example, we could split the file into 3 input splits, each containing one quarter of the file.

Mapping: Each input split is processed in parallel by a set of mapper functions. The mapping function takes an input split and produces a set of intermediate key-value pairs. For example, the mapping function could split the input split into individual words, and produce a set of key-value pairs where the key is the word and the value is 1, indicating that the word appears once in the input split.

Shuffling: The intermediate key-value pairs produced by the mapping phase are then sorted and grouped by key. For example, all the intermediate key-value

pairs with the key "bear" would be grouped together, and all the intermediate key-value pairs with the key "car" would be grouped together and so on.

Reducing: Each group of intermediate key-value pairs with the same key is processed in parallel by a set of reducer functions. The reducing function applies an aggregation function to the values associated with each key to produce the final output key-value pairs. For example, the reducing function could sum the values associated with each key to produce the final count for each word.

References :

- MapReduce - Wikipedia
- https://reberhardt.com/cs110/summer-2018/lecture-notes/lecture-14/
- https://coderscat.com/understanding-mapreduce/