Zewail City of Science and Technology

Communications and Information Engineering (CIE) Program

Big Data Analytics (CIE 427) - Fall 2020

Hadoop MapReduce

Original by: Muhammad Hamdy AlAref

The Bits and Pieces

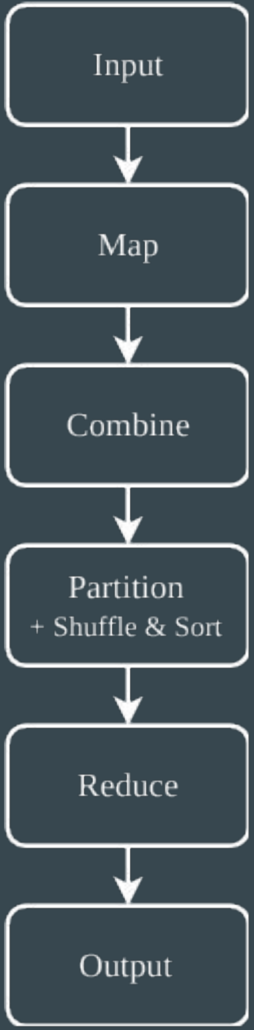
Overview

The MapReduce framework is quite straightforward yet it can be customized at multiple points!

The data moving through the framework is almost always in `<key, value>` format.

The machines that run Map tasks are called *mappers*, and the machines that run Reduce tasks are called *reducers*.

Most machines become both *mappers* and *reducers* at different times during a job.



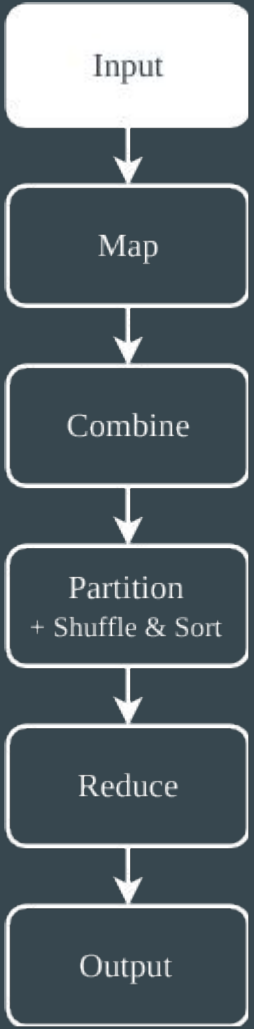
Input

`TextInputFormat` (default) splits each file line by line, emits `<LongWritable position, Text line>`.

`KeyValueTextInputFormat` splits each file line by line, and splits each line to a key and a value using a *separator* emits `<Text key, Text value>`.

`DBInputFormat` splits data from a database.

Many others and custom implementations are also possible.

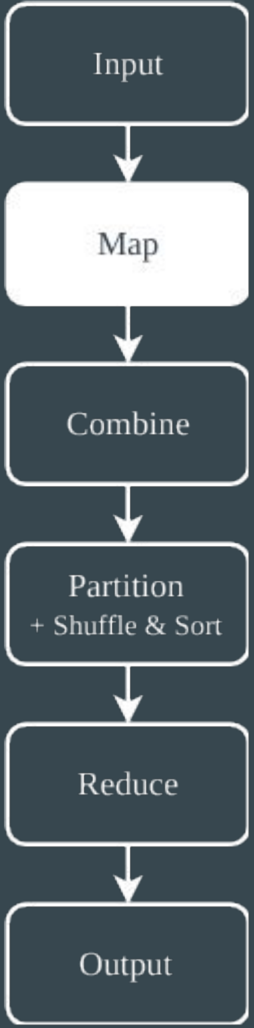


Map

Map is a transformation of the data to intermediate values.

Map takes *one split* of data, i.e., one `<key, value>` pair, and emits zero or more `<key, value>` pairs to the next stage.

Map is run *independently* for each input split.

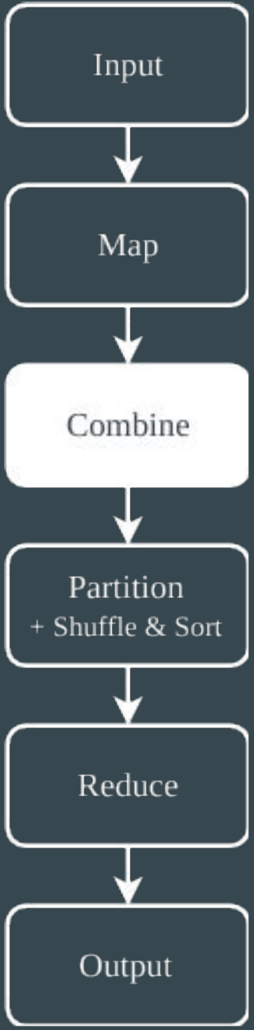


Combine

Combine is a local aggregator that runs on the *mappers*.

Combine runs on the output of Map to minimize data transfers from *mappers* to *reducers*.

Combine typically does the same work as Reduce.



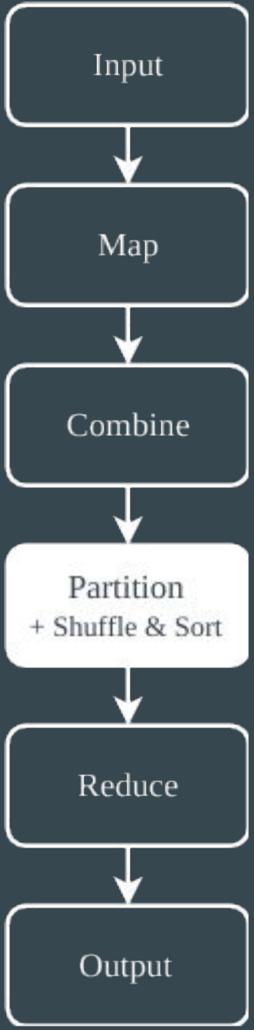
Partition

Partition divides the key space of the intermediate keys to distribute them to the *reducers*.

Partition typically uses a *hash function*.

Partitions are then sent to their designated *reducers* from all relevant *mappers* (and that is called *shuffle*).

Each partition is *sorted by the* `key` at the *reducers* for the next stage.

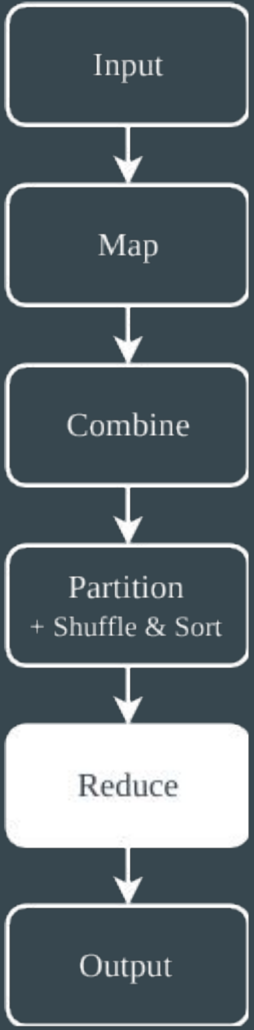


Reduce

Reduce is a the final aggregation of the intermediate values.

Reduce takes *one* `key` and all the `values` paired with that `key` and emits zero or more `<key, value>` pairs to be written to files.

Reduce is run *independently* for each `key`.

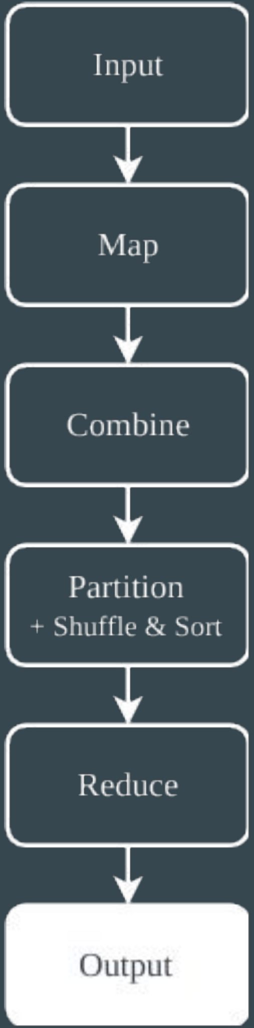


Output

`TextOutputFormat` (default) writes each `<key, value>` pair on one line, separated by a *separator*.

`DBOutputFormat` writes data to a database.

Many others and custom implementations are also possible.



Walkthrough

Which Pieces?

In general, every piece in the pipeline can be customly implemented.
However, for our purposes, we shall...

- choose **Input**
- implement or choose **Map**
- leave **Combine** unchanged or set it as **Reduce**
- leave **Partition (+ Shuffle & Sort)** unchanged
- implement or choose **Reduce**
- leave **Output** unchanged

Implementation

Java

Java is the primary language for the Hadoop framework.

It is sort of the recommended one, as it runs natively without any overhead making it arguably more efficient.

Streaming

Streaming is a utility that enables using executables written in any language as *mappers* and *reducers*.

It uses `stdin` and `stdout` for reading and writing `<key, value>` pairs which may add a little bit of overhead to the job execution.

Word Count Demo

https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html#Example:_WordCount_v1.0

https://www.folio3.ai/wp-content/uploads/2019/04/hadoop-architecture.png

Introduction to PageRank

PageRank is the famous ranking algorithm behind Google's early success. As the name suggents, it assigns a *rank* to every page in their index of the web to help sort search query results with the popular pages on top. It was first developed by Google founders Larry Page and Sergey Brin at Stanford University. In fact, the original MapReduce system was developed by Google (and published later as a paper) to run PageRank-like algorithms for their search engines!

We will discuss PageRank in details later in the course, but for now let's try something simpler!

Requirement: Pseudo-PageRank

Given a sample of the WDC Hyperlink Graphs, write MapReduce jobs to answer the following questions!

- How many outlinks does every domain have?
- How many inlinks does every domain have?
- Which domain has the most outlinks? (BONUS)
- Which domain has the most inlinks? (BONUS)

NOTE You can just use the `arcs` file and look up the domain name manually in the `index` file. Incorporating the `index` file to output the domain name directly gives you more BONUS!

Acknowledgement

A lot of the material introduced in Hadoop MapReduce Bits and Pieces were adapted from the official Apache Hadoop documentation.

Some details were gathered from various sources such as DataFlair.

A lot of the material introduced in Hadoop Streaming were adapted from the official Apache Hadoop documentation.

Thank you!