MapReduce Algorithm

15BCE0290 Varshini S

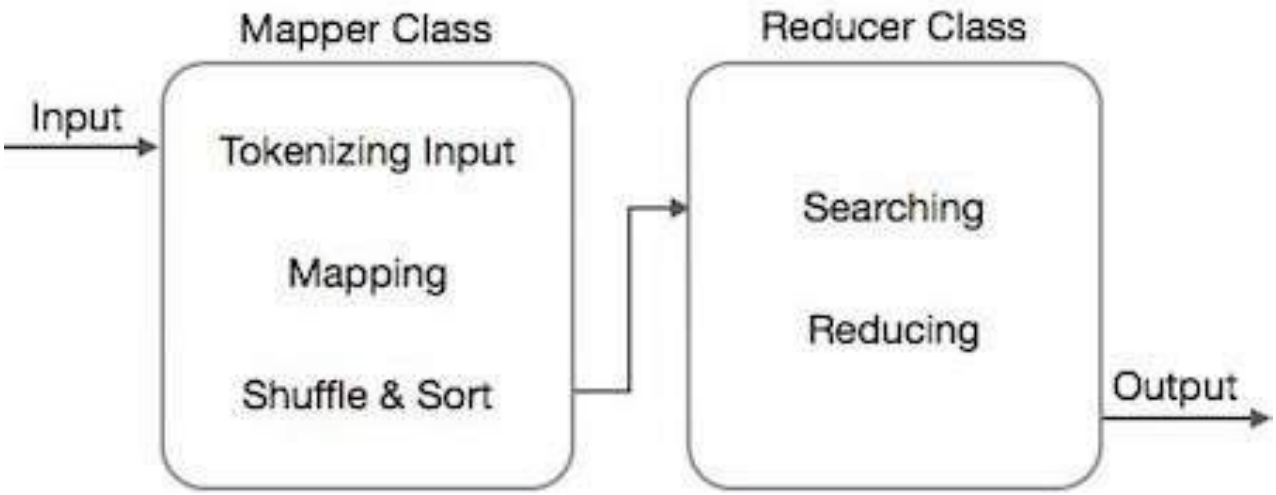
15BCE0491 Devi Pratyusha

15BCE0685 Jandhyala Katyayani

The MapReduce algorithm contains two important tasks, namely Map and Reduce.

- The map task is done by means of Mapper Class
- The reduce task is done by means of Reducer Class.

Mapper class takes the input, tokenizes it, maps and sorts it. The output of Mapper class is used as input by Reducer class, which in turn searches matching pairs and reduces them.



MapReduce implements various mathematical algorithms to divide a task into small parts and assign them to multiple systems. In technical terms, MapReduce algorithm helps in sending the Map & Reduce tasks to appropriate servers in a cluster.

These mathematical algorithms may include the following −

- Sorting
- Searching
- Indexing
- TF-IDF

Sorting

Sorting is one of the basic MapReduce algorithms to process and analyze data. MapReduce implements sorting algorithm to automatically sort the output key-value pairs from the mapper by their keys.

- Sorting methods are implemented in the mapper class itself.
- In the Shuffle and Sort phase, after tokenizing the values in the mapper class, the **Context** class (user-defined class) collects the matching valued keys as a collection.
- To collect similar key-value pairs (intermediate keys), the Mapper class takes the help of **RawComparator** class to sort the key-value pairs.
- The set of intermediate key-value pairs for a given Reducer is automatically sorted by Hadoop to form key-values (K2, {V2, V2, …}) before they are presented to the Reducer.

Algorithm:

• Takes advantage of reducer properties: (key, value) pairs are processed in order by key; reducers are themselves ordered

• Mapper: Identity function for value (k, v) (v, _)

• Reducer: Identity function (k', _) -> (k', "")

Searching

Searching plays an important role in MapReduce algorithm. It helps in the combiner phase (optional) and in the Reducer phase.

Algorithm:
• Mapper: – Given (filename, some text) and "pattern", if "text" matches "pattern" output (filename, _)
• Reducer: – Identity function

Indexing

Normally indexing is used to point to a particular data and its address. It performs batch indexing on the input files for a particular Mapper.

The indexing technique that is normally used in MapReduce is known as **inverted index.** Search engines like Google and Bing use inverted indexing technique.

Algorithm:

• Mapper: For each word in (file, words), map to (word, file)
• Reducer: Identity function
Map Reduce:
map(pageName, pageText):
foreach word w in pageText:
emitIntermediate(w, pageName);
done

```
reduce(word, values):
foreach pageName in values:
AddToOutputList(pageName);
 done
emitFinal(FormattedPageListForWord);
```

TF-IDF

TF-IDF is a text processing algorithm which is short for Term Frequency − Inverse Document Frequency. It is one of the common web analysis algorithms. Here, the term 'frequency' refers to the number of times a term appears in a document.

Term Frequency (TF)

It measures how frequently a particular term occurs in a document. It is calculated by the number of times a word appears in a document divided by the total number of words in that document.

Inverse Document Frequency (IDF)

It measures the importance of a term. It is calculated by the number of documents in the text database divided by the number of documents where a specific term appears.

While computing TF, all the terms are considered equally important. That means, TF counts the term frequency for normal words like “is”, “a”, “what”, etc.

$$\mathrm{tf_i} = \frac{n_i}{\sum_k n_k}$$

$$\mathrm{idf_i} = \log \frac{|D|}{|\{d : t_i \in d\}|}$$

$$\mathrm{tfidf} = \mathrm{tf} \cdot \mathrm{idf}$$