Hadoop MapReduce

Claudio Scheer[1] Gabriell Araujo[1]

[1]Master's Degree in Computer Science
Pontifical Catholic University of Rio Grande do Sul - PUCRS

High Performance for Big Data Applications

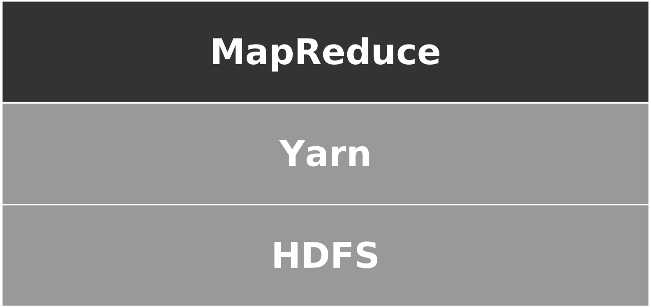
Table of Contents

Table of Contents

1. Hadoop

2. Examples

What we mean by Hadoop



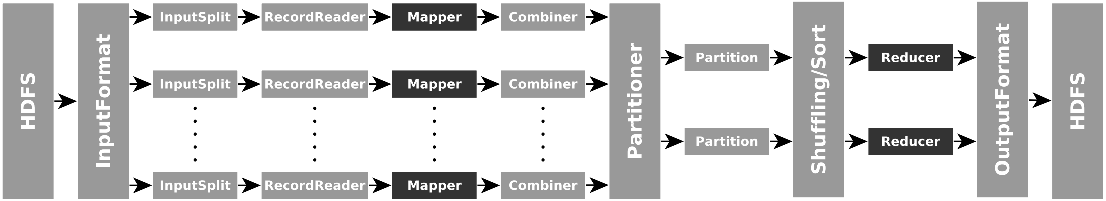
MapReduce

Yarn

HDFS

https://data-flair.training/blogs/hadoop-ecosystem-components

MapReduce execution flow



https://data-flair.training/blogs/hadoop-ecosystem-components

Custom data types

- LongWritable = long;
- IntWritable = int;
- Text = String;
- Other data types (link);

```
public class IntWritable implements WritableComparable<IntWritable> {
  private int value;

  public IntWritable(int value) { set(value); }

  public void set(int value) { this.value = value; }

  public int get() { return value; }

  @Override
  public void readFields(DataInput in) throws IOException {
    value = in.readInt();
  }

  @Override
  public void write(DataOutput out) throws IOException {
    out.writeInt(value);
  }

  @Override
  public int compareTo(IntWritable o) {
    int thisValue = this.value;
    int thatValue = o.value;
    return (thisValue < thatValue ? -1 : (thisValue == thatValue ? 0 : 1));
  }
}
```

▸ Source

InputFormat

- TextInputFormat: <LongWritable, Text>
- KeyValueTextInputFormat: <Text, Text>
  - Key splitted by \t;
- NLineInputFormat: <LongWritable, Text>
  - `config.setInt(NLineInputFormat.LINES_PER_MAP, 256);`
- Customs InputFormat must implement `getSplits` and `getRecordReader`;
  - https://hadoop.apache.org/docs/stable/api/org/apache/hadoop/mapred/InputFormat.html

InputSplit

- Defines the parallelism level;
- Usually splitted by the size of the block;
  - 10TB/128MB = 82000
- It is a logical division of the input data;
- Ways to change it:
  - `config.set(MRJobConfig.NUM_MAPS, 2)`;
  - `mapreduce.job.maps` on `mapsite-site.xml`;

https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

RecordReader

- Split InputSplit into <key, value> pairs;
  - Custom implementations can read values from outside the InputSplit;
- Max record length:
  - `config.setInt(LineRecordReader.MAX_LINE_LENGTH, Integer.MAX_VALUE);`
- Vanilla example:

```
run(Context context) {
    while(context.nextKeyValue())
    {
        map(context.setCurrentKey(), context.getCurrentValue(), context)
    }
}
```

Mapper

- Maps <key1, value1> to <key2, value2>;
- Vanilla example:

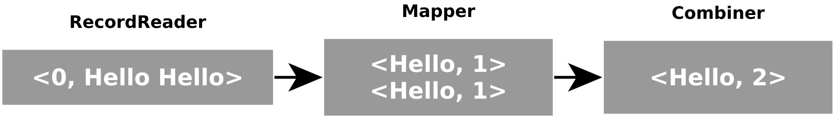
```
public class SimpleMapper extends Mapper<LongWritable, Text, Text, IntWritable> {
    private IntWritable one = new IntWritable(1);
    private Text word = new Text();

    @Override
    public void map(LongWritable key, Text value, Context context) {
        StringTokenizer itr = new StringTokenizer(value.toString());
        while (itr.hasMoreElements()) {
            word.set(itr.nextToken());
            context.write(word, one);
        }
    }
}
```

https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

Combiner

- Reduces data transfer between mapper and reducer;
- Reduces the amount of data to be processed in the reducer;



RecordReader

<0, Hello Hello>

Mapper

<Hello, 1>
<Hello, 1>

Combiner

<Hello, 2>

Partitioner

- Redirects the Combiner output to a specific reducer;
- Has the same number as the number of reducers;
- Used only when there is more than one reducer;
- Vanilla example:

```
public class StupidPartitioner extends Partitioner<Text, IntWritable> {
    public int getPartition(Text key, IntWritable value, int numPartitions) {
        if (value.get() < 35) {
            return 0;
        } else {
            return 1;
        }
    }
}
```

https://hadoop.apache.org/docs/stable/api/org/apache/hadoop/mapreduce/Partitioner.html

Shuffle/Sort

- Collects output from the mappers to the reducers using HTTP requests;
- Sorts the collected <key, value> pairs based on the key;

https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/PluggableShuffleAndPluggableSort.html

https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/EncryptedShuffle.html

Reducer

- Reduces <key2, list(value2)> to <key3, value3>;
- Change the number of reducers:
  - `job.setNumReduceTasks(2);`
- Vanilla example:

```
public static class SimpleReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
    private IntWritable result = new IntWritable();

    @Override
    public void reduce(Text key, Iterable<IntWritable> values, Context context) {
        int sum = 0;
        for (IntWritable val : values) {
            sum += val.get();
        }
        result.set(sum);
        context.write(key, result);
    }
}
```

https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

https://docs.cloudera.com/HDPDocuments/HDP2/HDP-2.6.5/bk_command-line-installation/content/determine-hdp-memory-config.html

OutputFormat

- Specifies how reducer output will be written;
- TextInputFormat: <LongWritable, Text>
- KeyValueTextInputFormat: <Text, Text>
  - Key splitted by \t;
- NLineInputFormat: <LongWritable, Text>
  - `config.set(TextOutputFormat.SEPARATOR, ‘‘,’’);`
- Customs InputFormat must implement `getSplits` and `getRecordReader`;
  - https://hadoop.apache.org/docs/stable/api/org/apache/hadoop/mapred/InputFormat.html

Table of Contents

1 Hadoop

2 Examples

Unquestionable!

Talk is cheap. Show me the code.

— Linus Torvalds —

Two-column slide

This is a text in first column.

$$E = mc^2$$

- First item
- Second item

This text will be in the second column and on a second tought this is a nice looking layout in some cases.