Cloud Computing

A Presentation by Ole Methler
Contents

• What is Cloud Computing?
  • How does it work?
  • Where did it come from?

• What is MapReduce, how do we use it in Cloud Computing?

• What are Containers, how can we use them?

• What is the future of Cloud Computing?

• Bibliography
Cloud Computing

• “Cloud computing is the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user”. (Wikipedia)

• Main Types of Cloud Computing:
  • Infrastructure as a Service, IaaS (computing power, data storage)
  • Platform as a Service, PaaS (databases, webservers)
  • Software as a Service, SaaS (email, game-streaming)
  • Mobile “backend” as a service and Serverless computing
Cloud Computing

- Data is Stored in a separate server
- When requested by the system, data is provided by the cloud
- Data can also be processed in the Cloud
Cloud Computing

- The cloud symbol/metaphor originates from diagrams where the cloud was used to represent “other” systems, that were not important to the main one.
Cloud Computing

Advantages
• Allows much smaller enterprises to work on much larger scales

Disadvantages
• High frequency, stable internet connection required
• Additional cost
History of Cloud Computing

Early development stated in the 1980s -1990s

2000s

By 2012, Microsoft, IBM, Google and others launched their own Cloud Computing platforms

The concept was popularized in the 2000s with the release of Amazon’s „Elastic Compute Cloud“
MapReduce is a processing technique that can be used to process huge amounts of data simultaneously by sending the computer to the data instead of the other way around. It is made up of two stages:

- **Mapping/Map stage** – Data is processed by a "mapper" and divided into several small chunks of data.
- **Reducing/Reduce stage** – These chunks are first "shuffled" and then processed individually.

The Reduced data is stored in the same place.
Hadoop, MapReduce and HDFS

Hadoop
- A framework for data analysis
- Developed by Apache

MapReduce
- A method that is implemented by hadoop
- Developed by Google

Hadoop File System (HDFS)
- The file system that makes access to the data possible
MapReduce wordcount example
Youth is not a time of life; it is a state of mind; it is not a matter of rosy cheeks, red lips and supple knees;
it is a matter of the will, a quality of the imagination, a vigor of the emotions; it is the freshness of the deep springs of life.

Youth means a temperamental predominance of courage over timidity of the appetite, for adventure over the love of ease.

This often exists in a man of sixty more than a body of twenty. Nobody grows old merely by a number of years. We grow old desiring our ideals.

Years may wrinkle the skin, but to give up enthusiasm wrinkles the soul. Worry, fear, self-distrust bows the heart and turns the spirit back to dust.

Whether sixty or sixteen, there is in every human being's heart the lure of wonder, the unfailing child-like appetite of what's next, and the joy of the game of living.

In the center of your heart and my heart there is a wireless station; so long as it receives messages of beauty, hope, cheer, courage and power from men and from the Infinite, so long are you young.

When the aerials are down, and your spirit is covered with snows of cynicism and the ice of pessimism, then you are grown old, even at twenty, but as long as your aerials are up, to catch the waves of optimism, there is hope you may die young at eighty.

"Youth" by Samuel Ullman
package polyu.bigdata;

import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

    // Mapper which implement the mapper() function
    public static class TokenizerMapper extends Mapper<Object, Text, Text, IntWritable> {

        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(Object key, Text value, Context context) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                context.write(word, one);
            }
        }
    }
}
//Reducer which implement the reduce() function
public static class IntSumReducer extends Reducer<Text, IntWritable, Text, IntWritable> {
    private IntWritable result = new IntWritable();

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

//Driver class to specify the Mapper and Reducer
public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
~/.Programs/hadoop/sbin/start-all.sh

- Starts the hadoop application

~/.Programs/hadoop/bin/hadoop fs -mkdir -p /user/bigdata/wordcount/input

- Creates a directory for the input

~/.Programs/hadoop/bin/hadoop fs -put ~/.hadoop.txt /user/bigdata/wordcount/input

- Uploads the input text file into the file system

~/.Programs/hadoop/bin/hadoop jar ~/.wordcount.jar polyu.bigdata.WordCount /user/bigdata/wordcount/input /user/bigdata/wordcount/output

- Executes the Code within the hadoop framework
In 1
Infinite, 1
Nobody 1
This 1
We 1
When 1
Which 1
Whether 1
Worry, 1
Years 1
Youth 2
a 11
adventure 1
eagles 2
and 8
A container is a System that is packaged into a singular unit for simpler usage.

Contained software can run on any system with minimal adjustments.

Singularity creates Containers for scientific computing with the main goals being reproducibility and mobility.
<table>
<thead>
<tr>
<th>Uses of Containers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Publish</td>
</tr>
<tr>
<td>Publish software easily</td>
</tr>
<tr>
<td>Integrate</td>
</tr>
<tr>
<td>Integrate programs into an existing system</td>
</tr>
<tr>
<td>Run</td>
</tr>
<tr>
<td>Run Linux-based software on Windows</td>
</tr>
<tr>
<td>Simplify</td>
</tr>
<tr>
<td>Simplify the process of similar tasks</td>
</tr>
</tbody>
</table>
How to use Singularity

- Running containers from the Cloud
- `sudo singularity build (platform)://(path)`
  - `sudo singularity build docker://sequenciq/hadoop-docker`
  - `sudo singularity pull library://library/default/ubuntu:18.04`
- Either downloading, or running directly in the cloud
How to use Singularity

• Building your own containers
• Requires a definition file
  • Consists of a header and a few sections defining how the container
• sudo singularity build (container.sif) (definition.def)
The Future of Cloud Computing

- Artificial Intelligence
- Gaming
- More Integration into Non-Tech Companies
- General Expansion
Thank you for watching