Cloud computing course 2018 - distributed computing with Apache Spark

Michał Bryła

January 2019

1 Introduction

This is a final project for the cloud computing course at Warsaw University of Technology, Faculty of Power and Aeronautical Engineering. First step was to apply for a student grant - Amazon provides a 40 USD in AWS Credits, which allows to test high power instances for free.Then, first instance was set up in order to test basic features of EC2 Ubuntu instance and to learn some basic things that were used later, for example how to connect to the instance using Putty with private key authentication.

Next step was to automate control of the first instance. Without it, every time the instance was used, it had to be manually started from AWS control panel and stopped after the work was done. Few Python scripts were written, which were used to start, stop, get public DNS and to connect to instance with Putty.

2 Installing Apache Spark

After learning basics, Apache Spark was installed. Amazon provides tools for automated setup of computing clusters, but they weren't used - Spark had been set up manually on the master instance, then the script was written to automate set up and configuration of slave instances.

3 Adding slave nodes

After initial setup of Apache Spark, it's basic features were tested - interactive console and script deployment Interactive console allows to get hold of Python programming and to play with it - some basic programs were tested.

At this point, one minor problem occured - I couldn't login into the instance I've created. Solution was simple - when instance was created I've chosen, that incoming traffic should be accepted only from one IP - mine, which isn't static by default in my ISP. Then, example script was locally deployed - kmeans.py.

In order to do so, numpy needed to be installed and some example data had to be generated - it was generated using make$_blobsfunctionfromscipy$

I've had problems with kmeans.py script - despite many attempts, I didn't make it work. In order to test my cluster, I've modified pi.py script to return it's execution time along estimated pi value, I've also modified it to be longer. After few attempts to make the execution as long as possible, the physical limit has been found - the instance didn't have enough memory. The longest run made was 13.02 seconds. This may be not enough to show the advantage of using distributed computing (comunicating between nodes also takes some time).

4 Adding slave nodes

First, new instances for slave noded had to be created. The script written in second step wasn't used - I've created image of first, master instance and launched another two instances using this image.

I've started the master node on master instance with start-master.sh script, then, I've manually started worker nodes on both of slave nodes, by launching start-slave.sh script with master node address as a parameter Then, interactive console was run on master node, to check if everything works - it worked, so I've moved to next step

5 Launching example application

inally, the main task of this project was completed - pi.py had been run in parallel on two slave instances, then, the time of execution was compared: 13.02 seconds for local deploy and 5.5s for distributed computing.

It clearly shows that the distributed computing is faster, even if the application is as simple as estimating value of the pi using Monte Carlo method. Adding more nodes probably wouldn't make it much faster - the more nodes we have in cluster, the more impact the comunication would have on such a short execution time.

6 Conclusions

Apache Spark is very powerful tool, which allows user to deploy distributed HPC taks very easily. It also provides scripts which allows to integrate it better with EC2 instances - for example, it can automaticly launch new instances if more worker nodes are needed

AWS provides powerful tools which makes using EC2 instances easier than doing all the things manually. Learning how to use this tools may be a very valuable skill, I would like to have, but at the end of the semester, there is no time for learning anything. I would probably come back to this project, to test a few more scripts or maybe launch more worker nodes to compare execution time.

7 Useful resources

I've used this Amazon tutorial to connect to my first instance using Putty, I've found it really helpful.

In order to write scripts to control my AWS instances, I've used boto3 documentation. It has really useful examples which helped me to understand how to use that tool.

The best source of Spark tutorials I've found is the Sparkour website. In my project, I've mostly used these two tutorials: installing Spark on EC2 and managing Spark clusters.

Last, but not least - I've used example pi estimation script from from Apache Spark examples. This was the script I've modified to measure and compare execution time on local machine vs distributed worker nodes.