Analyzing Data with Spark

Master M2 – Université Grenoble Alpes & Grenoble INP

2019

This assignment is about analyzing a large dataset using Apache Spark. The dataset has been made available by Google. It includes data about a cluster of 12500 machines, and the activity on this cluster during 29 days. This lab is an opportunity to process large amount of data and to implement complex data analyses using Spark.

1 Important information

- The assignment is to be done by groups of at most **2** students.
- The assignment must be implemented using Apache Spark. You can choose to program in Scala or in Python
- Information about the exact deadline and the way to submit your assignment will be provided to you later during the semester. The deadline will be towards the end of the semester.
- If you have any question, do not hesitate to send them by email to Thomas Ropars (thomas.ropars@univ-grenoble-alpes.fr).

2 Collaboration and plagiarism

You are encouraged to discuss ideas and problems related to this project with the other students. You can also look for additional resources on the Internet. However, we consider plagiarism very seriously. Hence, if any part of your final submission reflects influences from external sources, you must cite these external sources in your report. Also, any part of your design, your implementation, and your report should come from you and not from other students/sources. We will run tests to detect similarities between source codes. In case of plagiarism, your submission will not be graded and appropriate actions will be taken.

3 Your submission

Your submission must be an archive named with the last name of the two students involved in the project: `Name1_Name2_labSpark.tar.gz`.

The archive should include:

- Your report: a short file either in `md` (MarkDown) or `pdf` format[1], which should include the following sections:
  - The name of the participants.
  - A description of the analyses that you have conducted in Spark (including a description of the problems that you had to tackle if any).
  - A presentation of the corresponding results (displayed as graphs when possible)
  - A description of your results regarding analyzing the performance of Spark (if you chose to study this point)
- The code corresponding to the results presented in your report. Your code should be properly commented.

Grading: The following criteria will be taken into account for grading your work:

- The correctness and the significance of the analyses you have conducted
- The quality of your report and of your code
- The extent of your experiments

4 The dataset

4.1 About the dataset

The data we will study in this lab have been released by Google in 2011. It represents 29 days of activity in a large scale Google machine (a cluster) featuring about 12.5k machines. It includes information about the jobs executed on this cluster during that period as well as information about the corresponding resource usage.

The starting point to find information about the dataset is the following web page: `https://github.com/google/cluster-data/blob/master/ClusterData2011_2.md`. A documentation including a detailed description of the dataset written by people from Google can be found in this document. We refer to this document as the *Google Documentation* in the following.

We give additional information about the dataset in Section 4.3.

4.2 Downloading the data

The data we are going to manipulate are publicly available on Google Cloud Storage. We have to use the tool `GSUtil` to download them:

[1]Other formats will be rejected.

- Information about `GSUtil` can be found here: `https://cloud.google.com/storage/docs/gsutil`
- We recommend you to install `GSUtil` by directly downloading the archive, as described here: `https://cloud.google.com/storage/docs/gsutil_install#alt-install`
- The *Google Documentation* includes a section that describes how to download the data (see Section "Downloading the data" at the end of the document). In a few words:
  - `gsutil ls gs://clusterdata-2011-2/` allows you to see the available files
  - `gsutil cp gs://clusterdata-2011-2/machine_events/part-00000-of-00001.csv.gz ./` will copy the file `part-00000-of-00001.csv.gz` in the current directory.

Important comment: The total size of the dataset is huge (41 GB of data). Do not copy all the data at once. Large data *tables* have been divided into multiple files. It allows you to copy just one file for one *table* and to test your programs on *small* sub-parts of the data.

4.3 Description of the dataset

As already mentioned, a detailed description of the data we are studying is available in the Google Documentation.

Since this document can be difficult to read, we provide you with an overview of the data in the following. However, to decide which analyses you are going to conduct, we strongly encourage you to read the *Google Documentation*. Indeed, we do not discuss all available data in the following.

Important : The file `schema.csv` available on the data repository describes each of the field included in all CSV files comprised in the dataset. In case of doubt, always refer to this file.

4.3.1 About the machines:

Two data tables provide information about the machines:

Machine events The table about "machine events" gives a description of the machine available in the system:

- Each machine is identified with a unique ID
- The table gives us information about the resources available on each machine: CPU and Memory. These data are normalized between 0 and 1. It means that the machines with the value 1 for the amount of CPUs are the machines including the more CPUs. A machine with the value 0.4 for the number of CPUs include a number of CPUs that corresponds to 40% of the maximum number

- The table gives us information about the machines that went offline and reconnected during the period that the data covers (field *event type*).

Machine attributes The table about "machine attributes" stores information about the attributes of each machine (kernel version, clock speed, etc.). However these data have been *obfuscated* by Google to avoid revealing sensitive information. As such, it will be probably hard for us to make any good use of these data.

4.3.2 About jobs and tasks:

The programs that are executed on the machines are called *jobs*. A job can be composed of multiple *tasks*.

Jobs and tasks go through different states during their life cycle. A simple life cycle would be something like this: a job is SUBMITted and gets put into a pending queue; soon afterwards, it is SCHEDULEd onto a machine and starts running; some time later it FINISHes successfully. A task or a job might also be EVICTed, for instance if high priority tasks have to be executed instead. A detailed description of the possible states is available in the *Google Documentation*.

Job event table This table gives mostly information about the state changes of the jobs executing on the cluster (field *event type*) and the time when these changes occur (field *timestamp*). Each job is identified with a unique ID. Furthermore, each job has a *scheduling class*, that roughly represents how latency-sensitive it is.

Task event table This table includes a large set of information about each task, including:

- The ID of the task
- The ID of the job it belongs to
- The ID of the machine on which it is scheduled
- The priority of the task (A value between 0 and 11, 0 being the lowest priority).
- The amount of CPU cores and Memory space requested for the task (value normalized between 0 and 1, as explained previously).

Task constraints This table contains information about placement constraints for the tasks but since the data are mostly obfuscated, we are going to ignore them.

Task usage This table is the largest one. It provides for each task, information about resource usage aggregated over time windows of 5 minutes. These data include among other things:

- The average CPU usage over the time window (field *CPU rate*).

- The maximum CPU usage over the time window (field *maximum CPU rate*)
- The memory usage over the time window (field *canonical memory usage*)

Note that in this table CPU usage is not normalized between 0 and 1. A reported CPU usage of 2 means that on average the task has used 2 CPU cores during the 5-minute time window.

5 Work on the dataset

We are going to use Spark to analyze the data included in this dataset.

5.1 Working with Spark

We refer you to the previous lab for a description of how to use Spark. The documentation for the previous lab is available here: `https://tropars.github.io/teaching/#data-management-in-large-scale-distributed-systems`.

You are allowed to work with Spark in Python or in Scala.

5.2 Analyses to be conducted

You are free to decide which analyses you want to conduct. You should select analyses so as to show the different capabilities of Spark.

To give you some ideas, we provide below examples of questions that one might want to answer through an analysis of the data:

- What is the distribution of the machines according to their CPU capacity?
- What is the percentage of computational power lost due to maintenance (a machine went offline and reconnected later)?
- What is the distribution of the number jobs/tasks per scheduling class?
- What is the percentage of jobs/tasks that got killed or evicted depending on the scheduling class?
- Do tasks with low priority have a higher probability of being evicted?
- In general, do tasks from the same job run on the same machine?
- Are there tasks that consume significantly less resources than what they requested?
- How often does it happen that the resources of a machine are over-committed[2]?

It would be good to propose analyses that can illustrate the use of some *advanced* transformations on RDDs such as: `join()`, `groupBy()`, `reduceByKey()`, `sortBy()`, `union()`, etc.

[2]Over-committing means that if all tasks running on the machine try using all the resources they asked for, the amount of resources required is more than what is available on the machine, as explained in the Google Documentation.

5.3 Comments

A few additional comments:

- We are allowed to take inspiration from the code provided in the previous lab about the way to parse the CSV files.
- Some IDs used in the dataset cannot be converted to Integers. They have to be converted to `Long` Integers instead.
- For the tables that are split into several files, we strongly suggest you to start by working on one single file. Then multiple/all files can be used to run the final analyses and/or to evaluate performance.

5.4 Extending the work

We provide below some suggestions about directions in which the work on this lab could be extended. It is up to you to decide whether you want to extend your work in one/some of these directions. The list is not exhaustive. You may propose other directions to extend the study.

5.4.1 Studying performance

Studying the performance of Spark for running the analyses is a direction in which this work can be extended. The performance topic can be studied from different angles. Here are a few suggestions:

Study at the application level: we have seen different mechanisms at the application level that can have an impact of the performance of Spark applications (lazy evaluation, caching RDDs, order of transformations, partitioning scheme). We may run evaluations to illustrate these points.

Study at the system level: Many configuration parameters may impact the performance of Spark applications (for a complete list, see `https://spark.apache.org/docs/latest/configuration.html#available-properties`). The number of cores assigned to each executor is a good example of parameter that can impact the performance. We may want to run experiments that evaluate the impact of this parameter on performance (and relate it to the number of cores available on the machine on which you run the tests). You may also consider evaluating the impact of other parameters on the performance.

Comparison with other techniques: The study we are running with Spark uses the basic RDD interface. Spark SQL (`https://spark.apache.org/docs/latest/sql-programming-guide.html`) is a Spark module for structured data processing. Spark SQL introduces the concept of `DataFrame` to manipulate structured data. One may want to compare the performance of algorithms using RDDs and DataFrames. Another question that we may ask ourselves is how does the performance of Spark compares to the performance of the Python data analysis library `Pandas` (`https://pandas.pydata.org/`).

5.4.2 Studying other datasets

A dataset, to some extend similar to the one provided by Google, has been released by Microsoft Azure recently. It is similar in the sense that it also provides information about the workload in a data center. Note however that it does not include the same kinds of information and that data are not organized in the same way. Still, you may want to complement your work by running some analysis on this dataset. The dataset is described here: `https://github.com/Azure/AzurePublicDataset`. Note that this dataset is even larger than the one from Google.