Analyzing Data with Spark

Master M2 – Université Grenoble Alpes

2025

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. It is also an opportunity to better understand how computing resources are used in a Cloud environment.

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
- The deadline to submit you work is **January 15, 2026**. See submission instructions below.
- If you have any question, please feel free to send emails to Vania Marangozova (vania.marangozova@imag.fr) and Ifechukwu Ejiofor (ifechukwu.ejiofor@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 Identifying your team

Please go to , choose a team and fill in the coordinates of the participants. The table also gives you indications on the data subset to use, more information further down in the subject.

4 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)
  - If applicable, a description of the way you chose to extend your work and the corresponding results.
- The code corresponding to the results presented in your report. Your code should be properly commented.
- **DO NOT** include the data in your submission.

  Please send the archive to both Vania Marangozova (vania.marangozova@imag.fr) and Ifechukwu Ejiofor (ifechukwu.ejiofor@univ-grenoble-alpes.fr) from your official university address and putting as subject `[M2 MOSIG][LSDM] Spark lab submission`.

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

[1]Other formats will be rejected.

5 The dataset

5.1 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 5.3.

5.2 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:

- 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.

    For your tests, use the parts' numbers indicated in the cloud file.
    I.e. if it is written that you need to use `From - part No 10` and `To - part No 14`, you will need to copy `part-00010-of-00500.csv.gz`,
    `part-00011-of-00500.csv.gz`,
    `part-00012-of-00500.csv.gz`,
    `part-00013-of-00500.csv.gz` and
    `part-00014-of-00500.csv.gz`.

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.

5.3 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.

5.3.1 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.

5.3.2 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.

6 Work on the dataset

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

6.1 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://vmarangozova.github.io/teaching.html#LSDM`.

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

6.2 Analyses to be conducted

We give below a list of questions that we would like you to answer through an analysis of the data. For each question specify whether you have run the code on your data sample or on the full dataset.

In addition to these questions, we would like you to propose at least 2 original questions and to answer them. The main criteria that will be used to grade these additional questions are their originality and relevance (and of course the way you answer them).

1. What is the distribution of the machines according to their CPU capacity?
   Can you explain (motivate) it?
2. What is the percentage of computational power lost due to maintenance (a machine went offline and reconnected later)?

   [4pt]The computational power is proportional to both the CPU capacity and the unavailability period of machines.
3. Is there a class of machines, according to their CPU, that stands out with a higher maintenance rate, as compared to other classes ?
4. What is the distribution of the number of jobs/tasks per scheduling class? Comment on the results.
5. Would you qualify the percentage of jobs/tasks that got killed or evicted as *important*?
6. Do tasks with a low scheduling class have a higher probability of being evicted?
7. In general, do tasks from the same job run on the same machine? Comment on the observed *locality* strategy and its *pros* and *cons*.
8. Are the tasks that request the more resources the one that consume the more resources?
9. Can we observe correlations between peaks of high resource consumption on some machines and task eviction events?
10. How often does it happen that the resources of a machine are over-committed[2]?
11. Your original question 1. Motivate the originality of the question.
12. Your original question 2. Motivate the originality of the question.

Note that one purpose of these analyses is to illustrate the different functionalities of Spark. Hence, proposing a set of analyses that all require the same sequence of transformation on the data, is of little interest for this lab.

[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.

6.3 Comments

A few additional comments:

- You can choose to work with RDDs or Spark Dataframe (`https://spark.apache.org/docs/latest/sql-programming-guide.html`)
- You are allowed to take inspiration from the code provided in the previous lab about the way to parse the CSV files for working with RDDs.
- Some IDs used in the dataset cannot be converted to Integers. They have to be converted to `Long` Integers instead.

6.4 Extending the work

For this project, if you correctly complete the analyses described above, you can expect a *good* grade but not an excellent grade.

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 (check with the teachers).

6.4.1 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, such as:

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). You 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. You 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.

6.4.2 Comparison of Different Solutions

One way to extend your work would be to compare different technical solutions to process the data. The comparison could be made from different points of view (performance, ease of use, etc.). Here are some suggestions:

- If you used Spark RDDs for your analysis, you can run a comparison with Spark Dataframe (and *vice versa*).
- You can compare the use of Spark to the use a non-parallel Python data analysis library `Pandas` (`https://pandas.pydata.org/`), or even to its parallel extension (`https://dask.org/`).
- You can compare the use of Spark to the use of another distributed data processing framework, such as Apache Flink (`https://flink.apache.org/`).

6.4.3 Implementing a stream processing application

Following the lecture on stream processing, the next lab will be about building stream processing applications using Kafka and Spark.

Reusing what you learned during this lecture and lab, you could extend your work by implementing a stream processing system that would process in real time data coming from a Google cluster.

Of course, in the dataset that we consider, the data are not generated in real time since the dataset is from 2011. However, all data tables include timestamps. Hence, it should be simple to simulate a real time system by writing a small *client application*, that would replay the events included in a data table according to their timestamp[3].

You are free to choose the data table you want to analyze, and the kind of stream analysis you want to apply. An example of stream analysis would be to compute the CPU usage over the whole cluster using XX-minute time windows.

6.4.4 Working in the Cloud

Deploying in the Cloud: Most Cloud providers offer credits to new users that would like to discover the Cloud. You can envision to take advantage of such offers to deploy your Spark application in a real distributed environment and conduct some performance evaluation in this context.

The following webpage provides some information on how to use Google Cloud Platform (GCP): `https://roparst.gricad-pages.univ-grenoble-alpes.fr/cloud-tutorials/`. On this webpage, you can find:

- General information about the utilization of the credits on GCP (warning: these information might be partially outdated).
- Information about how to use GCP.
- Examples of code (for instance to automatically create VMs using Terraform)

Of course, you are not limited to GCP, and you can go with another Cloud provider if you prefer doing so.

[3]You may want to adapt the timestamps (for instance, transforming minutes into seconds) to be able to simply test your application.

Studying the new Google Dataset: The 2019 dataset discussed above is only accessible on GCP (see `https://github.com/google/cluster-data/blob/master/ClusterData2019.md`). You may want to try running some analyses on this new dataset directly on the Cloud.

6.4.5 Studying Other Datasets

Some datasets, to some extend similar to the ones provided by Google, have been made available by other Cloud/Big Data companies. Here are web pages about these datasets:

- Alibaba datasets: `https://github.com/alibaba/clusterdata`
- Azure datasets: `https://github.com/Azure/AzurePublicDataset`

These web pages describe the datasets and how to access them, as well as, the existing scientific publications that have been made using these data.

These datasets are not necessarily organized in the exact same way as the Google data, and may not include the same information. Still, you may want to complement your study by running some analyses on one of these datasets. You can compare the datasets in terms of size and informations, with the goal of comparing Google and Alibaba infrastructures/applications/usage.

Another possible direction to go is to review one of the published papers.