Covid Data Analysis

Middleware Technologies for Distributed Systems

Chiara Marzano
Massimiliano Nigro
Daniele Paletti

29/09/2021

Contents

1 Overview

This project implements a program that analyzes datasets to perform data analysis on the COVID-19 situation. Using a dataset of daily reported cases for each country, it computes useful statistics. The dataset we decided to use is that provided by the European Centre for Disease Prevention and Control [1].

2 Technology choice

Following the specifications, we decided to develop this project using **Apache Spark**. Spark is highly specialized for big data analysis and allows **SQL manipulation** of datasets providing implicit parallelism. Considering the type of information to be extracted and the dimensions of the dataset (≈ 62000 rows), a big data approach suited our needs. We decided against using MPI as it is better suited for simulations and compute-intensive data, while our application is **data-intensive** as it features simple operations over a large set of data.

3 Structure

1. Setup of Spark
2. Input parsing: Input file is parsed
3. Input processing: Original dataset is cleaned, useless columns are dropped and useful columns are renamed. The new dataset contains three columns: date, country and number of infected.
4. Compute moving average and percentage increase
5. Compute top ten and store it
6. Print results

4 Design choices

4.1 Spark SQL

Since our project features a columnar dataset, a natural interpretation of it is as a SQL table. Through view manipulation and queries, we are able to compute the required indices with ease.

4.2 Redistribution of infected over missing daily reports

As some countries featured partial or cumulative data in some periods of time, we decided to update the view used by the project and spreading the infected over the days with missing reports. Specifically, we divided the number of

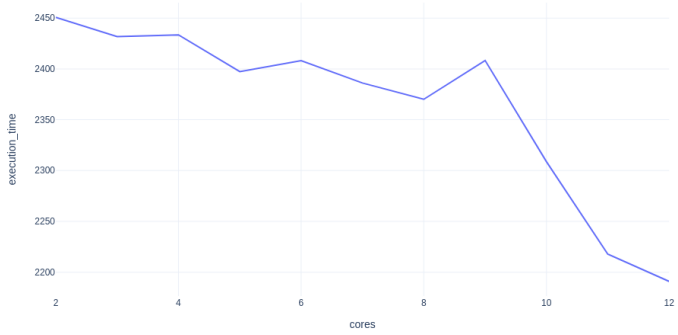

Figure 1: cores: Average execution time.

infected of the first report after at least one missing date for the number of days without reports.

5 Performance

5.1 Number of cores

Having 12 total cores available, we tested our application increasing the number of cores at each execution. From the data gathered, we observed that a general improvement in performance takes place. Some peaks are also visible: these can likely be attributed to the added overhead of increasing the number of working devices.

5.2 Number of countries

We could observe that increasing the number of countries seem to increase the execution time. We decided not to measure the performance on a varying number of rows as it is closely correlated to that of the number of countries as adding countries adds rows with an almost constant multiplication factor.

References

[1] *Download historical data (to 14 December 2020) on the daily number of new reported COVID-19 cases and deaths worldwide*. en. Dec. 2020. URL: `https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-19-cases-worldwide` (visited on 09/25/2021).

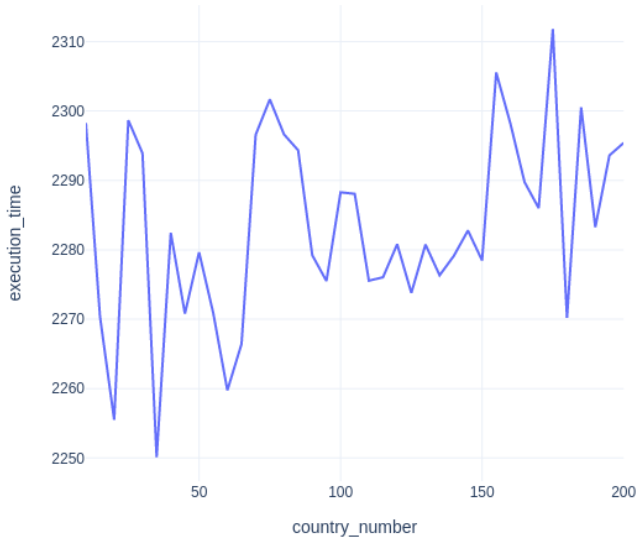

Figure 2: country: Average execution time.