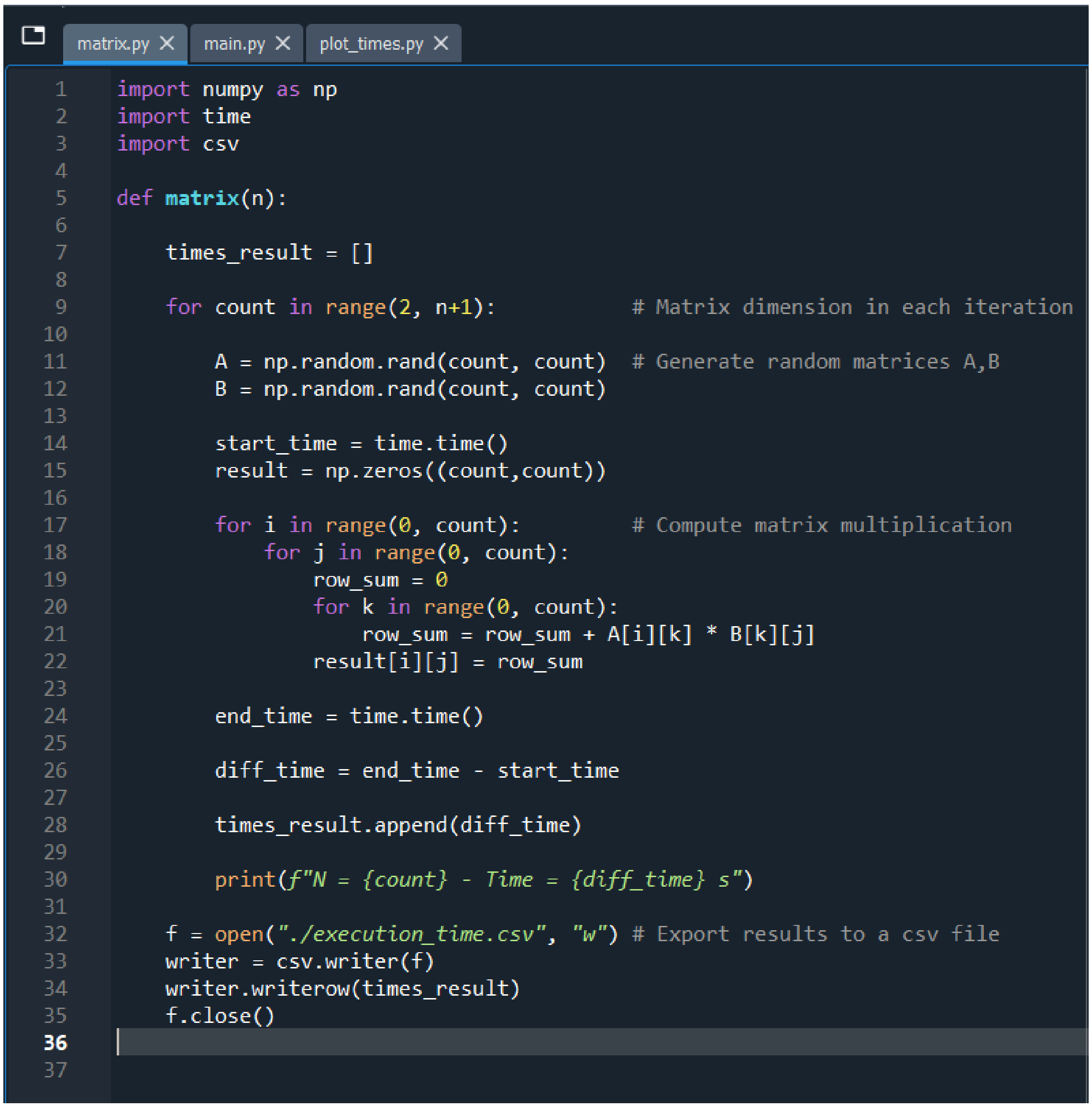
Sergio García López
ID: 122118616

CS3511 Lab 1 Report - Virtual machines and containers

Application

The application that has been developed to test the execution time in each environment has been a matrix multiplication. It is programmed in Python, in a file called **matrix.py**, which contains a function called **matrix**.

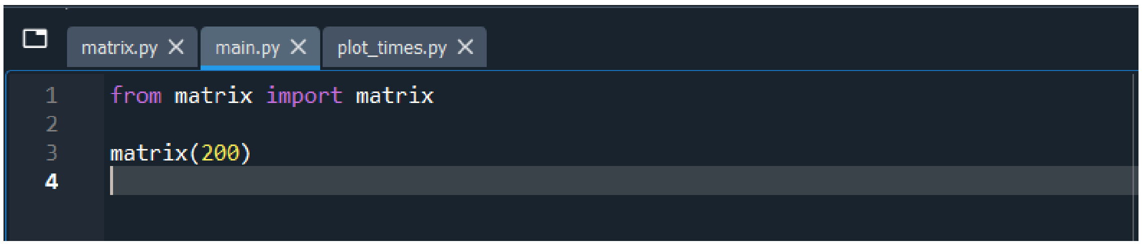
The function receives the maximum dimension of the matrices and in each iteration, it computes the matrix multiplication and records the execution time. The program uses certain libraries such as **numpy**, to generate the matrices, and **csv**, to export the results in a csv file:



```python
import numpy as np
import time
import csv

def matrix(n):

    times_result = []

    for count in range(2, n+1):           # Matrix dimension in each iteration

        A = np.random.rand(count, count)  # Generate random matrices A,B
        B = np.random.rand(count, count)

        start_time = time.time()
        result = np.zeros((count,count))

        for i in range(0, count):         # Compute matrix multiplication
            for j in range(0, count):
                row_sum = 0
                for k in range(0, count):
                    row_sum = row_sum + A[i][k] * B[k][j]
                result[i][j] = row_sum

        end_time = time.time()

        diff_time = end_time - start_time

        times_result.append(diff_time)

        print(f"N = {count} - Time = {diff_time} s")

    f = open("./execution_time.csv", "w") # Export results to a csv file
    writer = csv.writer(f)
    writer.writerow(times_result)
    f.close()

```

Figure 1: matrix.py

Also, there has been created a **main.py** file to test the execution time with different values for the maximum dimension of the matrices:



```python
from matrix import matrix

matrix(200)
```

Figure 2: main.py

The application has been run in a Windows 10 laptop with an Intel Core i7 1.80 GHz with 4 cores and 16 GB of RAM. The results obtained in the host machine (without virtualization) are the following:

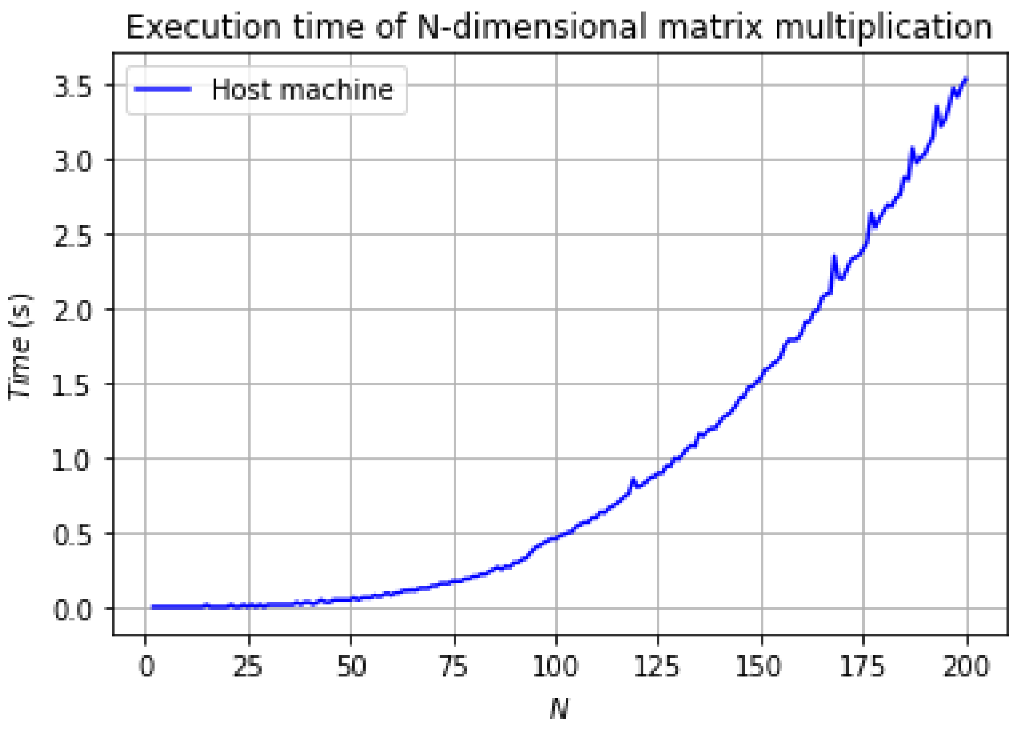


Figure 3: Execution time in host machine

Virtual machines and containers

A virtual machine is a computer that runs on software, instead of running directly on hardware. Each virtual machine runs its own operating system and truly "believes" that is directly accessing hardware resources, but its system calls are processed by another software, which is the hypervisor. Hypervisor type 1 acts like a lightweight operating system and runs on hardware, whereas hypervisor type 2 runs on a host OS as an additional software layer.

A container is an encapsulated environment that runs applications. It shares the OS kernel and it only contains the application, the software libraries and environment variables that are necessary. It takes advantage of some Linux features, like namespaces and cgroups, to isolate processes and restrict resource utilization. They are a different approach to virtual machines, being lighter and faster to deploy.

The application has been tested in an Ubuntu 20.04 LTS guest OS using VirtualBox as hypervisor. The results obtained are the following:

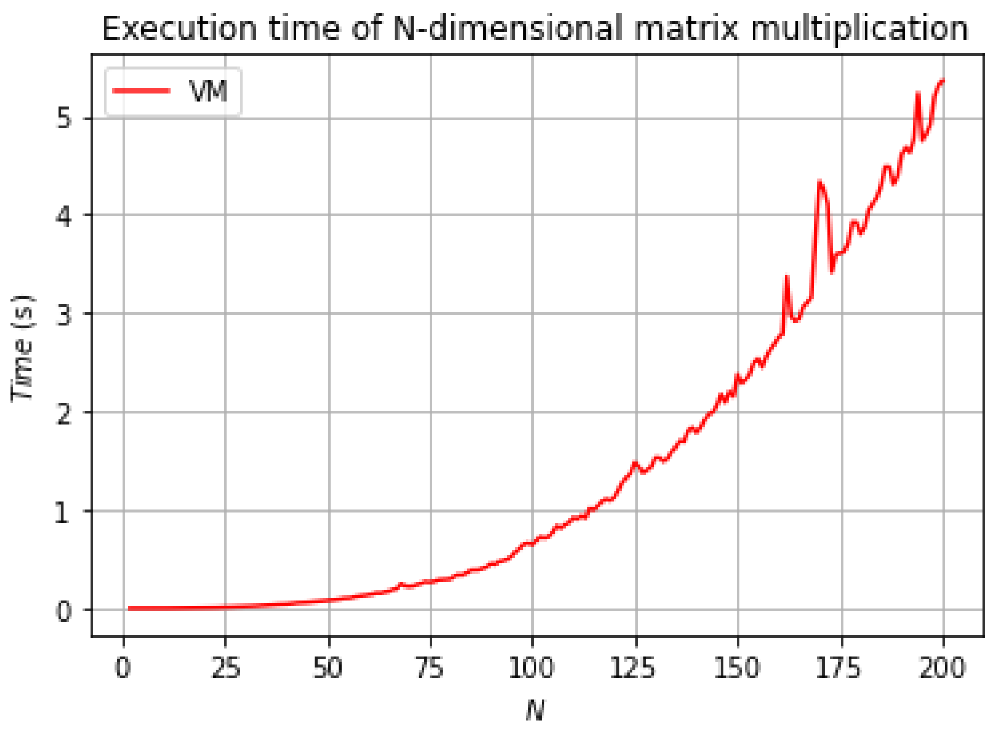


Figure 4: Execution time in virtual machine

Also, the application has been tested in a Python container making use of Docker Desktop for Windows with the following results:

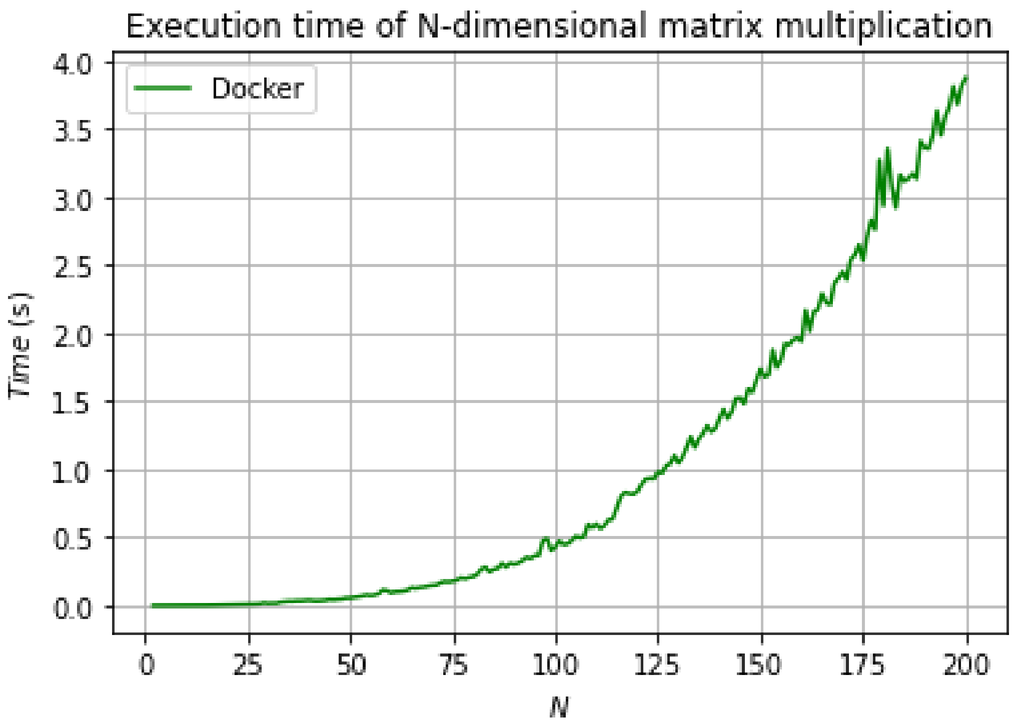


Figure 5: Execution time in Docker container

Execution time comparison

As mentioned before, the application has been tested in three different environments: in the host machine without virtualization, in an Ubuntu virtual machine and in a Docker container. If we plot the three curves together, we can notice several things:

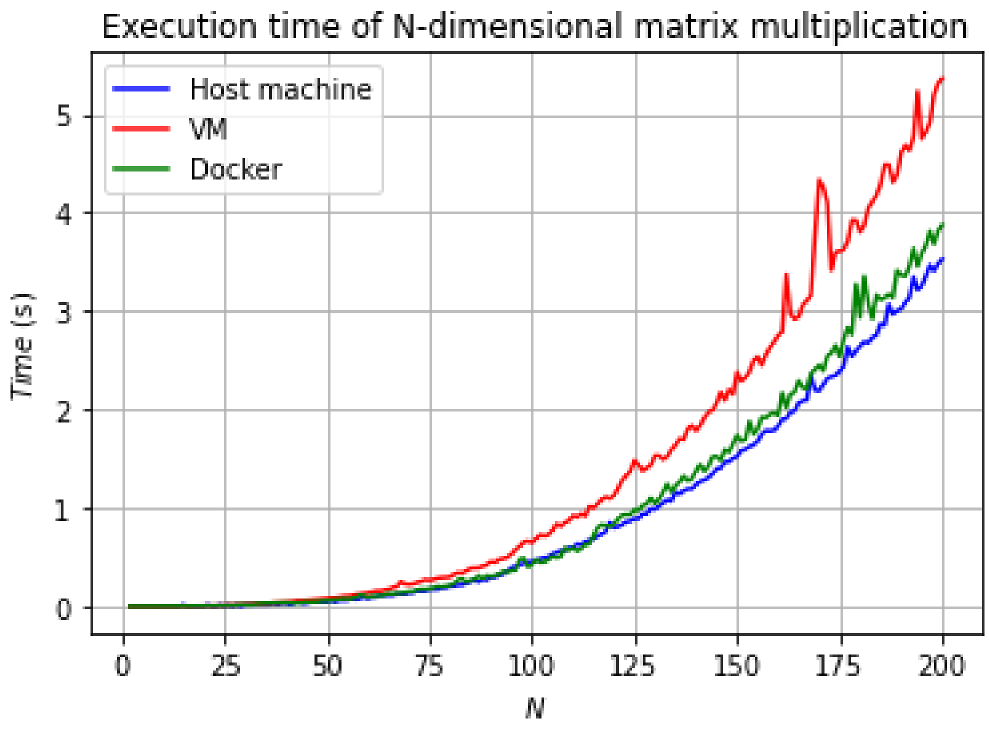


Figure 6: Execution time comparison

First, we can observe that the execution time in all cases grows exponentially due to the computational cost of a simple matrix multiplication algorithm. As we expected, the execution time is lower when the application is running without virtualization in the host machine, whereas it is significantly bigger when it is running in the virtual machine, due to the additional software layers. Here we can see that containers are much faster than virtual machines, with a performance close to the host machine, even though the container is running on top of the Docker container engine.

Conclusions

With this lab we have put in practice what we though theoretically about performance in virtual machines and containers. It shows us that virtual machines can be useful for software development, but they give us less performance, making them less suitable for running applications where time is critical. On the other hand, we see the advantages of using containers in terms of performance, with lower results but close to those of the host machine. Furthermore, with containers we can take advantage of isolation, resource utilization and portability, because we can test and run our applications in different hosts using the same container image.

I did not encounter many difficulties in developing the matrix multiplication algorithm. Python is a very powerful programming language and when it is used with numpy it makes matrices handling very easy. At first, I tried to use the matrix multiplication function that numpy provides, but it was so efficient that I though that it would be more interesting to to use a slower algorithm to test the execution time better.

Personally, I already had a lot of experience using virtual machines in VirtualBox and VMware in my home university, so I have not encountered any problem in installing the Ubuntu virtual machine. Also, I had a little experience with Docker in Linux virtual machines, but this is the first time that I have installed and tried Docker Desktop in Windows. However, I think it has helped me a lot and I have improved my skills in deploying Docker containers, using the Docker commands to run the container, copy the application files inside the container, spawning a bash shell inside it, running the application, and copying the result files back to the Windows machine.

References

Link to application code in GitHub: https://github.com/SergiDelta/CS3204_Lab_1