UNIVERSITÀ
DEGLI STUDI
FIRENZE

**Faculty of Engineering**

Master course in
Artificial Intelligence

# PERFORMANCE ANALYSIS REPORT FOR IMAGE READING

## *Parallel Programming for Machine Learning*

*Mid-term assignment*

**Candidate**
Loric TONGO GUIMTSA

**Professor**
Marco BERTINI
*Associate Professor*

Academic Year: 2023/2024

# Contents

# 1) Introduction

This report presents a comparative performance analysis between a sequential image reader program and a parallel image reader program implemented using threads.

# 2) Context

The subject concerns the development of a multithreaded JPEG image reader that can read a certain number of images from a given directory and store the uncompressed images or the compressed stream in an appropriate data structure. The goal is to evaluate the performance of both sequential and parallel approaches.

# 3) Methodology

We implemented two programs to read the images. The first is a sequential program that reads the images one by one, while the second is a parallel program that utilizes threads to read the images simultaneously. The performance of each program was measured in terms of execution time and processing speed.

# 4) Sequential vs Parallel

## *a) Sequential Program*

The sequential program reads each image in the specified directory and processes them one after the other. The total execution time and reading speed are measured. The program is simple and easy to understand, but it does not leverage multiple CPU cores.

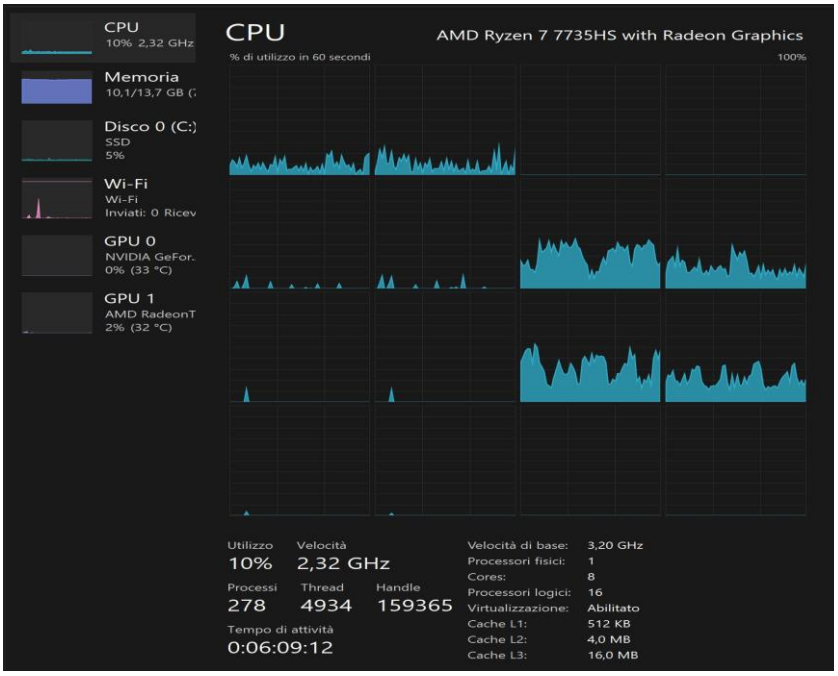


*Figure 1: CPU utilization*

As evidenced by Figure 1, which depicts CPU utilization during the execution of the sequential program, it's apparent that the sequential program does not utilize all CPU cores. The figure clearly illustrates that only about 10% of the CPU power is utilized during the execution of the sequential program.

### *b) Parallel Program*

The parallel program uses threads to read the images in parallel. It divides the images into multiple tasks, with each task being processed by a separate thread. The number of threads equals the number of logical CPU cores. The total execution time and reading speed are also measured for this approach.

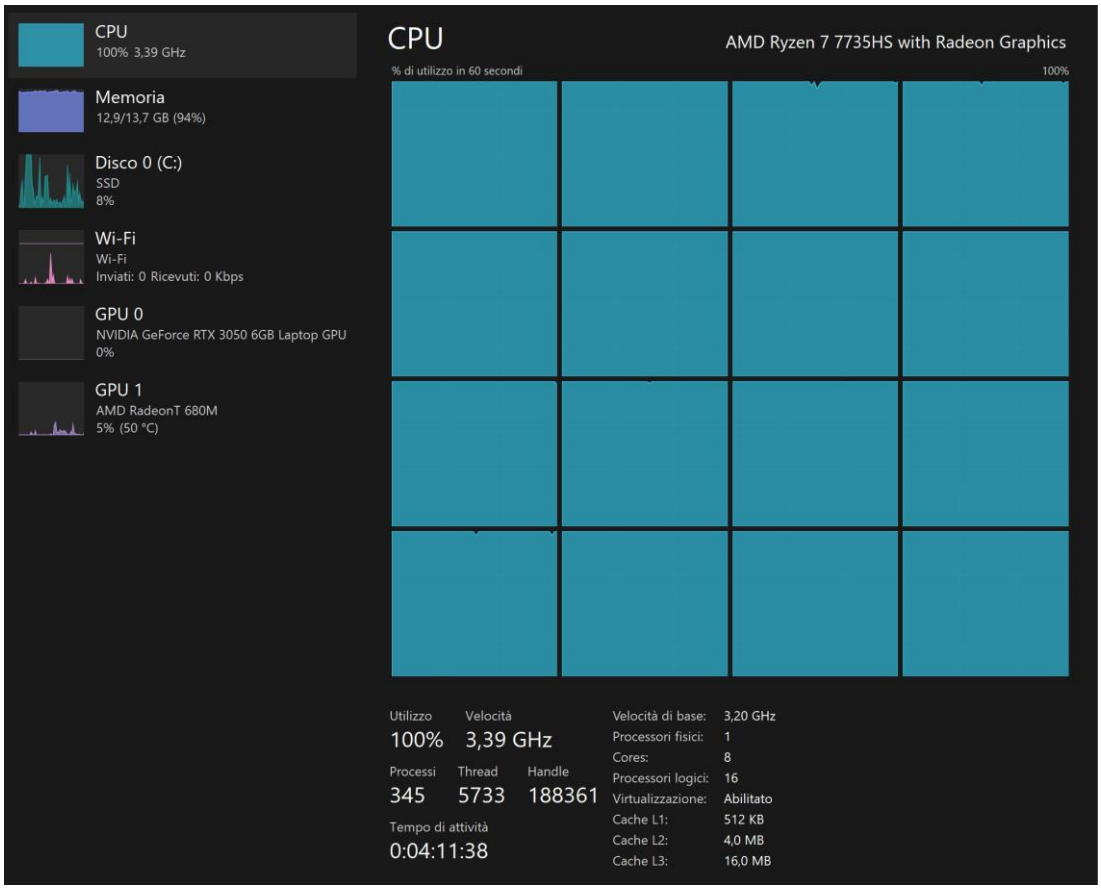


*Figure 2: CPU utilization*

As illustrated in Figure 2, which depicts CPU utilization during the execution of the parallel program, it's evident that the parallel program utilizes all CPU cores. The figure clearly shows that the parallel program efficiently utilizes the entire processing power of the CPU.

## 5) Results

### *a) Sequential Program*

- Number of images: 200.000
- Execution time: 728 seconds*
- Processing speed: 275 images/second*

### *b) Parallel Program*

- Number of images: 200.000
- Execution time: 113 seconds*
- Processing speed: 1763 images/second*

** These values are approximate*

## 6) Performance Analysis

Comparing the two approaches, we observe that the parallel program offers a significant improvement in performance compared to the sequential version. This is due to the efficient use of CPU resources, allowing multiple images to be processed in parallel. However, we also note that the parallel program may incur additional costs in terms of thread management and synchronization, which can lead to an increase in execution time in some cases. It is important to find a balance between the number of threads and task size to optimize performance.

## 7) Conclusion

Overall, transitioning to a parallel program offers a significant performance improvement for image reading. However, it is essential to consider aspects of thread management and synchronization to achieve the best results. Adjustments may be necessary depending on hardware and the number of images to be processed.

## 8) Recommendations

Based on this analysis, we recommend using the parallel approach for image reading applications requiring fast processing of large datasets. However, thorough testing on different hardware configurations and data sizes is recommended to optimize performance.