UNIVERSITÀ
DEGLI STUDI
FIRENZE

Faculty of Engineering

Master course in
Artificial Intelligence

PERFORMANCE ANALYSIS REPORT FOR PASSWORD DECRYPTION

Parallel Programming for Machine Learning

Final-term assignment

Candidate
Loric TONGO GUIMTSA

Professor
Marco BERTINI

Academic Year: 2023/2024

Associate Professor

Contents

1. Introduction

This report presents a performance analysis of sequential and parallel programs for decrypting passwords encrypted using the DES (Data Encryption Standard) algorithm. The sequential program decrypts passwords one by one, while the parallel program utilizes multiprocessing to decrypt passwords in parallel. The goal is to evaluate the speedup achieved by parallelizing the decryption process.

2. Context

The task involves decrypting passwords encrypted using the DES algorithm. Each password is 8 characters long and belongs to the character set [a-zA-Z0-9./]. The sequential program employs a simple approach to decrypt each password sequentially. In contrast, the parallel program utilizes multiprocessing to distribute the decryption tasks among multiple processes, aiming to accelerate the overall decryption process.

3. Methodology

Two programs were implemented for password decryption: a sequential program and a parallel program using multiprocessing. Both programs generate random passwords, encrypt them using DES, and then decrypt them. The performance of each program was measured in terms of execution time for encryption and decryption.

4. Sequential vs Parallel

4.1. Sequential Program

The sequential program generates random passwords, encrypts them using DES, and then decrypts them sequentially. It utilizes the “Crypto.Cipher.DES” module for encryption and decryption operations.

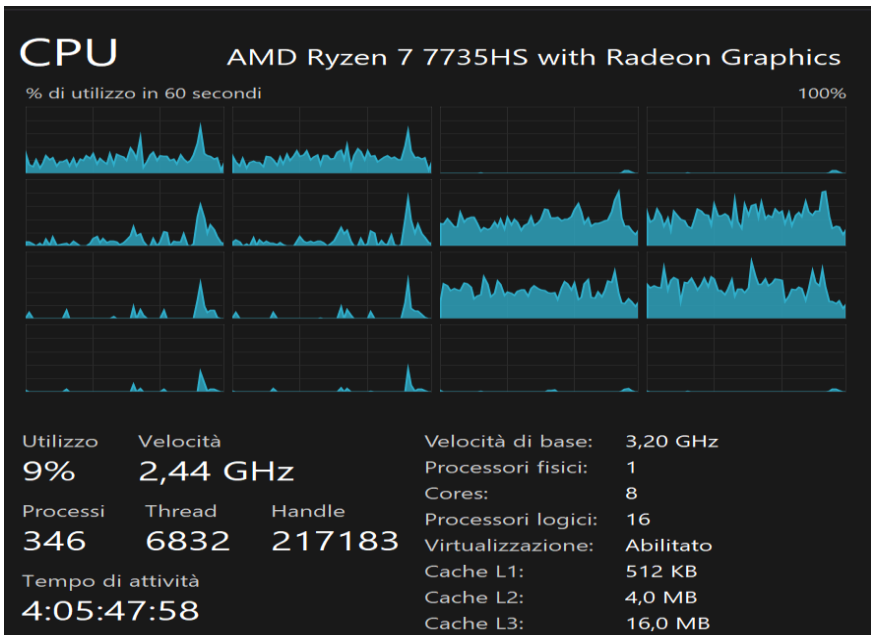

Figure 1: CPU utilization

As illustrated in Figure 1, which depicts CPU utilization during the execution of the sequential program, it's evident that the sequential program does not fully utilize all CPU cores. The figure clearly shows that only about 9% of the CPU power is utilized by the sequential program.

4.2. Parallel Program

The parallel program parallelizes the decryption process using multiprocessing. It divides the decryption tasks among multiple worker processes, each responsible for decrypting a subset of passwords. The “multiprocessing.Pool” class is used to manage the worker processes.

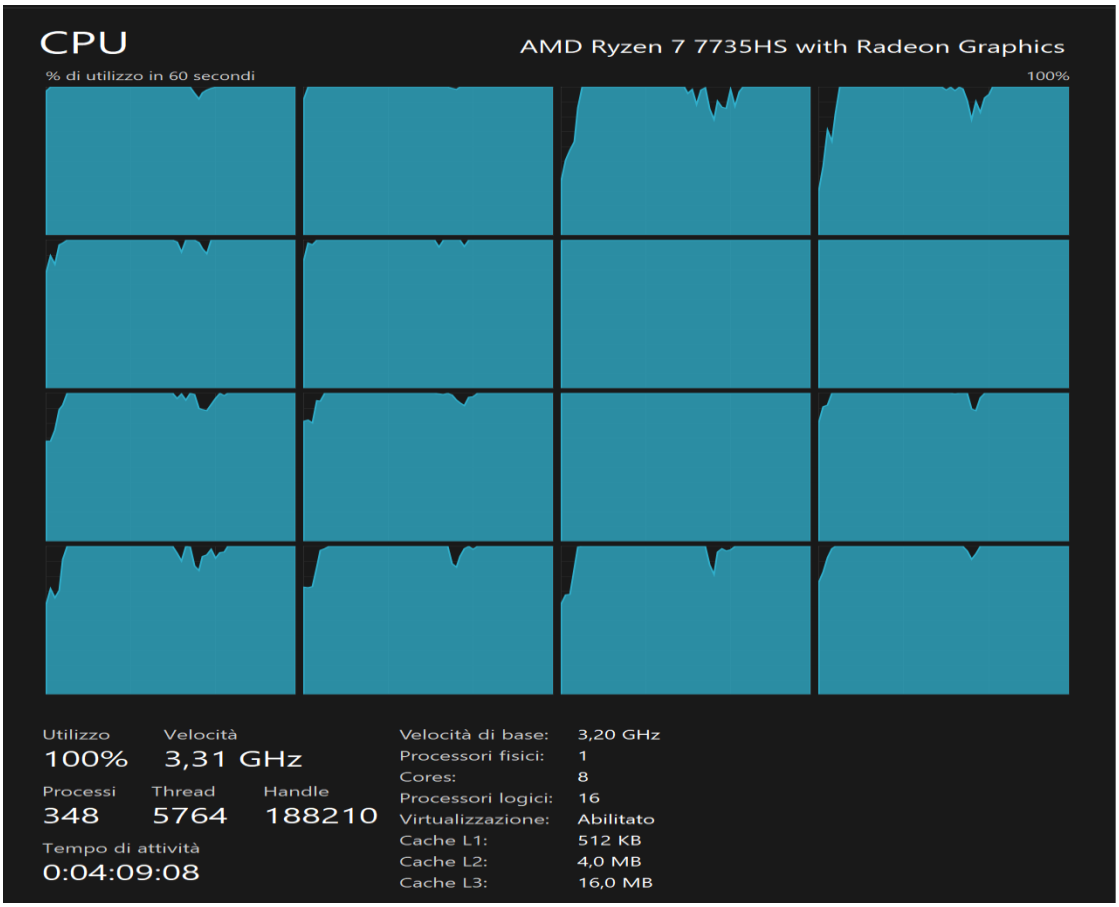

Figure 2: CPU utilization

As evidenced 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

5.1. Sequential Program

5.1.1. Execution time for:

- One (1) million passwords: 8 seconds*
- Ten (10) millions of passwords: 80 seconds*
- One hundred (100) millions of passwords: 953 seconds*

5.2.Parallel Program

5.2.1. Execution time for:

- One (1) million passwords: 1 second*
- Ten (10) million passwords: 11 seconds*
- One hundred (100) million passwords: 141 seconds*

** These values are approximate*

6. Performance Analysis

To compute the speedup, we'll use this formulation:
Speedup = Sequential Execution Time / Parallel Execution Time

- For one (1) million passwords: speedup = 8 / 1 = 8
- For ten (10) million passwords: speedup = 80 / 11 ≈ 7.27
- For one hundred (100) million passwords: speedup = 953 / 141 ≈ 6.75

7. Conclusion

The parallel program demonstrates significant speedup compared to the sequential program for all tested scenarios. As the number of passwords increases, the speedup slightly decreases, but parallelization still provides substantial performance improvement. Overall, parallelization is an effective approach for accelerating the decryption process, especially for large datasets.

8. Recommendations

Based on the performance analysis, it is recommended to use the parallel program for decrypting passwords, especially when dealing with large datasets. Further optimizations and fine-tuning of parallelization parameters may lead to even better performance results. Additionally, considering alternative algorithms or techniques for password decryption could also be explored to further enhance performance.