Computação em Larga Escala

Practical Assignment 1

João Monteiro (102690), João Sousa (103415), João Gaspar (107708)

Departamento de Eletrónica, Telecomunicações e Informática

Universidade de Aveiro

Contents

1 Introduction **1**
1.1 Problem 1: Weather Stations 1
1.2 Problem 2: Word Count 1
1.3 Objective 1

2 Performance Analysis **1**
2.1 Measured Execution Times 1
2.2 Speedup Calculations 2
2.3 Additional Details 3

3 Comparison of Results **4**
3.1 Word Count Analysis 4
3.2 Weather Stations Analysis 5
3.3 Discussion 5

4 Conclusion **6**

1 Introduction

This project aims to analyze and compare the performance of two programming approaches: single-threaded and multi-threaded, applied to two specific problems: "Weather Stations" and "Word Count."

1.1 Problem 1: Weather Stations

The "Weather Stations" problem involves reading meteorological data from various stations and calculating statistics such as the average, minimum, and maximum temperature for each station. The single-threaded version processes the data sequentially, while the multi-threaded version distributes the workload across multiple threads to accelerate processing. The goal is to determine whether parallelization can significantly reduce execution time and improve data processing efficiency.

1.2 Problem 2: Word Count

The "Word Count" problem consists of counting the number of words, lines, and characters in a set of text files. The single-threaded version performs the count sequentially, while the multi-threaded version distributes the files among multiple threads to perform the count in parallel. The objective is to assess the performance gains achieved through parallelization and identify potential limitations.

1.3 Objective

The main objective of this project is to migrate single-threaded applications to multi-threaded versions and analyze the resulting performance improvements. Through comparative testing, it will be possible to validate the results and discuss the possible causes of observed gains or limitations. Additionally, the project aims to provide a practical understanding of the benefits and challenges of parallel programming.

2 Performance Analysis

To evaluate the benefits of parallelization, we conducted comparative tests on both the Weather Stations and Word Count applications. For the Weather Stations problem, we processed an input file containing 10 million samples, while for the Word Count problem, approximately 641 text files (books) were used.

2.1 Measured Execution Times

The following execution times (in seconds) were recorded:

- **Word Count:**
  - Single-threaded: 1.06154 s
  - Multi-threaded (4 threads): 0.247982 s
  - Multi-threaded (8 threads): 0.193594 s
  - Multi-threaded (16 threads): 0.168629 s
  - Multi-threaded (32 threads): 0.184134 s
- **Weather Stations:**
  - Single-threaded: 1.1677 s
  - Multi-threaded (4 threads): 0.418642 s
  - Multi-threaded (8 threads): 0.297674 s
  - Multi-threaded (16 threads): 0.198808 s
  - Multi-threaded (32 threads): 0.189013 s

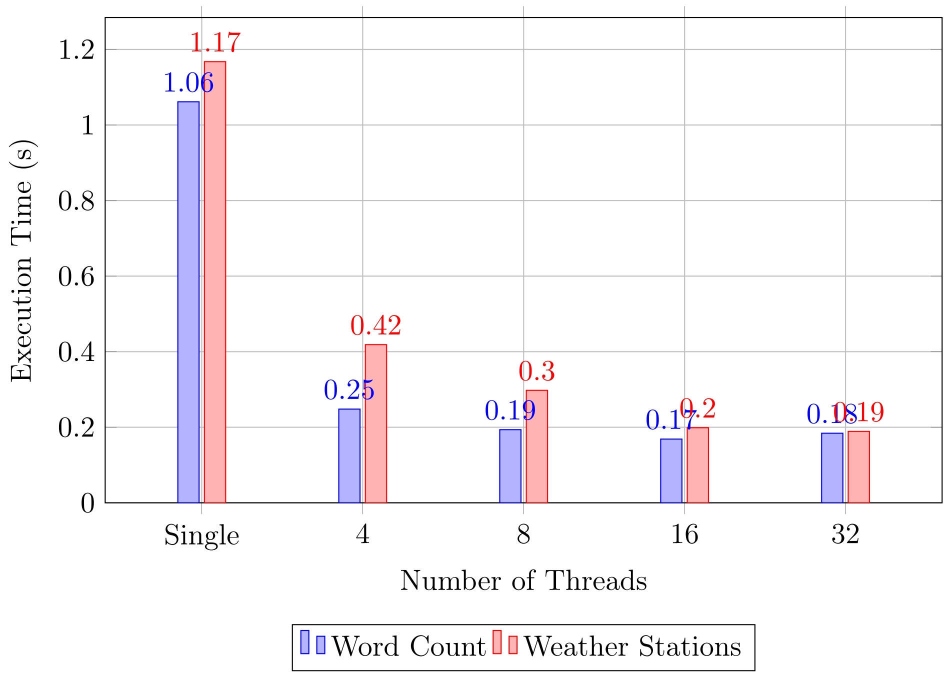

Figure 1: Execution Time vs. Number of Threads

2.2 Speedup Calculations

Speedup is computed as the ratio of the single-threaded execution time to the multi-threaded execution time. The results are as follows:

- **Word Count:**
  - 4 threads: $\frac{1.06154}{0.247982} \approx 4.28$
  - 8 threads: $\frac{1.06154}{0.193594} \approx 5.48$
  - 16 threads: $\frac{1.06154}{0.168629} \approx 6.30$
  - 32 threads: $\frac{1.06154}{0.184134} \approx 5.76$
- **Weather Stations:**
  - 4 threads: $\frac{1.1677}{0.418642} \approx 2.79$
  - 8 threads: $\frac{1.1677}{0.297674} \approx 3.92$
  - 16 threads: $\frac{1.1677}{0.198808} \approx 5.87$
  - 32 threads: $\frac{1.1677}{0.189013} \approx 6.17$

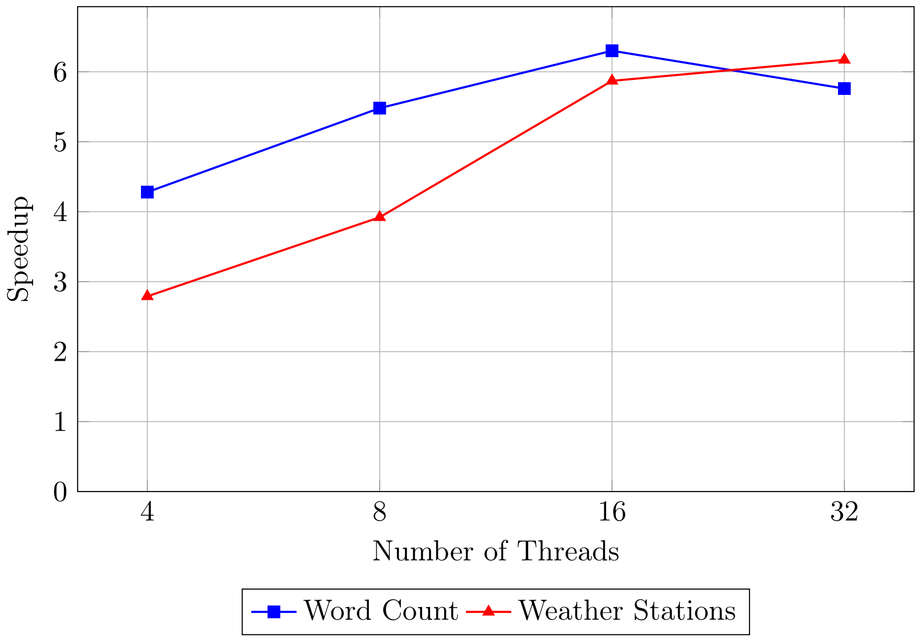

Figure 2: Speedup vs. Number of Threads

2.3 Additional Details

To provide a comprehensive understanding of the performance results, it is important to describe the test conditions and assumptions under which the experiments were conducted.

- **Input Data:**

  - For the Weather Stations problem, an input file containing **10 million samples** was used. Each sample represents a weather measurement in the format "StationName;Temperature".
  - For the Word Count problem, the experiments were carried out on approximately **641 text files** (books). Each file is expected to be encoded in UTF-8.

- **Thread Configurations:** Multi-threaded experiments were conducted using different numbers of threads (4, 8, 16, and 32) to observe the scaling behavior of each application. This allowed us to evaluate how effectively the workload is distributed and to identify the optimal thread count for each problem.

- **Hardware Environment:** The tests were conducted on a system equipped with a 13th Generation Intel Core i9-13980HX processor operating at 2.20 GHz and 32 GB of RAM. The processor features 24 cores (8 performance cores and 16 efficiency cores) and 32 threads.

- **Measurement Methodology:** Execution times were measured using high-resolution timers (e.g., `std :chrono :high_resolution_clock`) to capture the total runtime of each application under identical conditions.

3 Comparison of Results

The experimental results demonstrate a clear performance improvement when moving from the single-threaded to the multi-threaded implementations.

3.1 Word Count Analysis

For the Word Count problem, the multi-threaded version achieved the best performance using 16 threads, resulting in a speedup of approximately 6.30 times compared to the single-threaded version. However, when increasing to 32 threads, the execution time slightly increased (speedup of about 5.76x), likely due to increased overhead from thread management or hardware limitations (e.g., a limited number of physical cores).

Threads	Single-threaded (s)	Multi-threaded (s)	Speedup
Single	1.06154	-	-
4	1.06154	0.247982	4.28
8	1.06154	0.193594	5.48
16	1.06154	0.168629	6.30
32	1.06154	0.184134	5.76

Table 1: Performance Metrics for the Word Count Application

3.2 Weather Stations Analysis

In the Weather Stations application, performance improved as the number of threads increased, with a speedup of approximately 6.17 times when using 32 threads. This more consistent scaling might be attributed to differences in the workload distribution and synchronization requirements inherent in processing weather measurement data.

Threads	Single-threaded (s)	Multi-threaded (s)	Speedup
Single	1.1677	-	-
4	1.1677	0.418642	2.79
8	1.1677	0.297674	3.92
16	1.1677	0.198808	5.87
32	1.1677	0.189013	6.17

Table 2: Performance Metrics for the Weather Stations Application

3.3 Discussion

The observed performance gains indicate that parallelization significantly reduces the execution time for both problems. The Word Count application, however, shows diminishing returns beyond 16 threads, suggesting that overthreading may introduce overhead that counteracts the benefits of additional parallelism. In contrast, the Weather Stations application scales more consistently, possibly due to its processing model and the nature of the input data.

These results validate the effectiveness of multi-threading in improving performance, while also highlighting that the optimal number of threads can vary depending on the problem's characteristics and the available hardware resources.

4 Conclusion

The comparative analysis clearly demonstrates that the multi-threaded implementations outperform their single-threaded counterparts in both the Word Count and Weather Stations problems. While the Word Count application achieves peak performance at 16 threads, the Weather Stations problem benefits from scaling up to 32 threads. The study underscores the importance of tuning the degree of parallelism to balance the workload and minimize overhead. Future work could explore dynamic thread management and further optimizations to better leverage hardware capabilities.