UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO

Relatório - Trabalho 1
Computação de Alto Desempenho

Guilherme Lorete Schmidt - 13676857
João Victor Breches Alves - 13677142
Luana Hartmann Franco da Cruz - 13676350
Lucas Corlete Alves de Melo - 13676461

São Carlos
2025

1 Introdução

Este relatório busca detalhar o funcionamento e analisar o desempenho de três algoritmos distintos para solucionar o problema "Diga-me a frequência", visando encontrar aquele que melhor se adequa ao problema proposto. Cada algoritmo algoritmo desenvolvido toma uma abordagem distinta quanto à paralelização, indo desde uma implementação sequencial, até uma completamente paralelizada (como proposto no PCAM fornecido aos alunos).

2 Descrição dos Algoritmos

O primeiro algoritmo desenvolvido, representado pelo arquivo "`main_opt1`", traz uma implementação sequencial do problema, utilizando-se de ferramentas padrões da linguagem C e do compilador `gcc` para alcançar o maior desempenho possível. Ele servirá de base para avaliar o desempenho dos demais.

O segundo algoritmo, no arquivo "`main_opt2`", traz um paralelização inicial do problema, com granulação mais grossa. Ele gera uma *task* para processar cada linha lida do algoritmo de entrada, executando cada uma em paralelo dentro do limite de *threads* definido.

Por fim, o terceiro algoritmo, correspondente ao arquivo "`main_opt3`", realiza uma paralelização do problema de acordo com a descrição presente no PCAM fornecido pela disciplina. Isso consistiu em, além de criar uma *task* para processar cada linha lida, ele também preenche o vetor de frequências dos caracteres e realiza sua ordenação de forma paralela, tudo através de *tasks* criadas para executar no *pool* de *threads* originalmente definido.

3 Desempenho

As tabelas a seguir demonstra o tempo de execução dos algoritmos considerando diferentes cargas de trabalho n (em número de linhas da entrada) e diferentes números de *threads* p. Especificamente, trazemos o caso base, de $p = 1$, e um caso onde $p = 12$.

Os algoritmos foram executados em um processador *AMD Ryzen 5 5600 6-Core*. Os tempos foram determinados a partir da média de 30 execuções do mesmo algoritmo com a mesma entrada.

Nota-se que, mesmo quando $p = 1$, não há um custo adicional de paralelização muito alto envolvido no caso do algoritmo 2. No entanto, o algoritmo 3 demonstra possuir um alto custo, demandando quase o dobro do tempo de execução.

Já para $p = 12$, notamos que o algoritmo 2 torna-se significativamente mais rápido

Tabela 1 – Tempos de execução para $p = 1$, em segundos

Carga	Algoritmo 1	Algoritmo 2	Algoritmo 3
10	0.00006	0.00005	0.00008
100	0.00057	0.00042	0.00055
1000	0.00442	0.00347	0.00537
10000	0.03275	0.03361	0.05329
100000	0.32010	0.33160	0.52959

Tabela 2 – Tempos de execução para $p = 12$, em segundos

Carga	Algoritmo 1	Algoritmo 2	Algoritmo 3
10	0.00006	0.00022	0.00035
100	0.00057	0.00029	0.00109
1000	0.00442	0.00259	0.00967
10000	0.03275	0.02232	0.09183
100000	0.32010	0.18423	0.91320

que a versão sequencial. Simultaneamente, o algoritmo 3 torna-se consideravelmente mais lento.

Para melhor visualizar essas relações, expomos na figura 1 o *speedup* dos algoritmos 2 e 3 em função da versão sequencial dada pelo algoritmo 1. Consideramos uma carga de $n = 100\,000$.

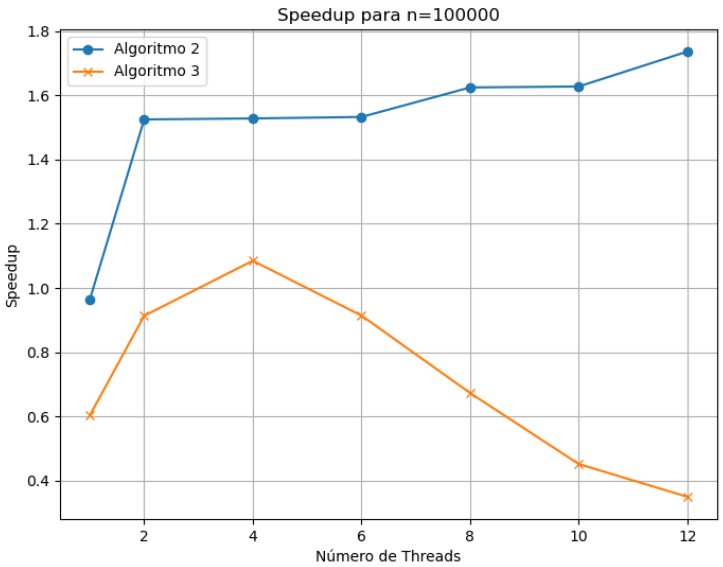

Figura 1 – Speedup dos algoritmos - AMD.

Note que o *speedup* do algoritmo 2 cresce rapidamente de início e se estabiliza a partir de $p = 2$, crescendo lentamente a partir de então. Já o algoritmo 3 tem um crescimento inicial até $p = 4$, mas logo decai chegando a um *speedup* menor que 1.

No entanto, ao executar os mesmos algoritmo em outro processador, um *Intel(R) Core(TM) Ultra 5 125U*, percebemos um comportamento diferente. A figura 2 mostra que o *speedup* do algoritmo 2 alcança seu pico para $p = 4$, decrescendo a partir de então. Já o algoritmo 3 sequer chega à marca de *speedup* 1.

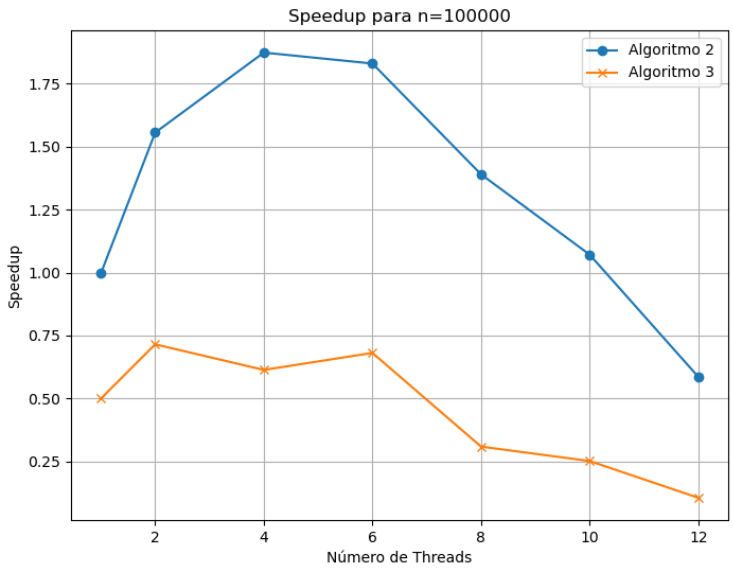

Figura 2 – Speedup dos algoritmos - Intel.

Esse conjunto de figuras nos mostra que, de forma geral, a paralelização adicional realizada no algoritmo 3 em relação ao 2 não proporciona um ganho significativo de desempenho dentro das arquiteturas analisadas, resultando majoritariamente em custos adicionais em sua execução.

Adicionalmente, vemos que fatores da arquitetura são extremamente relevantes no desempenho dos algoritmos, levando a *speedups* consideravelmente distintos a depender do processador empregado.

Ainda assim, dentre as arquiteturas testadas, vemos que o algoritmo 2 (`main_opt2`) mostra-se o mais adequado, considerando-se tempo de execução como o critério de desempenho adotado.