UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
MAC0219 - PROGRAMAÇÃO CONCORRENTE E PARALELA

PATRÍCIA RODRIGUES, NUSP: 11315590
SABRINA ARAÚJO DA SILVA, NUSP: 12566182
SAMANTHA MIYAHIRA, NUSP: 11797261

EP 1: Relatório

São Paulo
2023

1 EP1 - Simulação paralela de um gás ideal usando Pthreads e OpenMP

Foram executadas medições de tempo de execução para diferentes tamanhos de entrada (grid_size) e números de threads (num_threads) em cada uma das três versões do programa: sequencial, paralelizada com Pthreads e paralelizada com OpenMP.

A média e o intervalo de confiança das 10 medições (ou mais, caso haja grande variabilidade) foram utilizados na análise dos resultados.

Utilizamos o programa run_experiments.sh para obter os dados de todas as combinações e o programa show_experiments para mostrar os dados resultantes em gráficos. Os dados foram agrupados na tabela a seguir.

Tabela 1 – Tempo de Execução para Diferentes Implementações e Número de Threads

Grid Size	Implementação	Tempo (s)					
		1 Thread	2 Threads	4 Threads	8 Threads	16 Threads	32 Threads
32	seq	0.000705	-	-	-	-	-
	omp	0.000748	0.000424	0.000290	0.000387	0.001846	0.003571
	pth	0.001579	0.001699	0.002429	0.006002	0.012240	0.024882
64	seq	0.002872	-	-	-	-	-
	omp	0.002964	0.001523	0.000904	0.000855	0.002443	0.004071
	pth	0.004040	0.002889	0.003067	0.006272	0.012261	0.024678
128	seq	0.011755	-	-	-	-	-
	omp	0.012111	0.006694	0.003368	0.003360	0.004606	0.006071
	pth	0.013195	0.007484	0.005224	0.007468	0.012772	0.024902
256	seq	0.046723	-	-	-	-	-
	omp	0.047770	0.024203	0.012853	0.013733	0.017745	0.026503
	pth	0.067175	0.025937	0.014672	0.015281	0.017745	0.026503
512	seq	0.191314	-	-	-	-	-
	omp	0.195167	0.097602	0.056010	0.047840	0.042564	0.040003
	pth	0.198787	0.101194	0.052578	0.050090	0.043607	0.049621
1024	seq	0.769324	-	-	-	-	-
	omp	0.775330	0.390667	0.205015	0.184212	0.162272	0.143923
	pth	0.776519	0.395805	0.204216	0.191517	0.154722	0.155523
2048	seq	3.068454	-	-	-	-	-
	omp	3.123229	1.578376	0.790306	0.737891	0.621541	0.561380
	pth	3.067754	1.563511	0.798271	0.743270	0.624923	0.580782
4096	seq	12.126987	-	-	-	-	-
	omp	12.288760	6.198285	3.320094	2.939376	2.316637	2.175557
	pth	12.274873	6.340145	3.320094	2.939376	2.539036	2.176074

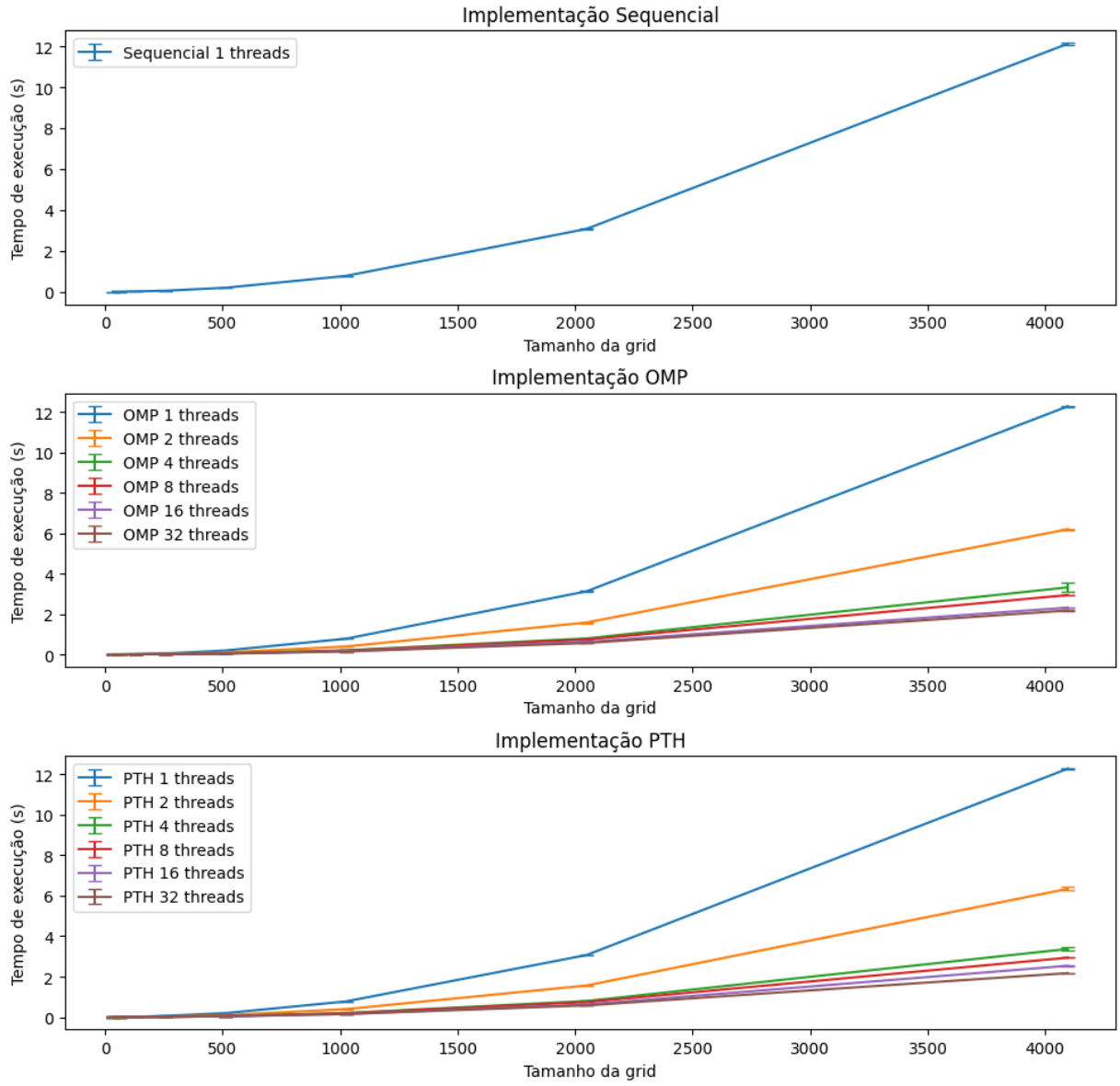

Figura 1 – Gráficos de cada implementação.

Cada gráfico representa uma impementação distinta e tem como eixo x o tamanho da grid e como eixo y o tempo de execução em segundos. As linhas representam o tempo médio de execução para cada combinação de tamanho da grid

e quantidade de threads, e as barras de erro destacam a variabilidade dos tempos de execução (confidence interval) para cada ponto de dados.

O aumento no tempo de execução conforme o tamanho da grid aumenta na implementação **sequencial** é esperado. Nesse caso, o algoritmo é executado em apenas uma thread, e a complexidade computacional do algoritmo pode ser diretamente proporcional ao tamanho da grid. À medida que a grid aumenta, mais operações e cálculos são necessários para processar os elementos da matriz, levando a um aumento linear no tempo de execução.

Na implementação com **OpenMP**, o aumento no tempo de execução com o aumento do tamanho da grid e do número de threads pode ser explicado pela interação entre o tamanho do problema e a forma como o paralelismo é aplicado. Quando o tamanho da grid é maior, o problema é mais complexo, exigindo mais cálculos e, consequentemente, mais tempo de execução. Além disso, o aumento do número de threads pode levar a um melhor paralelismo e, consequentemente, a uma redução do tempo de execução em alguns casos. No entanto, conforme o tamanho do problema cresce, pode haver uma sobrecarga na sincronização das threads, o que pode resultar em um aumento do tempo de execução com um grande número de threads.

Assim como na implementação OMP, o comportamento da implementação com **Pthreads** mostra aumento no tempo de execução à medida que o tamanho da grid e o número de threads aumentam. O aumento do número de threads também pode levar a uma melhoria inicial do desempenho, mas novamente, com um grande número de threads, a sobrecarga de sincronização pode resultar em um aumento do tempo de execução.

É importante notar que, em alguns casos, aumentar o número de threads não necessariamente reduziu o tempo de execução em todas as combinações de tamanho da grid e implementação. Isso se deve ao balanceamento entre o tempo de

computação e o tempo de sincronização das threads, sendo que um grande número de threads pode levar a uma competição por recursos e consequentemente a um aumento no tempo de execução.