


Relatório Trabalho 4 Arquitetura de Computadores - multi threads em C

Juan Veiga de Queiroz Silva - 21952878

Configurações da Máquina para o Experimento



Sistema Operacional	Linux Mint 21.1 Cinnamon
Versão do Cinnamon	5.6.5
Kernel do Linux	5.15.0-56-generic
Processador	Intel® Core™ i5-9500 CPU @ 3.00GHz × 6
Memória	7.5 GiB
Discos rígidos	502.6 GB
Placa de Vídeo	Intel Corporation CoffeeLake-S GT2 [UHD Graphics 630] (prog-if 00 [VGA controller])

- Arquitetura: x86_64
- Modo(s) operacional da CPU: 32-bit, 64-bit
- Address sizes: 39 bits (físico), 48 bits (virtual)
- Ordem dos bytes: Little Endian

CPU:

- CPU(s): 6
- Lista de CPU(s) on-line: 0-5
- ID de fornecedor: GenuineIntel
- Modelo da CPU: Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz
 - Família: 6
 - Modelo: 158
 - Thread(s) por núcleo: 1
 - Núcleo(s) por soquete: 6
 - Soquete(s): 1
 - Step: 10
 - CPU MHz máx.: 4400,0000
 - CPU MHz mín.: 800,0000
 - BogoMIPS: 6000.00

Caches:

- L1d: 192 KiB (6 instâncias)
- L1i: 192 KiB (6 instâncias)
- L2: 1,5 MiB (6 instâncias)
- L3: 9 MiB (1 instância)

NUMA:

- Nó(s): 1
- CPU(s) do Nó0 NUMA: 0-5

1. Execute o programa para diferentes ordens de matrizes e quantidades de threads.

Ordem das matrizes para teste: 500 1000 1500 2000 2500

Números de threads para teste: 2 4 8 16

Execute os testes pelo menos 15 vezes para cada combinação de N e T. Monte uma tabela com as médias e um gráfico de tempo em função de T para cada N com a média e as barras de erro

Matriz Tamanho 500 x 500				
Número de testes	Número de Threads			
	2	4	8	16
Teste 1	0.5082 seg	0.5213 seg	0.5291 seg	0.5508 seg
Teste 2	0.5088 seg	0.5261 seg	0.5279 seg	0.5322 seg
Teste 3	0.5153 seg	0.5233 seg	0.5294 seg	1.1240 seg
Teste 4	0.5195 seg	0.5259 seg	1.0626 seg	1.1338 seg
Teste 5	0.5185 seg	0.5223 seg	1.1529 seg	1.1489 seg
Teste 6	0.5101 seg	0.5413 seg	1.1848 seg	0.5340 seg
Teste 7	0.5145 seg	0.5204 seg	1.1803 seg	1.1409 seg
Teste 8	0.5131 seg	0.5233 seg	1.1845 seg	1.1535 seg
Teste 9	0.5080 seg	0.5193 seg	0.5495 seg	1.1654 seg
Teste 10	0.5072 seg	0.5203 seg	0.5278 seg	1.0693 seg
Teste 11	0.5095seg	0.5353 seg	0.5293 seg	0.6860 seg
Teste 12	0.5197 seg	0.5243 seg	0.5268 seg	0.6955 seg
Teste 13	0.5053 seg	0.5219 seg	0.5277 seg	1.1071 seg
Teste 14	0.5122 seg	0.5252 seg	0.5281 seg	0.5318 seg
Teste 15	0.5076 seg	0.5217 seg	0.5331 seg	0.5324 seg
Média	0,5118 seg	0,5248 seg	0,7383 seg	0,8737 seg

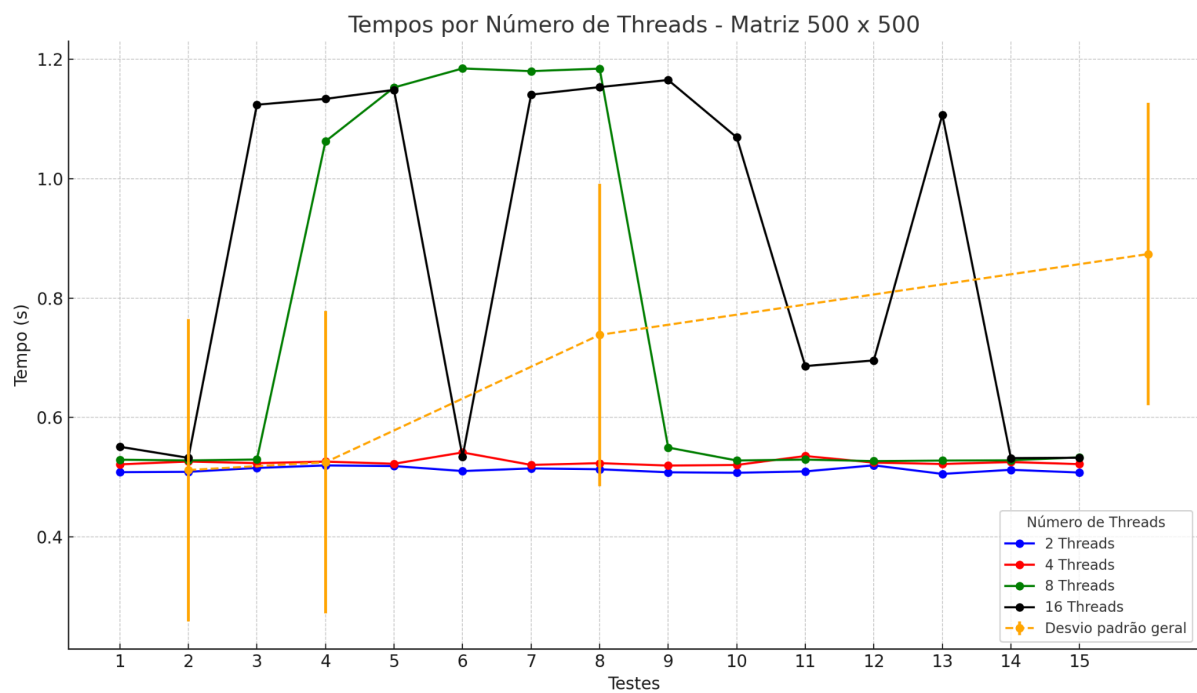


Imagem 1: Gráfico com as saídas da matriz de tamanho 500 x 500

Matriz Tamanho 1000 x 1000				
Número de testes	Número de Threads			
	2	4	8	16
Teste 1	4.3421 seg	4.3371 seg	4.4992 seg	4.4040 seg
Teste 2	4.2876 seg	4.3088 seg	4.4612 seg	4.3737 seg
Teste 3	4.3291 seg	4.4413 seg	4.8811 seg	4.3900 seg
Teste 4	4.4739 seg	4.2912 seg	4.4922 seg	4.4264 seg
Teste 5	4.3370 seg	4.4193 seg	4.5468 seg	4.5590 seg
Teste 6	4.2876 seg	4.3919 seg	4.5082 seg	4.5342 seg
Teste 7	4.4294 seg	4.3472 seg	4.5061 seg	4.4834 seg
Teste 8	4.2372 seg	4.4025 seg	4.4939 seg	4.9590 seg
Teste 9	4.4025 seg	4.3324 seg	4.5504 seg	4.5583 seg
Teste 10	4.4386 seg	4.3870 seg	4.5489 seg	4.5513 seg
Teste 11	4.6873 seg	4.7567 seg	4.4833 seg	4.4743 seg
Teste 12	4.3956 seg	4.3896 seg	4.4779 seg	4.5196 seg
Teste 13	4.3208 seg	4.4179 seg	4.5294 seg	4.6083 seg
Teste 14	4.3371 seg	4.3864 seg	4.4694 seg	4.5554 seg
Teste 15	4.3774 seg	4.4268 seg	4.6121 seg	5.0061 seg
Média	4,3789 seg	4,4024 seg	4,5373 seg	4,5602 seg

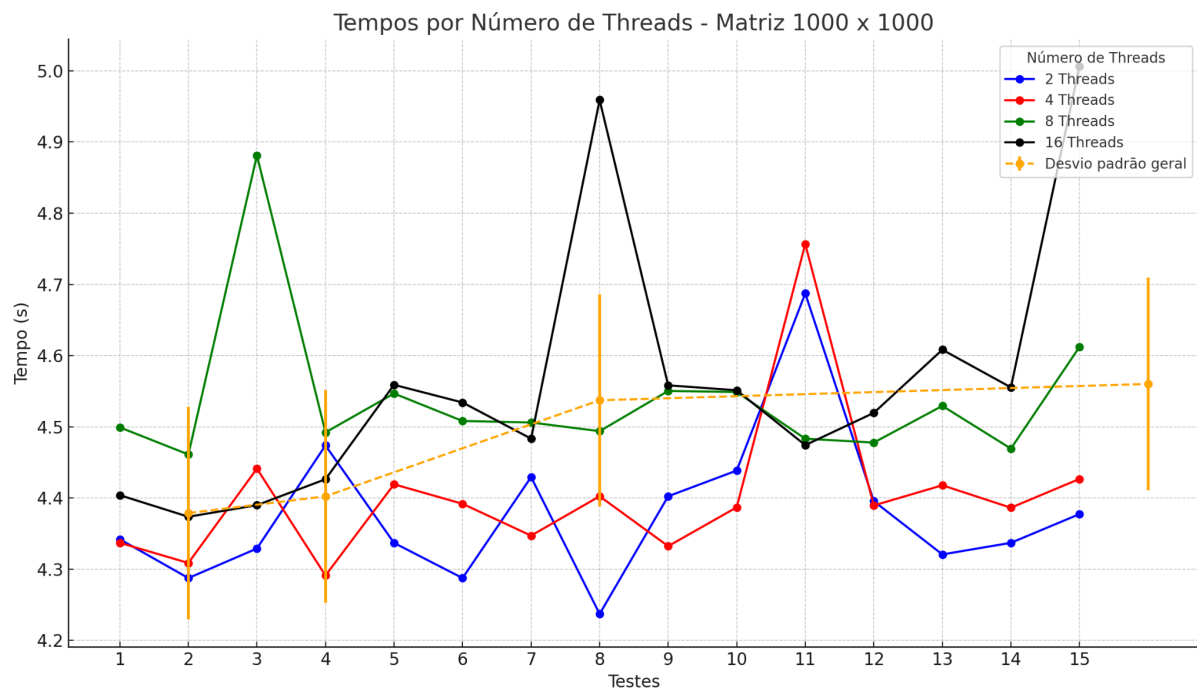


Imagem 2: Gráfico com as saídas da matriz de tamanho 1000 x 1000

Matriz Tamanho 1500 x 1500				
Número de testes	Número de Threads			
	2	4	8	16
Teste 1	20.0217 seg	20.1023 seg	20.4148 seg	20.4132 seg
Teste 2	20.0461 seg	20.0023 seg	20.2548 seg	20.4008 seg
Teste 3	20.3919 seg	19.8924 seg	20.3451 seg	20.2931 seg
Teste 4	19.6563 seg	20.2692 seg	20.1581 seg	20.3607 seg
Teste 5	19.6511 seg	20.1534 seg	20.1736 seg	20.5103 seg
Teste 6	19.7546 seg	20.0205 seg	20.3167 seg	20.2879 seg
Teste 7	19.7988 seg	20.2982 seg	20.2365 seg	20.4132 seg
Teste 8	19.9340 seg	19.9660 seg	20.3686 seg	20.5640 seg
Teste 9	19.7588 seg	20.0063 seg	20.4278 seg	20.3583 seg
Teste 10	19.8763 seg	20.2235 seg	20.4443 seg	20.3571 seg
Teste 11	19.7652 seg	20.0052 seg	20.2749 seg	20.4060 seg
Teste 12	19.9882 seg	20.0885 seg	20.4854 seg	20.3301 seg
Teste 13	20.5157 seg	20.3580 seg	20.4684 seg	20.4063 seg
Teste 14	20.8936 seg	20.3047 seg	20.2453 seg	20.4420 seg
Teste 15	20.5932 seg	20.0190 seg	20.4453 seg	20.3948 seg
Média	20,0430 seg	20,1140 seg	20,3373 seg	20,3959 seg

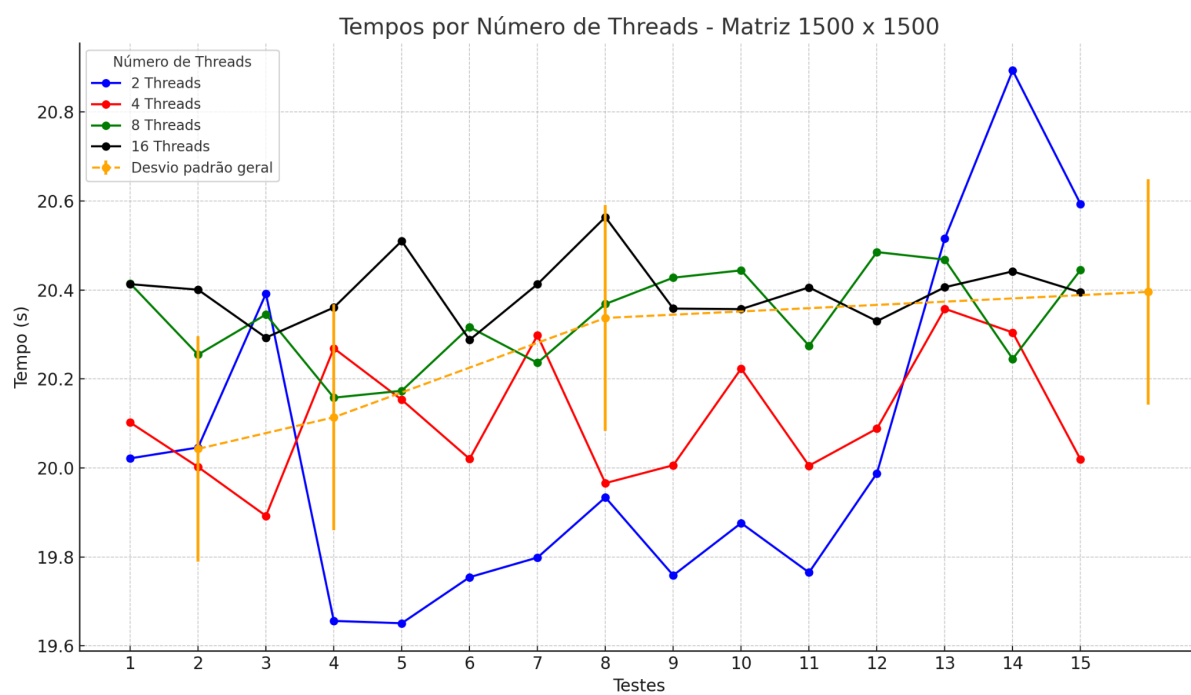


Imagem 3: Gráfico com as saídas da matriz de tamanho 1500 x 1500

Matriz Tamanho 2000 x 2000				
Número de testes	Número de Threads			
	2	4	8	16
Teste 1	61.8672 seg	64.7170 seg	61.3506 seg	61.5782 seg
Teste 2	62.4269 seg	62.6152 seg	61.3409 seg	60.4046 seg
Teste 3	61.7177 seg	62.5682 seg	61.7299 seg	60.4081 seg
Teste 4	62.1927 seg	62.8088 seg	61.5659 seg	60.6873 seg
Teste 5	61.1414 seg	62.7508 seg	61.4133 seg	60.6026 seg
Teste 6	62.4617 seg	62.6985 seg	61.7065 seg	60.4703 seg
Teste 7	61.3484 seg	62.8670 seg	62.3029 seg	60.6122 seg
Teste 8	62.3859 seg	62.0882 seg	62.0758 seg	60.8176 seg
Teste 9	61.5945 seg	63.0639 seg	62.0173 seg	60.3126 seg
Teste 10	62.9588 seg	61.9935 seg	62.1078 seg	60.6657 seg
Teste 11	63.8464 seg	63.7172 seg	62.2145 seg	61.4975 seg
Teste 12	63.5146 seg	64.5637 seg	62.1046 seg	61.4516 seg
Teste 13	63.2174 seg	63.9529 seg	62.2012 seg	61.2452 seg
Teste 14	62.1763 seg	63.6938 seg	62.5990 seg	61.6244 seg
Teste 15	63.2604 seg	62.3093 seg	62.4170 seg	61.7065 seg
Média	62,4074 seg	63,0939 seg	61,9431 seg	60,9390 seg

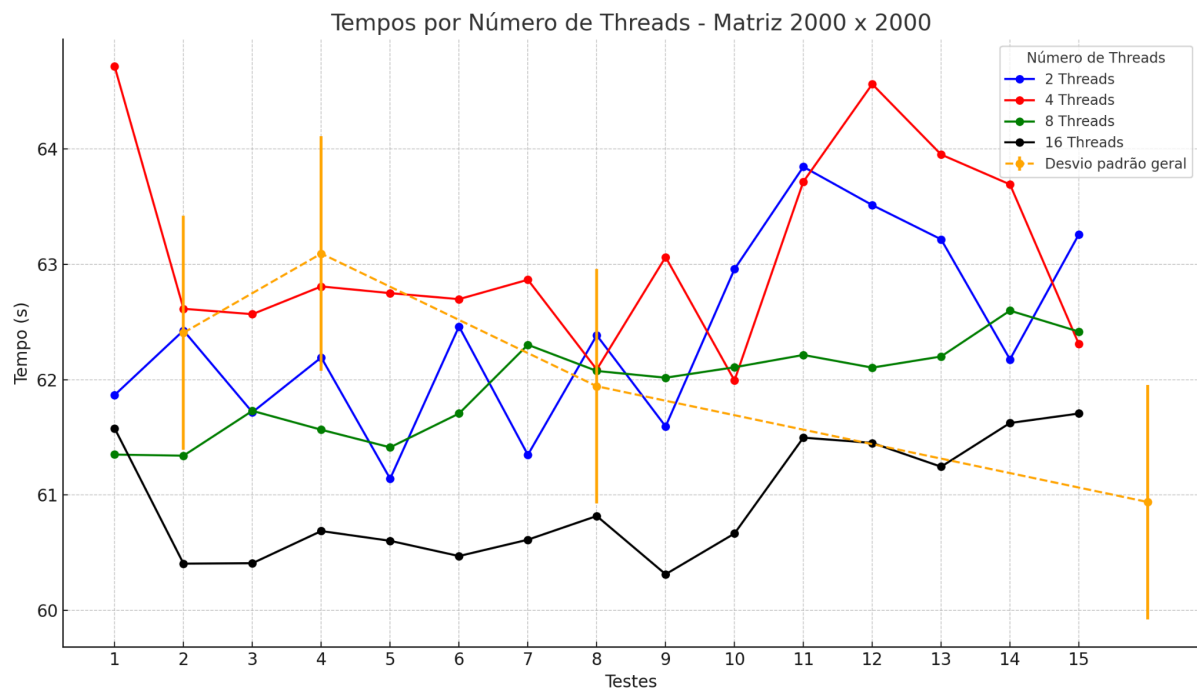


Imagem 4: Gráfico com as saídas da matriz de tamanho 2000 x 2000

Matriz Tamanho 2500 x 2500				
Número de testes	Número de Threads			
	2	4	8	16
Teste 1	129.4501 seg	126.2817 seg	131.4096 seg	131.0427 seg
Teste 2	124.7561 seg	132.2997 seg	131.3875 seg	131.4615 seg
Teste 3	127.6967 seg	128.9419 seg	130.3869 seg	132.9433 seg
Teste 4	128.9030 seg	129.7846 seg	130.2566 seg	130.8356 seg
Teste 5	131.0463 seg	127.2856 seg	132.4065 seg	130.3788 seg
Teste 6	131.4175 seg	131.7271 seg	130.9432 seg	129.5367 seg
Teste 7	125.3862 seg	131.2583 seg	130.6738 seg	130.9351 seg
Teste 8	131.9382 seg	131.1728 seg	131.9697 seg	130.5228 seg
Teste 9	129.8602 seg	130.5146 seg	130.4193 seg	130.2552 seg
Teste 10	130.4209 seg	131.6466 seg	130.8465 seg	130.1054 seg
Teste 11	126.7660 seg	128.9405 seg	131.2757 seg	131.1720 seg
Teste 12	129.3590 seg	131.1025 seg	131.9420 seg	129.9552 seg
Teste 13	126.2791 seg	129.1281 seg	130.6210 seg	131.9361 seg
Teste 14	130.0382 seg	131.8784 seg	131.5147 seg	131.0690 seg
Teste 15	132.0283 seg	130.8383 seg	131.2175 seg	130.3355 seg
Média	129,0231 seg	130,1867 seg	131,1514 seg	130,8323 seg

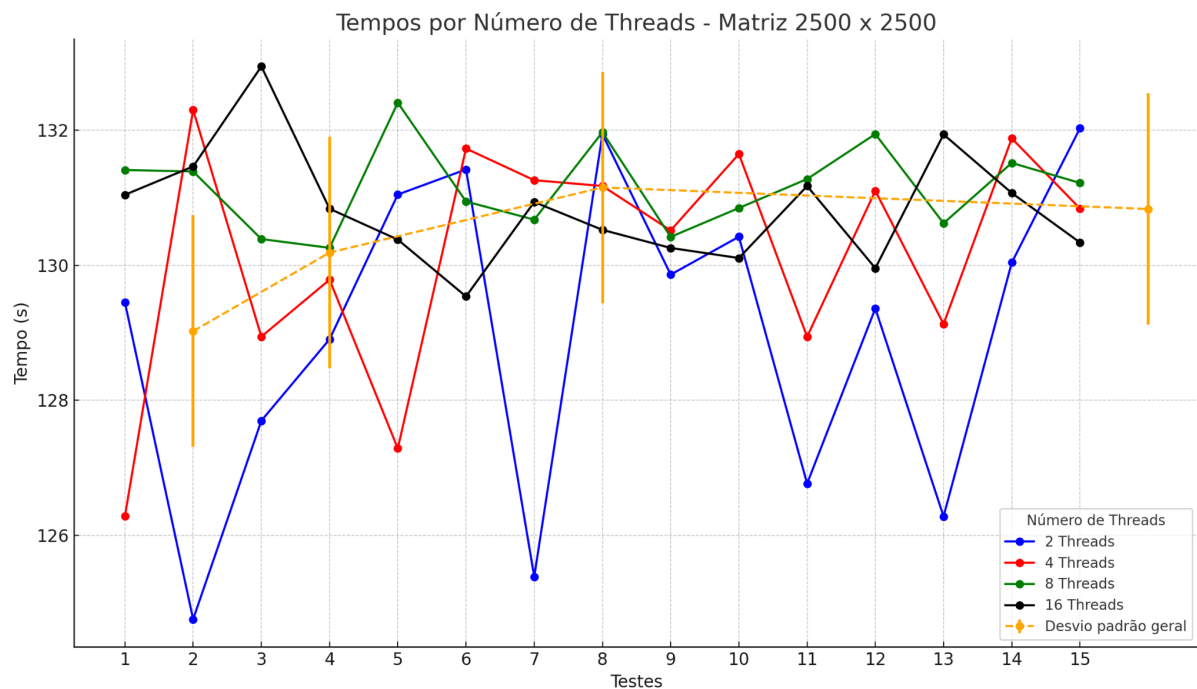


Imagem 5: Gráfico com as saídas da matriz de tamanho 2500 x 2500

2) Discuta o comportamento do desempenho conforme aumenta a quantidade de threads e a ordem da matriz.

Na matriz 500 x 500 com base nos tempos medidos, o desempenho varia conforme o número de threads. Para 2 e 4 threads, os tempos são consistentes e apresentam menor variabilidade, sugerindo eficiência no paralelismo. Para 8 e 16 threads, observa-se um aumento nos tempos médios e maior dispersão, indicando possível sobrecarga ou ineficiência no gerenciamento das threads. O desvio padrão geral reflete a variação entre todas as execuções, mostrando que configurações com mais threads introduzem instabilidade. Testes específicos (como o 3 e 4 para 16 threads) destacam tempos anômalos, sugerindo gargalos ou saturação do sistema. No geral, configurações com poucas threads equilibram melhor desempenho e estabilidade.

Na matriz 1000 x 1000, o desempenho varia conforme o número de threads. Para 2 e 4 threads, os tempos são consistentes, indicando bom aproveitamento do paralelismo com baixa variação. Para 8 e 16 threads, os tempos médios aumentam e apresentam maior dispersão, sugerindo sobrecarga no sistema ou limitações no escalonamento. O desvio padrão geral reflete a variabilidade entre as execuções, destacando maior instabilidade com maior número de threads. Testes específicos (como o 3 e 15 para 8 e 16 threads) apresentaram tempos significativamente maiores, evidenciando possíveis gargalos. No geral, configurações com 2 e 4 threads são mais eficientes e estáveis nesse cenário.

Na matriz 1500 x 1500, observa-se estabilidade para 2 e 4 threads, com tempos médios próximos e menor dispersão, indicando boa eficiência no uso de recursos. Configurações com 8 e 16 threads apresentam maior variabilidade e tempos médios mais altos, sugerindo ineficiências no paralelismo para este tamanho de matriz. O desvio padrão geral, representado no gráfico, reflete a dispersão entre as execuções, especialmente em configurações com mais threads. Alguns testes, como o 3 e 4 para 16 threads, destacaram tempos anômalos, indicando gargalos. No geral, a configuração com 2 threads oferece um melhor equilíbrio entre desempenho e estabilidade.

Na matriz 2000 x 2000, o desempenho é consistente para 2 e 4 threads, com tempos estáveis e baixa dispersão. Para 8 e 16 threads, os tempos médios são menores, especialmente em 16 threads, o que sugere melhor escalabilidade para matrizes maiores. Contudo, a variabilidade entre os testes aumenta, refletida no desvio padrão geral. Configurações com mais threads ainda apresentam instabilidade, como em alguns testes de 8 threads. O menor tempo médio para 16 threads indica eficiência no paralelismo, mas a estabilidade observada em 2 threads mantém-se como referência para maior confiabilidade.

Na matriz 2500 x 2500, os tempos médios aumentam com o número de threads, sendo mais altos para 8 threads. Configurações com 2 e 4 threads mostram estabilidade, com menor variabilidade e tempos médios próximos. Para 8 e 16 threads, observa-se maior dispersão nos tempos, evidenciando sobrecarga ou gargalos no gerenciamento do paralelismo. O desvio padrão geral reflete maior variabilidade para configurações com mais threads, especialmente em testes como o 2 e 3. No geral, enquanto 16 threads oferecem bom desempenho médio, as configurações com 2 e 4 threads continuam sendo as mais equilibradas para desempenho e estabilidade.

3. Em quais situações o aumento do número de threads apresenta um ganho de desempenho significativo, e em quais situações não?

- Situações com ganho significativo de desempenho:

Na matriz 2000 x 2000 (16 threads), o tempo médio diminui consideravelmente em relação a 2 e 4 threads, destacando melhor escalabilidade para matrizes maiores.

Na matriz 2500 x 2500 (16 threads), apesar da dispersão maior nos tempos, o uso de 16 threads apresenta tempos competitivos, sugerindo eficiência no uso de paralelismo para cargas elevadas.

Matrizes maiores (2000 x 2000 e 2500 x 2500) geralmente apresentam melhor aproveitamento dos recursos, reduzindo o tempo médio de execução com o aumento do número de threads.

- Situações sem ganho significativo de desempenho:

Nas matrizes 500 x 500 e 1000 x 1000 (8 e 16 threads), não há redução expressiva nos tempos médios, e a maior dispersão reflete possíveis gargalos no escalonamento.

Na matriz 1500 x 1500 (8 e 16 threads), apesar de leve redução em alguns tempos, o aumento de threads causa maior variabilidade, indicando ineficiências no gerenciamento para matrizes médias.

Matrizes pequenas (500 x 500 e 1000 x 1000) apresentam sobrecarga ao aumentar o número de threads para 8 e 16, sem ganhos relevantes, devido ao custo adicional de coordenação.

4. Discuta os possíveis overheads introduzidos pelo uso de threads na multiplicação de matrizes.

Os principais overheads do uso de threads na multiplicação de matrizes incluem custos de criação e gerenciamento das threads, que aumentam com o número de threads, e a necessidade de sincronização entre elas, resultando em atrasos. Além disso, há competição por recursos de cache, especialmente em sistemas com limites de threads por núcleo, e sobrecarga causada por trocas de contexto em ambientes com muitas threads.

Em matrizes pequenas, o custo de dividir e coordenar tarefas pode superar os ganhos paralelos. Esses fatores explicam a redução de eficiência e estabilidade observada no relatório, especialmente para 8 e 16 threads em matrizes menores.