MAC0219 - Relatório EP03

Nathália Yukimi Uchiyama Tsuno (14600541)
Renata Meyer Hobold (14605790)

December 2024

Contents

1 Introdução

O conjunto de Julia é um exemplo interessante de fractal, criado a partir da iteração de funções matemáticas com números complexos. Um dos tipos mais comuns dessa iteração envolve a função $f(z) = z^2 + c$, onde c é um valor complexo que determina a forma do conjunto. À medida que iteramos essa função para diferentes pontos no plano complexo, geramos padrões que se repetem em várias escalas, criando imagens visualmente impressionantes. No caso específico que vamos explorar neste trabalho, o valor de c é $c = -0.79 + 0.15i$, gerando um conjunto de Julia com uma aparência única.

Calcular o conjunto de Julia para cada ponto do plano complexo pode ser um processo bastante lento, especialmente quando a resolução da imagem é alta. Isso porque é preciso realizar muitas iterações para cada ponto, o que exige um grande poder de processamento. A paralelização oferece uma solução eficiente para esse problema. Nesse EP, vamos fazer uma versão sequencial de um programa gerador do conjunto de Julia e também uma versão paralela, o qual iremos executar com `MPICH` e com `SimGrid`.

Com auxílio de ferramentas de manipulação e visualização de dados, como as bibliotecas Pandas, Matplotlib, Numpy e Seaborn do Python, seremos capazes de expressar com um grande potencial e clareza as análises obtidas dos experimentos.

O modelo do computador em que foram executados os testes tem as seguintes especificações:

CPU	11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz
RAM	7.48Gi
SO	Ubuntu 20.04.6 LTS (Focal Fossa)

Table 1: Especificações

Para compilar o programa sequencial `sequential_julia.c`, execute o comando:

$ `make`

E um executável `sequential_julia` será criado.

2 Experimentos com **`MPICH`**

`MPICH` é um software livre com implmentação portátil do `MPI`. Por intermédio de suas aplicações, foram possíveis as execuções de alguns experimentos.

2.1 Medição de Tempo e Análise de Desempenho

A priori, foram realizados 10 experimentos para cada setagem de parâmetros, de tamanhos de imagem distintos (500, 1000, 5000), utilizando diferentes números de processos (2, 4, 8, 16). Abaixo, algumas representações gráficas.

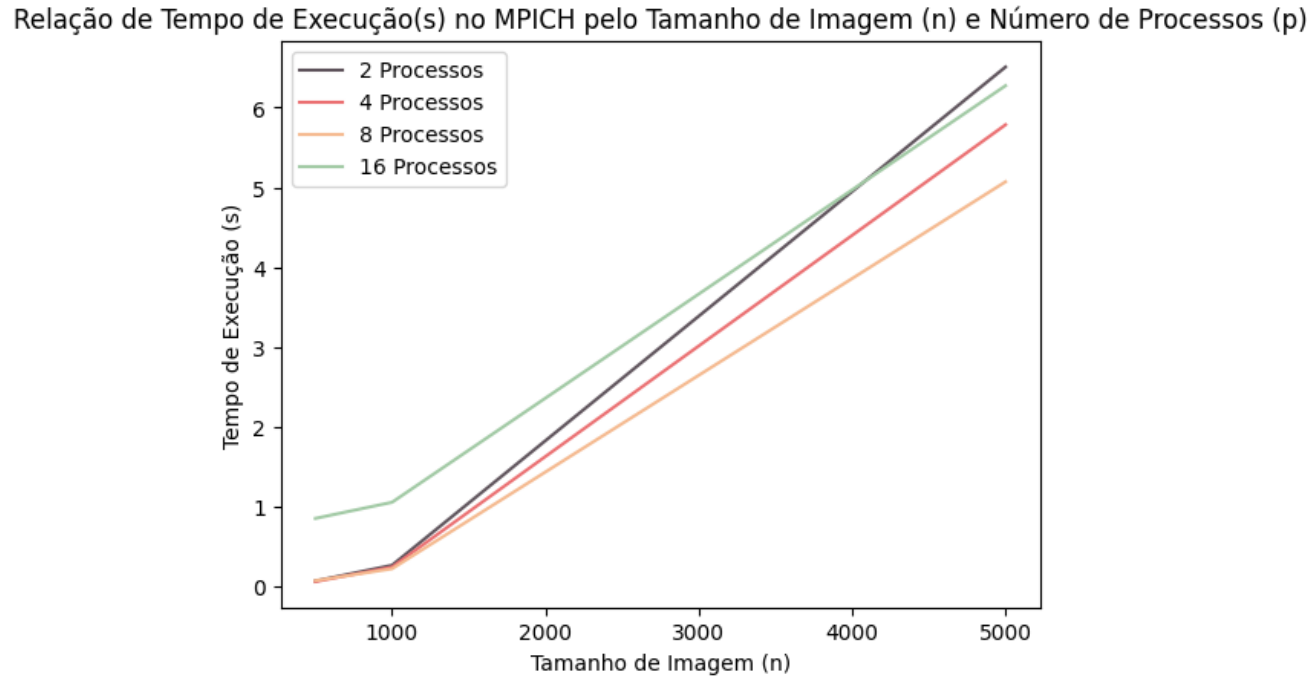
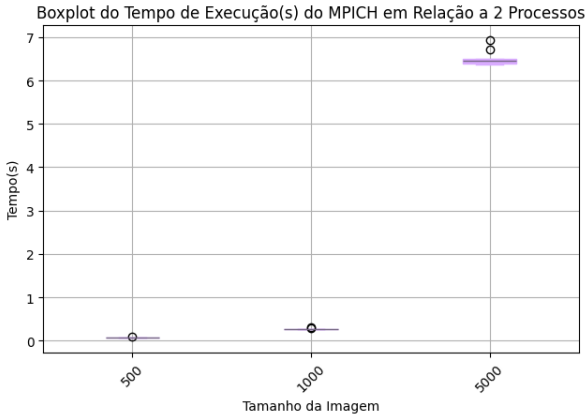
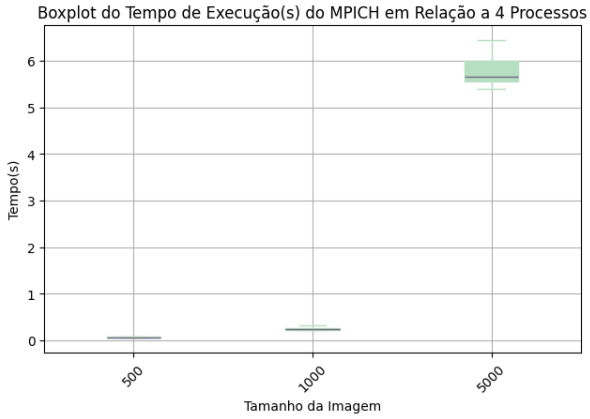


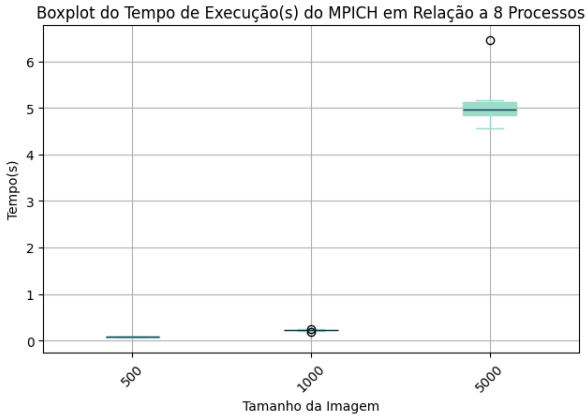
Figure 1: Tempo Médio de Execucução



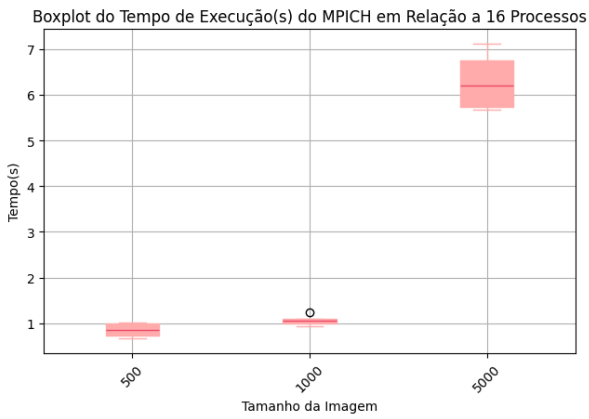
(a) Subfigura 1: 2 Processos



(b) Subfigura 2: 4 Processos



(c) Subfigura 3: 8 Processos



(d) Subfigura 4: 16 Processos

Figure 2: BoxPlots dos Testes de Execução para Processos = (2, 4, 8, 16) e Tamanho da Imagem = (500, 1000, 5000), para uma Amostra de Tamanho 10

Como é possível analisar, a quantidade de processos **não** é linear ao tempo de execução do programa. No que se refere aos experimentos com 2, 4 e 8 processos, os tempos de execução são próximos nos dois primeiros tamanhos. Em 500, conforme os boxplots, as variâncias são expressivamente baixas e próximas a 0, enquanto, para 1000, a 0.5s.

Contudo, para 5000, os programas já demonstram algumas disparidades. A tendência tracejada é de que um programa executado com 8 processos tenha um tempo de execução próximo a 5s e, consequentemente, mais rápido do que um com 4. Este, por sua vez, executado em cerca de 5.75s, melhor que um com 2, com tempo de execução média a 6.5s. Nessa etapa, os experimentos já indicaram alguma variabilidade entre si.

Finalmente, para 16 processos, foi o que demonstrou variação mais discrepante entre todos. Para uma imagem com tamanho 500, demandou-se um tempo próximo a 1s (pior do que para menos processos com uma imagem com o dobro do tamanho) e os experimentos já demonstraram uma forte variabilidade. Para 1000, ultrapassou-se a casa dos 1s. Em experimentos com imagens com tamanho 5000 (10 vezes maior), o desempenho médio foi melhor do que com 2 processos e, somente isso. Note que 50% dos dados eram mais próximos a 6 segundos. Ainda assim, a variância de dados foi a maior entre todos.

Ademais, como esperado, dado um número de processos fixo, a tendência é o aumento de tempo de execução do programa em relação ao tamanho da imagem, uma vez que são mais pixels para serem processados. Embora ela não seja linear, conforme visível pelo gráfico.

2.2 SpeedUp

Observe os tempos de execução das versões paralela e sequencial:

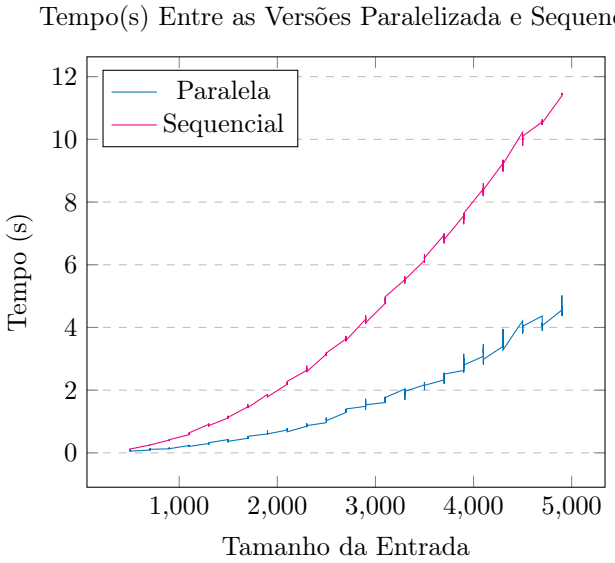


Figure 3: Tempo(s) Entre as Versões Paralelizada e Sequencial para uma Amostra com 10 Casos

Em cima desses valores, calculamos o speedUp, que é a razão speedUP $= \frac{\text{tempo versão sequencial}}{\text{tempo versão paralela}}$.

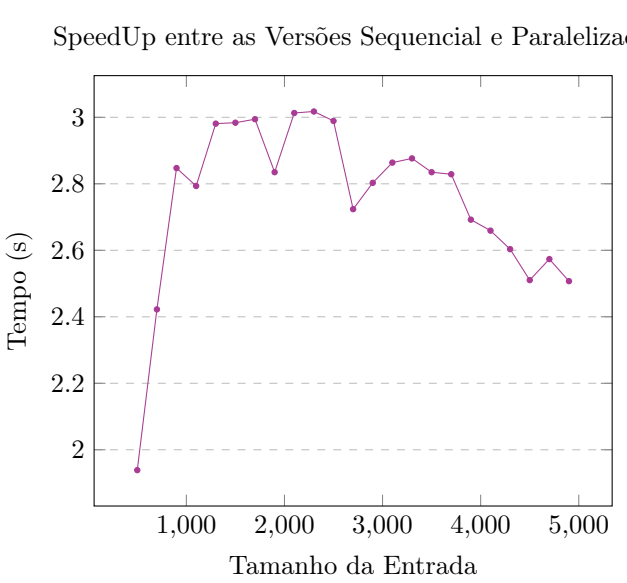


Figure 4: SpeedUp Médio entre a Versão Sequencial e Paralelizada para 7 Processos

n	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
500	2.089372	0.472254	0.957776	1.995668	2.243063	2.377386	2.573227
700	2.538232	0.557609	1.681037	2.188306	2.588559	2.987608	3.246696
900	2.877698	0.302425	2.510279	2.593181	2.830462	3.195119	3.220240
1100	2.811898	0.267439	2.302439	2.761949	2.821547	2.936773	3.199181
1300	2.996824	0.257446	2.678413	2.807861	2.978377	3.127698	3.440928
1500	3.004233	0.278007	2.631480	2.833239	2.953822	3.111236	3.516952
1700	3.006817	0.234667	2.735472	2.856164	2.910275	3.155507	3.443755
1900	2.843794	0.178752	2.509396	2.756204	2.831501	2.960494	3.090716
2100	3.019995	0.170747	2.828948	2.915855	2.977697	3.100779	3.400860
2300	3.020999	0.101472	2.883330	2.950610	3.005871	3.088351	3.203523
2500	2.994061	0.133180	2.830520	2.903882	2.993521	3.043570	3.274428
2700	2.726494	0.106985	2.562868	2.683607	2.705866	2.822193	2.858013
2900	2.817343	0.229689	2.484874	2.714110	2.833982	2.919393	3.185316
3100	2.868371	0.134958	2.679498	2.747701	2.889050	2.961061	3.077620
3300	2.884423	0.174111	2.710257	2.782769	2.843385	2.925541	3.315216
3500	2.838689	0.118493	2.677181	2.784725	2.826555	2.876388	3.099162
3700	2.835964	0.159765	2.617772	2.722629	2.776258	2.970915	3.061556
3900	2.702565	0.182294	2.372335	2.662449	2.728550	2.837812	2.901728
4100	2.668611	0.183083	2.400281	2.546381	2.657731	2.732145	3.059634
4300	2.609424	0.135053	2.324580	2.548309	2.633471	2.690135	2.806214
4500	2.512448	0.071590	2.428626	2.450835	2.491434	2.575566	2.631657
4700	2.575374	0.071109	2.418806	2.556671	2.592466	2.600102	2.695036
4900	2.512586	0.121042	2.278485	2.462272	2.571370	2.598237	2.616216

Table 2: Resumo estatístico do speedUp

No geral, podemos perceber que a paralelização funciona muito bem para tamanhos moderados de n, mas apresenta algumas limitações conforme o problema cresce além de certo ponto. Nos primeiros testes, com valores menores de n como 500 ou 700, o speedUp médio é relativamente bom, mas tem uma alta variabilidade nos resultados (note o desvio padrão mais elevado). Para esses tamanhos, a parte do problema que pode ser paralelizada ainda não domina completamente o tempo total de execução (a maior parte do tempo pode ser em decorrência da escrita no arquivo, que é sequencial, ou na própria criação dos processos, por exemplo). Conforme n aumenta, entre 1300 e 2300, o desempenho do programa melhora de forma consistente, com o speedUp médio chegando a valores próximos de 3. A paralelização parece estar atingindo seu potencial máximo para esses tamanhos. A carga computacional é bem dividida entre os processos. Além disso, o desvio padrão diminui, o que mostra que os resultados são mais consistentes e menos afetados por flutuações no ambiente de execução. Mas a partir de $n = 2500$, o speedUp médio começa a estabilizar e até decair levemente. Isso pode ser explicado por fatores como

a sobrecarga crescente da comunicação entre os processos e a saturação dos recursos do sistema, como o uso da memória ou a própria saturação dos processos, que no caso são apenas $t = 7$. Para valores muito grandes de n os ganhos da paralelização são menores. O problema começa a ser limitado mais pela parte sequencial do código e pela eficiência de acesso à memória do que pela capacidade de dividir o trabalho entre processos.

Podemos ver também o resumo estatístico da média dos speedUps:

Métrica	Valor
Média	2.772009
Desvio Padrão	0.312965
Mínimo	0.957776
Q1	2.600404
Mediana	2.798058
Q3	2.952197
Máximo	3.516952

Table 3: Resumo estatístico do speedUp médio

Os dados mostram que o speedUp médio é de 2.77, o que é proximo da médiana, com um desvio padrão moderado/baixo. O intervalo interquartil e a mediana parecem sugerir uma distribuição relativamente simétrica e positiva do desempenho. Isto é, que, para a maioria dos tamanhos de entrada n testados, houve uma melhoria consistente, embora houvessem valores abaixo de 1, o que é uma piora.

2.3 Medições de Eficiência

Para dar suporte ao observado acima, na subseção 2.1 desse relatório, foram realizados mais alguns experimentos. Para um número fixo de tamanho da imagem ($n = 1000$) e 10 casos amostrais para cada conjunto de argumentos do programa `1D_parallel_julia`. Foram analisados os tempos médios de execução, em segundos, para valores entre 2 a 20 processos.

Tempo de execução (`MPICH`) em função da quantidade de processos t

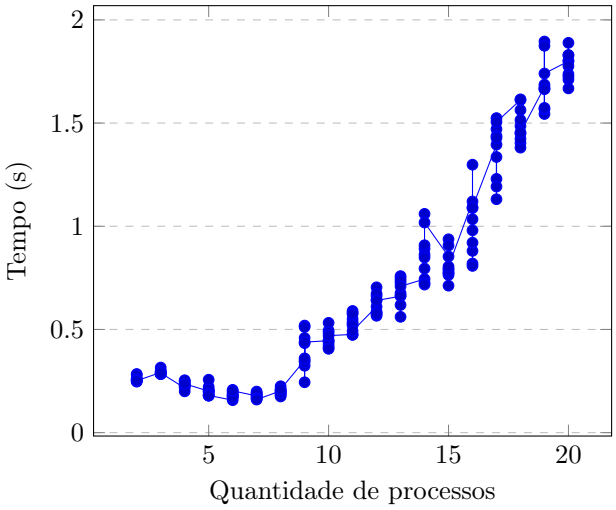


Figure 5: Tempo de execução (`MPICH`) em função de t

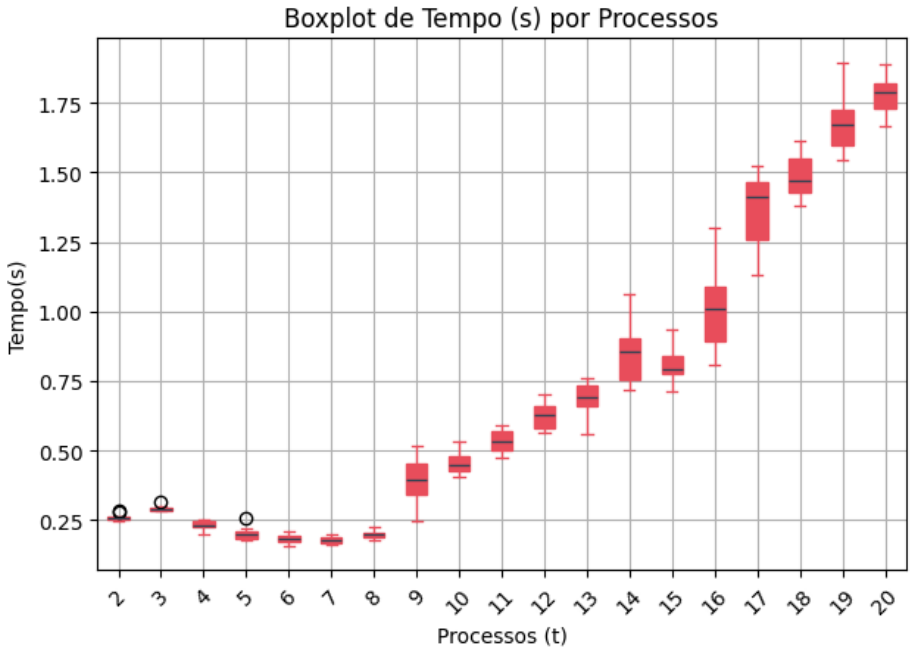


Figure 6: Caption

Conforme analisado pelos gráficos acima, os processos se iniciam em 2, com tempo médio em 0.25s e os valores demonstram uma tendência decrescente até a faixa dos 6, 7 e 8 processos, onde atingem cerca de 0.2s, com variabilidade expressivamente baixa. Este é o ponto onde o programa mantém uma eficiência aceitável e confirmando o visulizado em que programas com 8 processos eram melhores do que os de 2, 4 e 16.

Por outro lado, após esse pequeno vale, a tendência visualizada é de, quanto mais processos, maior o tempo de execução envolvido. No qual, para 16 processos, tem-se uma das maiores variâncias na análise de boxplots.

Esse fenômeno pode ser explicado por algumas causas:

- Overhead de Comunicação: envolve aumento dos custos de comunicação (latência e largura de banda associados aos meios de comunicação), havendo, cada vez mais, uma sobrecarga;
- Ociosidade de Processos;
- Overhead de Gerenciamento e Sincronização: o `MPICH` necessita realocar recursos para criar e finalizar processos. Assim, quanto mais deles, mais trabalho é necessário. Ademais, utilizam-se barreiras para sincronização para o cômputo do conjunto de Julia, o que também pode afetar no desempenho no caso de muitos processos;
- Escalabilidade de Recursos e Tempo de Latência;

O programa parece mantér uma eficiência aceitável até 7 processos, ponto a partir do qual se parece diminuir a eficiência.

3 Experimentos com `SimGrid`

O `SimGrid` é uma aplicação para simulação com sistemas distribuídos. Assim, foram aplicados alguns testes referentes à paralelização ao conjunto de Julia.

3.1 Impacto da Heterogeneidade dos Nós

A priori, executando-se em 10 casos amostrais o programa `1D_parallel_julia`, utilizando-se 8 processos, uma imagem de tamanho 5000 e o arquivo `simple_cluster.xml`, foi possível obter as distribuições abaixo.

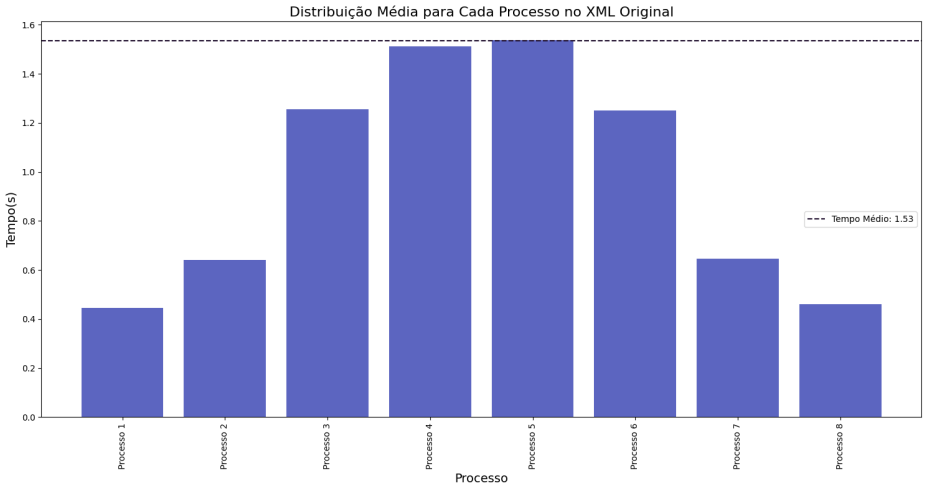


Figure 7: Para 8 Processos e Tamanho da Imagem 5000

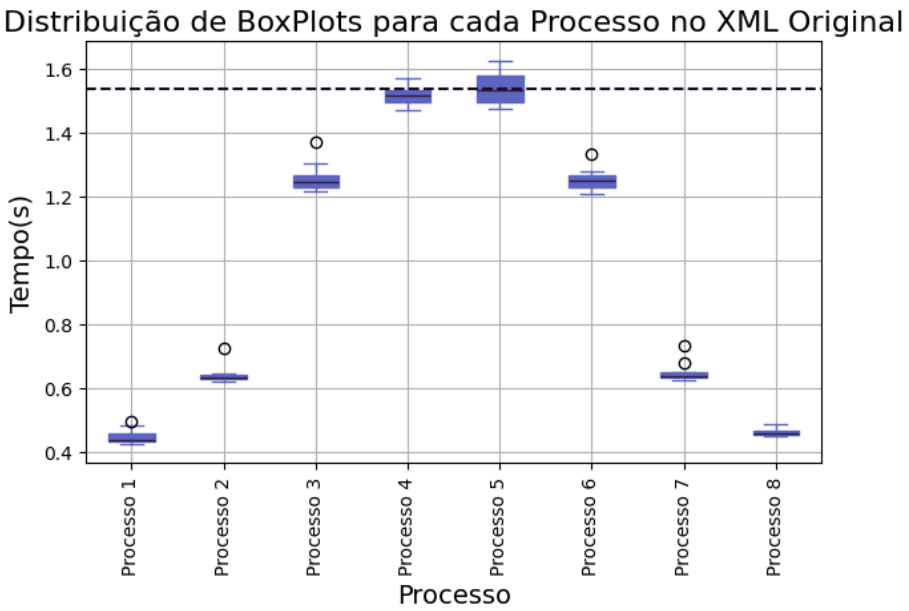


Figure 8: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.446683	0.022504	0.422727	0.431745	0.436354	0.454744	0.494430
2	0.641377	0.028016	0.619908	0.627194	0.632635	0.641543	0.722670
3	1.256952	0.043928	1.215255	1.225508	1.245441	1.263661	1.367277
4	1.513311	0.027505	1.468832	1.495468	1.515845	1.529534	1.567868
5	1.537436	0.048378	1.474059	1.495269	1.529711	1.577219	1.624139
6	1.250812	0.034785	1.207209	1.228014	1.248104	1.263797	1.333032
7	0.647153	0.031393	0.621630	0.629859	0.634279	0.647023	0.729478
8	0.460181	0.010712	0.447833	0.453401	0.457627	0.465839	0.483979

Table 4: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.23869104812850617, 0.24929857387149384)
2	(0.3340806193187789, 0.34277895468122116)
3	(0.6473623104889046, 0.6532727155110956)
4	(0.7814390807520727, 0.7937834832479272)
5	(0.7891705004610297, 0.8036064035389705)
6	(0.6342202849589218, 0.6488209450410782)
7	(0.33141323221384905, 0.3377683137861511)
8	(0.2400725763801312, 0.25243956161986886)

Table 5: Distribuição dos Intervalos de Confiança, em segundos

Nesse conjunto de testes, mantendo as configurações do arquivo, verificou-se que alguns processos tiveram mais impacto do que os outros. Por exemplo, os processos que menos tiveram influência foram os iniciais (com média de 0.45s) e os finais (com média de 0.46s), pouco disparsos no conjunto amostral.

Por outro lado, os processos com maior impacto foram os intermediários (4, com média 1.51s e 5, 1.54s e maior desvio padrão, indicando a variabilidade). Certamente, onde mais houve sobrecarga do sistema.

Apesar disso, o tempo entre o maior registro e o menor foi de 1.2s. Assim, os processos tiveram um desempenho relativamente próximos.

A posteriori, foram executados mais alguns experimentos, alterando-se o campo `speed` dos nós. Nessa sequência, foi se preenchendo o speed de 4 em 4 nós consecutivos como logaritmos de 2, inicializando-se no valor 1024. Por ilustração:

- Nó 1 : speed 1024Gf
- Nó 2 : speed 1024Gf
- Nó 3 : speed 1024Gf
- Nó 4 : speed 1024Gf
- Nó 5 : speed 512Gf
- Nó 6 : speed 512Gf
- ...
- Nó 9 : speed 256Gf
- Nó 10 : speed 256Gf
- ...
- Nó 63 : speed 0.03125 Gf
- Nó 64 : speed 0.03125 Gf

Obtendo-se a seguinte distribuição de tempo por processo.

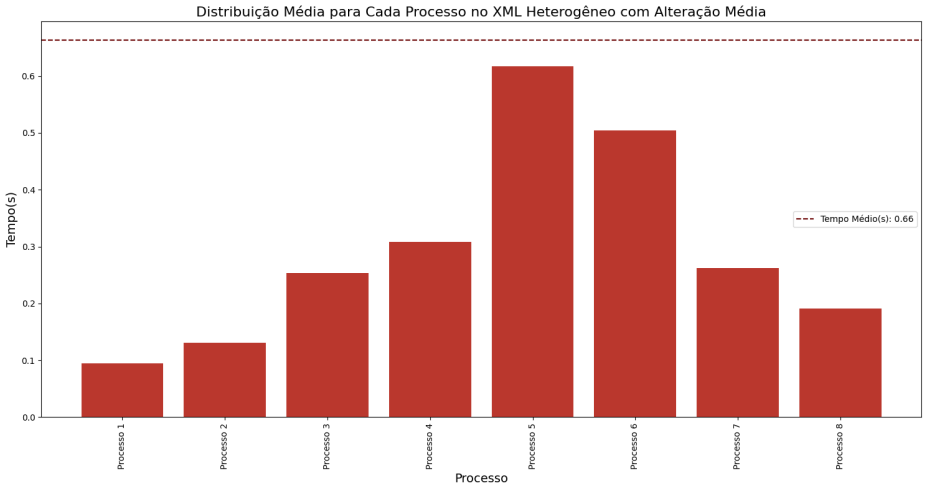


Figure 9: Para 8 Processos e Tamanho da Imagem 5000

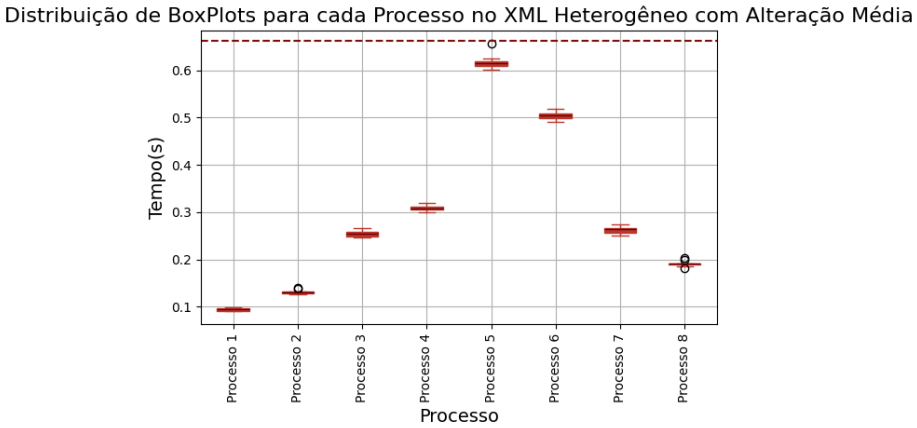


Figure 10: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.094238	0.002616	0.090592	0.091641	0.094471	0.096161	0.099022
2	0.131105	0.004153	0.126105	0.128854	0.130440	0.131338	0.139922
3	0.254060	0.006030	0.245540	0.248958	0.253484	0.257861	0.265225
4	0.307933	0.005308	0.300022	0.305073	0.307255	0.310924	0.319207
5	0.616766	0.014513	0.600737	0.608353	0.614137	0.618172	0.655858
6	0.504241	0.008296	0.490177	0.498932	0.504487	0.508546	0.517492
7	0.262568	0.006608	0.251264	0.257205	0.263311	0.265781	0.273166
8	0.191064	0.005960	0.181690	0.188402	0.190570	0.192038	0.202940

Table 6: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.09280346825663624, 0.09567240574336372)
2	(0.12882833911662475, 0.1333821628833752)
3	(0.25075393797177464, 0.25736590602822546)
4	(0.30502240622663446, 0.3108434557733655)
5	(0.6088085029255609, 0.6247225770744392)
6	(0.49969224350048563, 0.5087897624995144)
7	(0.25894491314931073, 0.2661910488506893)
8	(0.18779635441439418, 0.1943316235856058)

Table 7: Distribuição dos Intervalos de Confiança, em segundos

Nesse conjunto de dados, percebem-se algumas dispersões adicionais. Os processos com menor impacto foram os iniciais (1, com média 0.09s e 2, com média 0.13s), ambos com baixa variabilidade amostral. Por outro lado, os processos mais impactados foram os intermediários (5, 0.61s na média, 6, 0.5s na média). Possivelmente, também uma sobrecarga do sistema.

Ainda assim, as dispersões amostrais por processo foram relativamente baixas. E todos os casos tiveram uma diferença de, no máximo, 0.57s. Um valor muito próximo entre os processos, para diferenças em impactos globais do programa.

Por fim, foram testados algumas configurações de `speed` mais extremas, com valores variando no intervalo de [0.00004Gf, 400Gf]. Para uma melhor análise, foi variado a quantidade de processos e a ordem de inserção dessas velocidades. Impressionantemente, foi notado que esse fator importava, e muito.

Numa primeira ocasião, foram testados alguns nós da forma:

- Nó 1 : Speed 400Gf
- Nó 2 : Speed 40Gf
- Nó 3 : Speed 4Gf
- Nó 4 : Speed 0.4Gf
- Nó 5 : Speed 0.04Gf
- Nó 6 : Speed 0.004Gf
- Nó 7 : Speed 0.0004Gf
- Nó 8 : Speed 0.00004Gf
- Nó 9 : Speed 400Gf
- ...
- Nó 63 : Speed 0.0004Gf
- Nó 64 : Speed 0.00004Gf

Em outras palavras, intercalando os valores de forma brusca entre os nós. Consecutivamente, foi obtida a seguinte distribuição.

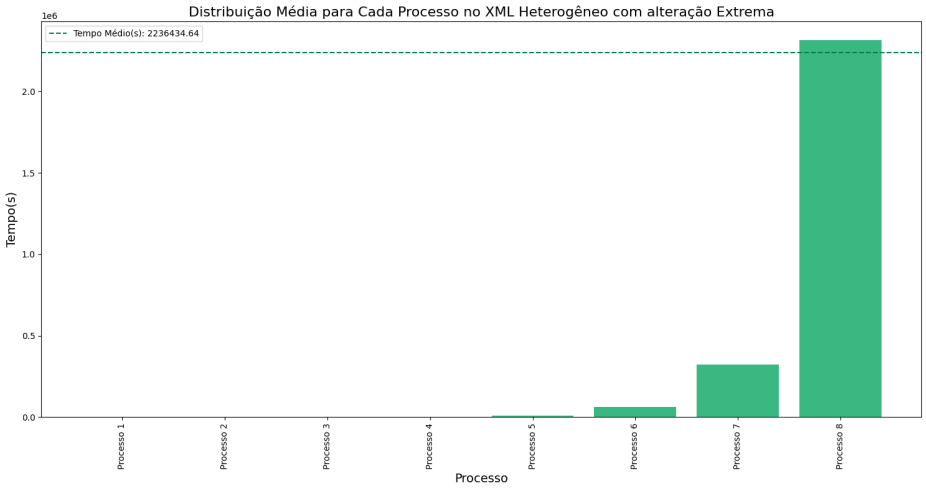


Figure 11: Para 8 Processos e Tamanho da Imagem 5000

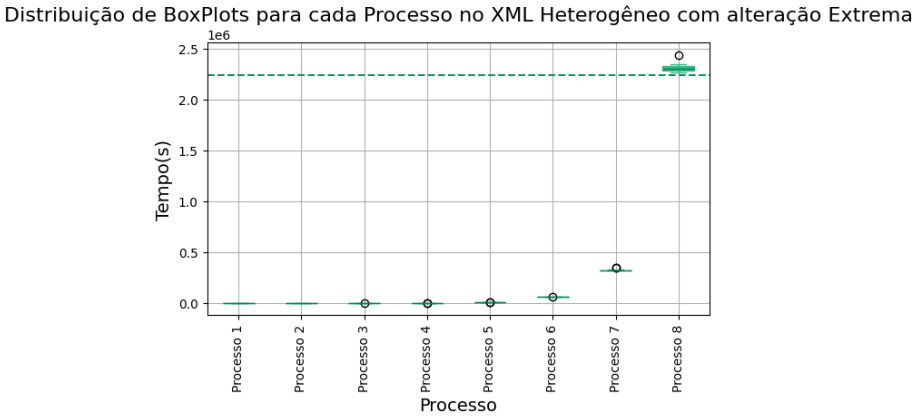


Figure 12: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.231755	0.002909	0.228076	0.229224	0.231046	0.234532	0.236559
2	3.211357	0.028595	3.162340	3.190386	3.213021	3.228677	3.266047
3	62.314521	0.561691	61.532988	61.947352	62.311339	62.480536	63.522919
4	766.653975	23.492850	752.058923	755.254917	756.266857	764.329607	831.784863
5	7605.872338	141.275370	7485.131610	7504.332656	7549.423708	7612.490148	7882.023770
6	62062.055580	673.398822	61400.138500	61580.521863	61831.538750	62187.078025	63544.739250
7	323205.402450	9894.539667	315771.080500	316837.296875	318193.938000	323482.503125	342762.068000
8	2313771.725500	47838.972500	2267613.610000	2277853.627500	2303857.807500	2323343.672500	2436658.940000

Table 8: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.2301599554776126, 0.23335005452238738)
2	(3.1956785956226916, 3.227035100377308)
3	(62.00655429408443, 62.62248748591556)
4	(753.7732081565302, 779.5347411834696)
5	(7528.413236185543, 7683.331438854459)
6	(61692.84141517071, 62431.26974486928)
7	(317780.3792973844, 328630.4256026556)
8	(2287542.3563577333, 2340001.094642307)

Table 9: Distribuição dos Intervalos de Confiança, em segundos

Nesse contexto, verifique que os nós com menor impacto global foram os iniciais, 1 (com média 0.23s, e desvio padrão relativa e expressivamente baixo, indicando pouca variabilidade amostral) e 2 (com média 3.21s e desvio padrão baixo, 0.02s). Por outro lado, o nó com maior impacto global foi o último, 8, com média indicada em 2313771.72s (26.77 dias!) e desvio padrão expressivamente alto, 47838.97s (13 horas), indicando alta variabilidade dos dados amostrais.

Isso é um forte indício para desbalanceamento de cargas entre os nós, sendo o oitavo processo, provavelmente, alocado em um nó com uma velocidade muito baixa, com capacidade de processamento muito inferior, tornando o processo mais lento.

Além disso, quando a carga de trabalho não é bem balanceada ou quando a distribuição de tarefas não leva em conta a enorme disparidade de poder de processamento entre os nós, pode ocorrer um congestionamento. Isso faz com que o processo 8 seja impactado significativamente. A execução de processos nesses nós será muito mais demorada, o que aumenta o tempo total de execução do trabalho, fazendo com que o desempenho global do cluster caia.

A seguir, utilizando a mesma versão de configuração dos clusters, mas, com 16 processos, obtem-se essa nova configuração:

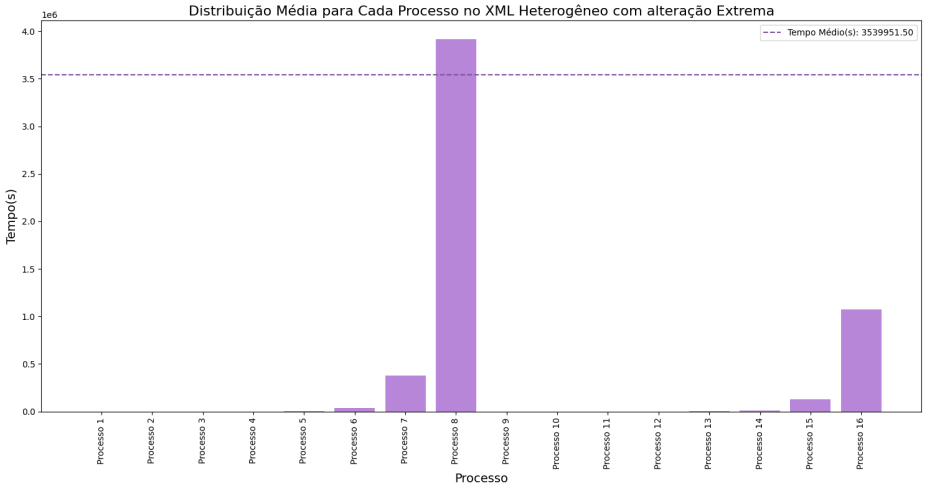


Figure 13: Para 16 Processos e Tamanho da Imagem 5000

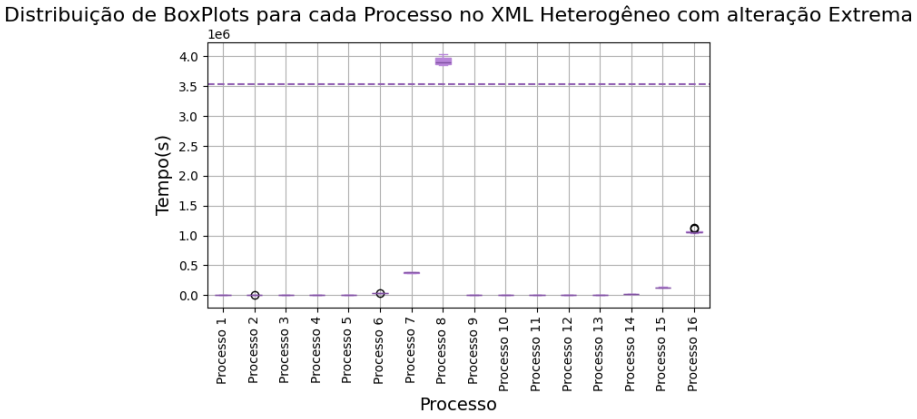


Figure 14: Para 16 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.106944	0.001897	0.104052	0.105281	0.106607	0.108937	0.109464
2	1.300564	0.055353	1.251349	1.269845	1.282698	1.304211	1.447135
3	13.526602	0.417689	13.110745	13.216475	13.371721	13.725230	14.439944
4	194.405379	5.405029	187.855220	189.586972	192.592768	198.614367	204.063867
5	2750.807203	63.396555	2677.885790	2700.329223	2733.597198	2782.207311	2882.220705
6	36179.549475	973.855966	35070.695500	35665.563400	35891.432775	36507.494488	38652.802500
7	379577.325600	5244.719078	370663.474500	374700.837500	381948.363000	383299.254125	386503.072000
8	3918726.197500	63716.725224	3851669.345000	3872001.192500	3896489.645000	3973057.207500	4033143.565000
9	0.393172	0.006773	0.382904	0.389074	0.391535	0.398466	0.403530
10	3.820244	0.076741	3.732471	3.771565	3.790637	3.877211	3.947401
11	35.606616	0.606476	34.789570	35.039569	35.621798	35.995709	36.802988
12	276.346145	5.640524	267.216716	271.697137	276.628481	280.810522	284.937718
13	1919.883358	40.888602	1859.939315	1884.960445	1920.827963	1955.340385	1975.316980
14	13477.903365	259.289068	13122.632650	13203.926250	13538.893175	13739.010375	13763.518000
15	128464.016100	3725.384470	123758.606500	125527.631250	128321.452750	130934.361750	135057.520000
16	1071405.170000	27414.667096	1044661.270000	1056395.768750	1060796.505000	1074560.583750	1134927.390000

Table 10: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.10590355057795135, 0.10798405142204866)
2	(1.270214528231125, 1.330913095768875)
3	(13.297589448815025, 13.755614511184977)
4	(191.44188442939196, 197.36887261060804)
5	(2716.047851718674, 2785.5665543213263)
6	(35645.599302641545, 36713.49964739846)
7	(376701.72720087797, 382452.92399916204)
8	(3883791.3019724283, 3953661.093027612)
9	(0.3894586149088956, 0.3968852930911044)
10	(3.778168199760671, 3.862319566239327)
11	(35.27409425250986, 35.93913700749014)
12	(273.2535332681855, 279.4387574718145)
13	(1897.464769634944, 1942.301946405056)
14	(13335.739177199619, 13620.06755284038)
15	(126421.44538104396, 130506.58681899605)
16	(1056374.1317985335, 1086436.2082015064)

Table 11: Distribuição dos Intervalos de Confiança, em segundos

Para 16 processos, os que tiverem menor impacto global foram o 1 (média 0.1s e desvio padrão expressivamente baixo 0.002s, indicando baixa dispersão dos dados amostrais) e o 9 (média 0.4s e desvio padrão 0.007s, denotando baixa variabilidade amostral). Enquanto os mais impactados globalmente foram o 8 (média de 3918726.2s (45 dias!), com desvio padrão expressivamente grande, 63716.72s (17 horas!)) e o 16 (1071405.17s (12 dias!), com desvio padrão relativamente alto, 27414.6s (7 horas!)).

A maior disparidade entre nós é de 4033143.46s (46 dias!).

Novamente, pode-se considerar como fator potencial o desbalanceamento de carga entre nós e o congestionamento causado em razão da enorme disparidade de poder de processamento entre os nós.

Finalmente, reorganizando os Gf em ordem decresecente de processamento, de 8 em 8 nós, fazendo uma redução na ordem de uma casa decimal. Por ilustração:

- Nó 1 : Speed 400Gf
- Nó 2 : Speed 400Gf
- Nó 3 : Speed 400Gf
- Nó 4 : Speed 400Gf
- Nó 5 : Speed 400Gf

- Nó 6 : Speed 400Gf
- Nó 7 : Speed 400Gf
- Nó 8 : Speed 400Gf
- Nó 9 : Speed 40Gf
- Nó 10 : Speed 40Gf
- ...
- Nó 16 : Speed 40Gf
- Nó 17 : Speedn 4Gf
- ...
- Nó 63 : Speed 0.00004Gf
- Nó 64 : Speed 0.00004Gf

Tem-se a seguinte distribuição:

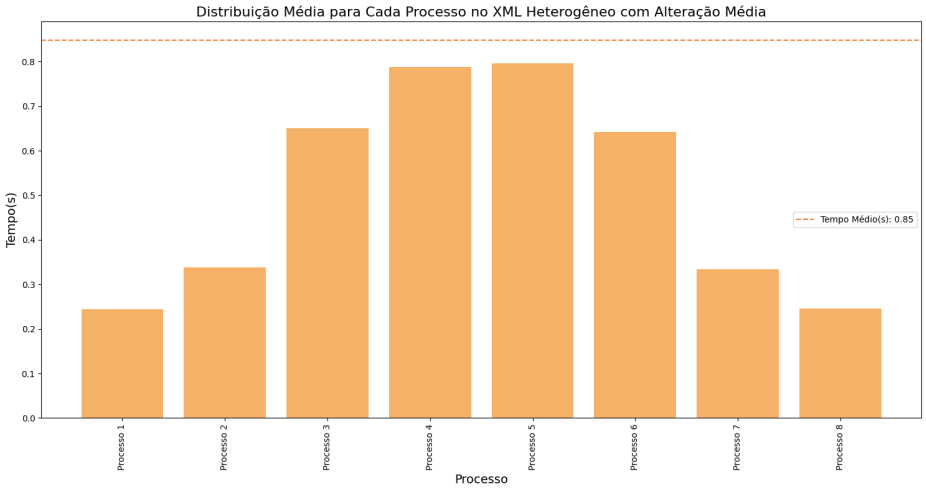


Figure 15: Para 8 Processos e Tamanho da Imagem 5000

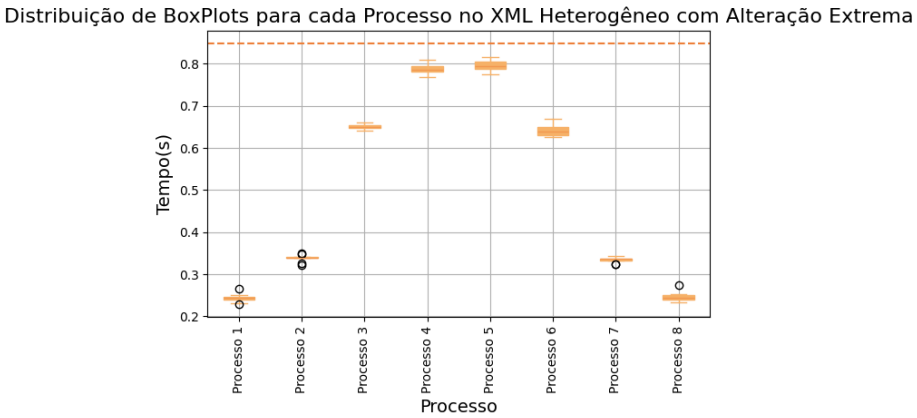


Figure 16: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.243995	0.009673	0.228662	0.240304	0.243432	0.247072	0.265567
2	0.338430	0.007932	0.322030	0.338315	0.339412	0.341826	0.349198
3	0.650318	0.005390	0.640963	0.647762	0.650641	0.653288	0.660922
4	0.787611	0.011257	0.767608	0.782131	0.786511	0.793508	0.809556
5	0.796388	0.013165	0.774965	0.788367	0.794610	0.804953	0.816732
6	0.641521	0.013315	0.625094	0.631119	0.638449	0.649861	0.668664
7	0.334591	0.005795	0.324079	0.333223	0.336210	0.337679	0.342435
8	0.246256	0.011278	0.232446	0.238882	0.244758	0.250423	0.273853

Table 12: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.23869104812850617, 0.24929857387149384)
2	(0.3340806193187789, 0.34277895468122116)
3	(0.6473623104889046, 0.6532727155110956)
4	(0.7814390807520727, 0.7937834832479272)
5	(0.7891705004610297, 0.8036064035389705)
6	(0.6342202849589218, 0.6488209450410782)
7	(0.33141323221384905, 0.3377683137861511)
8	(0.2400725763801312, 0.25243956161986886)

Table 13: Distribuição dos Intervalos de Confiança, em segundos

A distribuição de tempos entre os processos voltou a ser comparável com os exemplos anteriores. E os processos com menor impacto global foram os iniciais, 1 (média 0.24s, desvio padrão expressivamente baixo, 0.01s, indicando forte proximidade amostral dos dados) e os finais (média 0.25s, desvio padrão relativamente pequeno, 0.01s, denotando baixa variabilidade amostral). Enquanto os mais impactados foram os intermediários, 4 (com média 0.78s e desvio padrão pequeno, 0.01s) e 5 (com média 0.8s e desvio padrão baixo, 0.01s).

Ainda assim, a diferença entre os casos ficam abaixo de 0.6s. Dessa vez, houve um balanceamento melhor entre os nós, evitando-se sobrecarga e congestionamento.

3.2 Experimentos com Cluster Homogêneo

Por fim, realizou-se com alguns experimentos visando a setagem para clusters homogêneos. Ness ocasião, unificaram-se todos os valores do campo `speed` para 200Gf, alterando-se, dessa vez, a latência dos nós, a fim de se investigar seu impacto no desempenho global.

A priori, para uma setagem com a latência correspondente a 10µ, tem-se a seguinte distribuição entre os processos:

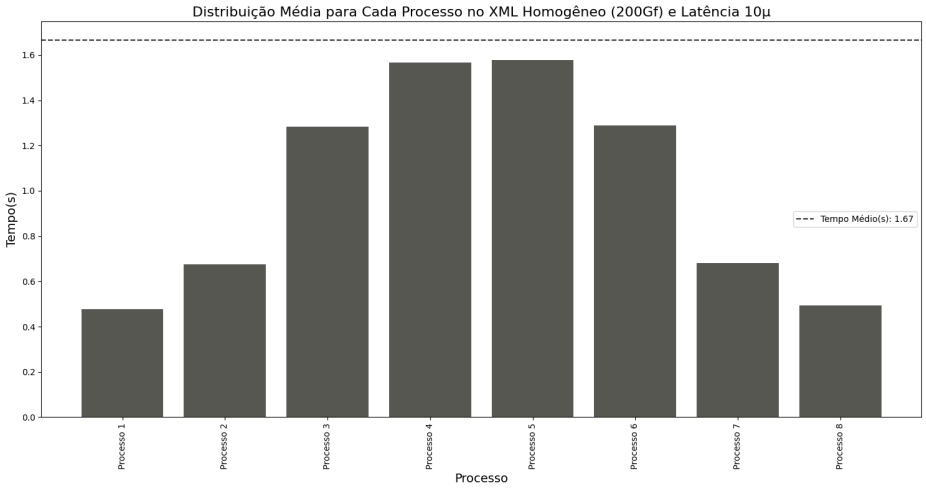


Figure 17: Para 8 Processos e Tamanho da Imagem 5000

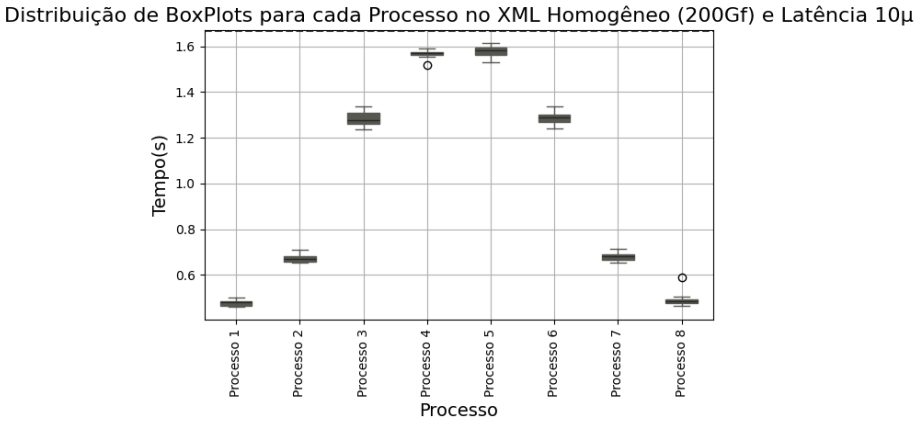


Figure 18: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.477767	0.013363	0.460434	0.464186	0.480244	0.486890	0.500893
2	0.675156	0.019962	0.653089	0.659334	0.669887	0.683624	0.710809
3	1.283698	0.031438	1.235312	1.261710	1.276564	1.308419	1.336270
4	1.566889	0.019730	1.517924	1.561290	1.569686	1.575915	1.591256
5	1.577254	0.024948	1.528866	1.562734	1.584296	1.595327	1.614280
6	1.288339	0.026893	1.242295	1.268820	1.287940	1.302357	1.335835
7	0.679871	0.019699	0.652094	0.664722	0.681169	0.689664	0.714171
8	0.493558	0.033640	0.464754	0.475179	0.483723	0.494687	0.588569

Table 14: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.47043954686993417, 0.48509353513006603)
2	(0.664210468296707, 0.6861005917032931)
3	(1.266460953225351, 1.3009347907746487)
4	(1.5560717693423585, 1.5777068786576418)
5	(1.5635751366629258, 1.590932863337075)
6	(1.2735937946539855, 1.303083305346015)
7	(0.669070389345355, 0.6906713166546449)
8	(0.4751130929169905, 0.5120019710830095)

Table 15: Distribuição dos Intervalos de Confiança, em segundos

Os valores, novamente, tendem a uma distribuição Normal, onde processos com menor impacto global são os iniciais, 1, (média 0.48s e desvio padrão baixo, 0.01s) e os finais, 8, (média 0.49s e desvio padrão baixo, 0.03s). Enquanto os de maior impacto são os intermediários, 4, (média 1.57s e desvio padrão 0.02s) e 5, (média 1.58s e desvio padrão baixo 0.02s).

Ainda assim, a disparidade máxima de processos é de, no máximo, 1.15s.

A posteriori, para uma latência correspondente a 100µ, obtem-se a distribuição por processo:

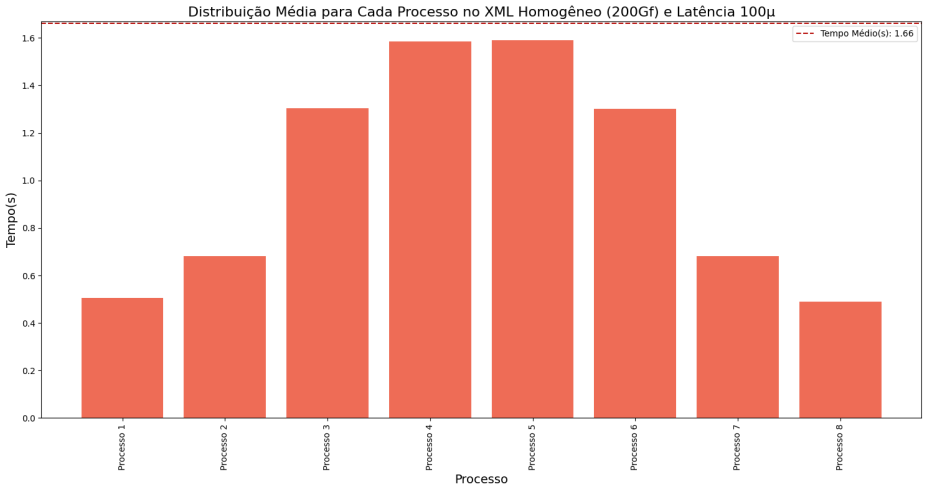


Figure 19: Para 8 Processos e Tamanho da Imagem 5000

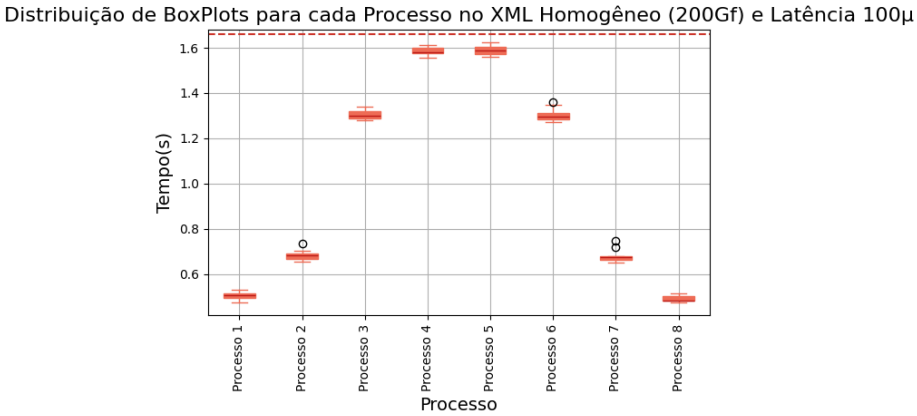


Figure 20: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.504287	0.016837	0.476206	0.492793	0.504405	0.513452	0.530649
2	0.682601	0.021999	0.655264	0.666859	0.682169	0.688777	0.733168
3	1.304724	0.019354	1.278641	1.288931	1.301052	1.318729	1.338527
4	1.584112	0.016967	1.556573	1.574512	1.578437	1.599152	1.610916
5	1.590017	0.021690	1.561474	1.571092	1.588905	1.605298	1.623322
6	1.302428	0.028575	1.269979	1.285468	1.294450	1.311091	1.361254
7	0.680849	0.027487	0.651960	0.664078	0.673996	0.677462	0.746730
8	0.490599	0.013441	0.473729	0.480385	0.483898	0.503370	0.513321

Table 16: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.49505571013434896, 0.5135189618656512)
2	(0.6705388287367985, 0.6946626412632017)
3	(1.2941118896610075, 1.315335220338993)
4	(1.5748095113755884, 1.5934146166244114)
5	(1.5781251663680493, 1.601909331631951)
6	(1.286760541356509, 1.3180948566434907)
7	(0.6657784750613668, 0.695919572938633)
8	(0.483229280351742, 0.49796792564825804)

Table 17: Distribuição dos Intervalos de Confiança, em segundos

Os valores, novamente, tendem a uma distribuição Normal, onde processos com menor impacto global são os iniciais, 1, (média 0.5s e desvio padrão baixo, 0.02s) e os finais, 8, (média 0.49s e desvio padrão baixo, 0.01s). Enquanto os de maior impacto são os intermediários, 4, (média 1.58s e desvio padrão 0.02s) e 5, (média 1.59s e desvio padrão baixo 0.02s).

Ainda assim, a disparidade máxima de processos é de, no máximo, 1.15s.

Enfim, para uma latência correspondente a 500µ, obtem-se a distribuição por processo:

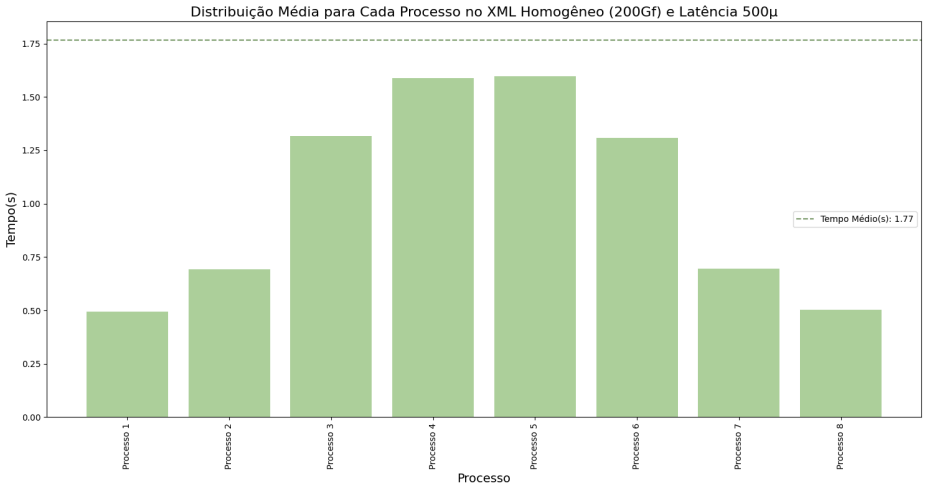


Figure 21: Para 8 Processos e Tamanho da Imagem 5000

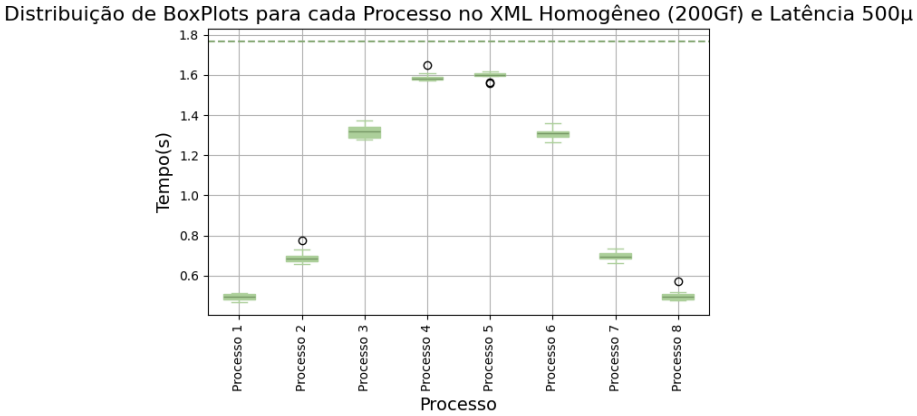


Figure 22: Para 8 Processos e Tamanho da Imagem 5000

Processo	Média	Desvio Padrão	Mínimo	Q1	Mediana	Q3	Máximo
1	0.494054	0.016536	0.467074	0.480051	0.494084	0.509738	0.514780
2	0.693945	0.034626	0.655747	0.670333	0.686306	0.698712	0.777598
3	1.317789	0.031058	1.277808	1.287988	1.317933	1.339433	1.373905
4	1.589553	0.022180	1.573098	1.575096	1.582090	1.588575	1.648606
5	1.595709	0.019379	1.559696	1.593418	1.599397	1.607808	1.619623
6	1.307986	0.025480	1.265086	1.289242	1.310291	1.318432	1.357959
7	0.695031	0.022740	0.661922	0.683533	0.693834	0.711335	0.736611
8	0.502619	0.026437	0.477864	0.483122	0.495290	0.510536	0.571236

Table 18: Distribuição das Métricas para cada Processo, em segundos

Processo	IC
1	(0.4849878475057742, 0.5031206164942257)
2	(0.674959813994259, 0.7129293940057411)
3	(1.3007606635837587, 1.3348180444162414)
4	(1.5773917585113528, 1.601713827488647)
5	(1.5850833906142392, 1.6063336993857606)
6	(1.2940157228574156, 1.3219566231425848)
7	(0.6825634863166269, 0.7074993276833729)
8	(0.48812387486044007, 0.5171142691395602)

Table 19: Distribuição dos Intervalos de Confiança, em segundos

Os valores, novamente, tendem a uma distribuição Normal, onde processos com menor impacto global são os iniciais, 1, (média 0.49s e desvio padrão baixo, 0.02s) e os finais, 8, (média 0.5s e desvio padrão baixo, 0.03s). Enquanto os de maior impacto são os intermediários, 4, (média 1.59s e desvio padrão 0.02s) e 5, (média 1.6s e desvio padrão baixo 0.02s).

Ainda assim, a disparidade máxima de processos é de, no máximo, 1.23s.

Verifique que, em razão dos clusters serem homogêneos, houve um bom balancemento entre os clusters, mantendo a concisão do sistema e evitando congestionamentos.

Por outro lado, mesmo com latências variadas (10µ, 100µ e 500µ), os valores se mantiveram próximos entre si, também, em razão das barreiras implementadas no interior do código, para garantir que todos os processos escrevam corretamente no arquivo final e evitando problemas de concorrência.

4 Conclusão

Este trabalho visa a análise sobre sistemas distribuídos sobre a paralelização da computação do conjunto de Julia.

A priori, analisou-se o impacto da quantidade de processos e discutiu-se sobre a não linearidade entre esse valor e o melhor tempo de execução. Nessa etapa, percebeu-se que, 6, 7 a 8 processos mantém uma eficiência ótima para o programa `1D_parallel_julia`. Ademais, constatou-se um speedup ótimo estabilizado em 3, para tamanhos entre 1300 e 2100. Mas, quanto mais se avança, essa métrica vai decaindo. Os maiores problemas discutidos foram a subrecarga de comunicação, saturação dos recursos do sistema e sincronização, além da ociosidade de processos.

Quanto aos experimentos no `SimGrid`, verificou que, para clusters heterogêneos, a disparidade entre grandezas de `speed` é um ponto importante a se discutir, uma vez que, se as cargas não forem bem balanceadas, podem levar a uma sobrecarga e congestionamentos, acarretando processos que impactam negativamente o desempenho global. Por outro lado, quanto às latências em clusters homogêneos, em razão do balanceamento e das barreiras inclusas no programa para evitar problemas de concorrência, não tiveram muitas diferenças entre si (10µ, 100µ e 500µ) quanto ao desempenho geral.