

PROGRAMAÇÃO PARALELA

NATAN LUIZ PAETZHOLD BERWALDT
TRABALHO 4

IMPLEMENTAÇÃO 1

A implementação contida no arquivo *fractalpar1.cpp* visa dividir o trabalho da criação do fractal **Distribuindo UM frame para cada thread** executar por vez.

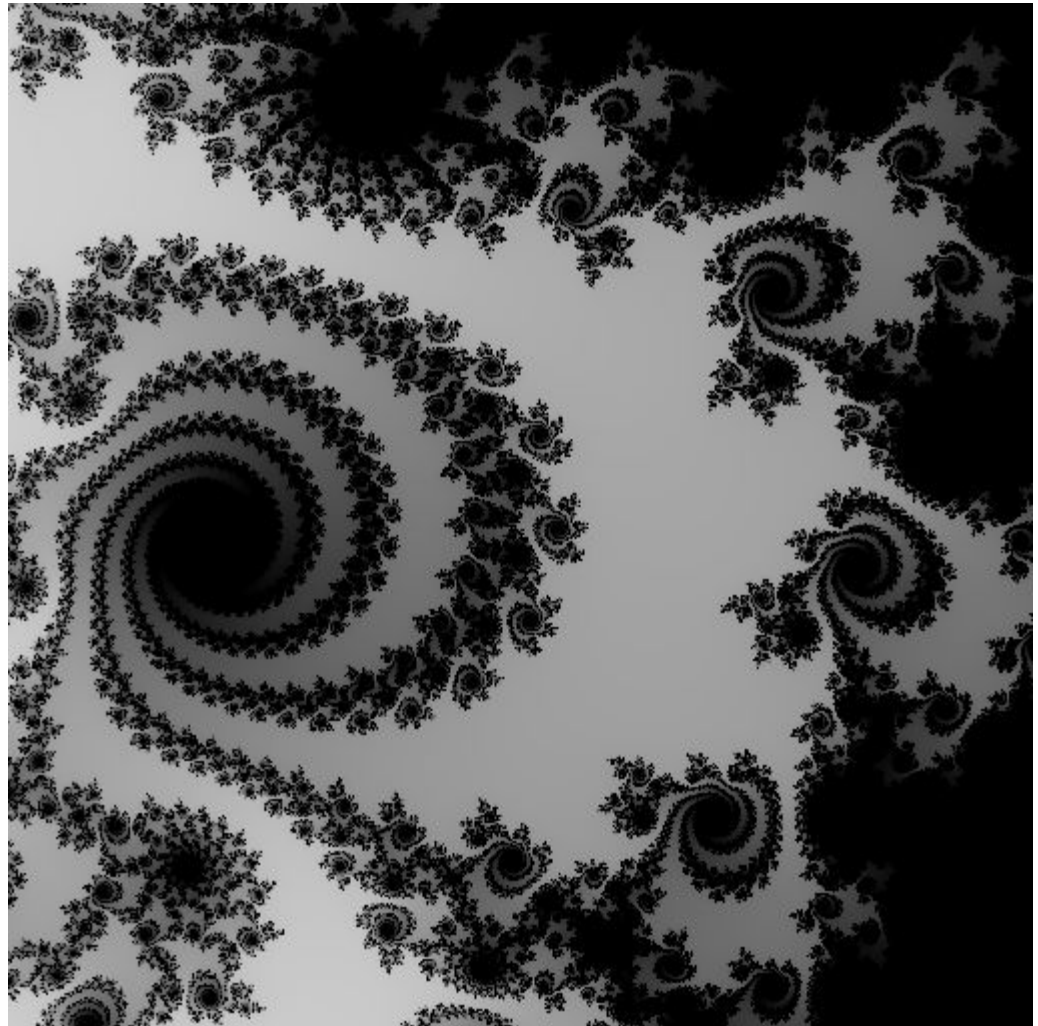
Ou seja, usa-se Schedule DYNAMIC com CHUNK 1.

```
#pragma omp parallel for schedule(dynamic) num_threads(NTH)
```

PROBLEMAS ??

Programa executado 1024x1024

com 4 threads, dynamic com chunk 4.



PROBLEMAS ??

Cada frame difere do anterior pela mudança da variável compartilhada Delta.

```
84     }  
85     delta *= 0.98;  
86 }
```

Logo, se eu acabar executando o frame 10 antes do frame 9, a imagem que deveria vir depois acaba aparecendo antes, e assim passando a sensação de travamento vista antes.

File Edit View Search Terminal Help

computing 100 frames of 128 by 128 fractal

```
Frame 0 ,Delta 0.001
Frame 8 ,Delta 0.00098
Frame 12 ,Delta 0.0009604
Frame 4 ,Delta 0.000941192
Frame 1 ,Delta 0.000922368
Frame 13 ,Delta 0.000903921
Frame 9 ,Delta 0.000885842
Frame 5 ,Delta 0.000868126
Frame 2 ,Delta 0.000850763
Frame 14 ,Delta 0.000833748
Frame 6 ,Delta 0.000817073
Frame 10 ,Delta 0.000800731
Frame 15 ,Delta 0.000784717
Frame 3 ,Delta 0.000769022
Frame 7 ,Delta 0.000753642
Frame 11 ,Delta 0.000738569
Frame 16 ,Delta 0.000723798
Frame 24 ,Delta 0.000709322
Frame 20 ,Delta 0.000695135
Frame 28 ,Delta 0.000681233
Frame 25 ,Delta 0.000667608
Frame 17 ,Delta 0.000654256
Frame 21 ,Delta 0.000641171
Frame 29 ,Delta 0.000628347
Frame 18 ,Delta 0.00061578
Frame 26 ,Delta 0.000603465
Frame 30 ,Delta 0.000591395
Frame 22 ,Delta 0.000579568
Frame 27 ,Delta 0.000567976
```

Execução de 100 frames
128 x 128 com 4 threads,
DYNAMIC com CHUNK 4

SOLUÇÃO

No início do laço de cada frame calcula qual é o valor de delta em relação ao frame que será calculado.

```
62      // Substitui delta *= 0.98;  
63      double delta = Delta;  
64      for(int i = 0; i < frame; i++){  
65          delta = delta * 0.98;  
66      }
```

Implementação 2

A implementação contida no arquivo *fractalpar2.cpp* visa dividir o trabalho da criação do fractal **Executando um frame por vez** e dentro da execução do frame **Dividir um conjunto de pixels (rows) para cada thread**.

Ou seja, usado Schedule STATIC.

```
#pragma omp parallel for schedule(static) num_threads(NTH)
```



THREAD 0

THREAD 1

THREAD 2

THREAD 3

Esquemática de Imagem
estática 1024x1024 dividida
artificialmente entre 4 threads.

TESTES

Os testes feitos em ambas as implementações foram:

Tamanhos: 256x256 | 1024x1024 | 2048x2048

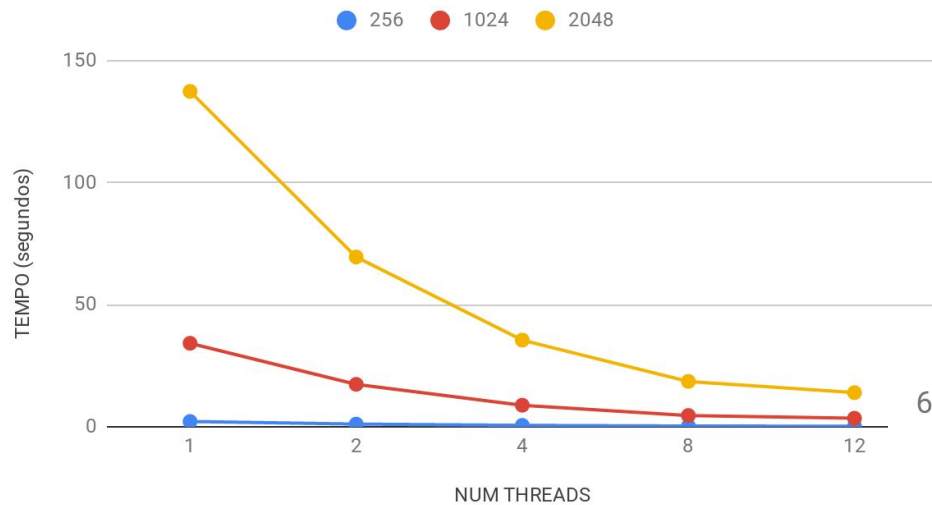
Numero de frames: 32 | 64

Numero de Threads: 1 | 2 | 4 | 8 | 12

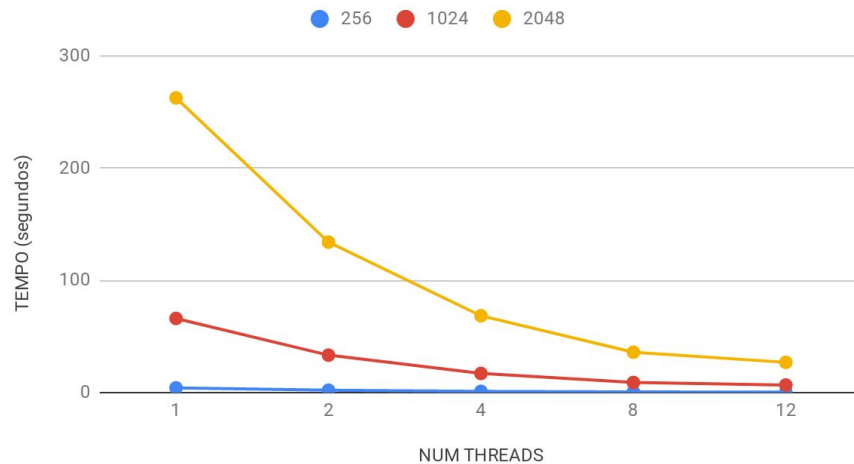
RESULTADOS IMPLEMENTAÇÃO 1

32 FRAMES					Num de Threads	
		1	2	4	8	12
	256	2,1548	1,1021	0,5686	0,2957	0,2195
TAMANHO	1024	34,1846	17,3802	8,8228	4,6172	3,4817
	2048	137,3185	69,5134	35,4761	18,5665	13,9763
64 FRAMES					Num de Threads	
		1	2	4	8	12
	256	4,154	2,1035	1,0695	0,5637	0,4258
TAMANHO	1024	66,0014	33,3241	17,0164	8,9574	6,7049
	2048	262,4032	133,9573	68,362	35,9508	26,9047

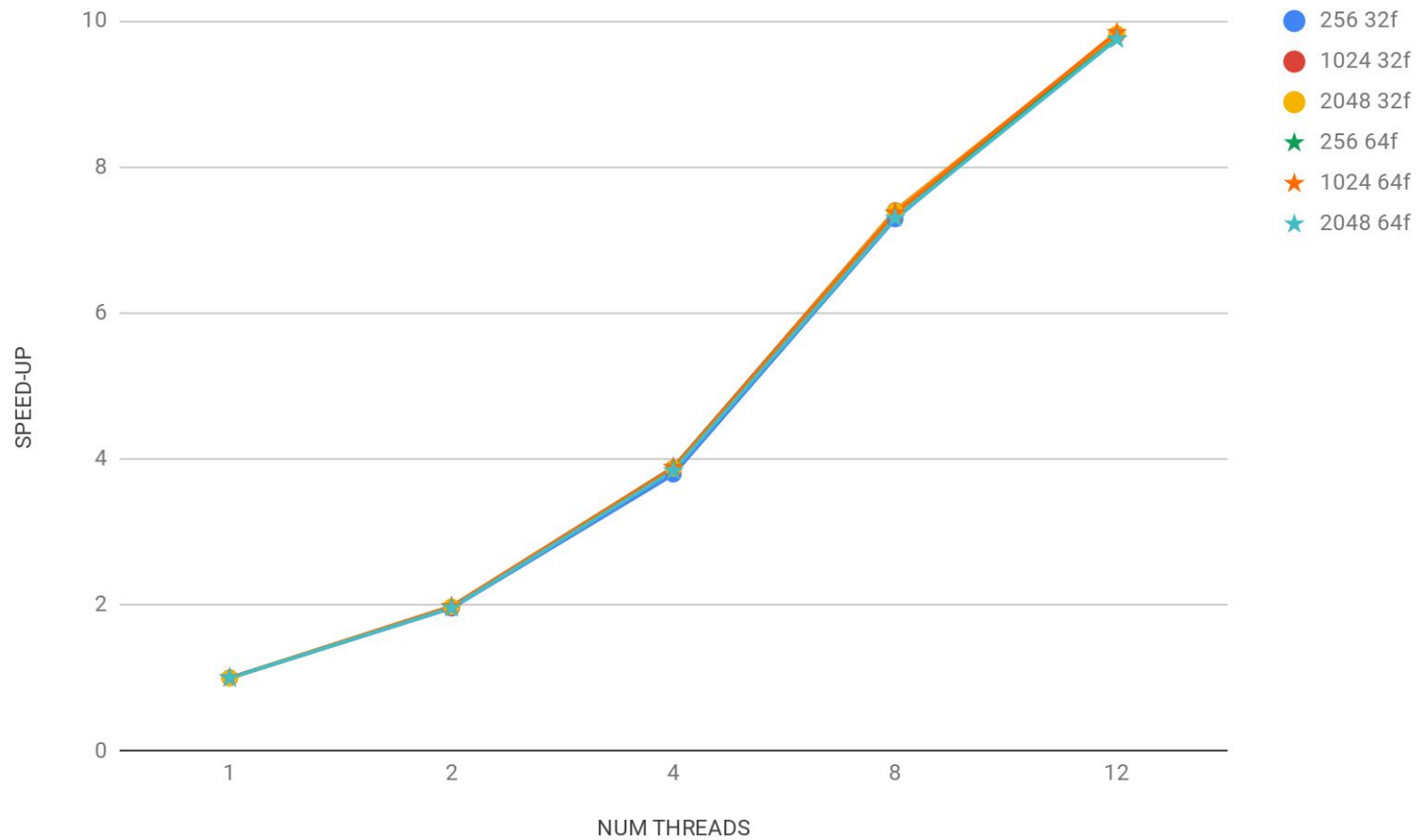
32 FRAMES



64 FRAMES

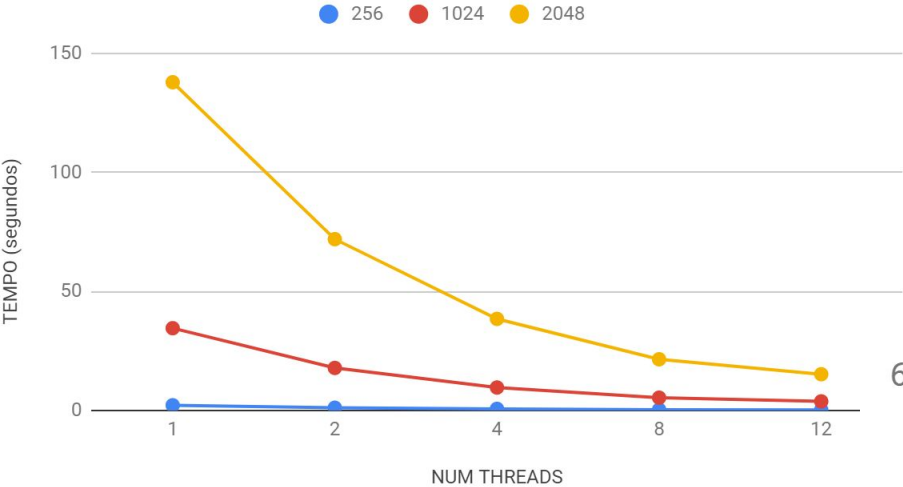


SPEED-UP

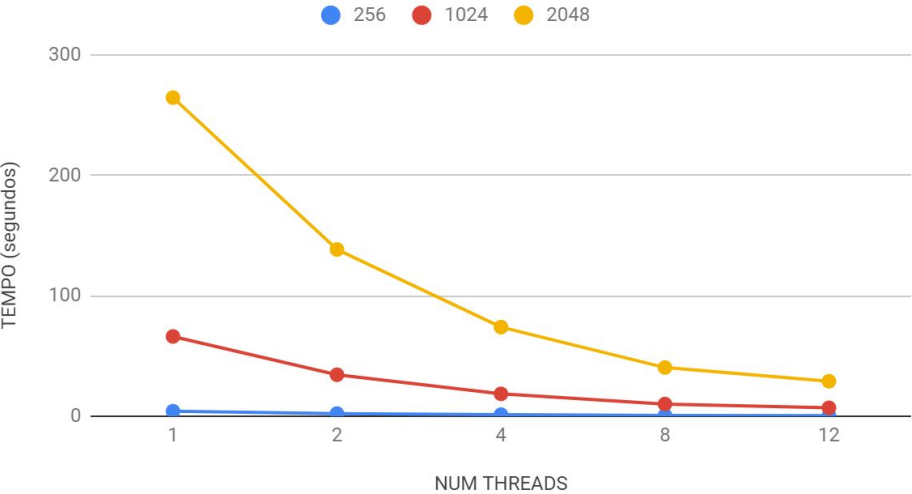


32 FRAMES				Num de Threads		
		1	2	4	8	12
	256	2,1802	1,1428	0,6755	0,342	0,2347
TAMANHO	1024	34,537	17,8469	9,6268	5,3532	3,8075
	2048	137,7758	71,8989	38,4403	21,4769	15,2001
64 FRAMES				Num de Threads		
		1	2	4	8	12
	256	4,1364	2,1637	1,2785	0,6437	0,4368
TAMANHO	1024	66,1501	34,3972	18,5135	10,0359	6,9682
	2048	264,3444	138,3624	73,937	40,3733	28,9907

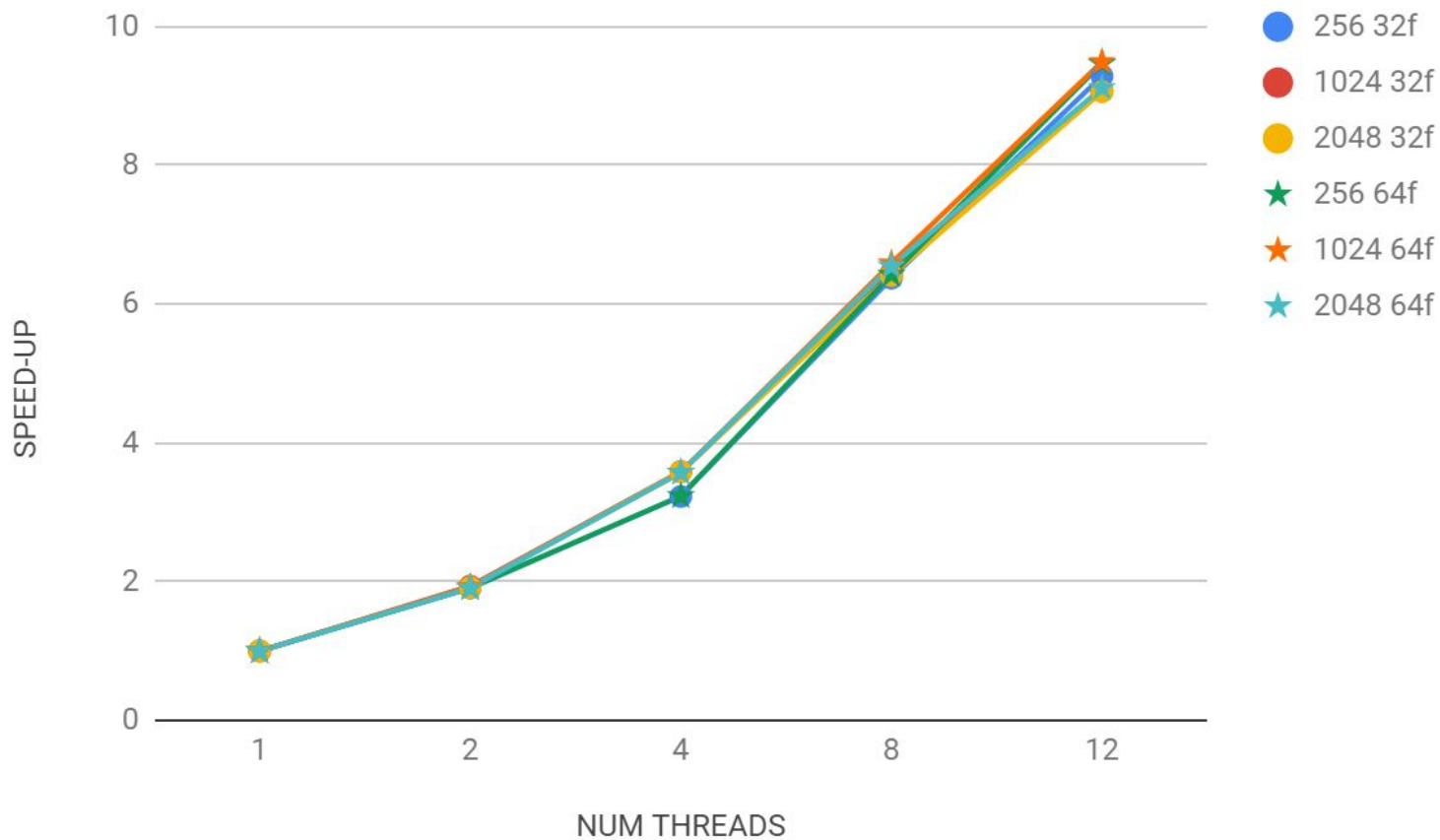
32 FRAMES



64 FRAMES



SPEED-UP



COMPARAÇÃO DE SPEEDUPS

SPEED-UP

