

EP1 - Cálculo do Conjunto de Mandelbrot em Paralelo com Pthreads e OpenMP

Lucas de Sousa Rosa, Alfredo Goldman

12 de setembro de 2024

Motivação

O termo fractal foi criado em 1975 por Benoît Mandelbrot para descrever um objeto geométrico que pode ser dividido em partes, cada uma das quais semelhante ao objeto original. Benoît trabalhava na IBM durante a década de 1960 e foi um dos primeiros a usar computação gráfica para mostrar como complexidade pode surgir a partir de regras simples.

Um desses fractais foi nomeado *Conjunto de Mandelbrot* pelo matemático Adrien Douady. O Conjunto de Mandelbrot pode ser informalmente definido como o conjunto dos números complexos c para os quais a função $f_c(z) = z^2 + c$ não diverge quando é iterada começando em $z = 0$. Isto é, a sequência $f_c(0), f_c(f_c(0)), f_c(f_c(f_c(0))), \dots$ é sempre limitada. A Figura 1 mostra uma região do Conjunto de Mandelbrot conhecida como *Seahorse Valley*.

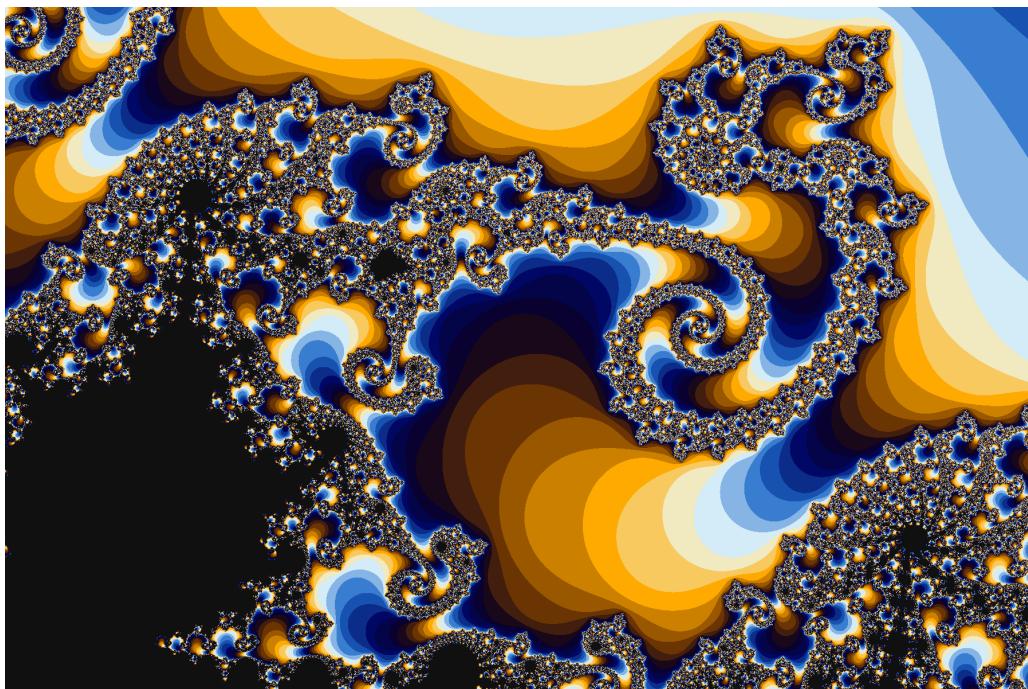


Figure 1: Seahorse Valley

As cores na Figura 1 indicam o número da iteração onde a função $f_c(z)$ convergiu. A Figura 1 foi gerada usando o código fonte fornecido junto com este documento. O código na linguagem C e os arquivos auxiliares estão inclusos no arquivo `ep1.zip` disponível no e-Disciplinas.

Tarefa

Você deverá paralelizar o código para o cálculo do Conjunto de Mandelbrot usando a biblioteca Pthreads e as diretivas de compilador fornecidas pelo OpenMP. Depois, você deverá medir o tempo de execução das versões sequencial, paralelizada com Pthreads e paralelizada com OpenMP. Você deverá medir o tempo de execução para diferentes tamanhos de entrada e números de *threads* nas versões paralelizadas, em quatro regiões do Conjunto de Mandelbrot, como descrito na seção de experimentos.

A implementação necessária para paralelizar o código fornecido usando Pthreads e OpenMP é relativamente simples. A parte mais trabalhosa do EP1 é a elaboração de um relatório que apresente gráficos com os resultados obtidos com as paralelizações, e os discuta de forma metódica.

Paralelização com Pthreads e OpenMP

O código fornecido está triplicado, e cada arquivo .c tem um propósito diferente:

- `mandelbrot_seq.c`: não precisa ser paralelizado
- `mandelbrot_pth.c`: deve ser paralelizado com Pthreads
- `mandelbrot_omp.c`: deve ser paralelizado com OpenMP.

Você pode modificar à vontade os arquivos `mandelbrot_omp.c` e `mandelbrot_pth.c`, desde que eles continuem produzindo a mesma saída produzida por `mandelbrot_seq.c`. Você também pode modificar o arquivo sequencial, dado que a saída continue correta.

Experimentos

Para cada uma das três versões do programa, você deverá realizar medições do tempo de execução para diferentes tamanhos de entrada. Nas versões paralelizadas você deverá também medir, para cada tamanho de entrada, o tempo de execução para diferentes números de *threads*.

Você deve fazer um número de medições e analisar a variação dos valores obtidos. Sugerimos 10 medições para cada experimento, e também que você use a média e o intervalo de confiança das 10 medições nos seus gráficos. Caso observe variabilidade muito grande nas medições, resultando num intervalo de confiança muito grande, você pode realizar mais medições, sempre apresentando a média e o intervalo de confiança. **Não é recomendado** fazer menos de 10 medições.

Você deve analisar também o impacto das porções não paralelizáveis do código sequencial: as operações de I/O e alocação de memória. Uma vez que você verifique que as versões paralelizadas produzem o resultado correto, elas **não precisam** realizar I/O e alocação de memória nos testes de desempenho, pois esses custos são fixos e assim aceleramos os experimentos.

A Tabela 1 lista os experimentos que devem ser feitos: os valores para o número de *threads* e de execuções, e os tamanhos de entrada. Cada experimento deverá ser repetido nas quatro regiões especificadas na Figura 6. As coordenadas para cada região podem ser obtidas executando no diretório:

```
$ make mandelbrot_seq
gcc -o mandelbrot_seq -std=c11 mandelbrot_seq.c
$ ./mandelbrot_seq
usage: ./mandelbrot_seq c_x_min c_x_max c_y_min c_y_max image_size
examples with image_size = 11500:
Full Picture:      ./mandelbrot_seq -2.5 1.5 -2.0 2.0 11500
Seahorse Valley:   ./mandelbrot_seq -0.8 -0.7 0.05 0.15 11500
Elephant Valley:   ./mandelbrot_seq 0.175 0.375 -0.1 0.1 11500
Triple Spiral Valley: ./mandelbrot_seq -0.188 -0.012 0.554 0.754 11500
```

O número de iterações para o critério de convergência foi escolhido¹ de forma a produzir uma imagem interessante em diferentes níveis de magnificação, mas ter um tempo de execução razoável para tamanhos grandes de entrada.

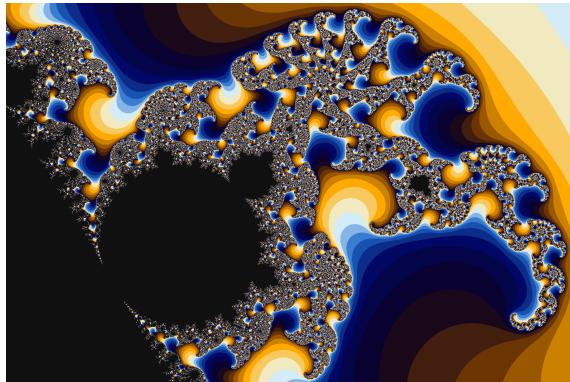


Figure 2: *Elephant Valley*

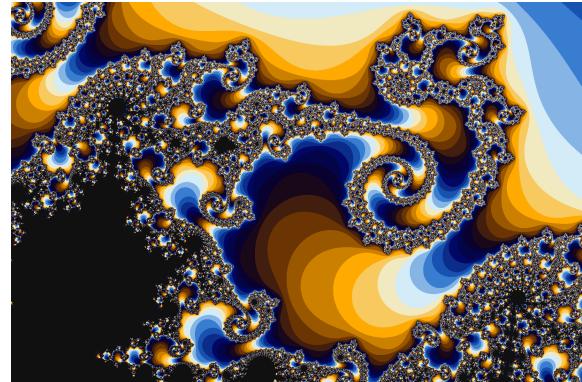


Figure 3: *Seahorse Valley*

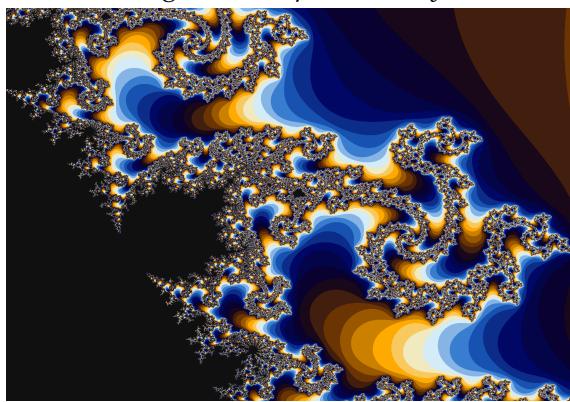


Figure 4: *Triple Spiral Valley*

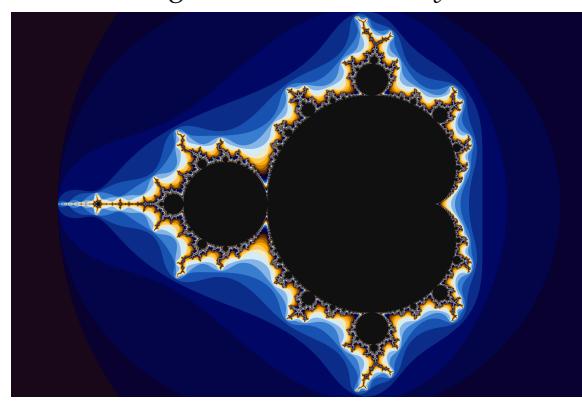


Figure 5: *Full Picture*

Figure 6: Regiões do Conjunto de Mandelbrot

	Pthreads	OpenMP	Seqencial
Regiões	<i>Triple Spiral, Elephant, Seahorse & Full</i>		
I/O e Aloc. Mem.	Sem	Com e sem	
Número de <i>Threads</i>	$2^0 \dots 2^5$		-
Tamanho da Entrada		$2^4 \dots 2^{13}$	
Número de Execuções		10	

Table 1: Experimentos

Apresentação e Discussão dos Resultados

Depois de realizar os experimentos você deverá elaborar gráficos que evidenciem o comportamento das três versões do programa com relação à variação dos parâmetros descritos na Tabela 1. Os gráficos deverão ser claros e legíveis, com eixos nomeados. Deverão apresentar a média e o intervalo de confiança das 10 execuções para cada cenário experimental.

¹<https://goo.gl/WpL9hs>

Você deverá analisar os resultados obtidos e tentar responder a algumas perguntas:

- Como e por que as três versões do programa se comportam com a variação:
 - Do tamanho da entrada?
 - Das regiões do Conjunto de Mandelbrot?
 - Do número de *threads*?
- Qual o impacto das operações de I/O e alocação de memória no tempo de execução?

Você consegue pensar em mais perguntas interessantes?

Entrega

Você deverá entregar no site do curso **um único arquivo**, no formato `tar/tgz/gz/zip/rar`, contendo:

- Um arquivo `.pdf` com as análises e gráficos
- As três versões do código fonte usado
- Um arquivo `.csv` com as medições feitas

Mais detalhes

O arquivo `run_measurements.sh` contém o necessário para gerar as medições de tempo de execução das três versões do programa, mas você vai precisar modificá-lo para medir o impacto das operações de I/O e alocações de memória.

Um dos comandos executados por `run_measurements.sh`, expandindo as variáveis, é:

```
$ perf stat -r 10 ./mandelbrot_seq -2.5 1.5 -2.0 2.0 512
```

O comando acima executa dez repetições da versão sequencial do programa, calculando a imagem inteira do Conjunto de Mandelbrot com um tamanho de imagem de 512 *pixels*. Modificar os parâmetros desse comando será suficiente para realizar os experimentos do EP1.

Agradecimentos

Este EP foi baseado em EPs de anos anteriores criados e modificados pelo Diogo Pina e Pedro Henrique R. Bruel.