Geração de Fratais com OpenMP

C

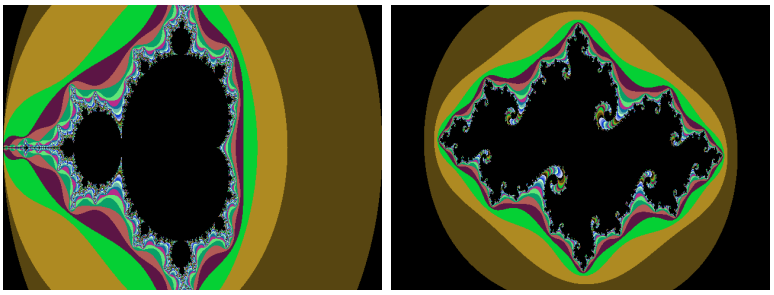

Figura C.1: Left: Julia fractal; Right: Mandelbrot fractal

C.1 Fractais

A palavra Fractal foi criada por Benoît Mandelbrot, e significa irregular ou quebrado, por um lado, e fragmento, por outro. Um Fractal é gerado a partir de uma fórmula matemática, muitas vezes simples, mas que aplicada de forma iterativa, produz resultados graficamente apelativos. Existem duas categorias de Fractais: os geométricos, que repetem continuamente um modelo padrão. E os aleatórios, que são feitos através dos computadores.

Uma das principais características dos fratais, que são estruturas geométrica, é a auto-semelhança, que pode ser descrita como a característica de que a estrutura do fratal ser semelhante em diversas escalas. Dito por outras palavras, a auto-semelhança é a simetria ao longo das escalas. Esta auto-semelhança pode ser exata ou estatística.

Nesta ficha é fornecido o código para gerar fratais e gravá-las sob a forma de ficheiros imagem do tipo PGM, como os da figura C.1. Este trabalho foi pensado para ser realizado em linux, mas se pretender visualizar no windows esses ficheiros pode começar por convertê-los para o formato PNG ou BPM usando um utilitário como o convert do pacote ImageMagick disponível, para windows, linux ou macos.

```
$ convert julia.pgm julia.png
```

C.2 Trabalho a realizar

Parte I

Exercício C.1

Parta do código fornecido para geração de imagens de fractais, quer do fractal Julia, quer do fratal Mandelbrot. Compile-o e execute-o. Verifique como os tempos de execução se podem tornar extremamente longos quando se geram imagens de resolução considerável. Veja o exemplo seguinte.

```
$ make
$ ./fractal 10000 10000
```

Para os testes iniciais, poderá usar uma resolução de imagem baixa (HD, por exemplo), mas para os testes finais e a escrita do relatório do trabalho deverá usar uma resolução de 4K (3840x2160), tendo o cuidado de adaptar o valor das iterações necessárias para o detalhe atingir o pixel.

Exercício C.2

Repare que a geração do fractal é evolutiva, sendo o número de iterações importante para criar uma imagem cada vez mais detalhada (cujo detalhe, e portanto o número de iterações necessárias, são função da resolução da imagem).

Crie uma versão modificada do código para gravar uma imagem (formato pgm) por iteração, no formato fractal_iter.pgm (ex: julia_35.pgm), mostrando a evolução da construção do fratal "ao longo do tempo".

Converta o conjunto de imagens num vídeo de formato mp4, usando a aplicação **ffmpeg** da seguinte forma:

```
$ ffmpeg -framerate 25 -pattern_type glob -i 'julia_*.pgm'
-c:v libx264 -r 30 -pix_fmt yuv420p out_julia.mp4
```

Note que para a geração do vídeo deverá ter instalado os pacotes ImageMagic e ffmpeg.

Visualize o vídeo gerado e repare na evolução do fratal. Faça a medição exata do tempo de geração das várias imagens do fratal. Este tempos serão usados como baseline para as melhorias a introduzir com o OpenMP.

Os vídeos devem ter uma duração mínima (30 segundos) para ser comparáveis, com e sem otimização pelo OpenMP. 20%

Exercício C.3

Crie uma versão modificada deste código que usando OpenMP distribua o cálculo pelos processadores disponíveis. Deve modificar apenas a função "void Generate(int)" sem alterar as funções julia e mandelbrot. 25%

Exercício C.4

Faça um estudo detalhado sobre o impacto de cada uma das diretivas do OpenMP no tempo de execução global, calculando o speedup de cada alteração e fazendo uma análise crítica de cada uma das alerações, bem como da solução com o melhor speedup global. 25%

Parte II

Exercício C.5

Partindo da matriz da imagem fractal gerada vamos simular a difusão das cores ao longo do tempo. Assim teremos de ir sucessivamente gerando novas imagens em que a imagem I_{k+1} é gerada a partir da I_k por difusão. Assim para um pixel de coordenadas i,j, o seu valor é dado por

$$\begin{aligned} I_{k+1}(i,j) &= (1-\alpha)I_k(i,j) + \alpha\frac{1}{8}\left[I_k(i-1,j-1) + I_k(i-1,j) + \right. \\ &+ I_k(i-1,j+1) + I_k(i,j-1) + I_k(i,j+1) + \\ &+ \left. I_k(i+1,j-1) + I_k(i+1,j) + I_k(i+1,j+1)\right], \end{aligned}$$

onde α é o parâmetro que controla a difusão, pesando a contribuição que o píxel na mesma posição tem versus os 8 vizinhos. Os valores extremos levam a que não haja difusão ($\alpha = 0$) ou a difusão seja muito rápida pelos vizinhos ($\alpha = 1$).

No final de ser gerada cada nova imagem, esta deve ser guardada num ficheiro cujo nome será da forma "output%04d.jpg"(ver função sprintf), onde %04d será substituido pelo numero de ordem do ficheiro.

Introduza o código para efetuar a difusão na zona indicada da função difusao(...) tirando partido de OpenMP para acelerar o processo. Note que para ativar a chamada da função de difusão deverá passar dois parâmetros adicionais que são o número de épocas (iterações) de difusão e o fator alfa que deverá estar entre 0 e 1.

```
$ make
$ ./fractal 3840 2160 100 0.5
```

30%

Exercício C.6

Elabore um breve relatório onde descreverá as abordagens e compare os tempos de execução com e sem OpenMP. **Este relatório deverá ser submetido juntamente com os ficheiros fonte num arquivo do tipo ZIP**.