Geração de Fractais com OpenMP

Arquitetura de Computadores 2022/2023

Daniela Fernandes
20192334632

João Vasco
2019236378

Manuel Santos
2019231352

Exercício C.1

Na alínea 1 foram implementados e testados os códigos fornecidos com o objetivo de analisar e compreender o processo de geração de imagens de fractais. Ao analisar o tempo de execução, foi possível concluir que estes se podem tornar longos para imagens com resoluções consideráveis. As diferentes resoluções testadas foram as seguintes: 8k (7680 x 4320), 4k (3840 x 2160), full HD (1920 x 1080) e HD (1280 x 720).

Exercício C.2

Tal como pedido no enunciado, nesta alínea foi criada uma versão modificada do código para gravar uma imagem em formato *pgm* por cada iteração. Deste modo, é possível obter uma representação gráfica da geração do fractal "ao longo do tempo". Foi, deste modo, desenvolvida a função *void saveimg(int *img, int rx, int ry, char *fname),* que recebe um ponteiro para o array dos pixels, o tamanho da imagem e o nome do ficheiro. Após todas as imagens terem sido geradas, estas são convertidas num vídeo de formato mp4 com recurso à aplicação *ffmpeg,* como sugerido no enunciado.

Neste exercício é também medido o tempo de geração dos fractais, estando os resultados obtidos explicitados na *Tabela* 1***Error! Reference source not found.***.

Exercício C.3

Na sequência da alínea anterior, o código desenvolvido foi modificado para que fosse possível tirar partido de OpenMP de modo que o cálculo fosse distribuído pelos vários processadores disponíveis, acelerando assim o processo de geração dos fractais. Assim, foi definida uma secção paralela através do uso de diretivas do compilador (as diretivas *parallel* e *for*, neste caso). Estas diretivas permitem criar uma "equipa" de fios, sendo que apenas são tidas em conta caso o programa seja compilado com suporte OpenMP.

Os resultados obtidos, à semelhança da alínea anterior, encontram-se também representados na *Tabela* 1***Error! Reference source not found.***.

Exercício C.4

	Tempo de execução	
Fractal	Sem OpenMP	Com OpenMP
Julia	151.28 Segundos	1.05 Segundos
MandelBrot	148.54 Segundos	0.91 Segundos
Total (comando time)	335 Segundos	4.815 Segundos

Tabela 1: Tempos de geração dos fractais (i5-7500, 4 Threads, 3840x2160)

Com base nos resultados obtidos nas alíneas *C2* e *C3* (representados na tabela anterior) é possível verificar que ao tirar partido do OpenMP, o processo de geração dos fractais acelera significativamente. Isto deve-se ao facto de o recurso da diretiva *for* permitir paralelizar a execução de ciclos e distribuir o cálculo pelos diversos processadores disponíveis, sendo o resultado desta divisão refletido num speedup global.

As diretivas *critical* e *reduction* não foram implementadas pois considera-se que o seu funcionamento não está de acordo com o propósito do exercício (definir um bloco de código que só pode ser executado por um fio de cada vez e reduzir os resultados de vários locais num único resultado, respetivamente).

Exercício C.5

O objetivo da alínea 5 consistia na simulação da difusão das cores de uma imagem de um fractal ao longo do tempo, onde cada imagem nova é criada com base nas cores presentes na imagem anterior por difusão com recurso à seguinte expressão:

$$I_{k+1}(i,j) = (1-\alpha)I_k(i,j) + \alpha\frac{1}{8}[I_k(i-1,j-1) + I_k(i-1,j) + I_k(i-1,j+1) + I_k(i,j-1) + I_k(i,j+1) + Ik(i+1,j-1) + I_k(i+1,j) + I_k(i+1,j+1)]$$

Nesta alínea foi desenvolvida a função *void difusion(int select)*. Esta função realizar a difusão da imagem do fractal com recurso à expressão referida anteriormente, guardando cada nova imagem na pasta *images,* com o nome "output%04d.jpg" com recurso à função *sprintf()*. Dado isto, para ser determinar o valor de cada pixel foi desenvolvida a *função int returnPixVal(int i, int j)* que analisa a posição do pixel atual e verifica se este se encontra localizado dentro nas margens da imagem. Caso esta condição não se cumpra, devolve o valor 0; por outro lado, quando isto se verifica, devolve o valor do pixel em questão (*imagem[i * resx + j]*).

É ainda relevante referir que este programa tira partido de OpenMP para acelerar todo este processo com recurso às diretivas *parallel* e *for*, à semelhança do

que ocorreu na alínea 3. Estas diretivas permitem criar um bloco de código paralelo que será executado por uma equipa de fios, bem como paralelizar a execução de ciclos (as iterações são divididas por vários fios).

Após todas as imagens serem devidamente guardadas e o processo de difusão terminar foi elaborado um vídeo que permite verificar este processo visualmente. Para criar o vídeo recorreu-se à aplicação *ffmpeg* tal como sugerido no enunciado.