

T4: Geração de Fractais de Mandelbrot em OpenMP

Testes:

- 1024 32 [1,2,4,8]
- 1024 64 [1,2,4,8]
- 1024 128 [1,2,4,8]
- 180 testes = 12 casos x 5 execuções x 3 soluções
 - 100 minutos de duração
- Width fixo?
 - $\text{frames+} : \text{width}^2$
 - $\text{width+} : 2 * \text{width} + 1$

Especificações:

- Procesador: Intel® Core™ i7-6700K
- Núcleos: 4 (8 threads)
- Frequência: 4.2 GHz
- Memória: 2 x 8 GB DDR4 2800 MHz
- SO: GNU (Debian Stretch Live 9.8 LXDE)

T4: Geração de Fractais de Mandelbrot em OpenMP

Laço externo:

- $\text{delta} = 0.98^i$;
- zoom
- executar sequencialmente;
- static: problema
 - frames se repetem
 - 256 100 2 -> 2 x 256 50 1
- dynamic:
 - distribuição na ordem
 - problemas com zoom
- com $\text{chunk_size} = \text{frames} * \text{width}^2$
 - funciona
 - porem equivaleria a execução sequencial;

Laços internos:

- para cada linha calcula todas as colunas;
 - onde de fato as coisas acontecem;
- paralelismo:
 - escalonamento fixo
 - particionamento variável: linha ou coluna
 - `#pragma omp parallel for schedule(dynamic)`
 - coluna: menos eficiente - fractalpar2.cpp
 - ate 4 threads: ~95%
 - 8 threads: ~77%
 - linha: melhores resultados - fractalpar1.cpp
 - ate 4 threads: 99~100%, speedup = nthreads
 - 8 threads: ~82%, speedup = 6,6
 - programa com 2 laços(?) - fractalpar3.cpp
 - até 4 threads: equivalente a linha
 - 8 threads: 5% menos eficiente que linha
 - introduz 2 divisões por laço
 - esperava piores resultados

```

#pragma omp parallel for schedule(dynamic)
for(int rowcol = 0; rowcol < width*width; rowcol++)
{
    int row = rowcol%width;
    int col = rowcol/width;

    const double cx = xMin + row * dw;
    double x = cx;
    double y = yMin + col * dw;
    int depth = 256;
    double x2, y2;

    do
    {
        x2 = x * x;
        y2 = y * y;
        y = 2 * x * y + (yMin + col * dw);
        x = x2 - y2 + cx;
        depth--;
    }
    while((depth > 0) && ((x2 + y2) < 5.0));

    pic[frame * width * width + col * width + row] = (unsigned char)depth;
}

```

fractalpar2

```
#pragma omp parallel for schedule(dynamic)
for(int row = 0; row < width; row++)
{
    ...
}
```

fractalpar1

```
for(int row = 0; row < width; row++)
{
    const double cy = yMin + row * dw;

    #pragma omp parallel for schedule(dynamic)
    for(int col = 0; col < width; col++)
    {
        ...
    }
}
```