Programação Paralela

Filipe Simões

Modificação do código original

- Foi colocada a entrada de dados como argumento apenas para facilitar os testes
- Adicionada a função wtime() para calcular o tempo

```c
long wtime(){
    struct timeval t;
    gettimeofday(&t, NULL);

    return t.tv_sec*1000000 + t.tv_usec;
}

int main(int argc, char** argv){
    int max_row, max_column, max_n;
    long start_time, end_time;

    max_row = atoi(argv[1]);
    max_column = atoi(argv[2]);
    max_n = atoi(argv[3]);

    start_time = wtime();

    end_time = wtime();

    printf("%ld usec\n", (long) (end_time - start_time));
```

Paralelização

- Mesmas alterações citadas no original
- Uso do schedule estático do OpenMP
- Chunk de max_row/5
- Poderia haver um chunk pro max_column também

```
#pragma omp parallel for private(r) shared(mat) schedule(static, chunk)
    for(r = 0; r < max_row; ++r){
        for(int c = 0; c < max_column; ++c){
            complex<float> z;
            int n = 0;
            while(abs(z) < 2 && ++n < max_n)
                z = pow(z, 2) + decltype(z)(
                    (float)c * 2 / max_column - 1.5,
                    (float)r * 2 / max_row - 1
                );
            mat[r][c]=(n == max_n ? '#' : '.');
        }
    }
```

Desempenho

Algoritmo	max_row	max_column	max_row	Tempo (seg)	speedup
Original	1000	1000	100	6.48	-
Paralelizado	1000	1000	100	3.79	1.71