
 <p>Universidade do Minho</p>	<p>Módulo 11</p> <p>Processamento Paralelo: correção e otimização</p>	
--	---	---

Objectivos:

Com esta sessão pretende-se introduzir as principais questões relativas à correção e optimização de programas paralelos.

Exercícios:

Pretende-se desenvolver uma versão paralela do seguinte programa:

```
#include <stdio.h>
#include <math.h>
#include <omp.h>
int main(){
    double soma=0;
    double time = omp_get_wtime();

    for(int i=0; i<1000000;i++) {
        soma += sin(i);
    }

    printf("Soma=%f Tempo=%f\n",soma, omp_get_wtime()-time);
    return(1);
}
```

- Crie um ficheiro designado `Opt.c` com o código e compile o programa usando o comando `gcc -O3 -fopenmp -std=c99 Opt.c`. Execute o binário criado (`./a.out`) e anote o resultado da execução (valor da soma e tempo de execução). Repita várias vezes a medição.
- Desenvolva uma versão paralela do programa colocando a diretiva OpenMP `#pragma omp parallel for` **shared(soma)** antes do ciclo. Repita várias vezes a medição e comente.
- Inclua a diretiva `#pragma omp critical` antes da linha `soma += sin(i);`. Repita várias vezes a medição e comente o resultado. Compare com o resultado obtido com `#pragma omp atomic`.
- Desenvolva agora uma versão que utilize uma variável `soma` para cada atividade paralela. Declare um vector `soma` com duas posições (em máquinas com dois núcleos) e cada atividade deverá incrementar a posição correspondente ao seu indentificador (`omp_get_thread_num()`).
- Inclua a diretiva `#pragma omp parallel for` **private(soma)** no programa original (alínea a)). Repita várias vezes a medição e comente o resultado.
- Inclua a diretiva `#pragma omp parallel for` **reduction(+:soma)** no programa original (alínea a)). Repita várias vezes a medição e comente o resultado.