Exercício 2: Paralelização do Crivo de Eratóstenes com OpenMP

Nome: Daniel Araujo Cavassani (COMP 25)

Data: 13/09/2024

Neste exercício, a tarefa foi paralelizar a versão sequencial do Crivo de Eratóstenes utilizando OpenMP no supercomputador Santos Dumont. O algoritmo original processa uma sequência de números de forma sequencial para determinar quais são primos, mas eu modifiquei o código para que ele pudesse utilizar várias threads, acelerando o processo de eliminação dos múltiplos.

Modificação no Código

A modificação foi feita no procedimento `Crivo` dentro do arquivo `Crivo.c`. Adicionei a diretiva `#pragma omp parallel for` para paralelizar o loop que percorre a sequência de números:

```
#include "Crivo.h"
#include <omp.h>

// Crivo: elimina os nao primos do marcador


void Crivo(ptrMarcador M) {

 // percorre bases primas
 // maior base limitada pela raiz quadrada do maior numero a marcar

 int maiorBase=(int)sqrtf((float)UltimoNumero(M));
 int base = PrimeiroPrimo(M);
 while(base <= maiorBase) {

     // elimina multiplos da base
     // a partir do quadrado da base
```

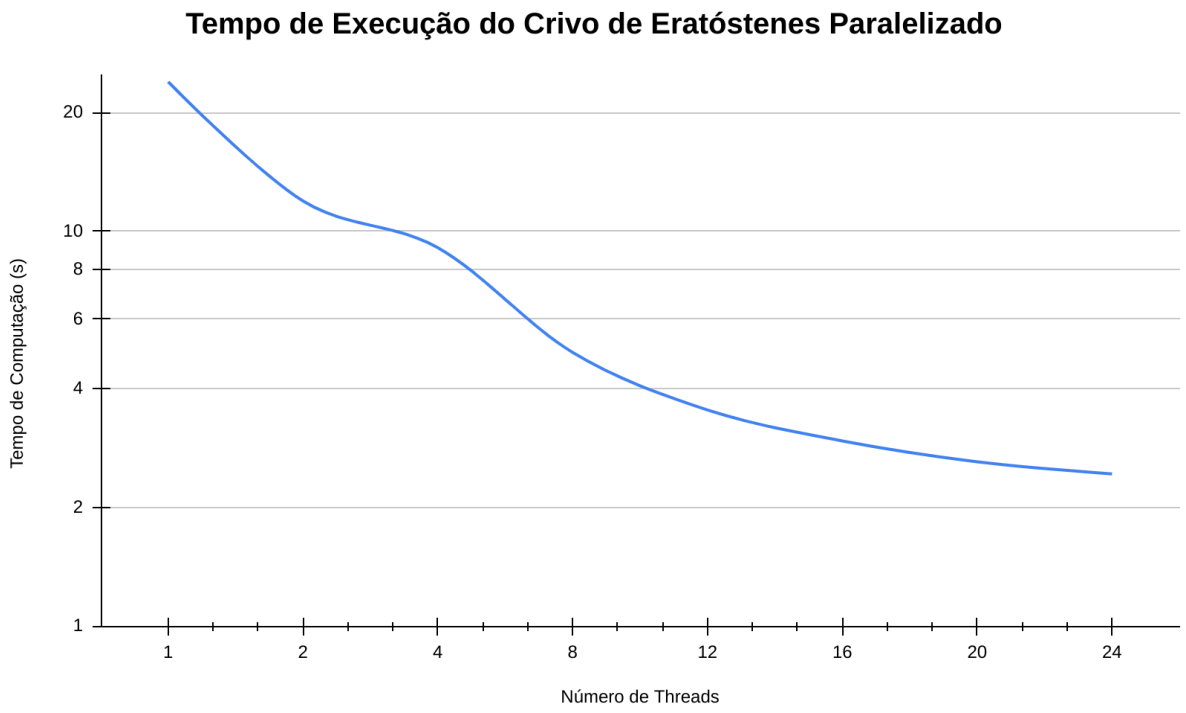
```
    #pragma omp parallel for
    for (int mult=base*base;
    mult<=UltimoNumero(M);
    mult+=base)
        MarcaMultiplo(M, mult);

      // avanca para a proxima base

      base = ProximoPrimo(M, base);
  }
}
```

Essa mudança permite que o loop que marca os múltiplos de números primos seja executado em paralelo, utilizando várias threads para acelerar a computação, especialmente para valores maiores de numMax.

Resultados

Depois de garantir que o código estava correto, rodei as execuções com diferentes números de threads (1, 2, 4, 8, 12, 16, 20, 24) para medir o tempo de computação. O speed-up foi calculado dividindo o tempo de execução com 1 thread pelo tempo de execução com múltiplas threads. Isso nos dá uma medida de quanto a paralelização melhorou a eficiência do programa. Aqui estão os tempos de execução e o respectivo speed-up para cada configuração:

Número de Threads	Tempo de Computação (s)	Speed-Up
1	23.906887	1
2	11.929981	2
4	9.08611	2.63
8	4.931167	4.85
12	3.52596	6.78
16	2.942777	8.13
20	2.605334	9.18
24	2.42829	9.84



Análise

Os resultados mostram uma clara redução no tempo de computação conforme o número de threads aumenta, especialmente a partir de 8 threads, onde o paralelismo se torna mais eficiente. No entanto, o aumento de threads não resulta em um ganho linear de desempenho, o que sugere a presença de overhead no gerenciamento das threads. Esse overhead é mais perceptível em configurações com menos threads, onde o tempo de computação não diminui de forma tão significativa. Já em configurações com 16 a 24 threads, o paralelismo se torna altamente vantajoso, demonstrando que o Crivo de Eratóstenes se beneficia amplamente da paralelização em cenários de maior escala.

Conclusão

A paralelização com OpenMP no Crivo de Eratóstenes trouxe ganhos consideráveis, especialmente para numMax elevado. Embora o overhead possa limitar os benefícios em alguns casos, o uso de múltiplas threads foi eficaz e reduziu o tempo de execução de forma substancial, mostrando que o paralelismo bem ajustado é essencial para otimizar a performance.