Tarefa 15 - Análise de Complexidade

Lucas Santiago de Oliveira

Março de 2022

Código escolhido para paralelizar:

```
// CRIVO DE ERASTOTENES

for(int p=2; p*p<=n; p++) {
  if(prime[p] == true)
    for(int i=p*2; i<=n; i += p)
      prime[i] = false;
}
```

Versão paralelizada do código:

```
// CRIVO DE ERASTOTENES PARALELO DE FORMA INGENUA

#pragma omp parallel for
for(int p=2; p*p<=n; p++) {
  if(prime[p] == true)
    #pragma omp parallel for
    for(int i=p*2; i<=n; i += p)
      prime[i] = false;
}
```

O grande problema do algoritmo de Erastótenes é que há grande dependência de dados entre os dois *for* presentes. O *if* que está entre os dois causa uma dependência de dados.

Uma forma de tornar esse código mais eficiente seria trocar a posição entre o segundo *for* com o primeiro *if*. Para que isso fosse possível seria necessário modificar parte do código para que o teste se o valor dentro do vetor de primos seja verdadeiro sem afetar o resultado correto do algoritmo.

Com um número de *threads* infinito seria possível tornar esse código de $\mathcal{O}(N^2)$ para $\mathcal{O}(\mathcal{N})$, se a primeira parte do código for executada de forma paralela entre todas as *threads*, a segunda parte que envolve uma dependência de dados seja feita de forma sequêncial entre todas as *threads* com tempo $\mathcal{O}(N)$.