



# **Aplicação do Método de Monte Carlo em OpenMP**

Lucas Ferreira da Silva

# Carga de trabalho

- ◎ Concentrada no **laço principal** da função *main*;
- ◎ Laços aninhados;
- ◎ Múltiplas **chamadas a funções** secundárias;
- ◎ Dependência de **acesso a objetos**;
- ◎ **Desbalanço de carga** entre os dois laços aninhados.


```
// para cada probabilidade, calcula o percentual de árvores queimadas
for (int ip = 0; ip < n_probs; ip++) {

    prob_spread[ip] = prob_min + (double) ip * prob_step;
    percent_burned[ip] = 0.0;
    rand.setSeed(base_seed+ip); // nova sequência de números aleatórios

    // executa vários experimentos
    for (int it = 0; it < n_trials; it++) {
        // queima floresta até o fogo apagar
        forest->burnUntilOut(forest->centralTree(), prob_spread[ip], rand);
        percent_burned[ip] += forest->getPercentBurned();
    }

    // calcula média dos percentuais de árvores queimadas
    percent_burned[ip] /= n_trials;

    // mostra resultado para esta probabilidade
    printf("%lf, %lf\n", prob_spread[ip], percent_burned[ip]);
}
```

A decorative background featuring a network diagram with nodes and connecting lines, primarily located in the top-left and bottom-right corners.

# **1ª Implementação com OpenMP**

# Implementação 1

- ◎ Definição da região paralela;
- ◎ Paralelização do **laço mais externo**;
- ◎ Tratamento dos objetos compartilhados ***rand*** e ***forest***;
- ◎ Definição do *schedule* ***dynamic***;
- ◎ Cuidado com a criação das **instâncias** do objeto ***forest***.

```
// para cada probabilidade, calcula o percentual de árvores queimadas
int ip, it;


#pragma omp parallel firstprivate(rand) private(ip, it) default(shared)
{
    Forest *forest = new Forest(forest_size);
    #pragma omp for schedule(dynamic)
    for (ip = 0; ip < n_probs; ip++){
        prob_spread[ip] = prob_min + (double)ip * prob_step;
        percent_burned[ip] = 0.0;
        rand.setSeed(base_seed + ip); // nova sequência de números aleatórios

        // executa vários experimentos
        for (it = 0; it < n_trials; it++)
        {
            // queima floresta até o fogo apagar
            forest->burnUntilOut(forest->centralTree(), prob_spread[ip], rand);
            percent_burned[ip] += forest->getPercentBurned();
        }

        // calcula média dos percentuais de árvores queimadas

        percent_burned[ip] /= n_trials;

        // mostra resultado para esta probabilidade
        printf("%lf, %lf\n", prob_spread[ip], percent_burned[ip]);
    }
}
```

A decorative background featuring a network diagram with nodes and connecting lines, primarily located in the top-left and bottom-right corners.

# **2ª Implementação com OpenMP**

# Implementação 2

- ⊙ Definição da região paralela;
- ⊙ Paralelização do **laço mais interno**;
- ⊙ Definição do *schedule* **static**;
- ⊙ Cuidado com a criação das **instâncias** do objeto **forest**.



```
// para cada probabilidade, calcula o percentual de árvores queimadas
for (int ip = 0; ip < n_probs; ip++){
    prob_spread[ip] = prob_min + (double)ip * prob_step;
    percent_burned[ip] = 0.0;
    rand.setSeed(base_seed + ip); // nova sequência de números aleatórios

    int it;
    #pragma omp parallel private(it) default(shared)
    {
        Forest *forest = new Forest(forest_size);

        // executa vários experimentos
        #pragma omp for schedule(static)
        for (it = 0; it < n_trials; it++)
        {
            // queima floresta até o fogo apagar
            forest->burnUntilOut(forest->centralTree(), prob_spread[ip], rand);
            percent_burned[ip] += forest->getPercentBurned();
        }
    }

    // calcula média dos percentuais de árvores queimadas

    percent_burned[ip] /= n_trials;

    // mostra resultado para esta probabilidade
    printf("%lf, %lf\n", prob_spread[ip], percent_burned[ip]);
}
```

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are larger and have concentric circles inside, suggesting a hierarchical or multi-layered structure. The lines are thin and gray, connecting the nodes in a non-linear fashion.

# Experimentos

# Experimentos

◎ 10 execuções de cada configuração;

◎ Execução das 3 versões:

- **Sequencial**
- **OpenMP**
- **OpenMP2**

◎ 3 categorias de configuração de tamanho:

	<b>./firesim</b>	<b>&lt;tamanho-do-problema&gt;</b>	<b>&lt;nro. experimentos&gt;</b>	<b>&lt;probab. maxima&gt;</b>
○ Grande		40	6000	101
○ Médio		20	3000	50
○ Pequeno		10	1000	10

# Experimentos

## ◎ 3 variações do número de threads:

- 2 threads;
- 4 threads;
- 6 threads;

## ◎ Hardware:

- Intel® Core™ i5-2410M
- 2.30GHz
- 2 Cores
- 4 Threads
- 6 GB de RAM

## ◎ Sistema Operacional:

- Debian GNU/Linux Buster
- Versão do Linux: 4.15.0-2-amd64
- Versão do gcc: 7.3.0

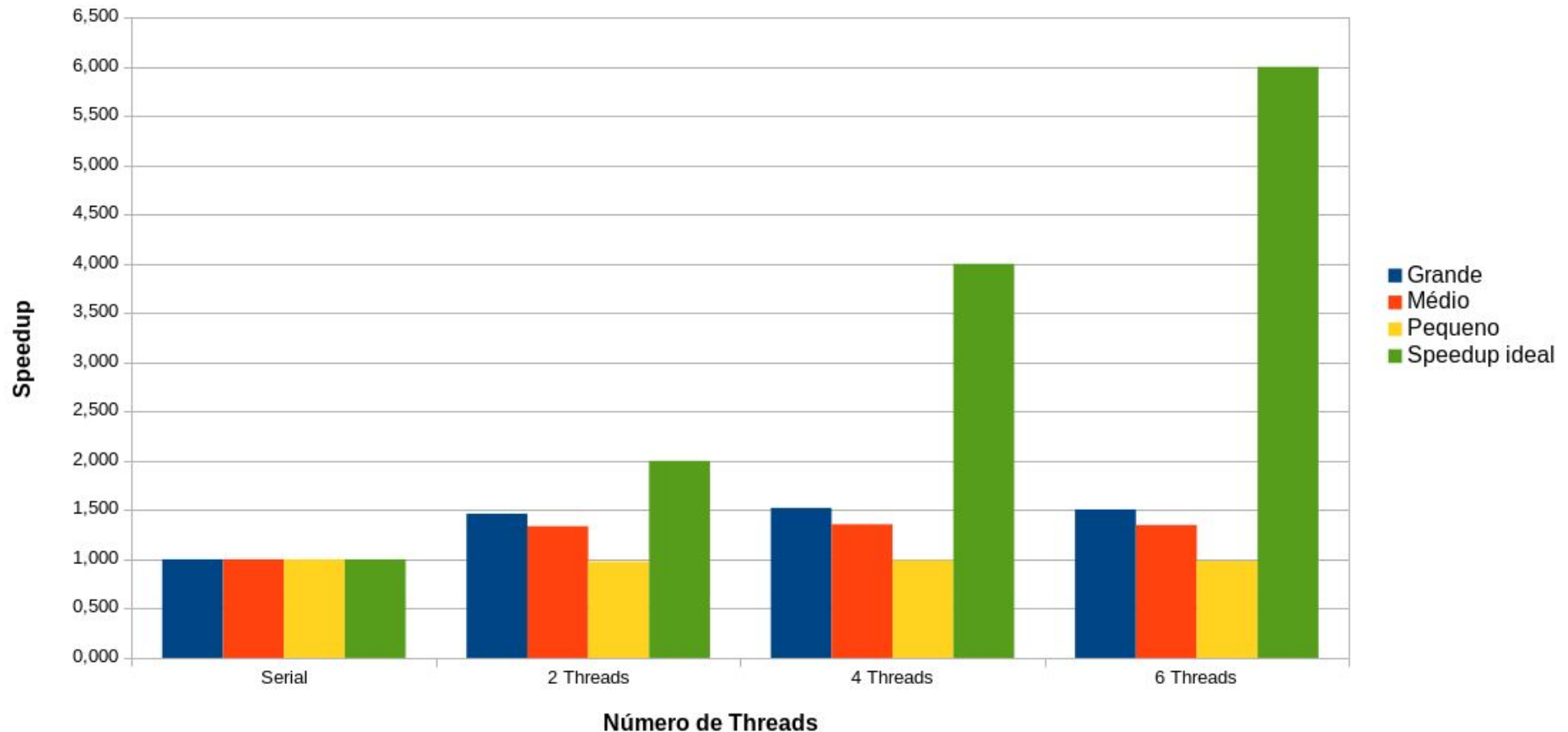
A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by small circles, some of which are larger and have concentric circles inside, suggesting a hierarchical or multi-layered structure. The lines are thin and gray, connecting the nodes in a non-linear fashion.

# Resultados

# Influência do tamanho do problema

Relação speedup pelo tamanho do problema

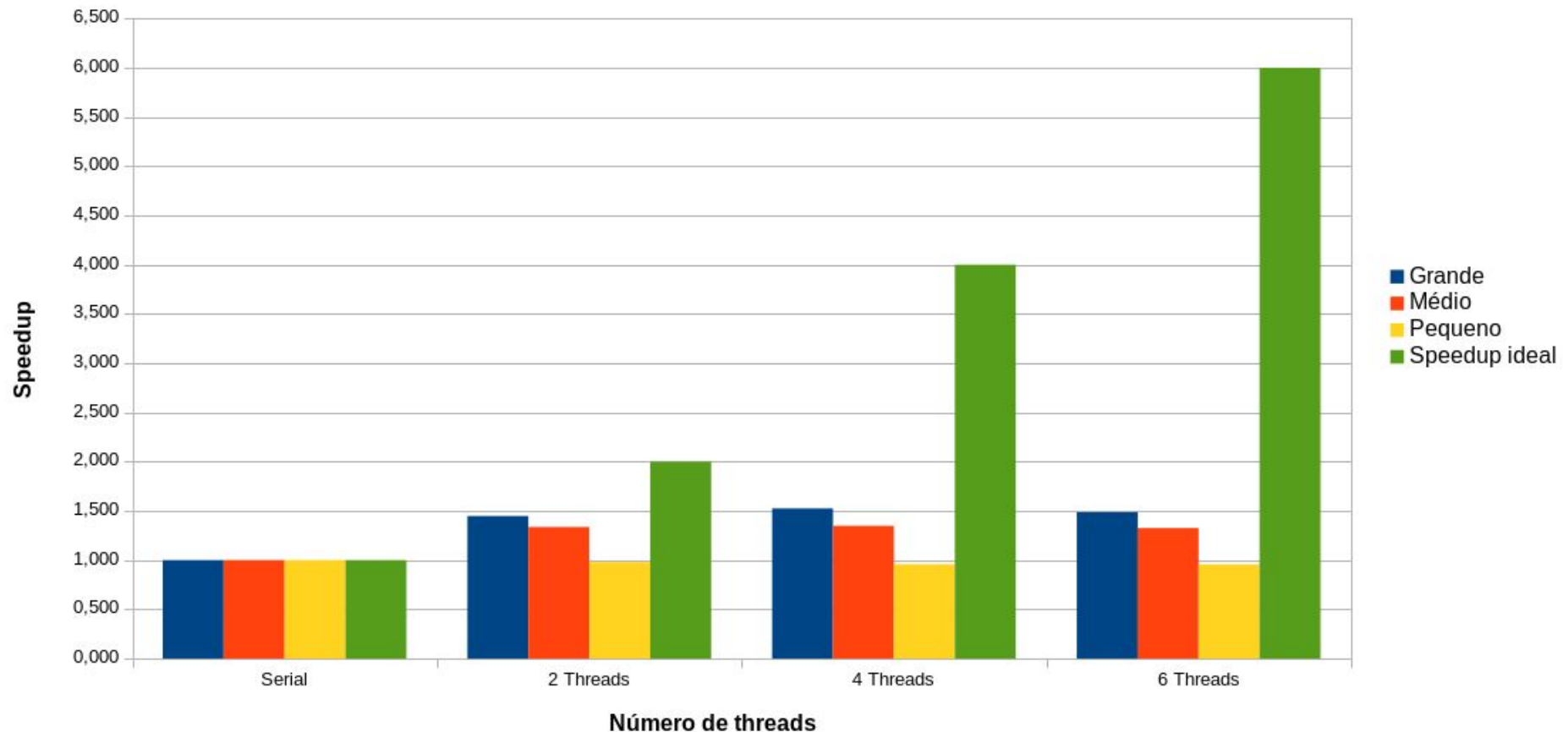
(Implementação 1)



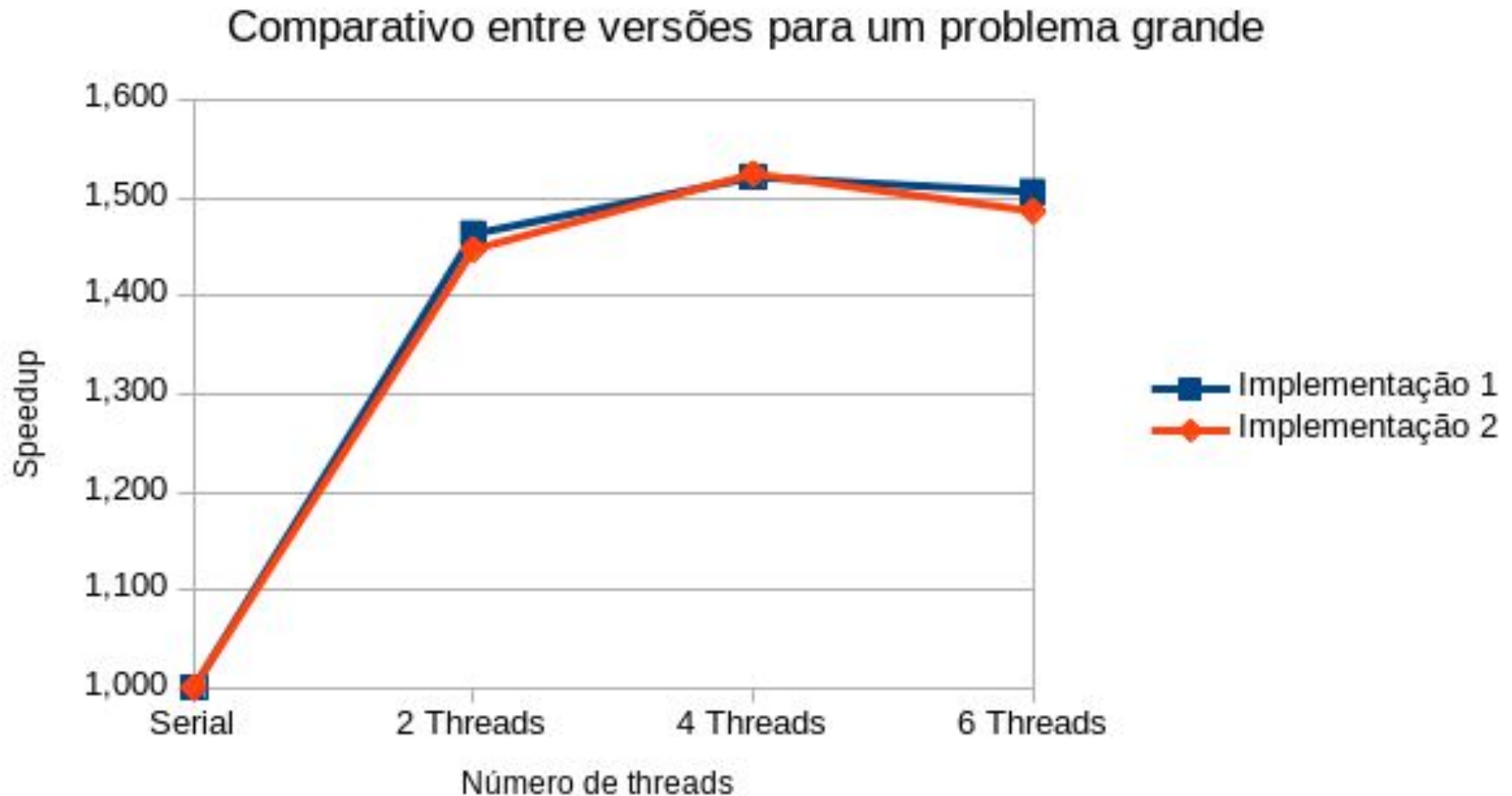
# Influência do tamanho do problema

Relação speedup pelo tamanho do problema

(Implementação 2)

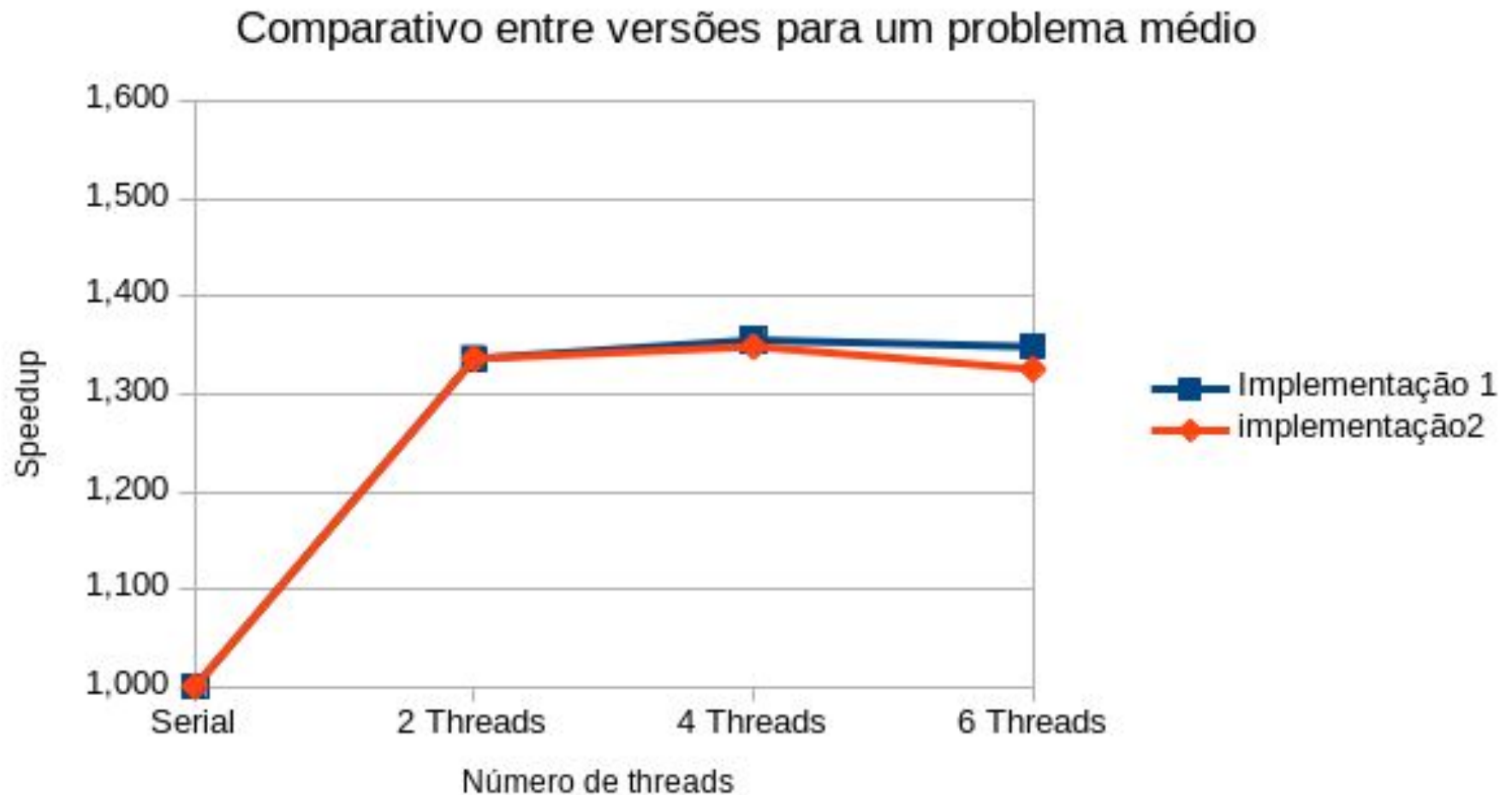


# Comparação das versões implementadas com OpenMP



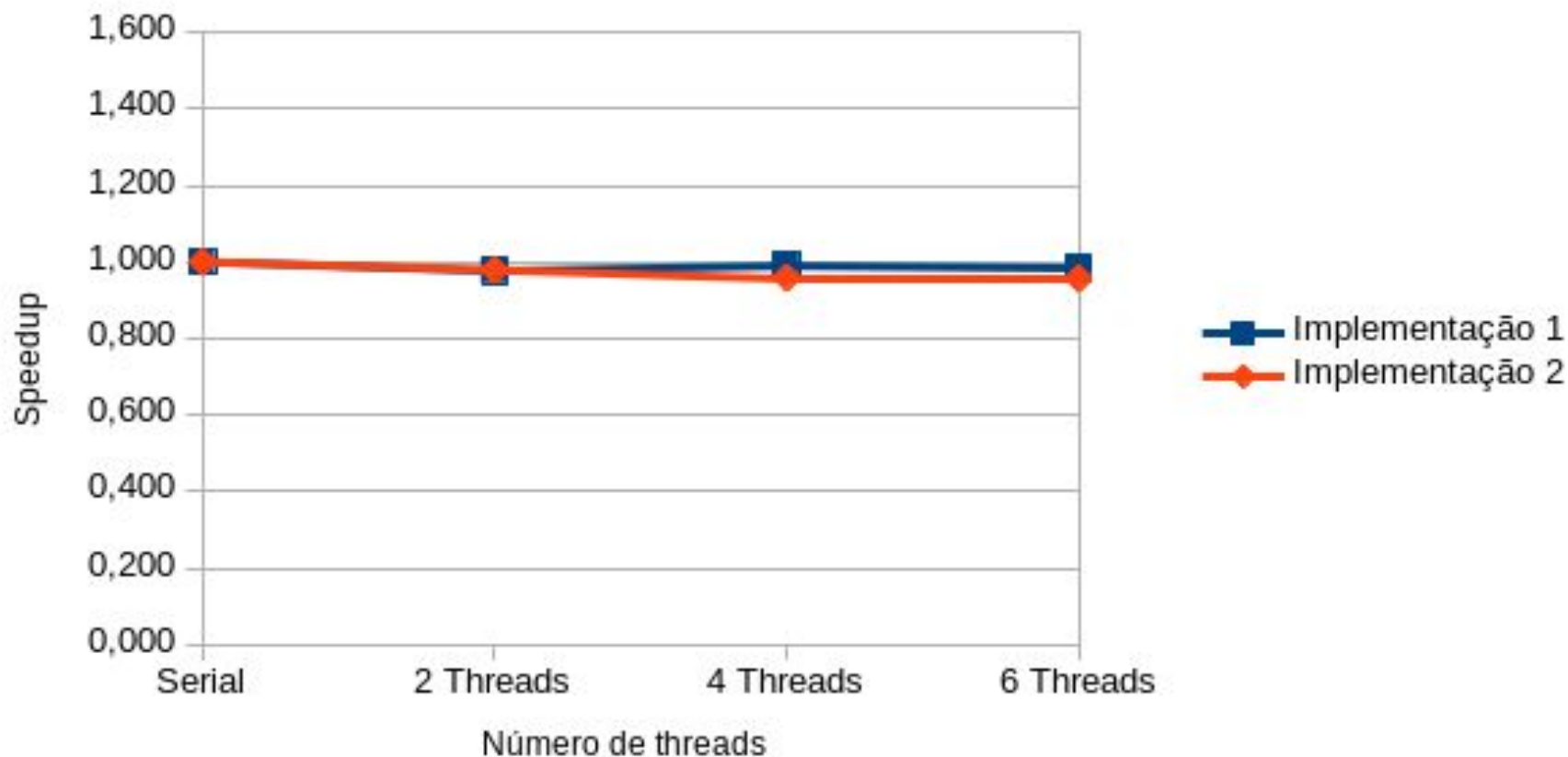


# Comparação das versões implementadas com OpenMP



# Comparação das versões implementadas com OpenMP

Comparativo entre versões para um problema pequeno



# Conclusões

- ◎ **Implementação 1** mostrou-se **melhor** que a implementação 2:
  - Overhead de criação da região paralela múltiplas vezes?
- ◎ **Ambas implementações** com OpenMP tiveram *speedup menor* que a versão serial para **problemas pequenos**:
  - Custo de criação da região paralela, criação de threads e controle da exclusão mútua.
  - Possível solução: Paralelizar as funções auxiliares chamadas no laço?
- ◎ Tentativa de utilização de outras estratégias de paralelização...