

# 1 Introdução

Seu trabalho consiste em implementar um algoritmo clássico de computação, transformá-lo em um programa paralelo e fazer o cálculo de *speedup*, eficiência e máximo teórico.

## 1.1 Problemas disponíveis

- Método de Gauss-Jacobi para resolução de sistemas lineares;
- Algoritmo de Dijkstra para distância em grafos;
- Equação do calor 2D (*2D Steady State Heat Equation*);
- Imagem PPM do *Mandelbrot set*;

# 2 O trabalho

Você deve escolher um dos problemas e implementar sua versão sequencial para resolver o problema. Você deve medir o tempo das regiões que considera paralelizáveis e a partir desses dados, deve calcular o *speedup* máximo teórico para o seu problema.

Depois, implemente a versão paralela, meça o tempo e calcule as métricas de *speedup* e eficiência.

## 2.1 Como medir o tempo

Você pode usar as funções de medição de tempo do próprio OpenMP ou até do próprio Linux, quando o tempo global for suficiente para os cálculos.

Você deve realizar 5 medições e calcular a média para fins de análise, **para cada medição**. Você deve realizar medições para diferentes tamanhos de entrada, e deve atingir exemplos que executem por pelo menos 10 minutos antes de gerar a resposta.

Quando testar a sua versão paralela, faça testes com várias opções de *threads*: 1, 2, 4, 8, ... Lembre-se de testar com as mesmas entradas. Os testes devem ser feitos na mesma máquina, para evitar interferência do hardware nos resultados.

## 2.2 Critérios de correção

Seu trabalho será corrigido considerando:

- Seus programas devem funcionar corretamente!
- Objetividade e clareza do relatório.
- Uso dos métodos e construções adequadas de programação e de OpenMP.
- Domínio do trabalho na defesa.
- Adequação à esta especificação.

Não serão aceitos trabalhos entregues fora do prazo. Trabalhos sem relatório ou que não sejam defendidos não serão considerados!

Plágio e/ou uso de inteligência artificial acarretará no trabalho sendo desconsiderado!

## 2.3 Relatório

A equipe deve entregar um relatório no formato `.pdf`, explicando como resolveu os desafios do trabalho.

O relatório deve explicar o problema escolhido, ideia da solução sequencial, discussão das regiões que o grupo julgou que são paralelizáveis, discussão das métricas estimadas e dos resultados reais. A discussão deve mostrar a escalabilidade do problema, relacionar os resultados obtidos com a estimativa esperada, e discutir se a solução proposta é interessante.

O relatório deve conter todas as informações necessárias para reproduzir a sua solução. Inclua no seu relatório as informações do hardware utilizado, e de cada medição de tempo realizada.

Inclua as referências de pesquisa e código que utilizar para seu trabalho no final de seu relatório.

## 3 Entrega

Você deverá entregar pelo SUAP, até 03/outubro, um arquivo `.zip` contendo uma pasta, que por sua vez contém:

- Os arquivos `.c` ou `.cpp` da versão sequencial E da versão paralela;
- Um arquivo de texto chamado `README.md` contendo o nome completo dos integrantes do grupo e informações sobre como compilar os programas;
- Um relatório chamado `relatorio.pdf` contendo o nome completo dos integrantes do grupo e o conteúdo descrito na seção 2.3.

O arquivo compactado descrito acima deve ser nomeado com as iniciais do nome de cada integrante separados por um hífen. Por exemplo: se o Fulano da Silva Sousa fez o trabalho com o João de Souza, o arquivo deve ser nomeado `fss-js.zip`. A pasta, dentro do arquivo compactado, deve ter o mesmo nome, a não ser pela extensão.

O trabalho pode ser feito em grupos de até 3 pessoas.

### Histórico das Revisões:

- 19/set/2025 - v1.0: primeira versão.