



Trabalho Prático 2

*** Data de entrega: até 12/12/2025 ***

Este trabalho prático envolve programação paralela com *threads*. O objetivo do trabalho é proporcionar aos estudantes uma experiência com o uso de *threads* para processamento paralelo e ganho de desempenho. O trabalho envolve algoritmos com vetores e matrizes e uma “simulação” de uma aplicação onde o uso de *threads* apresenta vantagens.

Nas duas primeiras questões, devem ser desenvolvidos programas paralelos e sequenciais para uma avaliação de desempenho simples e direta com base no ganho no tempo de execução de um programa em relação ao outro. Portanto, a métrica a ser utilizada é a *Aceleração* ou *speedup* (Sp), que é a razão entre o tempo de execução da versão sequencial (T_s) e o tempo de execução da versão paralela com p processadores (T_p). $Sp = T_s / T_p$

1. Desenvolva dois programas para o cálculo do produto escalar entre 2 vetores, sendo um algoritmo sequencial tradicional e outro paralelo. O algoritmo paralelo deve permitir um número variável de threads, a escolha do usuário. Execute o programa com vários tamanhos de vetor e apresente um gráfico com duas curvas: uma para o tempo de execução em função do tamanho do vetor para o programa paralelo e outra para o programa sequencial. Execute o mesmo programa com diferentes quantidades de threads em pelo menos dois computadores com diferentes quantidades de CPUs e relate qual o impacto das diferentes CPUs no tempo de execução dos programas e calcule a métrica de Aceleração para cada configuração de CPU e quantidade de threads utilizada. Os resultados das execuções com as diferentes configurações de CPUs e quantidade de threads usadas no programa devem ser apresentados (sugestão: mostrar os resultados na forma de tabelas).
2. Desenvolva dois programas para o cálculo da multiplicação entre 2 matrizes, sendo um algoritmo sequencial tradicional e outro paralelo. O algoritmo paralelo deve permitir um número variável de threads, a escolha do usuário. A equipe deve definir no relatório qual a forma de cálculo do algoritmo paralelo, considerando que há várias alternativas de divisão dos cálculos. Execute o programa com vários tamanhos de matrizes e apresente um gráfico com o tempo de execução em função do tamanho das matrizes para o programa paralelo e outro para o programa sequencial. Execute o mesmo programa com diferentes quantidades de threads em pelo menos dois computadores com diferentes quantidades de CPUs e relate qual o impacto das diferentes CPUs no tempo de execução dos programas e calcule a métrica de Aceleração para cada configuração de CPU e quantidade de threads utilizada. Os resultados das execuções com as diferentes configurações de CPUs e quantidade de threads usadas no programa devem ser apresentados (sugestão: mostrar os resultados na forma de tabelas).



3. Desenvolva dois programas que **simulem** uma aplicação, que pode ser uma das listadas mais abaixo ou outra qualquer à escolha da equipe. Os dois programas deverão ser:
1. Aplicação com o uso de *threads*;
 2. Aplicação equivalente sem o uso de *threads* (programação sequencial tradicional).

O programa paralelo deve receber como entrada a definição da quantidade de threads e outros parâmetros necessários para a execução do programa. O programa deve ser “interativo”, onde o usuário possa executar comandos que resultem em alguma ação. A saída do programa deve ilustrar o funcionamento do sistema, deixando bem visível o funcionamento das *threads*. Use muitos “prints” na tela para isso. Recomenda-se utilizar janelas, abas e/ou *popups*.

O mais importante é a equipe compreender o conceito e a aplicação de threads e demonstrar o ganho no desempenho e/ou apresentar funcionalidades que não seriam possíveis com a programação sequencial tradicional. Usem a criatividade para melhor ilustrar o funcionamento dos programas e facilitar a percepção das vantagens do uso de threads.

IMPORTANTE: Cada equipe deve obrigatoriamente escolher um exemplo de aplicação para a questão 3 e informar ao professor com antecedência para divulgar para a turma. Uma aplicação não pode ser usada como exemplo por mais de uma equipe.

Exemplos de aplicações onde *threads* são amplamente utilizadas:

- Navegador Web
- Servidor Web
- Jogos
- Editores de Texto
- Manipulação de matrizes (principalmente matrizes grandes, da ordem de milhares de elementos)
- IDEs (interfaces gráficas que apresentam janelas, abas, popups, etc)
- Media player

OBSERVAÇÕES SOBRE AS IMPLEMENTAÇÕES:

- Para este trabalho pode-se utilizar uma das 3 linguagens abaixo:
 1. C ou C++, por meio da biblioteca pthread;
 2. Java usando a classe Thread.
- Pode-se utilizar atrasos, usando a função *sleep*, se acharem necessário para deixar as *threads* com diferentes velocidades e simular o tempo de execução de ações como “salvar arquivo”, “correção ortográfica”, “um download”, “uma ação em um jogo”, etc, conforme o tema da aplicação escolhida pela equipe.
- Se o foco do tema da aplicação escolhida for desempenho, registrem o tempo de execução de cada versão (com threads e sem threads) para comparação dos resultados.



UFAM

UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO

SISTEMAS OPERACIONAIS
PROF. JOÃO MARCOS BASTOS CAVALCANTI

OBSERVAÇÕES SOBRE A ENTREGA DO TRABALHO:

- O trabalho pode ser feito em equipe de no máximo 4 (quatro) estudantes.
- O trabalho deverá ser apresentado ao professor, o qual fará algumas perguntas aos autores do trabalho. Deve-se combinar previamente com o professor dia, hora e local para a apresentação do trabalho até a data limite da entrega.
- A equipe deve elaborar um pequeno documento/relatório com a descrição e definição das ideias para as duas primeiras questões e para a aplicação de exemplo escolhida. Também devem ser relatadas as decisões tomadas sobre como demonstrar o uso das threads e os resultados das avaliações de desempenho.
- Os arquivos com a descrição textual da solução e os códigos fonte dos programas deverão ser entregues pelo ColabWeb com a identificação da equipe (basta um membro de equipe submeter o trabalho no ColabWeb).
- Deve-se registrar a(s) fonte(s), sejam URLs, livros, slides, etc, de onde foram obtidos códigos fontes ou referências para o desenvolvimento do trabalho.