Laboratório 3
Multiplicação de matrizes concorrente

Programação Concorrente (ICP-361) 2024-2
Prof. Silvana Rossetto

[1]Instituto de Computação/UFRJ

Introdução

O objetivo deste Laboratório é implementar uma **versão concorrente do problema de multiplicação de matrizes** e avaliar o ganho de desempenho obtido.

Nas vídeo-aulas que acompanham este laboratório apresenta-se, como exemplo, a implementação do problema de multiplicação **matrizXvetor** de forma sequencial e concorrente.

Nos códigos auxiliares, mostra-se como gerar matrizes de entrada para testar os programas.

Atividade 1

Objetivo: Projetar e implementar uma **solução concorrente** para o problema de **multiplicação de matrizes**, coletar informações sobre o seu tempo de execução, e calcular o ganho de desempenho obtido.

Requisitos de implementação: Os seguintes requisitos de implementação deverão ser atendidos:

- As matrizes de entrada e saída serão do tipo $float$, com N linhas e M colunas.
- As matrizes de entrada **devem** ser carregadas de **arquivos binários** previamente gerados, onde os dois primeiros valores (do tipo inteiro) indicam as dimensões da matriz (N e M), e os demais elementos (do tipo float) são a sequência de valores da matriz.
- As matrizes deverão ser representadas internamente como **vetores de float** (variável do tipo ponteiro, alocada dinamicamente). (Nas vídeo-aulas e códigos que acompanham este Laboratório há exemplos dessa forma de representação.)
- A matriz de saída deverá ser escrita em um **arquivo binário**, no mesmo formato dos arquivos de entrada.
- O programa deverá receber como entrada, na **linha de comando**, os **nomes dos arquivos de entrada e de saída**, e a **quantidade de threads** de processamento.
- O programa deverá incluir chamadas de **tomada de tempo de execução interna do programa**, separando as partes de **inicialização, processamento e finalização** do programa.

Roteiro para implementação e avaliação:

1. Comece implementando uma **versão sequencial** do programa para usá-la como referência para os testes de corretude.

2. Implemente o programa que realiza a multiplicação das matrizes de entrada de forma **concorrente**, seguindo todos os requisitos de implementação descritos acima.
3. Verifique a corretude da sua solução (matriz de saída correta). Para isso, pode-se comparar a matriz de saída da versão concorrente com a matriz de saída da versão sequencial. (sugestão, usar o comando $diff < arq1 >< arq2 >$, se retornar vazio significa que os arquivos são iguais).
4. Avalie o **tempo de execução** de cada parte do programa usando matrizes de entrada de dimensões $500X500$, $1000X1000$ e $2000X2000$. Importante: Repita a execução de cada configuração pelo menos $5vezes$ e registre **o valor médio** das medidas tomadas.
5. Calcule a **aceleração (A)** e a **eficiência (E)** alcançada, executando o programa concorrente com 1, 2, 4 e 8 threads:

$$A(n,t) = T_s(n)/T_p(n,t)$$

$$E(n,t) = A(n,t)/t$$

(n é a dimensão das matrizes e t é a quantidade de threads usadas na execução.
6. Registre todos os dados levantados e calculados em uma TABELA, gere os **gráficos de aceleração e eficiência**, informe a **configuração da máquina** (quantidade de unidades de processamento) usada para os testes e reporte tudo isso em um **documento PDF**.

Entrega do laboratório: Disponibilize o código implementado na **Atividade 1** em um ambiente de acesso remoto (GitHub ou GitLab). Use o formulário de entrega desse laboratório para enviar o link do repositório do código implementado e o **relatório de avaliação (documento PDF)**.