Laboratório 3
Implementação e avaliação de aplicações concorrentes (parte 2)

Computação Concorrente (MAB-117)
Prof. Silvana Rossetto

[1]DCC/IM/UFRJ — 22 de agosto de 2019

Introdução

O objetivo deste Laboratório é implementar uma versão **concorrente** para o problema de **multiplicação de matrizes** e medir e **ganho de desempenho** obtido. Usaremos a linguagem C e a biblioteca *Pthreads*.

Para cada atividade, siga o roteiro proposto e responda às questões colocadas.

Atividade 1

Objetivo: Finalizar a implementação concorrente do problema de **multiplicação matriz-vetor (Lab-2)** e mostrar sua solução para a professora.

Atividade 2

Objetivo: Projetar e implementar um algoritmo concorrente para o problema de **multiplicação de matrizes** e coletar informações sobre o seu tempo de execução.

Projete um algoritmo concorrente para a tarefa de multiplicar duas matrizes, tomando como base a implementação do problema multiplica **matriz-vetor** que você implementou no Lab2. A entrada e saída de dados deve continuar sequencial, apenas a parte central da multiplicação das matrizes deve ser paralelizada. Considere como **subtarefa mínima o cálculo de uma linha da matriz de saída**. Você pode escolher diferentes abordagens para distribuir as subtarefas entre as threads (por ex., cada thread é responsável por um bloco contínuo de linhas, ou por blocos de linhas intercaladas com as demais threads).

O número de threads e os nomes dos arquivos com as matrizes de entrada devem ser lidos da linha de comando, ou seja, o usuário do seu programa deverá poder alterar o número de threads e/ou as matrizes de entrada a cada execução, sem precisar recompilar o programa!

Roteiro:

1. Implemente uma solução **concorrente** do problema de multiplicação de matrizes.
2. Acrescente no seu programa chamadas da função `GET_TIME()` para medir separadamente os tempos de execução para: (a) carregar as matrizes de entrada; (b) criar as threads, executar a multiplicação, e aguardar o término das threads; e (c) finalizar o programa.
3. Verifique a corretude da sua solução (matriz de saída correta), usando matrizes de entrada menores.
4. Avalie o desempenho da sua solução, usando as matrizes da pasta **dados**, disponibilizada no LAB-2. (Obs.: multiplique cada matriz por ela mesma.)
5. Varie o número de threads entre 1, 2 e 4.
6. **Observe como os tempos medidos variam de acordo com a dimensão das matrizes de entrada.**

Relatório da atividade:

1. Crie o arquivo **lab3.txt** e registre as medidas de tempo coletadas (para cada uma das matrizes da pasta **dados**) e o cálculo do ganho de desempenho (*speedup/aceleração*) obtido com a versão concorrente ($T_{sequencial}/T_{concorrente}$). Considere a execução da aplicação com uma única thread como o tempo de execução sequencial. Faça 5 execuções de cada caso e registre a de menor tempo da parte concorrente.
2. No site da disciplina, acesse novamente a página com o roteiro da aula de hoje. Ao lado do roteiro está disponível um Link para um formulário no Drive.
3. Preencha os formulário com os dados registrados no arquivo **lab3.txt** e envie suas respostas até **30 de agosto**.