Centro Federal de Educação Tecnológica Celso Suckow da Fonseca
UNED Nova Friburgo
Bacharelado em Sistemas de Informação
Programação Paralela e Concorrente
Professor Bruno Policarpo Toledo Freitas

Paralelização de algoritmos com OpenMP

Data de entrega: 11 de julho de 2025

1 Objetivo

Aplicar os conceitos de paralelização com OpenMP ou MPI e análise de desempenho de programas paralelos vistas em aula

2 Algoritmos

O trabalho consistirá em paralelizar e apresentar uma análise dos seguintes quatro algoritmos utilizando a biblioteca OpenMP em C ou C++:

1. Multiplicação Matricial

   O enunciado do trabalho apresenta uma implementação dessa operação usando a biblioteca "LibPPC" que se encontra na mesma pasta do programa. Essa implementação pode ser adaptada para utilizar nos próximos algoritmos.

   A biblioteca LibPPC pode ser utilizada para facilitar a execução do trabalho. Ela consiste em diversas funções para gerar dados aleatórios; salvar e carregar dados de arquivos; e tipos de dados.

   O header libppc.h contem a lista com todas as funções que podem ser utilizadas, juntamente com suas descrições.

2. Os seguinte algoritmos de ordenação:
   (a) Bubblesort
   (b) Merge sort

3. Transformada Discreta de Cossenos (em 1-D)

   DCT-II, disponível em `https://en.wikipedia.org/wiki/Discrete_cosine_transform`

3 Especificação

Para cada um dos algoritmos, criar uma seção do algoritmo, criando uma subseção para realizar os seguintes passos, em ordem :

3.1. Descrever o algoritmo (preferencialmente com exemplos)

Não é para explicar o código! É o algoritmo! São 2 coisas diferentes!

3.2. Implementar os algoritmos seriais

Coloque no relatório a implementação

Critério (Tempo de execução, Speedup ou Eficiência)			
	Threads		
Tamanho das entradas (SIZE)	1 (serial)	2	4
TAMANHO_1			
TAMANHO_2			
TAMANHO_3			

Figura 1: Modelo de tabela de resultados

3.3. A partir da implementação do algoritmo do item anterior, indicar no código:

1. Regiões Críticas
2. Dependências de dados

3.4. Implementar o algoritmo em paralelo, baseado no algoritmo serial

Coloque no relatório a implementação

3.5. Uma comparação de desempenho entre os algoritmos serial e sua implementação paralela, sob os critérios de **Tempo de execução**, **Speedup** e **Eficiência**.

Para cada um desses critérios, crie uma tabela com os valores obtidos sob diferentes tamanhos de entradas e quantidade de threads, conforme a Figura 1. *TAMANHO_1*, *TAMANHO_2*, e *TAMANHO_3* são 3 valores que vocês deverão escolher para realizar os experimentos, podendo ser também diferentes para cada algoritmo do enunciado.

Um exemplo de valores que vocês podem colocar para cada algoritmo podem ser vistos na Tabela 1:

' **Algoritmo**	**TAMANHO_1**	**TAMANHO_2**	**TAMANHO_3**
Multiplicação Matricial	500x500	1000x1000	2000x2000
Algoritmos de ordenação	10000	200000	400000
DCT 1-D	10000	200000	400000

Tabela 1: Sugestões de valores iniciais para os algoritmos (vocês podem mudar, veriquem pelo tempo de execução!)

3.6. Crie os seguintes gráficos **em colunas** , baseados nos resultados do item 3.5:

3.6.1. *Speedup* x *tamanho das entradas*, para 2 e 4 threads, e variando o tamanho das entradas em *TAMANHO_1*, *TAMANHO_2*, e *TAMANHO_3*, no mesmo gráfico.

3.6.2. *Eficiência* x *tamanho das entradas*, para 2 e 4 threads, e variando o tamanho das entradas em *TAMANHO_1*, *TAMANHO_2*, e *TAMANHO_3*, no mesmo gráfico

4 Avaliação

4.1. O aluno/dupla deverá entregar um relatório em formato de artigo resumido, descrevendo a estratégia de paralelização e os resultados obtidos, e um arquivo compactado com os códigos-fonte desenvolvidos

O relatório deverá ser escrito no formato de artigo da SBC, disponível em `https://www.sbc.org.br/documentosinstitucionais/#publicacoes`.

5 Observações

- O programa com a implementação da Multiplicação Matricial, ao ser executado pela primeira vez, irá criar um arquivo com valores aleatórios para a execução dos testes. Ao executar o programa de novo, ele usará como entrada o arquivo criado, ao invés de gerar valores aleatórios de novo.

  Isso foi feito para que vocês possam comparar os programas seriais com os paralelos usando as mesmas entradas.

- Nesse mesmo programa, está a biblioteca *LibPPC*, com algumas funções criadas para ajudar vocês no trabalho. A lista de funções disponíveis está no arquivo *include/libppc.h*.

- Lembre-se de verificar se seu programa paralelo está funcionando corretamente – ou seja, compare a saída do programa serial com o paralelo, no mínimo!

- Cuidado com os timings! Ou seja, meça o tempo de execução apenas durante a execução do algoritmo em si!

  Por exemplo: não meça o tempo em que os dados de entrada do algoritmo são definidos!

- Caso seja necessário repetir a execução do algoritmo, lembre-se de usar os mesmos dados de entrada em todas repetições;