1º Trabalho Prático de Programação Concorrente EC 2024/1

Desenvolva uma aplicação paralela com espaço de endereçamento compartilhado, em C/OpenMP no Linux para resolver um sistema linear (***Ax=b***), segundo o Método Iterativo de Jacobi-Richardson (também conhecido como Gauss-Jacobi).

Para execução, a aplicação deve ser iniciada da seguinte forma:

jacobipar <N> <T><seed>

onde, ***jacobipar*** é o nome do arquivo binário para execução, ***<N>*** determina a ordem da matriz ***A[N,N]*** usada para armazenar os coeficientes do sistema, e o tamanho do vetor ***B[N]*** usado para os conter os termos constantes, ***<T>*** determina a quantidade de *threads* que serão usadas ao todo na aplicação, e ***<seed>*** determina a semente para a geração pseudoaleatória dos valores usados em ***A[N,N]*** e ***B[N]***. As gerações dos números pseudoaleatórios são feitas pela *thread* 0. Garanta a convergência do método ao criar os valores para ***A[N,N]***.

O resultado da aplicação será determinado pelos valores do vetor ***X*** que satisfaçam as ***N*** equações simultaneamente. Para finalizar, a *thread* 0 deverá perguntar qual equação do sistema o usuário deseja escolher para associar os resultados obtidos com o vetor ***X*** e, assim, comparar o resultado obtido com o valor correspondente no vetor ***B***.

Diferentes livros de Cálculo Numérico descrevem a solução do Método Iterativo de Jacobi-Richardson (ou Gauss-Jacobi). Em [1] há uma boa explicação do mesmo.

Desenvolva a aplicação concorrente em C/OpenMP segundo as diretrizes passadas em sala de aula. Sua solução deve: (1) determinar e explorar o paralelismo da aplicação de maneira flexível, (2) usar obrigatoriamente ***parallel***, ***for***, ***task***, ***reduction*** e ***simd***, e (3) com estas diretivas maximizar o ganho de desempenho em relação ao algoritmo sequencial, dados os valores de ***N***, ***T*** e ***seed***.

Execute as versões paralela e sequencial desenvolvidas com 03 valores diferentes de ***N*** e, cada um destes, com 03 valores diferentes de ***T***, no mínimo 30 vezes cada, coletando os tempos de resposta (*turnaround time*) com o utilitário *time* do Linux. Monte uma tabela com esses resultados, organizando os resultados para ***N*** e ***T***, medianas, médias, desvios padrão, *speedups* e eficiências. Monte gráficos, e analise e explique seus resultados.

Submeta no e-disciplinas um arquivo compactado (***padrão zip***) contendo: o código fonte sequencial (deve se chamar ***jacobiseq.c***), código fonte em C/OpenMP (deve se chamar ***jacobipar.c***), *makefile* para compilação e execução dos códigos, e um relatório final com os resultados (deve-se chamar ***resultados.pdf***) contendo a tabelas, gráficos e análise do *speedup* e eficiência. Coloque em todos os arquivos os nomes dos integrantes do grupo que participaram de fato do desenvolvimento do trabalho.

Questões omissas e/ou ambíguas serão fixadas pelo professor. Para saná-las, entre em contato para a orientação adequada.

Referências

[1] Ruggiero, Márcia A.G.; Lopes, Vera L.R. Cálculo Numérico: Aspectos Teóricos e Computacionais. McGraw-Hill, 1988.