Simulação e Análise de Modelos de Difusão de Contaminantes em Água

Amanda[1], **Éric F. C. Yoshida**[1], **Henrique C. Garcia**[1]

[1]Instituto de Ciência e Tecnologia – Universidade Federal de São Paulo (UNIFESP)
São José dos Campos – SP – Brazil

..@unifesp.br, eric.fadul@unifesp.br, henrique.garcia@unifesp.br

Abstract. *This report discusses the simulation of the diffusion equation using the finite difference method and parallelization with OpenMP. Initially, the study of the diffusion model is presented, highlighting the importance of the transient diffusion equation in describing the spread of contaminants in bodies of water. Subsequently, the configuration of the simulation environment is detailed, including the definition of the two-dimensional grid, the parameters used such as the diffusion coefficient DD, boundary and initial conditions, and the number of time iterations set at 1000. Finally, the implementation of the simulation on a CPU using OpenMP is described to parallelize computations, optimize performance, and reduce execution time. The parallelized code is presented and analyzed, highlighting the improvements achieved compared to the sequential version.*

Resumo. *Este relatório aborda a simulação da equação de difusão utilizando o método das diferenças finitas e paralelização com OpenMP. Inicialmente, apresenta-se o estudo do modelo de difusão, destacando a importância da equação de difusão transiente para descrever o espalhamento de contaminantes em corpos d'água. Em seguida, detalha-se a configuração do ambiente de simulação, incluindo a definição da grade bidimensional, os parâmetros utilizados como o coeficiente de difusão DD, as condições de contorno e iniciais, além do número de iterações temporais estabelecido em 1000. Por fim, descreve-se a implementação da simulação em CPU utilizando OpenMP para paralelizar os cálculos, otimizar o desempenho e reduzir o tempo de execução. O código paralelizado é apresentado e analisado, evidenciando as melhorias obtidas em relação à versão sequencial.*

1. Estudo do Modelo de Difusão

A equação de difusão é fundamental para descrever o processo pelo qual substâncias, como contaminantes em um corpo d'água, se espalham ao longo do tempo e do espaço. A equação de difusão transiente em duas dimensões é dada por:

$$\frac{\partial C}{\partial t} = D \cdot \nabla^2 C \tag{1}$$

onde:

- C é a concentração do contaminante,

- t é o tempo,
- D é o coeficiente de difusão,
- $\nabla^2 C$ representa a taxa de variação da concentração no espaço.

Para resolver numericamente esta equação, utilizamos o método das diferenças finitas, que discretiza as derivadas parciais em diferenças finitas sobre uma grade bidimensional. Isso permite aproximar as variações contínuas no espaço e no tempo por meio de cálculos em células discretas, onde cada célula atualiza seu valor com base nos valores das células vizinhas em cada iteração temporal.

2. Configuração do Ambiente e Parâmetros da Simulação

Para resolver numericamente a Equação de Difusão, utilizamos a aproximação por diferenças finitas centrais, expressa pela seguinte fórmula:

$$C_{i,j}^{t+1} = C_{i,j}^{t} + D\Delta t \left(\frac{C_{i+1,j}^{t} + C_{i-1,j}^{t} + C_{i,j+1}^{t} + C_{i,j-1}^{t} - 4C_{i,j}^{t}}{\Delta x^2} \right) \tag{2}$$

Grade 2D:

- A grade consiste em $N \times N$ células, com $N = 2000$.
- Cada célula representa a concentração de contaminantes em uma região específica do corpo d'água.

Parâmetros da Simulação:

- **Coeficiente de difusão (D):** $D = 0{,}1$.
- **Condições de contorno:** As bordas da grade são tratadas como barreiras onde o contaminante não se propaga (condições de contorno de Neumann homogêneas).
- **Condições iniciais:** Todas as células iniciam com concentração zero, exceto a região central, que possui uma alta concentração (por exemplo, $C[N/2][N/2] = 1{,}0$).
- **Número de iterações temporais:** $T = 1000$ iterações.
- **Passos espaciais e temporais:**
  - **Passo espacial (Δx):** $\Delta x = 1{,}0$.
  - **Passo temporal (Δt):** $\Delta t = 0{,}01$.

3. Implementação com OpenMP (Simulação Local em CPU)

Para simular a difusão de forma eficiente, utilizamos a biblioteca OpenMP para paralelizar os cálculos nas múltiplas unidades de processamento da CPU. A paralelização é aplicada principalmente nos loops que atualizam as concentrações das células, distribuindo o trabalho entre os núcleos disponíveis.

3.1. Código não paralelizado

O código original sem paralelização está disponível em: codigo_nao_paralelizado.c.

3.2. Implementação Paralela com OpenMP

Para paralelizar o código, adicionamos diretivas do OpenMP nos loops intensivos em computação. O código modificado esta disponível em: codigo_paralelizado.c

3.3. Explicação das Modificações

1. **Inclusão da Biblioteca OpenMP:**
   - Adicionamos `#include <omp.h>` para utilizar as funcionalidades do OpenMP.
2. **Paralelização dos Loops:**
   - Utilizamos `#pragma omp parallel for collapse(2)` antes dos loops que atualizam `C_new` e que inicializam as matrizes. A cláusula `collapse(2)` indica que os dois loops aninhados devem ser tratados como um único loop para fins de paralelização.
   - No cálculo de `difmedio`, adicionamos `reduction(+:difmedio)` para assegurar que a redução seja realizada corretamente em ambiente paralelo.
3. **Gerenciamento de Memória:**
   - Incluímos a liberação de memória alocada dinamicamente ao final do programa para evitar vazamentos de memória.

3.4. Resultados Esperados

Com a paralelização, esperamos uma redução significativa no tempo de execução da simulação, proporcional ao número de núcleos disponíveis na CPU. A eficiência da paralelização dependerá também da carga de trabalho e da sobrecarga introduzida pela gestão das threads.

3.5. Considerações sobre a Paralelização

- **Sincronização e Condições de Corrida:**
  - Como cada célula é atualizada com base em valores da iteração anterior e não há dependências entre as células sendo atualizadas na mesma iteração, a paralelização é segura e não introduz condições de corrida.
- **Escalabilidade:**
  - A escalabilidade pode ser limitada pela largura de banda da memória, especialmente em sistemas com muitos núcleos de processamento.
- **Overhead:**
  - O uso de `collapse(2)` e a paralelização de loops externos reduzem o overhead associado à criação e sincronização de threads.

4. Conclusão

A simulação da equação de difusão utilizando diferenças finitas e paralelização com OpenMP demonstra a eficiência computacional que pode ser alcançada em simulações numéricas intensivas. Ao distribuir o cálculo entre múltiplos núcleos, é possível acelerar significativamente o processamento, tornando viável a simulação de sistemas de grande escala em tempo razoável.

Esta abordagem pode ser estendida para simulações mais complexas, incluindo:

- Difusão em meios heterogêneos com coeficientes de difusão variáveis.
- Inclusão de termos de reação para modelar processos químicos ou biológicos.
- Adaptação para grades tridimensionais em estudos volumétricos.

A compreensão e implementação correta da paralelização são essenciais para aproveitar ao máximo os recursos computacionais disponíveis e obter resultados eficientes e precisos em simulações científicas.

5. Referências

[1] Chapra, S. C., & Canale, R. P. (2011). *Métodos Numéricos para Engenharia* (7ª ed.). McGraw-Hill.

[2] Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing* (3ª ed.). Cambridge University Press.

[3] OpenMP Architecture Review Board. (2018). *OpenMP Application Programming Interface Version 5.0*. Disponível em `https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5.0.pdf`