

PARALELIZAÇÃO DE CÓDIGO COMPUTACIONAL PARA REPRESENTAÇÃO DE ENXURRADAS: CPU E GPU

Mailon Roversi, Leonardo Romero Monteiro

INTRODUÇÃO

Nas últimas décadas, o uso de *softwares* de simulação de fenômenos físicos tem se consolidado como uma ferramenta essencial em diversas áreas da Engenharia, esses recursos permitem compreender e analisar situações próximas às reais, complementando estratégias como a dos modelos físicos reduzidos, tornando o processo mais ágil, eficiente e detalhado (Carvalho *et al*, 2015). Entre esses *softwares*, destaca-se o SuLi (Simulador de Escoamentos com Superfície Livre) (Monteiro, 2018; Soares, 2024), fundamentado nas equações de Navier-Stokes e na equação da continuidade para fluidos incompressíveis e multifásicos. O objetivo do presente trabalho foi otimizar a paralelização já existente no SuLi, ampliar sua aplicação em trechos ainda não paralelizados e implementar o uso de GPU, permitindo também a alternância entre execuções em CPU e GPU.

DESENVOLVIMENTO

O primeiro passo consistiu em identificar o método de paralelização mais adequado para cada parte do código, que neste caso foi considerado como sendo o OpenMP (2024), sendo esse programa incluso na própria instalação do Gfortran, pela sua facilidade de implementação em Fortran 90, a linguagem que o SuLi utiliza. Para garantir a precisão dos resultados, a paralelização foi implementada de forma gradual, sempre validando os resultados do programa com condições de contorno padronizadas previamente definidas, e tomadas como padrão. A cada implementação identificou se os resultados da simulação se alteravam. Como este é um código determinístico, espera-se que as rodadas sejam idênticas quando se usa os mesmos parâmetros de entrada, condições iniciais e condições de contorno.

A paralelização em GPU seguiu uma metodologia semelhante à utilizada para a CPU, mas exigiu ajustes específicos no processo de compilação, de modo a assegurar o correto aproveitamento dos recursos gráficos, foi instalado o compilador Nvfortran parte do NVIDIA HPC SDK Fortran (Nvidia, 2025), que oferece uma ampla gama de sinalizadores para otimização, depuração e direcionamento para arquiteturas específicas. Por conta da mudança no processo de compilação, houve pequena variação nos resultados obtidos, provocada pela diferença dos compiladores na execução dos cálculos.

Para a alternância em CPU e GPU, foi definida uma condição de compilação que permite escolher a unidade de processamento que será adotada, de forma simples e flexível, garantindo a execução conforme a demanda computacional e o hardware disponível. Quando o código é compilado apenas com o comando make, a execução ocorre na CPU, e as linhas de código situadas entre “`#ifdef USE_GPU`” e “`#else`” são ignoradas. Já ao adicionar a definição `USE_GPU=1` no momento da compilação (Figura 1), as instruções entre “`#ifdef USE_GPU`” e “`#else`” são interpretadas, enquanto aquelas entre “`#else`” e “`#endif`” são descartadas, direcionando a execução para a GPU.

A implementação da alternância entre CPU e GPU foi um dos pontos mais críticos, pois exigiu a inserção de estruturas condicionais (*if/else*) em trechos de interesse para a paralelização

do código. Essa solução possibilitou maior versatilidade ao *software*, permitindo que o usuário defina a arquitetura de execução de acordo com as necessidades do estudo e os recursos computacionais disponíveis.

Durante toda a pesquisa foi utilizado um único computador com o sistema operacional Ubuntu 24.04, com processador Intel Core i5-13500 de 14 núcleos, dos quais 6 são de performance, sendo esse o número de núcleos adotado como padrão nesse trabalho e placa de vídeo NVIDIA RTX 5000 Ada Generation.

RESULTADOS

Para a avaliação dos resultados, foram realizadas simulações em dois casos com condições de contorno distintas. O Caso 1 corresponde ao cenário utilizado como padrão para validar a otimização do código, consiste em um cubo preenchido parcialmente com água que sofre o efeito de seiche no nível da água, enquanto o Caso 2 representa a simulação de uma quebra de barragem em que os fluidos envolvidos eram água e ar.

Os resultados estão sintetizados na Tabela 1, considerada a partir de três simulações para cada configuração, sendo apresentados os valores médios para cada conjunto de caso e configuração. Essa abordagem busca minimizar eventuais variações pontuais e garantir maior confiabilidade aos dados obtidos. No Caso 1, a execução em CPU otimizada apresentou uma redução de 16% no tempo de processamento em comparação com a CPU sem modificações. Já a execução em GPU demonstrou uma redução ainda mais significativa, de 40% em relação ao mesmo parâmetro de referência. No Caso 2, os ganhos foram mais expressivos, a CPU otimizada reduziu o tempo de execução em 23%, enquanto a GPU apresentou uma redução de 51%, ambos em comparação à CPU sem modificações.

A paralelização via GPU apresentou as maiores reduções de tempo. O fenômeno representado no Caso 2 tem maior complexidade e apresentou ganhos de desempenho mais significativos. Toda a pesquisa foi realizada usando um único computador e número de núcleos constante, um melhor desempenho da CPU poderia ser obtido utilizando um processador com um número maior de núcleos de performance.

CONSIDERAÇÕES FINAIS

Tanto no Caso 1 quanto no Caso 2, observou-se redução expressiva no tempo de execução, validando a metodologia adotada e a importância da otimização realizada.

A implementação da paralelização em GPU apresentou os maiores ganhos de desempenho, especialmente em cenários de maior complexidade computacional, confirmando seu potencial como alternativa mais eficiente para execuções intensivas. Por outro lado, a paralelização em CPU também contribuiu de forma consistente, oferecendo melhorias relevantes, apesar das limitações de *hardware*.

De forma geral, o trabalho evidenciou que a estratégia de paralelização proposta amplia a flexibilidade e eficiência do *software*, possibilitando sua aplicação em diferentes contextos de simulação com maior rapidez e garantindo a confiabilidade nos resultados.

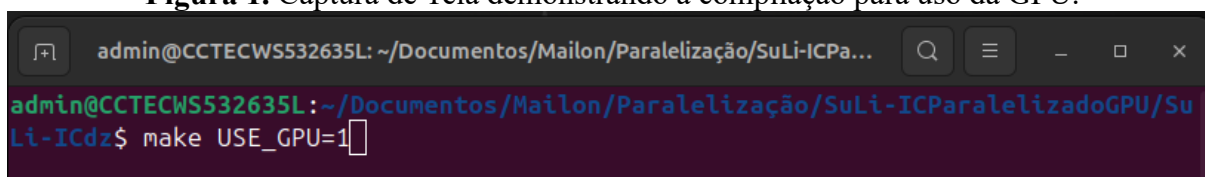
Palavras-chave: Otimização de código; Fortran 90; Dinâmica dos fluidos computacional; Escoamento com superfície livre; Equações de Navier-Stokes.

ILUSTRAÇÕES

Tabela 1. Tempo médio em minutos em cada caso e configuração.

CASO/CONFIGURAÇÃO	CPU sem modificações	CPU Otimizado	GPU
Caso 1	12,7	10,7	7,6
Caso 2	32,8	25,3	15,9

Fonte: O próprio autor, 2025

Figura 1. Captura de Tela demonstrando a compilação para uso da GPU.

Fonte: O próprio autor, 2025

REFERÊNCIAS BIBLIOGRÁFICAS

CARVALHO, N. J. S. ; NASCIMENTO, E. O. ; ALMEIDA, E. M. P. ; FIGUEIRA, K. S. ; GÔES, J. F. . **Importância da Modelagem Computacional na Engenharia**. 2015. Disponível em: <https://www.ufopa.edu.br/anaisdajornada/3/resumo/97/importancia-da-modelagem-computacional-na-engenharia>. Acesso em: 26 ago. 2025.

MONTEIRO, Leonardo Romero. **Simulação Numérica Direta de Alta Ordem para Escoamentos Bifásicos Água-ar**. Tese (Doutorado) – Universidade Federal do Rio Grande do Sul, Instituto de Pesquisas Hidráulicas, Programa de Pós-Graduação em Recursos Hídricos e Saneamento Ambiental, Porto Alegre, 2018.

NVIDIA. **HPC Compilers User's Guide**, 2025. Disponível em: <https://docs.nvidia.com/hpc-sdk/compilers/hpc-compilers-user-guide/index.html>. Acesso em: 26 ago. 2025.

OpenMP. **Reference Guide**, 2024. Disponível em: <https://www.openmp.org/resources/refguides/>. Acesso em: 26 ago. 2025.

SOARES, Bruna Fernanda. **Modelagem de Turbulência de Escoamentos em Canais de Fundo Rugoso**. Dissertação (Mestrado) – Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Programa de Pós-Graduação em Engenharia Civil, Joinville, 2024.

DADOS CADASTRAIS

BOLSISTA: Mailon Roversi

MODALIDADE DE BOLSA: PROBIC/UDESC (IC)

VIGÊNCIA: 09/2024 a 08/2025 – Total: 12 meses

ORIENTADOR(A): Leonardo Romero Monteiro

CENTRO DE ENSINO: Centro Ciências Tecnológicas CCT

DEPARTAMENTO: Departamento de Engenharia Civil CCT

ÁREAS DE CONHECIMENTO: Engenharias/Engenharia Civil

TÍTULO DO PROJETO DE PESQUISA: Estudo da Concepção, Dinâmica e Modelagem de Enxurradas e Fluxos Hiperconcentrados

Nº PROTOCOLO DO PROJETO DE PESQUISA: NPP4209-2023