Utilização do PaScal Suite na Análise de Escalabilidade de um Código de Inversão de Forma de Onda Completa

Julia A. S. dos Santos[1], João B. Fernandes [1], Samuel Xavier-de-Souza[1]

[1] Universidade Federal do Rio Grande do Norte (UFRN)

juliaalanne@ufrn.edu.br, joao.fernandes.highres@imd.ufrn.br,

samuel@dca.ufrn.br

Resumo:

A Inversão da Forma de Onda Completa *(FWI) é uma técnica utilizada para estimar o modelo de velocidades do subsolo a partir de dados sísmicos, apresentando alta demanda computacional devido à sua natureza iterativa e aos cálculos intensivos envolvidos. Diante disso, garantir a escalabilidade dessa aplicação é essencial para viabilizar sua execução em ambientes de alto desempenho. Neste trabalho, foram realizados testes de escalabilidade com o uso da ferramenta PaScal Suite, a fim de identificar gargalos de desempenho em um código de FWI implementado em C++. A análise permitiu mapear regiões críticas do código cuja eficiência é impactada quando há aumento dos recursos computacionais sem o correspondente crescimento no tamanho do problema. Os resultados obtidos são relevantes para futuras otimizações na melhoria da escalabilidade da aplicação.*

Palavras chaves: *Escalabilidade; Eficiência; FWI; Análise.*

1. Introdução

A Inversão da Forma de Onda Completa (FWI – *Full Waveform Inversion*) é uma técnica geofísica utilizada para estimar o modelo de velocidades do subsolo a partir de dados sísmicos registrados na superfície [Tarantola 1984]. Apesar do elevado potencial de acurácia, a FWI impõe um alto custo computacional, devido à natureza iterativa do processo de otimização e à complexidade dos cálculos envolvidos na propagação direta e retropropagação das ondas sísmicas [Li e Sun 2024].

Dessa forma, é fundamental que aplicações como a FWI, que exigem alto poder computacional, sejam escaláveis, garantindo eficiência e viabilidade em ambientes de alto desempenho. Nesse contexto, a realização de testes de escalabilidade é essencial não só para avaliar o comportamento do código diante do aumento dos recursos computacionais, mas também para identificar gargalos e limitações que possam ser otimizados, garantindo uma utilização mais eficiente dos recursos disponíveis. Trabalhos anteriores, como o de Santos-da-Silva et al. [2024], demonstram a relevância dessa análise em aplicações de FWI, evidenciando como a escalabilidade impacta diretamente na eficiência computacional. Assim, ferramentas que auxiliem esse processo tornam-se fundamentais.

Com esse objetivo, o conjunto de ferramentas *Parallel Scalability Suite* (PaScal Suíte) foi desenvolvido para auxiliar os desenvolvedores na realização de testes de escalabilidade. O PaScal Suíte conta com as ferramentas PaScal Analyzer e PaScal Viewer.

Considerando a importância da realização de testes de escalabilidade, o presente trabalho tem como objetivo analisar a escalabilidade de um código de FWI implementado em C++, utilizando as ferramentas do PaScal Suite, investigando seu desempenho sob diferentes

configurações de paralelização no ambiente de supercomputação do Núcleo de Processamento de Alto Desempenho (NPAD).

2. *Full Waveform Inversion - FWI*

A FWI é uma técnica que busca estimar as velocidades de propagação das ondas sísmicas, a partir da minimização da diferença entre os dados observados e os dados sintéticos simulados. Esse processo é formulado como um problema de otimização não linear, sensível à qualidade do modelo inicial e à fidelidade da propagação direta [Tarantola 1984]. Dessa forma, ela retorna um modelo de velocidades que gera dados sintéticos altamente semelhantes aos dados reais. A FWI pode ser definida como:

$$min_m \, \Phi(m) = \frac{1}{2} ||d_{obs} - G(m)||^2 \quad (1)$$

onde m representa os parâmetros do modelo (por exemplo, o campo de velocidades), d_{obs} são os dados sísmicos observados e $G(m)$ são os dados sintéticos gerados por meio da simulação da propagação de onda. A função objetivo $\Phi(m)$ quantifica o erro entre os dados medidos e os simulados, e sua minimização é realizada por métodos iterativos, com base no cálculo do gradiente.

No cálculo do gradiente da função objetivo, é utilizado frequentemente o método adjunto, que envolve a solução da equação de onda duas vezes por iteração: uma propagação direta e uma retropropagação adjunta. Essa característica torna a FWI extremamente custosa do ponto de vista computacional, especialmente em grandes modelos ou em inversões em 3D [Virieux e Operto. 2009]. Devido à complexidade da FWI, faz-se necessário o uso de estratégias avançadas de paralelização e gerenciamento de memória.

Neste trabalho, o objeto de estudo é a FWI implementada no software Mamute, desenvolvido em C++ com o uso de técnicas de paralelização para otimização do desempenho, conforme descrito por Fernandes et al. [2025].

3. PaScal Suite

O conjunto de ferramentas PaScal Suite, composto pelo PaScal Analyzer e pelo PaScal Viewer, foi desenvolvido para auxiliar os desenvolvedores nas análises de escalabilidade.

O PaScal Analyzer é compatível com programas escritos em C e C++ e permite a coleta automatizada de dados a partir de múltiplas execuções, o que facilita a comparação entre diferentes configurações de execução [da Silva et al. 2022]. A ferramenta também permite a variação de parâmetros definidos pelo usuário, ampliando a abrangência das análises de desempenho. Além disso, oferece suporte à instrumentação automática e manual, proporcionando flexibilidade na análise de diferentes trechos do código.

O PaScal Viewer é uma aplicação web desenvolvida para facilitar a visualização de métricas de desempenho de programas paralelos [da Silva et al. 2019]. Ele recebe como entrada os arquivos gerados pelo PaScal Analyzer e os apresenta por meio de representações visuais intuitivas. Essa abordagem permite identificar a eficiência do programa como um todo ou de regiões específicas do código. A interface do PaScal Viewer também possibilita a análise hierárquica de zonas paralelas, permitindo um perfilamento mais detalhado do código.

4. Experimentos

Os experimentos foram conduzidos em um nó do supercomputador pertencente ao NPAD. O nó utilizado conta com duas CPUs AMD EPYC 7713 de 2.0GHz, totalizando 128 núcleos de processamento, além de 512 GB de memória RAM.

Para as simulações, foram consideradas três dimensões espaciais (x_1, x_2, e x_3), com os respectivos números de amostras n_1, n_2, e n_3. Os parâmetros da modelagem foram mantidos

constantes em todos experimentos: frequência de pico de 5Hz, intervalo de amostragem temporal de 1ms, resoluções espaciais uniformes de $\Delta x_1 = \Delta x_2 = \Delta x_3 = 25$m e uma espessura de borda de 25 pontos da grade em todas as direções da malha tridimensional.

Foram utilizados quatro modelos de velocidades, denominados m_1, m_2, m_3 e m_4, composto por malhas de (n_1, n_2, n_3) = (25, 80, 187), (50, 80, 187), (100, 80, 187) e (200, 80, 187) pontos, respectivamente. O modelo m4 foi construído utilizando uma esfera com perturbação gaussiana, onde as camadas superiores possuem velocidades de 2500m/s e seu núcleo tem velocidade máxima de 6000 m/s. Os modelos m_1 a m_3 são cortes do modelo m_4 na dimensão x_1.

O PaScal Analyzer foi utilizado para o perfilamento do código, avaliando a escalabilidade da aplicação com combinações de 1, 2, 4, 8, 16, 32 e 64 núcleos de processamento, utilizando os quatro modelos de entrada (m_1 a m_4), e repetindo cada configuração três vezes. Além disso, foi aplicada a instrumentação manual para medir o tempo de execução em regiões específicas do algoritmo, com o objetivo de identificar possíveis gargalos da aplicação. Assim, duas regiões identificadas como críticas, a primeira corresponde ao trecho responsável pelo cálculo da função objetivo e gradiente, enquanto a segunda é uma sub-região da primeira, responsável pela leitura dos dados. Estas foram delimitadas por meio das funções pascal_start(*id*) e pascal_stop(*id*) identificando o início e o fim, respectivamente, de cada trecho monitorado.

5. Resultados

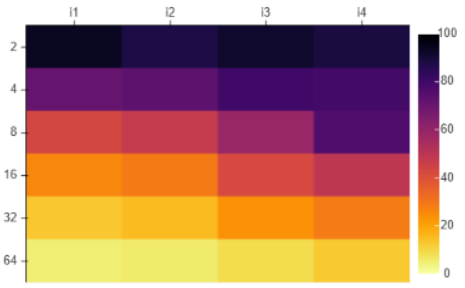
Na figura 1a observa-se que, até quatro núcleos, o código mantém uma boa escalabilidade, com eficiência próxima ao ideal. A partir disso, a eficiência diminui à medida que o número de núcleos aumenta, indicando sobrecarga paralela, especialmente quando o tamanho do problema não acompanha o crescimento dos recursos. Entretanto, como essa análise considera a execução completa da aplicação, é possível que diferentes regiões do código apresentem comportamentos distintos em relação à escalabilidade.

Na Figura 1b, observa-se uma eficiência de 100% para todos os tamanhos de problema quando executada com dois núcleos. No entanto, conforme aumentamos os recursos computacionais para os modelos menores (m_1 e m_2), a eficiência tende a cair, evidenciando sobrecarga paralela. Por outro lado, para modelos maiores (m_3 e m_4), essa região mantém uma eficiência próxima de 100% até 64 núcleos, indicando um bom aproveitamento do paralelismo quando há volume de dados suficiente.

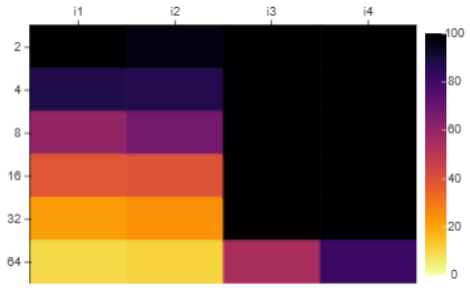
A Figura 1c mostra que a região crítica 2 é mais sensível à variação de recursos. A eficiência cai drasticamente ao aumentar o número de núcleos para modelos pequenos. Já para os modelos m_3 e m_4, a perda de eficiência é mais gradual. Um ponto positivo é que essa região atinge uma boa eficiência com dois núcleos no modelo m_4, demonstrando que, em determinadas configurações, ela consegue explorar bem o paralelismo. Esses resultados destacam a importância de se analisar o comportamento de regiões específicas do código, uma vez que a escalabilidade global pode esconder gargalos locais que afetam diretamente a eficiência da aplicação.

6. Conclusão

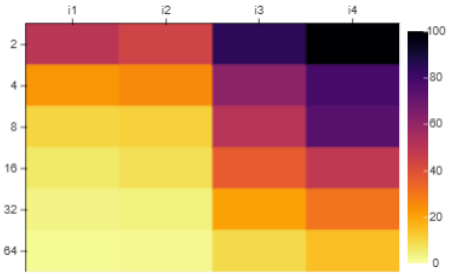
A análise realizada com a ferramenta PaScal Suite foi eficaz na avaliação da escalabilidade da aplicação. Por meio dos testes, foi possível identificar gargalos de desempenho e localizar regiões críticas que comprometem a eficiência. Com base nesses resultados, os próximos passos consistem na otimização das regiões identificadas e, posteriormente, na realização de novos testes de desempenho, com o objetivo de verificar se as otimizações implementadas melhoraram a escalabilidade da aplicação.



(a) Execução completa



(b) Região crítica 1



(c) Região crítica 2

Figura 1. Diagrama de eficiência da FWI. O número de núcleos varia entre 1, 2, 4, 8, 16, 32 e 64 no eixo vertical. O tamanho do problema varia conforme os modelos de i1 a 14 horizontal. Cada célula no diagrama tem uma média de três amostras.

Referências

da Silva, A. B. N., Cunha, D. A. M., Silva, V. R. G., de A. Furtunato, A. F., and Xavier-de Souza, S. (2019). Pascal viewer: A tool for the visualization of parallel scalability trends. In Bhatele, A., Boehme, D., Levine, J. A., Malony, A. D., and Schulz, M., editors, *Programming and Performance Visualization Tools*, pages 250–264, Cham. Springer International Publishing.

da Silva, V. R. G., da Silva, A. B. N., Valderrama, C., Manneback, P., and Xavier-de Souza, S. (2022). A minimally intrusive approach for automatic assessment of parallel performance scalability of shared-memory hpc applications. Electronics, 11(5).

Fernandes, J. B., Oliveira, A. D., Silva, M. C., Santos-da-Silva, F. H., Rodrigues, V. H., Schneider, K. A., ... & Xavier-de-Souza, S. (2025). Mamute: high-performance computing for geophysical methods. *arXiv preprint arXiv:2502.12350*.

Li, Shizhong; Sun, Chengyu. 3D numerical simulation of frequency-domain elastic wave equation with a compact second-order scheme. Journal of Applied Geophysics, v. 223, p. 105348, 2024.

Santos-da-Silva, Felipe H. et al. Análise de Escalabilidade em um Código de Inversão de Forma de Onda Completa. In: Simpósio em Sistemas Computacionais de Alto Desempenho (SSCAD). SBC, 2024. p. 121-132.

Tarantola, A. (1984). Inversion of seismic reflection data in the acoustic approximation. Geophysics, 49(8):1259–1266.

Virieux, Jean; Operto, Stéphane. An overview of full-waveform inversion in exploration geophysics. **Geophysics**, v. 74, n. 6, p. WCC1-WCC26, 2009.