Técnicas Computacionais de Alto Desempenho na manipulação de Malhas de Elementos Finitos

Guilherme M. F. Silva[1], **José J. Camata**[1]

[1]Universidade Federal de Juiz de Fora (UFJF)
Caixa Postal 20.010 – 36.036-230 – Juiz de Fora – MG – Brazil.

{guilhermemachado,camata}@ice.ufjf.br

Abstract. *Several physical phenomena and/or problems in engineering and science are modeled by partial differential equations. These equations can be solved using numerical methods, such as finite differences, finite elements and finite volumes. In common, these methods required some form of discretization of the problem domain, that is, to determine specific points of the domain where the solution of the differential equation will be determined. This discretization can be based on Cartesian railing (finite differences) or by subdividing the domain into discrete elements (finite elements and volume). In this work, we will focus on discretizations (meshes) specific to the finite element method and the development of techniques for manipulating meshes aimed at high performance computing.*

Resumo. *Diversos fenômenos físicos e/ou problemas da Engenharia e ciências são modelados por equações diferenciais parciais. Essas equações podem ser solucionadas através de métodos numéricos, tais como, diferenças finitas, elementos finitos e volumes finitos. Em comum, esses métodos requerem alguma forma de discretização de domínio do problema, ou seja, determinar pontos específicos do domínio onde a solução da equação diferencial será calculada. Essa discretização pode ser baseada em gradeamento cartesiano (diferenças finitas) ou através de subdivisões do domínio em elementos discretos (elementos e volume finitos). Neste trabalho, focaremos em discretizações (malhas) específicas para o método de elementos finitos e o desenvolvimento de técnicas para manipulação de malhas voltadas para a computação de alto desempenho.*

1. Introdução

O Método dos Elementos Finitos é uma técnica amplamente utilizada para solução computacional de equações diferenciais parciais [Hughes 2012]. Para a aplicação desse método é necessário a criação de um modelo tridimensional que represente suficientemente o domínio do problema, o qual será chamado de malha. Para malha entende-se uma estrutura tridimensional discreta dividida em estruturas nodais e elementares para aplicação do método numérico. Os elementos representam a parte contínua do domínio e esses são interconectados por pontos chamados de nós, os quais ambos podem carregar informações para os cálculos computacionais. Dessa forma, para a obtenção de resultados mais precisos é feito o refinamento da malha, ou seja, discretizações mais constantes pelo domínio, incrementando o número de elementos e nós, porém, tornando maior o custo computacional. Assim, para ser obtida uma execução em tempo adequado são visadas técnicas de computação de alto desempenho.

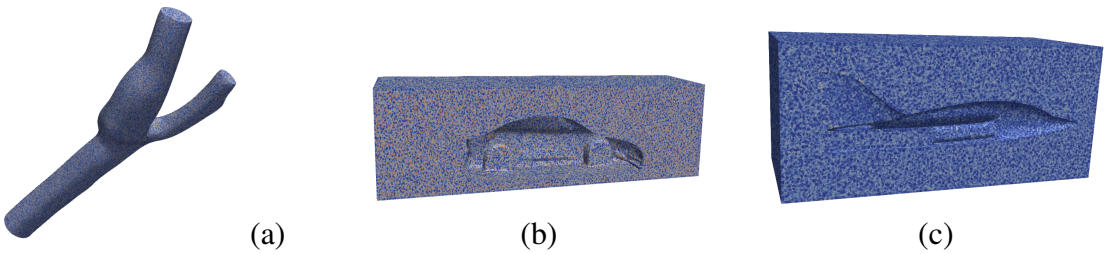
2. Técnicas de Computação de Alto Desempenho

O núcleo computacional principal de qualquer simulador de elementos finitos baseia-se na montagem e solução de um sistema de equações lineares e/ou não lineares. A implementação otimizada dessas duas fases ocasiona ganhos globais de desempenho da aplicação. Com o advento da computação em larga escala, permitiu-se obter soluções em resoluções inimagináveis até pouco tempo atrás. Para tanto, levando em consideração a qualidade da solução numérica, malhas cada vez mais refinadas são empregadas. Sendo assim, a aplicação de esquemas de paralelismo de memória compartilhada (OpenMP) [Dagum and Menon 1998] e distribuída (MPI) [Forum 1994] devem ser consideradas. Neste trabalho, usa-se as seguintes técnicas de alto desempenho:

1. **Particionamento do domínio.** Para que seja possível a aplicação de paralelismo com memória distribuída é necessário particionar o domínio do problema em conjuntos físicamente próximos, ou seja, dividir a malha em partes menores e distribuir cada parte para um processador do sistema paralelo. É utilizado então, para esse intuito, a biblioteca METIS [Karypis and Kumar 1998]. METIS é uma biblioteca de particionamento de grafos e malhas além de possuir rotinas para a otimização de largura de banda em matrizes esparsas.
2. **Coloração.** A técnica de coloração da malha possibilita a aplicação do paralelismo com memória compartilhada com multithreads. Para uma execução segura, utilizando diferentes threads, é necessário contornar a condição de corrida, de forma que nenhuma thread tente manipular simultaneamente um mesmo endereço de memória, o que pode causar inconsistência nos cálculos. Para a aplicação desta técnica é utilizado do algoritmo *First-Fit Coloring* [Rokos et al. 2015] o qual calcula a menor cor possível para o elemento a partir da informação das cores de seus elementos vizinhos.
3. **Reordenação Nodal.** A reordenação nodal se trata de uma técnica de reorganizar os nós da malha de forma que os elementos tenham conectividades numericamente mais próximas, aumentando a eficiência da busca de informações na memória principal pelo aprimoramento da localidade dos dados, reduzindo assim, o erro de acesso a cache e melhorando o desempenho global da aplicação. Para a aplicação da reordenação foram utilizados dois algoritmos, o Reverse Cuthill McKee (RCM) [Burkardt 2003] e o METIS_ND disponibilizado pela biblioteca METIS [Karypis and Kumar 1998], o qual aplica o algoritmo Multilevel Nested Dissection.

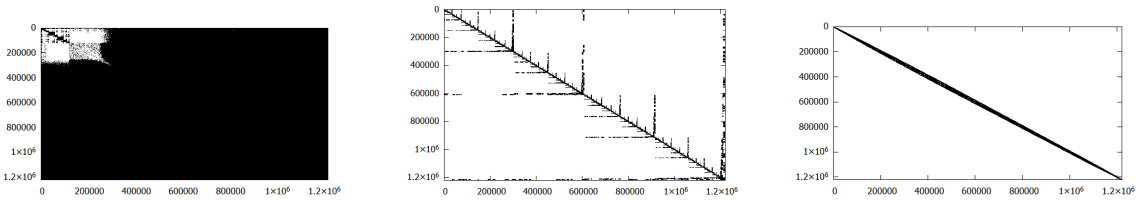
3. Resultados e Conclusões

Para os testes foram utilizadas três malhas realísticas as quais são aplicáveis aos estudos de Mecânica dos Fluidos. A primeira representa uma estrutura de uma artéria, a qual pode ser utilizada no estudo de doenças cardiovasculares, e as outras duas aplicáveis no estudo da aerodinâmica, sendo uma de um carro e a outra de um avião. A Figura 1 mostra os resultados da aplicação do algoritmo de coloração para essas três malhas. Foi observado que a coloração gerou a quantidade mínima de cores de modo que os elementos marcados com a mesma cor possam ser acessados independetemente em uma aplicação OpenMP. Porém foi observado que a quantidade total de elementos por cor não obteve uma distribuição homogênea. Uma possibilidade é não restringir ao números mínimo de cores mas ao número mínimo de elementos por cor. Tomando como exemplo, a malha do



(a) (b) (c)

Figura 1. Coloração da Malha. (a) Artéria com 8361025 elementos e 44 cores (b) Carro com 10323550 elementos e 46 cores. (c) Avião com 7282234 elementos e 86 cores.



(a) (b) (c)

Figura 2. (a) Matriz de adjacência referente a malha do avião antes da reordenação nodal. (b) Matriz de adjacência referente a malha do avião após a aplicação do algoritmo ND.(c) Matriz de adjacência referente a malha do avião após a aplicação do algoritmo RCM.

avião, é possível visualizar, a partir das Figuras 2 (a), (b) e (c) que o algoritmo RCM gerou a menor largura de banda. Esse comportamento também foi observado nos outros casos. Os resultados obtidos até o momento mostram que essas técnicas são importantes para melhorar o desempenho de um solucionador de elementos finitos. Para trabalhos futuros, desejamos realizar um estudo mais detalhado em uma aplicação de elementos finitos e criar a partir do particionamento realizada pela METIS um padrão de comunicação por trocas de mensagens para uma implementação em memória distribuída.

Referências

Burkardt, J. (2003). Reverse cuthill mckee ordering. `https://people.sc.fsu.edu/~jburkardt/cpp_src/rcm/rcm.html` *(Oct. 15, 2020)*.

Dagum, L. and Menon, R. (1998). Openmp: an industry standard api for shared-memory programming. *Computational Science & Engineering, IEEE*, 5(1):46–55.

Forum, M. P. (1994). Mpi: A message-passing interface standard. Technical report, USA.

Hughes, T. J. (2012). *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation.

Karypis, G. and Kumar, V. (1998). A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392.

Rokos, G., Gorman, G., and Kelly, P. H. (2015). A fast and scalable graph coloring algorithm for multi-core and many-core architectures. In *European Conference on Parallel Processing*, pages 414–425. Springer.