FEM implementation using PETSc to simulate large deformations of solids on distributed memory architectures

Eduardo Fernández, Dorian Bogucki, Simon Février, Martin Lacroix, Luc Papeleux, Romain Boman, Jean-Philippe Ponthot

LTAS-MN2L, Aerospace and Mechanical Engineering Department, University of Liege, Belgium.

1 Introduction

The Finite Element Method (FEM) can be significantly accelerated through parallel computing. In CPU-oriented codes, parallelization can be achieved using either shared-memory or distributed-memory architectures. The former is generally easier to implement, as all processors access the same memory space, and a specialized API (such as OpenMP) handles the synchronization of read and write operations. In contrast, distributed-memory architectures require the programmer to partition the model's data across separate memory spaces, each managed by a different processor. Additionally, communication between processors must be handled explicitly using specialized libraries like MPI (Message Passing Interface). Although both approaches enable code parallelization, they offer different advantages. Shared-memory parallelization is typically easier to implement and provides good scalability for a relatively small number of processors. On the other hand, distributed-memory parallelization is more complex to implement but allows for the use of a much larger number of processors, enabling the simulation of larger and more computationally demanding problems.

Currently, there are several libraries that facilitate the implementation of FEM on distributed-memory architectures. For example, PETSc and Trilinos are among the most widely used by the numerical simulation community. Around these libraries, dozens of FEM codes have been built, such as FEniCS, DEAL.II, FreeFEM, libMesh, among others. Although these codes provide open and easy access to distributed-memory FEM simulations, implementing new models or constitutive laws can be laborious, as it often requires a significant amount of time to study and understand the source code.

At present, MN2L has a distributed-memory FEM implementation for linear solids (small deformations, see Fig. 1). The code is written in C++ and is based solely on the PETSc library, making it quite compact and easy to read. The goal of this master thesis is therefore to extend this code to simulate solids undergoing large deformations.

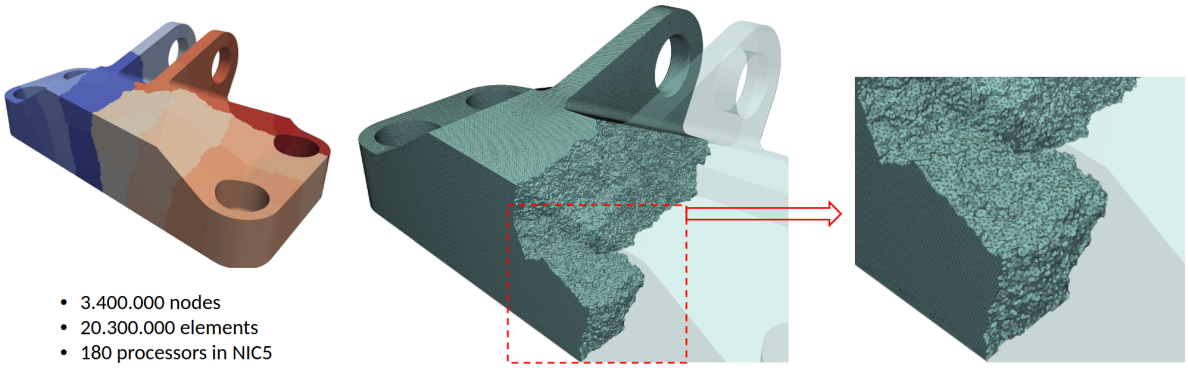

Figure 1: *Illustration of a bracket that is simulated using our code for small deformations.*

2 Objective

The goal of this work is to implement the FEM formulation for large deformations, in 2D and 3D, starting from an existing code that simulates linear solids in 2D and 3D. The main interest of this work is to compare the different solvers available in the PETSc library in the context of non-linear solid mechanics

(large deformations), although the student is welcome to explore other topics of interest, such as a performance comparison with the METAFOR software, the implementation of topology optimization for large deformations, or the large-scale simulation of a particular problem from the literature.

The work must be carried out in C++ using the PETSc library. Therefore, the main requirements to successfully carry out the work are to understand the FEM formulation for large deformations and to know C++. Knowledge of PETSc, C and MPI is not mandatory as it can be easily learned in practice, especially starting from an existing and compact code.