Project

Study and comparison of classical DDM – Application to linear elasticity –

The objective of this project is to analyze the behavior of different domain decomposition methods and to understand the role and the interest of the coarse problem and its resolution by an iterative method. In particular, we will be interested in the property of numerical scalability.

To simplify and facilitate understanding, we will do this analysis on an academical example of a bar with linear elastic behavior subjected to a tensile force F_d (**Figure 1**). Of course, this (too) simple example does not illustrate all the technical issues that may arise in more practical cases and some numerical developments or calculations may seem a little trivial in the present case. We will still try to make these calculations and stay in the mind. However, this example has the advantage of being able to easily and quickly implement the different approaches and to understand and illustrate certain fundamental points.

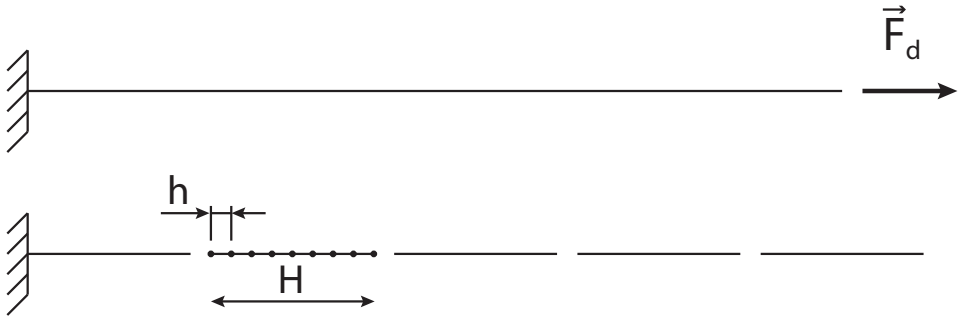

Figure 1 • Bar in tensile and decomposition into subdomains

The bar has a length L, a section S and a Young's modulus E.

All the calculations that can be carried out in parallel on different processors will be solved here sequentially on one machine. The parallelization of the code will not be considered here.

Instructions : Process Section 1, then at least one of the following three sections:

- Primal approach (Section 2.1) ;
- Dual approach (Section 2.2) ;
- Mixed approach (Section 2.3).

The work consists in providing:

- a **report** of about 15 pages (.pdf format);
- the **commented programs** with precision. For practical reasons, Matlab is preferred but Octave or Scilab can possibly be used.

Progress points will be made during two practical sessions in a CAD room at ENS Paris-Saclay, the dates of which will be communicated to you shortly.

All the documents (report and programs in an archive) will be sent a few days before the oral evaluation by email to pierre-alain.guidault@ens-paris-saclay.fr.

Oral evaluations, lasting approximately 20 minutes, will take place at ENS Paris-Saclay. The date and a schedule will be communicated to you later. All documents are authorized for this evaluation.

1 Preliminary data setting work

Question 1.1 • Recall the analytical solution of the problem of **Figure 1**.

Question 1.2 • Carry out a finite element bar code (1D) with linear displacement interpolation to solve the problem in Matlab (possibly Octave or Scilab). The **development constraints** are as follows:

- Possibility of having a **arbitrary number of elements of size** h;
- Uniform mesh (h constant). The functionality of having a non-uniform mesh is not necessary for the following;
- Calculation of the stiffness matrix by assembly of elementary stiffness matrixes (exact integration).
- Taking into account of the boundary conditions in displacement by direct elimination of the degrees of freedom concerned. Penalty or Lagrange multiplier can of course be used.

Question 1.3 • Extend the functionalities of your code in order to be able to solve the problem by domain decomposition methods:

- Definition of substructures of size H. A substructure is defined by a set of finite elements which one will be able to access, for example, by a list of numbers of elements numbered at the global level. The **number of substructures can be arbitrary and their size constant** (H left to the choice of the user, and constant). Each substructure will contain the same number of elements (but a variable number).

- Definition of the interfaces by the list of the degrees of freedom concerned for example.
- Calculation of the operators by substructure: stiffness matrix, primal Schur complement, rigid body modes...

To define numberings of degrees of freedom at the global level and at the local/substructure level is not essential. Note that in this case it would be necessary to define as in the course the assembly operators $A^{(s)}$ and $\underline{A}^{(s)}$ for each subdomain. For this very one-dimensional simple example, this formalism is not required even if in this case the implementation is not for all that more complicated.

2 Scalability study

We denote by H a characteristic length of the subdomains, and by h a characteristic length of the finite elements. Working at constant h is like keeping a constant size for reference (unsubstructured) problem. Working at constant H is like keeping the same number of subdomains. Finally, a last interesting parameter is $\frac{h}{H}$: working at $\frac{h}{H}$ constant keeps local problems by subdomain of constant size, the size of the reference problem increasing with the number of subdomains.

A domain decomposition method is said:

- **numerically scalable** if, working at $\frac{h}{H}$ constant, the number of iterations to converge does not depend very much on the number of subdomains (therefore of H);
- **optimal** if there is little dependence on $\frac{h}{H}$;
- **parallel scalable** if the computational time is little dependent on H.

Typically for monoscale approaches, one clearly observes a decrease in the rate of convergence with the number of substructures. It appears that to obtain the property of scalability, a domain decomposition method must imperatively be multiscale.

In the following, we will take h and H uniform over all the subdomains. The finite elements and the subdomains will therefore have the same size. Indeed, many results of the literature are demonstrated in this case.

2.1 Primal approach

Question 2.1 • By using the primal Schur approach, solve the assembled primal interface problem of **Figure 1** by a direct method. Check that you get the right solution.

Question 2.2 • How do the conditioning of the assembled primal Schur matrix, $\mathbf{S}_p$, vary as a function of h/H and that of the assembled global stiffness matrix, $\mathbf{K}$, of the unsubstructured problem according to h? Comment the results.

The interface problem is in practice not assembled and solved by a *direct linear solver*. An iterative method is used which only requires local calculations on each subdomain as much as possible. Only residual vectors (and not operators) are exchanged and assembled, which represents a small amount of data.

Question 2.3 • Solve the interface problem by a **distributed (parallelized) conjugate gradient method**. The convergence criterion will be that of the residue (normalized by the norm of the load F or the residue at first iteration) of the interface problem. By *distributed* is meant the fact that only vectors are exchanged/distributed to solve the interface problem and that the matrix-vector products are produced locally at the subdomain level.

Question 2.4 • Is the strategy thus programmed numerically scalable? Please justify your answer.

Question 2.5 • Solve the interface problem by a **preconditioned conjugate gradient** using a Neumann preconditioner (BDD method) (augmented Krylov solver). Is the strategy thus programmed numerically scalable? Please justify your answer.

Question 2.6 • By assembling (exceptionally) $\tilde{\mathbf{S}}_p^{-1}\mathbf{S}_p$, show that the conditioning of the preconditioned system $\kappa\left(\tilde{\mathbf{S}}_p^{-1}\mathbf{S}_p\right)$ is in $O(1)$.

2.2 Dual approach

Question 2.7 • By using the dual Schur approach, solve the assembled dual interface problem of **Figure 1** by a direct method. Check that you get the right solution. It should be noted that in the general case, $\mathbf{S}_d$ may be undefined due to the redundancy of multipliers at multiple nodes (shared by several subdomains) which is not the case here for this simple one-dimensional example.

The interface problem is in practice not assembled and solved by a *direct approach*. An iterative method is used which only requires local calculations on each subdomain as much as possible. Only residual vectors (and not operators) are exchanged and assembled, which represents a small amount of data.

Question 2.8 • The dual interface problem is this time solved by a **method of distributed projected conjugate gradient** to take into account the condition of edge forces balanced on each substructure (FETI method, constrained Krylov solver). Use a Dirichlet preconditioner. By *distributed* is meant the fact that only vectors are exchanged/distributed to solve the interface problem and that the matrix-vector products are produced locally at the subdomain level.

Question 2.9 • How does one obtain the rigid body modes of substructures at convergence? Plot the solution in displacement.

2.3 Mixed approach

Question 2.10 • By using the mixed monoscale LaTIn approach, solve the problem of **Figure 1**. Check that you get the right solution.

Question 2.11 • Determine numerically the optimal values of the search directions k^+ and k^- which lead to the fastest convergence. Interpret the obtained optimal values with respect to the stiffness of the bar.

Question 2.12 • Is the strategy thus programmed numerically scalable? Please justify your answer.

Question 2.13 • Introduce the coarse/macroscopic problem of the multiscale LaTIn mixed approach and solve the interface problem by a direct solver. Note that in this case, the macroscopic interface problem already contains all the information. The microscopic quantities are zero and the approach on this case where each interface is

reduced to a single point (of zero measure) is degenerated. All the same, as much as possible, we will try to do all the necessary calculations (definition of the projectors, homogenization procedure...).

Question 2.14 • Is the strategy thus programmed numerically scalable? Please justify your answer.