Balancing Domain Decomposition with Nonlinear Relocalization: Parallel Implementation for Laminates

Felipe Bordeu, Pierre-Alain Boucard, Pierre Gosselet

Balancing Domain Decomposition with Nonlinear Relocalization: Parallel Implementation for Laminates

Felipe Bordeu, Pierre-Alain Boucard, Pierre Gosselet

LMT-Cachan
ENS Cachan/CNRS/UPMC/PRES UniverSud Paris)
61 av. du Président Wilson
F-94230 Cachan, France
e-mail: {apf,gosselet,rey}@lmt.ens-cachan.fr,
web page: http://www.lmt.ens-cachan.fr

December 1, 2009

Abstract

Over the past thirty years, composite materials have been used increasingly in industry, especially in the aeronautical and spatial industries. Therefore, there is a great interest in the prediction of their degradation. The objective of this work is to develop a program capable of simulating structures on an industrial level using the latest models developed. In the first section, we present a review of the damage mesomodel used for the simulations. The second section presents the partition technique, and the parallel resolution technique utilized for the simulation. A new Newton loop is added to the classic algorithm to improve the performance in the case of localized nonlinearities. The third section present the implementation on a C++ home made finite element code. And the technique utilized to parallelized the problem. Finally, the forth section gives illustrations showing the level of performance which can be expected from such an approach.

Keywords: Composite Material, Partitioning, Parallelization, Domain decomposition, Newton-Krylov-Schur methods, Damage.

1 Introduction

Today, the simulation of the degradation mechanisms in composite structures until final failure is still an industrial challenge, specially in the aeronautical and the spatial industry. This is mainly due to: (i) the complexity of the degradation mechanisms involved in the failure of composite structures and (ii) the size and the geometric complexity of the parts. Also structural phenomena like buckling can appear in these cases when stability is lost.

Degradation in composites is a complex phenomenon involving both damage and inelastic residual deformation. We observe fiber-matrix interface debonding, plasticity, progressive transverse cracking, brittle fracture of fibers (in plies) and delamination (between plies). These mechanisms are highly nonlinear, fiber-matrix debonding, transverse cracking and delamination present a strong unilateral feature which depends on whether the cracks are open or close. In order to take all these phenomena into account, a model proposed in [1] was chosen. In our work an enhanced version of this model was used [2]. This model is defined at the mesoscale characterized by the thickness of the plies [3]. Then, the laminated structure is described as a stacking sequence of homogeneous layers and interlaminar interfaces. The model of each mesoconstituent (i.e. the elementary layer and the interface) are introduced using the internal variable framework. Damage is quantified by mesodamage indicators, which can be connected directly to the loss of stiffness. An important point is that the state of damage is assumed to be uniform throughout the thickness of the elementary layer, but not throughout the whole thickness of the laminate.

On can observe that the scale of the model regarding to the scale of the composite structure are very different. Indeed, this difference results in a high computational cost in the case of a simulation of an industrial size structure. Today the increasing power of computers and the efficiency of parallel resolution algorithms offer the opportunity to perform simulation of very complex geometries.

2 Damage mesomodel

The first description of the mesomodel for laminates can be found in [3]. Two main assumptions lead to such a mesomodel. The first one is that the behavior of any laminated structure can be reconstructed starting from two elementary mesoconstituents: the elementary layer [4] and the interface [5] (fig. 1). The second assumption is that the damage state is uniform throughout the whole thickness of the elementary layer (but not throughout the thickness of the laminated composite).

In the earlier version of the model, the behavior law of each mesoconstituent was considered to be intrinsic, but recent works showed that at the mesoscale the behavior of an interface is coupled with the internal variables of the adjacent layers.

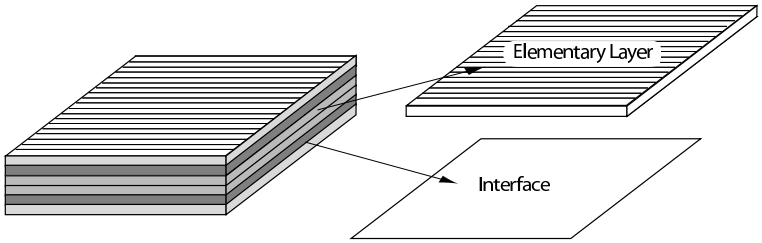

Figure 1: The model of a laminate

2.1 The elementary layer

The elementary layer is defined in classical energy form: $2e = \sigma^t S_{layer} \sigma$ [2], where σ is the stress and S_{layer} the compliance operator. This equation is written in a local reference frame defined by the fiber's direction (N_1) and the transverse direction (N_2). Operator S_{layer} takes into account the damage state of the elementary layer [2]. The damage indicators, which are constant throughout the thickness, are associated with the following degradation mechanisms:

- fiber breakage,
- diffuse intralaminar degradation,
- transverse microcracking

For the sake of simplicity, we present the general form of the evolution laws (for more details, one can refer to [2]). The damage evolution laws use classical thermodynamic forces. The calculation of these forces involves a "mean value" operator whose role is to calculate the mean value of the thermodynamic forces throughout the thickness of the elementary layer. This operator is very important because it ensures that the damage indicator remains constant throughout the thickness of the elementary layer. Then, the damage indicators for fiber breakage and diffuse intralaminar degradation are updated using the evolution laws. These evolution laws depend on the material and can be modeled as progressive or brittle. For a complete description of these functions see [6].

The case of transverse microcracking is a little different. Transverse microcracking is taken into account by using an equivalent microcracking rate ρ for the elementary layer [2]. Then, with this microcracking rate and the evolution law, one can update the damage indicators for this phenomenon.

Other phenomena, like fiber hiperelasticity, matrix plasticity or fiber rotation can be easily taken into account.

2.2 The interface

The interface, which is introduced in order to model the debonding of two adjacent plies, can be interpreted as the thin layer of matrix between two plies and is a two-

dimensional entity which ensures the transfer of stresses and displacements from one elementary layer to another.

The displacement jump between the upper and lower surfaces of the interface is expressed in the local reference frame defined by the orthotropic directions N_1, N_2 and N_3; N_3 is normal to the interface, and N_1 and N_2 are the bisectors of the angle formed by the fibers of the upper and lower plies (fig. 2).

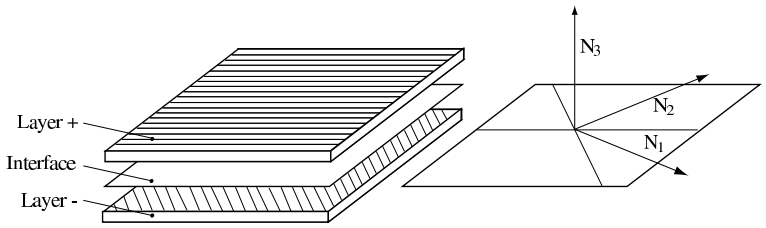

Figure 2: Orthotropic directions of the interface

As in the case of the elementary layer, the elastic strain energy of the interface ($2e = \sigma^t S_{interface}\sigma$) is defined by a compliance operator $S_{interface}$. This operator take into account the damage indicators as well as the unilateral behavior of the interface in mode I.

In the current version of the mesomodel, the interface's damage evolution laws are strongly coupled with the state variables of the adjacent layers [2]. This coupling is achieved through the mean value of the microcracking rate ρ of the upper and lower adjacent layers (1).

$$\rho = \frac{\rho^+ + \rho^-}{2} \tag{1}$$

The mean values operator used for the layer and the coupled evolution law of the interface make the model highly nonlocal. Special care must be taken to ensure the correct implementation and integration of the evolution laws.

3 Parallel Nonlinear Solving Method

The previous section allow us to describe all the main mechanisms involved in the ruin of a composite structures. All this makes the problem highly nonlinear and non local, and also because the model is written at the mesoscale, a large number of d.o.f. is expected.

To treat these kind of problems, parallel solvers seems to be the best choice. The Balancing Domain Decomposition method (BDD) [7] was chosen to parallelized the problem. As we will further explain, this method rely on Schur non-overlapping decomposition of the structure and Krylov iterative solvers. This method is used as a linear solver inside a Newton-Raphson scheme to solve the nonlinear problem. Like every parallel solver a partition of the domain (the mesh) is required.

3.1 Preprocessing and partitioning

The discretization and partitioning of a composite structure prior to a parallel simulation with the mesomodel becomes a very delicate task. Because the behavior law is nonlocal and the integration of the evolution law is done at Gauss points, the mesh must be regular in the thickness of the composite (fig. 3).

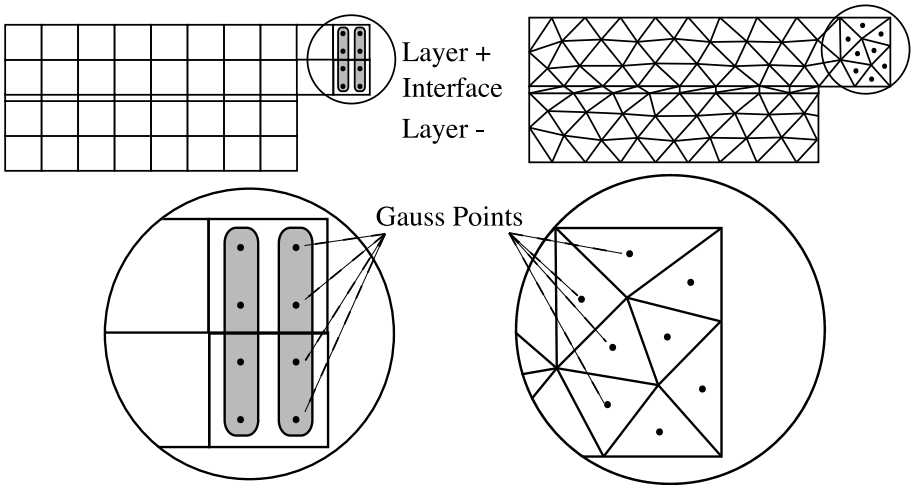

Figure 3: Regular (left) and unregular (right) mesh (grey area: Integration point aligned in the thickness)

To achieve this, the mesh is created only with hexahedral and wedges elements, this ensure that the integration points are always aligned in the thickness of the mesh. This point add some difficulty in the meshing process but is not a real restriction because virtually all composite structures are made from a stack of single laminas.

It is known that the performance of parallel solvers are very dependent of the quality of the partitioning [8]. On one hand, one would seek partitions with the same number of elements in order to achieve a good balance of the load among the processors during the parallel resolution. On the other hand, one also wants to minimize the interaction zones between subdomains in order to minimize communications among processors. Finally, one must take into account the fact that the behavior law is nonlocal and requires information of the neighboring elements in order to be integrated correctly.

In the case of the use of a classical algorithm, two elements that are aligned in the thickness of the composite could end in different subdomains. In this case, one will be forced to transfer information across subdomains during the behavior law integration. One way to prevent this is to force the partition algorithm not to cut between element that are aligned in the thickness of the composite structure (fig. 4).

In order to restrict the way the partitioning is performed, we calculate the dual graph of the mesh (fig. 5.a and 5.b). Then this graph is reduced in such a way that elements that are aligned in the thickness of the composite are condensed into one single vertex. The reduce graph is then weighted to take into account the number of elements

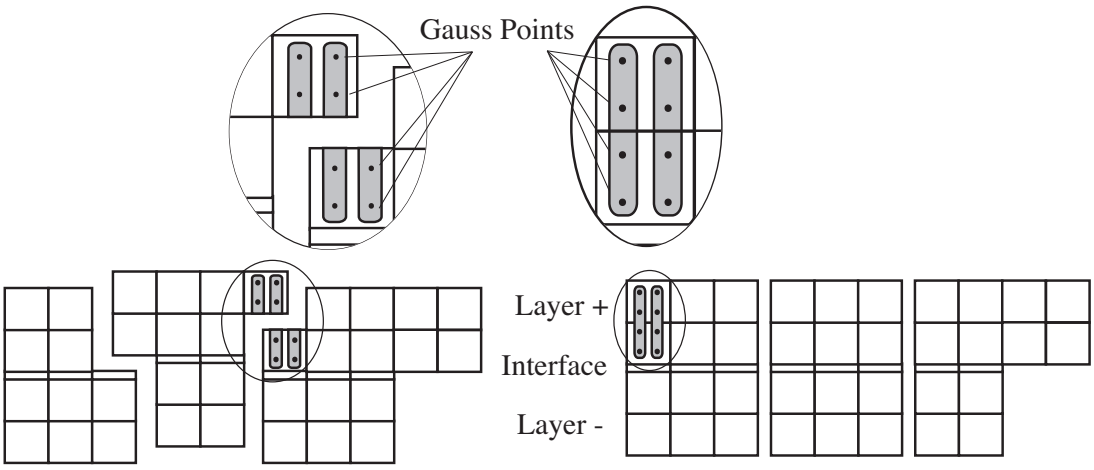

Figure 4: Poor partitioning (left), Good partitioning (right)

aligned in the thickness (fig. 5.c). Now that the condensed graph was constructed, we can use any partition algorithm to perform the partition of the graph (fig. 5.d). In our case we used Metis [9], an open source software capable of performing fast and high quality partitions for irregular graphs. After the partition, the graph is expanded (fig. 5.e) to finally calculate the partition of the mesh (fig. 5.f).

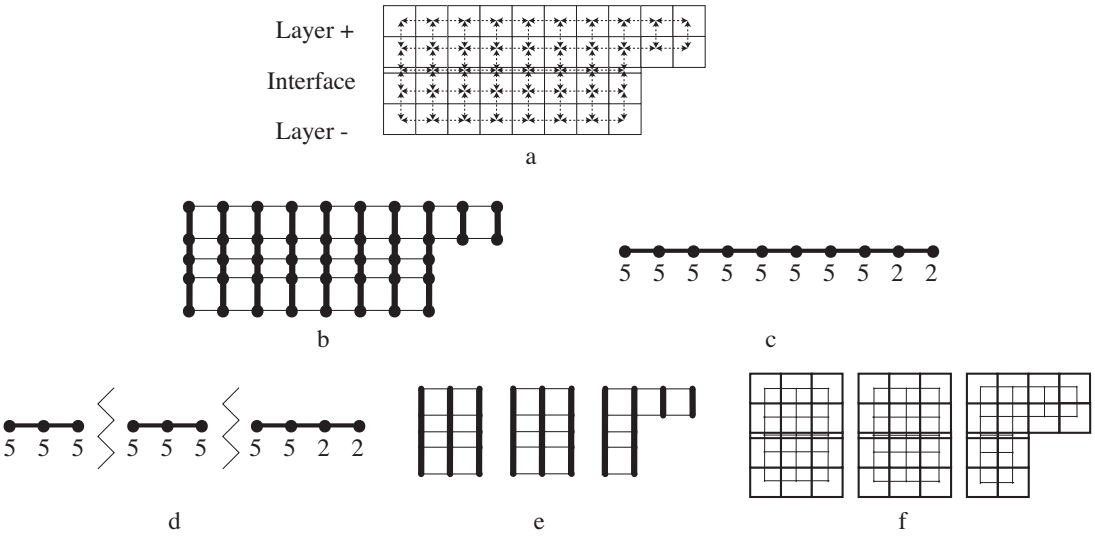

Figure 5: (a) Original mesh and its graph, (c) dual graph of the mesh (the bold lines represent element aligned in the thickness), (c) weighted reduce graph, (d) partitioned weighted graph, (e) expanded graph, (f) partitioned final mesh

At the same time that the partition is calculated, a file with the connectivity of the subdomains is created. This information will be used to optimize the reading of the meshes once the parallel simulation begins.

This technique enable us to perform fast, hight quality and automatic partitions of the mesh.

3.2 The resolution algorithm

As we said before, a BDD method was used to parallelize the resolution. The principle of the BDD is to condense the internal unknowns of each subdomain on the boundary, and then solve the condensed problem in parallel using an iterative method. First, for each substructure s we construct the Schur complement of the tangent operator and the condensed residue through the elimination of inner degrees of freedom (2) and (3). In equations (2) and (3) the i subscript indicates the internal unknowns and b the unknowns on the boundaries. These Schur complements and condensed residues computed for each subdomain can be assembled to construct a global linear problem over all boundary unknowns (4). Solving the global linear problem enables to compute the displacements on the boundaries. Finally a localization step is needed to determine the inner displacement in each subdomain (5).

$$S_T^s = K_{Tbb}^s - K_{Tbi}^s K_{Tii}^{s^{-1}} K_{Tib}^s \quad (2)$$

$$b^s = r_b^s - K_{Tbi}^s K_{Tii}^{s^{-1}} r_i^s \quad (3)$$

$$S_T \Delta u_b = b \quad (4)$$

$$\Delta u_i^s = K_{Tii}^{s^{-1}} (r_i^s - K_{Tib}^s \Delta u_b^s) \quad (5)$$

The equations (2), (3) and (5) are defined within subdomains, and only the equation (4) is defined over all the boundary unknowns. To avoid the heavy transfer (in the case of a direct resolution) the equation (4) is solved using a Krylov solver. The conjugate gradient, used to solve the global problem, only requires local matrix-vector products and vectors assemblies, avoiding the assemblage of the global system. More detail about this method can be found in [7].

Then, this linear solver is used inside a Newton loop which treats the non linear problem. Algorithm 1 summarized the structure of the Newton-Krylov-Schur method.

Algorithm 1: Classic Newton-Krylov-Shur algorithm

repeat
- -Assemble local tangent operators K_T^s
- -Factorize K_{Tii}^s (eq. (2))
- -Assemble local residue r^s (eq. (3))
- -Solve condensed global problem (eq. (4)) (Krylov solver)
- -Compute inner displacement in each subdomain (eq. (5))

until *error*$< \varepsilon_{global}$

In general the evolution of the nonlinearity in a composite structure is not uniform. Indeed, in most of the cases the nonlinearity is localized in a reduce zone (holes, crack tips, junctions), having sometime little or no influence on the rest of the structure. When the algorithm 1 is used, the convergence of the Newton is directly related to the convergence of the localized nonlinearity. To correctly treat this type of nonlinearity at the correct scale, an enhanced version of the NKS solver was proposed [10]. In this version, a new Newton loop is introduced instead of the equation (5). This local

Newton forces the subdomains to obey the nonlinear behavior and the equilibrium. Algorithm 2 summarizes the structure of the Newton-Krylov-Schur method with a nonlinear localization.

Algorithm 2: Newton-Krylov-Shur with nonlinear localization

repeat
- -Assemble local tangent operators K_T^s
- -Factorize K_{Tii}^s (eq. (2))
- -Assemble local residue r^s (eq. (3))
- -Solve condensed global problem (eq. (4))
- **repeat**
  - -Compute inner displacement $\Delta u_i^s = K_{Tii}^{s^{-1}}(r_i^s - K_{Tib}^s u_b^s)$
  - -Update u_i, K_{Tii}^s, K_{Tib}^s and r_i^s
- **until** *error*$< \varepsilon_{local}$

until *error*$< \varepsilon_{global}$

4 Implementation

The strategy presented in the previous section was implemented in a home made 3D finite element software. The software was written in C++ using a preprocessor [11] written in Python that was in charge of generating the C++ code for elements and behavior laws. This makes the code very scalable for new functions, behavior laws or element types. Because the code generated is already optimized, the performance of the final code is guaranteed.

Most of todays supercomputer are SMP (Symmetric Multi-Processors) Clusters, this means that the cluster is composed of multi-core (or multi-processor) nodes. Figure 6 show a simple schema of a SMP Cluster. One can see that each node contains more than one core (or processor). All cores belonging to the same node share some resources, like RAM, local hard drive and network connections. Two level of synchronization and communication can be distinguished, the node level (between threads) and the cluster level (between nodes). To take advantage of this configuration, the idea is to run only one process per node, and make use of multithreading to take advantage of all the available cores.

Communication between threads is almost instantaneous (because they share the same RAM) only the synchronization must be taken care of. The synchronization at this level was ensured by the use of an open source library named ZThread. The second level of communication and synchronization was implemented using MPI.

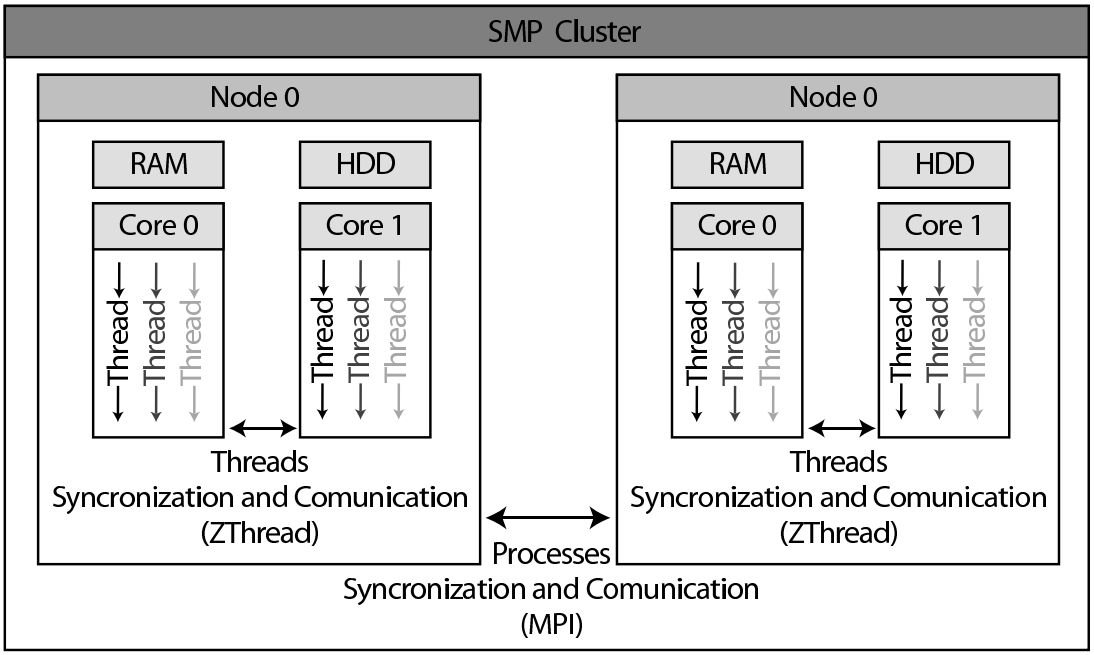

Figure 6: Schema of a SMP Cluster Super-calculator

5 Simulation Results

To verify the implementation of the domain decomposition method in parallel an elastic problem was treated. Figure 7.a shows the mesh and one of the partition (50 subdomains) used for the test. The different partitions were done with a classical algorithm, because the behavior law is elastic, there is no need to used the algorithm presented before. The boundary condition of the problem were blocked displacement in all directions in one side (face o in fig. 7.b), and an unitary displacement in the three direction on the other side (face p in fig. 7.b). The material is linear elastic, the piece has 150.000 degrees of freedom. Figure 7.b show the deformation and the strain on the piece.

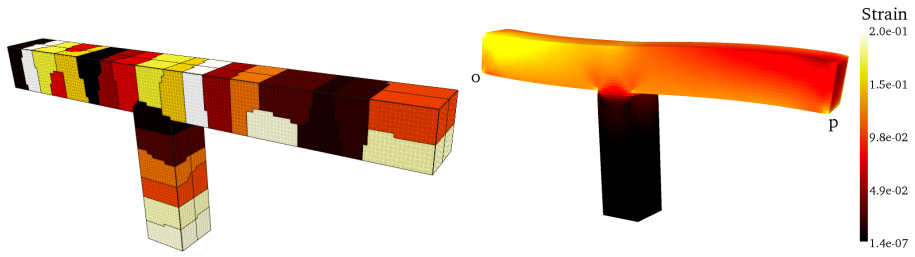

Figure 7: The mesh (each color represent a subdomain) (left), deformation and strain (right)

First, we solved the problem for a different number of subdomains. Figure 8 show that the number of iteration of the iterative method rest almost constant, this is impor-

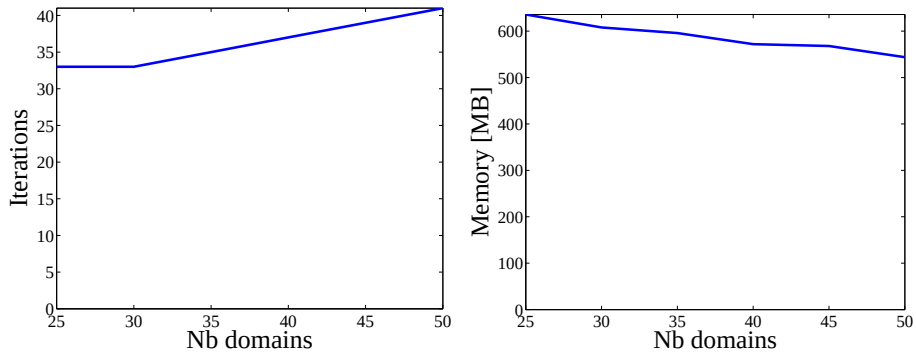

Figure 8: Iterations of the Krylov solver vs subdomain number (left), memory usage vs subdomain number (right)

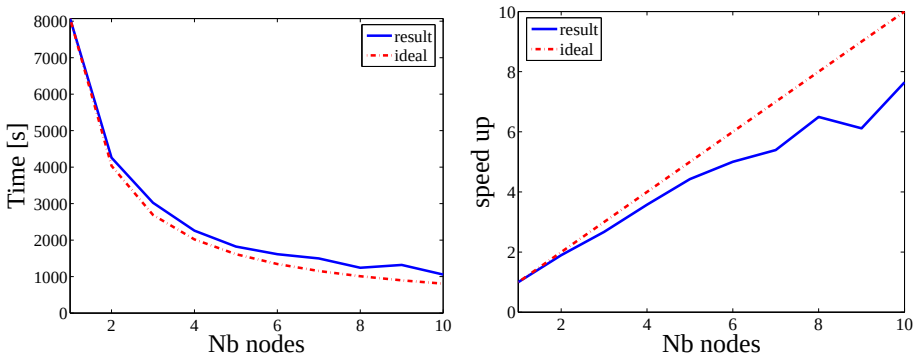

Figure 9: Total time vs number of node (left), speed up vs number of nodes (right)

tant to ensure the extensibility of the method. The number of iteration may vary from decomposition to decomposition, due mainly to the partition algorithm [8]. Figure 8 show the memory used for the resolution.

Then, using a fix number of subdomains (50), the simulation was executed for a different number of nodes. Each cluster node has 2 cores at 2.4 GHz and 8 GB of RAM. In Figure 9 one can see the time spend and the speed up of the resolution in function of the number of node used.

The second case is holed carbon fiber reinforced laminate plate, figure 10 show the geometry and the partition used for the simulation, fiber orientation of the layers are $[45_4\, 90_4\, \text{-}45_4\, 0_4]_s$. The partitioning of the mesh was done with the proposed technique.

In this simulation all the degradation mechanisms of the damage model were active. The loading (0.075% of deformation) was apply in 6 steps. Table 1 show, the number of iteration executed for a classic Newton algorithm and for the algorithm with nonlinear localization.

Each global Newton computations of the classic algorithm implies one global resolution with the Krylov solver plus one local computation per subdomain. In the case of the algorithm with nonlinear localization the number of local resolution may varies from subdomain to subdomain, according to the local nonlinearities. Even though the

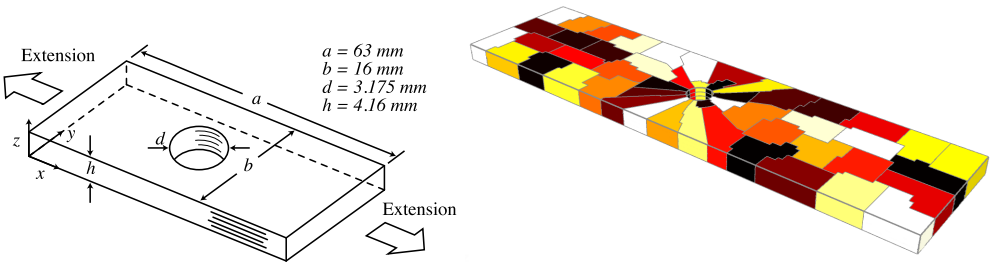

Figure 10: Holed carbon fiber reinforced laminate plate geometry and partition

Loading step	Number of Iteration	nonlinear localization	linear localization
1	Newton Global	1	2
	Newton Local	89	100
2	Newton Global	1	2
	Newton Local	99	100
3	Newton Global	1	2
	Newton Local	89	100
4	Newton Global	1	2
	Newton Local	89	100
5	Newton Global	1	2
	Newton Local	91	100
6	Newton Global	3	3
	Newton Local	197	150

Table 1: Number of Newton iteration for each step: with nonlinear localization (left), with linear localization (right)

number of local iteration of the algorithm with nonlinear localization may be superior to the classic algorithm, the most spending time task is the execution of a global loop.

6 Conclusion

In the first section, we reviewed briefly one of the possible model for the simulation of composites and pinpointed the difficulties of implementing such a model.

In the second section we proposed a technique for partitioning a mesh correctly so no additional effort would be necessary in a parallel calculation. Also we proposed a calculation strategy for the simulation of large composite structures in parallel. The strategy enable us to reduce the iteration number executed by global newton.

Third section showed the techniques used for implementation of the strategy. These techniques enable us to run the software in parallel in SMP cluster.

Finally, we showed some preliminary results of the partitioning tool and calculation strategy.

References

[1] P. Ladevèze and G. Lubineau. A damage computational method for composite structures. *Computers and Structures*, 44:79–87, 1992.

[2] G. Lubineau and P. Ladevèze. Construction fo a micromechanics-based intralaminar mesomodel, and illustrations in abaqus/standard. *Computational Materials Science*, 43(1):137–145, 2008.

[3] P. Ladevèze. Sur la mécanique de l'endommagement des composites. In E. Basthias and D. Menkès, editors, *Comptes-Rendus des JNC5*, pages 667–683, Paris, 1986. Pluralis Publication.

[4] P. Ladevèze and EL. Dantec. Damage modelling of the elementary ply for laminated composites. *Composites Science and Technology*, 43(3):257–267, 1992.

[5] O. Allix, P. Ladevèze, and A. Corigliano. Damage analysis of interlaminar fracture specimens. *Composites Structures*, 31:61–74, 1995.

[6] P. Ladevèze, O. Allix, JF Deü, and D. Lévêque. A mesomodel for localisaton and damage computation in laminates. *Computer Methods in Applied Mechanics and Engineering*, 183:105–122, 2000.

[7] J. Mandel. Balancing domain decomposition. *Communications in Numerical Methods in Engineering*, 9:233–241, 1993.

[8] P. Gosselet and C. Rey. Non-overlapping domain decomposition methods in structural mechanics. *Archives of Computational Methods in Engineering*, 13:515–572, 2006.

[9] G. Kasypis and N. Takeda. A fast and high quality multilevel scheme for partitioning irregular graphs. *Journal on Scientific Computing*, 20:359–392, 1998.

[10] P. Cresta, O. Allix, C. Rey, and S. Guinard. Nonlinear localization strategies for domain decomposition methods: Application to post-buckling analyses. *Computer Methods in Applied Mechanics and Engineering*, 196:1436–1446, 2007.

[11] H. Leclerc. Platform metil: Optimizations and facilities linked to the generation of code (in french). *8nd National Colloquium in Structurs Calculation*, 2007.