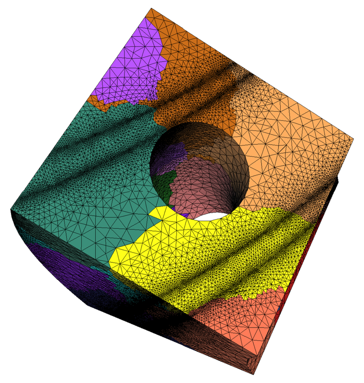
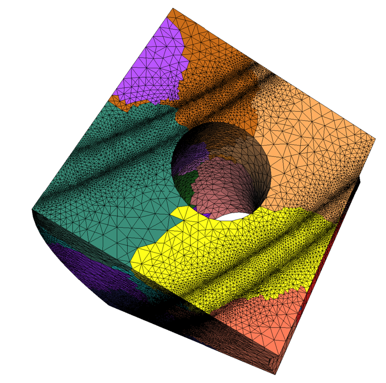


Parallel mesh adaptation

C. Dobrzynski and J.-F. Remacle

GCE - Université catholique de Louvain



Introduction

Extending mesh adaptation algorithms to parallel is a recent trend in the field of mesh generation. Most of today's finite element solvers are working in parallel. They are able to scale reasonably when they are run on distributed memory clusters. Mesh adaptation should therefore also be used in parallel.

We propose a parallel mesh adaptation procedure that make use of an existing serial mesh adaptation algorithm. In the parallel extension of the algorithm, edges that are on inter-processor boundaries are frozen *i.e.* are left unchanged during the iteration. At the end of the iteration, the elements that are situated at inter-processor boundaries are migrated so that they can be modified at the next iteration.

The advantage of the approach is its simplicity. Yet, despite of the very naive nature of the principle, we are able to produce very large meshes that are well adapted *i.e.* that respect well a given metric field. Moreover, a reasonable scaling was obtained up to 64 processors.

Mesh adaptation

Metric definition: the mesh adaptation procedure is based on metric tensor. This metric is a symetric definite positive matrix:

$$\mathcal{M} = \mathcal{R}\Lambda\mathcal{R}^{-1}$$

with $\Lambda = diag\,(\lambda_1, \lambda_2, \lambda_3)$, $\mathcal{R}$ the eigenvector matrix and λ_i the eigenvalues of $\mathcal{M}$.
The eigenvectors of $\mathcal{M}$ are the directions wished for the mesh edges and its eigenvalues theirs sizes.

Unit mesh: we search to generate an *unit mesh*: a mesh such that all its edges $\vec{e}$ has a length equal (or close) to one in $\mathcal{M}$:

$$l_{\mathcal{M}}(\vec{e}) = \langle \vec{e}, \vec{e} \rangle_{\mathcal{M}}^{\frac{1}{2}} = \sqrt{{}^t\vec{e}\mathcal{M}\vec{e}} = 1.$$

Mesh adaptation procedure: Given the mesh metric field defined over the domain, local mesh modification is applied to yield the desired anisotropic mesh. Mesh modification operators include entity (i) split, (ii) collapse, (iii) swap and (iv) reposition. To make any given mesh satisfying the given mesh size field by mesh modifications, we take philosophy as follows:

- identify those mesh entities not satisfying the mesh size field;
- perform appropriate mesh modifications so that local mesh will better satisfy the mesh size field;
- repeat above steps until the mesh size field is satisfied to an acceptable degree.

Since it is not possible to ensure that all mesh edges exactly match the requested lengths, the goal of mesh modifications is to make the transformed length of all mesh edges fall into an interval close to one. Particularly, we choose interval $[0.5, 1.4]$ in the examples which is large enough to avoid oscillations [1, 2].

References

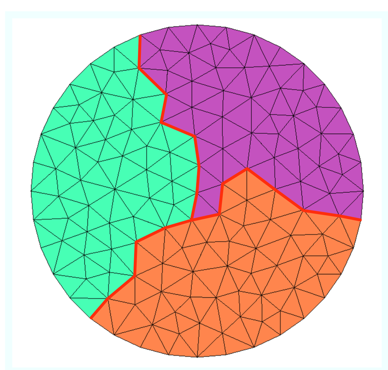
[1] X. Li. *Mesh Modification Procedures for General 3-D Non-manifold Domains.* PhD thesis, Rensselear Polytechnic Institute, August, 2003.

[2] X. Li, M.S. Shephard, and M.W. Beall. 3-d anisotropic mesh adaptation by mesh modifications. *Computer Methods in Applied Mechanics and Engineering*, 2003. in preparation.

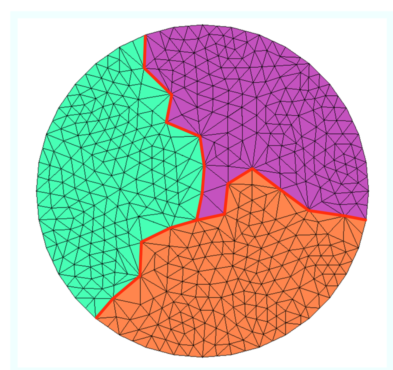
Parallel mesh algorithm

Principle: on each processor, we put contraints on a part of the boundary and we use an existing remeshing software. During the remeshing phase, the edge that is shared by several processors is not touched. So the mesh is adapted with constraints independently on each processor. Then, inter processors boundaries are moved: if a vertex is shared by several processors, all its adjacent elements are migrated into a common processor. And the remeshing process is again applied.

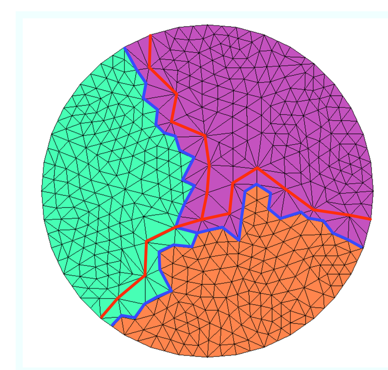
Algorithm: Each figure represents a stage of the algorithm.



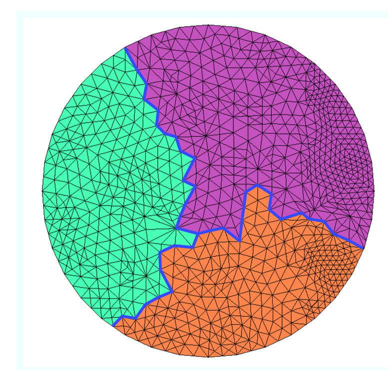
Initial mesh : each color represents a partition. The interface edges are drawn in red.



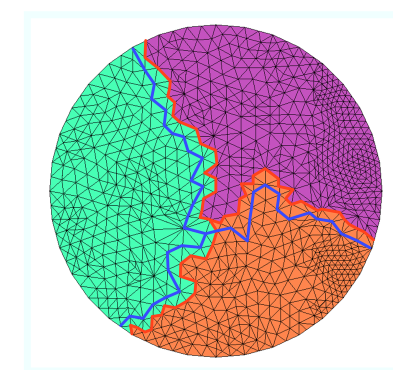
Each partition has been adapted without touching interface edges (red color).



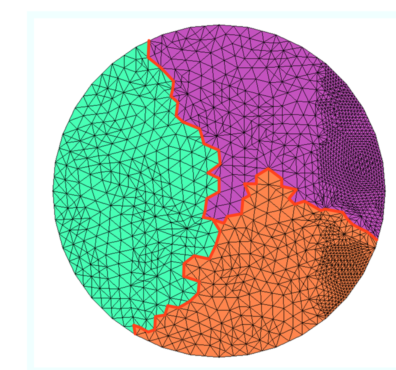
The interface edges moved: if an edge was shared by several processor, the two adjacent triangles are migrated into a common processor. The new interface edges are drawn in blue.



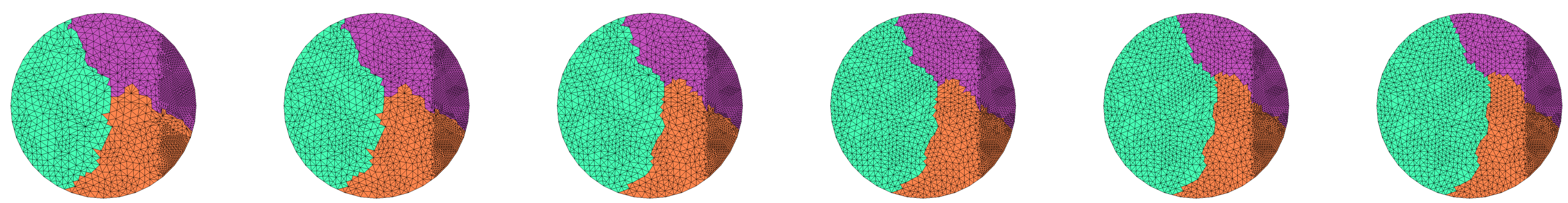
Each new partition has been adapted.



The interface edges moved. The new interface edges are drawn in red.



The new partitions have been adapted.



Examples

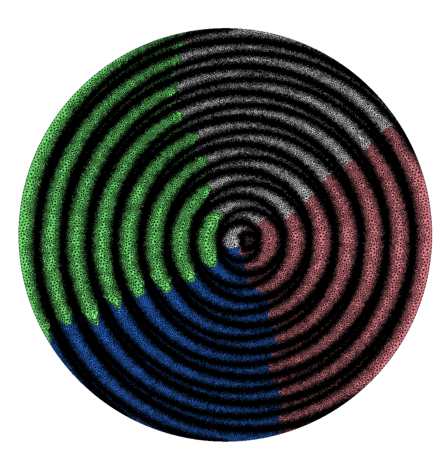
2d isotropic case: two Archimede's spirals.
The following size is defined:

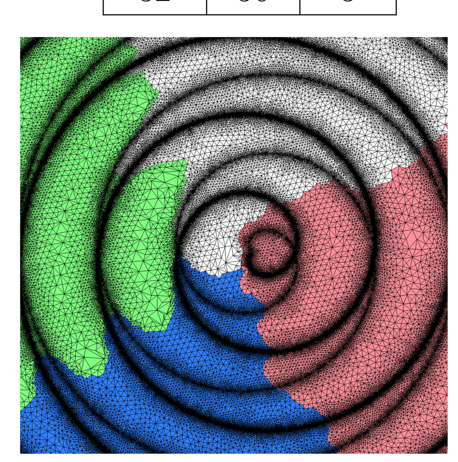
$$h = min(h_{max}|\rho - a \times \theta_1| + h_{min1}, h_{max}|\rho - a \times \theta_2| + h_{min2})$$

with $a = 1$, $h_{max} = 0.8$, $h_{min1} = 0.02$, $h_{min2} = 0.05$,$\rho = \sqrt{x^2 + y^2}$, $\theta_1 = \theta + \rho/\pi$ and $\theta_2 = \theta - \rho/\pi$ where $\theta = atan(y/x) + \pi$

	nb vertex	nb triangles
init. mesh	12,705	25,016
adapt. mesh	745,000	1,490,000

nproc	CPU	migr.
1	199	-
2	146	29
4	85	18
8	53	9
16	43	5
32	36	3





2d anisotropic case: planar shocks.
We defined the following analytical size field:

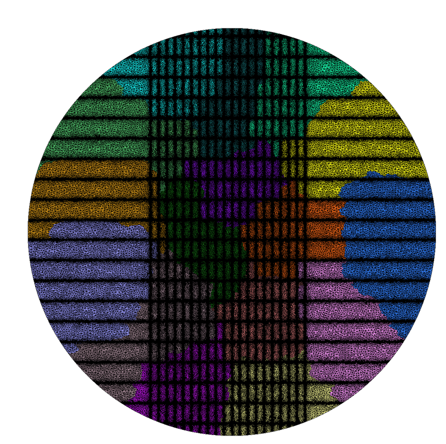
$$hx_i = 0.6|1 - e^{-|x - xc_i|}| + 0.003$$
$$hy_i = 0.6|1 - e^{-|y - yc_i|}| + 0.003$$

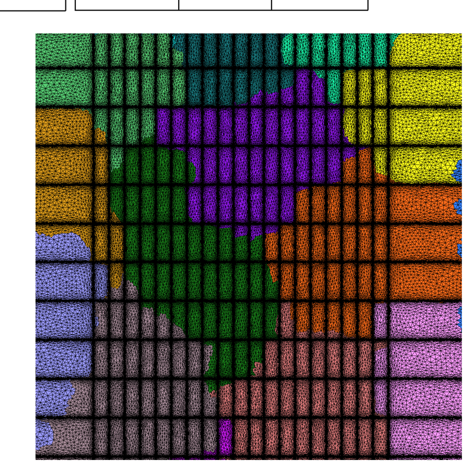
where $xc_i = -20 + 2 \times i$ and $yc_i = -47 + 5 \times i$.
The analytical metric is given by $\mathcal{R} = Id$ and

$$\Lambda = diag\left(\left(\min_{i=0..20}(hx_i)\right)^{-2} \left(\min_{i=0..20}(hy_i)\right)^{-2} h_{max}^{-2} \right)$$

	nb vertex	nb triangles
init. mesh	6855	13,392
adapt. mesh	362,000	726,000

nproc	CPU	migr.
1	698	-
8	45	22





3d isotropic case: some portions of Archimede's spiral.
We use the following parameters: $a = 0.1$, $h_{max} = 0.2$ and $h_{min1} = h_{min2} = 0.008$.

	nb vertex	nb triangles
init. mesh	2992	16,653
adapt. mesh	430,000	2,500,000

nproc	CPU	migr.
1	1297	-
2	386	147
4	244	143
8	202	114
16	179	106
32	144	72