PaMPA

Parallel Mesh Partitioning and Adaptation

Cédric Lachat[1], François Pellegrini[2,3,1] and Cécile Dobrzynski[4,1]

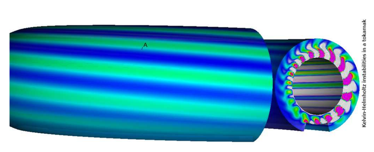
Tel: 05 35 00 26 30
Mail: cedric.lachat@inria.fr
TADaaM Team
Inria Bordeaux – Sud-Ouest
200 Avenue de la Vieille Tour
33405 Talence Cedex

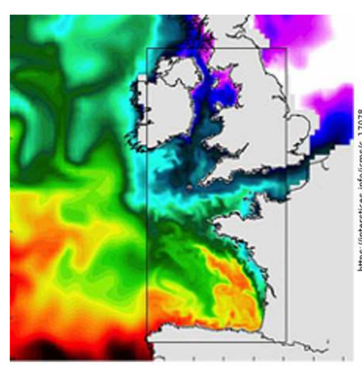
Introduction

Today's large scale simulations can only be run in parallel, because many meshes are now too big to fit in the memory of a single computer. Since shared-memory architectures are subject to memory bottlenecks, scalability can only be achieved by using distributed-memory architectures such as workstation clusters. Therefore, a prerequisite for such simulations is to be able to generate huge meshes in a parallel, distributed-memory fashion. Moreover, in the case where users would like to perform mesh adaptation, the latter must also be performed in parallel.

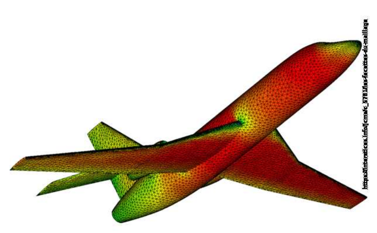
Context

- Numerical simulations are needed in multiple domains including:
  - thermonuclear fusion
  - aeronautics
  - meteorology, . . .

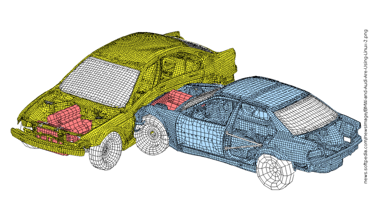




- Problems become bigger and more and more complex: numerical simulations cannot be run on a single workstation
  → Want of parallelized computations
  → Need to distribute data across the processors: domain decomposition
- Space discretization:
  - mesh



- Finite number of points on which values of the problem are computed, e.g.:
  - temperature
  - pressure
  - speed,. . .



- Solution precision depends on mesh quality:
  - need for remeshing

State of the art

- First class of parallel remeshing techniques:
  - Parallelization of existing sequential remeshing techniques
    - ∗ introduced in 1996 [1] for 2D meshes
    - ∗ 3D remeshing in 2000 for homogeneous meshes [8]
    - ∗ Delaunay triangulation in 2003 [3]
    - ∗ 3D remeshing for mixed meshes [7]
  - Problems:
    - ∗ Difficulties to parallelize each operator of the remesher
    - ∗ Remeshing some element requires neighborhood information
    - ∗ Too much communication between subdomains is required to achieve quality as high as in sequential processing

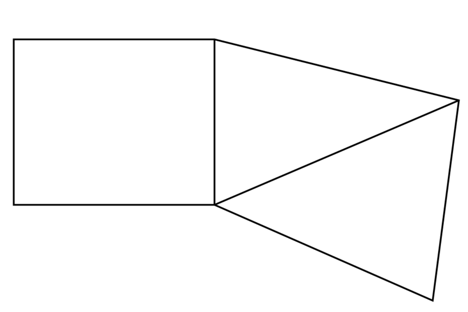
→ This class of parallel remeshing methods cannot handle large-size meshes distributed across a large number of processors

- Second class of parallel remeshing:
  - Re-use sequential remeshers in a parallel framework:
    - ∗ introduced in 2000 [4, 6]
    - ∗ 3D remeshing for mixed meshes [2]
    - ∗ remeshing with hierarchical transport [5]
    - ∗ multi-grid remeshing [9]

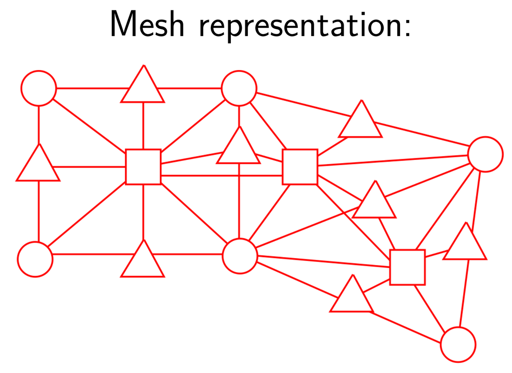
→ Our approach is to generalize this class of parallel remeshing techniques so as to allow for plugging-in any sequential remesher

Data structures for parallel remeshing

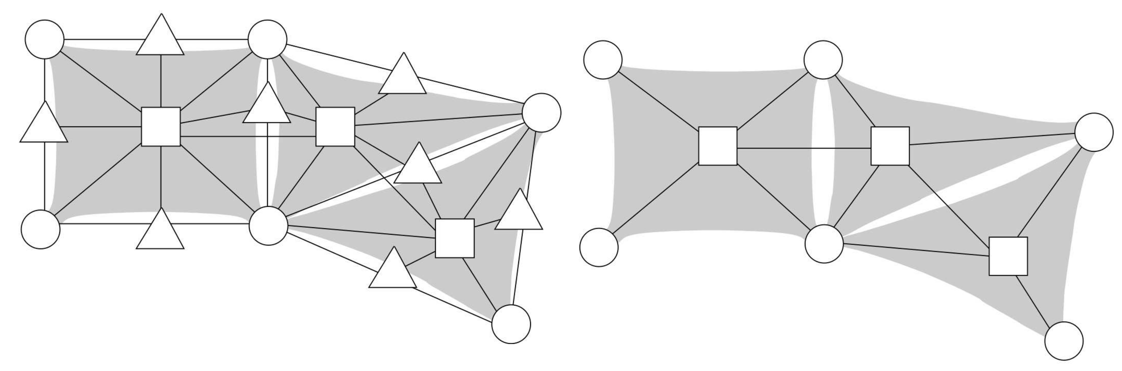
Mesh:



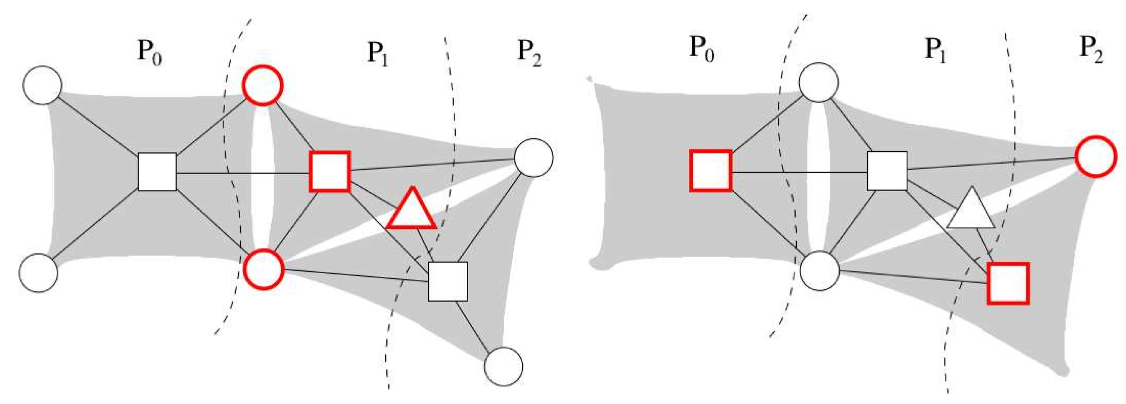
Mesh representation:



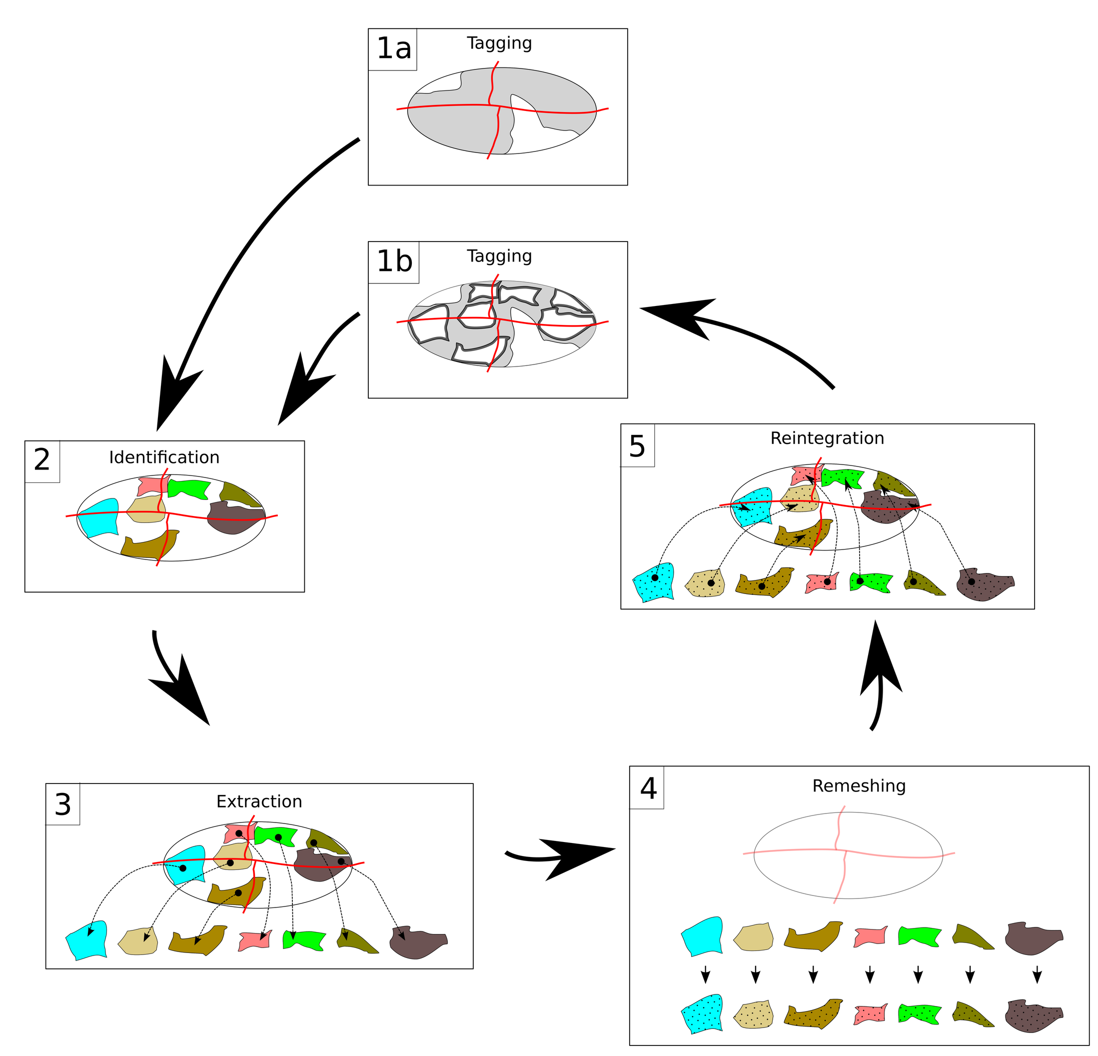
- The same mesh can lead to different enriched graphs
  - Depending on the requirements of the numerical schemes



- Distributed mesh on three processors



Parallel remeshing

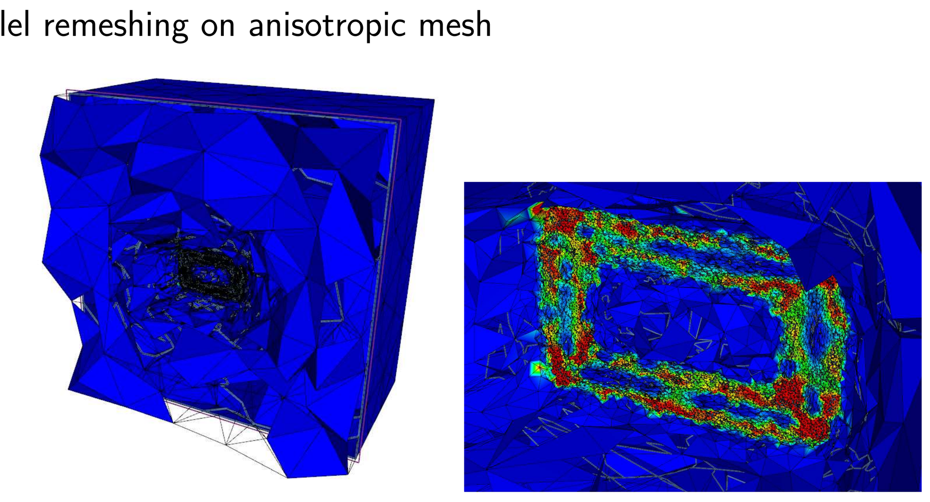


Some results

- Parallel remeshing on isotropic mesh

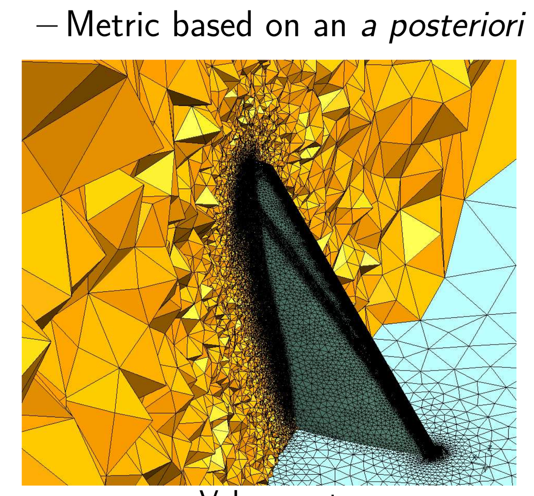
	PaMPA-MMG3D4 on 240 processors
Initial number of elements	27 044 943
Final number of elements	609 671 387
Elapsed time	00h34m59s
Elapsed time × number of PEs	139h56m
Smallest edge length	0.2911
Largest edge length	8.3451
Worst element quality	335.7041
% element quality between 1 and 2	98.92%
% edge length between 0.71 and 1.41	97.20%

- Parallel remeshing on anisotropic mesh

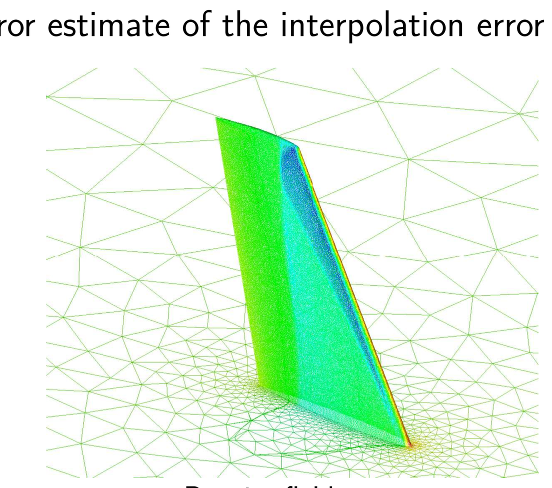


	PaMPA-MMG3D4 on 48 processors
Initial number of elements	715 791
Final number of elements	29 389 210
Elapsed time	00h34m
Elapsed time × number of PEs	27h12m
Smallest edge length	0.1116
Largest edge length	8.2191
Worst element quality	14.8259
% element quality between 1 and 2	99.61%
% edge length between 0.71 and 1.41	93.65%

- Mesh adaptation within adaptation loop:
  - Solve transsonic Eularian flow arround a M6 swing
  - Discretisation using a residual distribution scheme
  - Metric based on an *a posteriori* error estimate of the interpolation error



Volume cut



Density field

	PaMPA-MMG3D5 on 5 processors	MMG3D5 on 1 processor
Initial number of elements	570 775	
Final number of elements	5 089 972	5 132 259
Elapsed time	00h07m	00h29m
Elapsed time × number of PEs	00h34m	00h29m
Smallest edge length	0.0422	0.2092
Largest edge length	2.4416	2.4416
Worst element quality	26.42	9.12
% element quality greater than 0.5	99.66%	99.65%
% edge length between 0.71 and 1.41	96.62%	95.67%

Bibliography

[1] J. G. Castanos and J. E. Savage. The dynamic adaptation of parallel mesh-based computation. 1996.

[2] P. A. Cavallo, N. Sinha, and G. M. Feldman. Parallel unstructured mesh adaptation method for moving body applications. *AIAA journal*, 43(9):1937–1945, 2005.

[3] N. Chrisochoides and D. Nave. Parallel delaunay mesh generation kernel. *International Journal for Numerical Methods in Engineering*, 58(2):161–176, 2003.

[4] T. Coupez, H. Digonnet, and R. Ducloux. Parallel meshing and remeshing. *Applied Mathematical Modelling*, 25(2):153–175, 2000.

[5] H. Digonnet, M. Bernacki, L. Silva, and T. Coupez. Adaptation de maillage en parallèle, application à la simulation de la mise en forme des matériaux. In *Congrès Français de Mécanique Grenoble-CFM 2007*, page 6 pages, Grenoble, France, 2007. http://hdl.handle.net/2042/16046.

[6] C. Dobrzynski and J. Remacle. Parallel mesh adaptation. *International Journal for Numerical Methods in Engineering*, 2007.

[7] O. Lawlor, S. Chakravorty, T. Wilmarth, N. Choudhury, I. Dooley, G. Zheng, and L. Kalé. Parfum: a parallel framework for unstructured meshes for scalable dynamic physics applications. *Engineering with Computers*, 22:215–235, 2006. 10.1007/s00366-006-0039-5.

[8] L. Oliker, R. Biswas, and H. N. Gabow. Parallel tetrahedral mesh adaptation with dynamic load balancing. *Parallel Computing*, 26(12):1583–1608, 2000.

[9] M. Ramadan, L. Fourment, and H. Digonnet. A parallel two mesh method for speeding-up processes with localized deformations: application to cogging. *International Journal of Material Forming*, 2:581–584, 2009. 10.1007/s12289-009-0440-x.

① Inria ② université de BORDEAUX ③ ④ Bordeaux INP AQUITAINE