10th US National Congress on Computational Mechanics

ParTopS:
Compact Topological Framework for Parallel Fragmentation Simulations

Rodrigo Espinha[1]
Prof. Waldemar Celes[1]
Prof. Noemi Rodriguez[1]
Prof. Glaucio H. Paulino[2]

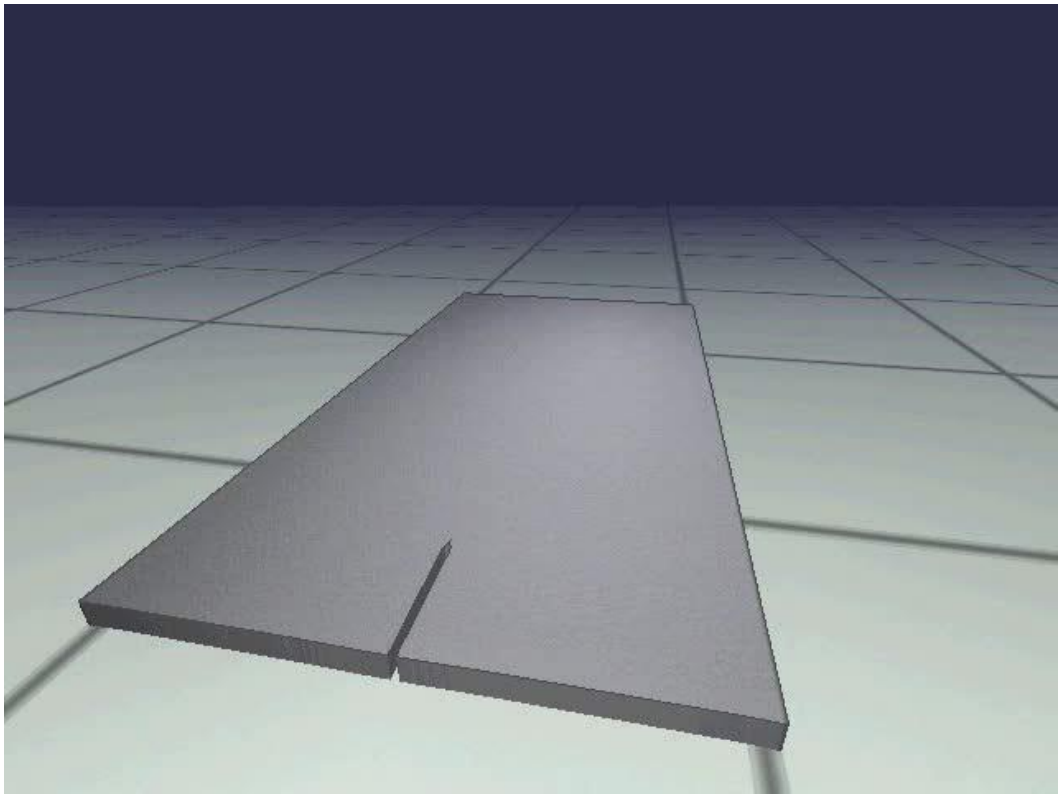
Acknowledgement

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

1. Computer Science Dept., Pontifical Catholic University of Rio de Janeiro, Brazil
2. Dept. of Civil and Environmental Engineering, University of Illinois at Urbana-Champaign

Motivation

- Large-scale fragmentation simulations using extrinsic cohesive models
  - Evolutive problems in space and time
  - Cohesive elements inserted dynamically
  - High mesh discretization level at crack tip region



ParTopS[1]

- Parallel framework for finite element meshes
  - Distributed mesh representation
    - Extension of the TopS[3,4] topological data structure
  - Parallel algorithm for inserting cohesive elements
    - Extension of the serial algorithm by Paulino et al.[2]

1. Espinha R, Celes W, Rodriguez N, Paulino GH (2009) ParTopS: compact topological framework for parallel fragmentation simulations. *Engineering with Computers, doi:10.1007/s00366-009-0129-2 (in press)*

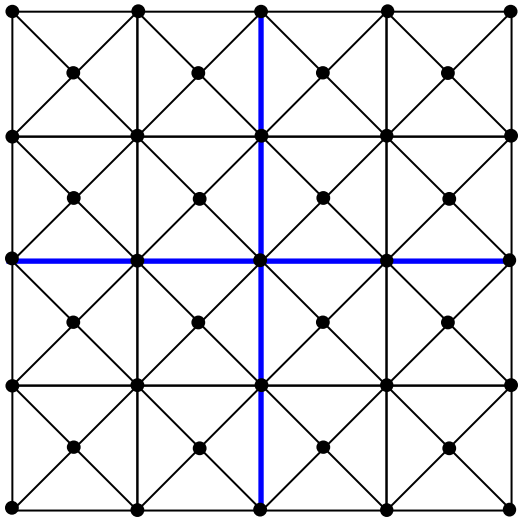
2. Paulino GH, Celes W, Espinha R, Zhang Z (2008) A general topology-based framework for adaptive insertion of cohesive elements in finite element meshes. Engineering with Computers 24(1):59-78

3. Celes W, Paulino GH, Espinha R (2005) Efficient Handling of Implicit Entities in Reduced Mesh Representations. Journal of Computing and Information Science in Engineering, Special Issue on Mesh-Based Geometric Data Processing, 5(4), pp. 348-359

4. Celes W, Paulino GH, Espinha R (2005) A compact adjacency-based topological data structure for finite element mesh representation. Int J Numer Methods Eng 64(11):1529–1565

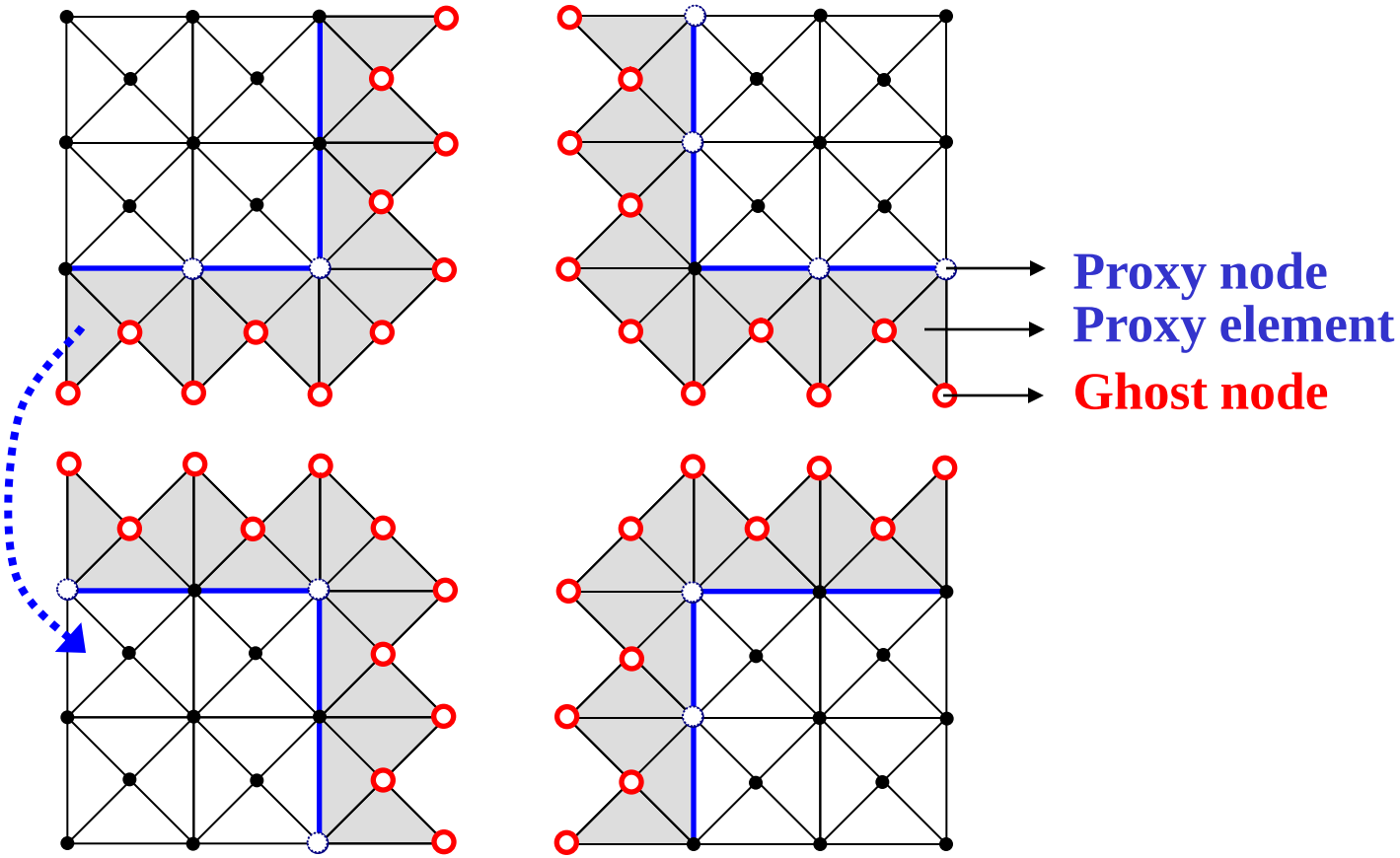
Distributed mesh representation

- Sample mesh



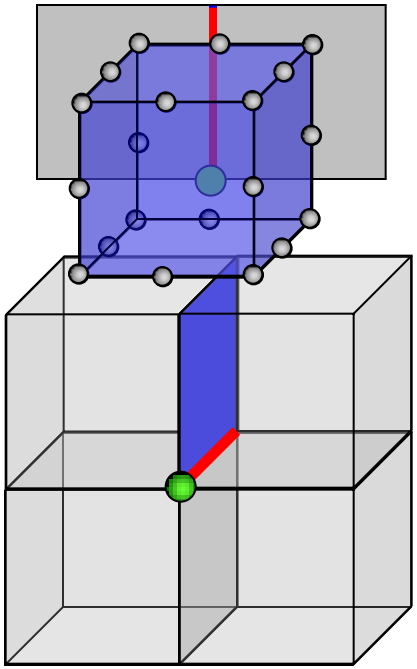
Distributed mesh representation

Communication layer



Topological entities

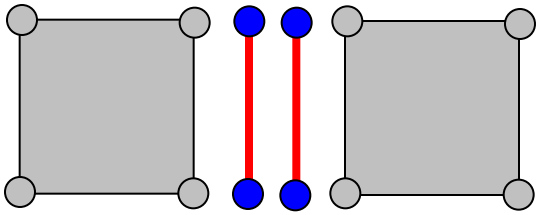
- Element
- Node
- Vertex
- Edge
- Facet
  - Interface between elements



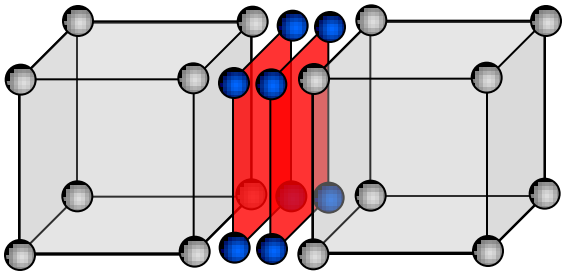
Cohesive elements

- True extrinsic elements
  - Inserted “on the fly”, where needed and when needed
  - No element activation or springs
- Two-facet elements
- Inserted between two adjacent bulk elements

2D

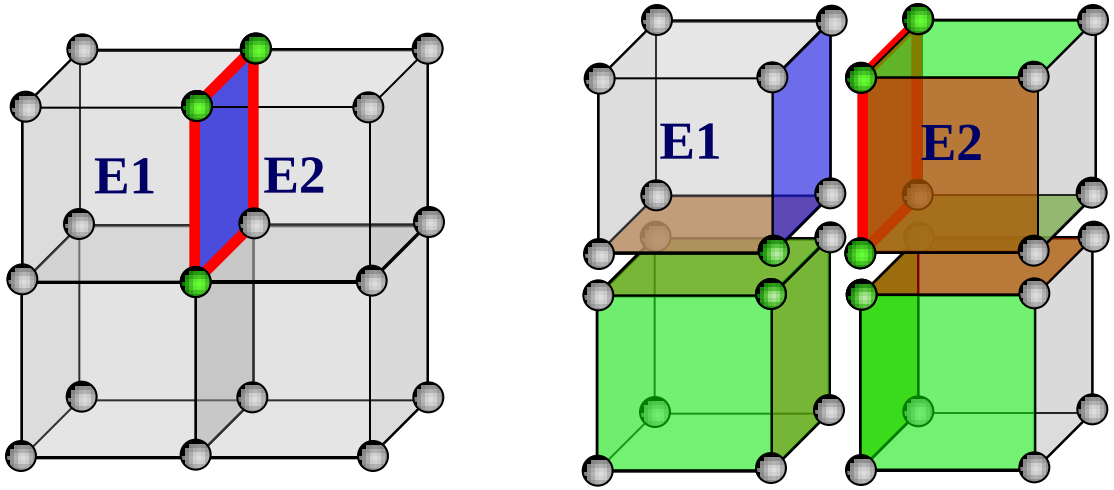


3D



Serial insertion of cohesive elements[1]

- Insert cohesive element at a facet shared by E1 and E2
  1. Create cohesive element at facet
  2. Traverse non-cohesive elements adjacent to edges of E2
     - If E1 is not visited, duplicate edge and mid-nodes (if any)
  3. Traverse non-cohesive elements adjacent to vertices of E2
     - If E1 is not visited, duplicated vertex

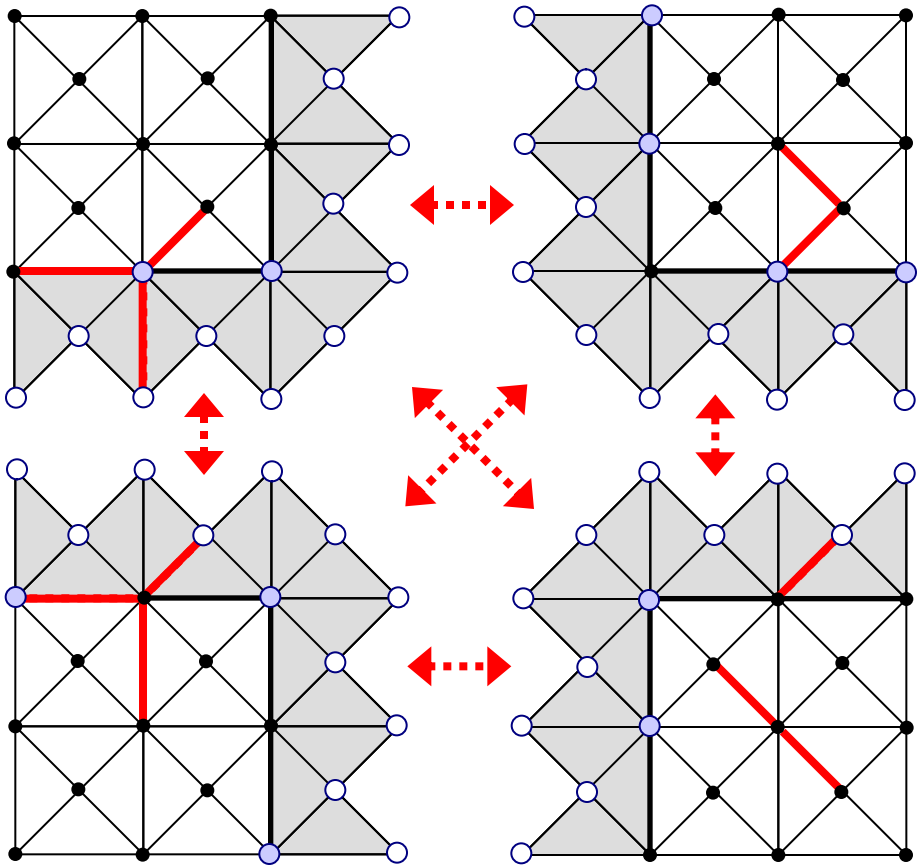


1. Paulino GH, Celes W, Espinha R, Zhang Z (2008) A general topology-based framework for adaptive insertion of cohesive elements in finite element meshes. Engineering with Computers 24(1):59-78

Parallel insertion of cohesive elements

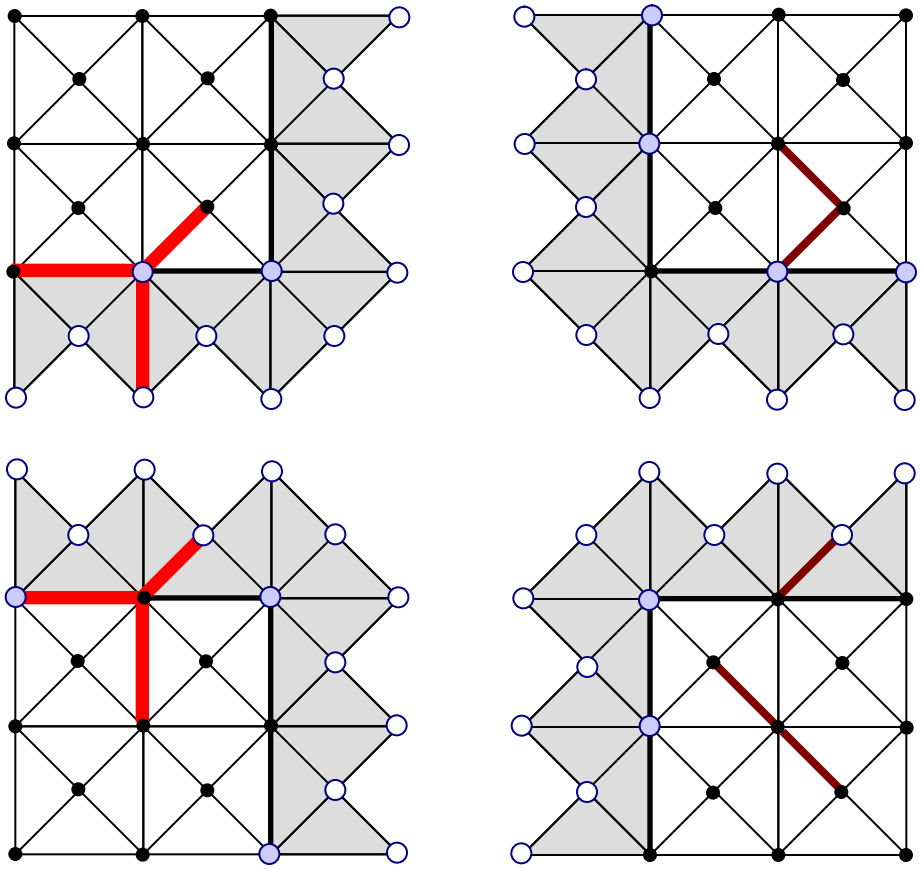
- Simulation loop
  - At each step
    - Analysis application identifies fractured facets
    - Insert cohesive elements
      1. Insert elements at local / proxy facets (serial algorithm)
      2. Update new proxy entities
      3. Update affected ghost entities

Identification of fractured facets



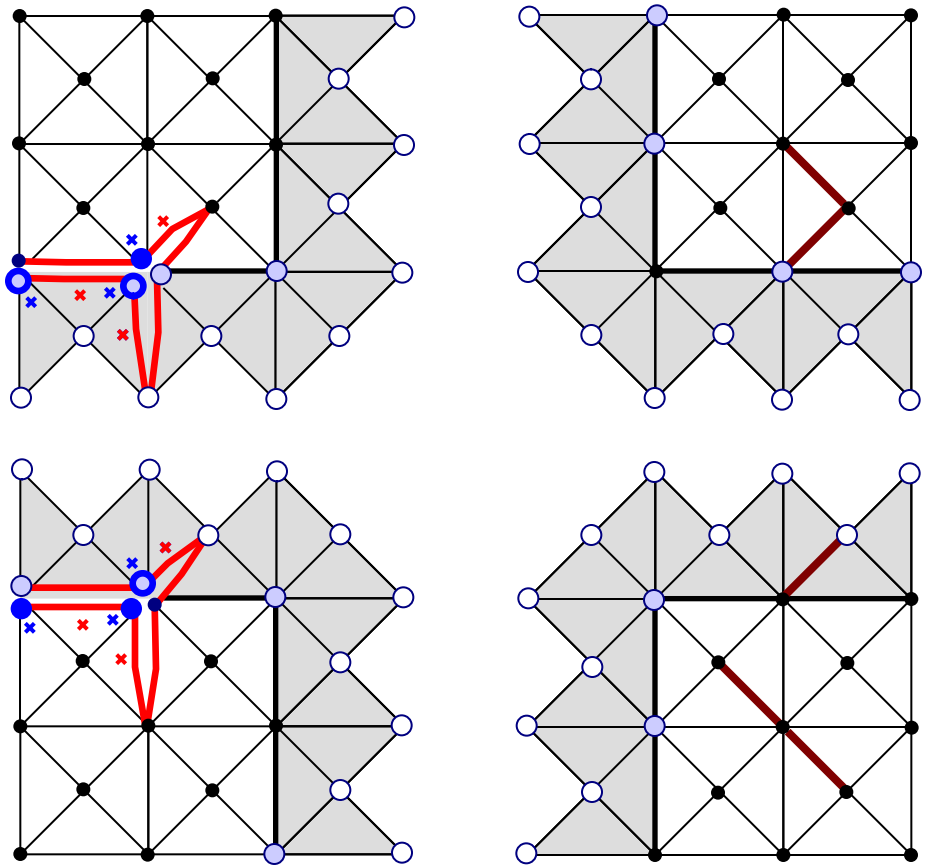
1. Insert elements at local and proxy facets

- Modified serial algorithm
  - Local topological consistency
  - All copies of a new element or node are owned by the same partition
  - Ghost nodes are not duplicated at this moment
    - Due to dependence on remote information



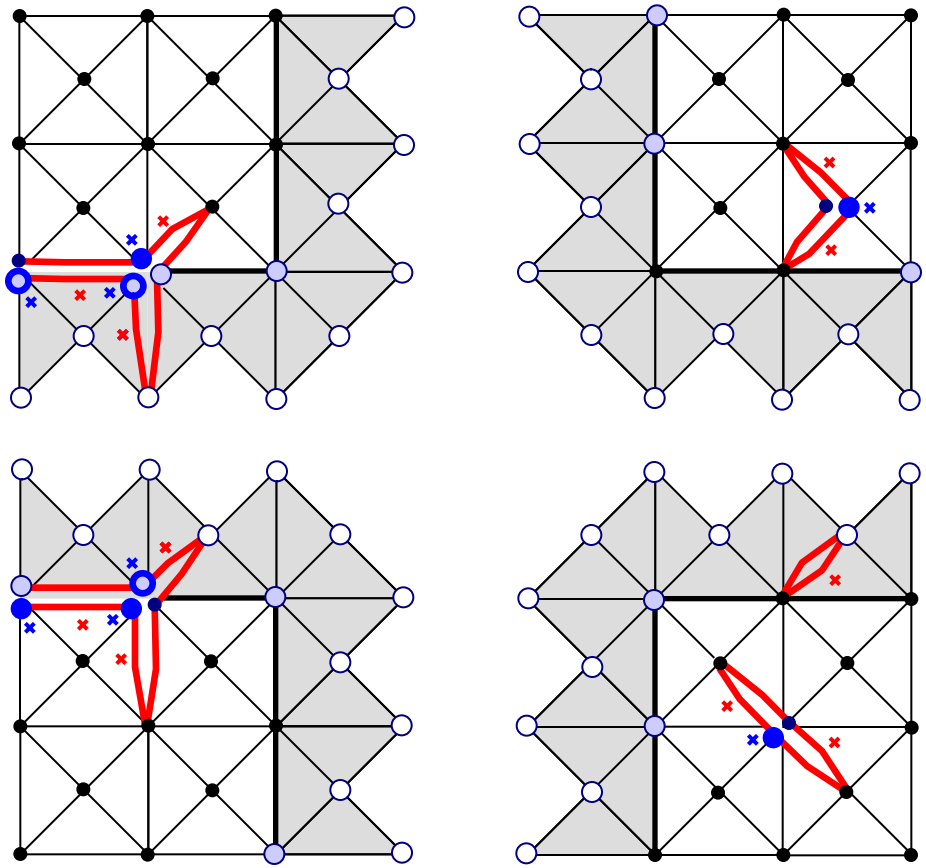
1. Insert elements at local and proxy facets

- Uniform criterion for selecting proxy entity's owner partition
  - e.g. partition of the adjacent element with smallest *id*
- Symmetrical topological results in both partitions
  - No need for communication to synchronize topology



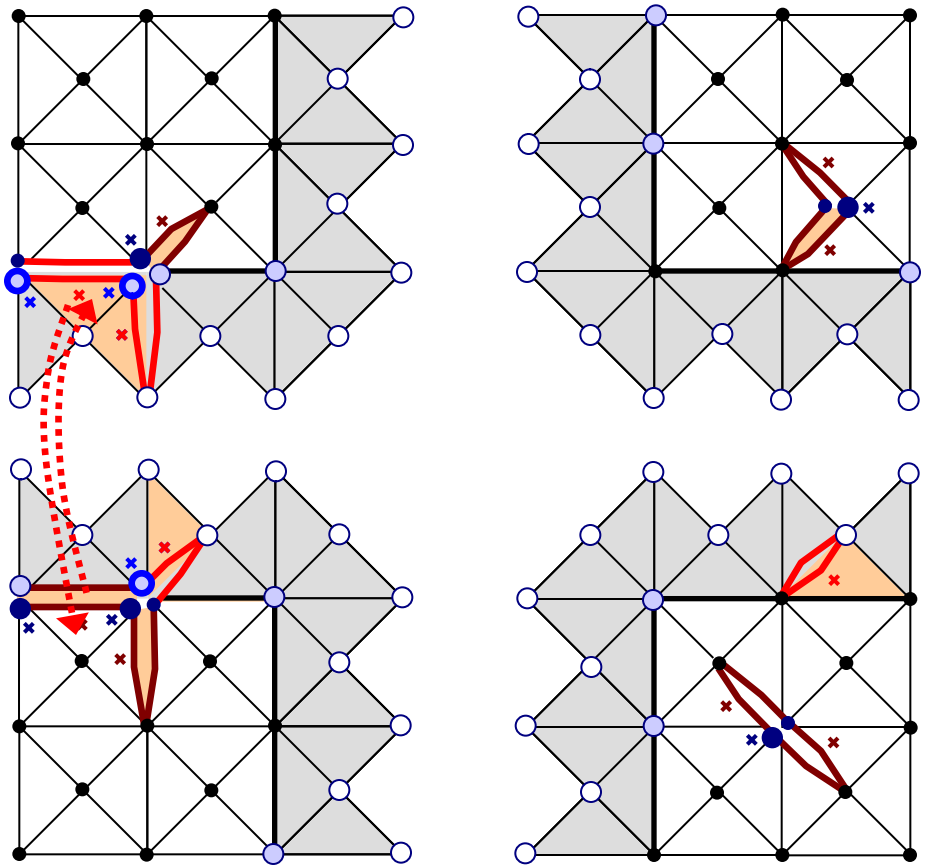
1. Insert elements at local and proxy facets

- At the end of Phase 1
  - Local topology is consistent
  - References from proxies to real entities still have to be computed



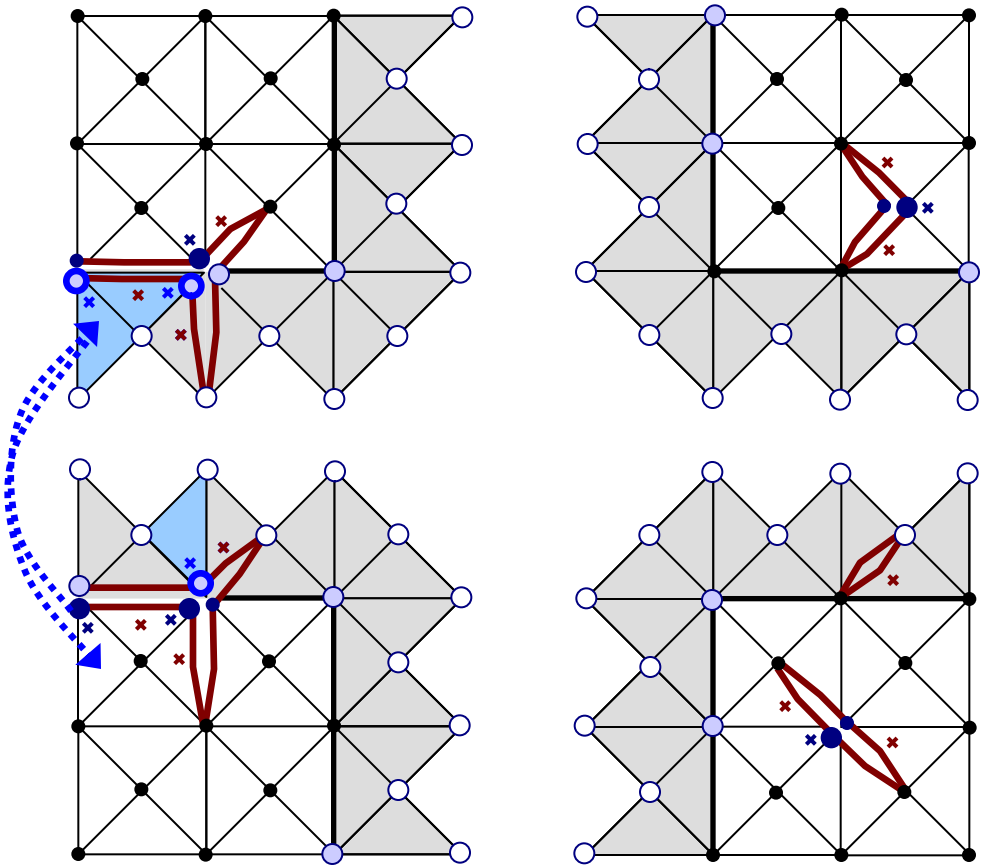
2. Update new proxy entities

- Create references from the new proxy elements and nodes to the corresponding real entities
  - Adjacent elements are used for requesting data



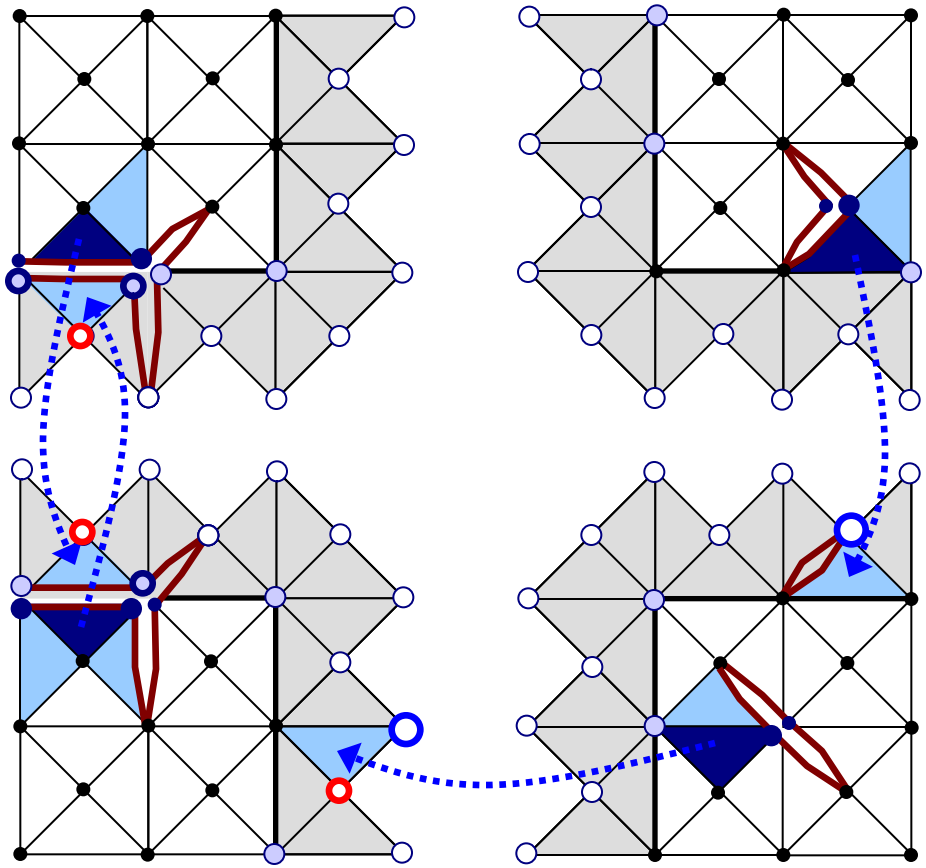
2. Update new proxy entities

- Same procedure applied to nodes

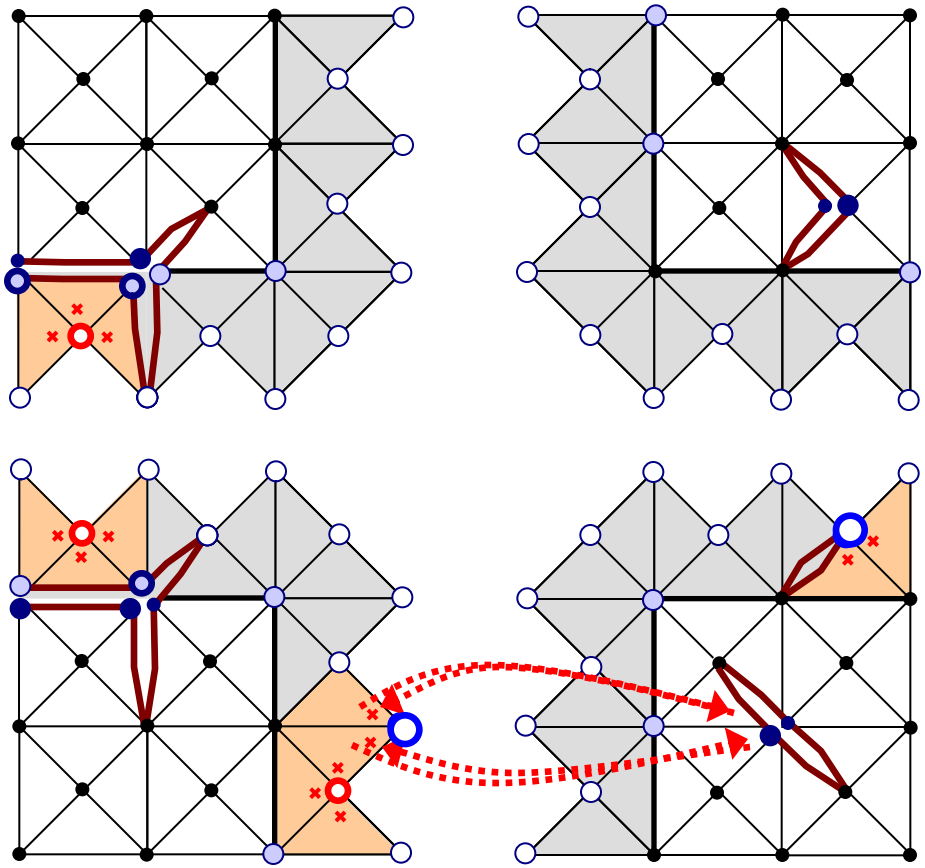


3. Update affected ghost entities

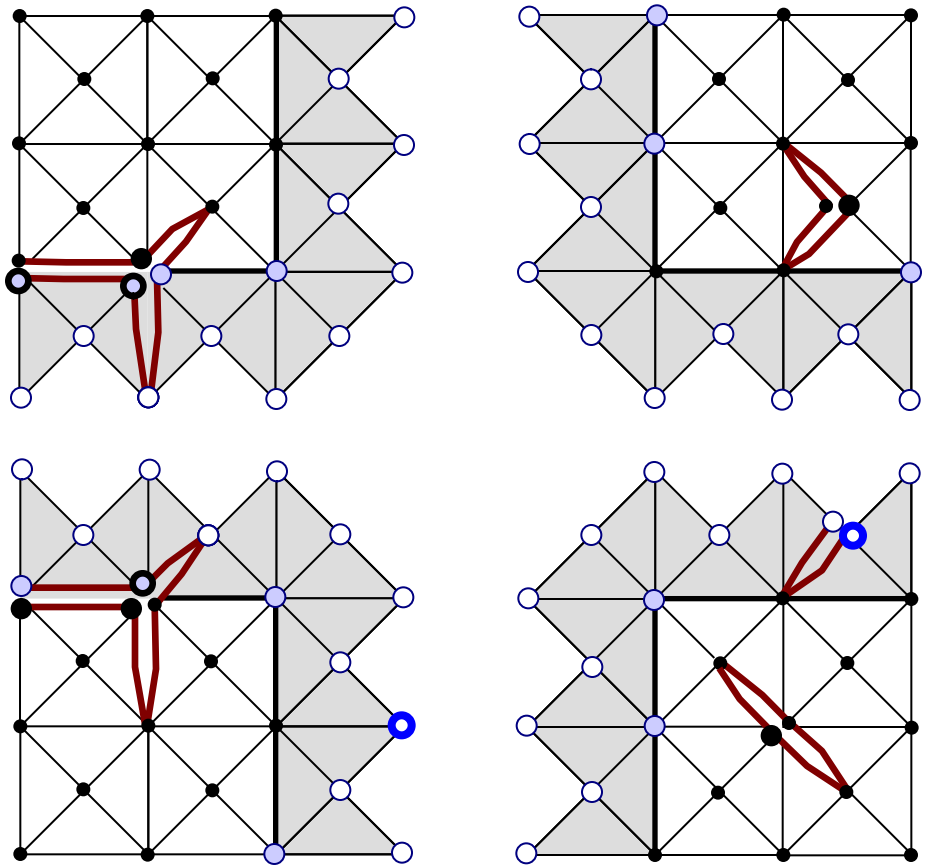
- Replace ghost nodes affected by remote cohesive elements
  - "Per-element" approach
    - Local elements adjacent to duplicated nodes are used to notify other partitions



3. Update affected ghost entities

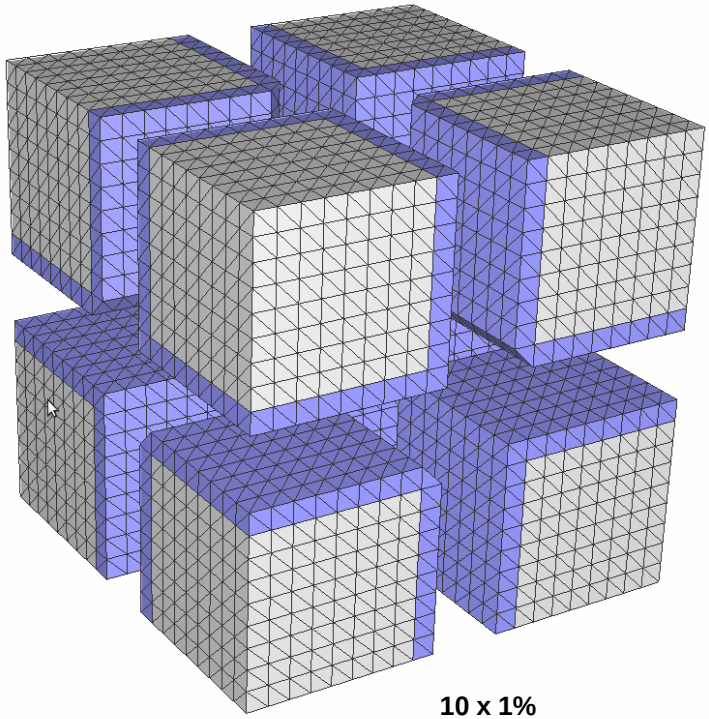


Resulting mesh

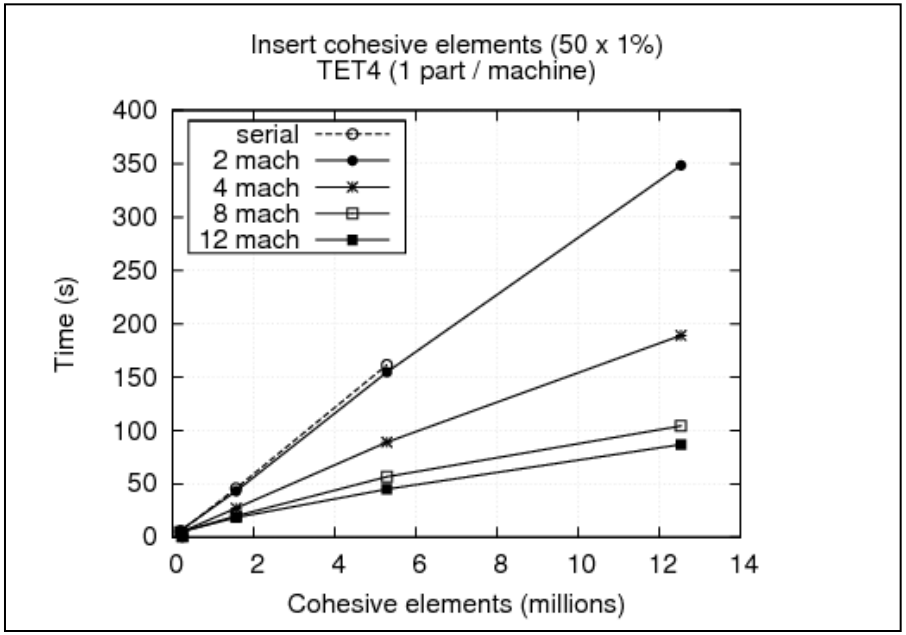


Verification

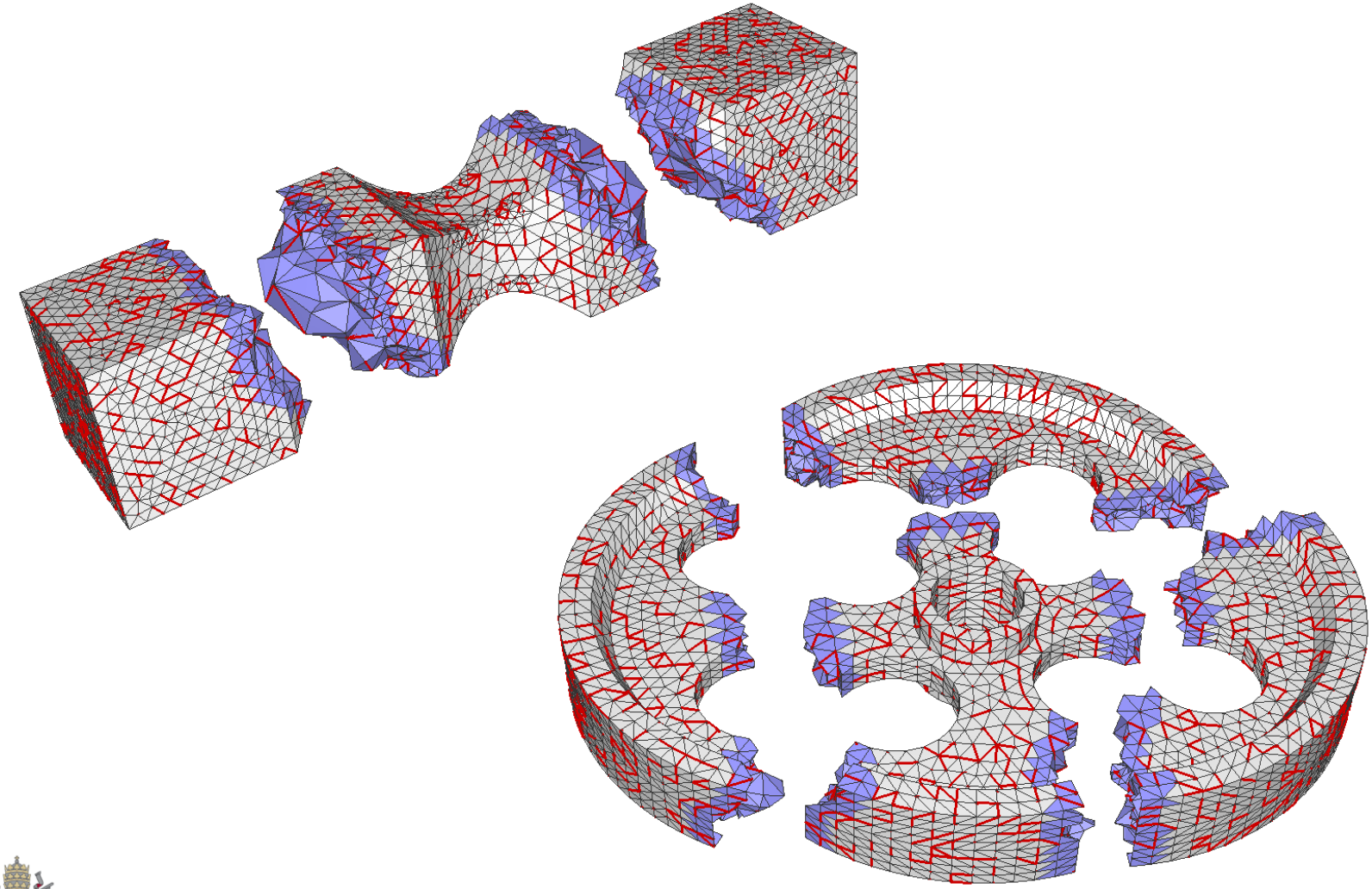
- Cluster of 12 machines
  - Intel(R) Pentium(R) D processor 3.40 GHz (dual core) with 2GB of RAM, Gigabit Ethernet
- Cohesive elements randomly inserted at 1% of internal facets x 50 steps
- Meshes with different discretizations and types of elements (T3, T6, Tet4, Tet10)



10 x 1%



Results



Concluding remarks

- ParTopS: parallel topological framework
  - Dynamic insertion of cohesive elements
    - True extrinsic cohesive elements
      - Inserted “on the fly”, where needed and when needed
    - Generic branching patterns are supported
    - General 2D or 3D meshes
    - Executed on a limited number of machines
      - However, linear scaling is expected

Next steps

- Other parallel adaptive operators
  - E.g. refinement & coarsening
- Integrate mechanical analysis computer code

Thank you!