10th JLPC Workshop

PaMPA : Parallel Mesh Partitioning and Adaptation

François Pellegrini & Cédric Lachat & Cécile Dobrzynski

Novemver 25, 2013

Contents

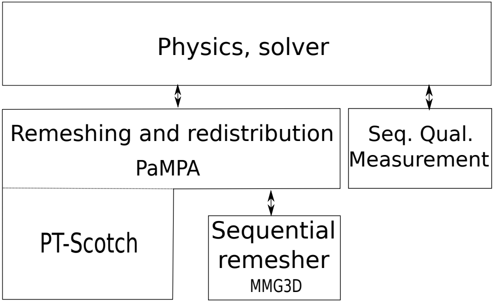
Common needs of solvers regarding meshes

- Handling of mesh structures
- Distribution of meshes across the processors of a parallel architecture
  - Handling of load balance
- Data exchange across neighboring entities
- Iteration on mesh entities
  - Entities of any kind: e.g. elements, faces, edges, nodes, ...
  - Entity sub-classes: e.g. regular or boundary faces, ...
  - Inner or frontier entities with respect to neighboring processors
  - Maximization of cache effects thanks to proper data reordering
- Dynamic modification of mesh structure
  - Dynamic redistribution
- Adaptive remeshing

Contents

What is PaMPA

- PaMPA: "Parallel Mesh Partitioning and Adaptation"
- Middleware library managing the parallel repartitioning and remeshing of unstructured meshes modeled as interconnected valuated entities
- The user can focus on his/her "core business":
  - Solver
  - Sequential remesher
    - Coupling with MMG3D provided for tetrahedra

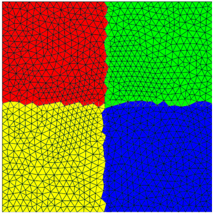


Features of version 1.0

- Overlap greater than 1
- Point-to-point or collective communications
- Parallel I/O
- Parallel partitioning
- Parallel remeshing based on sequential remesher

Renumbering

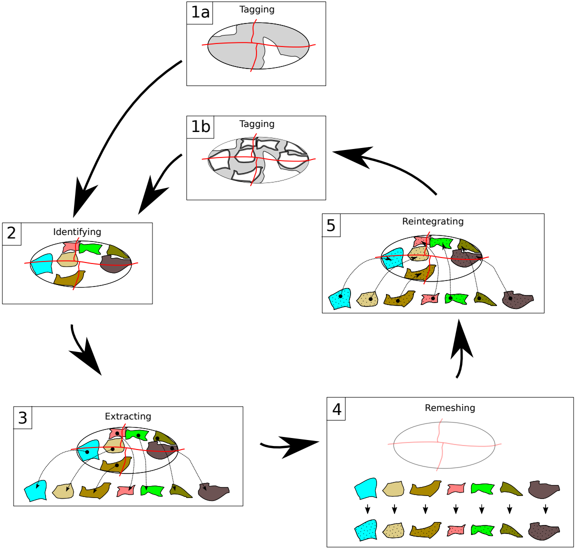
- Execution speed greatly benefits from memory locality
  - Algorithms must take into account memory hierarchy
- Adjacency lists already take advantage of L1 locality
  - L2 locality must be increased
    - regards ordering the entities themselves
- Principle : over-decompose mesh into subdomains of about 500 elements (so as to fit in L2 cache)
  - renumbering elements inside each subdomain
  - order other entities according to element numbers



Parallel remeshing (1/2)

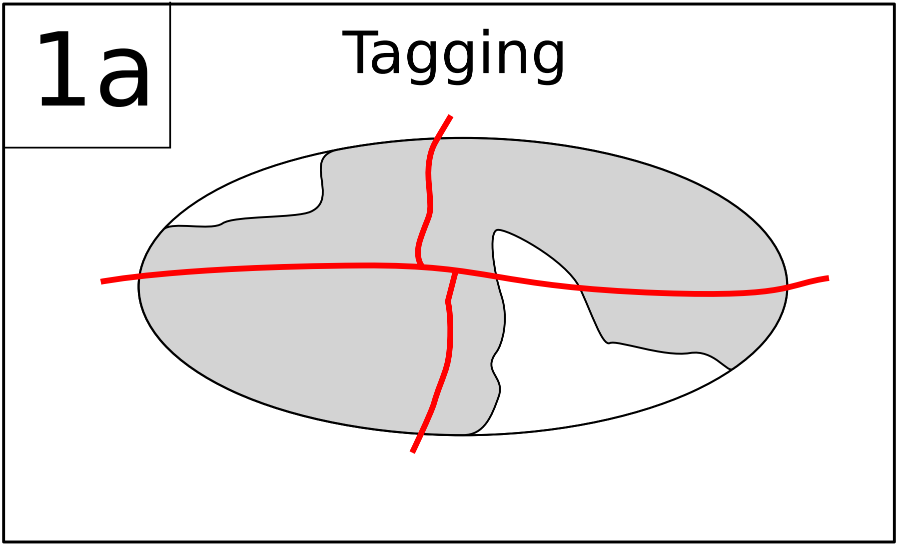
- Parallel remeshing based on user criteria:
  - Smallest and largest edge lengths
  - Maximum quality of wrong element
- Multi-pass method:
  - Before each pass, mesh is checked according to user criteria
    - Each element which not verify criteria is tagged to be remeshed
  - Each pass uses iterative method to remove tag on all elements
  - Stopped when number of elements to be remeshed divided by number of previous pass is less than a ratio defined by user

Parallel remeshing (1/7)

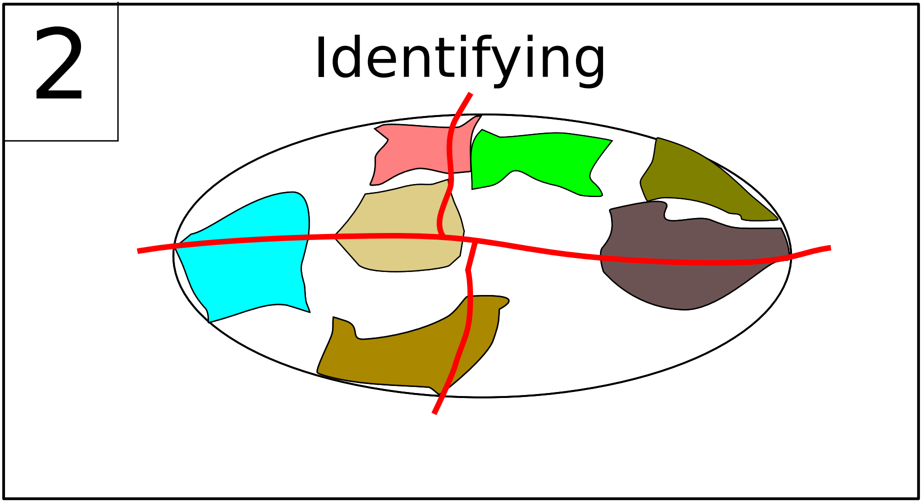


Next section

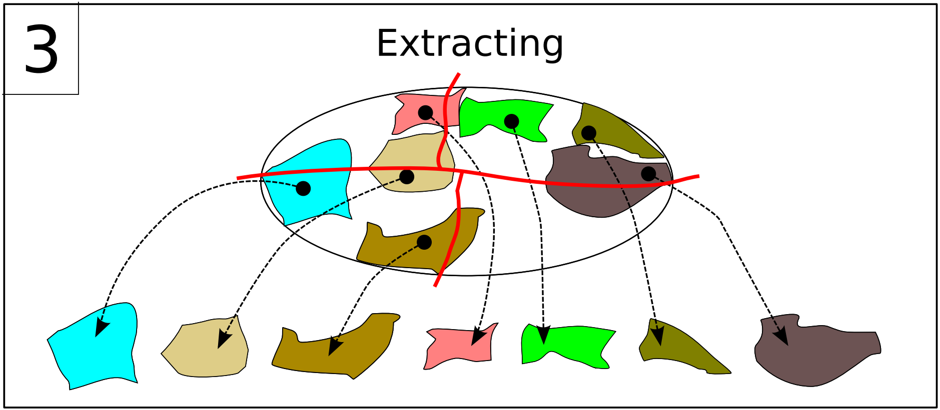
Parallel remeshing (1/7)



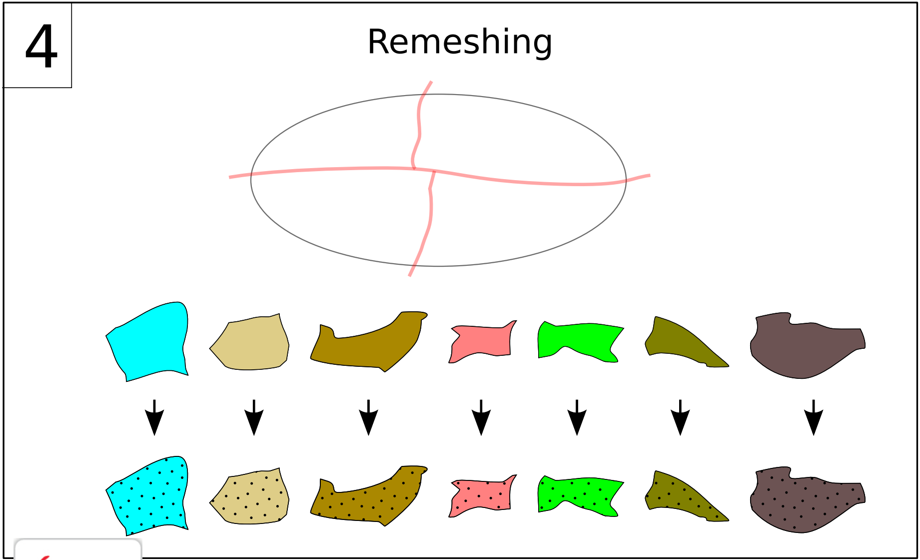
Parallel remeshing (1/7)



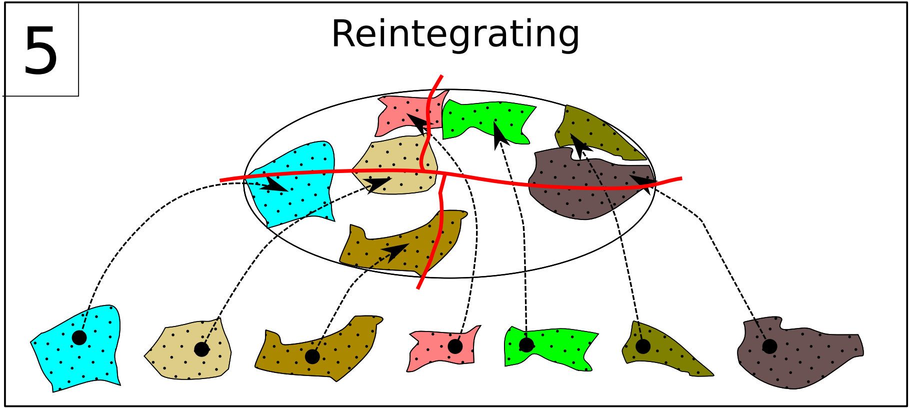
Parallel remeshing (1/7)



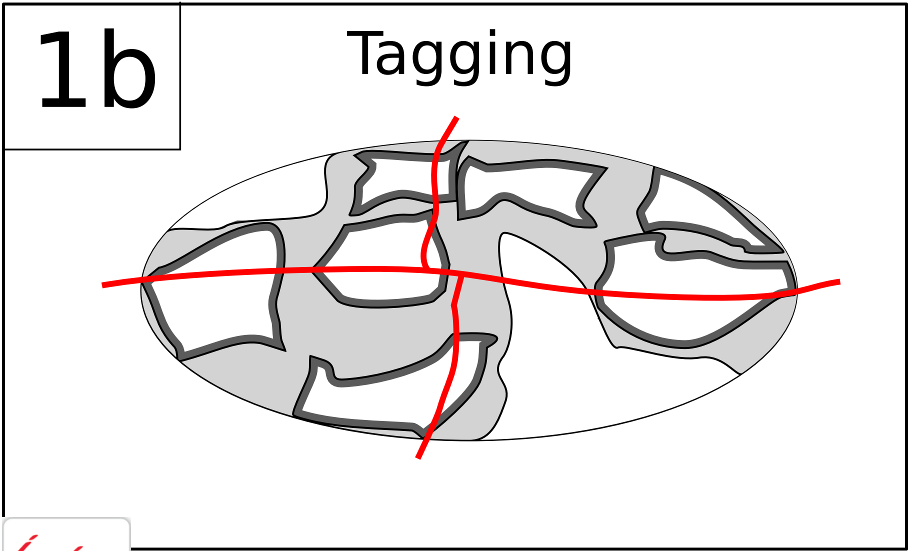
Parallel remeshing (1/7)



Parallel remeshing (1/7)

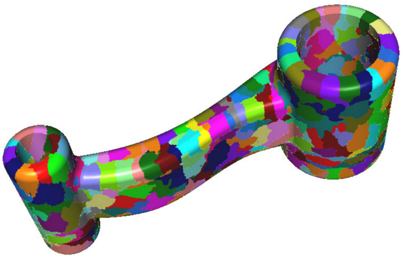


Parallel remeshing (1/7)



Contents

Some results (1/4)



Before remeshing	
Number of elements	2 423 029
Number of nodes	1 071 626

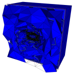
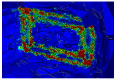
Some results (2/4)

	MMG3D on 1 processor	PaMPA-MMG3D on 24 processors
Processor frequency (GHz)	2,40	3,06
Used memory (kb)	27 588 940	51 116 044
Elapsed time	17h15m12s	00h21m14s
Number of elements	108 126 515	115 802 876
Smallest edge length	0.1470	0.1395
Largest edge length	6.3309	11.2415
Worst element quality	294.2669	294.2669
Element quality between 1 and 2	99.65%	99.38%
Edge length between 0.71 and 1.41	97.25%	97.65%

Some results (3/4)

	PaMPA-MMG3D isotropic tetrahedral mesh on 120 processors
Elapsed time	00h34m54s
Number of elements	318 027 812
Smallest edge length	0.2862
Largest edge length	6.2161
Worst element quality	235.6651
Element quality between 1 and 2	99.58%
Edge length between 0.71 and 1.41	97.91%

Some results (4/4)

	PaMPA-MMG3D anisotropic tetrahedral mesh on 6 processors
Elapsed time	00h08m07s
Number of elements	11 687 798
Smallest edge length	0.1265
Largest edge length	11.0146
Worst element quality	45.1691
Element quality between 1 and 2	98.21%
Edge length between 0.71 and 1.41	93.15%

Contents

Work in progress

- Release of version 1.0
  - Available at the end of September from Inria Gforge
  - Licensed under GPL
- Multigrid meshes

Contents

Upcoming features

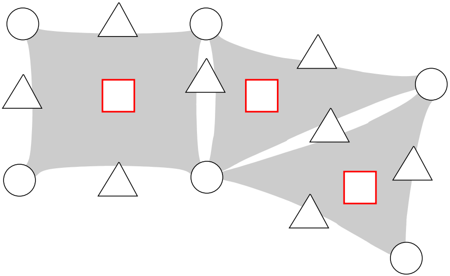
- Code industrialisation
- Mesh definition with a grammar
- Face orientation and displacement
- Unbreakable relations
  - Partitioner will not cut these edges
  - E.g. to implement DG methods
- Periodic meshes
- Parallel I/O with HDF5
- Parallel mesh adaptation scalability

Thank you

Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

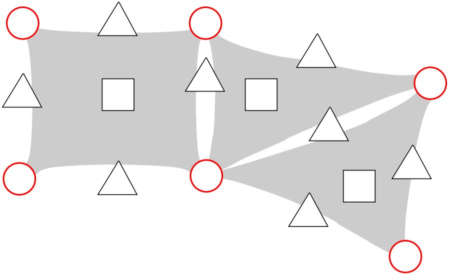
Top-level mesh entity
May bear some data (volume, pressure, etc.)



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

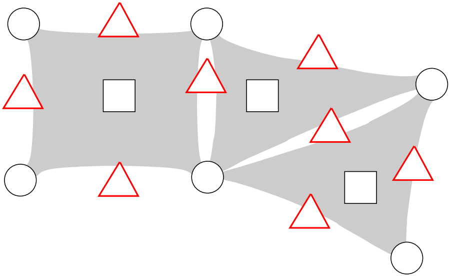
May bear some data (geometry, etc.)



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

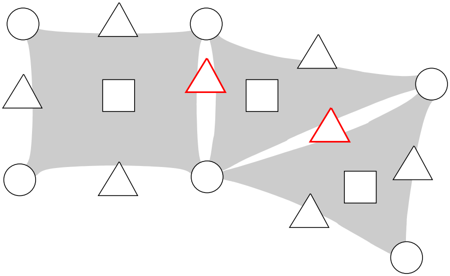
May bear some data (flux, etc.)



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

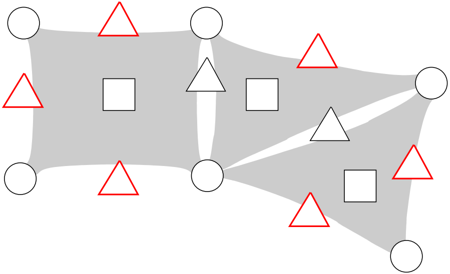
Regular mesh edge



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

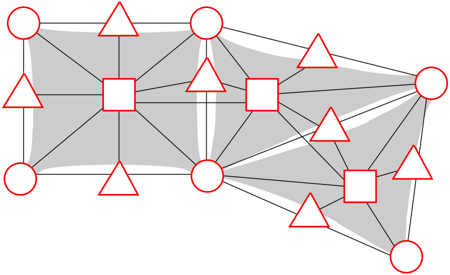
Boundary mesh edge



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

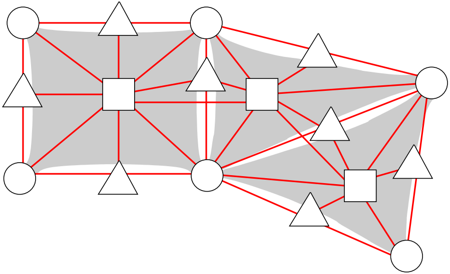
What all entities are in fact. . .



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

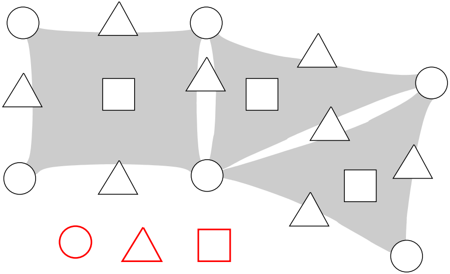
Subset of edges between vertices belonging to prescribed entity types



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

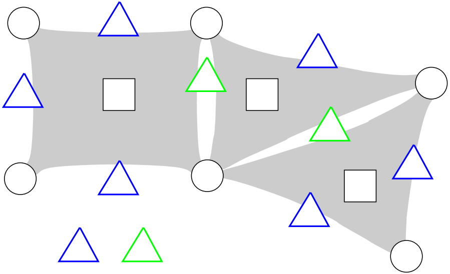
Subset of vertices bearing the same data



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

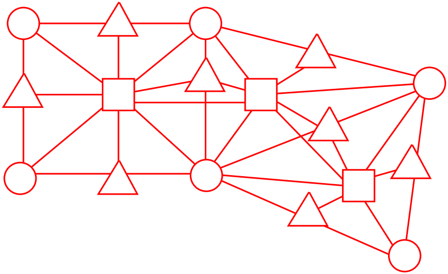
Subset of entity vertices that may bear additional specific data



Definitions

- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**

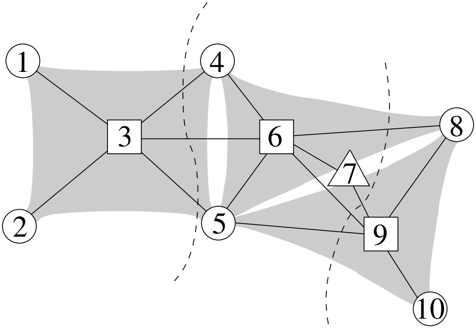
Whole set of vertices and relations
Every vertex belongs to one and only one entity (and sub-entity)



Global vue

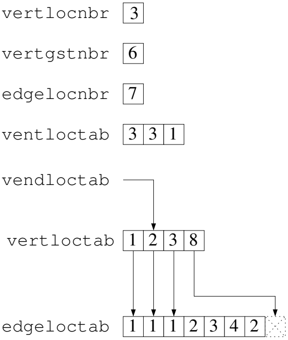
- All vertices have a global unique number

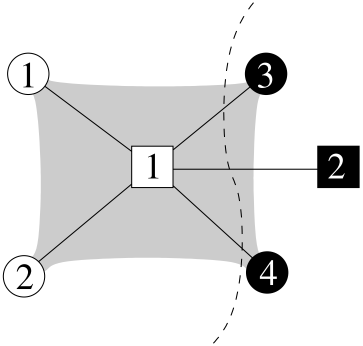
baseval	1			
enttglbnbr	3			
proccnttab	3	4	3	
procvrttab	1	4	8	11



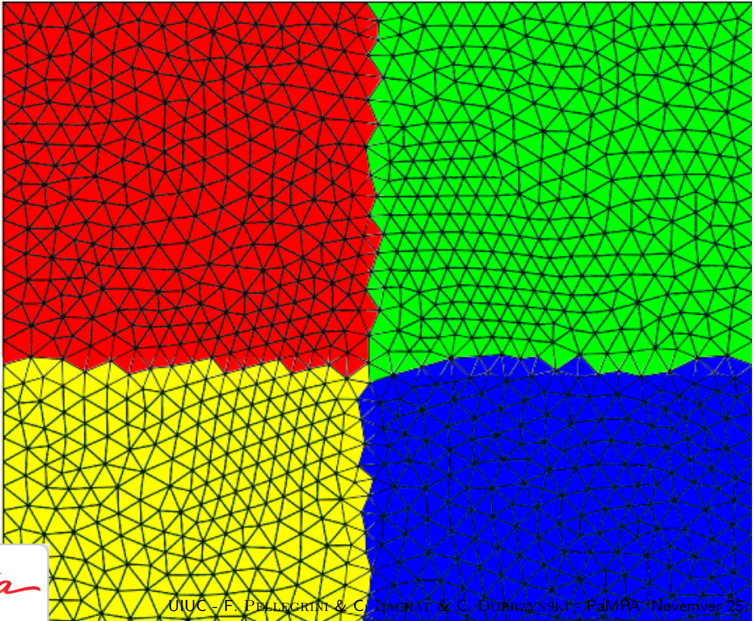
Local vision of process 0

- All local and ghost vertices have a compact local index
  - Per-entity numbering



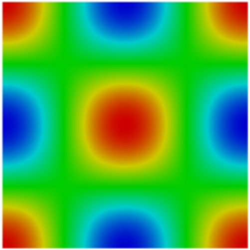


Renumbering



Definition

- Solving 2D Poisson equation:
  - $\Delta u(x,y) = f(x,y)$
  - $g(x,y) = u(x,y)$ on the boundary Γ
- Test case:
  - $f(x,y) = -2 * \cos(x) * \cos(y)$ in the domain Ω
  - $g(x,y) = \cos(x) * \cos(y)$ on the boundary Γ
  - $u(x,y) = \cos(x) * \cos(y)$



Mesh properties

- Entities:
  - Elements
  - Nodes
  - Boundary edges
- Relations:
  - Element to element
  - Element to node
  - Element to boundary edge
  - Node to node
- Overlap of size 1
- Values:
  - Coordinates and solution on nodes
  - Type on boundary edges
  - Area, volume on elements

All steps

```
! On all processors:
CALL DistributedMesh() ! Build PaMPA distributed mesh:
!                        1- Read in parallel a centralized mesh
!                        2- Call PaMPA mesh partioner
!                        3- Redistribute distribute mesh
CALL ElementVolume()
CALL InitializeMatrixCSR()

! Solution computation
!-----------------------

CALL InitSol()
CALL FillMatrix()
CALL SolveSystem()
CALL WriteDistributedMeshAndSolFiles()
```

FillMatrix

```
RHS = 0.
CALL PAMPAF_dmeshItInitStart (dm, ENTITY_ELEM, PAMPAF_VERT_ANY, it_vrt, ierr)
CALL PAMPAF_dmeshItInit (dm, ENTITY_ELEM, ENTITY_NODE, it_ngb, ierr)
DO WHILE (PAMPAF_itHasMore (it_vrt))
  jt   = PAMPAF_itCurEnttVertNum (it_vrt)
  Volt = VolEl (jt)
  ngb  = 0
  CALL PAMPAF_itStart (it_ngb, jt, ierr)
  DO WHILE (PAMPAF_itHasMore (it_ngb))
    ngb = ngb + 1
    is  = PAMPAF_itCurEnttVertNum (it_ngb)
    NuElemt(ngb)       = is
    CoorElemt(:, ngb) = Coor(:, is)
    PAMPAF_itNext (it_ngb)
  END DO
  CALL GradPhi (CoorElemt(:,1), CoorElemt(:,2), CoorElemt(:,3), GrdPhi)
  DO i = 1, Nsmplx
    is = NuElemt(i)
    DO j = 1, Nsmplx
      js   = NuElemt(j)
      JJac = Volt * Sum (GrdPhi(:, i) * GrdPhi(:, j))
      CALL assembly_addCSR (JJac, is, js)
    END DO ! loop on j
    RHS(is) = RHS(is) - Volt * SourceTerm (Coor(1, is), Coor(2, is)) / Nsmplx
  END DO ! loop on i
  PAMPAF_itNext (it_vrt)
END DO
```

Solve system: Jacobi (1/2)

```
UaPrec = 0.          ! Suppose A = L + D + U, system to solve : A x = b
CALL PAMPAF_dmeshItInit(dm, ENTITY_NODE, ENTITY_NODE, it_ngb , ierr)
DO irelax = 1, Nrelax
  res       = 0.
  CALL PAMPAF_dmeshItInitStart(dm, ENTITY_NODE, PAMPAF_VERT_BOUNDARY, it_vrt ,  ierr)
  DO WHILE (PAMPAF_itHasMore(it_vrt))
    is = PAMPAF_itCurEnttVertNum(it_vrt)
    CALL PAMPAF_dmeshMatLineData(dm, ENTITY_NODE, is , I1 , I1Fin , ierr)
    CALL PAMPAF_itStart( it_ngb ,  is ,  ierr)
    res0      = RHS(is)                              ! res0  = b

    iv = i1
    DO WHILE (PAMPAF_itHasMore(it_ngb))
      js = PAMPAF_itCurEnttVertNum(it_ngb)
      PAMPAF_itNext(it_ngb)
      res0  = res0 - MatCSR%Vals(iv) * UaPrec(js) !res0  = b - (L + U) x^n
      iv = iv + 1
    END DO
    Ua(is)  = res0 / MatCSR%Diag(is)              !x^n+1 = ( b - (L + U) x^n )/D
    PAMPAF_itNext(it_vrt)
  END DO

  CALL PAMPAF_dmeshHaloValueAsync(dm, ENTITY_NODE, PAMPA_TAG_SOL, req , ierr)
```

Solve system: Jacobi (2/2)

```
   CALL  PAMPAF_dmeshItInitStart(dm, ENTITY_NODE, PAMPAF_VERT_INTERNAL, it_vrt , ierr)
   DO WHILE (PAMPAF_itHasMore(it_vrt))
     is = PAMPAF_itCurEnttVertNum(it_vrt)
     CALL PAMPAF_dmeshMatLineData(dm, ENTITY_NODE, is , i1 , i1Fin , ierr)
     CALL PAMPAF_itStart( it_ngb ,  is ,  ierr)
     res0      = RHS(is)                              ! res0   = b

     iv = i1
     DO WHILE (PAMPAF_itHasMore(it_ngb))
       js = PAMPAF_itCurEnttVertNum(it_ngb)
       PAMPAF_itNext(it_ngb)
       res0   = res0 - MatCSR%Vals(iv) * UaPrec(js)  !res0   = b - (L + U) x^n
       iv = iv + 1
     END DO
     Ua(is)   = res0 / MatCSR%Diag(is)                !x^n+1 = ( b - (L + U) x^n )/D
     PAMPAF_itNext(it_vrt)
   END DO

   CALL PAMPAF_dmeshHaloWait(req , ierr)

   UaPrec     = Ua
END DO ! end loop on irelax
```