HOSCAR

PaMPA : Parallel Mesh Partitioning and Adaptation

François Pellegrini & Cédric Lachat & Cécile Dobrzynski

Septemper 3, 2013

Contents

State of the art

Common needs of solvers regarding meshes

What is PaMPA

Example: Laplacian equation using P_1 finite element method

Some results

Work in progress

Upcoming features

Contents

State of the art
- Parallel remeshers
- Load balancing
- Existing tools

Context

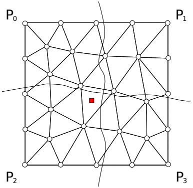
- Distributed meshes
- Subdomain decomposition
- Data exchanges between subdomains

Parallel mesh adaptation algorithms

- Parallel mesh generation (N. Chrisochoides 2005 [3])
  - Delaunay triangulation
  - Frontal method
  - Refinement by subdivisions (B. G. Larwood 2003 [7])
- Communication between subdomains:
  - Data migration
  - Matching algorithms

Problems induced by parallelism

- High complexity to parallelize remesh methods
  - Too much communication on boundaries
  - Boundaries not remeshed:
    - Local remeshing



Sequential algorithms

- Methods:
  - Moving nodes and remeshing locally (O. Hassan 2006 [6])
  - Coarse-grain parallel remeshing through multiple successive sequential remeshings (U. Tremel 2006 [8]):
    - Finding zones to remesh according to error estimator (T. Coupez 2000 [4])
    - Identifying zones to remesh in parallel
    - Remeshing zones in sequential
    - subdomain load balancing
- Main benefits:
  - Scalability of algorithms
  - Re-use of sequential codes

Contents

State of the art
- Parallel remeshers
- Load balancing
- Existing tools

Generic dynamic load balancing

- General purpose (re)partitioning:
  - Jostle
  - Zoltan (K. Devine 2000 [5])
  - LB_Migrate (R. Chaube 2007 [2])
- Require a lot of extra coding

Specialized dynamic load balancing softwares

- Libraries:
  - DRAMA (A. Basermann 2000 [1])
- Pros and cons:
  - Mainly interfaced with solvers
  - Data structures based on meshes

Contents

State of the art
- Parallel remeshers
- Load balancing
- Existing tools

Existing tools for handling unstructured meshes

- Partitioners:
  - Chaco
  - MeTiS
  - Mondriaan
  - Patoh
  - Scotch
  - Zoltan
- Intermediate:
  - DRAMA
  - PaMPA
  - PHG
- Advanced:
  - Arcane

Contents

State of the art

Common needs of solvers regarding meshes

What is PaMPA

Example: Laplacian equation using P_1 finite element method

Some results

Work in progress

Upcoming features

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

State of the art

Common needs of solvers regarding meshes

What is PaMPA

Example: Laplacian equation using P_1 finite element method

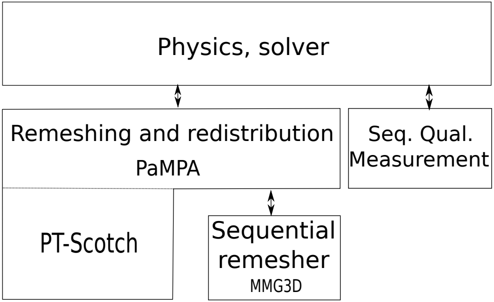
Some results

Work in progress

Upcoming features

What is PaMPA

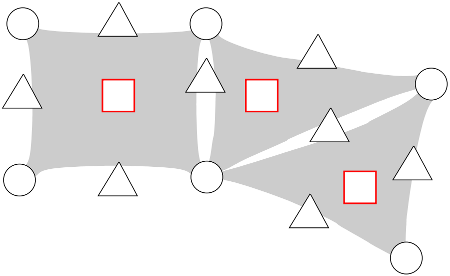
- PaMPA: "Parallel Mesh Partitioning and Adaptation"
- Middleware library managing the parallel repartitioning and remeshing of unstructured meshes modeled as interconnected valuated entities
- The user can focus on his/her "core business":
  - Solver
  - Sequential remesher
    - Coupling with MMG3D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



Definitions

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

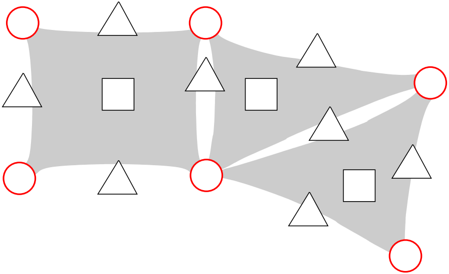
Top-level mesh entity
May bear some data (volume, pressure, etc.)



Definitions

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

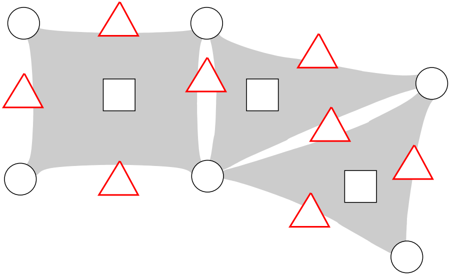
May bear some data (geometry, etc.)



Definitions

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

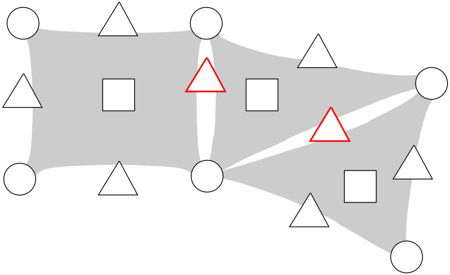
May bear some data (flux, etc.)



Definitions

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

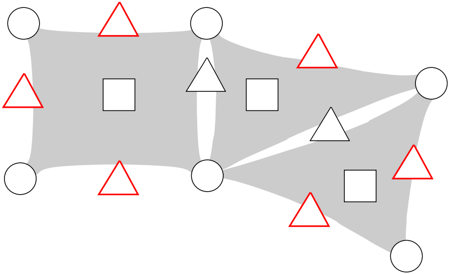
Regular mesh edge



Definitions

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

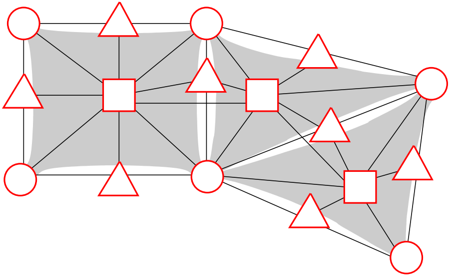
Boundary mesh edge



Definitions

What all entities are in fact...

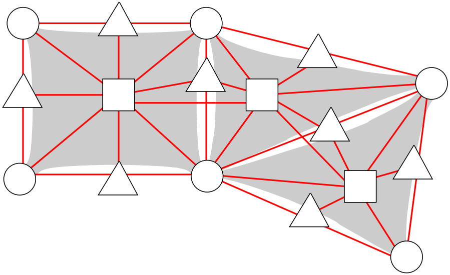
- Mesh:
  - **Element**
  - **Node**
  - **Edge**
    - **Internal**
    - **Boundary**
- PaMPA Mesh:
  - **Vertex**
  - **Relation**
  - **Entity**
  - **Sub-entity**
  - **Enriched graph**



Definitions

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

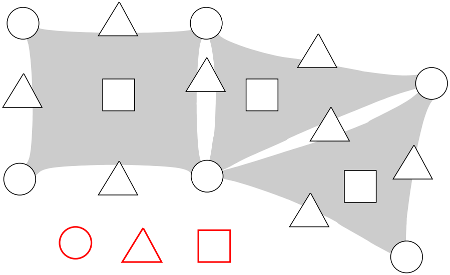
Subset of edges between vertices belonging to prescribed entity types



Definitions

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

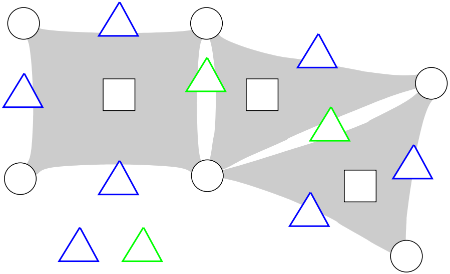
Subset of vertices bearing the same data



Definitions

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

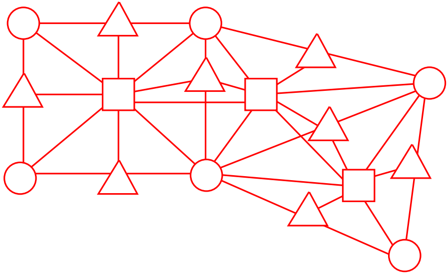
Subset of entity vertices that may bear additional specific data



Definitions

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

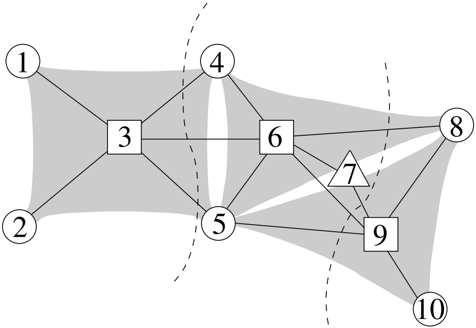
Whole set of vertices and relations
Every vertex belongs to one and only one entity (and sub-entity)



Global vue

- All vertices have a global unique number

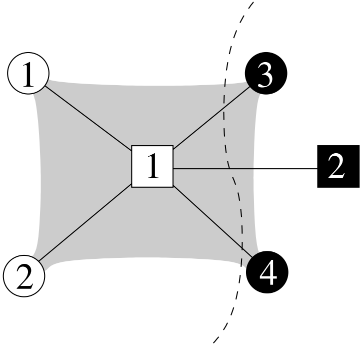
`baseval`	1			
`enttglbnbr`	3			
`proccnttab`	3	4	3	
`procvrttab`	1	4	8	11



Local vision of process 0

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

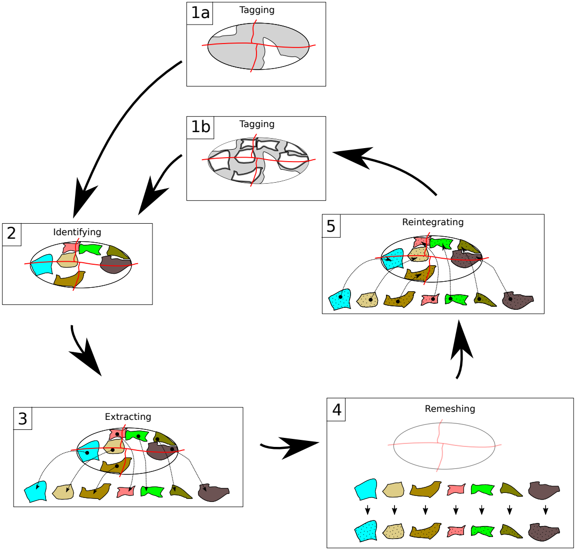
vertlocnbr	3
vertgstnbr	6
edgelocnbr	7
ventloctab	3, 3, 1
vendloctab	
vertloctab	1, 2, 3, 8
edgeloctab	1, 1, 1, 2, 3, 4, 2



Features of version 0.2

- Overlap greater than 1
- Parallel I/O
- Parallel partitioning
- Parallel remeshing based on sequential remesher

Parallel remeshing



Contents

State of the art

Common needs of solvers regarding meshes

What is PaMPA

Example: Laplacian equation using P_1 finite element method

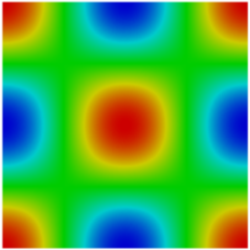
Some results

Work in progress

Upcoming features

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
```

Contents

State of the art

Common needs of solvers regarding meshes

What is PaMPA

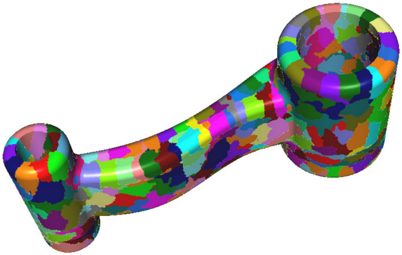
Example: Laplacian equation using P_1 finite element method

Some results

Work in progress

Upcoming features

First results (1/2)



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

First results (2/2)

	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	17:15:12	00:21:14
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%

Contents

State of the art

Common needs of solvers regarding meshes

What is PaMPA

Example: Laplacian equation using P_1 finite element method

Some results

Work in progress

Upcoming features

Work in progress

- Release of version 0.2
  - Available soon from Inria Gforge
  - Licensed under GPL
- Quality of parallel adapted meshes
- Periodic meshes

Contents

State of the art

Common needs of solvers regarding meshes

What is PaMPA

Example: Laplacian equation using P_1 finite element method

Some results

Work in progress

Upcoming features

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
- Multi-grid meshes
- Parallel I/O with HDF5
- Parallel mesh adaptation scalability

Thank you

- A. Basermann, J. Clinckemaillie, T. Coupez, J. Fingberg, H. Digonnet, R. Ducloux, J.-M. Gratien, U. Hartmann, G. Lonsdale, B. Maerten, D. Roose, and C. Walshaw.
  Dynamic load balancing of finite element applications with the drama library, 2000.

- R. Chaube, R. L. Carino, and I. Banicescu.
  Effectiveness of a dynamic load balancing library for scientific applications.
  In *ISPDC '07: Proceedings of the Sixth International Symposium on Parallel and Distributed Computing*, page 32, Washington, DC, USA, 2007. IEEE Computer Society.

- N. Chrisochoides.
  Parallel mesh generation.
  In *in Numerical Solution of Partial Differential Equations on Parallel Computers*. Springer, 2005.

T. Coupez, H. Digonnet, and R. Ducloux.
Parallel meshing and remeshing.
Applied Mathematical Modelling, 25(2):153 – 175, 2000.
Dynamic load balancing of mesh-based applications on parallel.

K. Devine, B. Hendrickson, E. Boman, M. St. John, and C. Vaughan.
Design of dynamic load-balancing tools for parallel applications.
In *ICS '00: Proceedings of the 14th international conference on Supercomputing*, pages 110–118, New York, NY, USA, 2000. ACM.

O. Hassan, K. A. Sørensen, K. Morgan, and N. P. Weatherill.
A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing.
In *International Journal for Numerical Methods in Fluids*, volume 53, pages 1243–1266. John Wiley & Sons, 2007.

B. G. Larwood, N. Weatherhill, O. Hassan, and K. Morgan.
Domain decomposition approach for parallel unstructured mesh generation.
In *International Journal for Numerical Methods in Engineering*, volume 58, pages 177–188, 2003.

U. Tremel, K. A. Sørensen, S. Hitzel, H. Rieger, O. Hassan, and N. P. Weatherill.
Parallel remeshing of unstructured volume grids for cfd applications.
In *International Journal for Numerical Methods in Fluids*, volume 53, pages 1361–1379. John Wiley & Sons, 2007.