PaMPA: Parallel Mesh Partitioning and Adaptation
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
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
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
Common needs of solvers regarding meshes

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.)
What is PaMPA

Definitions

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

May bear some data (geometry, etc.)
What is PaMPA

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
What is PaMPA

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

```
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

1a Tagging
1b Tagging
2 Identifying
3 Extracting
4 Remeshing
5 Reintegrating
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
Example: Laplacian equation using $P_1$ finite element method

Definition

- Solving 2D Poisson equation:
  - $\Delta u(x, y) = f(x, y)$
  - $g(x, y) = u(x, y)$ on the boundary $\Gamma$

- Test case:
  - $f(x, y) = -2 \cos(x) \cos(y)$ in the domain $\Omega$
  - $g(x, y) = \cos(x) \cos(y)$ on the boundary $\Gamma$
  - $u(x, y) = \cos(x) \cos(y)$
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 partitioner
!
3— Redistribute distribute mesh

CALL ElementVolume()
CALL InitializeMatrixCSR()

! Solution computation
!
CALL InitSol()
CALL FillMatrix()
CALL SolveSystem()
CALL WriteDistributedMeshAndSolFiles()
Example: Laplacian equation using $P_1$ finite element method

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 = VolEI (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)
    CALL 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
    RHS(is) = RHS(is) − Volt * SourceTerm (Coor(1, is), Coor(2, is)) / Nsmplx
  END DO
  CALL PAMPAF_itNext (it_vrt)
END DO
Solve system: Jacobi (1/2)

\[ \text{UaPrec} = 0. \]  \hspace{1cm} \text{! Suppose } A = L + D + U, \text{ system to solve: } A x = b \\
\text{CALL PAMPAF_dmeshItInit(dm, ENTITY\_NODE, ENTITY\_NODE, it\_ngb, ierr)} \\
\text{DO irelax = 1, Nrelax} \\
\hspace{1cm} \text{res} = 0. \\
\hspace{1cm} \text{CALL PAMPAF_dmeshItInitStart(dm, ENTITY\_NODE, PAMPAF\_VERT\_BOUNDARY, it\_vrt, ierr)} \\
\hspace{1cm} \text{DO WHILE (PAMPAF\_itHasMore(it\_vrt))} \\
\hspace{2cm} \text{is = PAMPAF\_itCurEnttVertNum(it\_vrt)} \\
\hspace{2cm} \text{CALL PAMPAF\_dmeshMatLineData(dm, ENTITY\_NODE, is, l1, l1Fin, ierr)} \\
\hspace{2cm} \text{CALL PAMPAF\_itStart(it\_ngb, is, ierr)} \\
\hspace{2cm} \text{res0 = RHS(is)} \hspace{1cm} \text{! res0 = b} \\
\hspace{1cm} \text{iv = i1} \\
\hspace{1cm} \text{DO WHILE (PAMPAF\_itHasMore(it\_ngb))} \\
\hspace{2cm} \text{js = PAMPAF\_itCurEnttVertNum(it\_ngb)} \\
\hspace{2cm} \text{PAMPAF\_itNext(it\_ngb)} \\
\hspace{2cm} \text{res0 = res0 - MatCSR\%Vals(iv) * UaPrec(js)} \hspace{1cm} \text{!res0 = b - (L + U) x^n} \\
\hspace{2cm} \text{iv = iv + 1} \\
\hspace{1cm} \text{END DO} \\
\hspace{1cm} \text{Ua(is) = res0 / MatCSR\%Diag(is)} \hspace{1cm} \text{!x^{n+1} = ( b - (L + U) x^n )/D} \\
\hspace{1cm} \text{PAMPAF\_itNext(it\_vrt)} \\
\hspace{1cm} \text{END DO} \\
\text{CALL PAMPAF\_dmeshHaloValueAsync(dm, ENTITY\_NODE, PAMPA\_TAG\_SOL, req, ierr)}
Example: Laplacian equation using $P_1$ finite element method

Solve system: Jacobi (2/2)

```fortran
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, l1, l1Fin, 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

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of elements</td>
<td>2 423 029</td>
</tr>
<tr>
<td>Number of nodes</td>
<td>1 071 626</td>
</tr>
</tbody>
</table>
## First results (2/2)

<table>
<thead>
<tr>
<th></th>
<th>MMG3d on 1 processor</th>
<th>PaMPA-MMG3d on 24 processors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor frequency (GHz)</td>
<td>2.40</td>
<td>3.06</td>
</tr>
<tr>
<td>Used memory (kb)</td>
<td>27 588 940</td>
<td>51 116 044</td>
</tr>
<tr>
<td>Elapsed time</td>
<td>17:15:12</td>
<td>00:21:14</td>
</tr>
<tr>
<td>Number of elements</td>
<td>108 126 515</td>
<td>115 802 876</td>
</tr>
<tr>
<td>Smallest edge length</td>
<td>0.1470</td>
<td>0.1395</td>
</tr>
<tr>
<td>Largest edge length</td>
<td>6.3309</td>
<td>11.2415</td>
</tr>
<tr>
<td>Worst element quality</td>
<td>294.2669</td>
<td>294.2669</td>
</tr>
<tr>
<td>Element quality between 1 and 2</td>
<td>99.65%</td>
<td>99.38%</td>
</tr>
<tr>
<td>Edge length between 0.71 and 1.41</td>
<td>97.25%</td>
<td>97.65%</td>
</tr>
</tbody>
</table>
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


Parallel remeshing of unstructured volume grids for cfd applications.