PaMPA : Parallel Mesh Partitioning and Adaptation
Contents

Common needs of solvers regarding meshes

What is PaMPA

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

Common needs of solvers regarding meshes

What is PaMPA

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
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)

1a: Tagging

1b: Tagging

2: Identifying

3: Extracting

4: Remeshing

5: Reintegrating

Next section
Parallel remeshing (1/7)

1a Tagging
Parallel remeshing (1/7)

Identifying
Parallel remeshing (1/7)
What is PaMPA

Parallel remeshing (1/7)

1a Tagging
2 Identifying
3 Extracting
4 Remeshing
5 Reintegrating

Next section
Parallel remeshing (1/7)
Parallel remeshing (1/7)

1b Tagging
Contents

Common needs of solvers regarding meshes

What is PaMPA

Some results

Work in progress

Upcoming features
Some results (1/4)

Before remeshing

<table>
<thead>
<tr>
<th>Number of elements</th>
<th>2 423 029</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes</td>
<td>1 071 626</td>
</tr>
</tbody>
</table>
### Some results (2/4)

<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>17h15m12s</td>
<td>00h21m14s</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>
Some results (3/4)

<table>
<thead>
<tr>
<th>PaMPA-MMG3D isotropic tetrahedral mesh on 120 processors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed time</td>
</tr>
<tr>
<td>Number of elements</td>
</tr>
<tr>
<td>Smallest edge length</td>
</tr>
<tr>
<td>Largest edge length</td>
</tr>
<tr>
<td>Worst element quality</td>
</tr>
<tr>
<td>Element quality between 1 and 2</td>
</tr>
<tr>
<td>Edge length between 0.71 and 1.41</td>
</tr>
<tr>
<td>00h34m54s</td>
</tr>
<tr>
<td>318 027 812</td>
</tr>
<tr>
<td>0.2862</td>
</tr>
<tr>
<td>6.2161</td>
</tr>
<tr>
<td>235.6651</td>
</tr>
<tr>
<td>99.58%</td>
</tr>
<tr>
<td>97.91%</td>
</tr>
</tbody>
</table>
PaMPA-MMG3D anisotropic tetrahedral mesh on 6 processors

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed time</td>
<td>00h08m07s</td>
</tr>
<tr>
<td>Number of elements</td>
<td>11,687,798</td>
</tr>
<tr>
<td>Smallest edge length</td>
<td>0.1265</td>
</tr>
<tr>
<td>Largest edge length</td>
<td>11.0146</td>
</tr>
<tr>
<td>Worst element quality</td>
<td>45.1691</td>
</tr>
<tr>
<td>Element quality between 1 and 2</td>
<td>98.21%</td>
</tr>
<tr>
<td>Edge length between 0.71 and 1.41</td>
<td>93.15%</td>
</tr>
</tbody>
</table>
Common needs of solvers regarding meshes

What is PaMPA

Some results

Work in progress

Upcoming features
Work in progress

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

Common needs of solvers regarding meshes

What is PaMPA

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
- 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

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

![Diagram of local vertices and indices](diagram.png)
Renumbering
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

\[
\text{RHS} = 0.
\]

CALL PAMPAF_dmeshIfInitStart (dm, ENTITY_ELEM, PAMPAF_VERT_ANY, it_vrt, ierr)
CALL PAMPAF_dmeshIfInit (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)
    NuElem (ngb) = is
    CoorElem (is, ngb) = Coor (:, is)
    CALL PAMPAF_itNext (it_ngb)
  END DO
  CALL GradPhi (CoorElem (:, 1), CoorElem (:, 2), CoorElem (:, 3), GrdPhi)
  DO i = 1, Nsmplx
    is = NuElem (i)
    DO j = 1, Nsmplx
      js = NuElem (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
  CALL PAMPAF_itNext (it_vrt)
END DO
Example: Laplacian equation using $P_1$ finite element method

Solve system: Jacobi (1/2)

$$Ua_{\text{Prec}} = 0.$$  \text{Suppose } A = L + D + U, \text{ system to solve: } A \mathbf{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, l1, l1Fin, ierr)
    CALL PAMPAF_itStart(it\_ngb, is, ierr)
    res0 = RHS(is)  \text{! res0 = } b
    iv = i1
    DO WHILE (PAMPAF_itHasMore(it\_ngb))
      js = PAMPAF_itCurEnttVertNum(it\_ngb)
      PAMPAF_itNext(it\_ngb)
      res0 = res0 - MatCSR%Vals(iv) * Ua\_{Prec}(js)  \text{! res0 = } b - (L + U) \mathbf{x}^n
      iv = iv + 1
    END DO
  Ua(is) = res0 / MatCSR%Diag(is)  \text{! } \mathbf{x}^{n+1} = (b - (L + U) \mathbf{x}^n)/D
  CALL PAMPAF_dmeshHaloValueAsync(dm, ENTITY NODE, PAMPAF\_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
```