General Grid Interface

Theoretical Basis and Implementation

Hrvoje Jasak

Wikki Ltd, United Kingdom

`h.jasak@wikki.co.uk`

General Grid Interface

Objective

- Review implementation of the General Grid Interface (GGI) in OpenFOAM and its use in turbomachinery applications

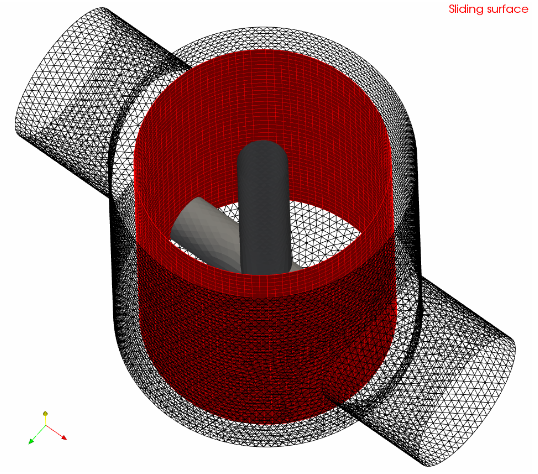
Topics

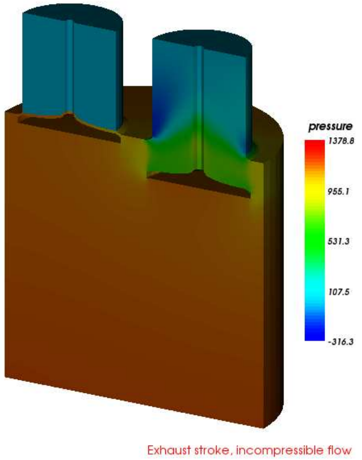
- Background: sliding mesh components
- General Grid Interface: design rationale
- Numerical considerations: discretising GGI interface
- GGI interpolation and weight calculation
- Derived forms: cyclic GGI and partial overlap GGI
- Code components
- Parallelisation of GGI interfaces
- Preparing a mesh for GGI
- Example of use
- Summary

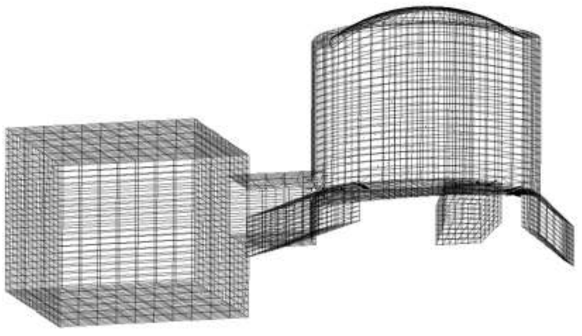
Background

Handling Sliding Mesh Interfaces

- Turbomachinery applications typically involve components in relative motion: need to handle a set of separate regions as one contiguous mesh
- Components move relative to each other, but at each time instance create a single contiguous region: “attaching and detaching” the mesh during simulation
- This is a subset of **topological mesh changes**, already implemented in OpenFOAM: do we need anything further?
- Unfortunately, topological changes do not satisfy all our needs



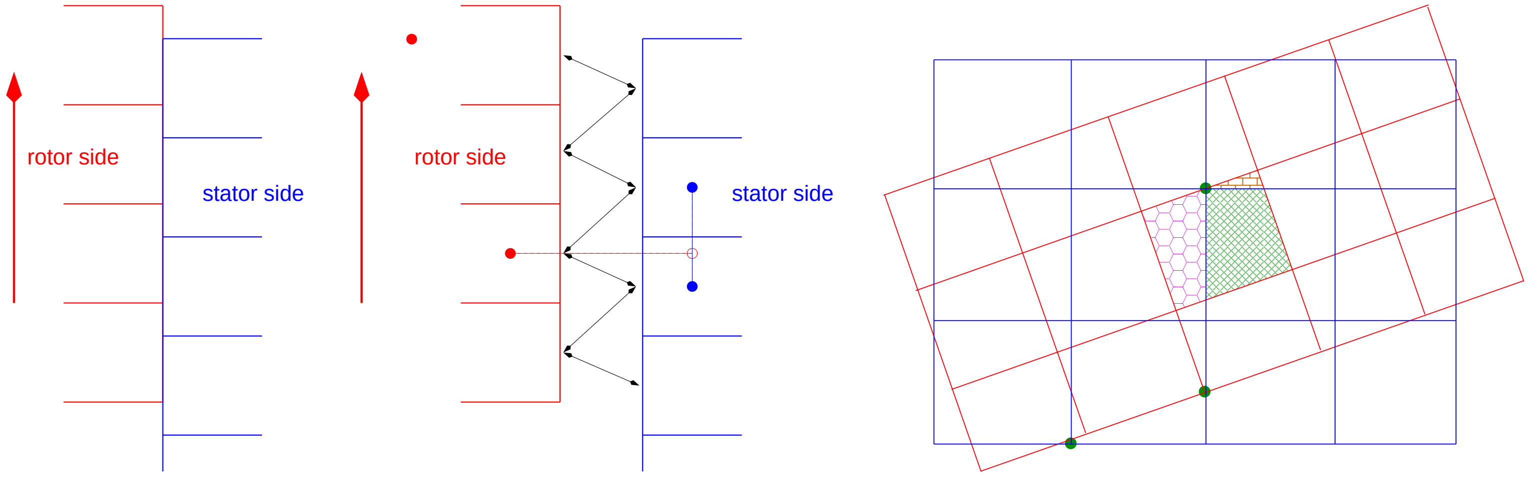




Sliding Interface

Topological Mesh Changes: Sliding Interface

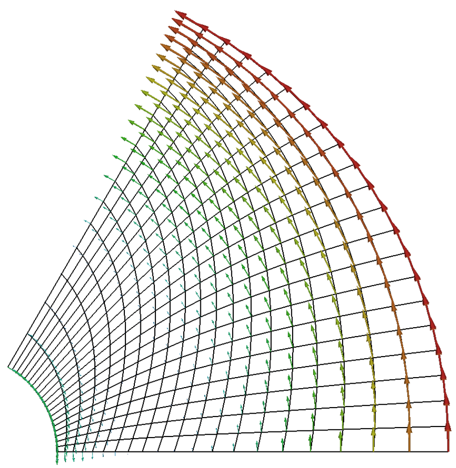
- Sliding interface topology modifier
  - Defined by a master and slave surfaces
  - As surfaces move relative to each other, perform mesh cutting operations and replace original faces with facets
  - Re-assemble mesh connectivity on all cells and faces touching the sliding surface: fully connected 3-D mesh
- Polyhedral mesh support in OpenFOAM facilitates topological changes
- Once the mesh is complete, there is **no further impact in the code!**
- Connectivity across interface changes with relative motion

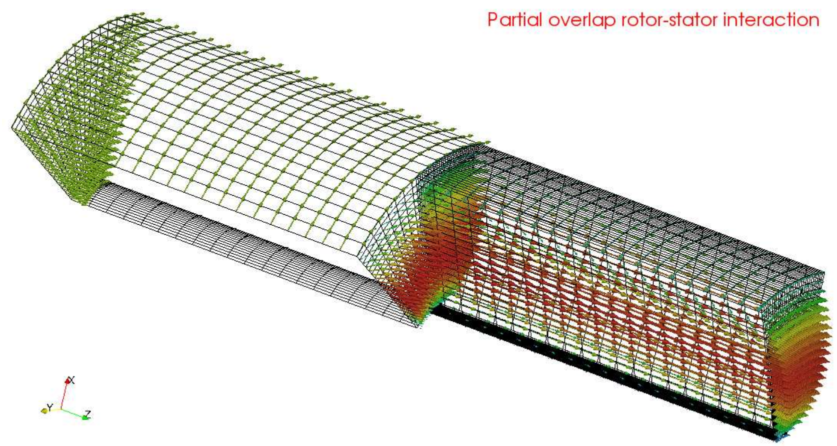


GGI Design Rationale

GGI Interface in Turbomachinery

- Apart from “fully overlapped” cases, turbomachinery meshes contain similar features that should employ identical methodology, but are not quite the same
  - **Non-matching cyclics** for a single rotor passage
  - **Partial overlap** for different rotor-stator pitch
  - **Mixing plane**: perform averaging instead of coupling directly
- Component coupling requires data manipulation (copy, transform, average)
- In such cases, the behaviour is closer to a **coupled boundary condition**, but the numerics is similar to sliding interface
- **Objective: mimic behaviour of sliding interface without changing the mesh**





GGI Discretisation

FVM Discretisation on a GGI Interface

- Review discretisation of convection and diffusion when faces are replaced; volumetric integral terms are not affected
- **Convection operator** splits into a sum of face flux integrals

$$\int_V \nabla \bullet (\phi \mathbf{u})\, dV = \oint_S \phi(\mathbf{n} \bullet \mathbf{u}) dS = \sum_f \phi_f (\mathbf{s}_f \bullet \mathbf{u}_f) = \sum_f \phi_f F$$

  where ϕ_f is the face value of ϕ and $F = \mathbf{s}_f \bullet \mathbf{u}_f$ is the **face flux**

- **Diffusion operator** captures the gradient transport

$$\oint_S \gamma(\mathbf{n} \bullet \nabla \phi) dS = \sum_f \int_{S_f} \gamma(\mathbf{n} \bullet \nabla \phi)\, dS = \sum_f \gamma_f\, \mathbf{s}_f \bullet (\nabla \phi)_f$$

- Face terms: interpolated value and face gradient

$$\phi_f = f_x \phi_P + (1 - f_x) \phi_N, \quad \mathbf{s}_f \bullet (\nabla \phi)_f = |\mathbf{s}_f| \frac{\phi_N - \phi_P}{|\mathbf{d}_f|}$$

GGI Discretisation

FVM Discretisation on a GGI Interface

- When cutting is performed, total face area is replaced by facets
- Discretisation on the interface can be rewritten as a sum of facet operations. Inverting the loop, we can is introduce **shadow neighbour values** ϕ_N^s values for the in front of the face, creating the effect as if the interface is integrally matched

$$\phi_N^s = \sum_t w_t \phi_t$$

  where t denotes a selection of cell/face values on the “other side”
- Consistency conditions: simple averaging is not flux-conservative
  - Area of original face must be equal to sum of facet areas replacing it

$$\sum_t w_t = 1 \quad \text{for all faces on both sides}$$

  - If face A touches face B, perceived facet area must be the same

$$w_{A \rightarrow B} S_A = w_{B \rightarrow A} S_B$$

GGI Interpolation

GGI Interpolation

- Role of GGI interpolation is to calculate shadow interpolation weights
- Idea 1: form a matrix equation for weights and solve: does not work (HJ)
- **Idea 2**: use geometrical cutting as in sliding interface and calculate weights as per original definition. Also provides the addressing

$$w_t = \frac{S_{facet}}{S}$$

GGI Intersection Algorithm

- Developed and implemented by Martin Beaudoin, Hydro Quebec
- Components
  1. Quick reject in 3-D: **Axis-Aligned Bounding Box**
  2. Projection into common plane
  3. Quick reject in 2-D: **Separating Axis Theorem**
  4. Point in polygon detection: **Horman-Agathos algorithm**
  5. Polygon intersection: **Sutherland-Hodgman clipping algorithm**
- Result: facet area, GGI addressing and weights

Derived Forms of GGI

Extending Basic GGI Algorithm

- GGI operates as a coupled patch field condition: interpolate shadow and update
- **Cyclic GGI**
  - Create transformed surface of the shadow patch and calculate weights
  - Transform scalar/vector/tensor data according to rank
  - Use GGI interpolation on transformed shadow data and update as usual
- **Partial Overlap GGI**
  - Create transformed surface of the shadow patch by copying the geometry multiple times to achieve full overlap and calculate weights
  - Transform scalar/vector/tensor data according to rank and expand over number of copies
  - Use GGI interpolation on transformed shadow data and update as usual
- GGI interpolation is useful beyond GGI: provides flux conservative and function-monotonic interpolation
  - Conjugate heat transfer with non-matching solid-fluid boundaries
  - Fluid-structure interaction: force-conservative interpolation

GGI Code Components

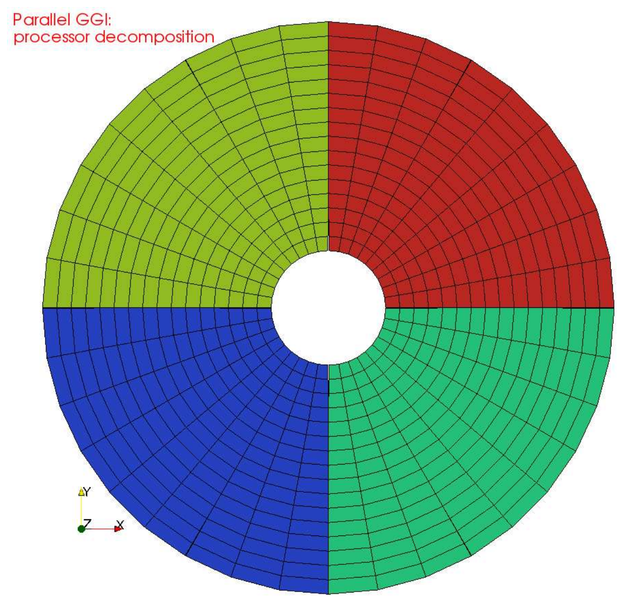
Implementation of GGI in OpenFOAM

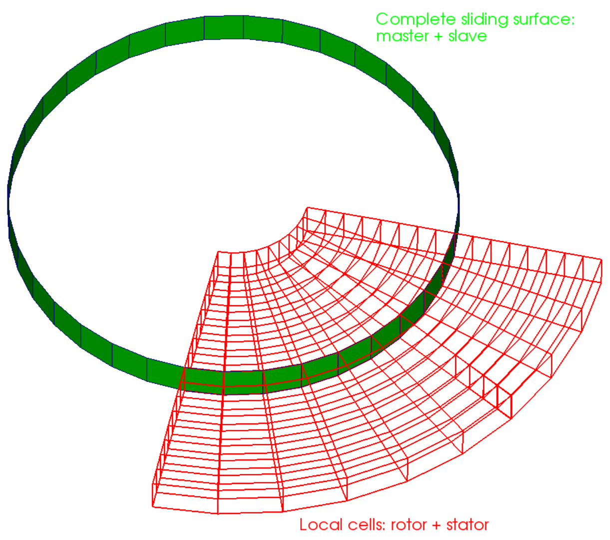
- **Interpolation and geometry**
  - Basic algorithms: Horman-Agathos, Sutherland-Hodgman
  - Templated GGI interpolation, abstracting patch type
  - Instantiated interpolation for stand-alone patch and `polyPatch`
- **GGI patch and discretisation** (identical for cyclic GGI and partial overlap)
  - Mesh patch with interpolation: `ggiPolyPatch`, `ggiPointPatch`
  - Matrix support: `ggiLduInterface`, `ggiLduInterfaceField`
  - Coupled FV patch with discretisation support `ggiFvPatch` and `ggiFvPatchField`: constrained patch
  - Special support for AMG coarsening, to be done consistently on all levels: `processorGAMGInterface` and `processorGAMGInterfaceField`

Parallelisation of GGI

Parallelisation of GGI

- In parallel, sliding GGI changes processor-to-processor connectivity
- Trouble in weights calculation and in scheduling of processor-to-processor communications: dangerous or inefficient
- Solution: **Global sync of GGI data**
  - Complete sliding surface must be present on all CPUs: decomposition
  - In each evaluation, gather-scatter of shadow data for complete interface
  - Evaluate GGI as usual: local patch only addresses a part of sliding surface





Prepare for GGI

Definition of a GGI Patch and Field

- Build the mesh in the usual way, with disconnected components
- Prepare face zone for master and slave surface

```
wooster*685-> setSet
faceSet insideZone new patchToFace insideSlider
faceSet outsideZone new patchToFace outsideSlider
quit

wooster*685-> setsToZones -noFlipMap
```

- Boundary file definition: `constant/polyMesh/boundary`

```
insideSlider
{
    type            ggi;
    nFaces          36;
    startFace       1192;
    shadowPatch     outsideSlider;
    zone            insideZone;
    bridgeOverlap false;
}
```

```
outsideSlider
{
    type            ggi;
    nFaces          36;
    startFace       1228;
    shadowPatch     insideSlider;
    zone            outsideZone;
    bridgeOverlap false;
}
```

Prepare for GGI

Field Definition

- GGI is a constrained condition: forces patch field type

```
boundaryField
{
    insideSlider
    {
        type            ggi;
    }
    ...
}
```

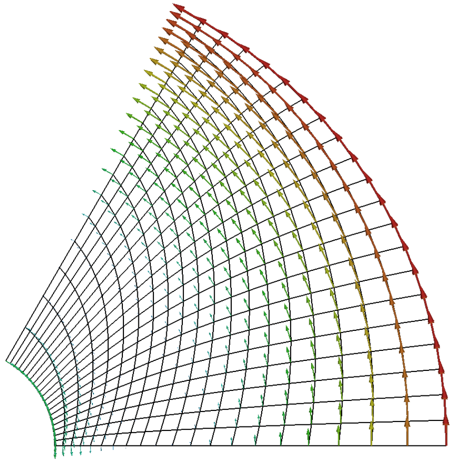
- Parallel decomposition: GGI patch surface must be present on all CPUs in its entirety
- Decomposition dictionary: new entry for global face zones

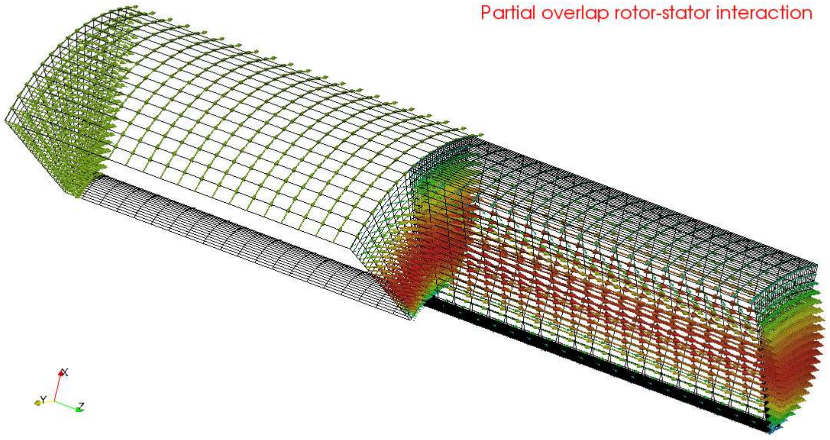
```
globalFaceZones ( insideZone outsideZone );
```

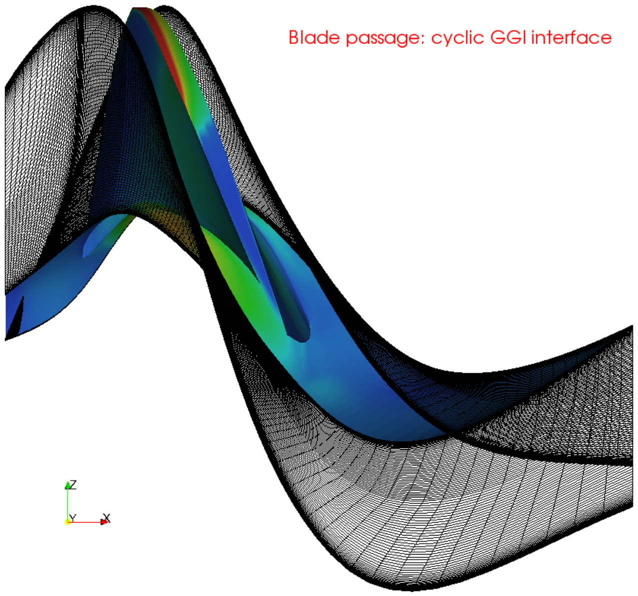
- Some care is required in choice of linear equation solvers

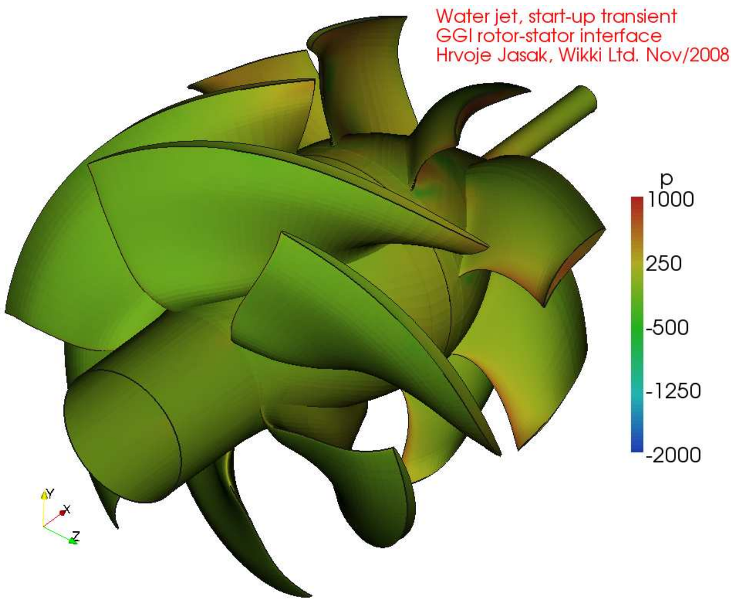
Simple Examples

Simple Examples of GGI Interfaces in Use





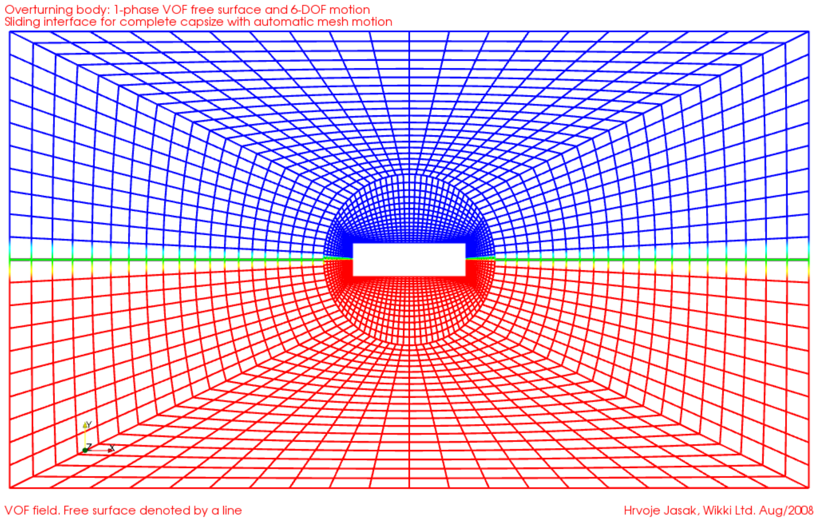


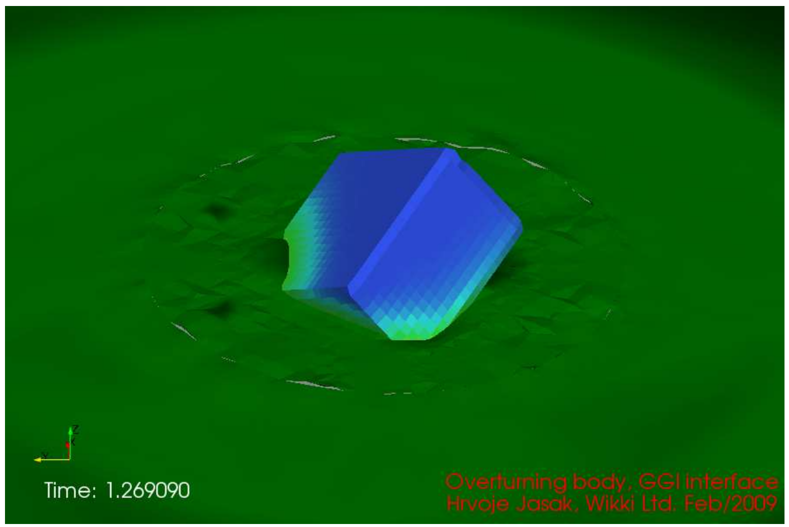


Simple Examples

Capsizing Body with Topological Changes or GGI

- Full capsize of a floating body cannot be handled without topology change
- Mesh motion is decomposed into translational and rotational component
  - External mesh performs only translational motion
  - Rotation on capsize accommodated by a GGI interface
- Automatic motion solver handles the decomposition, based on 6-DOF solution
- Mesh inside of the sphere is preserved: boundary layer resolution
- Precise handling of GGI interface is essential: boundedness and mass conservation for the VOF variable must be preserved





Summary

Summary

- GGI interface allows coupling of mesh components without the need for topological mesh changes
- GGI discretisation is identical to sliding interface with mesh cutting
- Interpolation weights calculated using polygon clipping
- Derived forms: cyclic GGI and partial overlap re-use interpolation code
- **Parallelisation**: complete GGI surface present on all CPUs. Added option of preserving faces in decomposition without attached cells
- Recent updates for communication scheduling and improved parallel scaling
- The code is complete and ready for use