SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

ESPRESO - Highly parallel finite element package for engineering simulations

Tomas Kozubek, IT4Innovations, VSB – Technical University of Ostrava

June/2021

Univerza v Ljubljani

TECHNISCHE UNIVERSITÄT WIEN

CINECA consorzio interuniversitario

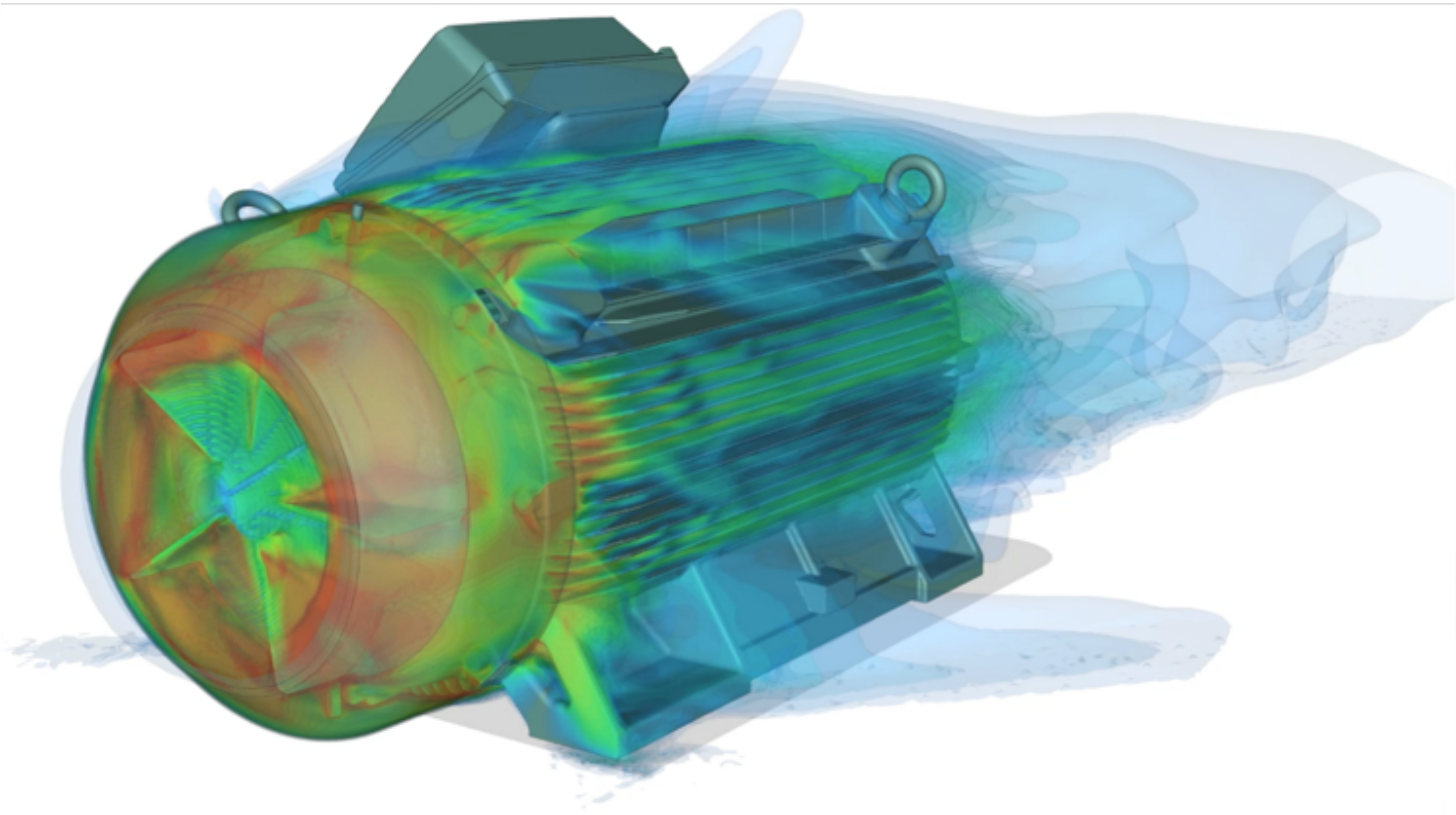
VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Co-funded by the Erasmus+ Programme of the European Union

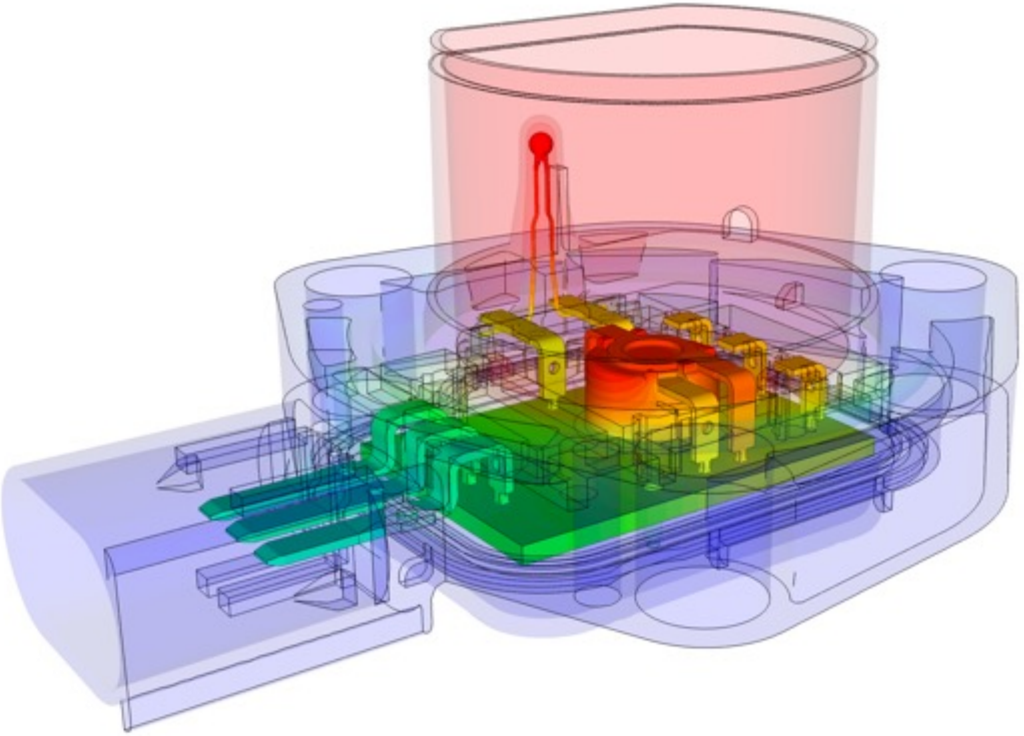
This project has been funded with support from the European Commission.
This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Motivation

T. Kozubek, IT4Innovations, VSB-TUO



SIEMENS



Continental

Engineering group portfolio

T. Kozubek, IT4Innovations, VSB-TUO

In-house Code - Third Party Open-Source - Commercial solution

Customisation

we create solution templates tailored to the specific problems

Code Development

Functionality expansion according to customer needs

Problem Solving

we offer complex problem solutions with efficient hardware utilization

ESPRESO library

T. Kozubek, IT4Innovations, VSB-TUO

ESPRESO

Key Features

Scalable I/O

Connection to popular commercial and open source tools

Massively parallel solvers

Scalable solvers for today's most powerful supercomputers

High-end application execution

Easy and intuitive access to the supercomputing infrastructure

Mesh processing and morphing

Change the meshed geometry without its remeshing.

Finite element library

Complex library for nonlinear multiphysical simulations

Simple configuration interface

Fast and modular approach for templating your computations

ESPRESO Implementation Details

T. Kozubek, IT4Innovations, VSB-TUO

Implementation in C++

- start – 2014 as a HTFETI solver
- CPU/GPU/MIC version
- int32 (default) and int64 bit version
  - for problems larger than ~2 billion DOF
- Intel compiler preferred
- GCC also supported

Parallelization tools and strategies

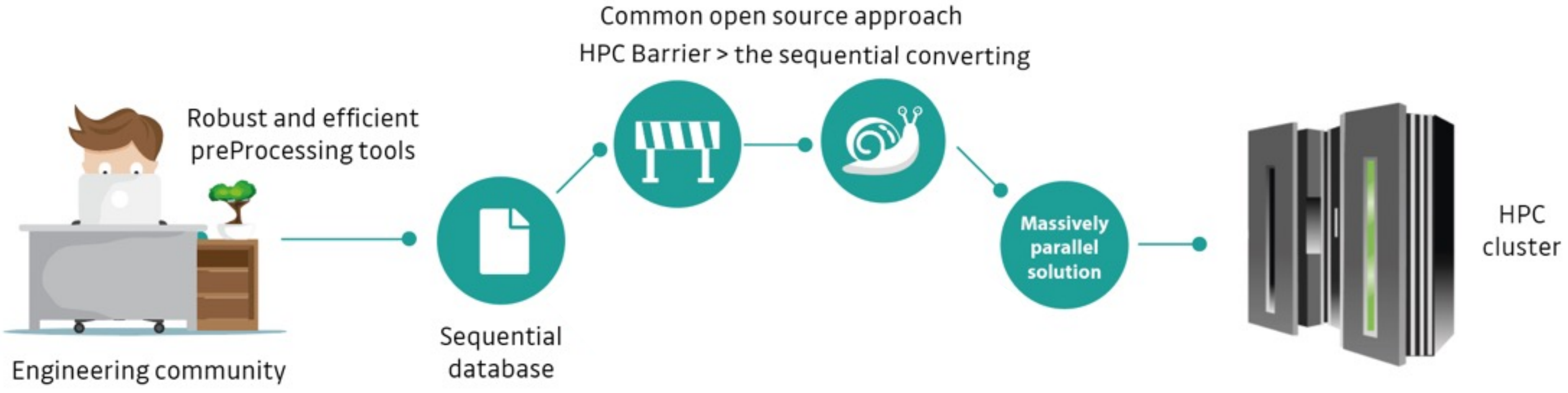
- hybrid parallelization for multi-socket, multicore compute nodes
- distributed memory parallelization – MPI
- shared memory parallelization – using OpenMP
- vectorization using Intel MKL and compiler

Dependencies

- Math library - Intel MKL – required for sparse BLAS
- EXPRTK – math expression toolkit
- ASYNC – library for asynchronous output
- Graph Decomposition library
  - METIS/ParMETIS
  - optional
    - SCOTCH/PTSCOTCH
    - KaHIP
- Optional
  - PARDISO sparse direct solver (both the original version and the MKL versions are supported)
  - Super LU
  - WATSON
  - HYPRE Solvers

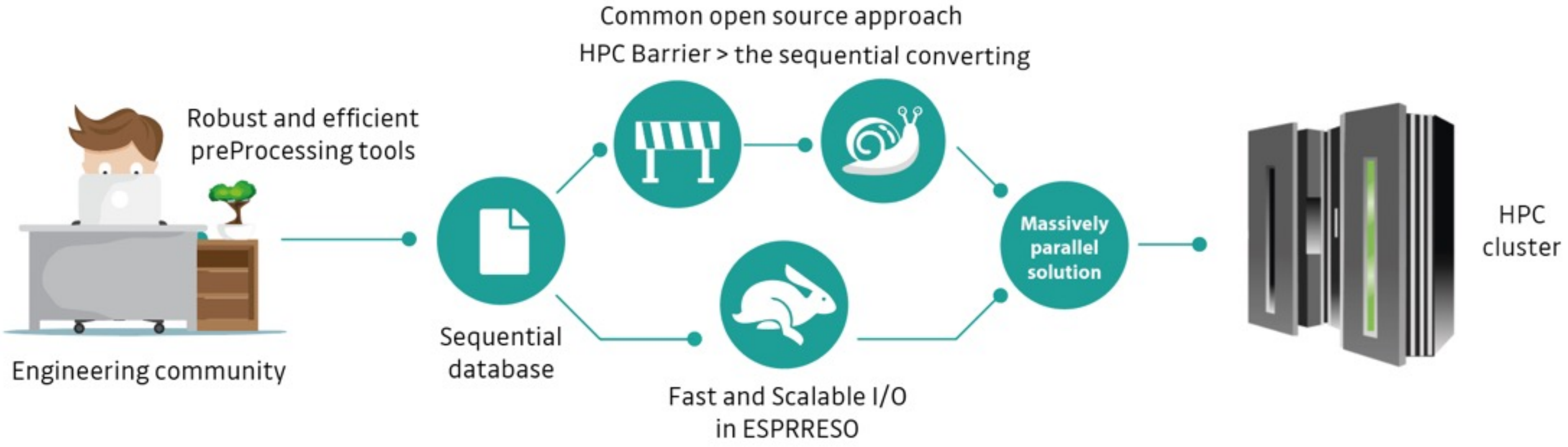
Scalable I/O

Scalable I/O


Common open source approach
HPC Barrier > the sequential converting
Robust and efficient
preProcessing tools
Engineering community
Sequential
database
Massively
parallel
solution
HPC
cluster

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

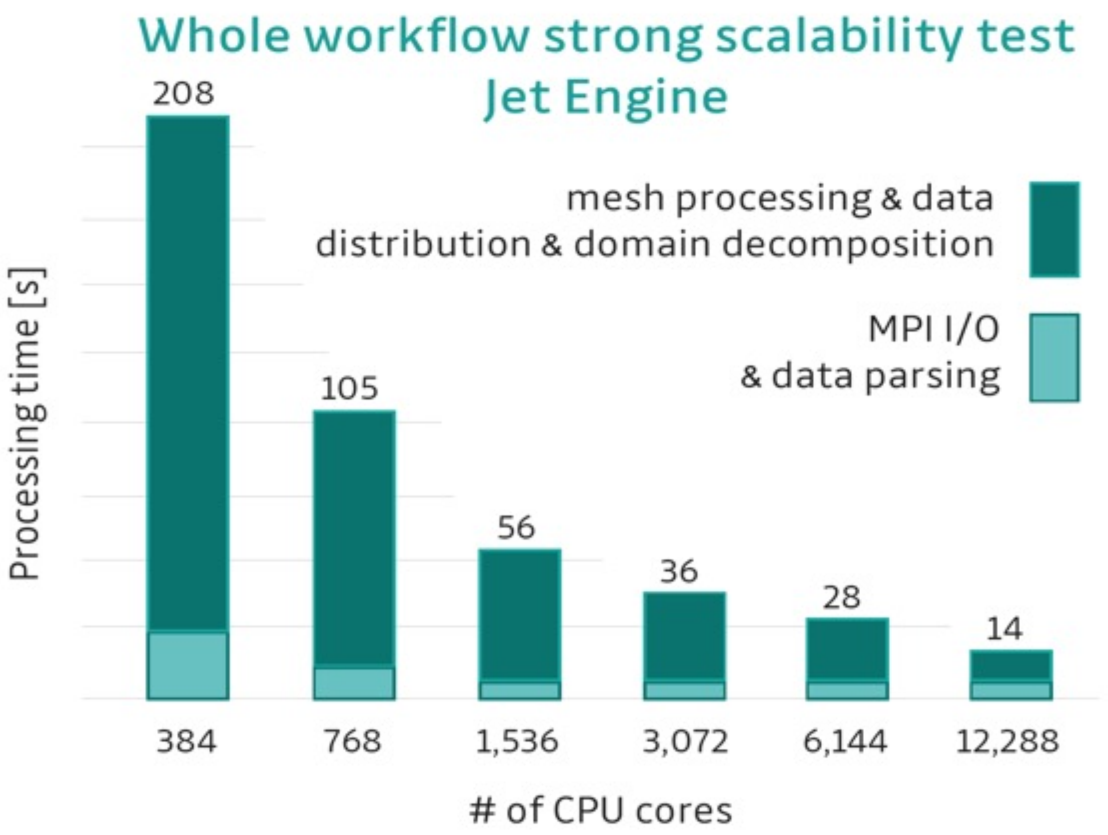
Scalable I/O



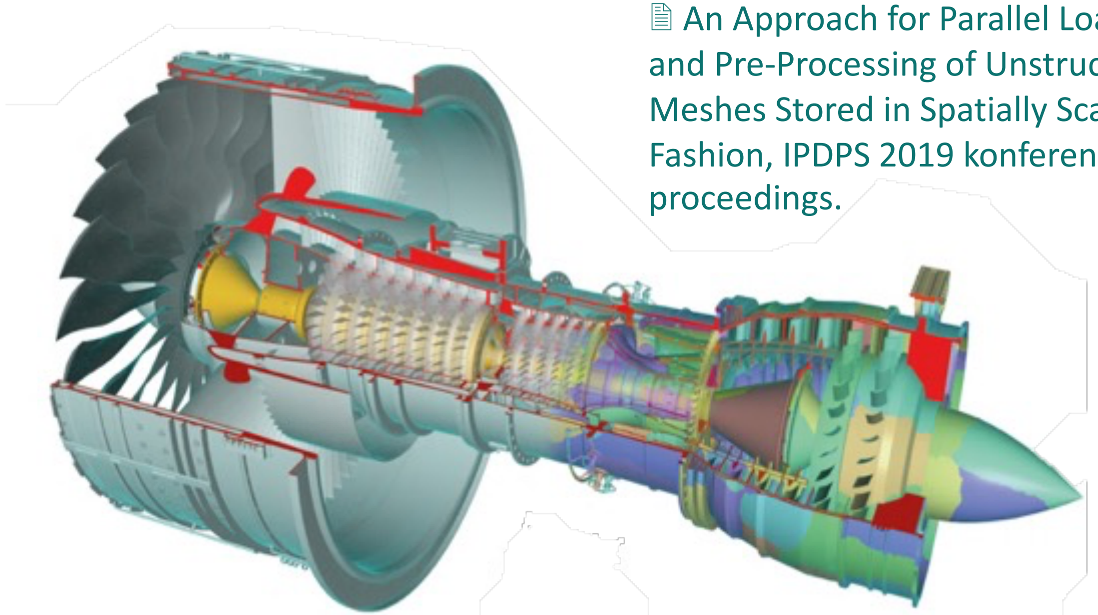
Scalable I/O

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



1. Direct input to the parallel solvers without data conversion
2. Support for EnSight, ANSYS CDB, OpenFOAM, ABAQUS formats
3. Solution checkpoint and restart with different amount of resources
4. Geometric decomposition based on Hilbert space-filling curve
5. Decomposers: PT-Scotch/Scotch, ParMETIS/Metis, KaHIP



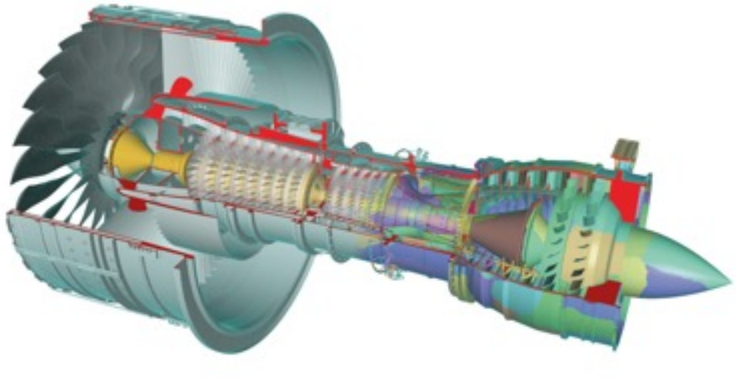
An Approach for Parallel Loading and Pre-Processing of Unstructured Meshes Stored in Spatially Scattered Fashion, IPDPS 2019 konference proceedings.

Scalability of I/O module for jet engine benchmark created in ANSYS Workbench and stored to the sequential text ANSYS solver input database file. The file has size of 150 GB and contains 822 million mesh nodes. The whole workflow consists of the MPI I/O, data parsing, mesh processing, data distribution, and two level domain decomposition and enables data preparation for the FETI solver in 14 seconds using 12,288 CPU cores.

Scalable I/O

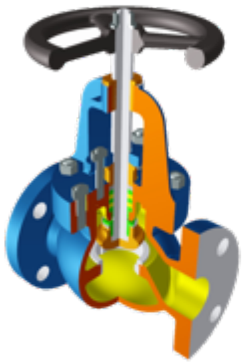
T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



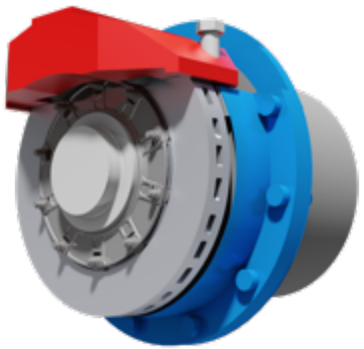
Jet Engine
>7200 s --> 13.7 s

	Valve	Manifold	Disc Brake	Loader Arm	Jet Engine
file size (GB)	6.3	21	19	28	142
nodes (millions)	31	101	122	169	822
elements (millions)	23	73	45	85	484
elements regions	6	1	10	62	26
boundary regions	2	2	31	0	0
nodes regions	0	5	1	4	3



Valve
433 s --> 2.22 s

Manifold
1359 s --> 3.77 s



Disc Brake
938 s --> 4.21 s

Loader Arm
1270 s --> 4.87 s

Asynchronous Output

T. Kozubek, IT4Innovations, VSB-TUO

Asynchronous Output

For the PostProcessing purpose, storing result outputs to commonly used PostProcessing formats are implemented.

Output to commonly used postProcessing formats EnSight and VTK

Overlapping ongoing computation by storing solution results

Result monitoring of selected regions for statistical and optimization toolchains

Solution restarting without ties to the previous number of resources

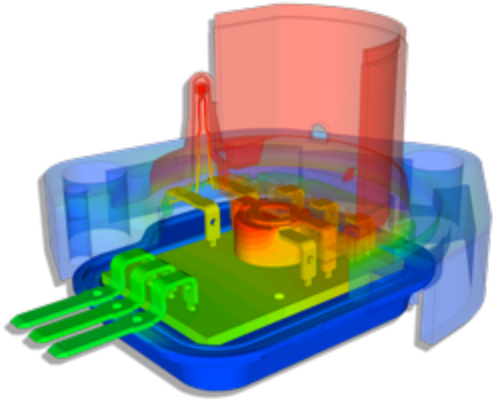
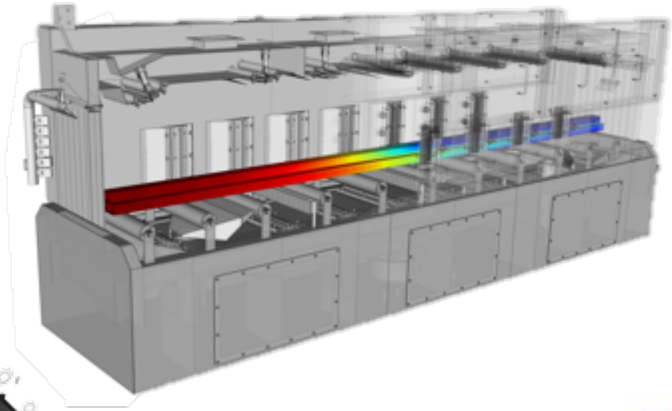
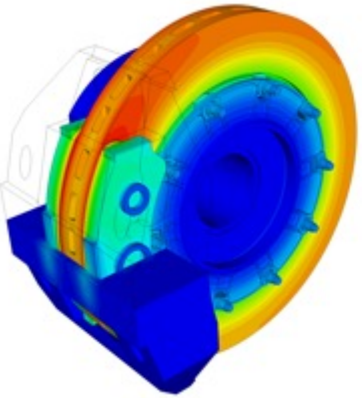
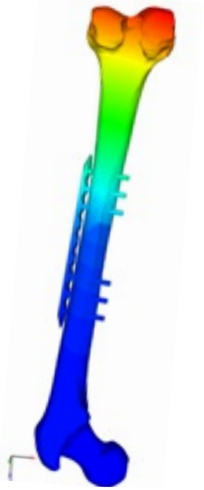
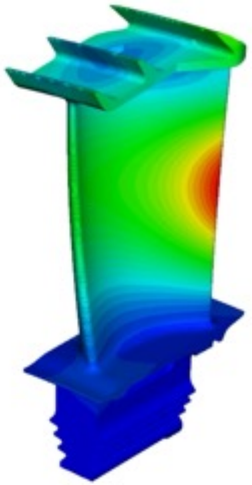
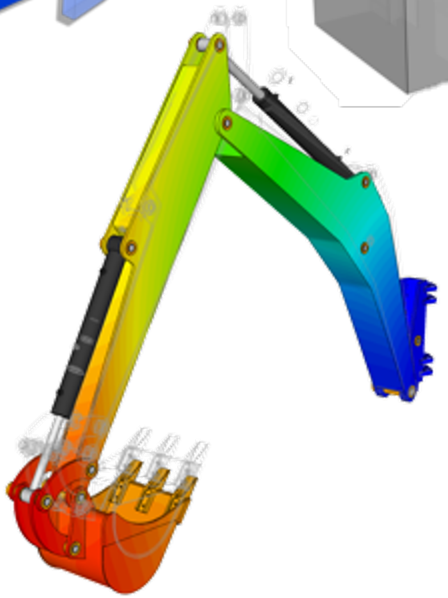
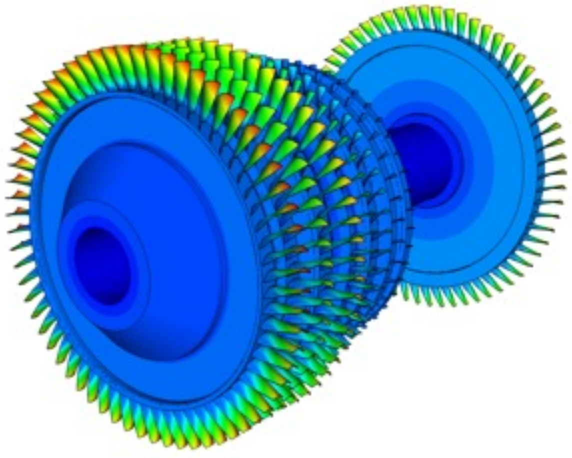
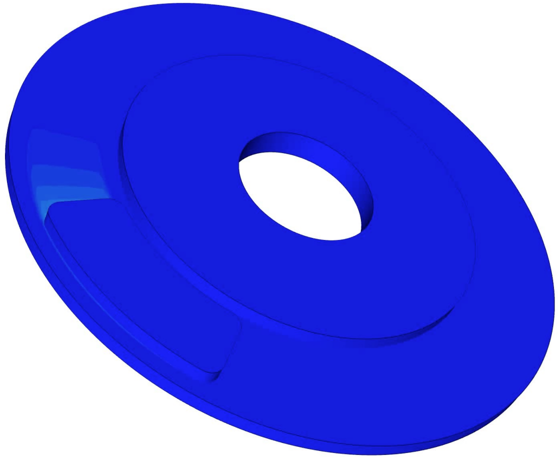
Finite Element Library

Finite Element Library

T. Kozubek, IT4Innovations, VSB-TUO

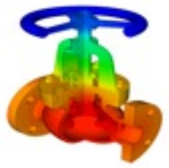
SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

Heat transfer | Phase Change | Structural mechanics | Harmonic problems | Transient simulations | linear | nonlinear

Heat Transfer Module Capability List:

Load steps definition for combination of multiple steady-state and time dependent analyses



Transient solvers
- Generalized trapezoidal rule
- Automatic time stepping based on response frequency approach

Nonlinear solvers
- Newton Raphson – full and symmetric
- Newton Raphson with constant tangent matrices
- Line search damping
- Sub-steps definition
- Adaptive precision control for iterative solvers

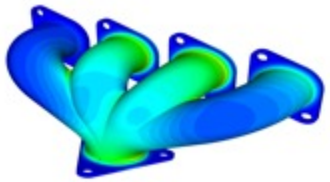
Linear and quadratic finite element discretization

Gluing nonmatching grids by mortar discretization techniques

Full-fledged material models
- nonlinear materials
- isotropic, orthotropic and anisotropic material models
- materials for phase change

Element coordinate system definition – cartesian, polar and spherical

Temperature and time dependent boundary conditions
- linear convection
- nonlinear convection
- heat flow
- heat flux
- diffuse radiation
- heat source
- translation motion



Consistent SUPG and CAU stabilization for Translation Motion (advection), Inconsistent stabilization

Phase Change based on apparent heat capacity method

Boundary element discretization for selected physical applications

Highly parallel multilevel FETI domain decomposition based solver for billions of unknowns for symmetric and non-symmetric systems with accelerators support and combination of MPI and OpenMP techniques

Asynchronous parallel I/O

Input mesh format from popular open source and commercial packages like OpenFOAM, ELMER or ANSYS

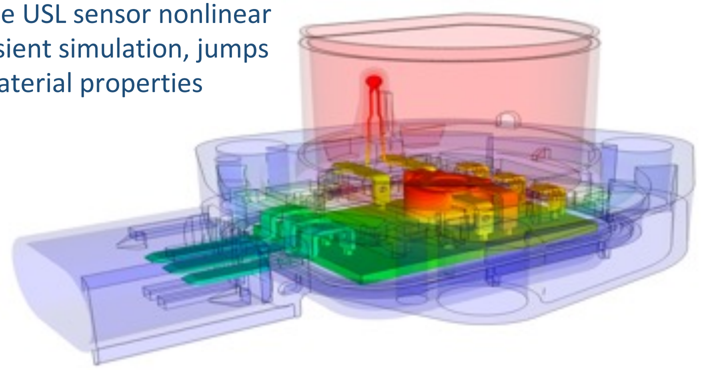
Output to commonly used post-processing formats, VTK and EnSight

Monitoring results on selected regions for statistic and optimization toolchain

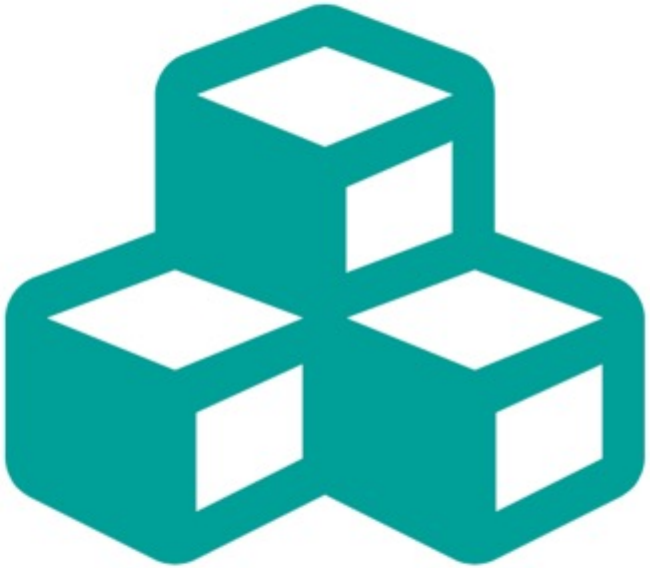
Simple text Espreso Configuration File (ecf) for setting all ESPRESO FEM solver parameters without GUI. Control each parameter in ecf file from command line

Continental

Response time optimization of the USL sensor nonlinear transient simulation, jumps in material properties



T. Kozubek, IT4Innovations, VSB-TUO

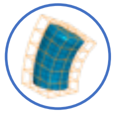


Mesh Morphing

Mesh morphing

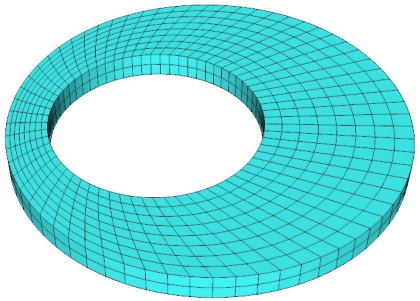
T. Kozubek, IT4Innovations, VSB-TUO

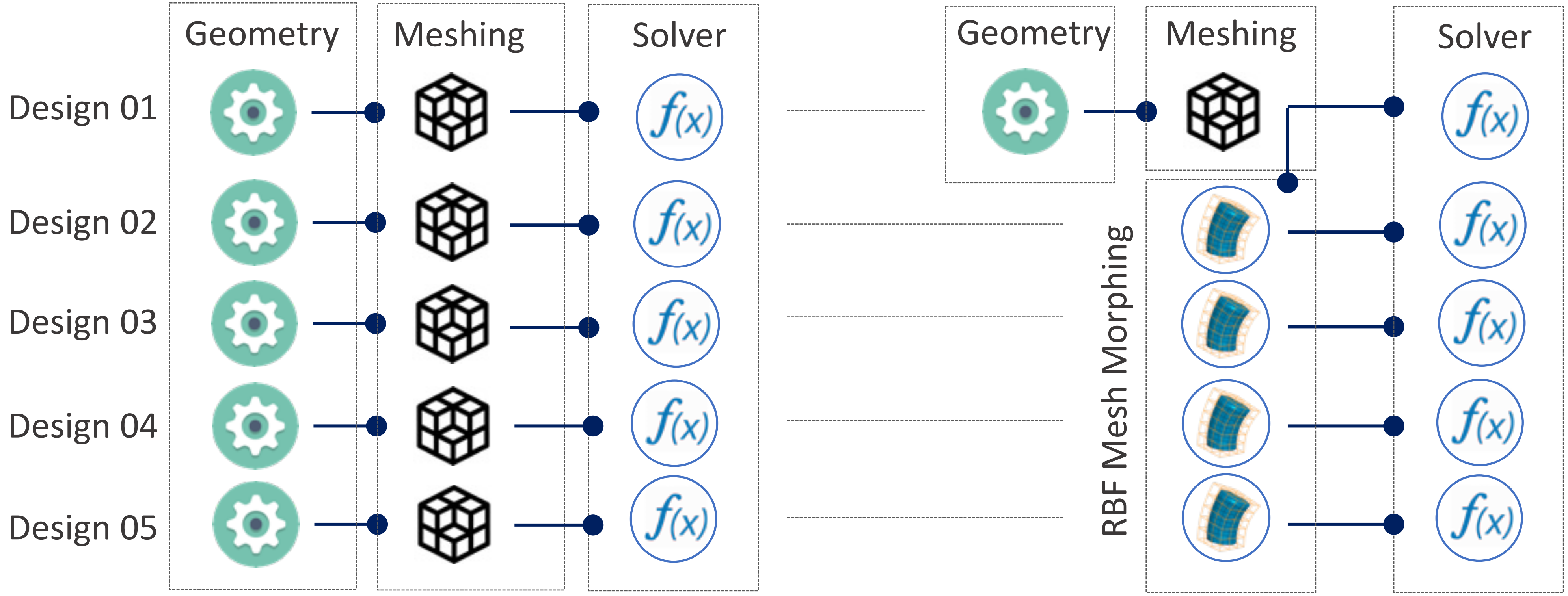
Radial Basis Functions Mesh Morphing



fast and mesh independent solution for

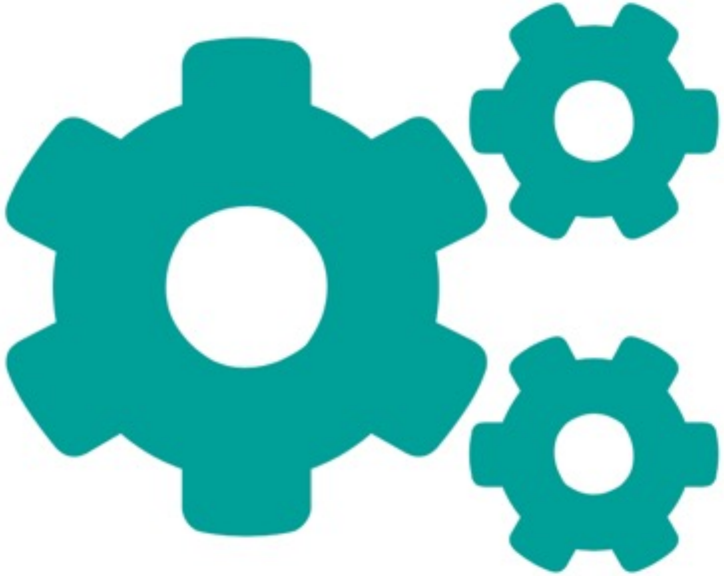
- design Developments
- multi-configuration studies
- sensitivity Studies
- DOE (Design Of Experiment)
- optimization





5x meshing, 5x file transfer

1x meshing, 1x file transfer



Simple Configuration Interface

Simple Configuration Interface

T. Kozubek, IT4Innovations, VSB-TUO

Simple configuration interface

Maths Expresions

Input Arguments

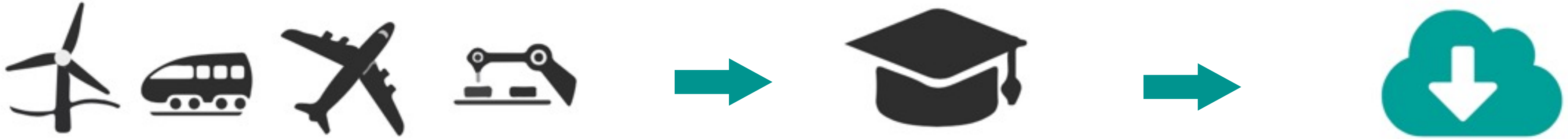
Custom Templates

```
#[EXTERNAL_FILE,GENERATOR]
INPUT_TYPE   EXTERNAL_FILE;

INPUT {
  PATH                         .;
  #[ANSYS_CDB,OPENFOAM,ABAQUS]
  FORMAT             ANSYS_CDB;
  KEEP_MATERIAL_SETS     FALSE;
  CONVERT_DATABASE       FALSE;
  SCALE_FACTOR               1;

  #[NODES,PROCESSES]
  GRANULARITY        PROCESSES;
  REDUCTION_RATIO            1;

  DECOMPOSITION {
    #[PARMETIS,PTSCOTCH]
    PARALLEL_DECOMPOSER   PARMETIS;
    #[METIS,SCOTCH,KAHIP]
    SEQUENTIAL_DECOMPOSER    METIS;
    MESH_DUPLICATION             1;
    DOMAINS                      0;

    #[NODES,PROCESSES]
    GRANULARITY          PROCESSES;
    REDUCTION_RATIO              1;
```

High-end application execution

Solver as a Service

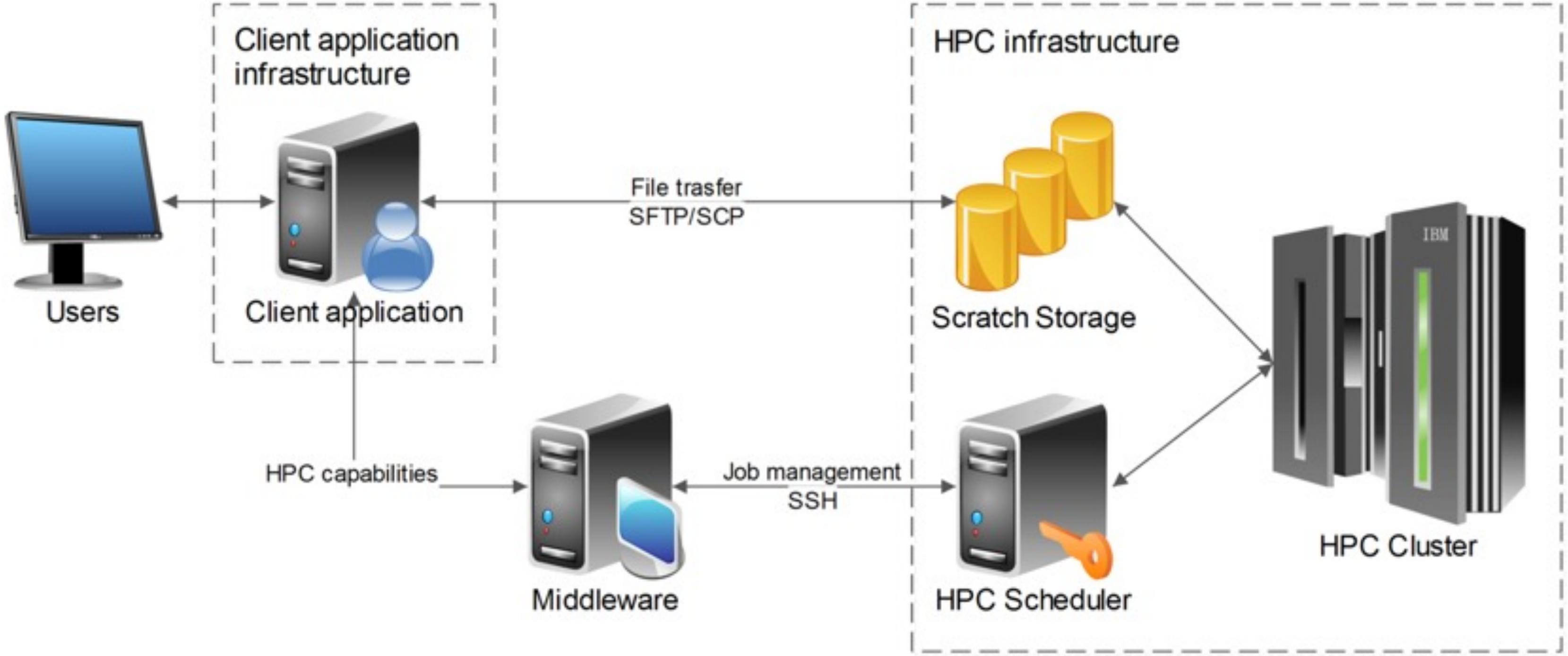
T. Kozubek, IT4Innovations, VSB-TUO



Customer product portfolio

Solver template

High-end application execution



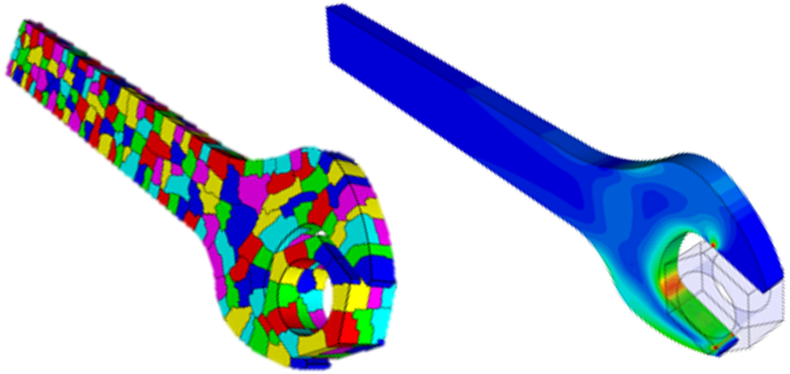
T. Kozubek, IT4Innovations, VSB-TUO

Massively parallel solvers

Methods

Efficient Solvers in Mechanics

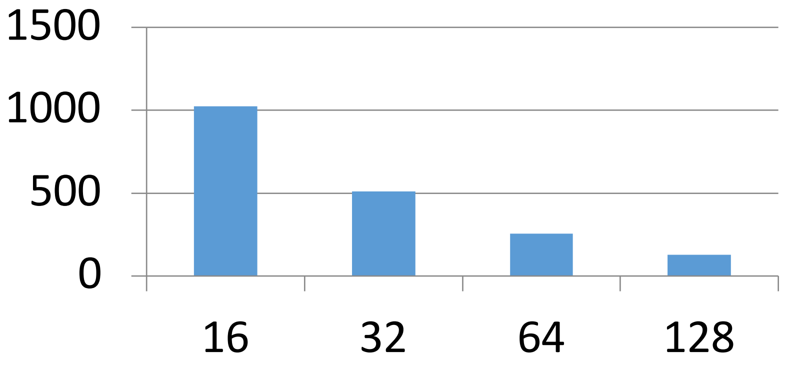
T. Kozubek, IT4Innovations, VSB-TUO



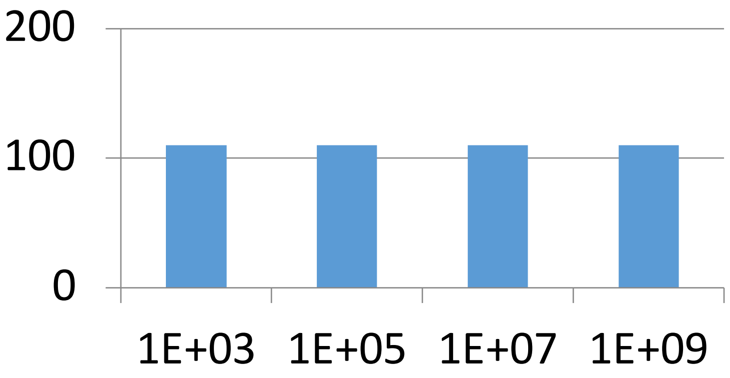
$$\mathbf{Ax = b}$$

$$\min_{\substack{lb \le x \le ub \\ B_{eq}x = c_{eq} \\ B_{ineq}x \le c_{ineq}}} \frac{1}{2} x^T A x - x^T b$$

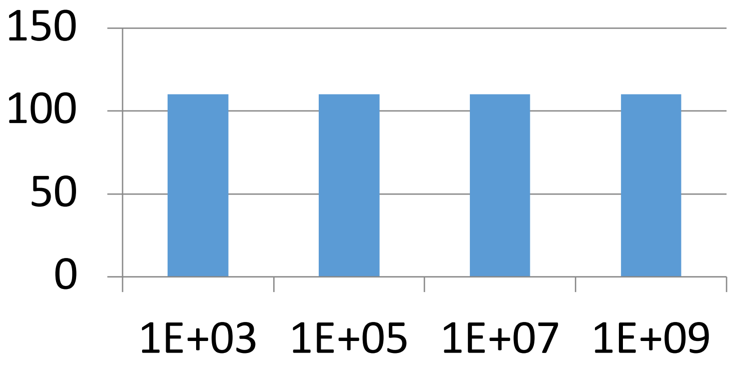
Strong parallel scalability
solution time / cores



Weak parallel scalability
sol. time / problem size



Numerical scalability
iterations / problem size

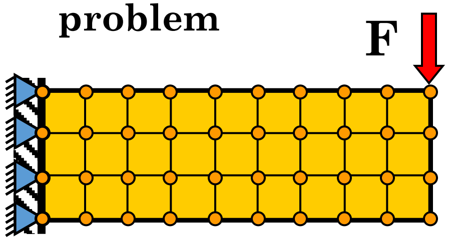


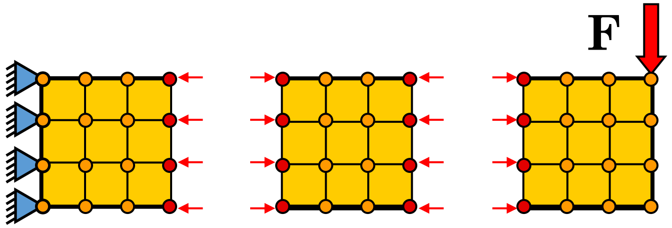
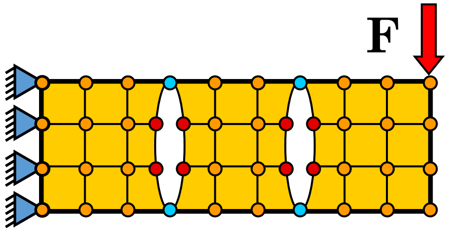
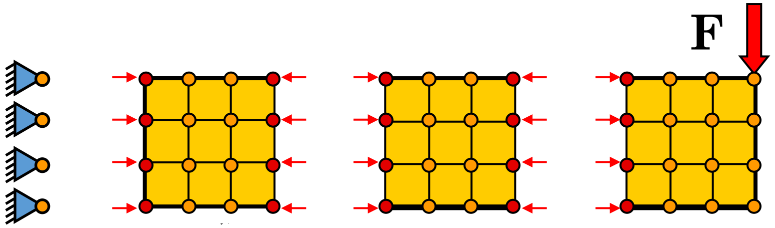
FETI approaches

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

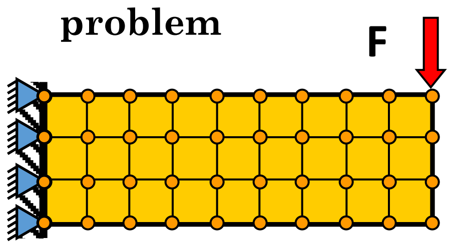
problem



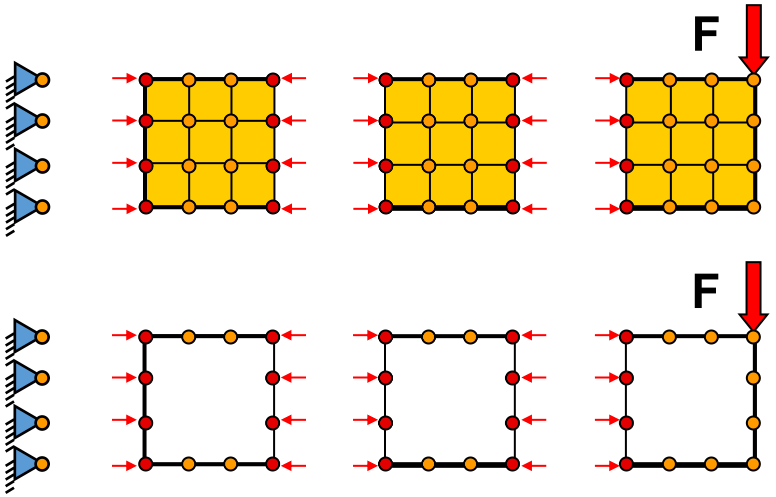
1. FETI		subdomains are fixed or free but with different defects
2. FETI-DP		FETI-DP (partial splitting, nonsingular)
3. TFETI		all subdomains are free with the same defect

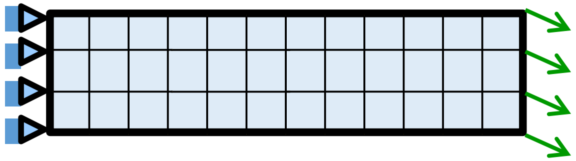
Discretization types

T. Kozubek, IT4Innovations, VSB-TUO

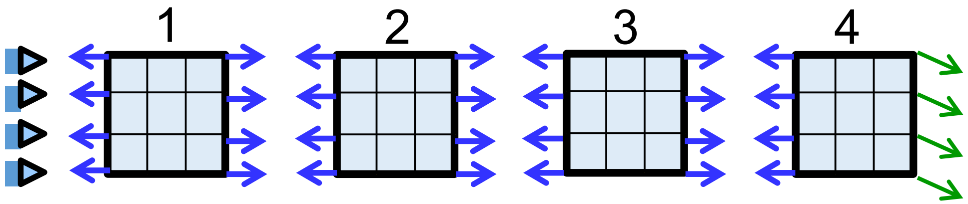


TFETI/TBETI

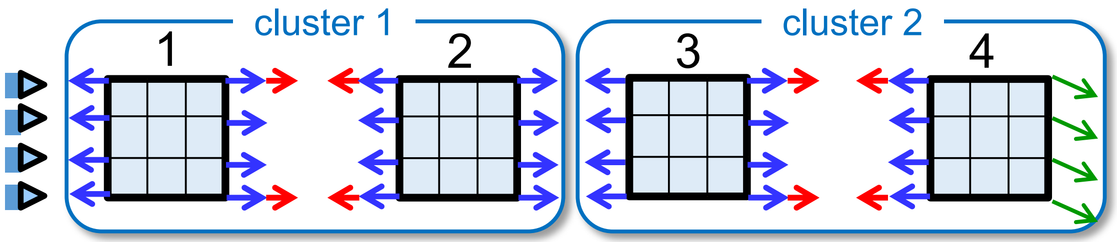




FEM discretization



FETI method

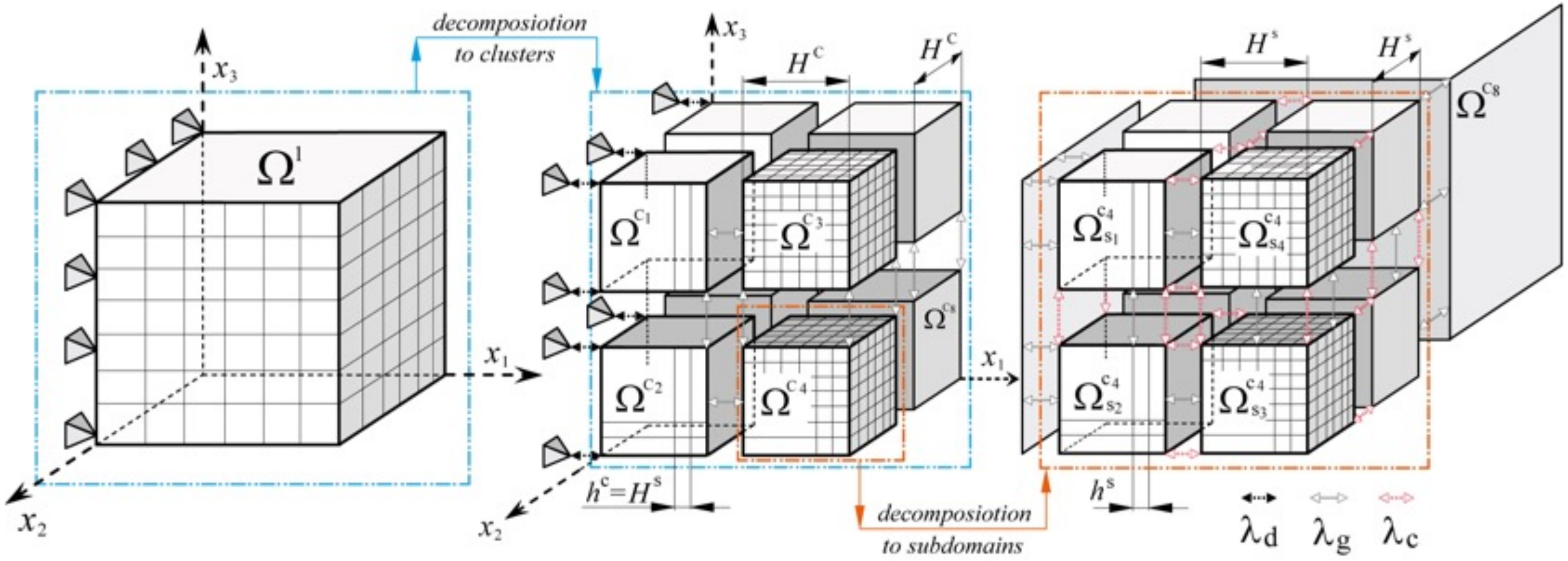


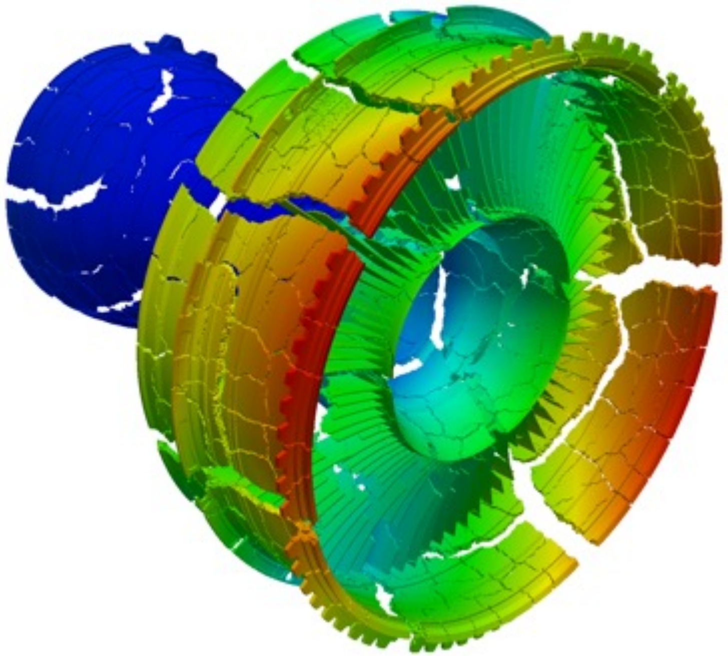
Hybrid FETI method (FETI among clusters and FETI-DP among subdomains in each cluster)

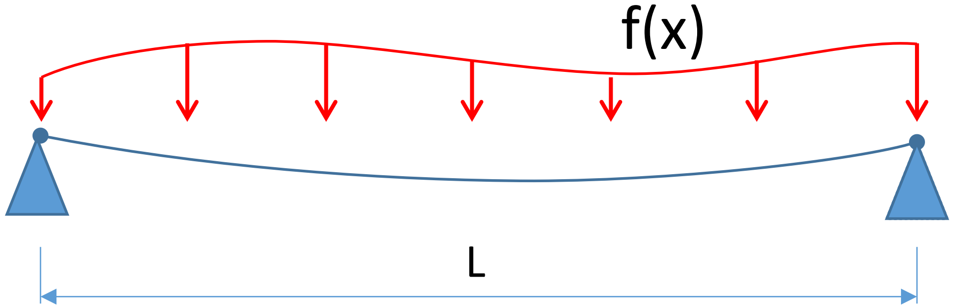
Hybrid Total FETI Method

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP







$$-Tu''(x) = f(x)$$

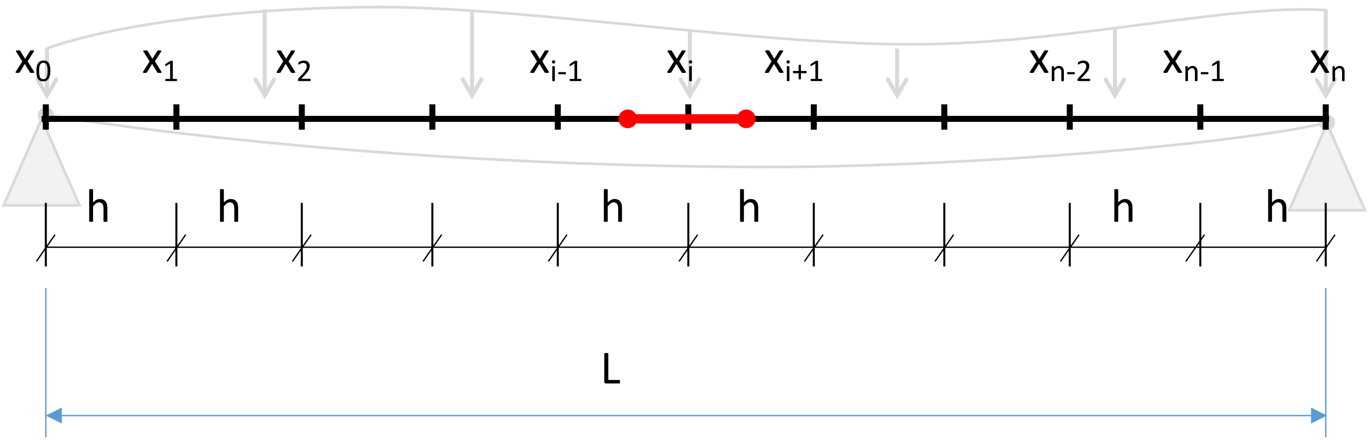
T – *tension force*

u – *deflection of the string*

f – *load force*

Algebraic formulation

T. Kozubek, IT4Innovations, VSB-TUO



$$-Tu''(x) = f(x)$$

$$-u_{i-1} + 2u_i - u_{i+1} = \frac{f_i h^2}{T}$$

$$\begin{pmatrix} 2 & -1 & 0 & & \cdots & 0 \\ -1 & 2 & -1 & 0 & & \vdots \\ 0 & -1 & 2 & -1 & \ddots & \\ & 0 & -1 & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & 2 & -1 \\ 0 & \cdots & & 0 & -1 & 2 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{pmatrix} = \frac{h^2}{T} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{pmatrix} + \begin{pmatrix} u_0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ u_n \end{pmatrix}$$

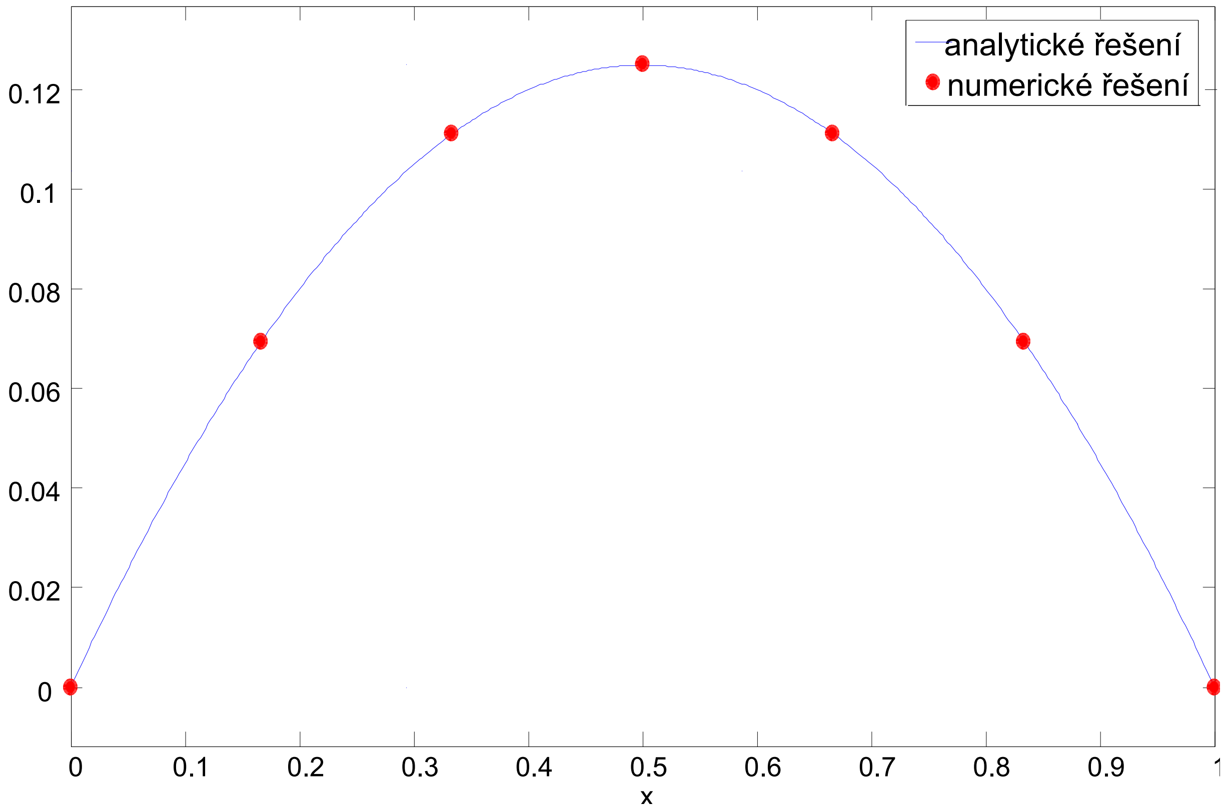
$$Ku = f$$

K – stiffness matrix

u – vector of unknowns

f – load vector

```
function string(n)
% -u''=1, u(0)=u(1)=0,
% n ... number of elements
L=1; f=1; T=1;
h=L/n;
e=ones(n-1,1);
A=spdiags([-e,2*e,-e],[-1,0,1],n-1,n-1);
b=f*ones(n-1,1)*h^2/T;
u=A\b; % Gaussian elimination

% Compare numer. and analytical solutions
U=[0;u;0];
x=(0:h:1)';
U_ex=@(x) -1/2*x.*(x-1);
plot(x,U,'o',x,U_ex(x))
norm(U-U_ex(x))
```

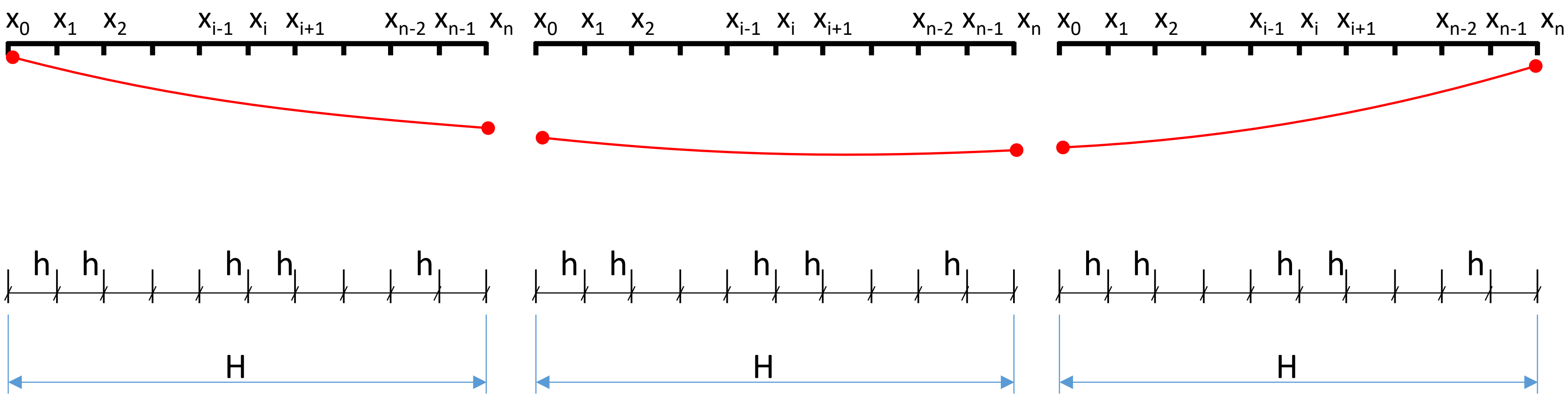


Algebraic formulation

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

The string is split into 3 parts, discretized by FEM, and related objects are assembled.



$$Ku = f \qquad \begin{pmatrix} K^1 & 0 & 0 \\ 0 & K^2 & 0 \\ 0 & 0 & K^3 \end{pmatrix} \begin{pmatrix} u^1 \\ u^2 \\ u^3 \end{pmatrix} = \begin{pmatrix} f^1 \\ f^2 \\ f^3 \end{pmatrix}$$

Continuity of the solution

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

$$u_i = u_j, \quad u_i - u_j = 0$$

$$B_1 u = 0, \ B_1 = [\dots, 0, 0, 1_i, -1_j, 0, \dots],$$
$$B_2 u = 0, \ B_2 = [\dots, 0, 0, 1_r, -1_s, 0, \dots]$$

$$B\,u = o, \ B = \begin{pmatrix} B_1 \\ B_2 \end{pmatrix}$$

B – constraint matrix

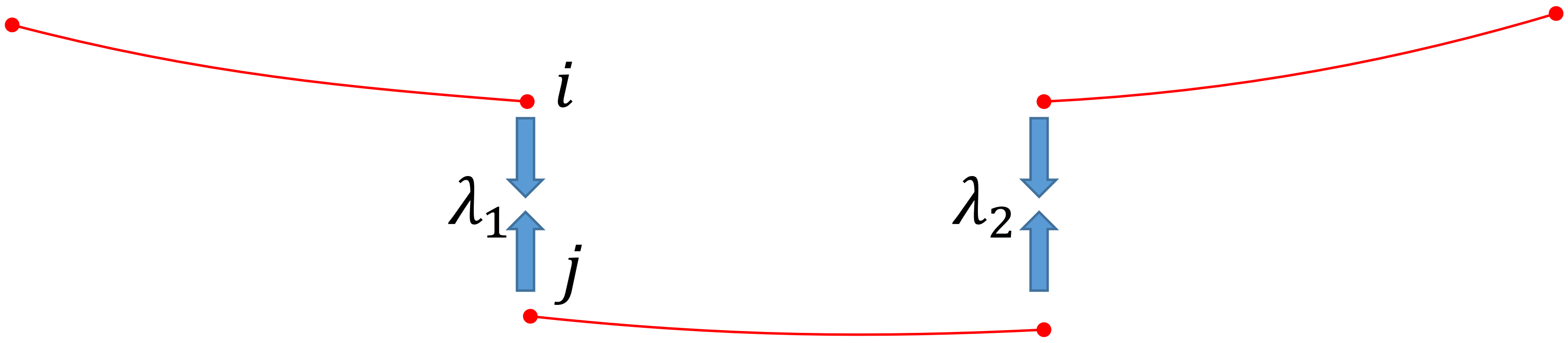
$u_i - u_j = 0$ – gluing condition

λ – Lagrange multiplier

Algebraic formulation

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



$$Ku = f - B^T \lambda$$

$$B\,u = o$$

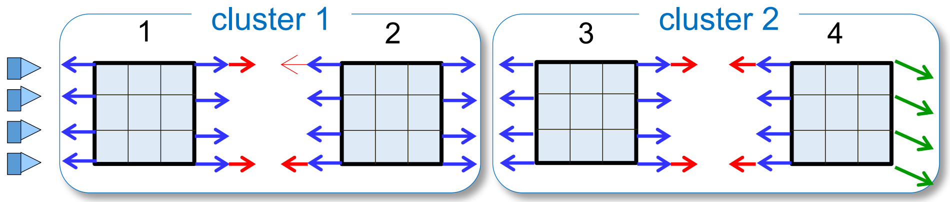
$$\begin{pmatrix} K & B^T \\ B & O \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ o \end{pmatrix}$$

We can solve the resulting system as a whole or express u from the first equation, substitute it to the second and solve smaller better conditioned problem only in λ.

Back to Hybrid FETI

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



$$\begin{pmatrix} K & B^T \\ B & O \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ o \end{pmatrix}$$

$$\mathbf{B}_c = \begin{pmatrix} \mathbf{B}_{c,1} & \mathbf{B}_{c,2} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{B}_{c,3} & \mathbf{B}_{c,4} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{B}_3 & \mathbf{B}_4 \end{pmatrix}$$

Additional constraints: duplication of ‘corners’ Lagrange multiplicators

$$\left(\begin{array}{cccc|cc|c} \mathbf{K}_1 & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{B}_{c,1}^\top & \mathbf{O} & \mathbf{B}_1^\top \\ \mathbf{O} & \mathbf{K}_2 & \mathbf{O} & \mathbf{O} & \mathbf{B}_{c,2}^\top & \mathbf{O} & \mathbf{B}_2^\top \\ \mathbf{O} & \mathbf{O} & \mathbf{K}_3 & \mathbf{O} & \mathbf{O} & \mathbf{B}_{c,3}^\top & \mathbf{B}_3^\top \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{K}_4 & \mathbf{O} & \mathbf{B}_{c,4}^\top & \mathbf{B}_4^\top \\ \hline \mathbf{B}_{c,1} & \mathbf{B}_{c,2} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{B}_{c,3} & \mathbf{B}_{c,4} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \hline \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{B}_3 & \mathbf{B}_4 & \mathbf{O} & \mathbf{O} & \mathbf{O} \end{array}\right) \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \\ \mathbf{u}_4 \\ \hline \boldsymbol{\lambda}_{c,1} \\ \boldsymbol{\lambda}_{c,2} \\ \hline \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \mathbf{f}_4 \\ \hline \mathbf{o} \\ \mathbf{o} \\ \hline \mathbf{c} \end{pmatrix}$$

Augmented system by the matrix $\mathbf{B}_c$ and λ_c

$$\left(\begin{array}{ccc|ccc|c} \mathbf{K}_1 & \mathbf{O} & \mathbf{B}_{c,1}^\top & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{B}_1^\top \\ \mathbf{O} & \mathbf{K}_2 & \mathbf{B}_{c,2}^\top & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{B}_2^\top \\ \mathbf{B}_{c,1} & \mathbf{B}_{c,2} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \hline \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{K}_3 & \mathbf{O} & \mathbf{B}_{c,3}^\top & \mathbf{B}_3^\top \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{K}_4 & \mathbf{B}_{c,4}^\top & \mathbf{B}_4^\top \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{B}_{c,3} & \mathbf{B}_{c,4} & \mathbf{O} & \mathbf{O} \\ \hline \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{O} & \mathbf{B}_3 & \mathbf{B}_4 & \mathbf{O} & \mathbf{O} \end{array}\right) \begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \boldsymbol{\lambda}_{c,1} \\ \hline \mathbf{u}_3 \\ \mathbf{u}_4 \\ \boldsymbol{\lambda}_{c,2} \\ \hline \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{o} \\ \hline \mathbf{f}_3 \\ \mathbf{f}_4 \\ \mathbf{o} \\ \hline \mathbf{c} \end{pmatrix}$$

Reordering according to clusters

Pipelining

T. Kozubek, IT4Innovations, VSB-TUO

Algorithm 4 Standard BiCGStab

function BICGSTAB(A, b, x_0)
  $r_0 := b - Ax_0$; $p_0 := r_0$
  for $i = 0, \dots$ **do**
    $s_i := Ap_i$
    compute (r_0, s_i)
    $\alpha_i := (r_0, r_i) / (r_0, s_i)$
    $q_i := r_i - \alpha_i s_i$
    $y_i := Aq_i$
    compute (q_i, y_i) ; (y_i, y_i)
    $\omega_i := (q_i, y_i) / (y_i, y_i)$
    $x_{i+1} := x_i + \alpha_i p_i + \omega_i q_i$
    $r_{i+1} := q_i - \omega_i y_i$
    compute (r_0, r_{i+1})
    $\beta_i := (\alpha_i/\omega_i)(r_0, r_{i+1}) / (r_0, r_i)$
    $p_{i+1} := r_{i+1} + \beta_i (p_i - \omega_i s_i)$
  end for
end function

dot-prod
SpMV
axpy

Traditional BiCGStab:

(non-preconditioned)

Global communication

- 3 global reduction phases

Semi-local communication

- 2 non-overlapping SpMVs

Local communication

- 4 axpy(-like) operations

General two-step framework for deriving pipelined Krylov methods:

Step 1. **Avoiding communication:** *merge global reductions*

Step 2. **Hiding communication:** *overlap SpMVs & global reductions*

Pipelined BiCGStab (p-BiCGStab)

T. Kozubek, IT4Innovations, VSB-TUO

Algorithm 6 Pipelined BiCGStab

function PIPE-BICGSTAB(A, b, x_0)
  $r_0 := b - Ax_0$; $w_0 := Ar_0$; $t_0 := Aw_0$;
  for $i = 0, \dots$ **do**
    $p_i := r_i + \beta_{i-1}\,(p_{i-1} - \omega_{i-1} s_{i-1})$
    $s_i := w_i + \beta_{i-1}\,(s_{i-1} - \omega_{i-1} z_{i-1})$
    $z_i := t_i + \beta_{i-1}\,(z_{i-1} - \omega_{i-1} v_{i-1})$
    $q_i := r_i - \alpha_i s_i$
    $y_i := w_i - \alpha_i z_i$
    compute (q_i, y_i) ; (y_i, y_i)
    $\omega_i := (q_i, y_i) \,/\, (y_i, y_i)$
    overlap $v_i := Az_i$
    $x_{i+1} := x_i + \alpha_i p_i + \omega_i q_i$
    $r_{i+1} := q_i - \omega_i y_i$
    $w_{i+1} := y_i - \omega_i\,(t_i - \alpha_i v_i)$
    compute (r_0, r_{i+1}) ; (r_0, w_{i+1}) ; (r_0, s_i) ; (r_0, z_i)
    $\beta_i := (\alpha_i/\omega_i)\,(r_0, r_{i+1}) \,/\, (r_0, r_i)$
    $\alpha_{i+1} := (r_0, r_{i+1}) \,/\, ((r_0, w_{i+1}) + \beta_i\,(r_0, s_i) - \beta_i \omega_i\,(r_0, z_i))$
    overlap $t_{i+1} := Aw_{i+1}$
  end for
end function

dot-prod
SpMV
axpy

p-BiCGStab:
(non-preconditioned)

Global communication

- 2 global red. phases (vs. 3)

Semi-local communication

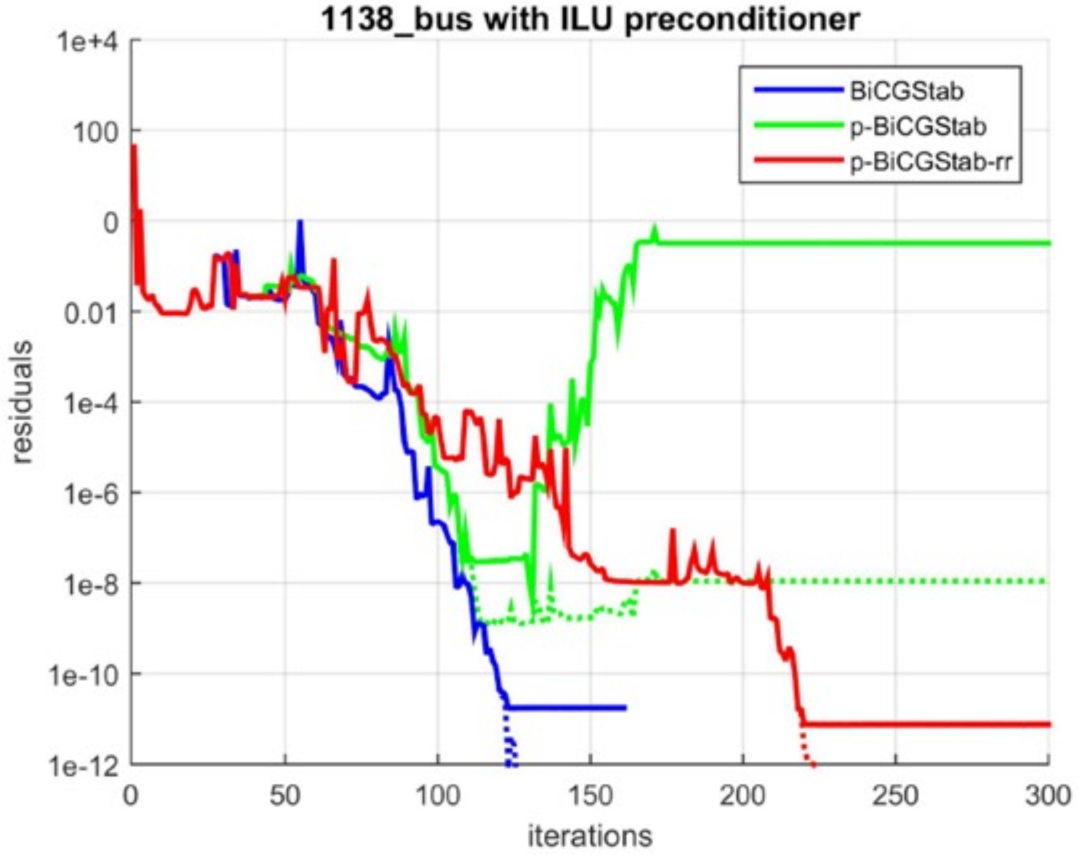
- 2 overlapping SpMVs

Local communication

- 8 axpy(-like) operations (vs. 4)

Status after Step 2: both global comm. phases are overlapped with SpMV computations ('hidden'), at the cost of 4 additional axpys

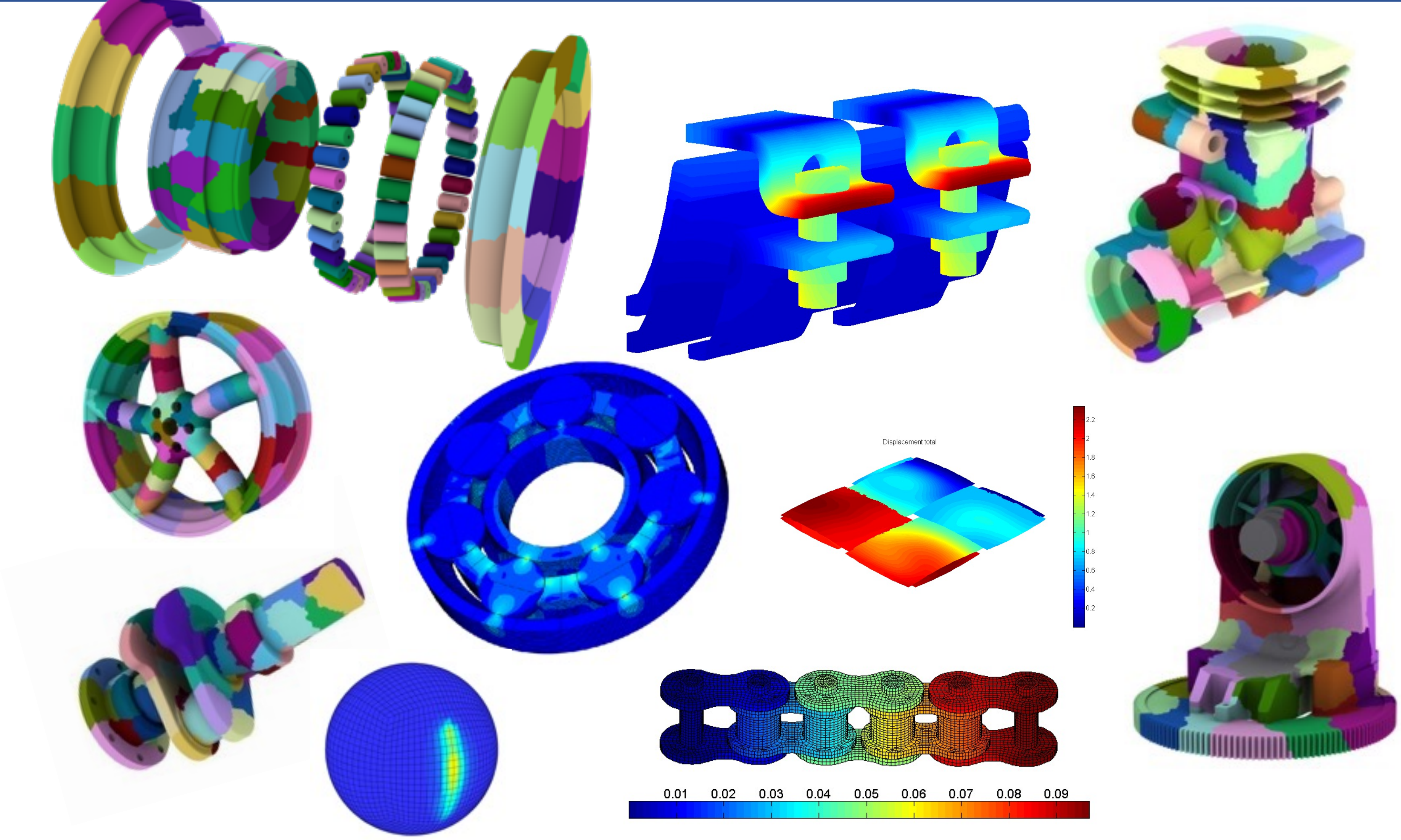
Robustness and attainable accuracy: p-BiCGStab-rr (with residual replacements)



Contact problems

Benchmarks

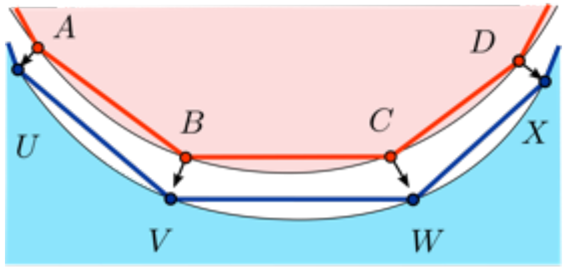
T. Kozubek, IT4Innovations, VSB-TUO



Contact discretization types

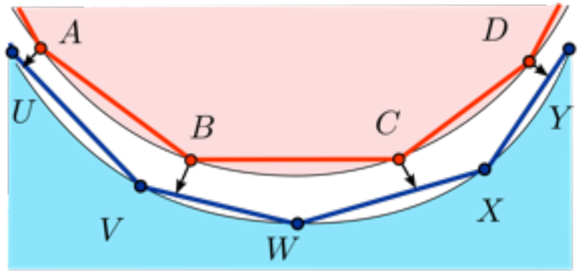
T. Kozubek, IT4Innovations, VSB-TUO

- node x node



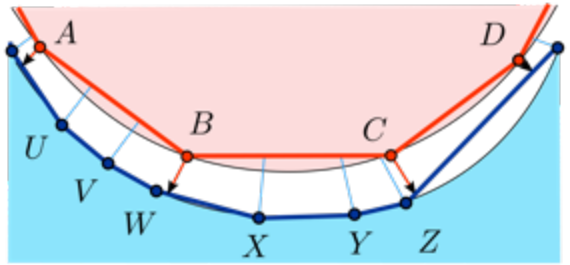
$$\begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 1 & \cdots & -1 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \mathbf{u} = \begin{bmatrix} \vdots \\ 0 \\ \vdots \end{bmatrix}$$

- node x element



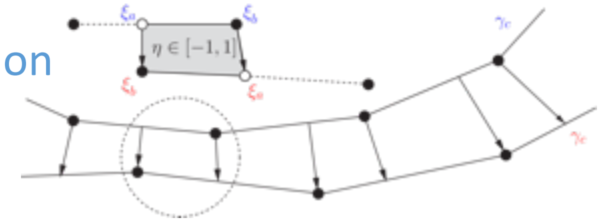
$$\begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & 1 & \cdots & -\alpha & \cdots & -1+\alpha & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \mathbf{u} = \begin{bmatrix} \vdots \\ 0 \\ \vdots \end{bmatrix}$$

- mortars

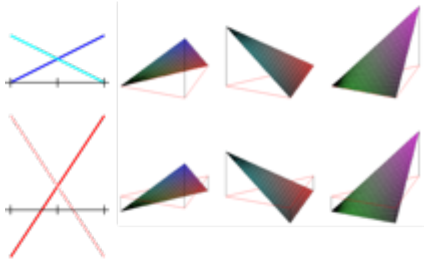


$$\begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \int_{\gamma_c} \psi_i \mathrm{d}\gamma & \cdots & \int_{\gamma_c} \psi_i \phi_{j_1} \mathrm{d}\gamma & \cdots & \int_{\gamma_c} \psi_i \phi_{j_n} \mathrm{d}\gamma & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \mathbf{u} = \begin{bmatrix} \vdots \\ 0 \\ \vdots \end{bmatrix}$$

Segmentation

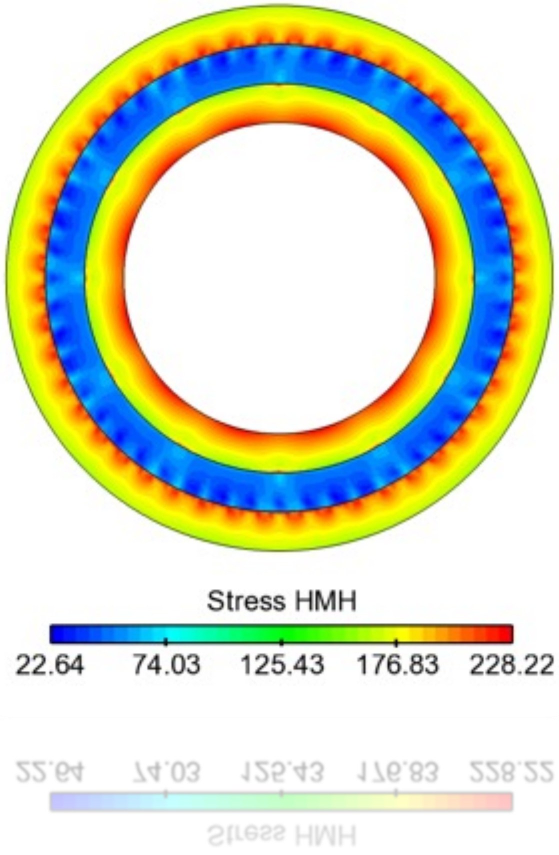


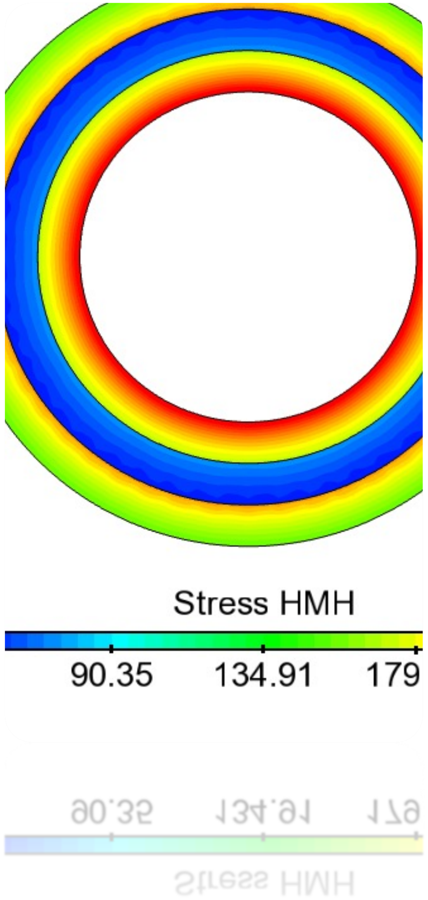
Primal x dual
Basis functions



Contact discretization

T. Kozubek, IT4Innovations, VSB-TUO





SemiSmooth Newton Methods

T. Kozubek, IT4Innovations, VSB-TUO

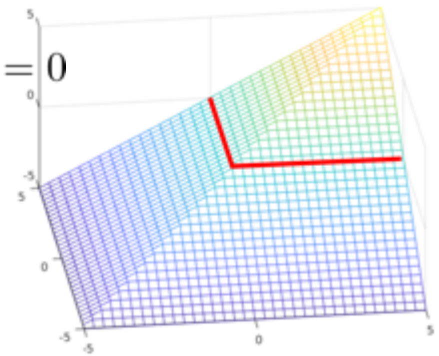
SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

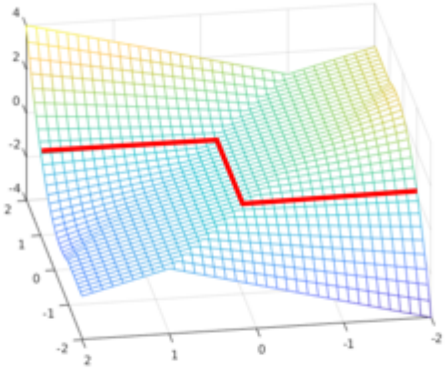
- Write inequality constraints as level 0 of nonsmooth function

$$a \geq 0,\ b \geq 0,\ ab = 0$$ nonpenetration

$$\Updownarrow$$

$$C(a,b) := a - \max\{0, a-b\} = 0$$





Tresca friction

$$g - |a| \geq 0,\ b - \beta a = 0,\ \beta \geq 0,\ \beta\,(g - |a|) = 0$$

$$\Updownarrow$$

$$D(a,b) := \max\{g, |a + \alpha b|\} a - g\,(a + \alpha b) = 0$$

- Solve equality system by SSNM

$$\begin{bmatrix} \mathbf{K}_{\mathcal{N}\mathcal{N}} & \mathbf{K}_{\mathcal{N}\mathcal{M}} & \mathbf{K}_{\mathcal{N}\mathcal{I}} & \mathbf{K}_{\mathcal{N}\mathcal{A}} & 0 & 0 \\ \mathbf{K}_{\mathcal{M}\mathcal{N}} & \mathbf{K}_{\mathcal{M}\mathcal{M}} & \mathbf{K}_{\mathcal{M}\mathcal{I}} & \mathbf{K}_{\mathcal{M}\mathcal{A}} & -\mathbf{M}_{\mathcal{I}}^{\top} & -\mathbf{M}_{\mathcal{A}}^{\top} \\ \mathbf{K}_{\mathcal{I}\mathcal{N}} & \mathbf{K}_{\mathcal{I}\mathcal{M}} & \mathbf{K}_{\mathcal{I}\mathcal{I}} & \mathbf{K}_{\mathcal{I}\mathcal{A}} & \mathbf{D}_{\mathcal{I}} & 0 \\ \mathbf{K}_{\mathcal{A}\mathcal{N}} & \mathbf{K}_{\mathcal{A}\mathcal{M}} & \mathbf{K}_{\mathcal{A}\mathcal{I}} & \mathbf{K}_{\mathcal{A}\mathcal{A}} & 0 & \mathbf{D}_{\mathcal{A}} \\ 0 & 0 & 0 & 0 & \mathbf{I}_{\mathcal{I}} & 0 \\ 0 & \widetilde{\mathbf{M}}_{\mathcal{A}} & \widetilde{\mathbf{S}}_{\mathcal{A}\mathcal{I}} & \widetilde{\mathbf{S}}_{\mathcal{A}} & 0 & 0 \\ 0 & 0 & \widetilde{\mathbf{F}}_{\mathcal{A}\mathcal{I}} & \widetilde{\mathbf{F}}_{\mathcal{A}\mathcal{A}} & 0 & \mathbf{T}_{\mathcal{A}} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}_{\mathcal{N}} \\ \Delta\mathbf{u}_{\mathcal{M}} \\ \Delta\mathbf{u}_{\mathcal{I}} \\ \Delta\mathbf{u}_{\mathcal{A}} \\ \boldsymbol{\lambda}_{\mathcal{I}} \\ \boldsymbol{\lambda}_{\mathcal{A}} \end{bmatrix} = - \begin{bmatrix} \mathbf{r}_{\mathcal{N}} \\ \mathbf{r}_{\mathcal{M}} \\ \mathbf{r}_{\mathcal{I}} \\ \mathbf{r}_{\mathcal{A}} \\ 0 \\ \widetilde{\mathbf{g}}_{\mathcal{A}} \\ 0 \end{bmatrix}$$

(nonpenetration only)

Initialize $(\mathbf{u}^0, \boldsymbol{\lambda}^0)$ and $\mathcal{A}^0, \mathcal{I}^0$

Find $(\Delta\mathbf{u}^k, \boldsymbol{\lambda}^{k+1})$ solving

Update $\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta\mathbf{u}^k$ and $\mathcal{A}^{k+1}, \mathcal{I}^{k+1}$

Repeat until

$$\mathcal{A}^{k+1} = \mathcal{A}^k, \mathcal{I}^{k+1} = \mathcal{I}^k, \|\mathbf{r}^{k+1}\| \leq \varepsilon$$

Final results

Massively parallel solvers

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

- FETI with core oversubscription
- FETI with hybrid parallelization
- Hybrid FETI
  - clusterization by classical corners
  - clusterization by local kernels
- Preconditioning
  - orthogonal projector
  - conjugate projector for dynamic analysis
  - LUMPED
  - DIRICHLET
  - LIGHT DIRICHLET
  - scaling for coefficient jumps
  - GENEO
- Iterative solvers
  - PCG, Pipelined PCG
  - GMRES, BiCGStab
  - Full Orthogonal PCG
    - Adaptive precision control for nonlinear loops
  - SMALSE - semi-monotonic augmented Lagrangian method with separable convex constraints and general equality constraints
- Third party Solvers Interface
  - HYPRE BoomerAMG as a solver
  - HYPRE BoomerAMG as a preconditioner
  - Parallel Sparse Direct Solvers
    - Intel MKL - PDSS
    - IBM – WATSON
    - SuperLU

Designed to take full advantage of today's most powerful petascale supercomputers

FETI based domain decomposition solver
Hybrid FETI domain decomposion solever (with hybrid implementation)
Preconditioners, direct and iterative solvers

CPU version

for Massively Parallel systems

GPU version

Nvidia GPU accelerated version for systems with hybrid architectures

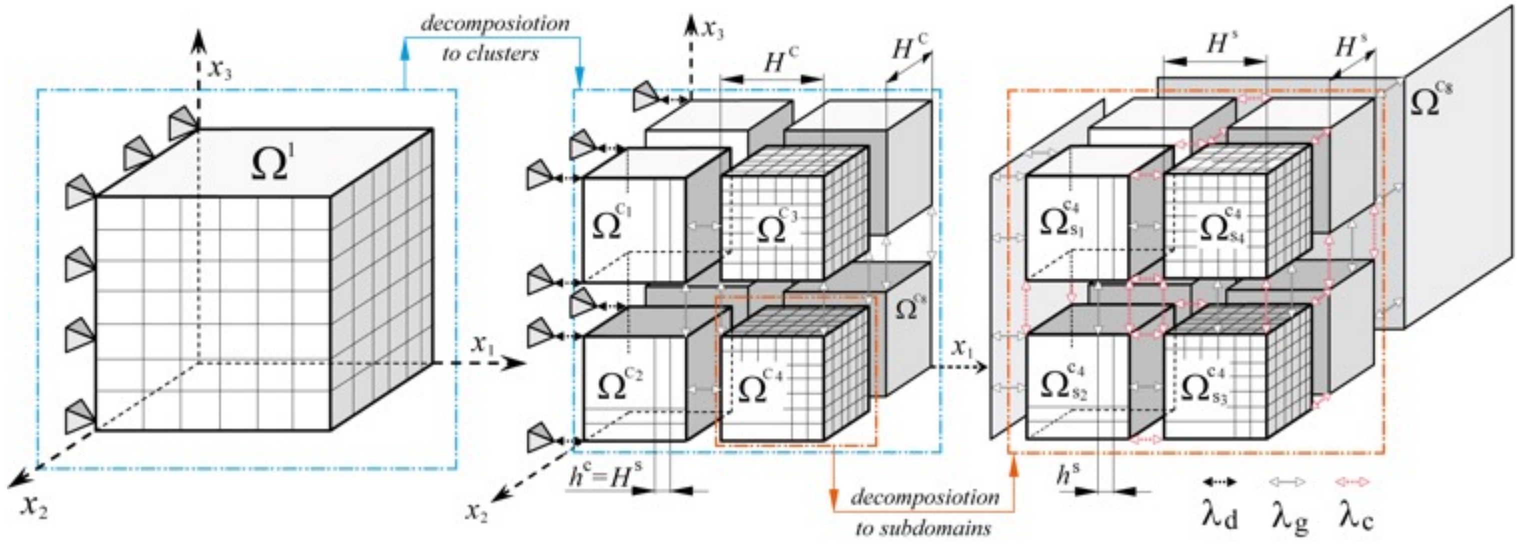
Automatic tuning

solver setting by evolutionary and swarm optimisation algorithms

Successfully Tested on large Peta-scale machines

125 on **281,216**

Billion unknowns — #CPU Cores



Massively parallel solvers

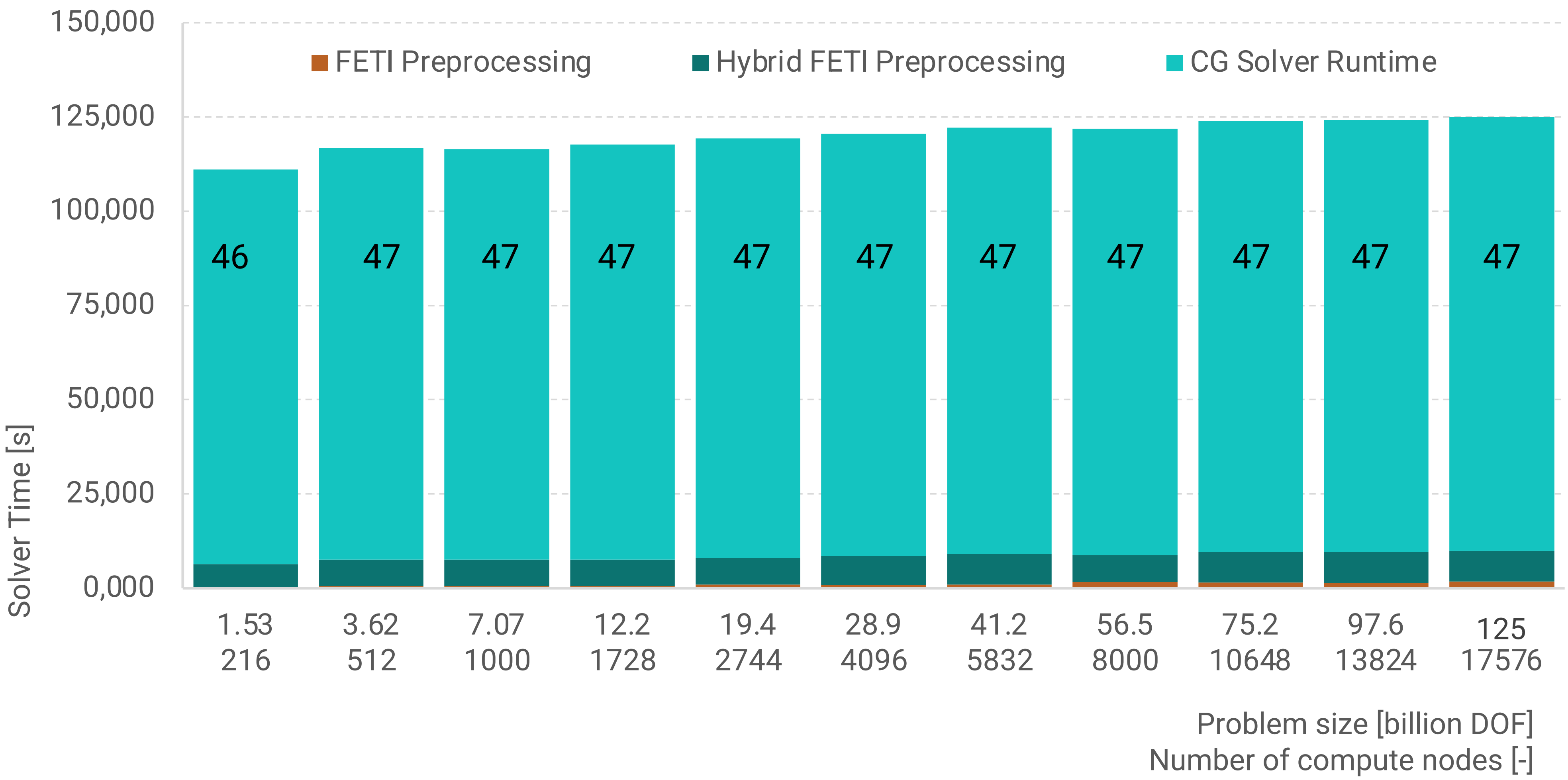
T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

Weak Scalability Test

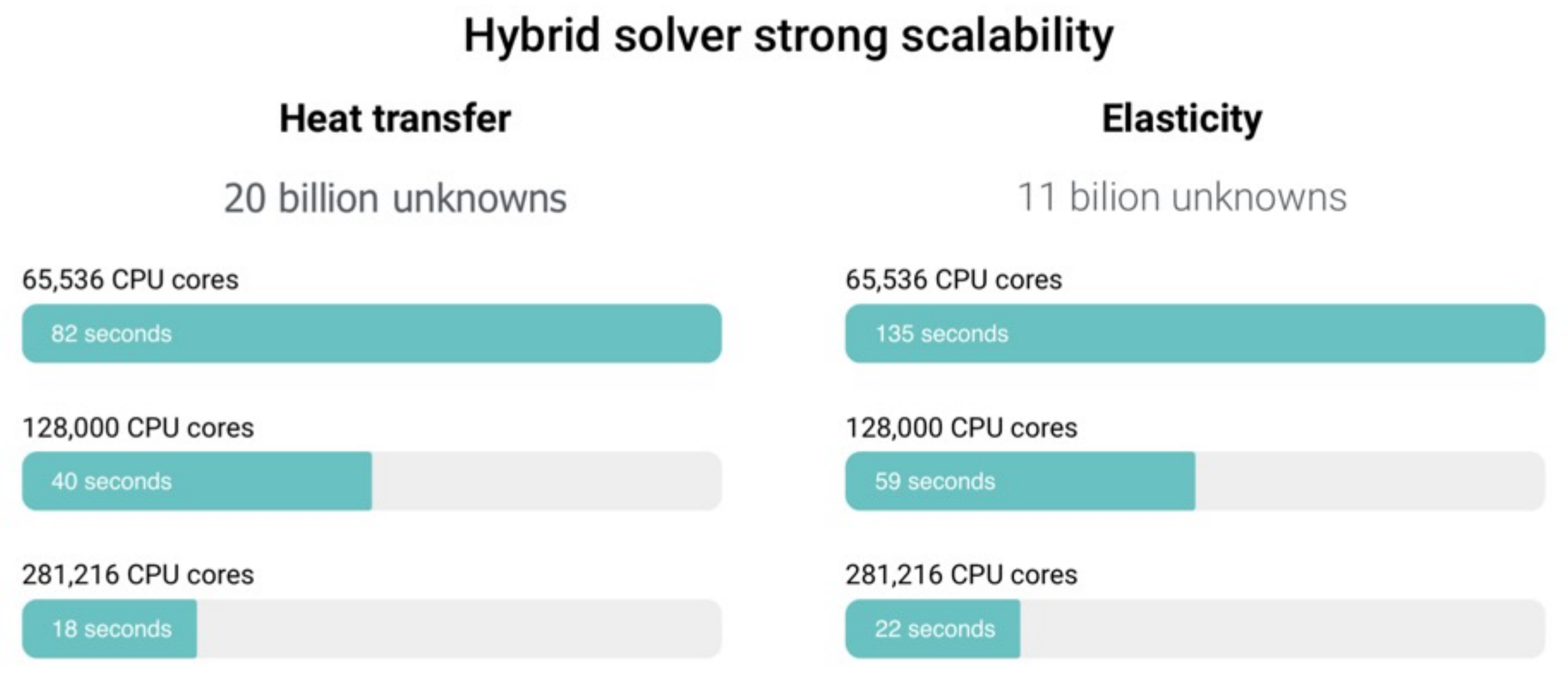
Up to **125 billion** DOF on 17576 Compute Nodes (281 216 cores)

Heat transfer (Laplace equation)



Massively parallel solvers

T. Kozubek, IT4Innovations, VSB-TUO

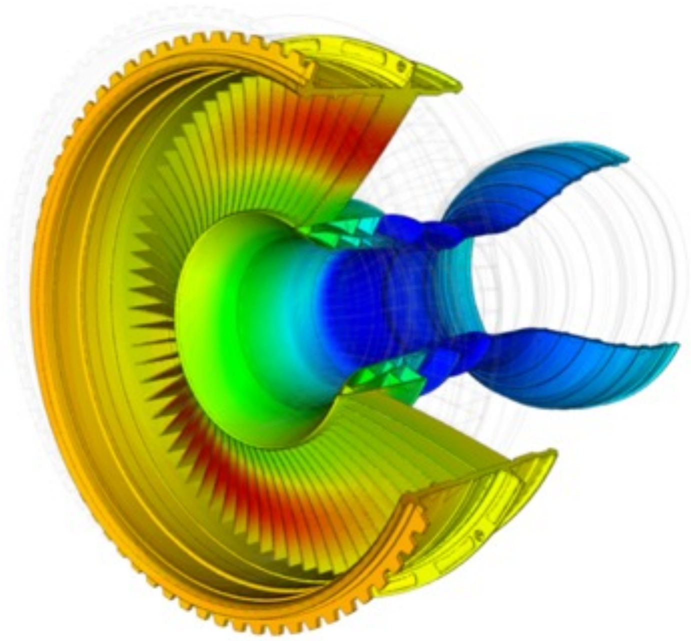


Massively parallel solvers

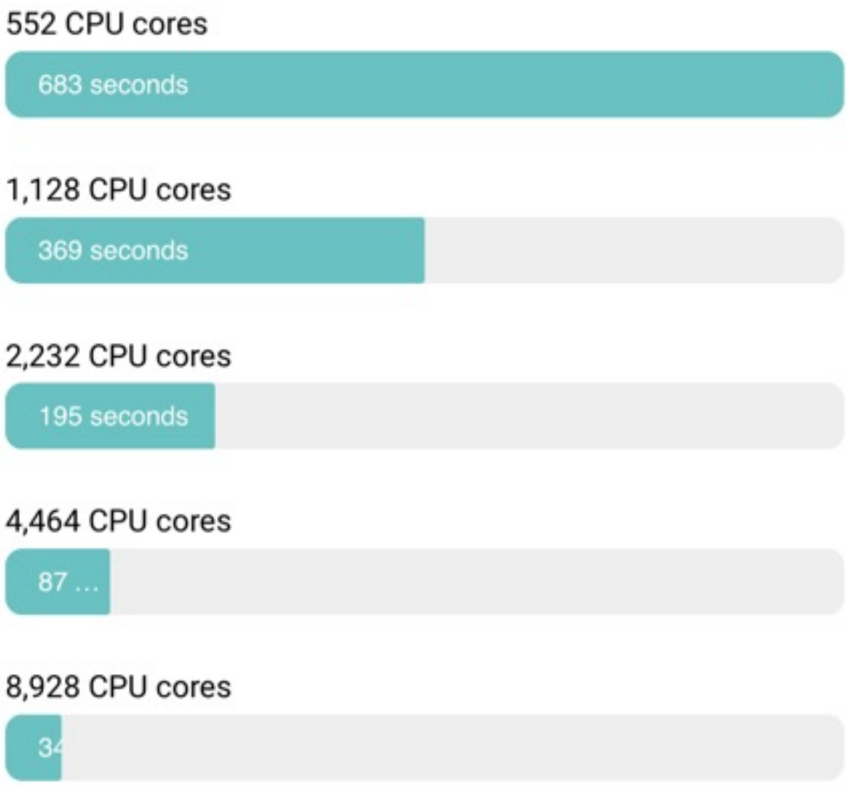
T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

Strong scalability - elasticity problem



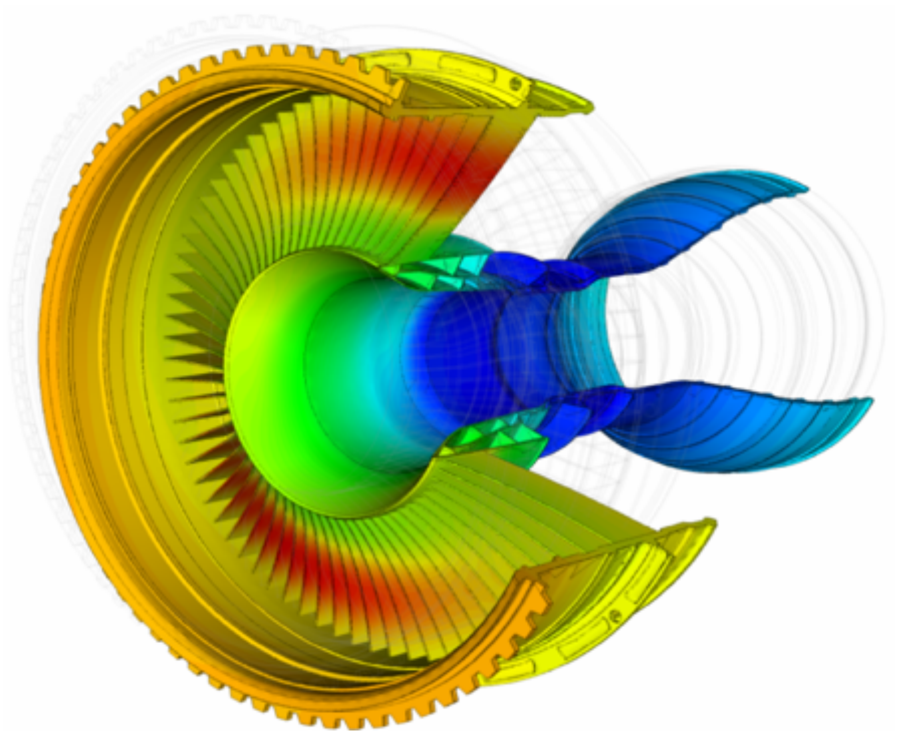
300 Million unknowns Jet Engine case

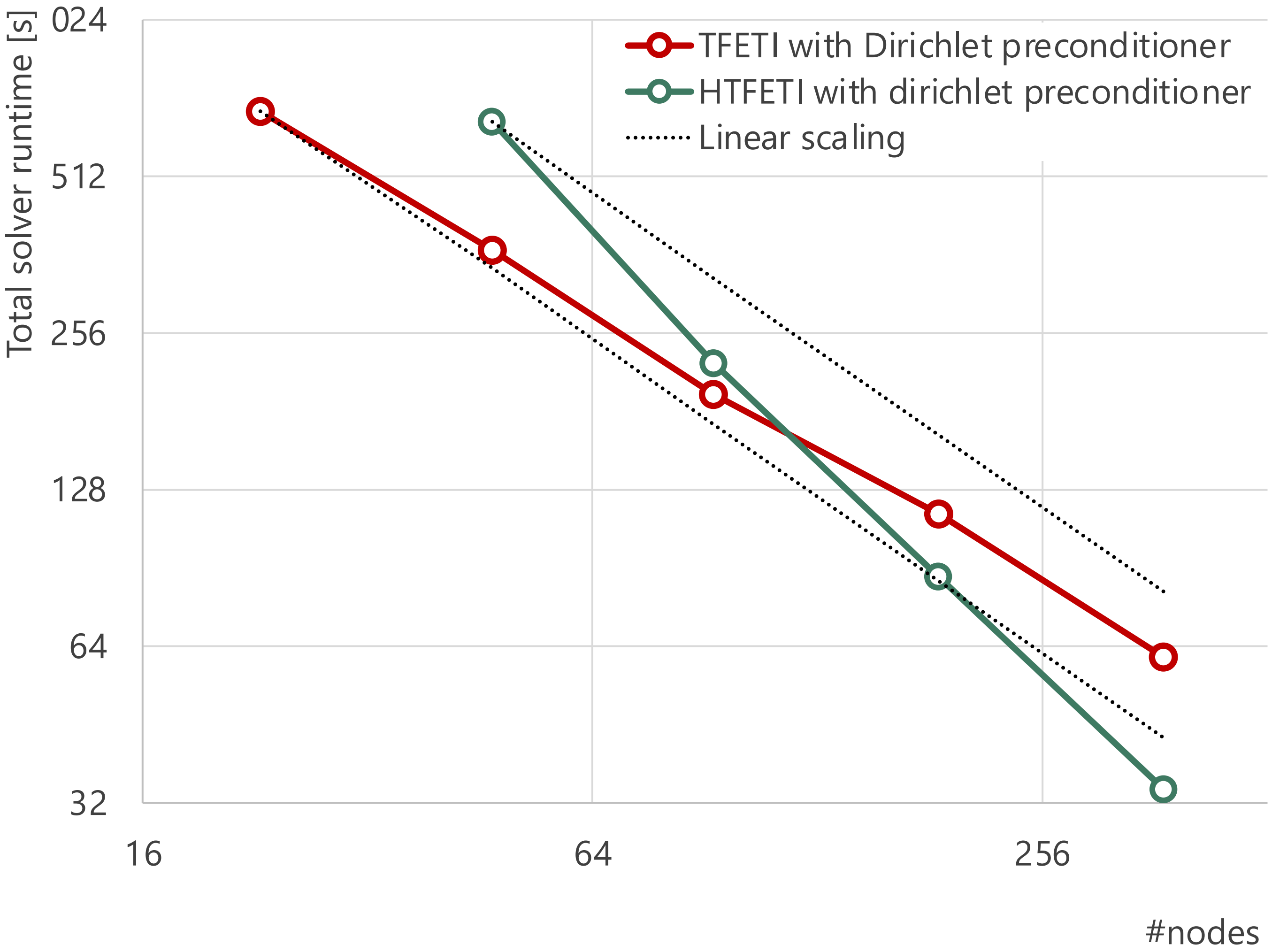


Massively parallel solvers

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP



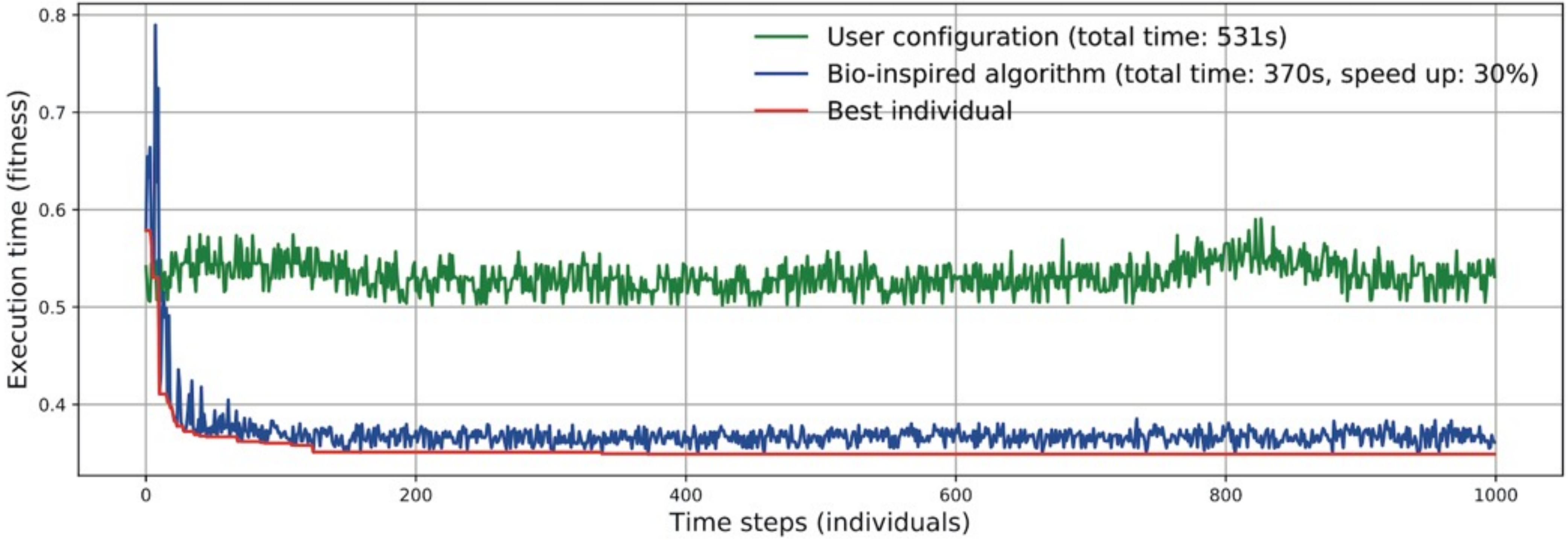


Automatic tuning

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

Automatic tuning for the parallel solver based on evolutionary and swarm algorithms



- FETI variant
- Precondition type
- Iterative solver type
- Corse space
- Clusterization by corners/kernels

30% speedup in comparison with reference user configuration

ESPRESO CG in FETI

T. Kozubek, IT4Innovations, VSB-TUO

Projected Conjugate Gradient in FETI

$r_0 := b - Ax_0;\ u_0 := M^{-1} r_0;\ p_0 := u_0$
for $i = 0, \ldots, m-1$ **do**
  $s := Ap_i$
  $\alpha := \langle r_i, u_i \rangle / \langle s, p_i \rangle$
  $x_{i+1} := x_i + \alpha p_i$
  $r_{i+1} := r_i - \alpha s$
  $u_{i+1} := M^{-1} r_{i+1}$
  $\beta := \langle r_{i+1}, u_{i+1} \rangle / \langle r_i, u_i \rangle$
  $p_{i+1} := u_{i+1} + \beta p_i$
end for

Pre-processing – K factorization
1.) $x = B_1^T \cdot \lambda$ - **SpMV**
2.) $y = K^{-1} \cdot x$ - **solve**
3.) $\lambda = B_1 \cdot y$ - **SpMV**
4.) stencil data exchange in λ
- MPI – Send and Recv
- OpenMP – shared mem. Vec

Pre-processing - $S_c = B_1 K^{-1} B_1^T$
1.) - **nop**
2.) $\lambda = S_c \cdot \lambda$ - **DGEMV, DSYMV**
3.) - **nop**
4.) stencil data exchange in λ
- MPI – Send and Recv
- OpenMP – shared mem. Vec

Pre-processing - $S_c = B_1 K^{-1} B_1^T \rightarrow$ GPU/MIC
1.) $\lambda \rightarrow$ GPU/MIC - **PCIe transfer from CPU**
2.) $\lambda = S_c \cdot \lambda$ - **DGEMV, DSYMV on GPU/MIC**
3.) $\lambda \leftarrow$ GPU/MIC - **PCIe transfer to CPU**
4.) stencil data exchange in λ
- MPI – Send and Recv
- OpenMP – shared mem. vec

90 – 95% of runtime spent in Ap_i

GPU acceleration of ESPRESO

T. Kozubek, IT4Innovations, VSB-TUO

SCtrain | SUPERCOMPUTING KNOWLEDGE PARTNERSHIP

0.3 - 300 million DOF Hybrid FETI CG Solver Runtime

Linear elasticity

ORNL Titan 5rd in TOP500 LIST

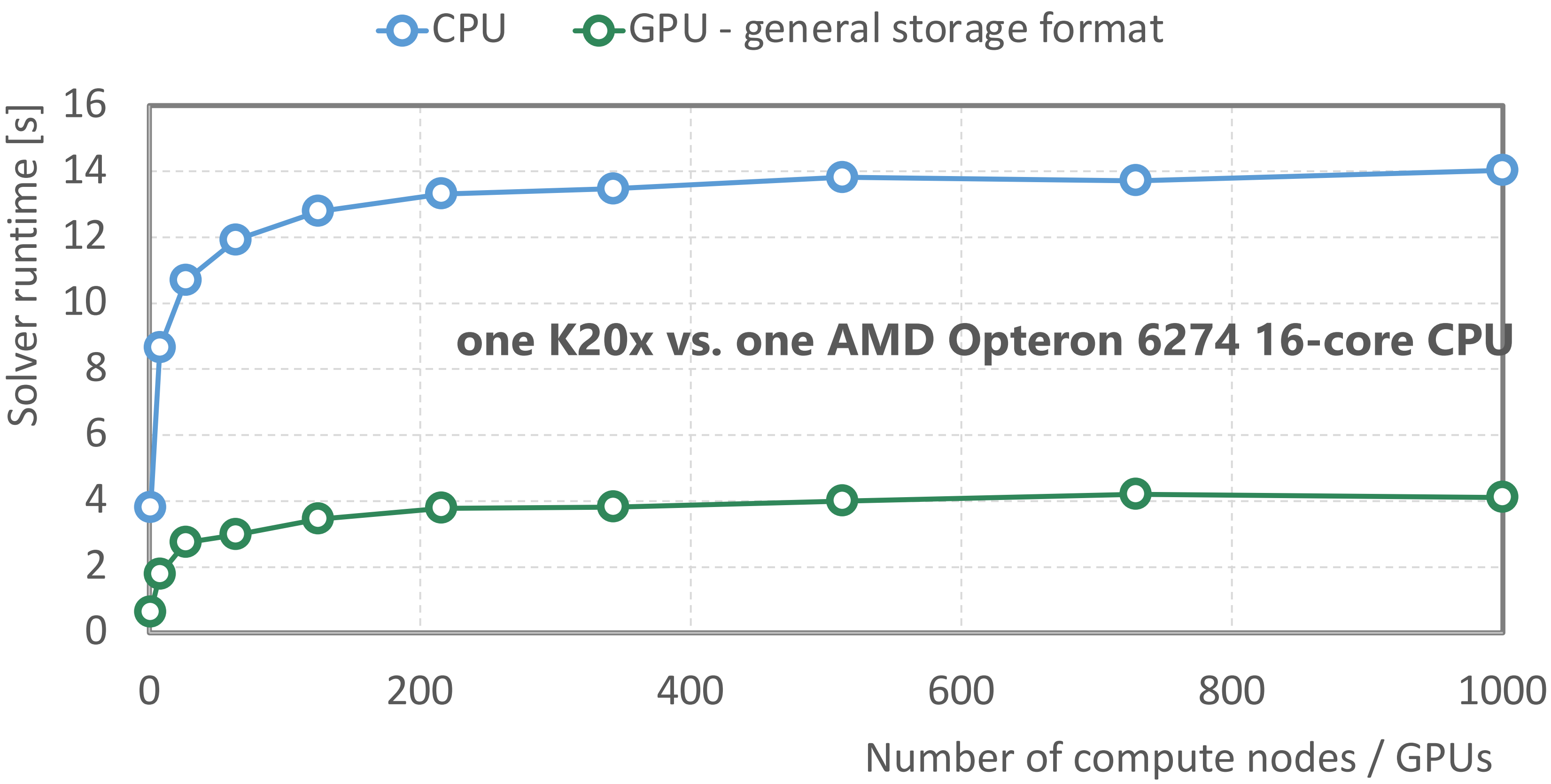
speedUp 3.4

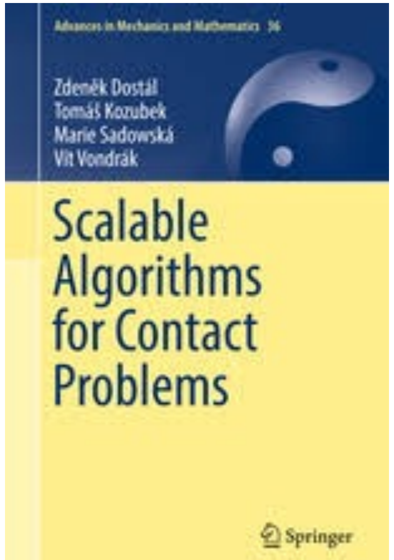
Current work:

- support for cuDense and cuSparse solvers from CUDA toolkit
- memory optimization for symmetric local schur complements
- implementation of memory efficient methods

Future work:

- Support for Power8/9 platforms with Pascal/Volta GPUs with NVLink



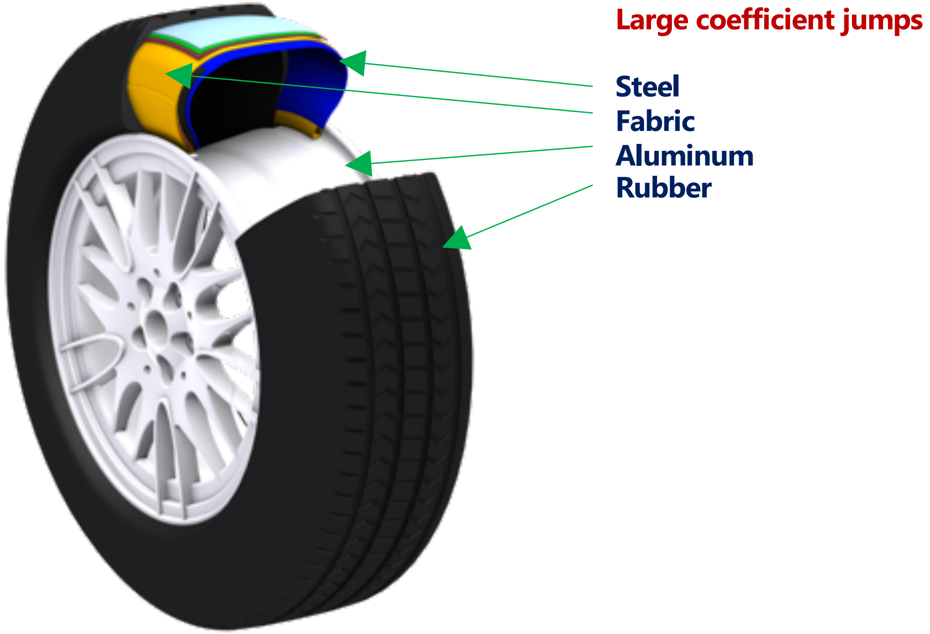


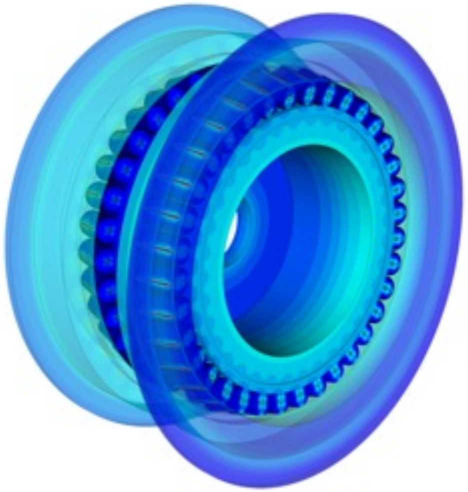
ESPRESO Contact Problems

TFETI with QPCE – Real world problem – Tire Rim assembly

Geometrically nonlinear problem - contact with rigid roadway

IT4Innovations – SALOMON Supercomputer





Thank you for your attention!

http://sctrain.eu/

Univerza v Ljubljani

TECHNISCHE UNIVERSITÄT WIEN

CINECA consorzio interuniversitario

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission.
This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.