Urychlení návrhu a optimalizace teplotních senzorů s využitím HPC technologií

Tomáš Kozubek
IT4Innovations, VŠB-TUO

Konference e-infrastruktury CESNET, 29-30. ledna 2019
Motivation for Exascale - Digital Twin Technology

- Electric fields
- Electromagnetism
- Heat transfer
  - heat generated by magnetism
  - cooling system
- Structural Mechanics
  - structural integrity
  - vibration from motion
  - high speed motors
  - influenced by electromagnetism
- Active cooling system
  - fluid flow
- Acoustic
  - generated by fluid flow
  - generated by electromagnetism
  - generated by vibrations

Complex nonlinear multiphysical and multiscale problem – electric motor
Multiphysics simulations

Simulation results
• Complex velocity and pressure fields
  • Ventilation looses
  • Cooling efficiency

One simulation
• 35 hours / 1200 Cores on Salomon
• “Standard” powerful workstation
• 256GB RAM 30 Cores 5 ~2 months!

What we need for optimization?
• Geometry morphing
• different number of ribs
• different fan shape
• boundary conditions variation
• ...

Lead’s to 1000’s simulations
One cluster isn't enough
Software for engineering simulations

• Creating complex model, computational mesh, boundary condition definition, material models – pre-processing
• Solution by numerical methods
• Results analysis – post-processing

1. Open source: OpenFOAM, Code Saturne, SU², Elmer, Dune, ...
https://www.cfd-online.com/Wiki/Codes#Free_codes

2. Commercial codes: ANSYS CFD, FLUENT, CFX, StarCCM+, COMSOL, ...
https://www.cfd-online.com/Wiki/Codes#Commercial_codes

3. Or try to write it in your own way
Matlab, R, Octave, Python, C, C++, Fortran, ...
Scalable algorithms development

- Boeing 787 Dreamliner: 106.2 Million Cells
- Time Saved: 99%
- Times faster solution: 141

Graph showing the relationship between number of cores and time saved.

Bar chart comparing FETI Preprocessing, Hybrid FETI Preprocessing, and CG Solver Runtime.

- Problem size [billion DOF]
- Number of compute nodes [-]

Examples:
- FETI Preprocessing: 1.53, 1.94, 7.07, 12.2, 19.4, 28.9, 41.2, 56.5, 75.2, 97.6, 124
- Hybrid FETI Preprocessing: 3.62, 4.06, 7.07, 12.2, 19.4, 28.9, 41.2, 56.5, 75.2, 97.6, 124
- CG Solver Runtime: 512, 1000, 1728, 2744, 4096, 5832, 8000, 10648, 13824, 17576
ESPRESO parallel solver

Preprocessing
- ANSYS/
  OpenFOAM
- ESPRESO
  Generator
- ESPRESO API

Mesh Processing
- Matrix Assembler
- FEM/BEM (BEM4I)
- TFETI/Hybrid TFETI

Solvers

ESPRESO C++
library

Postprocessing
- Visualization
- EnSight / Visit
- Paraview
  Catalyst

CPU
MIC
GPU

MPI, cilk++(or openMP), MKL, METIS, PARDISO(in MKL) or PARDISO SC for MIC/GPU version
Optional: SCOTCH, MUMPS, CuSolver, VTK, HYPRE interface
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 Espresso Configuration File (ecf) for setting all ESPRESO FEM solver parameters without GUI. Control each parameter in ecf file from command line**
Response time optimization of the USL sensor - nonlinear transient simulation

Heat transfer

![Graph showing response time optimization](image)
Structural mechanics

Elasticity BVP

\[-\text{div}\sigma^{(k)} = f^{(k)} \quad \text{in } \Omega^{(k)}\]
\[\sigma^{(k)}_{ij} = c_{ijkl}\varepsilon^{(k)}_{kl} \quad \text{in } \Omega^{(k)}\]
\[\varepsilon^{(k)} = \frac{1}{2}(\nabla u + \nabla^T u) \quad \text{in } \Omega^{(k)}\]
\[u^{(k)} = 0 \quad \text{on } \Gamma^{(k)}_D\]
\[\sigma^{(k)} n^{(k)} = t^{(k)} \quad \text{on } \Gamma^{(k)}_N\]

Terms on contact boundary

\[-g(X) = u_n(X) - d(X)\]
\[t^m_C \ d\gamma^m_C = -t^s_C \ d\gamma^s_C\]
\[t^s_C = t_n n + t_1 T_1 + t_2 T_2\]

Unilateral contact (non-penetration)
\[u_n - d \leq 0, \ t_n \leq 0, \ t_n(u_n - d) = 0\]

Friction (Tresca or Coulomb)

\[\|t_T\|_2 \leq F, \quad \begin{cases} \|t_T\|_2 < F & \Rightarrow u_T = 0 \\ \|t_T\|_2 = F & \Rightarrow u_T = -ct_T, \ c \geq 0 \end{cases}\]

\[F = \begin{cases} sb & \ldots \text{Tresca} \\ \mathcal{F}|t_n| & \ldots \text{Coulomb} \end{cases}\]
Parallelization by DDM/FETI
## FETI approaches

<table>
<thead>
<tr>
<th></th>
<th>FETI</th>
<th>FETI-DP (partial splitting, nonsingular)</th>
<th>TFETI</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1.</strong></td>
<td>subdomains are fixed or free but with different defects</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2.</strong></td>
<td>FETI-DP</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3.</strong></td>
<td>all subdomains are free with the same defect</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
BETI approach

problem

TFETI/TBETI
Hybrid Total FETI Method

FEM discretization

FETI method

Hybrid FETI method
Hybrid Total FETI Method
In Hybrid Total FETI method the dimension of $GG^T$ is smaller (dim=6) compared to Total FETI (dim=12) due to additional constraints which remove local rigid body modes.
Hybrid Total FETI Method

Implementation

\[ B_c = \begin{pmatrix} B_{c,1} & B_{c,2} & 0 & 0 \\ 0 & 0 & B_{c,3} & B_{c,4} \end{pmatrix}, \quad B = \begin{pmatrix} B_1 & B_2 & B_3 & B_4 \end{pmatrix} \]

additional constraints: duplication of ‘corners’ Lagrange multipliers

augmented KKT system by the matrix \( B_c \) and \( \lambda_c \)

reordering according to clusters
The matrix $B_c$ is a copy of specific rows from the matrix $B$ corresponding to components of $\lambda$ acting on the corners between subdomains 1,2, and 3,4, respectively (red arrows).
Hybrid Total FETI Method

**Modified KKT** system is solvable by the same family of algorithms like FETI (Preconditioned Conjugate Projected Gradient method, ...).

Size of $\tilde{G}\tilde{G}^T$ (or defect of global stiffness matrix) can be **significantly smaller** than in original FETI approach ...

\[
\begin{pmatrix}
\tilde{F} & \tilde{G} \\
\tilde{G}^T & O
\end{pmatrix}
\begin{pmatrix}
\tilde{\lambda} \\
\tilde{\alpha}
\end{pmatrix}
=
\begin{pmatrix}
\tilde{d} \\
\tilde{e}
\end{pmatrix}
\]

\[
\tilde{K} = \text{diag}(\tilde{K}_1, \tilde{K}_2)
\]

\[
\tilde{R}^T = (\tilde{R}_1^T, \tilde{R}_2^T)
\]

\[
\tilde{B} = (\tilde{B}_1, \tilde{B}_2)
\]

\[
\tilde{F} = \tilde{B}\tilde{K} + \tilde{B}^T
\]

\[
\tilde{d} = \tilde{B}\tilde{K}^+\tilde{f} - \tilde{c}
\]

\[
\tilde{G} = -\tilde{B}\tilde{R}
\]

\[
\tilde{e} = -\tilde{R}^T\tilde{f}
\]
Pipelining
Bi-Conjugate Gradients Stabilized (BiCGStab)

Algorithm 4 Standard BiCGStab

1: function BICGSTAB(A, b, x₀)
2: \[ r₀ := b - Ax₀; p₀ := r₀ \]
3: for \( i = 0, \ldots \) do
4: \[ sᵢ := Apᵢ \]
5: \[ \text{compute } \langle r₀, sᵢ \rangle \]
6: \[ αᵢ := \frac{(r₀, rᵢ)}{(r₀, sᵢ)} \]
7: \[ qᵢ := rᵢ - αᵢsᵢ \]
8: \[ yᵢ := Aqᵢ \]
9: \[ \text{compute } \langle qᵢ, yᵢ \rangle ; \langle yᵢ, yᵢ \rangle \]
10: \[ ωᵢ := \frac{(qᵢ, yᵢ)}{(yᵢ, yᵢ)} \]
11: \[ xᵢ₊₁ := xᵢ + αᵢpᵢ + ωᵢqᵢ \]
12: \[ rᵢ₊₁ := qᵢ - ωᵢyᵢ \]
13: \[ \text{compute } \langle r₀, rᵢ₊₁ \rangle \]
14: \[ βᵢ := \frac{(αᵢ/ωᵢ) (r₀, rᵢ₊₁)}{(r₀, rᵢ)} \]
15: \[ pᵢ₊₁ := rᵢ₊₁ + βᵢ (pᵢ - ωᵢsᵢ) \]
16: end for
17: end function

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
Algorithm 6 Pipelined BiCGStab

1: function PIPE-BICGSTAB(A, b, x0)
2:     r0 := b − Ax0; w0 := Ar0; t0 := Aw0;
3:     for i = 0, . . . do
4:         pi := ri + βi−1 (pi−1 − ωi−1 si−1)
5:         si := wi + βi−1 (si−1 − ωi−1 zi−1)
6:         zi := ti + βi−1 (zi−1 − ωi−1 vi−1)
7:         qi := ri − αi si
8:         yi := wi − αi zi
9:         compute (qi, yi) ; (yi, yi)
10:        ωi := (qi, yi) / (yi, yi)
11:        overlap vi := Ai zi
12:        xi+1 := xi + αi pi + ωi qi
13:        r i+1 := qi − ωi yi
14:        wi+1 := yi − ωi (ti − αi vi)
15:        compute (r0, ri+1) ; (r0, wi+1) ; (r0, si) ; (r0, zi)
16:        βi := (αi ωi) (r0, ri+1) / (r0, ri)
17:        αi+1 := (r0, ri+1) / ((r0, wi+1) + βi (r0, si) − βi ωi (r0, zi))
18:        overlap ti+1 := Aw i+1
19:     end for
20: end function

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 axphys
Pipelined BiCGStab vs. Improved BiCGStab


<table>
<thead>
<tr>
<th>Method</th>
<th>GLRED</th>
<th>SPMV</th>
<th>Flops (AXPY + DOT-PROD)</th>
<th>Time (GLRED + SPMV)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>BiCGStab</td>
<td>3</td>
<td>2</td>
<td>20</td>
<td>3 GLRED + 2 SPMV</td>
<td>7</td>
</tr>
<tr>
<td>IBiCGStab</td>
<td>1</td>
<td>2</td>
<td>30</td>
<td>1 GLRED + 2 SPMV</td>
<td>10</td>
</tr>
<tr>
<td>p-BiCGstab</td>
<td>2</td>
<td>2*</td>
<td>38</td>
<td>2 max(GLRED, SPMV)</td>
<td>11</td>
</tr>
</tbody>
</table>

If \( \text{time(GLRED)} \approx \text{time(SPMV)} \):

- speed-up factor p-BiCGStab/BiCGStab = 2.5
- speed-up factor IBiCGStab/BiCGStab = 1.66

If \( \text{time(GLRED)} \gg \text{time(SPMV)} \):

- speed-up factor p-BiCGStab/BiCGStab = 2.5
- speed-up factor IBiCGStab/BiCGStab = 3.0
Robustness and attainable accuracy: p-BiCGStab-rr

Residual replacement every $rr$-th iteration
(non-automated, i.e. $rr$ is a parameter of the method, but chosen large s.t. no. res. repl. is small)

\[
\begin{align*}
    r_i &:= b - Ax_i, \\
    \hat{r}_i &:= M^{-1}r_i, \\
    w_i &:= A\hat{r}_i, \\
    s_i &:= A\hat{p}_i, \\
    \hat{s}_i &:= M^{-1}s_i, \\
    z_i &:= A\hat{s}_i.
\end{align*}
\]

- increased maximal attainable accuracy: comparable to BiCGStab level
- increased robustness: negates instable true residual behaviour
- increased number of iterations possible
ESPRESO FETI

Optimization of global communication for FETI Solvers

![Graph showing optimization of global communication for FETI Solvers](image)
Hybrid FETI vs. FETI preprocessing

Tested on Salomon Supercomputer

# - number of nodes used for solution
3

Accelerators
DOE HPC Facilities Systems

<table>
<thead>
<tr>
<th>Pre-Exascale Systems</th>
<th>Exascale Systems</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td>2021-2022</td>
</tr>
<tr>
<td>2016</td>
<td></td>
</tr>
<tr>
<td>2018</td>
<td></td>
</tr>
<tr>
<td>2020</td>
<td></td>
</tr>
</tbody>
</table>

**Pre-Exascale Systems**

- Argonne IBM BG/Q Unclassified
- Argonne Intel/Cray KNL Unclassified
- ORNL Cray/NVidia K20 Unclassified
- ORNL IBM/NVidia P9/Volta Unclassified
- LLNL IBM BG/Q Classified
- LLNL IBM/NVidia P9/Volta Classified
- LANL/SNL Cray/Intel Xeon/KNL Classified

**Exascale Systems**

- A21
- Argonne Intel/Cray TBD Unclassified
- LBNL TBD
- ORNL TBD Unclassified
- NERSC-9 Unclassified
- Frontier ORNL TBD
- El Capitan LLNL TBD Classified
- Crossroads LANL/SNL TBD Classified
ESPRESO CG in FETI

Projected Conjugate Gradient in FETI

1: \( r_0 := b - A u_0; \quad u_0 := M^{-1} r_0; \quad p_0 := u_0 \)
2: \textbf{for} \( i = 0, \ldots, m - 1 \) \textbf{do}
3: \( s := Ap_i \)
4: \( \alpha := \langle r_i, u_i \rangle / \langle s, p_i \rangle \)
5: \( x_{i+1} := x_i + \alpha p_i \)
6: \( r_{i+1} := r_i - \alpha s \)
7: \( u_{i+1} := M^{-1} r_{i+1} \)
8: \( \beta := \langle r_{i+1}, u_{i+1} \rangle / \langle r_i, u_i \rangle \)
9: \( p_{i+1} := u_{i+1} + \beta p_i \)
10: \textbf{end for}

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

90 – 95% of runtime spent in \( A p_i \)
GPU acceleration of the ESPRESO Solver

0.3 - 300 million DOF Hybrid FETI CG Solver Runtime
Linear elasticity

ORNL Titan 5th in TOP500 LIST

• **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

---

![Graph](image)

Solver runtime [s]

Number of compute nodes / GPUs

- **CPU**
- **GPU - general storage format**

one K20x vs. one AMD Opteron 6274 16-core CPU

**speedUp**

3.4
4

Parallel pre-processing
ESPRESO FEM – pre-processing
Efficient parallel loading of seq. stored unstructured mesh databases

Direct connection with commercial pre-processing tools
Highly parallel I/O of external sequentially stored data with domain decomposition techniques

Jet Engine
- file size: 142 GB
- Nodes: 822 M
- Elements: 484 M
- seq. processing: 7200s
- par. Processing: 13.69 s
GUI and Solver as a Service
Online Solver as a Service (HPC as a Service)
HPC as a Service (HEAppE) Overview

- Providing HPC capabilities as a service to client applications and their users
- Unified interface for different operating systems and schedulers
- Authentication and authorization to provided functions
- Monitoring and reporting of executed jobs and their progress
- Current information about the state of the cluster
Thank you for attention