ESPRESO - ExaScale PaRallel FETI Solver

Hybrid FETI Solver Report

Lubomir Riha, Tomas Brzobohaty

IT4Innovations
Outline

• HFETI theory
  – from FETI to HFETI
  – communication hiding and avoiding techniques

• our new HFETI Solver
  – implementation description
  – performance and scalability

• conclusions
  – work done during stay in FRG
From FETI to HFETI

The matrix $B$ per subdomains:

$$
\begin{bmatrix}
K_1 & 0 & 0 & 0 & B_1^T \\
O & K_2 & 0 & 0 & B_2^T \\
O & 0 & K_3 & 0 & B_3^T \\
O & O & 0 & K_4 & B_4^T \\
B_1 & B_2 & B_3 & B_4 & 0
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
\lambda
\end{bmatrix}
=
\begin{bmatrix}
f_1 \\
f_2 \\
f_3 \\
f_4 \\
c
\end{bmatrix}
$$

Both KKT systems are identical, only matrix $B$ on right side is reordered and divided per rows (with matrix $P$)

$$
P B_i =
\begin{bmatrix}
B_{0,i} \\
B_{1,i}
\end{bmatrix}$$
From FETI to HFETI

\[\begin{pmatrix} K & B^T \\ B & O \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ c \end{pmatrix} \]

\[K = \begin{pmatrix} K_1 & O & O & O \\ O & K_2 & O & O \\ O & O & K_3 & O \\ O & O & O & K_4 \end{pmatrix}\]

Matrix B per subdomains

Both KKT systems are identical, only matrix B on right side is reordered and divided per rows (with matrix P)

\[PB_i = \begin{pmatrix} B_{0,i} \\ B_{1,i} \end{pmatrix}\]
From FETI to HFETI

\[
\begin{pmatrix}
\hat{K}_{12} & O & \hat{B}_{12}^T \\
O & \hat{K}_{34} & \hat{B}_{34}^T \\
\hat{B}_{12} & \hat{B}_{34}^T & O
\end{pmatrix}
\begin{pmatrix}
\hat{u}_{12} \\
\hat{u}_{34} \\
\hat{\lambda}
\end{pmatrix} =
\begin{pmatrix}
\hat{f}_{12} \\
\hat{f}_{34} \\
\hat{c}
\end{pmatrix}
\]

Substitutions

\[
\hat{K}_{12} =
\begin{pmatrix}
K_1 & O & B_{0,1}^T \\
O & K_2 & B_{0,2}^T \\
B_{0,1} & B_{0,2} & O
\end{pmatrix},
\quad
\hat{B}_{12} =
\begin{pmatrix}
B_{1,1} & B_{1,2} & O
\end{pmatrix},
\quad
\hat{u}_{12} =
\begin{pmatrix}
u_1 \\
u_2 \\
\lambda_{0,12}
\end{pmatrix},
\quad
\hat{f}_{12} =
\begin{pmatrix}f_1 \\
f_2 \\
c_{12}
\end{pmatrix}
\]

\[
\hat{K}_{34} =
\begin{pmatrix}
K_3 & O & B_{0,3}^T \\
O & K_4 & B_{0,4}^T \\
B_{0,3} & B_{0,4} & O
\end{pmatrix},
\quad
\hat{B}_{34} =
\begin{pmatrix}
B_{1,3} & B_{1,4} & O
\end{pmatrix},
\quad
\hat{u}_{34} =
\begin{pmatrix}
u_3 \\
u_4 \\
\lambda_{0,34}
\end{pmatrix},
\quad
\hat{f}_{34} =
\begin{pmatrix}f_3 \\
f_4 \\
f_{34}
\end{pmatrix}
\]

\[
\tilde{\lambda} = \lambda_1, \quad \tilde{c} = c_1
\]
From FETI to HFETI

4 independently floating subdomains

\[
\begin{pmatrix}
K_1 & O & O & O \\
O & K_2 & O & O \\
O & O & K_3 & O \\
O & O & O & K_4 \\
B_1 & B_2 & B_3 & B_4
\end{pmatrix}
\begin{pmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
\lambda
\end{pmatrix}
= 
\begin{pmatrix}
f_1 \\
f_2 \\
f_3 \\
f_4 \\
c
\end{pmatrix}
\]

\[\text{2 independently floating subdomains}
\text{equivalent with FETI-DP approach, subdomains 1 and 2 (3 and 4) are connected per corners nodes)\]

\[
\begin{pmatrix}
\tilde{K}_{12} & O & \tilde{B}_{12}^T \\
O & \tilde{K}_{34} & \tilde{B}_{34}^T \\
\tilde{B}_{12} & \tilde{B}_{34}^T & O
\end{pmatrix}
\begin{pmatrix}
\tilde{u}_{12} \\
\tilde{u}_{34} \\
\lambda
\end{pmatrix}
= 
\begin{pmatrix}
\tilde{f}_{12} \\
\tilde{f}_{34} \\
\tilde{c}
\end{pmatrix}
\]
From FETI to HFETI

Kernels of augmented stiffness matrices

\[
\begin{pmatrix}
\tilde{K}_{12} & O & \tilde{B}_{12}^T \\
O & \tilde{K}_{34} & \tilde{B}_{34}^T \\
\tilde{B}_{12} & \tilde{B}_{34}^T & O
\end{pmatrix}
\begin{pmatrix}
\tilde{u}_{12} \\
\tilde{u}_{34} \\
\tilde{\lambda}
\end{pmatrix}
= 
\begin{pmatrix}
\tilde{f}_{12} \\
\tilde{f}_{34} \\
\tilde{c}
\end{pmatrix}
\Rightarrow 
\begin{pmatrix}
\tilde{R}_{12} & O \\
O & \tilde{R}_{34}
\end{pmatrix}
\]

Kernel of one cluster is assembled from kernels of its all subdomains. All subdomains' kernels have to be written from one coordinate system (requirement of the compatibility).
HFETI Matrix $G$ (and $GG^T$)

$$\tilde{G}^T = -\tilde{B} \tilde{R}$$

$$\tilde{B} = \begin{pmatrix} B_1 & B_2 & O & B_3 & B_4 & O \end{pmatrix}$$

$$\tilde{R} = \begin{pmatrix} R_1 & O \\ R_2 & O \\ O & O \\ O & O \end{pmatrix}$$

$$G^T = -(B_1 R_1 + B_2 R_2 \quad B_3 R_3 + B_4 R_4)$$

Size of $GG^T$ is given by defect of global stiffness matrix

$$GG^T = \begin{pmatrix} R_1^T B_1^T + R_2^T B_2^T \\ R_3^T B_3^T + R_4^T B_4^T \end{pmatrix} \begin{pmatrix} B_1 R_1 + B_2 R_2 \\ B_3 R_3 + B_4 R_4 \end{pmatrix}$$

Compare $GG^T$ assembled for classical TFETI
HFETI $\tilde{x} = \tilde{K}^+ \tilde{b}$ for i-th cluster

$$
\begin{bmatrix}
K_1 & O & B_{0,1}^T \\
O & K_2 & B_{0,2}^T \\
B_{0,1} & B_{0,2} & O
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\mu
\end{bmatrix}
=
\begin{bmatrix}
b_1 \\
b_2 \\
d
\end{bmatrix}
\quad
\begin{bmatrix}
K & B_0^T \\
B_0 & O
\end{bmatrix}
\begin{bmatrix}
x \\
\mu
\end{bmatrix}
=
\begin{bmatrix}
b \\
d
\end{bmatrix}
$$

\[
\begin{bmatrix}
F_0 & G_0^T \\
G_0 & O
\end{bmatrix}
\begin{bmatrix}
\mu \\
\beta
\end{bmatrix}
=
\begin{bmatrix}
g_0 \\
e_0
\end{bmatrix}
\]

\[F_0 = B_0 K^+ B_0^T \]
\[G_0 = -R_0^T B_0^T \]
\[g_0 = B_0 K^+ f_0 - c_0 \]
\[e_0 = -R_0^T f_0^T \]
\[\beta_0 = (G_0 F_0^{-1} G_0^T)^+ (G_0 F_0^{-1} g_0 - e_0) \]
\[\mu = F_0^{-1} (g_0 - G_0^T \beta_0) \]
\[x = K^+ (b - B_0^T \mu) + R_0 \beta \]
Our new TFETI and HFETI Solver

• Implementation in C++
  – based on Intel MKL
    • sparse and dense BLAS routines
    • PARDISO direct sparse solver
  – compiled with Intel Compiler and Intel MPI

• Parallelization tools and strategies
  – hybrid parallelization for multi-socket, multicore compute nodes
    • enables over subscription of CPU cores – each core can process multiple sub-domains
  – distributed memory parallelization – MPI
    • use of MPI 3.0 standard features (Intel MPI 5.0 Beta)
      – non-blocking global operations (MPI_IallRedeuce, ...) for global DOT product
  – shared memory parallelization – using Intel Cilk+
    • enables parallel reduction using custom reduce operations with C++ objects
    • conversion to OpenMP is planned using OpenMP standard 4.0
HFETI Solver – development stages

• Stage 1
  – optimization of global communication of FETI solver
    • using communication hiding and avoiding techniques
      – pipelined CG solvers
        » single global reduction
      – solving coarse problem using distributed inverse matrix
        » single global AllGather operation (instead of MPI_Gather and MPI_Scatter)
        » parallel solve – each subdomain has only 6 dense rows of \((GG^T)^{-1}\)
      – nearest neighbor communication optimization – multiplication with gluing matrix
        » \(B\) for FETI and \(B1\) for HFETI (next slide)
        » fully scalable with growing number of compute nodes

• Stage 2
  – optimization of inter cluster processing
  – efficient solution of inner coarse problem

• Stage 3
  – combine both together
Reduction of global communication in FETI and HFETI
Hiding latencies in Krylov solvers

Preconditioned Conjugate Gradient

1: \( r_0 := b - Ax_0; u_0 := M^{-1}r_0; p_0 := u_0 \)
2: for \( i = 0, \ldots, m - 1 \) 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: end for

Chronopoulos/Gear CG

Only one global reduction each iteration.

1: \( r_0 := b - Ax_0; u_0 := M^{-1}r_0; w_0 := Au_0 \)
2: \( \alpha_0 := \langle r_0, u_0 \rangle / \langle w_0, u_0 \rangle; \beta := 0; \gamma_0 := \langle r_0, u_0 \rangle \)
3: for \( i = 0, \ldots, m - 1 \) do
4: \( p_i := u_i + \beta_i p_{i-1} \)
5: \( s_i := w_i + \beta_i s_{i-1} \)
6: \( x_{i+1} := x_i + \alpha p_i \)
7: \( r_{i+1} := r_i - \alpha s_i \)
8: \( u_{i+1} := M^{-1}r_{i+1} \)
9: \( w_{i+1} := Au_{i+1} \)
10: \( \gamma_{i+1} := \langle r_{i+1}, u_{i+1} \rangle \)
11: \( \delta := \langle w_{i+1}, u_{i+1} \rangle \)
12: \( \beta_{i+1} := \gamma_{i+1} / \gamma_i \)
13: \( \alpha_{i+1} := \gamma_{i+1} (\delta - \beta_{i+1} \gamma_{i+1} / \alpha_i) \)
14: end for

Sparse Matrix-Vector product
- Only communication with neighbors
- Good scaling

Dot-product
- Global communication
- Scales as \( \log(P) \)
- Scalar vector multiplication, vector-vector addition
- No communication
Hiding latencies in Krylov solvers

**Chronopoulos/Gear CG**

Only one global reduction each iteration.

1: $r_0 := b - A x_0$; $u_0 := M^{-1} r_0$; $w_0 := A u_0$
2: $\alpha_0 := \langle r_0, u_0 \rangle / \langle w_0, u_0 \rangle$; $\beta := 0$; $\gamma_0 := \langle r_0, u_0 \rangle$
3: for $i = 0, \ldots, m - 1$ do
4: $\rho_i := u_i + \beta_i \rho_{i-1}$
5: $s_i := w_i + \beta_i s_{i-1}$
6: $x_{i+1} := x_i + \alpha_i \rho_i$
7: $r_{i+1} := r_i - \alpha_i s_i$
8: $u_{i+1} := M^{-1} r_{i+1}$
9: $w_{i+1} := A u_{i+1}$
10: $\gamma_{i+1} := \langle r_{i+1}, u_{i+1} \rangle$
11: $\delta := \langle w_{i+1}, u_{i+1} \rangle$
12: $\beta_{i+1} := \gamma_{i+1} / \gamma_i$
13: $\alpha_{i+1} := \gamma_{i+1} / (\delta - \beta_{i+1} \gamma_{i+1} / \alpha_i)$
14: end for

**Preconditioned pipelined CG**

1: $r_0 := b - A x_0$; $u_0 := M^{-1} r_0$; $w_0 := A u_0$
2: for $i = 0, \ldots$ do
3: $\gamma_i := \langle r_i, u_i \rangle$
4: $\delta := \langle w_i, u_i \rangle$
5: $m_i := M^{-1} w_i$
6: $n_i := A m_i$
7: if $i > 0$ then
8: $\beta_i := \gamma_i / \gamma_{i-1}$; $\alpha_i := \gamma_i / (\delta - \beta_i \gamma_i / \alpha_{i-1})$
9: else
10: $\beta_i := 0$; $\alpha_i := \gamma_i / \delta$
11: end if
12: $z_i := n_i + \beta_i z_{i-1}$
13: $q_i := m_i + \beta_i q_{i-1}$
14: $s_i := w_i + \beta_i s_{i-1}$
15: $p_i := u_i + \beta_i p_{i-1}$
16: $x_{i+1} := x_i + \alpha_i p_i$
17: $r_{i+1} := r_i - \alpha_i s_i$
18: $u_{i+1} := u_i - \alpha_i q_i$
19: $w_{i+1} := w_i - \alpha_i z_i$
20: end for

**Sparse Matrix-Vector product**
- Only communication with neighbors
- Good scaling

**Dot-product**
- Global communication
- Scales as log$(P)$
- Scalar vector multiplication, vector-vector addition
- No communication
Hiding latencies in Krylov solvers: Cost model

- $G :=$ time for a global reduction
- $\text{SpMV} :=$ time for a sparse-matrix vector product
- $\text{PC} :=$ time for preconditioner application
- local work such as AXPY is neglected

<table>
<thead>
<tr>
<th></th>
<th>flops</th>
<th>time (excl, AXPYS, DOTs)</th>
<th>#glob syncs</th>
<th>memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>10</td>
<td>$2G + \text{SpMV} + \text{PC}$</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Chro/Geapipe-CG</td>
<td>12</td>
<td>$G + \text{SpMV} + \text{PC}$</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>$\max(G, \text{SpMV} + \text{PC})$</td>
<td>1</td>
<td>9</td>
</tr>
</tbody>
</table>

Reduction of global communication in FETI and HFETI

• MPI processes iterate only over $\lambda$s “they need” for local processing
  • in case of FETI – the $\lambda$s required for multiplication with B matrix of a subdomain
  • in case of HFETI – the $\lambda$s required for multiplication with B1 matrices of all subdomains per cluster

• global update of vector $\lambda$
  • becomes only nearest neighbor type of communication
  • scales with number of MPI processes (subdomains)
HFETI Solver - Benchmark

- 3D CUBE - elasticity
  - FETI
    - $H^c$ - FETI decomposition into sub-domains
    - $(h^c)^3$ – number of elements per subdomain
  - HFETI
    - $H^c$ – decomposition into clusters
    - $H^S$ – cluster decomposition into sub-domains
      - $(h^S)^3$ – number of elements per sub-domain
Stage 1: optimization of global communication for FETI solver

FETI iteration time (measured on Anselm)

- **CG algorithm with 2 reductions** – is a general version of the CG algorithm
- **CG algorithm with 1 reduction** – is based on Preconditioned Pipelined CG algorithm, where projector is used in place of preconditioner (this algorithm is ready to use non-blocking global reduction for further performance improvements (comes in Intel MPI 5.0))
- **GGTINV** – parallelizes the solve of coarse problem plus merges two Gather and Scatter global operations into single AllGather
- $5^3 = 3*(5+1)^3 = 648$ DOFs – small domain size is chosen to identify all communication bottlenecks of the solver
Engine Benchmark – Superlinear Strong Scaling with FETI

- 2.5 millions of DOFs
- decomposed into: 34, 64, 128, 256, 512 and 1024 subdomains
- FETI solver
- measured on Anselm
  - 128 nodes with 16 CPU cores (2x8)
  - 8 subdomains per node
- ..
Engine Benchmark –
Superlinear Strong Scaling with FETI

Engine 2.5 millions - FETI - iteration time

- REGCG - LUMPED - GGTINV
- PIPECG - NOPREC - GGTINV
- REGCG - LUMPED - NOGGTINV
- PIPECG - NOPREC - NOGGTINV
- Linear strong scaling (based on 32)

Anselm - 8 subdomains per node for 2x8 CPU cores
Engine Benchmark – Strong Scaling

Engine 2.5 Milions - FETI - iterations

Number of iterations [-] vs Number of subdomains [-]

- REGCG - LUMPED - GGTINV
- PIPEG - NOPREC - GGTINV
- REGCG - LUMPED - NOGGTINV
- PIPEG - NOPREC - NOGGTINV
FETI - time per iteration for various utilization of compute node:

Anselm: 8, 15 and 16 MPI processes per node

Since solver is memory bounded using 8 CPU cores per node gives each process faster access to the memory and significantly improves the performance.
Large Scale Benchmarks

• 3D cube - elasticity benchmark
  – highly parallel benchmark generator scales up to thousands of sub-domains
  – large sub-domains 120,000 to 160,000 DOFs
    • sparse direct solver takes most of the memory and processing time (over 99%)

• Tested on Total FETI method only
  – shows the bottleneck of the method
    – the coarse problem that will be reduced by using total Hybrid FETI method

• No preconditioner
  – will be implemented in near future

Visualization of displacement
TFETI - Scalability of iterative solver
TFETI - Scalability of iterative solver

FETI - 3D cube elasticity benchmark - Time per iteration - SurfSara
Cartesius

- Iteration time [s]
- Linear model (Iteration time [s])

Number of subdomains [-]

<table>
<thead>
<tr>
<th>Number of subdomains [-]</th>
<th>DOFs</th>
</tr>
</thead>
<tbody>
<tr>
<td>72,412,707</td>
<td>48,582,831</td>
</tr>
<tr>
<td>141,137,643</td>
<td>102,984,375</td>
</tr>
<tr>
<td>243,548,211</td>
<td>309,485,127</td>
</tr>
<tr>
<td>386,362,875</td>
<td>386,362,875</td>
</tr>
<tr>
<td>475,021,263</td>
<td>576,300,099</td>
</tr>
<tr>
<td>691,039,191</td>
<td>691,039,191</td>
</tr>
<tr>
<td>820,078,347</td>
<td>964,257,375</td>
</tr>
</tbody>
</table>
TFETI - Large Scale Benchmarks

FETI - 3D cube elasticity benchmark - coarse problem time, solution time and number of iterations - SurfSara Cartesius

SurfSara Cartesius
- BULLx system with 64 GB of RAM per node and 24 CPU cores (2 x 12 cores)
- benchmark used all 24 cores per node – 2.6 GB of RAM per subdomain
TFETI - Large Scale Benchmarks

Preprocessing, number of iterations and solution time

Solution and preprocessing time [s]

Problem size [DOFs]
TFETI - Large Scale Benchmarks

Coarse problem preprocessing time, $K$ factorization time, solution time and number of iterations - subdomains size 192000 DOFs

- $K$ factorization
- GGT time [s] - $39^3$
- Solution time [s] - $39^3$

Number of iterations:
- 49.22
- 49.75
TFETI - Large Scale Benchmarks

Coarse problem preprocessing time, K factorization time, solution time and number of iterations - subdomains size 206763 DOFs

- K factorization
- GGT time [s] - 40^3
- Solution time [s] - 40^3

<table>
<thead>
<tr>
<th></th>
<th>1,124,416,083</th>
<th>1,322,133,243</th>
<th>1,541,767,203</th>
</tr>
</thead>
<tbody>
<tr>
<td>K factorization</td>
<td>50</td>
<td>50</td>
<td>51</td>
</tr>
<tr>
<td>GGT time [s]</td>
<td>103.10</td>
<td>125.82</td>
<td>148.87</td>
</tr>
<tr>
<td>Solution time</td>
<td>66.44</td>
<td>66.48</td>
<td>66.46</td>
</tr>
<tr>
<td>Number of iterations</td>
<td>48.70</td>
<td>48.96</td>
<td>50.55</td>
</tr>
</tbody>
</table>
HFETI Solver – development stages

• Stage 1
  – optimization of global communication of FETI solver
    • using communication hiding and avoiding techniques
      – pipelined CG solvers
        » single global reduction
      – solving coarse problem using distributed inverse matrix
        » single global AllGather operation (instead of MPI_Gather and MPI_Scatter)
        » parallel solve – each subdomain has only 6 dense rows of \((GG^T)^{-1}\)
      – nearest neighbor communication optimization – multiplication with gluing matrix
        » \(B\) for FETI and \(B_1\) for HFETI (next slide)
        » fully scalable with growing number of compute nodes

• Stage 2
  – optimization of inter cluster processing
  – efficient solution of inner coarse problem

• Stage 3
  – combine both together
Stage 2: Inter Cluster Processing

- It is important to select correct cluster decomposition
  - affects both preprocessing and solution time

<table>
<thead>
<tr>
<th>domain size</th>
<th>subdomains</th>
<th>size</th>
<th>avg</th>
<th>sum</th>
<th>prec</th>
<th>iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>343</td>
<td>128625</td>
<td>0.123941</td>
<td>5.949182</td>
<td>75.503219</td>
<td>48</td>
</tr>
<tr>
<td>5</td>
<td>216</td>
<td>139968</td>
<td>0.072914</td>
<td>3.718598</td>
<td>37.732576</td>
<td>51</td>
</tr>
<tr>
<td>6</td>
<td>125</td>
<td>128625</td>
<td>0.040325</td>
<td>2.217873</td>
<td>16.381135</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
<td>139968</td>
<td>0.030154</td>
<td>1.718762</td>
<td>10.04497</td>
<td>57</td>
</tr>
<tr>
<td>11</td>
<td>27</td>
<td>139968</td>
<td>0.034425</td>
<td>1.893399</td>
<td>7.490033</td>
<td>55</td>
</tr>
<tr>
<td>17</td>
<td>8</td>
<td>139968</td>
<td>0.064372</td>
<td>3.347334</td>
<td>7.876874</td>
<td>52</td>
</tr>
</tbody>
</table>

Anselm
Problem size: 
~120 000
Optimal decomposition:
dom. size: 3*(8+1)^3 = 2187
cluster size: 4^3 = 64

<table>
<thead>
<tr>
<th>domain size</th>
<th>subdomains</th>
<th>size</th>
<th>avg</th>
<th>sum</th>
<th>prec</th>
<th>iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>512</td>
<td>331776</td>
<td>0.309891</td>
<td>17.973677</td>
<td>338.035839</td>
<td>58</td>
</tr>
<tr>
<td>6</td>
<td>343</td>
<td>352947</td>
<td>0.198848</td>
<td>12.328554</td>
<td>180.041968</td>
<td>62</td>
</tr>
<tr>
<td>7</td>
<td>216</td>
<td>331776</td>
<td>0.118965</td>
<td>7.613787</td>
<td>90.402162</td>
<td>64</td>
</tr>
<tr>
<td>11</td>
<td>64</td>
<td>331776</td>
<td>0.080177</td>
<td>5.13133</td>
<td>27.456335</td>
<td>64</td>
</tr>
<tr>
<td>15</td>
<td>27</td>
<td>331776</td>
<td>0.095755</td>
<td>6.03256</td>
<td>19.761467</td>
<td>63</td>
</tr>
<tr>
<td>23</td>
<td>8</td>
<td>331776</td>
<td>0.156146</td>
<td>8.431877</td>
<td>29.455252</td>
<td>54</td>
</tr>
</tbody>
</table>

Anselm
Problem size: 
~330 000
Optimal decomposition:
dom. size: 3*(11+1)^3 = 5184
cluster size: 4^3 = 64

Note: domains size ‘N’ is in number of elements per edge of the cube – real size is equal to 3*(N+1)^3
Note: avg – average iteration time [s]; sum – total time of all iterations = solution time; prec – cluster preprocessing time; iter – number of iterations;
HFETI Solver – development stages

• Stage 1
  – optimization of global communication of FETI solver
    • using communication hiding and avoiding techniques
      – pipelined CG solvers
        » single global reduction
      – solving coarse problem using distributed inverse matrix
        » single global AllGather operation (instead of MPI_Gather and MPI_Scatter)
        » parallel solve – each subdomain has only 6 dense rows of \( (GGT)^{-1} \)
      – nearest neighbor communication optimization – multiplication with gluing matrix
        » \( B \) for FETI and \( B_1 \) for HFETI (next slide)
        » fully scalable with growing number of compute nodes

• Stage 2
  – optimization of inter cluster processing
  – efficient solution of inner coarse problem

• Stage 3
  – combine both together
Stage 3 – HFETI vs FETI

HFETI vs FETI – one iteration time (measured on Anselm)

Weak scaling – domain size is fixed; cluster size is fixed; number of domains goes from 1 to 2000
- domain size $5^3 = 3 \times (5+1)^3 = 648$
- cluster size is 16 domains (1 cluster per node; each node has 16 cores - two 8-core CPUs)
HFETI vs FETI - Performance

HFETI vs FETI – one iteration time (measured on Anselm)

Weak scaling – domain size is fixed; cluster size is fixed; number of domains goes from 1 to 2000
- domain size $5^3 = 3*(5+1)^3 = 648$
- cluster size is 16 (1 cluster per node; each node has 16 cores - two 8-core CPUs)
- efficient communication algorithms helps both FETI and HFETI
  - CG with 1 reduction + coarse problem solved using distributed inverse matrix (GGTINV)
HFETI vs FETI - Performance

HFETI vs FETI - one iteration time (measured on Cartesius)

- cluster size is 8 – (3 clusters per node; each node has 24 cores - two 12-core CPUs)
- comparison of (1) FETI and (2) HFETI, where both FETI a HFETI uses CG with 1 reduction and coarse problem is solved using distributed inverse matrix of GGᵀ
Conclusions

The solver has the following main features:

• support for Total FETI and Hybrid FETI domain decomposition iterative methods

• support for discretization of static and dynamic linear elasticity problems
  – assembly of the stiffness/mass matrices of unstructured meshes using tetra4, prism6, pyramid5, brick8, tetra10, prism13, pyramid15, and brick20 elements

• lumped local subdomain pre-conditioner

• Implementation of the main iterative solver using the following Krylov solvers
  – regular Conjugate gradient method (CG) with or without the lumped pre-conditioner
  – pipelined CG with or without the lumped pre-conditioner

• coarse problem processing
  – solved sequentially on a master node using the LU decomposition
  – solved in parallel using a distributed inverse matrix

• support for Intel MIC many-core architecture
  – entire solver can run on a (1) single accelerator or a (2) cluster of accelerators using Intel MPI