Parallel Preconditioning Methods for Ill-conditioned Problems

Kengo Nakajima

Information Technology Center, The University of Tokyo

2014 Conference on Advanced Topics and Auto Tuning in High Performance Scientific Computing (2014 ATAT in HPSC)

March 14-15, 2014
National Taiwan University, Taipei, Taiwan
• Comm./Synch. Avoiding/Reducing Algorithms
  – Direct/Iterative Solvers, Preconditioning, s-step method
  – Coarse Grid Solvers on Parallel Multigrid

• Preconditioning Methods on Manycore Architectures
  – SPAI, Polynomials, ILU with Multicoloring/RCM, (no CM-RCM), Geometric MG
  – Asynchronous ILU on Manycore Arch. (E.Chow (Ga.Tech.)) ?

• Preconditioning Methods for Ill-Conditioned Problems
  – Low-Rank Approximation

• Block-Structured AMR
  – SFC: Space-Filling Curve
TOC

• SIAM PP14
• Ill-conditioned Problems
• Hetero 3D, BILU \((p,d,t)\)
• Summary & Future Works
Large-scale Simulations by Parallel FEM Procedures

- Unstructured grid with irregular data structure
- Large-scale sparse matrices
- Preconditioned parallel iterative solvers
- “Real-world” ill-conditioned problems
What are ill-conditioned problems?

- Various ill-conditioned problems
  - For example, matrices derived from coupled NS equations are ill-conditioned even if meshes are uniform.
- We have been focusing on 3D solid mechanics applications with:
  - heterogeneity
  - Contact B.C.
  - BILU/BIC
- Ideas can be extended to other fields.
Ill-Conditioned Problems

Heterogeneous Fields, Distorted Meshes
Contact Problems in Simulations of Earthquake Generation Cycle
Preconditioning Methods (of Krylov Iterative Solvers) for Real-World Applications

- are the most critical issues in scientific computing
- are based on
  - Global Information: condition number, matrix properties etc.
  - Local Information: properties of elements (shape, size …)
- require knowledge of
  - background physics
  - applications
Technical Issues of “Parallel” Preconditioners in FEM

- Block Jacobi type Localized Preconditioners
- Simple problems can easily converge by simple preconditioners with excellent parallel efficiency.
- Difficult (ill-conditioned) problems cannot easily converge
  - Effect of domain decomposition on convergence is significant, especially for ill-conditioned problems.
  - Block Jacobi-type localized preconditioners
  - More domains, more iterations
  - There are some remedies (e.g. deep fill-ins, deep overlapping), but they are not efficient.
  - ASDD does not work well for really ill-conditioned problems.
Technical Issues of “Parallel” Preconditioners for Iterative Solvers

- If domain boundaries are on “stronger” elements, convergence is very bad.
Remedies: Domain Decomposition

- Avoid “Strong Elements”
  - not practical
- Extended Depth of Overlapped Elements
  - Selective Fill-ins, Selective Overlapping [KN 2007]
    - adaptive preconditioning/domain decomposition methods which utilize features of FEM procedures
- PHIDAL/HID (Hierarchical Interface Decomposition) [Henon & Saad 2007]
- Extended HID [KN 2010]
Extension of Depth of Overlapping

Cost for computation and communication may increase

●: Internal Nodes, ■: External Nodes
■: Overlapped Elements
HID: Hierarchical Interface Decomposition [Henon & Saad 2007]

- Multilevel Domain Decomposition
  - Extension of Nested Dissection
- Non-overlapping at each level: Connectors, Separators
- Suitable for Parallel Preconditioning Method
Parallel ILU for each Connector at each LEVEL

• The unknowns are reordered according to their **level** numbers, from the lowest to highest.

• The block structure of the reordered matrix leads to natural parallelism if ILU/IC decompositions or forward/backward substitution processes are applied.
Results: 64 cores
Contact Problems
BILU(p)-(depth of overlapping)
3,090,903 DOF
Development of robust and efficient parallel preconditioning method

Construction of strategies for optimum selection of preconditioners, partitioning, and related methods/parameters.

By utilization of both of:
  - global information obtained from derived coefficient matrices
  - very local information, such as information of each mesh in finite-element applications.

Final goal of my recent work in this area after 2000
• SIAM PP14
• Ill-conditioned Problems
• Hetero 3D, BILU \((p,d,t)\)
• Summary & Future Works
Hetero 3D (1/2)

- Parallel FEM Code (Flat MPI)
  - 3D linear elasticity problems in cube geometries with heterogeneity
  - SPD matrices
  - Young’s modulus: $10^{-6}$–$10^{+6}$
    - $(E_{\text{min}} - E_{\text{max}})$: controls condition number

- Preconditioned Iterative Solvers
  - GP-BiCG [Zhang 1997]
  - BILUT($p,d,t$)

- Domain Decomposition
  - Localized Block-Jacobi with Extended Overlapping (LBJ)
  - HID/Extended HID
Hetero 3D (2/2)

- based on the framework for parallel FEM proc. of GeoFEM
  - Benchmark developed in **FP3C** project under Japan-France collaboration

- Parallel Mesh Generation
  - Fully parallel way
    - each process generates local mesh, and assembles local matrices.
  - Total number of vertices in each direction \((N_x, N_y, N_z)\)
  - Number of partitions in each direction \((P_x, P_y, P_z)\)
  - Number of total MPI processes is equal to \(P_x \times P_y \times P_z\)
  - Each MPI process has \((N_x/P_x) \times (N_y/P_y) \times (N_z/P_z)\) vertices.
  - Spatial distribution of Young’s modulus is given by an external file, which includes information for heterogeneity for the field of \(128^3\) cube geometry.
    - If \(N_x\) (or \(N_y\) or \(N_z\)) is larger than 128, distribution of these \(128^3\) cubes is repeated periodically in each direction.
BILUT\((p,d,t)\)

- Incomplete LU factorization with threshold (ILUT)
- ILUT\((p,d,t)\) [KN 2010]
  - \(p\): Maximum fill-level specified before factorization
  - \(d, t\): Criteria for dropping tolerance before/after factorization
- The process (b) can be substituted by other factorization methods or more powerful direct linear solvers, such as MUMPS, SuperLU and etc.

![Diagram](chart.png)
Preliminary Results

• Hardware
  – 16-240 nodes (160-3,840 cores) of Fujitsu PRIMEHPC FX10 (Oakleaf-FX), University of Tokyo

• Problem Setting
  – $420 \times 320 \times 240$ vertices ($3.194 \times 10^7$ elem’s, $9.677 \times 10^7$ DOF)
  – Strong scaling
  – Effect of thickness of overlapped zones
    • BILUT($p,d,t$)-LBJ-X (X=1,2,3)
  – Effect of $d$ is small
  – HID is slightly more robust than LBJ
BILUT($p,0,0$) at 3,840 cores

NO dropping: Effect of Fill-in

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>NNZ of $[M]$</th>
<th>Set-up (sec.)</th>
<th>Solver (sec.)</th>
<th>Total (sec.)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>BILUT(1,0,0)-LBJ-1</td>
<td>$1.920 \times 10^{10}$</td>
<td>1.35</td>
<td>65.2</td>
<td>66.5</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(1,0,0)-LBJ-2</td>
<td>$2.519 \times 10^{10}$</td>
<td>2.03</td>
<td>61.8</td>
<td>63.9</td>
<td>1288</td>
</tr>
<tr>
<td>BILUT(1,0,0)-LBJ-3</td>
<td>$3.197 \times 10^{10}$</td>
<td>2.79</td>
<td>74.0</td>
<td>76.8</td>
<td>1367</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-1</td>
<td>$3.351 \times 10^{10}$</td>
<td>3.09</td>
<td>71.8</td>
<td>74.9</td>
<td>1339</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-2</td>
<td>$4.394 \times 10^{10}$</td>
<td>4.39</td>
<td>65.2</td>
<td>69.6</td>
<td>939</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-3</td>
<td>$5.631 \times 10^{10}$</td>
<td>5.95</td>
<td>83.6</td>
<td>89.6</td>
<td>1006</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-1</td>
<td>$6.468 \times 10^{10}$</td>
<td>9.34</td>
<td>105.2</td>
<td>114.6</td>
<td>1192</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-2</td>
<td>$8.523 \times 10^{10}$</td>
<td>12.7</td>
<td>98.4</td>
<td>111.1</td>
<td>823</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-3</td>
<td>$1.101 \times 10^{11}$</td>
<td>17.3</td>
<td>101.6</td>
<td>118.9</td>
<td>722</td>
</tr>
<tr>
<td>BILUT(1,0,0)-HID</td>
<td>$1.636 \times 10^{10}$</td>
<td>2.24</td>
<td>60.7</td>
<td>62.9</td>
<td>1472</td>
</tr>
<tr>
<td>BILUT(2,0,0)-HID</td>
<td>$2.980 \times 10^{10}$</td>
<td>5.04</td>
<td>66.2</td>
<td>71.7</td>
<td>1096</td>
</tr>
</tbody>
</table>

[NNZ] of $[A]$：$7.174 \times 10^9$
BILUT($p,0,0$) at 3,840 cores
NO dropping: Effect of Overlapping

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>NNZ of $[M]$</th>
<th>Set-up (sec.)</th>
<th>Solver (sec.)</th>
<th>Total (sec.)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>BILUT(1,0,0)-LBJ-1</td>
<td>$1.920\times10^{10}$</td>
<td>1.35</td>
<td>65.2</td>
<td>66.5</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(1,0,0)-LBJ-2</td>
<td>$2.519\times10^{10}$</td>
<td>2.03</td>
<td>61.8</td>
<td>63.9</td>
<td>1288</td>
</tr>
<tr>
<td>BILUT(1,0,0)-LBJ-3</td>
<td>$3.197\times10^{10}$</td>
<td>2.79</td>
<td>74.0</td>
<td>76.8</td>
<td>1367</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-1</td>
<td>$3.351\times10^{10}$</td>
<td>3.09</td>
<td>71.8</td>
<td>74.9</td>
<td>1339</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-2</td>
<td>$4.394\times10^{10}$</td>
<td>4.39</td>
<td>65.2</td>
<td>69.6</td>
<td>939</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-3</td>
<td>$5.631\times10^{10}$</td>
<td>5.95</td>
<td>83.6</td>
<td>89.6</td>
<td>1006</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-1</td>
<td>$6.468\times10^{10}$</td>
<td>9.34</td>
<td>105.2</td>
<td>114.6</td>
<td>1192</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-2</td>
<td>$8.523\times10^{10}$</td>
<td>12.7</td>
<td>98.4</td>
<td>111.1</td>
<td>823</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-3</td>
<td>$1.101\times10^{11}$</td>
<td>17.3</td>
<td>101.6</td>
<td>118.9</td>
<td>722</td>
</tr>
<tr>
<td>BILUT(1,0,0)-HID</td>
<td>$1.636\times10^{10}$</td>
<td>2.24</td>
<td>60.7</td>
<td>62.9</td>
<td>1472</td>
</tr>
<tr>
<td>BILUT(2,0,0)-HID</td>
<td>$2.980\times10^{10}$</td>
<td>5.04</td>
<td>66.2</td>
<td>71.7</td>
<td>1096</td>
</tr>
</tbody>
</table>

$[\text{NNZ}]$ of $[A]$: $7.174\times10^9$
BILUT($p,0,0$) at 3,840 cores
NO dropping

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>NNZ of [M]</th>
<th>Set-up (sec.)</th>
<th>Solver (sec.)</th>
<th>Total (sec.)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>BILUT(1,0,0)-LBJ-1</td>
<td>$1.920 \times 10^{10}$</td>
<td>1.35</td>
<td>65.2</td>
<td>66.5</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(1,0,0)-LBJ-2</td>
<td>$2.519 \times 10^{10}$</td>
<td>2.03</td>
<td>61.8</td>
<td>63.9</td>
<td>1288</td>
</tr>
<tr>
<td>BILUT(1,0,0)-LBJ-3</td>
<td>$3.197 \times 10^{10}$</td>
<td>2.79</td>
<td>74.0</td>
<td>76.8</td>
<td>1367</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-1</td>
<td>$3.351 \times 10^{10}$</td>
<td>3.09</td>
<td>71.8</td>
<td>74.9</td>
<td>1339</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-2</td>
<td>$4.394 \times 10^{10}$</td>
<td>4.39</td>
<td>65.2</td>
<td>69.6</td>
<td>939</td>
</tr>
<tr>
<td>BILUT(2,0,0)-LBJ-3</td>
<td>$5.631 \times 10^{10}$</td>
<td>5.95</td>
<td>83.6</td>
<td>89.6</td>
<td>1006</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-1</td>
<td>$6.468 \times 10^{10}$</td>
<td>9.34</td>
<td>105.2</td>
<td>114.6</td>
<td>1192</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-2</td>
<td>$8.523 \times 10^{10}$</td>
<td>12.7</td>
<td>98.4</td>
<td>111.1</td>
<td>823</td>
</tr>
<tr>
<td>BILUT(3,0,0)-LBJ-3</td>
<td>$1.101 \times 10^{11}$</td>
<td>17.3</td>
<td>101.6</td>
<td>118.9</td>
<td>722</td>
</tr>
<tr>
<td>BILUT(1,0,0)-HID</td>
<td>$1.636 \times 10^{10}$</td>
<td>2.24</td>
<td>60.7</td>
<td>62.9</td>
<td>1472</td>
</tr>
<tr>
<td>BILUT(2,0,0)-HID</td>
<td>$2.980 \times 10^{10}$</td>
<td>5.04</td>
<td>66.2</td>
<td>71.7</td>
<td>1096</td>
</tr>
</tbody>
</table>

[NNZ] of [A]: $7.174 \times 10^9$
### Optimum Value of $t$

**BILUT($p,0,t$) at 3,840 cores**

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>NNZ of $[M]$</th>
<th>Set-up (sec.)</th>
<th>Solver (sec.)</th>
<th>Total (sec.)</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>BILUT(1,0,2.75×10^{-2})-LBJ-1</td>
<td>7.755×10^9</td>
<td>1.36</td>
<td>2.05</td>
<td>45.0</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(1,0,2.75×10^{-2})-LBJ-2</td>
<td>1.019×10^{10}</td>
<td>2.81</td>
<td>3.11</td>
<td>42.2</td>
<td>1383</td>
</tr>
<tr>
<td>BILUT(1,0,2.75×10^{-2})-LBJ-3</td>
<td>1.285×10^{10}</td>
<td>3.11</td>
<td>3.11</td>
<td>42.2</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(2,0,1.00×10^{-2})-LBJ-1</td>
<td>1.118×10^{10}</td>
<td>2.81</td>
<td>3.11</td>
<td>42.2</td>
<td>1383</td>
</tr>
<tr>
<td>BILUT(2,0,1.00×10^{-2})-LBJ-2</td>
<td>1.487×10^{10}</td>
<td>3.11</td>
<td>3.11</td>
<td>42.2</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(2,0,1.00×10^{-2})-LBJ-3</td>
<td>1.893×10^{10}</td>
<td>3.11</td>
<td>3.11</td>
<td>42.2</td>
<td>1383</td>
</tr>
<tr>
<td>BILUT(3,0,2.50×10^{-2})-LBJ-1</td>
<td>8.072×10^{9}</td>
<td>3.11</td>
<td>3.11</td>
<td>42.2</td>
<td>1383</td>
</tr>
<tr>
<td>BILUT(3,0,2.50×10^{-2})-LBJ-2</td>
<td>1.063×10^{10}</td>
<td>3.11</td>
<td>3.11</td>
<td>42.2</td>
<td>1916</td>
</tr>
<tr>
<td>BILUT(3,0,2.50×10^{-2})-LBJ-3</td>
<td>1.342×10^{10}</td>
<td>3.11</td>
<td>3.11</td>
<td>42.2</td>
<td>1383</td>
</tr>
<tr>
<td>BILUT(1,0,2.75×10^{-2})-HID</td>
<td>6.850×10^{9}</td>
<td>2.25</td>
<td>3.11</td>
<td>42.2</td>
<td>1383</td>
</tr>
<tr>
<td>BILUT(2,0,1.00×10^{-2})-HID</td>
<td>1.030×10^{10}</td>
<td>2.25</td>
<td>3.11</td>
<td>42.2</td>
<td>1916</td>
</tr>
</tbody>
</table>

Strong Scaling up to 3,840 cores
according to elapsed computation time (set-up+solver) for BILUT(1,0,2.5\times10^{-2})-HID with 256 cores
SIAM PP14

• Ill-conditioned Problems
• Hetero 3D, BILU $(p,d,t)$

Summary & Future Works
Summary

• Hetero 3D
• Generally speaking, HID is more robust than LBJ with overlap extention
• BILUT\((p,d,t)\)
  – effect of \(d\) is not significant
  – \([\text{NNZ}]\) of \([M]\) depends on \(t\) (not \(p\))
  – BILU\((3,0,t_0)\) > BILU\((2,0,t_0)\) > BILU\((1,0,t_0)\), although cost of a single iteration is similar for each method
• Critical/optimum value of \(t\)
  – \([\text{NNZ}]\) of \([M]\) = \([\text{NNZ}]\) of \([A]\)
  – Further investigation needed.
Future Works

• Theoretical/numerical investigation of optimum $t$
  – Eigenvalue analysis etc.
  – Final Goal: Automatic selection BEFORE computation
  – (Any related work?)

• Further investigation/development of LBJ & HID

• Comparison with other preconditioners/direct solvers
  – (Various types of) Low-Rank Approximation Methods
  – Collaboration with MUMPS team in IRIT/Toulouse
    • They are testing their LRA method for final meeting in Paris next week 😊

• Hetero 3D will be released as a deliverable of FP3C project soon
  – OpenMP/MPI Hybrid version
    • BILU(0) is already done, factorization is (was) the problem
  – Extension to Manycore/GPU clusters
Reordering for extracting parallelism in each domain

- Krylov Iterative Solvers
  - Dot Products
  - SMVP
  - DAXPY
  - Preconditioning

- IC/ILU Factorization, Forward/Backward Substitution
  - Global Dependency
  - Reordering needed for parallelism
  - Multicoloring (MC), RCM, CM-RCM ([Washio & Doi 1999], [KN 2003])
Ordering Methods

Elements in “same color” are independent: to be parallelized

Talk by Y.Saad’s group in SIAM PP14

- **MC**: Good parallel efficiency with smaller # of colors, bad convergence. Better convergence with many colors, synch. overhead
- **RCM**: Good convergence, poor parallel efficiency, synch. overhead
- **CM-RCM**: Reasonable convergence & efficiency
(●: MC, △: RCM, -: CM-RCM)

- Heterogenous Poisson Equations
- FVM, $100^3$ cells
- ICCG Solver
- OpenMP with 16-threads on a single node of Fujitsu FX10
- CM-RCM/RCM are more robust than MC
- Optimum Color # depends on HW
Effects of colors are different according to thread #, H/W etc. 128³


Fujitsu FX10
16 cores, 16 threads
Effect of HW barrier (?)

Intel KNC
57 cores, 228 threads
ELL: Fixed Loop-length, Nice for Pre-fetching

\[
\begin{bmatrix}
1 & 3 & 0 & 0 & 0 \\
1 & 2 & 5 & 0 & 0 \\
4 & 1 & 3 & 0 & 0 \\
0 & 3 & 7 & 4 & 0 \\
1 & 0 & 0 & 0 & 5 \\
\end{bmatrix}
\]

(a) CRS  

(b) ELL