Parallel Preconditioning Methods for Contact Problems with FEM

Kengo Nakajima
The 21st Century Earth Science COE Program
Department of Earth & Planetary Science
The University of Tokyo & CREST/JST
Overview

• Background
  – Simulations for Earthquake Generation Cycle
  – Selective Blocking

• More General Problems
  – Extension of Overlapped Zones
  – Target Application

• Preconditioning/Partitioning Methods
  – Selective Fill-in
  – Selective Extension of Overlapped Zones

• Results

• Summary
  – Future Works
Contact Problems in Simulations of Earthquake Generation Cycle

- Quasi-static stress accumulation process at plate boundaries
- Non-linear contact problems with Newton-Raphson iter's
- Ill-conditioned problem due to penalty constraint by ALM (Augmented Lagrangean).
- Parallel FEM with domain decomposition
Contact Problems in Simulations of Earthquake Generation Cycle (cont.)

• Assumptions
  – Infinitesimal deformation, static contact relationship.
    • Location of nodes is in each "contact pair" is identical.
    • “Consistent” node number and position
  – No friction : Symmetric coefficient matrix

• Large-scale problems
  – Parallel preconditioned iterative solvers

• Special preconditioning : Selective Blocking.
  – provides robust and smooth convergence in 3D solid mechanics simulations for geophysics with contact.
Selective Blocking [Nakajima, 2001]

Special Method for Contact Problem

Strongly coupled nodes are put into the same diagonal block.

Full LU factorization for each block.

\[ 2\lambda u_{x0} = \lambda u_{x1} + \lambda u_{x2} \]
\[ 2\lambda u_{y0} = \lambda u_{y1} + \lambda u_{y2} \]
\[ 2\lambda u_{z0} = \lambda u_{z1} + \lambda u_{z2} \]

3 nodes form 1 selective block.

\[ \lambda u_{x0} = \lambda u_{x1} \]
\[ \lambda u_{y0} = \lambda u_{y1} \]
\[ \lambda u_{z0} = \lambda u_{z1} \]

2 nodes form 1 selective block.
Convergence is slow if nodes in each contact group locate on different partition.

Repartitioning so that nodes in contact pairs would be in same partition as INTERIOR nodes will be effective.
Results on Hitachi SR2201 (U.Tokyo)  
Parallel Performance of SB-BIC(0)-CG  
2,471,439 DOF, 784,000 Elements, $\frac{\lambda}{E}=10^6$  
Iterations/CPU time until convergence ($\varepsilon=10^{-8}$)

<table>
<thead>
<tr>
<th>Preconditioning</th>
<th>16 PEs</th>
<th>32 PEs</th>
<th>48 PEs</th>
<th>64 PEs</th>
<th>96 PEs</th>
<th>144 PEs</th>
<th>192 PEs</th>
<th>256 PEs</th>
<th>Memory Size (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIC(0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.10</td>
</tr>
<tr>
<td>Iterations</td>
<td>14459</td>
<td>14583</td>
<td>15018</td>
<td>15321</td>
<td>15523</td>
<td>15820</td>
<td>16084</td>
<td>16267</td>
<td></td>
</tr>
<tr>
<td>sec.</td>
<td>13500</td>
<td>7170</td>
<td>4810</td>
<td>3630</td>
<td>2410</td>
<td>1630</td>
<td>1270</td>
<td>1230</td>
<td></td>
</tr>
<tr>
<td>Speed-up</td>
<td>16</td>
<td>30</td>
<td>45</td>
<td>60</td>
<td>90</td>
<td>133</td>
<td>170</td>
<td>211</td>
<td></td>
</tr>
<tr>
<td>BIC(1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8.39</td>
</tr>
<tr>
<td>Iterations</td>
<td>N/A</td>
<td>N/A</td>
<td>236</td>
<td>175</td>
<td>119</td>
<td>81</td>
<td>62</td>
<td>48</td>
<td></td>
</tr>
<tr>
<td>sec.</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speed-up</td>
<td>48</td>
<td>65</td>
<td>95</td>
<td>140</td>
<td>183</td>
<td>236</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BIC(2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14.4</td>
</tr>
<tr>
<td>Iterations</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>212</td>
<td>140</td>
<td>112</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>sec.</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speed-up</td>
<td>96</td>
<td>145</td>
<td>182</td>
<td>217</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB-BIC(0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3.52</td>
</tr>
<tr>
<td>Iterations</td>
<td>511</td>
<td>524</td>
<td>527</td>
<td>538</td>
<td>543</td>
<td>567</td>
<td>569</td>
<td>584</td>
<td></td>
</tr>
<tr>
<td>sec.</td>
<td>555</td>
<td>295</td>
<td>193</td>
<td>144</td>
<td>96</td>
<td>64</td>
<td>48</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td>Speed-up</td>
<td>16</td>
<td>30</td>
<td>46</td>
<td>62</td>
<td>92</td>
<td>139</td>
<td>185</td>
<td>235</td>
<td></td>
</tr>
</tbody>
</table>
Results on Hitachi SR2201 (U.Tokyo) Parallel Performance of SB-BIC(0)-CG, $\lambda/E=10^6$
More General Problems

• Moving boundaries due to large slip conditions
• Inconsistent node number (and location) at boundary surfaces
  – Assembly structure for machine parts.
    • where meshes for each part are separately generated.
  – Commercial FEM codes (e.g. ABAQUS, NASTRAN) can treat problems for this type of “inconsistent” cases. (single PE, direct method for linear equations).
More General Problems
Inconsistent Number of Nodes at Boundary Surfaces

• Difficult to apply "selective blocking"
  – Size of each "selective block" may be too large for full LU factorization

• Difficult to apply "special partitioning"

• Remedy
  – Higher-order fill-in’s
  – Extension of overlapped zones for parallel computing
Extension of Overlapped Zones

Cost for computation and communication may increase

●: Internal Nodes, ●: External Nodes
■: Overlapped Elements
Effect of Extended Overlapped Zones

- [Nakajima, 2005]
  - BILU(0,1,2)
  - for “consistent” node number cases

<table>
<thead>
<tr>
<th>Preconditioning</th>
<th>partitioning (overlap #)</th>
<th>PE #</th>
<th>iter’s solve(sec.)</th>
<th>set-up+ solve(sec.)</th>
<th>parallel speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>SB-BILU(0)</td>
<td>special [3] 1-layer</td>
<td>16</td>
<td>386</td>
<td>506.2</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>410</td>
<td>63.9</td>
<td>126.7</td>
</tr>
<tr>
<td>BILU(1)</td>
<td>special [3] 1-layer</td>
<td>16</td>
<td>225</td>
<td>563.2</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>247</td>
<td>95.0</td>
<td>94.8</td>
</tr>
<tr>
<td>BILU(1)</td>
<td>regular 1-layers</td>
<td>16</td>
<td>444</td>
<td>1033.2</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>529</td>
<td>191.0</td>
<td>86.6</td>
</tr>
<tr>
<td>BILU(1)</td>
<td>regular 2-layers</td>
<td>16</td>
<td>405</td>
<td>1063.3</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>430</td>
<td>204.6</td>
<td>83.2</td>
</tr>
<tr>
<td>SPAI</td>
<td>regular 2-layers</td>
<td>16</td>
<td>891</td>
<td>626.3</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128</td>
<td>888</td>
<td>105.1</td>
<td>95.4</td>
</tr>
</tbody>
</table>
Example for “Inconsistent” Cases
This model simulates contact problem in assembly structure

- Each block is discretized into cubic tri-linear elements
  - elastic material: $E = 1.00$, Poisson ration $= 0.30$
- Each block is connected through elastic truss elements generated on each node on contact surfaces.
  - Truss elements are crossing.
Example for “Inconsistent” Cases
This model simulates contact problem in assembly structure

- Elastic coefficient of truss elements is set to $10^3$-$10^4$ times as large as that of solid elements.
  - This condition simulates constraint boundary conditions for contact.
- Distributed uniform force at $z=z_{\text{max}}$ surface
  - $u=0@x=0$, $v=0@y=0$, $w=0@z=0$
Summary of Problem Setting

• Problem Size
  – Small: 32,000 elements (except truss’s) 111,132 DOF
  – Large: 864,000 elements (except truss’s), 2,723,772 DOF

• Preconditioned BiCGSTAB/GPBiCG
  – Localized preconditioning (block Jacobi type)
    • BSSOR, BILU(0,1,2), Selective Fill-in
  – Ordering
    • Cuthill-McKee(CM)

• Partitioning
  – Recursive Coordinate Bisection (RCB): 8~64
    • (Selective) Extension of Overlapped Zones

• Environment
  – 64core AMD Opteron(275 2.2GHz), F90+MPI
Selective Fill-in

• Apply higher order of fill-in’s between nodes which connect to truss-type elements.
  – Similar concept as “selective blocking”

• In this work:
  – BILU(2) for these special nodes
  – BILU(1) for general nodes

• Cost is similar to that of BILU(1), but effect of preconditioning is expected to be competitive with that of BILU(2).
Idea of “Selective Fill-in”

- 2nd order fill-in’s are considered for these nodes
- 2nd order fill-in’s are NOT considered for these nodes
- 2nd order fill-in’s are NOT considered for these nodes
Internal Nodes for Partitioning

- Internal Nodes

Domain Boundary
One-Layer Overlapping

- Internal Nodes
- External Nodes
- Overlapped Elements

This is the general configuration of local data set for parallel FEM (one-layer of overlapping).
Extension of Overlapped Zones (2-layers)

- Internal Nodes
- External Nodes
- Overlapped Elements
Extension of Overlapped Zones (2-layers)

● Internal Nodes
● External Nodes
■ Overlapped Elements

Selective Extension
“Delayed” extension for elements which do not include nodes connected to truss-type elements
Extension of Overlapped Zones (3-layers)

- Internal Nodes
- External Nodes
- Overlapped Elements

Selective Extension
Reduced cost for computations and communications

delayed delayed
Small Case (1PE): Memory, Iterations
111,132 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$
Small Case (1PE): Set-Up + Solver

111,132 DOF, \( \lambda=10^3 \), \( \varepsilon=10^{-8} \)

![Bar chart comparing BSSOR, BILU(0), BILU(1), BILU(2), and S-Fill-in methods in terms of seconds (sec.)]

- BSSOR
- BILU(0)
- BILU(1)
- BILU(2)
- S-Fill-in

Legend:
- BiCGSTAB
- GPBiCG
Small Case (1PE): Set-Up + Solver

GPBiCG, 111,132 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$

![Bar chart showing set-up and solver times for different methods: BSSOR, BILU(0), BILU(1), BILU(2), S-Fill-in.](chart.png)
Large Case: Number of Iterations

2,723,772 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$
GPBiCG, Depth of Overlapping=2, Normal Extension

![Bar chart showing number of iterations for different values of PE#]
## Normal vs. Selective Extension: 64 PEs

<table>
<thead>
<tr>
<th>Overlap</th>
<th>Normal Extension</th>
<th>Selective Extension</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td>18498</td>
<td>23179</td>
</tr>
<tr>
<td>Elements</td>
<td>16360</td>
<td>22874</td>
</tr>
<tr>
<td>Ext. Nodes</td>
<td>4312</td>
<td>8993</td>
</tr>
<tr>
<td>Nodes</td>
<td>20793</td>
<td>28657</td>
</tr>
<tr>
<td>Elements</td>
<td>18498</td>
<td>28451</td>
</tr>
<tr>
<td>Ext. Nodes</td>
<td>6607</td>
<td>14471</td>
</tr>
<tr>
<td>Nodes</td>
<td>23260</td>
<td>34883</td>
</tr>
<tr>
<td>Elements</td>
<td>20793</td>
<td>34741</td>
</tr>
<tr>
<td>Ext. Nodes</td>
<td>9074</td>
<td>20697</td>
</tr>
</tbody>
</table>
Large Case: Number of Iterations
Effect of Depth of Overlapping

2,723,772 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$
GPBiCG, 64 PEs

- BILU(0)
- BILU(1)
- BILU(2)
- S-Fill-in

[Graphs showing iterations vs. depth of overlapping for Normal Extension and Selective Extension]
Large Case: Set-up + Solver
Effect of Depth of Overlapping

2,723,772 DOF, $\lambda=10^3$, $\varepsilon=10^{-8}$
GPBiCG, 64 PEs

Normal Extension

Selective Extension
Summary

• Preconditioning method for contact problems with inconsistent node number on boundary surface
  – “Selective Fill-in”
  – “Selective Extension for Overlapped Zones” for Parallel Computing
  – After all, problem specific preconditioning method, not general

• Number of iterations for convergence increases, as number of PEs is increasing
  – Much more effects compared to “consistent” cases
Future Works

• Matrices obtained from distributed FEM models
  – similar to those obtained by “global” nested dissection ordering
  – treatment of overlapped zone

• Different partitioning/local data distribution for finite-element models and coefficient matrices, will be effective.
Global reordering method for finding independent sets in distributed data sets is to be developed

• Fully global ordering for $10^9$ unknowns with $10^4$ processors with multicolor reordering is not realistic.
  – Too many communications/synchronization overhead

• Some hierarchical approach should be useful.