Reducing Complexity in Algebraic Multigrid

Hans De Sterck
Department of Applied Mathematics
University of Waterloo

Ulrike Yang
Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
Outline

- introduction: AMG
- complexity growth when using classical coarsenings
- Parallel Modified Independent Set (PMIS) coarsening
- scaling results
- improving interpolation
- conclusions and future work
Introduction

- solve \( Au = f \)
- \( A \) from 3D PDE – sparse!
- large problems (10^9 dof) - parallel
- unstructured grid problems
Algebraic Multigrid (AMG)

- multi-level
- iterative
- algebraic: suitable for unstructured!
## AMG building blocks

### Setup Phase:

- **Select coarse “grids”**
- **Define interpolation,** \( P^{(m)}, \ m = 1,2,... \)
- **Define restriction and coarse-grid operators**

\[
R^{(m)} = P^{(m)T} \quad A^{(m+1)} = P^{(m)T} A^{(m)} P^{(m)}
\]

### Solve Phase

\[
\text{Relax } A^{(m)} u^m = f^m
\]

- **Compute** \( r^m = f^m - A^{(m)} u^m \)
- **Restrict** \( r^{m+1} = R^{(m)} r^m \)
- **Solve** \( A^{(m+1)} e^{m+1} = r^{m+1} \)
- **Interpolate** \( e^m = P^{(m)} e^{m+1} \)
- **Correct** \( u^m \leftarrow u^m + e^m \)

---

**IMACS 2005**
AMG complexity - scalability

- **Operator complexity** \( C_{op} = \frac{\sum_i \text{nonzeros}(A_i)}{\text{nonzeros}(A_0)} \)

  e.g., 3D, ideally: \( C_{op} = 1 + 1/8 + 1/64 + \ldots < 8/7 \)

  *measure of memory use, and work in solve phase*

- **Scalable algorithm:**

  \( O(n) \) operations per V-cycle \( (C_{op} \text{ bounded}) \)

  AND

  *number of V-cycles independent of \( n \) \( (\rho_{AMG} \text{ independent of } n) \)
AMG coarsening and interpolation

- large $a_{ij}$, ‘strong connections’ are important
- define strength matrix $S$:

$$A = \begin{bmatrix}
  x & x & x & x \\
  x & x & x & x \\
  x & x & x & x \\
  x & x & x & x \\
\end{bmatrix} \quad \text{and} \quad S = \begin{bmatrix}
  1 & 1 & 0 \\
  1 & 0 & 1 \\
  0 & 0 & 1 & 1 \\
  1 & 0 & 1 \\
\end{bmatrix}$$

- consider the undirected graph of $S$
- apply parallel maximal independent set algorithm to graph($S$) [Luby, 1986]
classical AMG coarsening (CLJP)

- **(C1)** Maximal Independent Set:
  Independent: no two **C**-points are connected
  Maximal: if one more **C**-point is added, the independence is lost

- **(C2)** All **F**-**F** connections require connections to a common **C**-point (for good interpolation)
  **F-points** have to be changed into **C-points**, to ensure **(C2)**; **(C1)** is violated
  more **C-points**, higher complexity
Classical coarsening: scalability results

- example: finite difference Laplacian, parallel CLJP coarsening algorithm

- 2D (5-point): near-optimal scalability (250^2 dof/proc)

<table>
<thead>
<tr>
<th>Procs</th>
<th>C_{op}</th>
<th>t_{tot}</th>
<th>Iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>4.48</td>
<td>2.89</td>
<td>9</td>
</tr>
<tr>
<td>64</td>
<td>4.50</td>
<td>3.85</td>
<td>9</td>
</tr>
<tr>
<td>256</td>
<td>4.50</td>
<td>5.01</td>
<td>9</td>
</tr>
</tbody>
</table>
Classical coarsening: complexity growth in some cases

- 3D (7-point): complexity growth

<table>
<thead>
<tr>
<th>dof</th>
<th>$C_{op}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$32^3$</td>
<td>16.17</td>
</tr>
<tr>
<td>$64^3$</td>
<td>22.51</td>
</tr>
</tbody>
</table>
Classical coarsening: complexity growth in some cases

- **4D (9-point), 5D (11-point): complexity growth!!**

<table>
<thead>
<tr>
<th></th>
<th>dof</th>
<th>( C_{op} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>4D</td>
<td>( 20^4 )</td>
<td>127.5</td>
</tr>
<tr>
<td>5D</td>
<td>( 9^5 )</td>
<td>256.9</td>
</tr>
</tbody>
</table>

- **excessive memory use**

*(results obtained by Jeff Butler)*
our approach to reduce complexity

- *do not add C points for strong F-F connections that do not have a common C point*

- *less C points, reduced complexity, but worse convergence factors expected*

- *compensate by GMRES acceleration*
PMIS: select 1

- select C-pts with maximal measure locally
- make neighbour F-pts
- remove neighbour edges
PMIS: remove and update 1

- select C-pts with maximal measure locally
- make neighbours F-pts
- remove neighbour edges
PMIS: select 2

- select C-pts with maximal measure locally
- make neighbours F-pts
- remove neighbour edges
PMIS: remove and update 2

- select C-pts with maximal measure locally
- make neighbours F-pts
- remove neighbour edges
PMIS: final grid

- select C-pts with maximal measure locally
- make neighbour F-pts
- remove neighbour edges
- parallel algorithm
PMIS coarsening: reduce complexity

- finite difference Laplacian \((CLJP-PMIS+GMRES)\)

<table>
<thead>
<tr>
<th>dof</th>
<th>(C_{op})</th>
<th>(t_{tot})</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>(120^2)</td>
<td>4.16</td>
</tr>
<tr>
<td></td>
<td>(120^2)</td>
<td>1.90</td>
</tr>
<tr>
<td>3D</td>
<td>(100^3)</td>
<td>25.94</td>
</tr>
<tr>
<td></td>
<td>(100^3)</td>
<td>2.36</td>
</tr>
<tr>
<td>4D</td>
<td>(20^4)</td>
<td>127.5</td>
</tr>
<tr>
<td></td>
<td>(20^4)</td>
<td>2.95</td>
</tr>
<tr>
<td>5D</td>
<td>(9^5)</td>
<td>256.9</td>
</tr>
<tr>
<td></td>
<td>(8^5)</td>
<td>3.14</td>
</tr>
</tbody>
</table>
Parallel PMIS results: 7-point finite difference Laplacian in 3D, $40^3$ dof per proc

CLJP and PMIS-GMRES(10)

<table>
<thead>
<tr>
<th>proc</th>
<th>$C_{op}$</th>
<th>Iter</th>
<th>$t_{total}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.39</td>
<td>6</td>
<td>3.35</td>
</tr>
<tr>
<td>512</td>
<td>17.02</td>
<td>10</td>
<td>35.83</td>
</tr>
<tr>
<td>1331</td>
<td>17.19</td>
<td>10</td>
<td>46.25</td>
</tr>
<tr>
<td>1</td>
<td>2.32</td>
<td>13</td>
<td>1.28</td>
</tr>
<tr>
<td>512</td>
<td>2.37</td>
<td>25</td>
<td>12.77</td>
</tr>
<tr>
<td>1331</td>
<td>2.37</td>
<td>28</td>
<td>17.99</td>
</tr>
</tbody>
</table>
Convergence problems on PMIS-coarsened grids

- PMIS coarsening works well for many problems
- for some problems, too many iterations are necessary because interpolation is not accurate enough (“not enough C-points”)
- one solution: add C-points (CLJP…)
- other solution: use distance-two C-points for interpolation = long-range interpolation
  - Stueben’s multipass interpolation
  - F-F interpolation
Convergence problems

- 3D elliptic PDE with jumps in coefficient $a$

$$\left( au_x \right)_x + \left( au_y \right)_y + \left( au_z \right)_z = 1$$

- 1000 processors, $40^3$ dof/proc

<table>
<thead>
<tr>
<th></th>
<th>$t_{tot}$</th>
<th>$C_{op}$</th>
<th>Iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLJP</td>
<td>52.48</td>
<td>17.00</td>
<td>17</td>
</tr>
<tr>
<td>PMIS</td>
<td>211.79</td>
<td>2.40</td>
<td>686</td>
</tr>
</tbody>
</table>

- remedy: improve interpolation used with PMIS
F-F interpolation

- when strong F-F connection without a common C-point is detected, do not add C-point, but extend interpolation stencil to distance-two C-points
- no C-points added, but larger interpolation stencils
results using long-range interpolation

- 3D elliptic PDE with jumps in coefficient \( a \)

\[
(au_x)_x + (au_y)_y + (au_z)_z = 1
\]

- 1 processor, AMG+GMRES, \( 80^3 \) dof

<table>
<thead>
<tr>
<th>Method</th>
<th>( t_{\text{tot}} )</th>
<th>( C_{\text{op}} )</th>
<th>Iter</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLJP</td>
<td>48.0</td>
<td>21.54</td>
<td>7</td>
</tr>
<tr>
<td>PMIS</td>
<td>94.6</td>
<td>2.46</td>
<td>188</td>
</tr>
<tr>
<td>PMIS + mp</td>
<td>13.7</td>
<td>2.47</td>
<td>21</td>
</tr>
<tr>
<td>PMIS + F-F</td>
<td>21.4</td>
<td>4.90</td>
<td>9</td>
</tr>
</tbody>
</table>
Conclusions and future work

- **PMIS leads to reduced, scalable complexities for large multi-D problems on parallel computers**

- **for difficult problems, nearest-neighbour interpolation is not sufficient on PMIS grids**

- **long-range interpolation improves convergence**

- **multipass and F-F interpolations were studied**
# Top 500 Supercomputer list (November 2004)

<table>
<thead>
<tr>
<th>Rank</th>
<th>Site</th>
<th>Computer</th>
<th>Country</th>
<th>TeraFlops</th>
<th>Processors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lawrence Livermore National Laboratory</td>
<td>IBM BlueGene/L</td>
<td>US</td>
<td>135</td>
<td>65,536</td>
</tr>
<tr>
<td>2</td>
<td>NASA/Ames Research Center/NAS</td>
<td>SGI Altix</td>
<td>US</td>
<td>51</td>
<td>10,160</td>
</tr>
<tr>
<td>3</td>
<td>The Earth Simulator Center</td>
<td>NEC Earth-Simulator</td>
<td>Japan</td>
<td>35</td>
<td>5,120</td>
</tr>
<tr>
<td>4</td>
<td>Barcelona Supercomputer Center</td>
<td>IBM eServer</td>
<td>Spain</td>
<td>20</td>
<td>3,564</td>
</tr>
<tr>
<td>5</td>
<td>Lawrence Livermore National Laboratory</td>
<td>Intel Itanium2</td>
<td>US</td>
<td>19</td>
<td>4,096</td>
</tr>
<tr>
<td>6</td>
<td>Los Alamos National Laboratory</td>
<td>ASCI Q - HP AlphaServer</td>
<td>US</td>
<td>13</td>
<td>8,192</td>
</tr>
<tr>
<td>7</td>
<td>Virginia Tech</td>
<td>1100 Dual 2.3 GHz Apple XServe</td>
<td>US</td>
<td>12</td>
<td>2,200</td>
</tr>
<tr>
<td>8</td>
<td>IBM - Rochester</td>
<td>IBM BlueGene/L</td>
<td>US</td>
<td>11</td>
<td>8,192</td>
</tr>
<tr>
<td>9</td>
<td>Naval Oceanographic Office</td>
<td>IBM eServer</td>
<td>US</td>
<td>10</td>
<td>2,944</td>
</tr>
<tr>
<td>10</td>
<td>NCSA</td>
<td>Dell P4 Xeon</td>
<td>US</td>
<td>10</td>
<td>2,500</td>
</tr>
<tr>
<td>11</td>
<td>ECMWF</td>
<td>IBM eServer</td>
<td>UK</td>
<td>10</td>
<td>2,176</td>
</tr>
<tr>
<td>12</td>
<td>ECMWF</td>
<td>IBM eServer</td>
<td>UK</td>
<td>10</td>
<td>2,176</td>
</tr>
<tr>
<td>17</td>
<td>Shanghai Supercomputer Center</td>
<td>Dawning 4000A, Opteron</td>
<td>China</td>
<td>8</td>
<td>2,560</td>
</tr>
<tr>
<td>18</td>
<td>Los Alamos National Laboratory</td>
<td>LNX Opteron</td>
<td>US</td>
<td>8</td>
<td>2,816</td>
</tr>
<tr>
<td>19</td>
<td>Lawrence Livermore National Laboratory</td>
<td>MCR Linux Cluster Xeon</td>
<td>US</td>
<td>8</td>
<td>2,304</td>
</tr>
<tr>
<td>20</td>
<td>Lawrence Livermore National Laboratory</td>
<td>ASCI White, IBM SP Power3</td>
<td>US</td>
<td>7</td>
<td>8,192</td>
</tr>
</tbody>
</table>

- **scalable results were presented for MCR (#19), 2,000 procs**
- **next target: Blue Gene/L (#1), 65,000 procs (will be 130,000 soon)**
our code currently runs on LLNL Blue Gene/L

one preliminary result, on 8,000 processors:
PMIS works, CLJP runs out of memory...

more tests to follow...

scalability up to 130,000 processors?