UPC Applications Overview

Parry Husbands
17 May 2004
Overview

• Our mission
  - Help improve Berkeley UPC Environment
    - Stress translator and runtime
    - Determine performance bottlenecks (if any)
  - Outreach
    - Showcase features of UPC
  - Continue explorations in PGAS languages
    - Validate design decisions
    - Question others
    - Suggest extensions
Our environment at scale

• NAS Multigrid Class D (1024 x 1024 x 1024)
• Written using 1-sided style

<table>
<thead>
<tr>
<th>Threads</th>
<th>Mflop/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>35,510</td>
</tr>
<tr>
<td>256</td>
<td>63,552</td>
</tr>
<tr>
<td>512</td>
<td>116,211</td>
</tr>
<tr>
<td>1024</td>
<td>202,141</td>
</tr>
</tbody>
</table>

Lemieux @ PSC
Alpha/Quadrics
Sparse Matrix – Vector Multiplication

- Compute $y = A \times x$ for distributed sparse matrix $A$
- Key component of solvers
- Irregular – threads require variable number of elements of $x$

\[
\begin{array}{ccc}
X & X & X & X \\
X & X & X & X \\
\end{array}
\times
\begin{array}{c}
X \\
X & X & X & X \\
\end{array}
= 
\begin{array}{c}
\hline \\
\end{array}
\]
SPMV using memcpy extensions

• Implemented using non-blocking indexed gets

\[ \text{spmv}(A, x) \]

// Compute Ax for sparse A
non-blocking get of remote values of x
computation of part of result using local values of x
sync
computation of rest of result using remote values of x

• Significantly simplified code, particularly setup
• Not tuned yet, but shows promise
Outreach: Parallel Triangulation

• First step in many physical simulations
• Lots of dynamic data movement (~ 4K LOC + Triangle)
Improving the Delaunay Triangulation Code

- Message coalescing performed manually
  - Eventually automatic
- Teams library further optimized and extended
  - Now includes a distributed vector facility
- Fast redistribution code implemented

Time for 1 million points in a sphere
HP UPC
Alpha/Quadrics@MTU

<table>
<thead>
<tr>
<th>Threads</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.6</td>
</tr>
<tr>
<td>2</td>
<td>12.61</td>
</tr>
<tr>
<td>4</td>
<td>7.50</td>
</tr>
<tr>
<td>8</td>
<td>5.19</td>
</tr>
</tbody>
</table>
The effort continues…

- Message coalescing in practice
- Further improvements to Triangulation code
- More examples using the memcpy extensions
  - Performance tuning
- Programs using collectives
  - Are they the “right” interface?
- More involvement in language specification