RELAP5-3D Architectural Developments in 2004

Dr George L Mesina

2004 RELAP5 International Users Seminar

August 25-27, 2004
Outline

• Motivation for architectural improvements
• Completion of parallel conversion
• Completion of PVM
• Vectorization work
• Conversion to Fortran 90
• Summary
Motivation for Architectural Improvements

- Longevity
- Modernization
- Simplification
  - Reduced development time
  - Reduced maintenance costs
  - Faster debugging
  - Increased reliability
- Code run speed
Completion of Parallel Conversion

- Reason for conversion to OpenMP directives
  - Older form of parallel uses direct calls to the KAI library which is no longer maintained.
  - OpenMP is available on all major vendor platforms
    - Built into many compilers
- Previous work
  - Convert parallel in hydrodynamic subprograms
  - Convert parallel in heat structure subprograms
- Current task: Convert parallel in neutron kinetics
Parallel Kinetics Task Summary

- Kinetics parallelized via domain decomposition
  - Axial levels are grouped into subdomains
    - One parallel thread for each subdomain
      - Algorithm causes different answers when the # of subdomains or the grouping changes.
- All calls to KAI library calls were converted to OpenMP
- Environmental variable that activates/deactivates parallel for kinetics
- Almost all test cases all run properly
  - Problem with Cartesian, Krylov with 64-bit integers
Parallel Maintenance & Development

- Parallel is difficult to develop and maintain and it is easy to break.
- A parallel tool was developed during this task to help develop and maintain parallel.
  - By itself, this is not sufficient to maintain parallel.
- Training and study is needed to work on parallel code.
- Tutorial on OpenMP parallel programming will be taught later in the seminar.
  - All RELAP5-3D code developers should attend.
RELAP5-3D Vectorization Task

• A recent trend in HPC is a return to vector computing
  – Federal government initiative
• INEEL is vectorizing its codes for use on its Cray SV1 computers
  – INEEL funded task to improve RELAP5-3D vector performance.
• Performance analysis of RELAP5-3D
  – neither PHANTV nor PHANTJ vectorizes.
  – Both are in top 5 in execution time percentage.
PHANTV and PHANTJ analysis

- PHANTV & PHANTJ: have huge loops that do not vectorize:
  - DO-11, DO-111, DO-10
- These had multiple vector inhibitors:
  - Subroutine calls, improper module use
  - Variable length inner loops
  - Backward GO TO
  - Actual & false recursion
  - If-tests too deeply nested
- Loop too long: PHANTJ is effectively 7100 lines!
Solutions to Vectorization Problems

- Inline subroutine calls
- Compile modules as “inlinable” & have no allocatables
- See paper for handling variable length inner loops
- Move recurrence relations outside loop and store results in temporary arrays for use in loop.
- Turn coding within backward Go To into subroutine
- Eliminate if-tests with use of logical variables
- Combat “loop too long” with use of both concurrent directive and aggressive compiler flag.
## Vectorization Speed-up

<table>
<thead>
<tr>
<th>Test Case</th>
<th>PHANTV MFLOPS</th>
<th>PHANTJ MFLOPS</th>
<th>RELAP5 MFLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Orig.</td>
<td>Vector</td>
<td>V/O</td>
</tr>
<tr>
<td>TYP PWR</td>
<td>13.5</td>
<td>57</td>
<td>4.2</td>
</tr>
<tr>
<td>ROSA</td>
<td>18.8</td>
<td>76.5</td>
<td>4.1</td>
</tr>
<tr>
<td>AP600</td>
<td>14.9</td>
<td>92.5</td>
<td>6.2</td>
</tr>
<tr>
<td>3Dflow 15</td>
<td>16.0</td>
<td><strong>98.5</strong></td>
<td>6.2</td>
</tr>
</tbody>
</table>

- Speed is measured in MFLOPS
- Lesser RELAP5-3D speed-up in big problems; solver uses most of the time
Value of Converting to Fortran 90

• Greater machine independence
  – Use Fortran 90 library of intrinsic functions
• Ability to run any size problem
  – Replacement of FA-array with allocatable arrays
• Modernization
  – Conversion to derived types
  – Use of whole array operations
• Longevity
  – Replace any Fortran 66 & 77 constructs that may be illegal in Fortran 2000.
Progress on Fortran 90 Conversion

• Developed program that converts comdecks into Fortran 90 modules.
  – Tested & debugged.
  – Used to convert 3 internal FA files to modules.

• Developed program that converts array access into derived type access.
  – Tested & debugged.
  – Successfully applied to many RELAP5 subroutines.

• Conversion of many subroutines completed by hand.
  – Includes applying FORSTRUCT to reorganize code.
Summary

• Motivation for architectural improvements
• Completion of parallel conversion
• Completion of PVM
• Vectorization work
• Conversion to Fortran 90