Introduction to Distributed-Memory Parallel MM5

John Michalakes
michalak@ucar.edu
PSU/NCAR MM5 Tutorial

Outline
• Parallelism in MM5
• Performance
• Building and using the code
• “Same source” approach
• Linux cluster experiences
• Additional information

Parallelism in MM5
• What is meant by “parallel”?
  -- Increase computational and memory resources available for larger, faster runs by having more than one computer work on the problem
• Isn’t MM5 already parallel?
  -- Yes, the model has been able to run shared-memory parallel since MM4 using Cray Microtasking directives
  -- More recently, standardized OpenMP directives have been incorporated
• What is DM-parallelism? Why?
  -- Processors store part of model domain in local memory, not shared with other processors, and work together on a problem by exchanging messages over a network
  -- “Scalable” because it eliminates bottlenecks on shared resources such as bus or memory
  -- Possibly also more cost effective since systems can be “commodity”
• You already have the DM-parallel version of MM5

Parallelism in MM5
• What is meant by “parallel”?
  -- Increase computational and memory resources available for larger, faster runs by having more than one computer work on the problem
• Isn’t MM5 already parallel?
  -- Yes, the model has been able to run shared-memory parallel since MM4 using Cray Microtasking directives
  -- More recently, standardized OpenMP directives have been incorporated
• What is DM-parallelism? Why?
  -- Processors store part of model domain in local memory, not shared with other processors, and work together on a problem by exchanging messages over a network
  -- “Scalable” because it eliminates bottlenecks on shared resources such as bus or memory
  -- Possibly also more cost effective since systems can be “commodity”
• You already have the DM-parallel version of MM5
Parallelism in MM5

• Increase computational and memory resources available for larger, faster runs by having more than one computer work on the problem

Isn’t MM5 already parallel?

• Yes, the model has been able to run shared-memory parallel since MM4 using Cray Microtasking directives

• More recently, standardized OpenMP directives have been incorporated

What is DM-parallelism? Why?

• Processors store part of model domain in local memory, not shared with other processors, and work together on a problem by exchanging messages over a network

• “Scalable” because it eliminates bottlenecks on shared resources such as bus or memory

• Possibly also more cost effective since systems can be “commodity”

You already have the DM-parallel version of MM5

Uniprocessor (non-parallel): workstations

– Vector shared memory: C90, T90, J90, SV1; NEC; ...

– Shared-memory multi-processors: Sun, Compaq, IBM, SGI, ...

Pure Distributed Memory: IBM SP, Cray T3E, Fujitsu, Beowulf clusters

– Distributed Memory clusters of SMPs: IBM SP, Compaq, SGI, NEC, …
Building the DM-parallel MM5

- Download:
  - ftp://ftp.ucar.edu/mesonnc/mm5v3/MM5.tar.gz
  - ftp://ftp.ucar.edu/mesonnc/mm5v3/MPP.tar.gz
- Unzip and untar:
  - gzip -d -c MM5.tar.gz | tar af - c MM5
  - gzip -d -c MPP.tar.gz | tar af -
- Edit configure.user file for computer and configuration

Editing configure.user

- Find the MPP subsection in Section 7 of configure.user pertaining to your computer and uncomment those rules
- Adjust PROCMIN_NS and PROCMIN_EW settings at top of Section 7 for memory scaling
- Please see http://www.mmm.ucar.edu/mm5/mpp/cowbench/details.html

Memory scaling example

<table>
<thead>
<tr>
<th>Processor Memory</th>
<th>MM5</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 MB</td>
<td>MIX=56</td>
</tr>
<tr>
<td></td>
<td>MJX=68</td>
</tr>
<tr>
<td></td>
<td>46 MB</td>
</tr>
</tbody>
</table>

Memory scaling example

<table>
<thead>
<tr>
<th>Processor Memory</th>
<th>MM5</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 MB</td>
<td>MIX=112</td>
</tr>
<tr>
<td></td>
<td>MJX=136</td>
</tr>
<tr>
<td></td>
<td>184 MB</td>
</tr>
</tbody>
</table>

Memory scaling example

<table>
<thead>
<tr>
<th>Processor Memory</th>
<th>MM5</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 MB</td>
<td>MIX=112</td>
</tr>
<tr>
<td></td>
<td>MJX=136</td>
</tr>
<tr>
<td></td>
<td>MIX=56</td>
</tr>
<tr>
<td></td>
<td>MJX=68</td>
</tr>
<tr>
<td></td>
<td>184 MB</td>
</tr>
</tbody>
</table>

Memory scaling example

<table>
<thead>
<tr>
<th>Processor Memory</th>
<th>MM5</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 MB</td>
<td>MIX=112</td>
</tr>
<tr>
<td></td>
<td>MJX=136</td>
</tr>
<tr>
<td></td>
<td>MIX=56</td>
</tr>
<tr>
<td></td>
<td>MJX=68</td>
</tr>
<tr>
<td></td>
<td>Only 46 MB needed on each processor</td>
</tr>
<tr>
<td></td>
<td>MIX = MIX_GLOBAL / PROCMIN_NS = 56</td>
</tr>
<tr>
<td></td>
<td>MJX = MJX_GLOBAL / PROCMIN_EW = 68</td>
</tr>
</tbody>
</table>
**Memory scaling example**

- Processor Memory: 50 MB
- Processor Memory: 50 MB
- Processor Memory: 50 MB
- Processor Memory: 50 MB

**PROCMIN variables**

- **Determine horizontal dimensions of MM5 arrays for each processor at compile time**
  - **PROCMIN_NS** divides **MIX** (north-south decomposition)
  - **PROCMIN_EW** divides **MJX** (east-west decomposition)
- Can reduce per processor size of MM5 arrays to exploit aggregate memory size of the parallel machine
- **(PROCMIN_NS) x (PROCMIN_EW)** specifies the minimum number of processors at compile time for which the MM5 executable is valid

**PROCMIN variables (cont.)**

- An executable compiled with **PROCMIN_NS=1** and **PROCMIN_EW=1** uses maximum per processor memory but is valid for any number of processors.
- Warning! An executable compiled with **PROCMIN_NS=2** and **PROCMIN_EW=2** can be run on no fewer than 4 processors, but for example it can NOT be run on 5 processors (MIX/2 dimension is too small for 1x5 decomposition)
- Violation will cause runtime abort with message in rsl.error.0000 file: 'MPASPECT: UNABLE TO GENERATE PROCESSOR MESH. STOPPING.'

**Building the code (cont.)**

- Build the model: `make mpp`
- Resulting executable: `Run/mm5.mpp`
- To remake the code in different configuration:
  - `make mpclean`
- To reinstall the code in different location:
  - `make uninstall`

**Running the model**

- Generate the mmlif (namelist) file
  - `make mmlif`
  - `make mmlif` (creates namelist file in `Run/mmlif`; does not run code)
- Run the model
  - `cd Run`
  - `mpirun -np 4 ./mm5.mpp` (standard, MPICH)
  - `mpirun` (DEC MPI)
  - `mpirun` (Sun MPI)
  - `mpirun` (LINUX/ScaMPI)
  - `poe` (IBM)
Running the model (cont.)

- Model generates normal MMOUT_DOMAIN output files and 3 text files per processor:
  - rsl.out.0000 (contains standard output)
  - rsl.error.0000 (contains standard error)
  - show_domain_0000 (shows the domain decomposition)

Test datasets

- Storm of the Century
  ftp://ftp.ucar.edu/mesouser/MM5V3/TESTDATA/input2mm5.tar.gz
  - Good small case for initial testing
  - Includes a test
- Large domain (World Series Rain-out)
  - Representative problem sizes for distributed memory

“Same source” concept

- Ideal – Source code for the DM-parallel and non-DM parallel model are identical at the science level
- Hide parallel details “under the hood”: automate and encapsulate
- Parallel toolboxes:
  - FLIC - automatic generation of I and J loop indexes
  - RSL - routines for domain decomposition and message passing

“Same source” (cont.)

sound.F:

```fortran
#ifdef MPP1
#include <mpp_sound_30.incl>
#endif

MPP/RSL/mpp_sound_30.incl:
  CALL RSL_EXCH_STENCIL(DOMAINS(INEST),STEN_SB(INEST))

MPP/RSL/parallel_src/define_comms.F:
  COMM_3PT_NE(u3d,3)
  COMM_3PT_NE(v3d,3)
  messages(1) = RSL_INVALID
  messages(2) = n1
  messages(3) = ne
  messages(4) = RSL_INVALID
  messages(5) = e1
  messages(6) = RSL_INVALID
  messages(7) = RSL_INVALID
  messages(8) = RSL_INVALID
  call rsl_create_stencil(sten_sb(inest))
  call rsl_describe_stencil(did,sten_sb(inest)),RSL_8PT,messages)
```

DM-parallel MM5 and Linux clusters

- Cost effective
- Scale well with good interconnect
  - Dolphin/Scali
  - Myrinet
- Reliable, but in-house expertise needed
- Distributed memory version of MM5 necessary

More

- Reporting problems with DM-parallel version
  - First rebuild the code and reproduce problem
  - Test non-DM parallel version with same configuration
  - Check for consistent MM5.TAR and MPP.TAR versions
  - Provide: good description of aberrant behavior, the version of code, plus the configure.user, mmlif, rsl.out.0000, rsl.error.0000, and any other rsl.error.* or rsl.out.* files that contain tell-tale error messages.
- Advanced topics
  - Adding/modifying code for DM-parallelism
  - Porting to new platforms
- Additional information
  - README-MPP file
  - www.mmm.ucar.edu/mm5/mpp
    downloading, compiling, running, Helpdesk, MPP Design and Implementation Document
- All your base are belong to us

Ask Rotang