Scalability of ACCESS-OM2 on Gadi - Paul Leopardi 18 March 2020

The aim is to better tune ACCESS-OM2 on Gadi, especially the 0.1 degree experiments

1. Scalability of standalone models, starting with MOM5

The performance of the individual models is best measured by running them as standalone. This also more easily allows performance tuning tools, such as Scorep, ARM, Intel to be used, since Payu is not needed.

Rui provided a starting configuration in the directory **`pom25`**. This uses MOM_SIS and is based on **`global_0.25_degree_NYF.`**

A version of **`input.nml`** that is close to the one in the **`pom25`** directory addresses the following issues.

a. **`mpp_io_init: error reading input.nml`**

   Removed the variable **`parallel_netcdf`** from namelist **`mpp_io_nml`**.

b. **`ocean_topog_mod: min_thickness must be explicitly specified in ocean_topog_nml`**

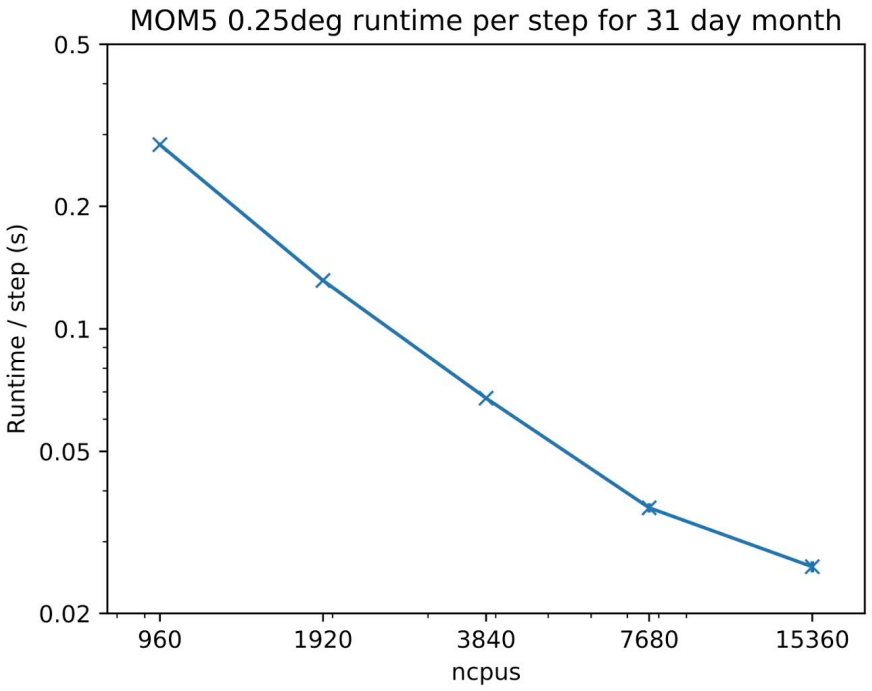
   Set variable **`min_thickness`** to **`1.0e-3`** in namelist **`ocean_topog_nml.`**

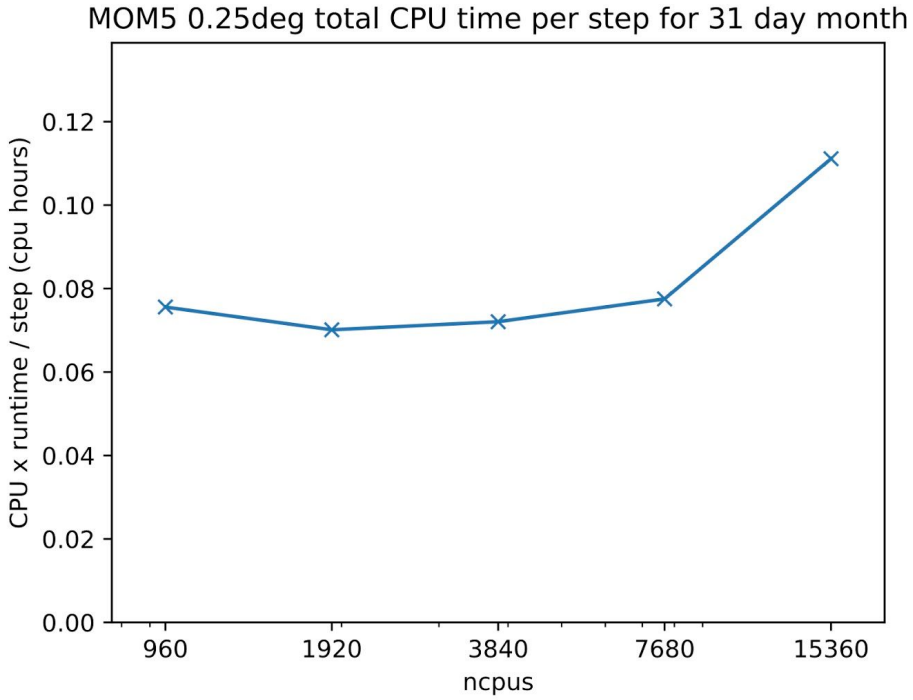
c. **`NetCDF: One or more variable sizes violate format constraints`**

   Took the **`input.nml`** file from **`pom25`** directory, made changes a. and b., and changed **`io_layout`** from 1,1 to 4,5. Used the **`diag_table`** file from **`pom25`**.

The MOM_SIS 0.25 degree experiment was scaled to 960, 1920, 3840, 7680 and 15360 cores. Each run is for a simulated 31 day month, and runtime is listed per 1800 second (0.5 hour) time step. CPU time is runtime in hours multiplied by the number of CPUs. The number of cores is doubled on each row, and land masking is ignored in calculating the number of cores. Each row summarises 5 runs.

Cores	Layout	Runtime (s)	CPU time (cpu hours)
960	= 48 x 20:	0.2834 +/- 0.0004	0.076
1920	= 48 x 40:	0.1315 +/- 0.0003	0.070
3840	= 96 x 40:	0.0676 +/- 0.0008	0.072
7680	= 96 x 80:	0.0363 +/- 0.0010	0.078
15360	=192 x 80:	0.0260 +/- 0.0008	0.111


MOM5 0.25deg runtime per step for 31 day month
Runtime / step (s)
0.5
0.2
0.1
0.05
0.02
960
1920
3840
7680
15360
ncpus


MOM5 0.25deg total CPU time per step for 31 day month
CPU x runtime / step (cpu hours)
0.12
0.10
0.08
0.06
0.04
0.02
0.00
960
1920
3840
7680
15360
ncpus

2. Effect of Intel compiler flags on MOM5 performance

Preliminary results of testing with MOM_SIS with 15360 cores indicates that the use of AVX2 vs AVX512 instructions has no significant effect on performance, and neither does the use of reproducibility flags. In the table below, each row summarises 5 runs (4 for the last row) and the figure after +/- is 2 x standard deviation.

Flags	Runtime (s)	CPU time (cpu hours)
AVX2:	0.0263 +/- 0.0016	0.112
AVX512:	0.0261 +/- 0.0012	0.111
AVX512-REPRO:	0.0264 +/- 0.0018	0.113

```
Flags:
AVX2:          -axCORE-AVX2
AVX512:        -axCORE-AVX512
AVX512-REPRO: -axCORE-AVX512 \
               -fp-model precise -fp-model source -align all
```

To get a better idea of the effect of compiler flags on performance, I intend to run the comparison again with 960 cores instead of 15360 cores.

3. MOM_SIS intermittent slowdown

During testing, one run was encountered where the ocean model runtime took 48.4 seconds, as opposed to the 37.8 to 41.3 seconds for the 14 other runs -- Gadi job 2219844 run on 12 March 15:15 to 15:48. This run was treated as an outlier.

The cause of the approx 20% slowdown is unknown, although the qstat output for the job indicates that it was one of the few jobs (or possibly the only job) that was run on Gadi nodes with node numbers above 2900.

4. MOM_SIS crashing in OpenMPI 4.0.2

Two of the MOM_SIS benchmarking runs crashed in OpenMPI 4.0.2:

Case 1: job 2213158

```
[gadi-cpu-clx-1129:6414 :0:6414] Caught signal 11 (Segmentation
fault: invalid permissions for mapped object at address
0x154711b7f17c)
==== backtrace (tid:  6414) ====
 0 0x0000000000012d80 .annobin_sigaction.c() sigaction.c:0
 1 0x000000000003417c opal_progress()
/home/900/z30_apps/builds/_UYaaG8i/0/nci/gadi-apps/openmpi/4.0.2/sour
ce/openmpi-4.0.2/gcc-opt/opal/../../opal/runtime/opal_progress.c:231
 2 0x000000000004ddd5 ompi_request_wait_completion()
/home/900/z30_apps/builds/_UYaaG8i/0/nci/gadi-apps/openmpi/4.0.2/sour
ce/openmpi-4.0.2/gcc-opt/ompi/../../ompi/request/request.h:415
 3 0x000000000004ddd5 ompi_request_default_wait()
/home/900/z30_apps/builds/_UYaaG8i/0/nci/gadi-apps/openmpi/4.0.2/sour
ce/openmpi-4.0.2/gcc-opt/ompi/../../ompi/request/req_wait.c:42
```

…

Case 2: job 2234702

```
[gadi-cpu-clx-1129:57776:0:57776] Caught signal 11 (Segmentation fault: invalid permissions for mapped object at address 0x14c0f1b0edd5)
==== backtrace (tid: 57776) ====
 0 0x0000000000012d80 .annobin_sigaction.c() sigaction.c:0
 1 0x000000000004ddd5 ompi_request_wait_completion() /home/900/z30_apps/builds/_UYaaG8i/0/nci/gadi-apps/openmpi/4.0.2/source/openmpi-4.0.2/gcc-opt/ompi/../../ompi/request/request.h:415
 2 0x000000000004ddd5 ompi_request_default_wait() /home/900/z30_apps/builds/_UYaaG8i/0/nci/gadi-apps/openmpi/4.0.2/source/openmpi-4.0.2/gcc-opt/ompi/../../ompi/request/req_wait.c:42
…
```

This could be caused by a bug in UCX, rather than MOM5. Perhaps valgrind could be used to investigate further. This would need a Valgrind wrapper for OpenMPI 4.0.2.
https://valgrind.org/docs/manual/mc-manual.html#mc-manual.mpiwrap

5. Comparison of MOM5 performance on Gadi to Raijin and scalability of 0.1 degree experiments

To compare with Raijin performance it is best to go back to using ACCESS-OM2.

This benchmarking includes load balancing between the MOM5 ocean model and the CICE5 sea ice model.

To perform this benchmarking, it would be best to start with the exact same 1 degree, 0.25 degree and 0.1 degree configurations and data as were used in the report

https://github.com/COSIMA/ACCESS-OM2-1-025-010deg-report

To this end, I will need some help in understanding the contents of https://github.com/COSIMA/ACCESS-OM2-1-025-010deg-report/tree/master/namelists as well as the locations on Gadi of copies of the original files that were used on Raijin.

Issue: The original scalability work was done on Raijin, and some of the original configuration files (owned by Marshall Ward) may have been lost in the transition to Gadi. https://arccss.slack.com/archives/C9Q7Y1400/p1581897666071800

Issue: Not all of the scalability work was fully documented, and the documentation is scattered between the report, various Github repositories, and the upstream documentation.