ACCESS-OM2 Performance Report

Performance of ACCESS-OM2 on NCI Gadi

Document History

Name	Date	Comments
Paul Leopardi	2020-06-02	Began
Paul Leopardi	2020-06-19	Initial draft
Paul Leopardi	2020-06-24	Include years per day
Paul Leopardi	2020-06-30	Include instructions
Paul Leopardi	2020-07-06	Include comparison to Raijin
Paul Leopardi	2020-07-14	Update comparison to Raijin
Paul Leopardi	2020-07-17	Use mean times, include initialization

Contents

Introduction

This report examines the performance of ACCESS-OM2 on Gadi, as revealed by performance tests. The aim of the performance testing was to better understand the scalability of ACCESS-OM2 on Gadi, so that performance could be improved by tuning, and to give options to the COSIMA researchers in terms of the number of CPUs used versus walltime.

Background

ACCESS-OM2 is an ocean and sea ice model developed by the Consortium for Ocean-Sea Ice Modelling in Australia (COSIMA). It comprises the MOM5 ocean model, the CICE5.1 sea ice model, and atmospheric forcing provided via YATM. The MOM5 sources include a coupler and the SIS sea-ice model, which is still used for testing MOM5, as the coupled MOM-SIS model. For more details, see the paper [1], the report [2], and the COSIMA web page [3] that describes these models.

Summary of tests conducted

Testing included:

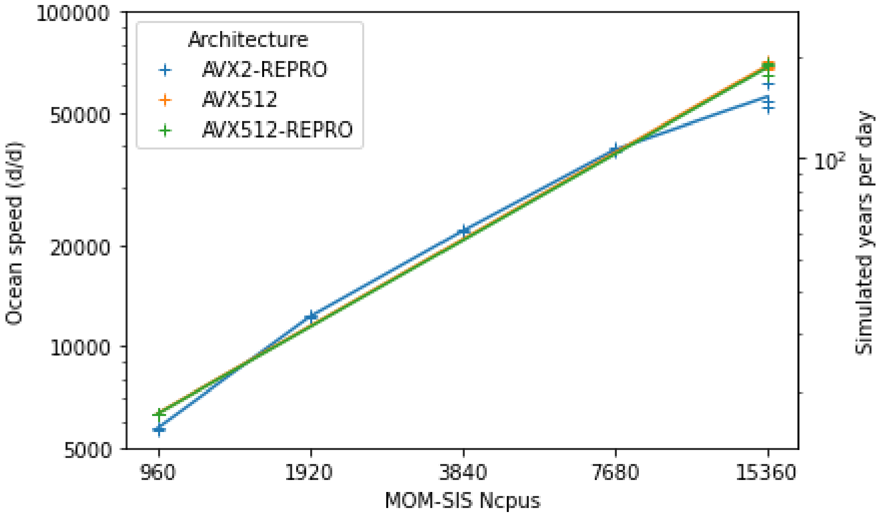
1. MOM-SIS at 0.25 degrees and 0.1 degrees:
    a. Testing MOM with SIS, unmasked, without writing restart files, to determine the "standalone" scalability of MOM.
2. ACCESS-OM2 at 0.1 degrees:
    a. Testing ACCESS-OM2 with land masking, scaling MOM and CICE proportionally;
    b. Investigating the effects of varying the number of blocks used by CICE;
    c. Investigating the effects of varying the proportion of CICE CPUs to MOM CPUs;
    d. Unsuccessful reproduction of Marshall Ward's tests as documented in the report [2].
3. MOM-SIS at 0.25 degrees:
    a. Analysis of traces of MOM-SIS runs to find potential bottlenecks, which has not yet been completed and is not included in this report.

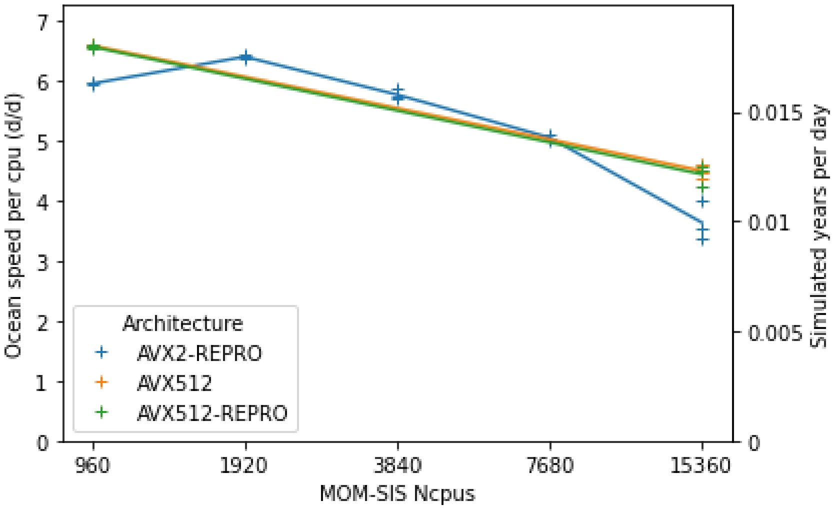
Note: Testing of ACCESS-OM2 with CICE parallel IO has not yet been done.

Summary of results

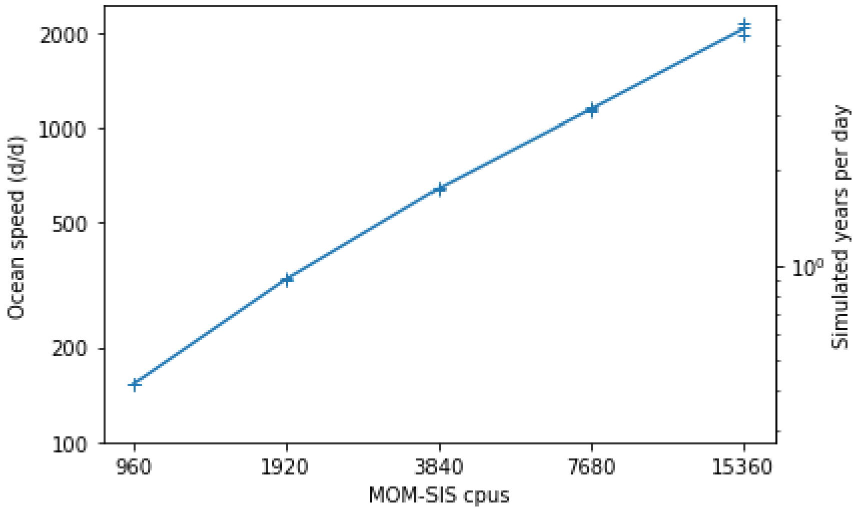
Note: in the following “d/d” denotes “simulated days per real day”, and “Y/d” denotes “simulated Gregorian years per real day”, derived by dividing the “d/d” amount by 365.2425.

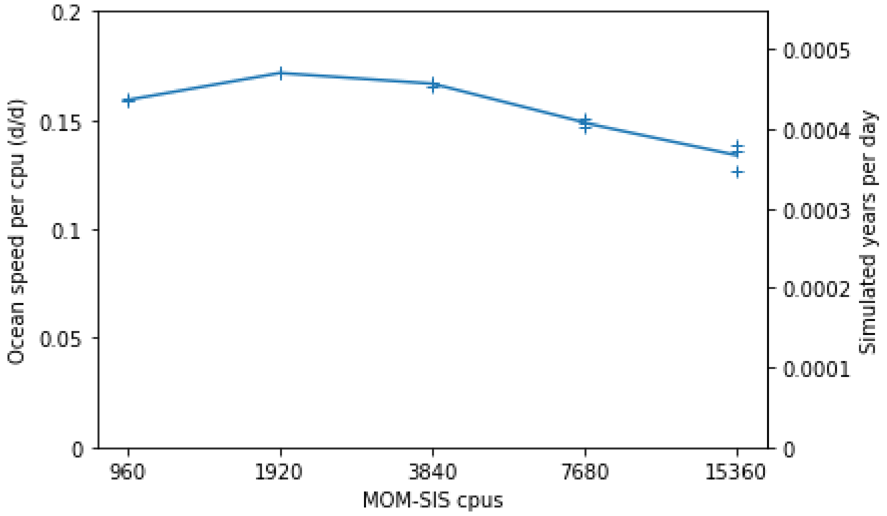
1. Testing MOM-SIS on Gadi revealed that:
    a. At 0.25 degrees:
        i. MOM-SIS compiled with AVX2 and reproducibility flags scaled reasonably well up to 7680 CPUs, but was most efficient at 1920 CPUs;
        ii. MOM-SIS compiled with AXV512 and reproducibility flags ran faster than case (i.): about 10% faster at 960 CPUs and about 22% faster at 15360 CPUs.



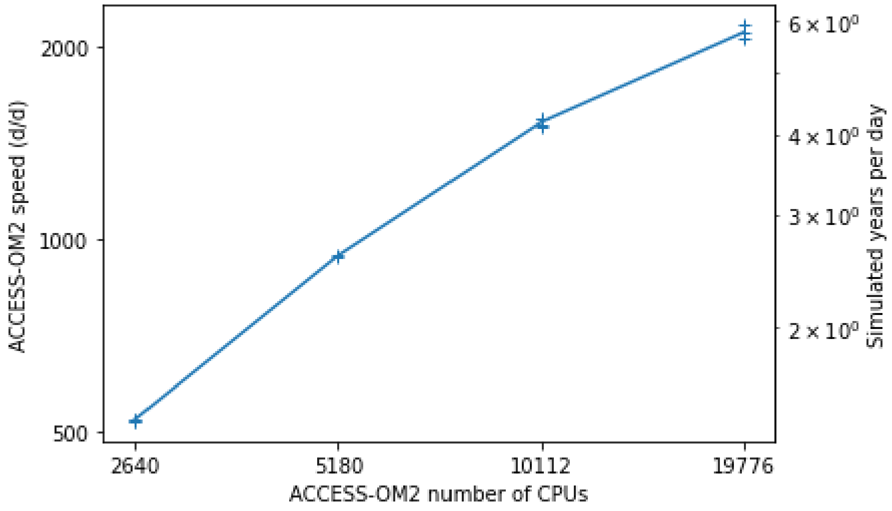


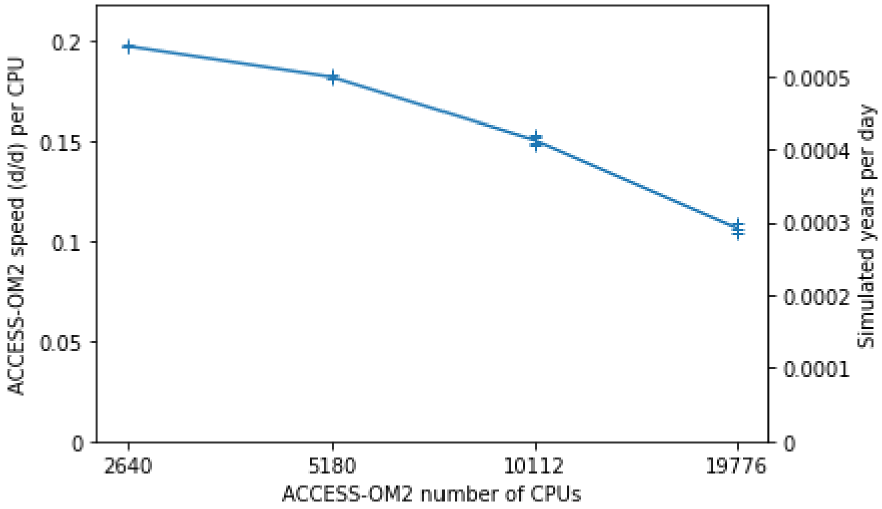
b. At 0.1 degrees, MOM-SIS compiled with AVX512 and reproducibility flags scaled reasonably well up to 15360 CPUs, but was most efficient at 1920 CPUs.



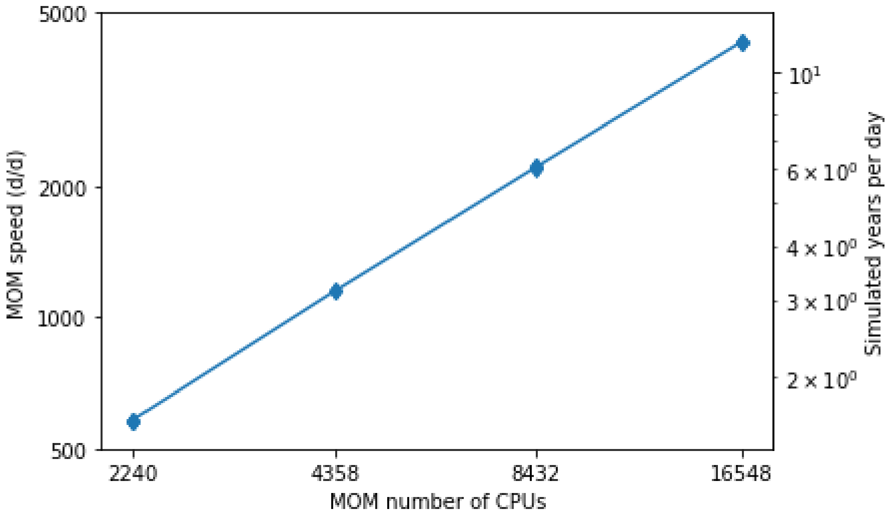


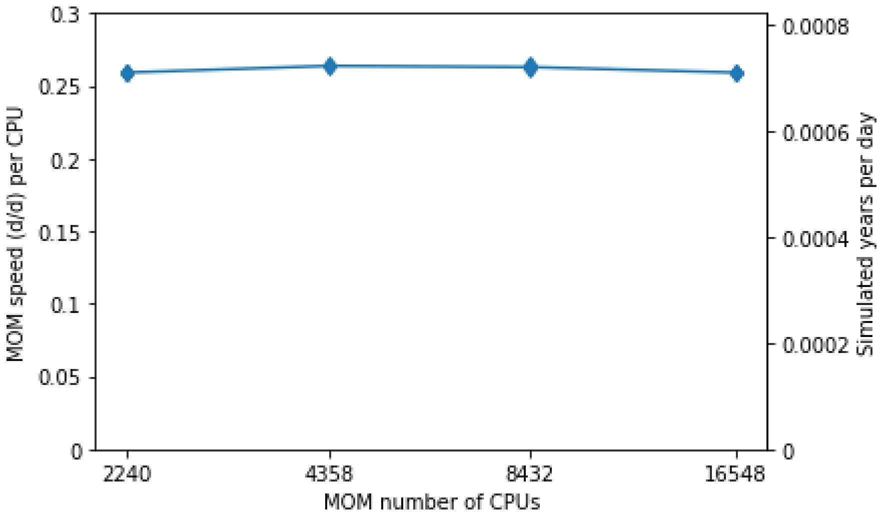
2. Testing ACCESS-OM2 at 0.1 degrees revealed that:
    a. With the currently used proportion of CICE CPUs to MOM CPUs (eg. 799 to 4358):
        i. ACCESS-OM2 does not scale well in terms of walltime. Speed per CPU drops from about 0.20 d/d per CPU at 2640 CPUs to about 0.11 at 19776 CPUs (0.00054 Y/d per CPU to 0.00029);



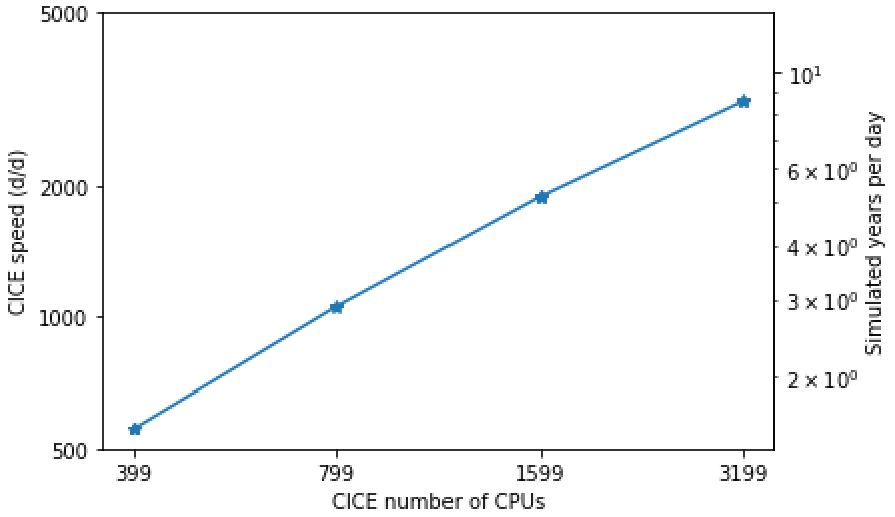


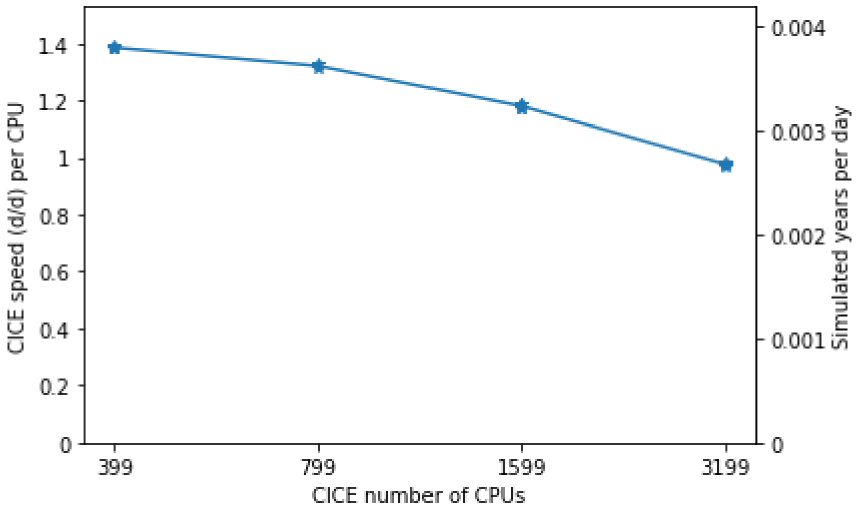
ii. MOM scales linearly in terms of reported "Ocean" time for its main loop;



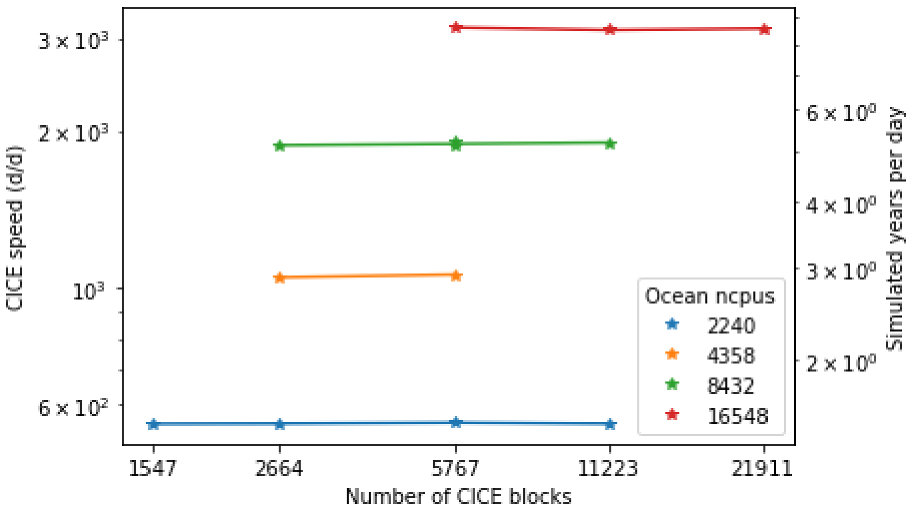


iii. CICE speed in terms of reported “TimeLoop” drops from about 1.4 d/d per CPU at 399 CPUs to about 0.98 at 3199 CPUs (0.0038 Y/d per CPU to 0.0027).



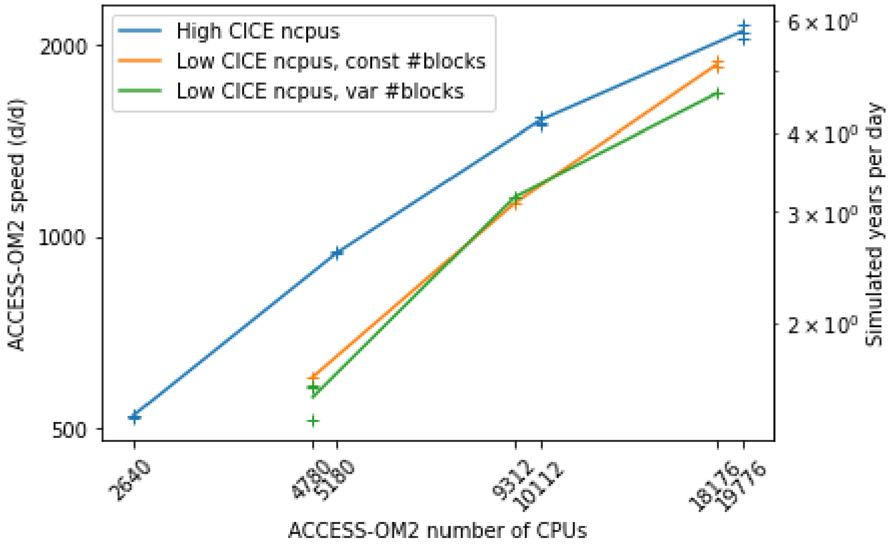


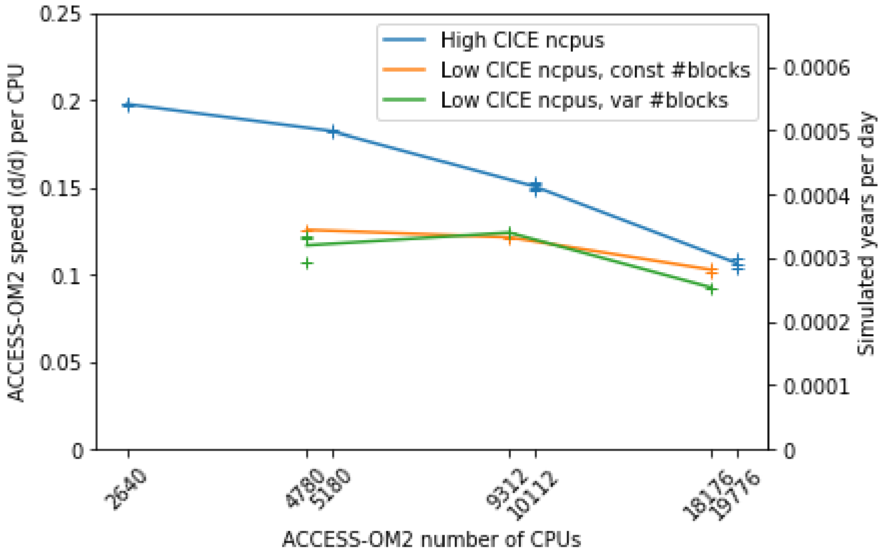
iv. Varying the number of CICE blocks has no significant effect.



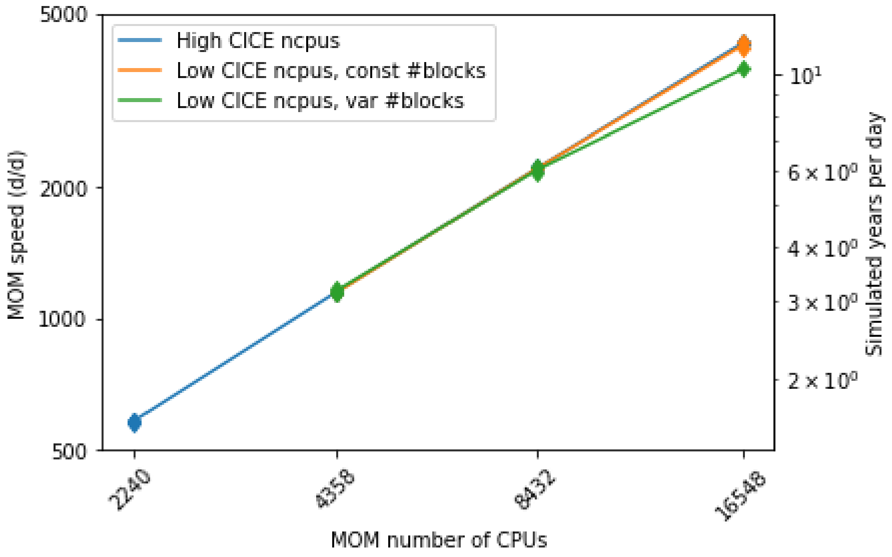
b. With half the proportion of CICE CPUs to MOM CPUs (eg. 399 to 4358) and a constant number of CICE blocks:
   i. ACCESS-OM2 runs more slowly than case (a.), but scales better in terms of overall walltime, dropping from about 0.13 d/d per CPU at 4780 CPUs to about 0.10 at 18176 CPUs (0.00034 Y/d per CPU to 0.00028);
   ii. MOM runs at about the same speed as per case (a.) and continues to scale linearly;
   iii. CICE runs faster than case (a.) and scales better, dropping from about 1.8 d/d per CPU at 399 CPUs to about 1.6 at 1599 CPUs (0.0049 Y/d per CPU to 0.0044).

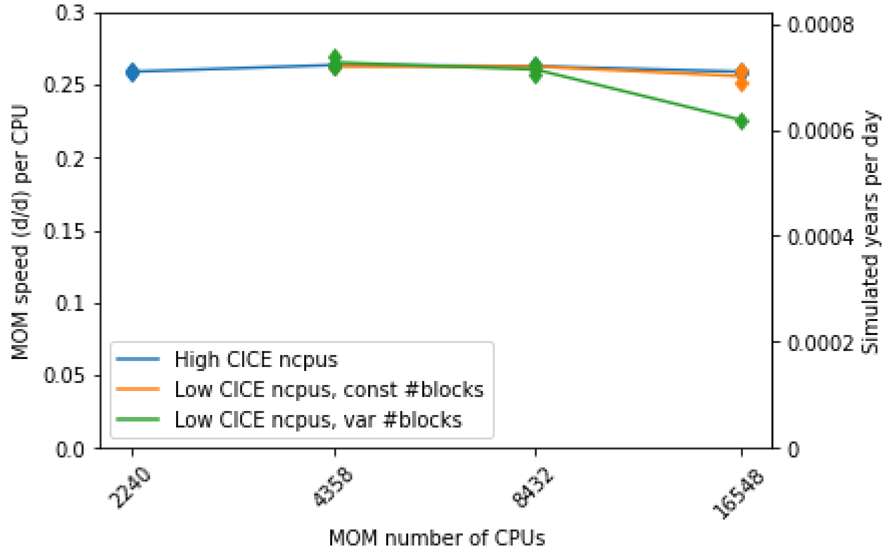
c. With half the proportion of CICE CPUs to MOM CPUs and varying the number of CICE blocks in proportion with CICE CPUs:
   i. ACCESS-OM2 generally runs slightly more slowly than case (b.), and scales slightly worse in terms of overall walltime, dropping from about 0.12 d/d per CPU at 4780 CPUs to about 0.09 at 18176 CPUs (0.00032 Y/d per CPU to 0.00025);



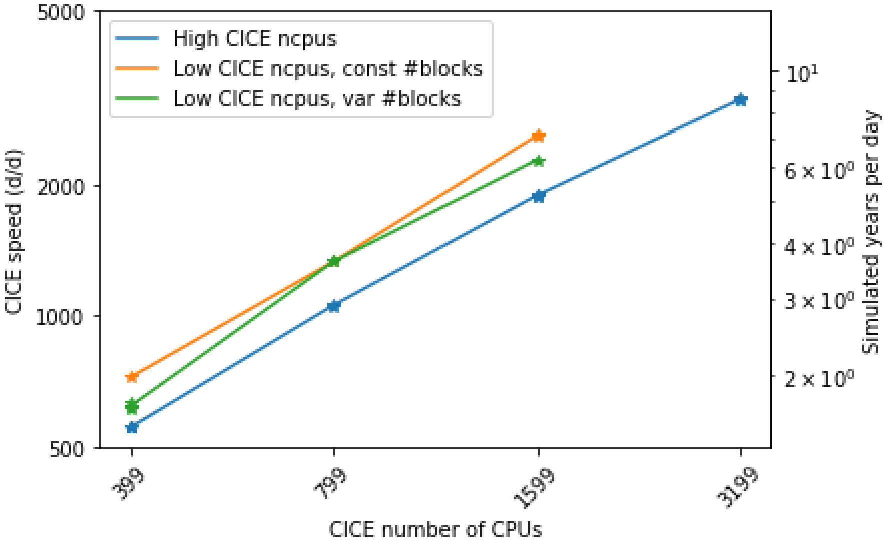


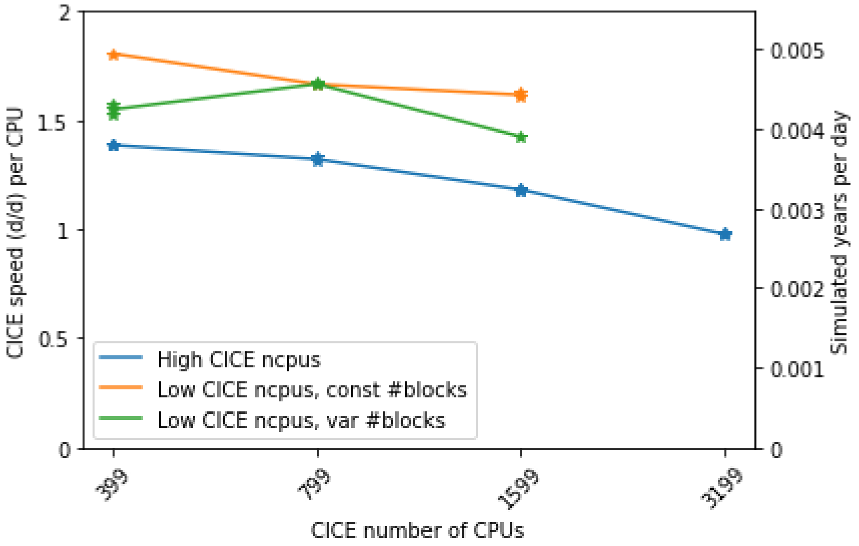
ii. MOM runs at about the same speed as per case (b.) up to 8432 CPUs, but scales worse, dropping from about 0.27 d/d at 4358 CPUs to about 0.25 at 16548 CPUs (0.00074 Y/d per CPU to 0.00069);





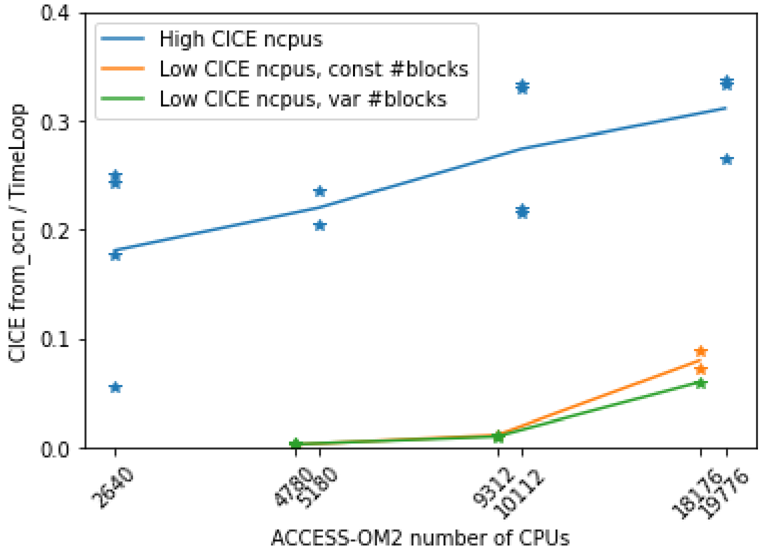
iii. CICE generally runs more slowly than case (b.), and scales in a surprising way, increasing from about 1.5 d/d at 399 CPUs to about 1.7 at 799 CPUs, then dropping to about 1.4 at 1599 CPUs (0.0042 Y/d per CPU to 0.0046 then 0.0039).

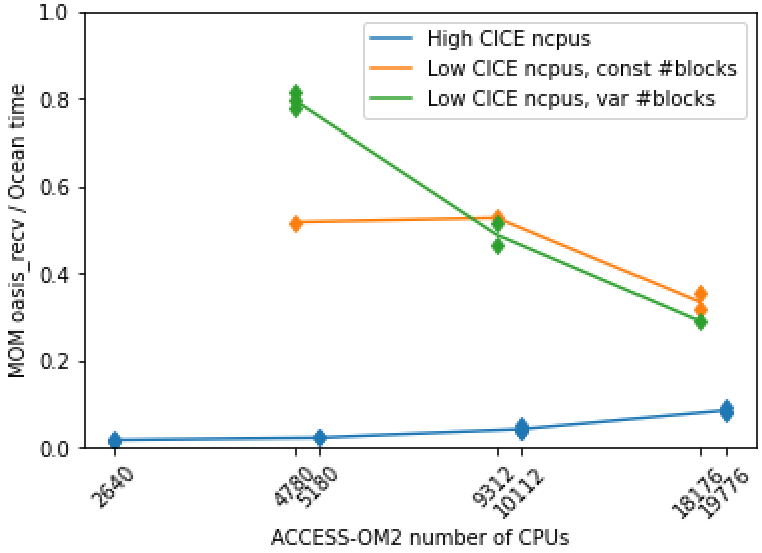




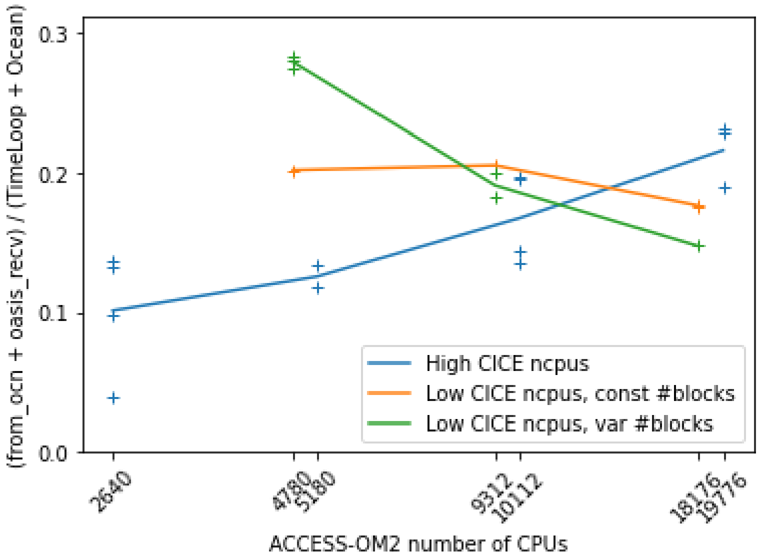
Cases (b.) and (c.) differ from case (a.) in terms of wait times.

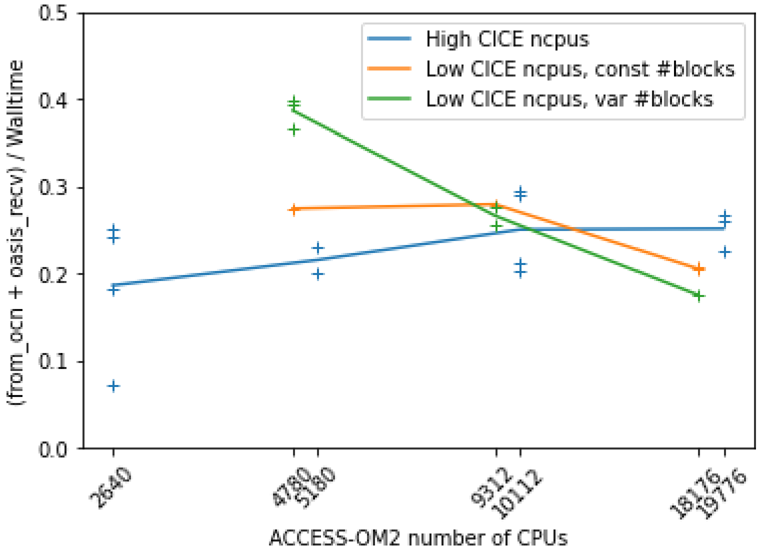
i. In case (a.), CICE waits for MOM. In cases (b.) and (c.) MOM waits for CICE. The difference between (a.) and (b.)/(c.) reduces with increasing numbers of CPUs.



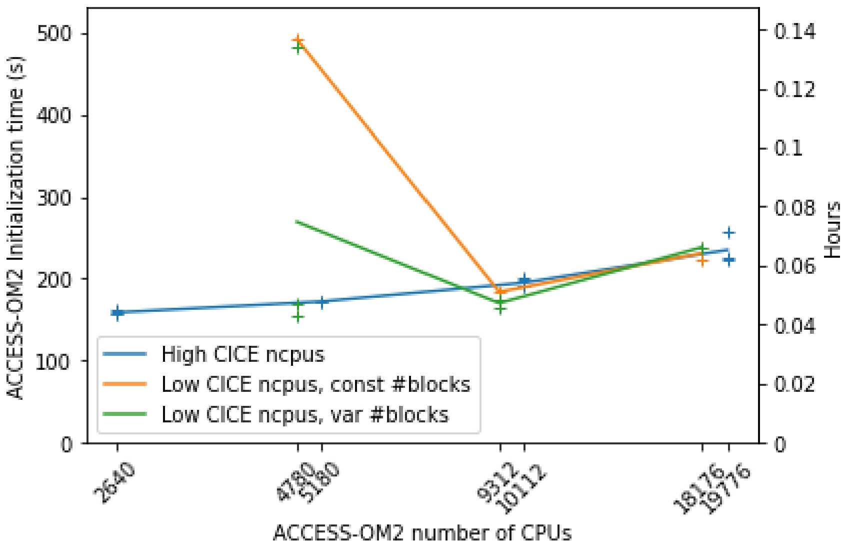


ii. At 19776 CPUs, case (a.) has a larger relative wait time than either of cases (b.) or (c.) at 18176 CPUs. Here relative wait time is calculated as `(CICE from_ocn + MOM oasis_recv) / (CICE TimeLoop + MOM Ocean time)`. Similar behaviour is seen for `(CICE from_ocn + MOM oasis_recv) / (OM2 Walltime)`.

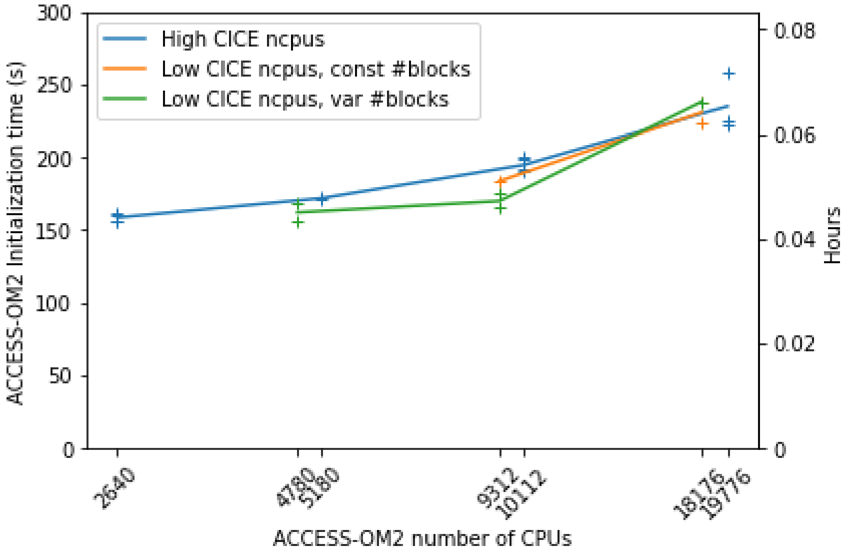




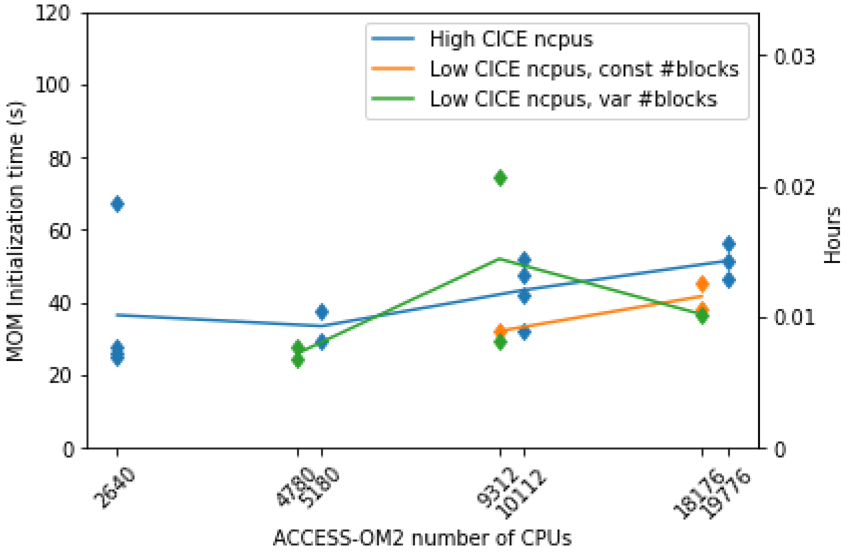
Initialization times show a wide variation. Two runs had an initialization time about twice that of all the other runs.



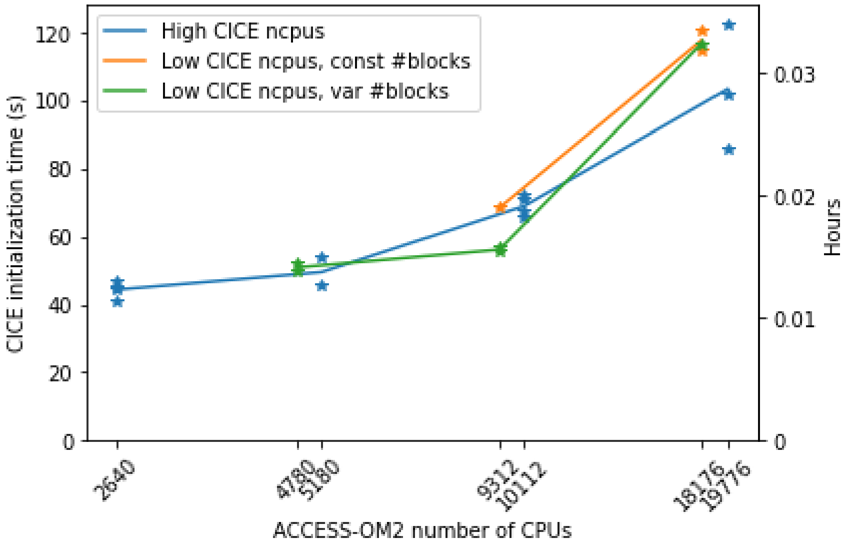
When these two outlier runs are excluded, initialization time is seen to increase with increasing numbers of CPUs, with little observable dependence on the proportion of CICE CPUs.



When the two outlier runs are excluded, ocean initialization time shows a wide variation in time between about 20 seconds to about 80 seconds with very little observable dependence on the number of OM2 CPUs.

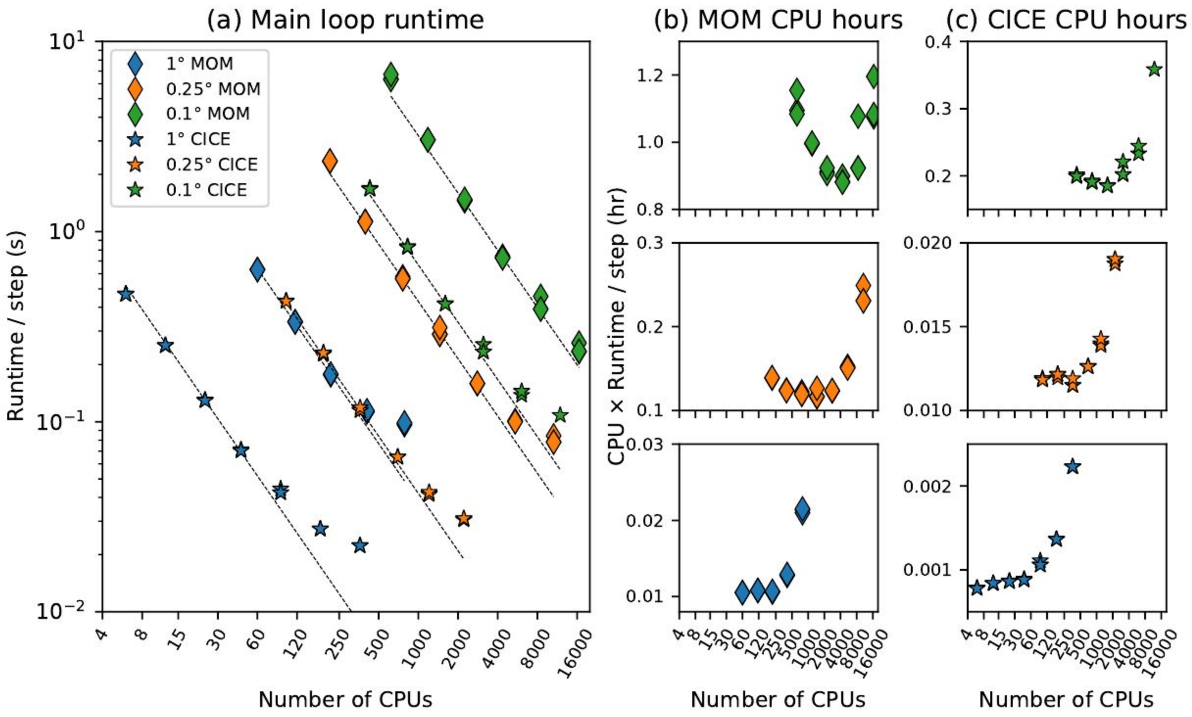


When the two outlier runs are excluded, ice initialization time is seen to increase with increasing number of OM2 CPUs.



Comparison to parallel scaling in “A technical description …”

The technical description of ACCESS-OM2 [2] contains the following Figure 12, showing ACCESS-OM2 scaling on Raijin.



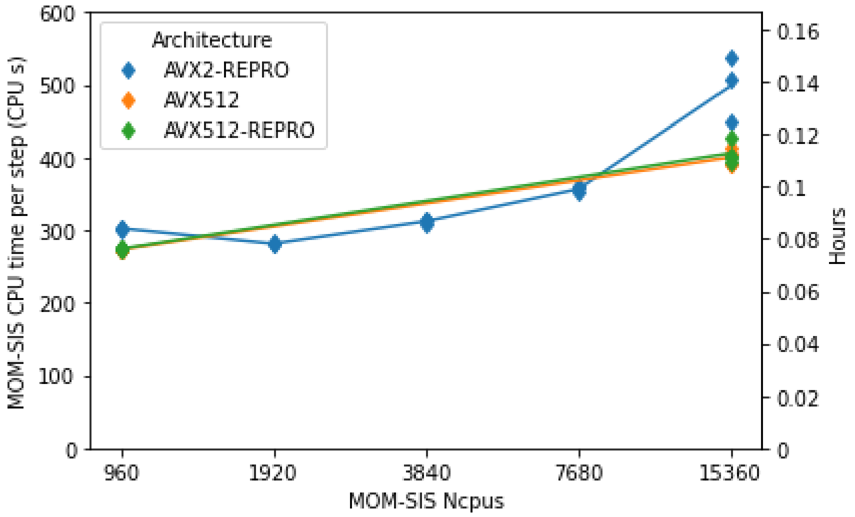
The results presented in this report are not directly comparable to the Raijin results [2, Figure 12] for the following reasons:

1. The MOM-SIS results are based on a different coupler and sea ice model than the ACCESS-OM2 model represented by the Raijin results [2, Figure 12]. Additionally, the number of CPUs used by MOM-SIS is not split into separate MOM and SIS numbers.
2. The ACCESS-OM2 results are based on different configurations from those used to produce the results on Raijin [2, Figure 12]. Key differences other than the migration to Gadi include:
    a. A continuation run was used, rather than an initial run,
    b. The time step used was 540 seconds rather than 400 seconds,
    c. MOM and CICE were scaled in proportion rather than separately,
    d. Scaling was taken to almost 20,000 CPUs rather than 16,000.

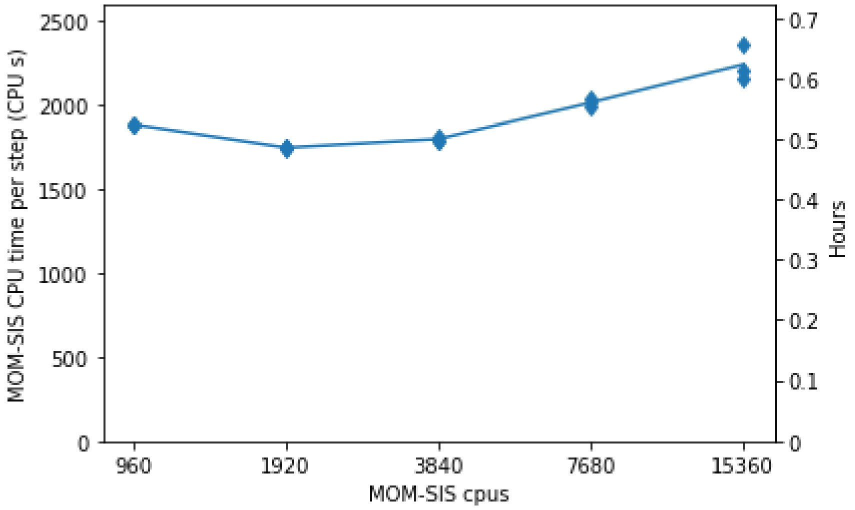
For details of the configurations, see below. With these differences in mind, the remainder of this section presents a comparison to the scalability of ACCESS-OM2 on Raijin.

Comparison of MOM-SIS results to ACCESS-OM2 results on Raijin

1. At 0.25 degrees, MOM-SIS CPU time varies from about 0.08 CPU hours per step to about 0.15, depending on architecture. The Raijin results [2, Figure 12 (b), middle] show a variation in CPU time from about 0.1 MOM CPU hours per step to about 0.25. This would make MOM-SIS on Gadi between 25% faster than ACCESS-OM2 MOM on Raijin at the low end of CPU scaling, to about twice as fast at the high end.

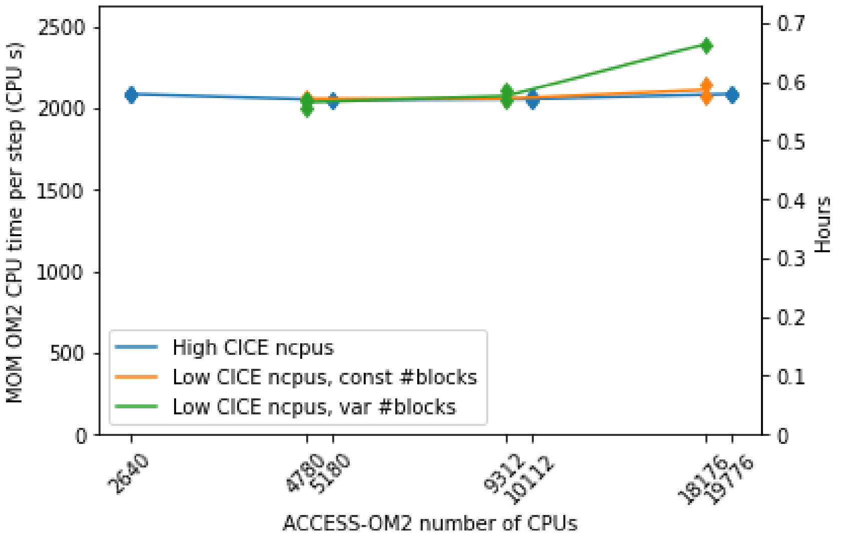


2. At 0.1 degrees, MOM-SIS CPU time varies from about 0.5 CPU hours per step to about 0.65. The Raijin results [2, Figure 12 (b), top] show a variation in CPU time from about 0.85 MOM CPU hours per step to about 1.2. This would make MOM-SIS on Gadi between 70% faster than ACCESS-OM2 MOM on Raijin, to almost twice as fast at both the low and high ends of CPU scaling.

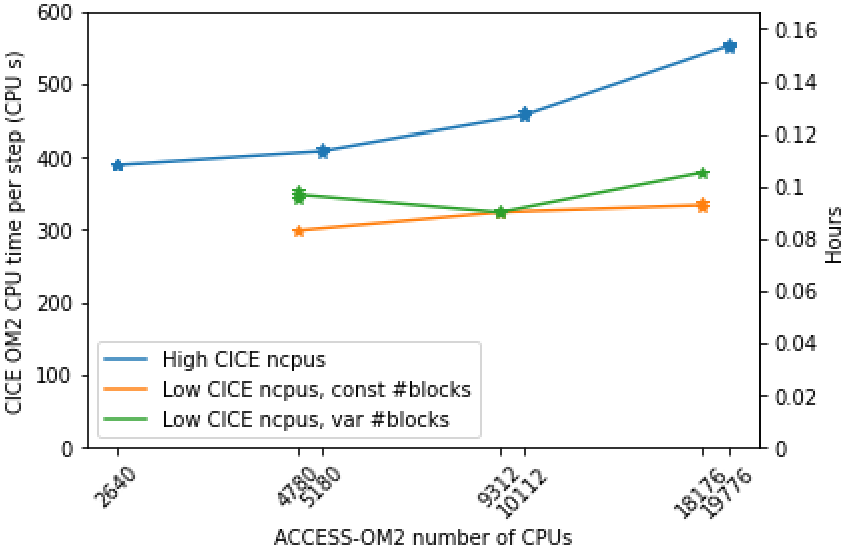


Comparison of ACCESS-OM2 results to results on Raijin

1. At 0.1 degrees, MOM CPU time varies from about 0.55 CPU hours per step to about 0.7, with a small variation with respect to the ratio between the number of CICE and MOM CPUs. This result is broadly consistent with the MOM-SIS results, when the differences in configuration are noted. The Raijin results [2, Figure 12 (b), top] show a variation in CPU time from about 0.85 MOM CPU hours per step to about 1.2. At around 5000 ACCESS-OM2 CPUs, MOM on Gadi looks to be about 50% faster than MOM on Raijin. This scales to almost twice as fast at around 16,000 CPUs, with good scaling to almost 20,000 CPUs.



2. At 0.1 degrees, CICE CPU time varies from about 0.08 CPU hours per step to about 0.16, depending largely on the ratio between the number of CICE and MOM CPUs. For a given number of ACCESS-OM2 CPUs, a higher proportion of CICE CPUs gives a higher CICE CPU time. The Raijin results [2, Figure 12 (c), top] show a variation in CPU time from about 0.18 CICE CPU hours per step to about 0.38. At around 2500 ACCESS-OM2 CPUs, CICE on Gadi with a high proportion of CPUs looks to be between 50% faster than CICE on Raijin. This scales to about 2.5 times as fast at around 16,000 CPUs.



Note on experiment directory structure

The source code, configurations and output directories given in the detailed experiment descriptions below were originally on Gadi at `/scratch/fp0/pcl900` and were copied to equivalent directories at `/g/data/v45/pcl900` for longer term storage and easier access by COSIMA members. Thus configuration files, scripts, etc. contain references to either `/scratch/fp0/pcl900` or the symbolic link `/home/900/pcl900/scratch`

MOM-SIS testing at 0.25 degrees and 0.1 degrees

The performance and scalability of MOM was measured by running the MOM-SIS model at 0.25 degrees and 0.1 degrees. The idea was to reduce the complexity of the system to be analysed and be able to run the model without using Payu. Running without Payu eased the use of performance tuning tools such as Scorep and Scalasca.

Source code

The source code used is the `4-instrument-for-scalability` branch of the `penguian/MOM5` repository on GitHub. This repository is forked from `mom-ocean/MOM5` as of 2019-12-12, and cloned to Gadi at `/g/data/v45/pcl900/access-om2/mom-sis/src/mom`

The main code change from 2019-12-12 is to ensure that all modules that can write a restart file have a corresponding namelist containing a boolean flag `write_a_restart`, and that the corresponding restart file is only written to if that flag is set to `.true.`

Compilation

Within the source code directory, the build script is at `exp/MOM_compile.csh`

Module versions used for compilation are defined in `bin/environs.nci` and are:

`intel-compiler/2019.5.281, netcdf/4.7.1,` and `openmpi/4.0.2`

Three software versions were used for MOM_SIS testing, differing by compilation flags:

1. AVX2-REPRO:
   `-fp-model precise -fp-model source -align all -axCORE-AVX2`
2. AVX512:
   `-axCORE-AVX512`
3. AVX512-REPRO:
   `-fp-model precise -fp-model source -align all -axCORE-AVX512`

The remaining compilation flags were:

```
-fpp -Wp,-w -fno-alias -safe-cray-ptr -fpe0 -ftz -assume byterecl
-i4 -r8 -traceback -nowarn -check noarg_temp_created -assume
nobuffered_io -convert big_endian -grecord-gcc-switches -align all
-g3 -O2 -debug all -check none -qopt-report=5
-qopt-report-annotate -c -pthread
```

Each software version was compiled into the corresponding subdirectory of `exec`, complete with annotated listings.

Runtime configurations at 0.25 degrees

The 0.25 degree configurations are under the subdirectory
`work/global_0.25_degree_NYF/`

They are based on `/g/data/fp0/shared/mom5/examples/pom25/`.

This uses MOM_SIS and is in turn based on
`/g/data/ua8/mom/test_data/global_0.25_degree_NYF.input.tar.gz`

The configuration changes from the version at `global_0.25_degree_NYF.input.tar.gz` are:

`INPUT/diag_table`

- All diagnostics are commented out.

`INPUT/input.nml`

- Changed `layout` from `32,30` to one of `48,20; 48,40; 96,40; 96,80` or `192,80`
- Changed `io_layout` from `32,30` to `4,5`.
- In namelist `coupler_nml`:
  a. Set `months = 1, days = 0, hours = 0`

     Since the start date is 1 January 1900, this gives a 31 day month.

  b. Removed the line setting `atmos_npes = 0`
  c. Set `ice_npes = 0`
- In namelist `diag_manager_nml`:
  a. Set `debug_diag_manager = .false.`
- In namelist `fms_io_nml`:
  a. Set `fms_netcdf_restart = .false.`
- In namelists `ice_model_nml, ocean_barotropic_nml, ocean_bih_friction_nml, ocean_density_nml, ocean_lap_friction_nml, ocean_nphysics_nml, ocean_sbc_nml, ocean_sigma_transport_nml, ocean_thickness_nml, ocean_tracer_advect_nml, ocean_tracer_nml, ocean_velocity_nml, ocean_vert_gotm_nml:`
  a. Set `write_a_restart = .false.`
- Added namelist `mpp_io_nml`:
  a. Set `io_clocks_on = .true.`
- In namelist `ocean_topog_nml`:
  a. Set `min_thickness = 1.0e-3.`
- In namelist `xgrid_nml`:
  a. Removed the line setting `xgrid_log = .false.`
  b. Set `do_alltoall = .true.`
  c. Set `do_alltoallv = .true.`

No ocean masking was used.

Runtime configurations at 0.1 degrees

The 0.1 degree configurations are under the subdirectory `work/global_0.1_degree/`.

They are based on `/scratch/fp0/shared/sis_720` with `INPUT/input.nml` based on `work/global_0.25_degree_NYF/`.

The configuration differences from the version at `work/global_0.25_degree_NYF` are:

`INPUT/data_table`

- Commented out the line containing `"ICE" , "runoff"`.

`INPUT/input.nml`

- Changed `layout` from `32,30` to one of `48,20; 48,40; 96,40; 96,80` or `192,80`
- Changed `io_layout` from `32,30` to `4,5`
- In namelist `coupler_nml`:
    a. Set `months = 0, days = 1`
- In namelist `ocean_bihgen_friction_nml`:
    a. Changed `ncar_boundary_scaling_read` from `.true.` to `.false.`
- In namelist `ocean_model_nml`:
    a. Changed `dt_ocean` from `1800` to `300`
    b. Changed `barotropic_split` from `80` to `35`
- In namelist `ocean_sbc_nml`:
    a. Changed `salt_restore_tscale` from `60.0` to `-60.0`
    b. Set `zero_river_fluxes=.false.`
    c. Set `zero_runoff_fluxes=.true.`
    d. Set `zero_pme_fluxes=.false.`
- In namelist `xgrid_nml`:
    a. Set `nsubset = 1024`

No ocean masking was used.

Experiments at 0.25 degrees

Each experiment at 0.25 degrees was for a simulated 31 day month, with a time step of 1800 seconds. For AVX2-REPRO, each layout was run 3 times. For AVX512 and AVX512-REPRO, only the 48x20 (960 CPU) and 192x80 (15360 CPU) layouts were run.

For each of 34 runs, the log file `logfile.000000.out` was copied to the directory `work/global_0.25_degree_NYF/logs.1m` using the naming convention `logfile.$LAYOUT.$ARCH.$RUN.1m`

Four other runs were unusually slow. The cause is unknown. They were excluded from the analysis, but the log files were kept with the naming convention `logfile.$LAYOUT.$ARCH.1m.$RUN.slow`

Experiments at 0.1 degrees

Each experiment at 0.1 degrees was for a single simulated day, with a time step of 300 seconds. For AVX512-REPRO, each layout was run 3 times.

For each of 15 runs, the log file `logfile.000000.out` was copied to the directory `work/global_0.1_degree/logs.1d` using the naming convention `logfile.$LAYOUT.$ARCH.$RUN.1d`

Two other runs with the `192x80` layout failed with segmentation faults. The cause is unknown.

The log files were kept with the naming convention `logfile.$LAYOUT.$ARCH.$RUN.1d.bug`

Analysis

The directory `/g/data/v45/pcl900/access-om2/analysis` contains the Python scripts and the Jupyter notebook `plot_mom_sis_scaling.ipynb` used for the analysis.

ACCESS-OM2 testing at 0.1 degrees

Testing of ACCESS-OM2 was conducted at 0.1 degree resolution because this is the highest resolution used by ACCESS-OM2 and scaling at this resolution is of most interest to COSIMA members, especially after the transition from NCI Raijin to Gadi.

Source code

The source code used is the `gadi-scaling-01` branch of the `penguian/access-om2` repository on GitHub. This repository is forked from `COSIMA/access-om2` as at 2020-04-21 and cloned to Gadi at `/g/data/v45/pcl900/access-om2/access-om2-gadi-scaling-01`

The `gadi-scaling-01` branch is based on the `master` branch of `COSIMA/access-om2` as at 2020-04-21. The main change from 2020-04-21 is to add configurations and modify scripts to match all of the CICE executables that needed to be built to test the scaling of ACCESS-OM2. In particular, the `src/cice5` submodule uses the `gadi-scaling-01` branch of the `penguian/cice5` repository, and the `control/01deg_jra55v13_ryf9091` submodule has been added, based on the `penguian/01deg_jra55v13_ryf9091` repository.

Compilation

Within the source code directory, the build script is at `install.sh`

Module versions used for compilation are defined in the following files:

`src/cice5/bld/config.nci.auscom_${ZNTASK}_${DIVX}x${DIVY}.3600x2700`

`src/libaccessom2/build_on_gadi.sh`

`src/libaccessom2/build/oasis3-mct-prefix/src/oasis3-mct/util/make_dir/config.gadi`

`src/mom/bin/environs.nci`

and are:

`intel-compiler/2019.5.281, netcdf/4.7.1,` and `openmpi/4.0.2`

Compilation flags are defined in the following files, and are

`src/cice5/bld/Macros.nci`

> `-c -r8 -i4 -traceback -w -fpe0 -ftz -convert big_endian -assume byterecl`
> `-check noarg_temp_created` **`-fp-model precise -fp-model source -align all`** `-g3 -O2`
> **`-axCORE-AVX2 -debug all -check none -qopt-report=5 -qopt-report-annotate`**
> `-assume buffered_io -pthread`

`src/libaccessom2/build/datetime-fortran-prefix/src/datetime-fortran/CMakeLists.txt`

> `-O1`

`src/libaccessom2/build/oasis3-mct-prefix/src/oasis3-mct/util/make_dir/make.nci`

> `-c -O2` **`-fp-model precise -fp-model source -align all`** `-g3 -O2` **`-axCORE-AVX2`**
> **`-debug all -check none -qopt-report=5 -qopt-report-annotate`** `-assume buffered_io`
> `-ip -pthread`
>
> and in some cases
>
> `-r8 -i4 -traceback -fpe0 -convert big_endian -fno-alias -ip`
> `-check noarg_temp_created` **`-fp-model precise -fp-model source -align all`** `-g3 -O2`
> **`-axCORE-AVX2 -debug all -check none -qopt-report=5 -qopt-report-annotate`**
> `-assume buffered_io -c -pthread`

`src/libaccessom2/CMakeLists.txt`

> `-r8 -fpe0` **`-fp-model precise -fp-model source -align all`** `-traceback -g3 -O2`
> **`-axCORE-AVX2 -debug all -check none -qopt-report=5 -qopt-report-annotate`**
> `-module include -c -pthread`

`src/mom/bin/mkmf.template.nci`

> `-fno-alias -safe-cray-ptr -fpe0 -ftz -assume byterecl -i4 -r8 -traceback -nowarn`
> `-check noarg_temp_created -assume nobuffered_io -convert big_endian`
> `-grecord-gcc-switches` **`-align all`** `-g3 -O2` **`-axCORE-AVX2 -debug all -check none`**
> **`-qopt-report=5 -qopt-report-annotate`** `-c -pthread`
>
> and in some cases also the flags
>
> `-fpp -Wp,-w`
>
> and for `mom/exec/nci/lib_FMS:`
>
> `-O2 -debug minimal` **`-axCORE-AVX2 -qopt-report=5 -qopt-report-annotate`** `-c -pthread`

Here the bolding indicates common reproducibility, architecture, optimization and report flags.

The build script compiled the software into appropriate subdirectories of the following directories, complete with annotated listings, as indicated by the compilation flags above:

> `src/cice5/build_auscom_${ZNTASK}_${DIVX}x${DIVY}.3600x2700_${NTASK}p`
>
> `src/libaccessom2/build/CMakeFiles`
>
> `src/libaccessom2/build/oasis3-mct-prefix/src/oasis3-mct/Linux/build/lib`
>
> `src/mom/exec/nci`

The build script then copied the resulting executables into the subdirectory `bin`.

Experiment runtime configurations

The configurations use the `config.*` branches of the penguian/01deg_jra55v13_ryf9091 repository on GitHub. This repository is forked from aidanheerdegen/01deg_jra55v13_ryf9091

based on `/home/157/amh157/payu/01deg_jra55v13_ryf9091/` and copied to Gadi under the directory `/g/data/v45/pcl900/access-om2/access-om2-gadi-scaling-cosima/control/`.

Each experiment is contained in a subdirectory of `control` with the naming convention `01deg_jra55v13_ryf9091.${LAYOUT}.${CICE_NCPUS}.${ZNTASK}_${XDIV}x${YDIV}`

Each experiment consists of two runs: an initial run for the month of January 1900 to create restart files, and a continuation run for the month of February 1900. These two runs use different `accessom2.nml` files, called `accessom2.nml.init` and `accessom2.nml.cont`, which differ by time step. The file `accessom2.nml` is a symbolic link to one of these two files.

The configuration differences from the version at `/home/157/amh157/payu/` are:

`accessom2.nml.init`

- In namelist `accessom2_nml`:
    a. Changed `ice_ocean_timestep` from `540` to `300`
- In namelist `date_manager_nml`:
    a. Set `allow_forcing_and_exp_date_mismatch = .true.`
    b. Changed restart_period from `0, 3, 0` to `0, 1, 0`

`accessom2.nml.cont`

- As per `accessom2.nml.init` except `ice_ocean_timestep = 540`

`config.yaml`

- Extensive changes. See ‘Example model configuration’ below.

`ocean/data_table`

- Commented out all diagnostics.

`ocean/input.nml`

- Changed `layout` from `80,75` to one of `60,50; 80,75; 120,100` or `160,100`

`ocean/INPUT`

- The file `ocean_mask_table` is a symbolic link to a mask file with naming convention `mask_table.${MASKED_CPUS}.${LAYOUT}`

`ice/cice_in.nml`

- Changed `nprocs` from `799` to one of `399`, `799`, `1599` or `3199`

Example model configuration

This example is from `01deg_jra55v13_ryf9091.080x075.0399.0361_045x045/config.yaml`
Long lines have been split.

```
# Model configuration
name: common
model: access-om2
input: /g/data/ik11/inputs/access-om2/input_08022019/common_01deg_jra55
submodels:
    - name: atmosphere
      model: yatm
      exe: /scratch/fp0/pcl900/access-om2/access-om2-gadi-scaling-01/bin/yatm_c30e907.exe
      input: /g/data/ik11/inputs/access-om2/input_08022019/yatm_01deg
      ncpus: 1

    - name: ocean
      model: mom
      exe: /scratch/fp0/pcl900/access-om2/access-om2-gadi-scaling-01/bin/
           fms_ACCESS-OM_75e267c_libaccessom2_c30e907.x
      input:
          - /scratch/fp0/pcl900/access-om2/access-om2-gadi-scaling-cosima/control/
            01deg_jra55v13_ryf9091.080x075.0399/ocean/INPUT
          - /g/data/ik11/inputs/access-om2/input_08022019/mom_01deg
      ncpus: 4358

    - name: ice
      model: cice5
      exe: /scratch/fp0/pcl900/access-om2/access-om2-gadi-scaling-01/bin/
           cice_auscom_0361_045x045_3600x2700_361p.exe
      input: /g/data/ik11/inputs/access-om2/input_08022019/cice_01deg
      ncpus: 399
```

Experiments

Each experiment consists of two runs, an initial run with a time step of 300 seconds for the month of January 1900 to create restart files, and a continuation run with a time step of 540 seconds for the month of February 1900. This is 160 steps per day for 28 days, giving 4480 steps.

The output of each experiment is in a subdirectory of `/g/data/v45/pcl900/access-om2/archive` with the same naming convention as the experiment configuration directories, that is

`01deg_jra55v13_ryf9091.${LAYOUT}.${CICE_NCPUS}.${ZNTASK}_${XDIV}x${YDIV}`

The output of the continuation runs is in one or more of the subdirectories `output001`, `copied.output001`, or `copy2.output001`. These last two subdirectory names indicate that the output file was copied from a different experiment directory structure.

Analysis

The directory `/g/data/v45/pcl900/access-om2/analysis` contains the Python scripts and the Jupyter notebook `plot_access_om2_scaling.ipynb` used for the analysis.

Instructions to reproduce the executables and experiments

The Bash script `/g/data/v45/pcl900/access-om2/install-gadi-scaling-01.sh` contains:

```
#!/bin/bash
set -uxe
access_om2_dir=${1:-access-om2-gadi-scaling-01}
archive_dir=${2:-archive}
DRYRUN=""
$DRYRUN git clone --single-branch --branch gadi-scaling-01 --recurse-submodules \
git@github.com:penguian/access-om2.git "$access_om2_dir"
$DRYRUN pushd "$access_om2_dir"
$DRYRUN   git submodule update --init --remote src/cice5
$DRYRUN   git submodule add -b gadi-scaling-01 \
https://github.com/penguian/01deg_jra55v13_ryf9091.git \
control/01deg_jra55v13_ryf9091
$DRYRUN   git submodule update --init --remote control/01deg_jra55v13_ryf9091
$DRYRUN   pushd control/01deg_jra55v13_ryf9091
$DRYRUN     ./setup_experiments.sh
$DRYRUN   popd
$DRYRUN   ./install.sh
$DRYRUN   pushd control/01deg_jra55v13_ryf9091
$DRYRUN     ./correct_exes.sh
$DRYRUN   popd
$DRYRUN popd
```

In this script,

- `access_om2_dir` is set to argument 1, defaulting to `access-om2-gadi-scaling-01`, and is the directory where the executables and experiment configurations are to be installed,
- `archive_dir` is set to argument 2, defaulting to `archive` and is the directory where the archive of experiment results is to be stored.

To reproduce the executables and experiment configurations, run this script.

References

[1] ACCESS-OM2 v1.0: a global ocean-sea ice model at three resolutions.

[2] A technical description of ACCESS-OM2, The Consortium of Ocean-Sea Ice Modelling in Australia’s global ocean and sea ice model.

[3] COSIMA Models.