natESM

Sprint documentation #3

Booster for FESOM 2.1 Sprint

Contact: info@nat-esm.de

Published on 20.07.23 on https://www.nat-esm.de/services/accepted-sprints

1 Summary

The goal of the sprint was to prepare the AWI Finite-Element/volumE Sea ice-Ocean Model (FESOM) for the OpenACC programming standard on JUWELS Booster and Levante. During the sprint, we ported the components for tracer advection, sea-ice dynamics, and the pressure and density-dynamics calculations from FESOM to the GPU using OpenACC. For the tracer advection, we developed a second ported version with first-level GPU optimizations and loop recording. The previous existence of dwarfs with modular specialized functionality in FESOM proved very helpful to the porting efforts. Performance evaluations show that in a node-to-node comparison, around 10 times reduction in kernel computation time can be expected when moving to the GPU, employing decorations and data movement optimization only. If additional loop recording is possible, we expect around 20 times reduction in compute time.

2 General information

Start and end date:	08.11.22 – 20.04.23
Intended period:	6 months
Responsible RSE:	Wilton Jaciel Loch (DKRZ)
Responsible scientist:	Dmitry Sidorenko (AWI)

FESOM (Finite-Element/volumE Sea ice-Ocean Model) is a multi-resolution sea ice-ocean model that solves the equations of motion on unstructured meshes. The model is developed and supported by researchers at the Alfred Wegener Institute, Helmholtz Centre for Polar and Marine Research (AWI), in Bremerhaven, Germany. The model has been successfully used in many ocean and sea-ice studies.

FESOM is written in Fortran 90 with some C/C++ code for providing bindings to third-party libraries. The code currently employs both the distributed- (MPI) and shared-memory (OpenMP) parallelizations to run on HPC systems.

3 Sprint objectives

The general focus of the sprint revolved around porting FESOM 2.1 code to GPUs, specifically targeting support for Levante and JUWELS Booster accelerator infrastructures. The initial work was to be directed towards the porting of components of the model with an existing dwarf for their functionalities, namely the tracer advection and sea-ice dynamics. Porting of other parts of the model was to be conducted upon the conclusion of the work on the dwarfs. Exploratory investigations of different directive-based approaches for the parallelization/porting of the model

were also among the objectives. Finally, test runs were to be performed both on Levante and on the JUWELS Booster to verify the execution of the ported code.

4 Procedure and insights

4.1 Technical Approach / procedure

The approach for selecting the next steps of technical work was defined to be iterative, in direct communication with AWI scientists. In this sense, smaller packages of work to be performed were agreed upon in meetings with AWI scientists and once such smaller goals were reached, a new meeting was held to present the current results and choose the next direction.

From the first meeting, an initial goal of porting the tracer advection was defined. This was selected due to the high computational time percentage taken up by the tracer advection in the full model execution and the existence of a dwarf for this portion of the code. A dwarf is a granular application that comprises and executes only a specific functionality of the full model, allowing better isolation, easier testing, and increased development speed.

After the successful porting of the tracer advection dwarf, the following meeting resulted in the start of a first-level optimization for it, which upon finalization led to the first performance evaluation, presented to AWI scientists with execution time and speedup data for the two resulting ported versions.

Concerns regarding readability and major code changes for highly optimized GPU versions steered the next work phase towards the porting of the second existing dwarf, focused on the sea-ice dynamics. This following development cycle also comprised an exploratory comparison of OpenMP and OpenACC multicore performances, which similarly emerged from discussions regarding code readability and performance portability, fundamentally trying to answer the question of whether OpenACC could act as a suitable replacement for OpenMP on multicore execution targets.

In the same period, a third development was made towards the creation of an experimental performance portability layer, which would allow both CPU- and GPU-optimized versions with a single, more abstract code base. Results were presented in a follow-up meeting, resulting in the decision of the last porting cycle to be focused on the dynamics, specifically pressure and density calculation subroutines.

4.2 General Insights

Before the start of the sprint, there was already work done on evaluating the potential of FESOM for executing effectively on GPUs. The work investigated individual tracer advection subroutines and showed high performance benefits of employing the accelerators. We could further confirm these results by the data we obtained during the sprint, not only for the tracer advection but also for the sea-ice dynamics.

Another valuable piece of information gathered was regarding OpenACC multicore performance and how it compares to OpenMP. The results show that for most parallel kernels OpenACC performance is comparable to OpenMP, and in certain cases only marginally worse. For the implementation of critical regions in parallel kernels, OpenACC showed superior performance. However, the compiler support for OpenACC when targeting a multicore setting seems to be less mature than the OpenMP counterpart.

The collected data also showed a significant degradation of MPI communication time when employing direct GPU-to-GPU data exchanges. The current hypothesis is that this is due to a combination of direct GPU communication and the use of MPI derived datatypes, which when employed together have been observed to cause a similar magnitude of performance degradation in other projects.

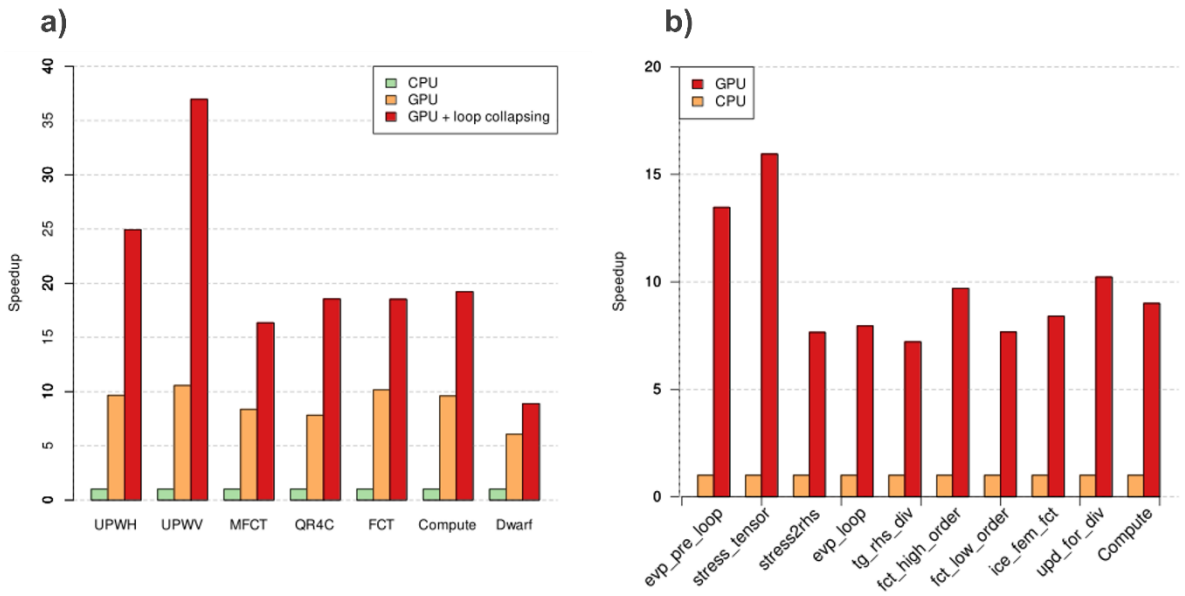

Figure 1: GPU speedups in a node-to-node comparison on Levante for a) tracer advection and b) sea-ice dynamics. The average compute speedup achieved for the tracer advection is around 10 times for the ported version and around 20 times with the addition of loop collapsing. For the sea ice dynamics, the average compute time GPU speedup is around 9 times.

The previous existence of dwarfs with modular specialized functionality in FESOM proved very helpful to the porting efforts, and similar strategies should be expanded to the rest of the model, not only simplifying porting but the general development as well.

5 Results

Various components from FESOM were ported to the GPU using OpenACC. These include the tracer advection, the sea-ice dynamics and the pressure and density dynamics calculations. For the tracer advection, a second ported version with first-level GPU optimizations - most importantly loop reordering - was also developed.

Performance evaluations were conducted to assess the benefits of the ported versions. Results show that in a node-to-node comparison around 10 times reduction in kernel computation time can be expected when moving to the GPU, employing decorations and data movement optimization only. If additional loop reordering is possible, as in the tracer advection case, the benefits of adding this change average around 20 times reduction in compute time. For the complete execution of such sections, with the difference mainly being the inclusion of communication time, the reductions average around 5 times, for only ported and data movement optimized versions, and around 10 times with the addition of loop reordering.

The specific results of the sprint are the following:

- Porting of the tracer advection code.
- Porting of the sea-ice dynamics code.
- Porting of the pressure and density calculation code.
- Creation of an experimental first-level optimized version of the tracer advection code, to be used as an example for further optimizations in the rest of the model.
- Full model experimental run with partial GPU execution of ported sections on Levante and JUWELS Booster.

All the code created during the sprint is stored within the FESOM repository on Github. Initial ported versions of the tracer advection and sea-ice dynamics were merged into the main branch of

FESOM and are now part of the distribution version. Additional work done is available on separate branches of the repository, as they contain more intrusive or experimental changes.

Performance improvement data for the ported versions are available in Figure 1 relating to the speedups obtained respectively for the tracer advection and sea-ice dynamics dwarfs. Total dwarf speedup is omitted for the sea-ice dynamics, as the MPI communication degradation severely hurts its final performance.

6 Conclusions and Outlook

The work done during the sprint represents the first steps towards the full GPU porting of FESOM and stands as a reference for further work to be conducted by AWI scientists as well as other interested parties. A set of preliminary evaluations were conducted with the ported version, demonstrating its potential to achieve increased performance results when compared to pure CPU parallel execution.

As future work, several tasks can be listed, such as:

1. The execution of the ported version of FESOM in different supercomputers, among which perhaps the most important being LUMI, as it presents a different hardware and software ecosystem from the ones currently tested and is a major host for future FESOM projects;
2. Investigate the MPI communication time degradation issues discovered during the porting, which could be a limiting factor for achieving additional performance benefits in future full model GPU runs;
3. Integration of the first level optimized version of the tracer- advection code into the main distribution branch of FESOM, along with the necessary definition of a strategy and corresponding software infrastructure for addressing performance portability between CPU and GPU executions and;
4. Conduct novel performance evaluation experiments to assess the scalability aspects of the ported code, obtaining information such as scaling curves and resolution limits for effective GPU execution.

7 References

A full documentation can be found on:
https://gitlab.dkrz.de/natESM/natesm_sprints_documentation/-/wikis/home