Is the YAXT communication library as portable as
the OASIS community coupler ?

E. Maisonnave

WN/CMGC/23/177

Abstract

The YAC Kritsikis-et-al interpolation is supposed to be soon made available in the OASIS coupler. To preserve the portability and simplicity of use of the community software, we evaluate the characteristics in one YAC critical part: the YAXT communication library. We found it fully compatible with four supercomputers of our community (CRAY, Atos-Intel, Atos-AMD, Lenovo), plus one experimental desktop platform (Raspberry Pi), and three different compilers and MPI libraries. Even if additional studies would be necessary to check the level of performance of YAXT as a communication library for the whole OASIS software, it is already possible to conclude that its portability and the portability of the original SCRIP interpolation package, are the same

Table of Contents

The OASIS3-MCT coupler [Craig et al. 2017] is the keystone of dozens of geophysical models, since this software offers for decades an efficient, portable and perennial way to exchange boundary conditions at runtime. The resulting coupled systems are many. At a first level of complexity, an added module can be seen as a more complex boundary layer of the original model, e.g. a wave model added to an ocean in [Couvelard et al. 2020]. At larger scale, a collection of inter-dependent models can make complex systems such as full Earth System Models [Séférian et al. 2019].

The OASIS3-MCT software mainly relies on two included libraries: the SCRIP interpolation package [Jones 1999] and the MCT communication pattern definition [Larson et al. 2005]. The robustness of both contributions is unquestionable and is probably one explanation of the OASIS coupler longevity. However, algorithmic limitations from one hand and the coming end of an active support on the other hand makes mandatory at medium term the replacement of both OASIS pillars. In that perspective, existing solutions are evaluated. In particular, the [Kritsikis et al. 2017] "conservative" interpolation, in its YAC[1] [Hanke et al. 2016] version, that is supposed to be soon included in the list of the online OASIS interpolations.

We assume that the OASIS3-MCT relevance is notably due to the stability and the efficiency of its functions. In order to keep making available our coupler to its community, any modification or substitution made in this software should not damage these characteristics. For that reason, a special care to the retro-compatibility, portability and simplicity of use must be preserved by the inclusion work of the YAC-Kritsikis-et-al interpolation. In this document, we will focus on the YAC communication library (YAXT[2]) portability. On a larger perspective, an evaluation of the characteristics of the communication pattern and MPI exchanges could be relevant in the perspective of the MCT library substitution.

The YAXT library

The YAC-core part of the DKRZ software, which ensures among other things the computation and the use of the Kritsikis-et-al interpolation weights, has been designed for running in parallel on a dedicated set of processors (encapsulated by a communicator) and has efficient redistributing capacities provided by the YAXT communication layer. Since YAC cannot be used without YAXT, YAXT is one mandatory component that must be tested in an evaluation of the YAC robustness.

YAXT is a communication layer on top of the MPI library. It is implemented in C language, with a FORTRAN interface. It is regularly upgraded and freely available from a server[3]. At starting, the library generates a mapping table between source and target decomposition

1 https://dkrz-sw.gitlab-pages.dkrz.de/yac/
2 https://swprojects.dkrz.de/redmine/projects/yaxt
3 https://swprojects.dkrz.de/redmine/projects/yaxt/wiki/Downloads
yaxt-0.10.0.tar.gz (1411269 bytes)

(communication pattern) and a specific redistribution objects, based on MPI derived datatypes, to perform the exchanges. YAXT is used by YAC for both interpolation weight generation and the coupling field exchange. To calculate the communication patterns of exchanges, more efficient than a simple MPI "all to all", the libraries rely on a rendezvous algorithm [Pinar & Hendrickson 2001] based on a distributed directory of global indices. YAXT not only optimises the exchange pattern but also reduces the communication number, grouping the exchanged arrays on MPI derived datatypes, supposed to reduce the array copies and pack-unpack operations.

YAXT portability

YAXT portability is already guaranteed and regularly tested by the ICON community, since the derived Earth System Model [Crueger et al 2018] includes the communication library. However, we propose to increase the number of tested platforms by accessing selected machines made available during the recent OASIS dedicated support program [Maisonnave 2022]. On these platforms (see Table 1), OASIS based coupled models have been installed and are still in operation. In order to put YAXT in real OASIS coupling conditions, the compilers and MPI libraries used by our OASIS coupled systems are also chosen to compile and test the YAXT library.

Machine name	Location	Vendor/CPU	Compiler	MPI
belenos[4]	Météo-France Toulouse	Atos/AMD Rome	Intel 2018.5.274	Intel MPI 2018.5.274
fram[5]	Sigma2, NRIS, Norway	Lenovo/Intel Broadwell	gcc 11.2.0	OpenMPI 4.1.1
			gcc 12.3.0	OpenMPI 4.1.5
lise[6]	HLRN Berlin	Atos/Intel Cascade	gcc 9.3.0	OpenMPI 4.1.4
piz daint[7]	CSCS Lugano	CRAY/Intel Haswell	Cray PE 2.7.10	Cray MPICH/7.7.18
raspberrypi[8]	*personal resources*	*RP/ARM Cortex*	*gcc 10.2.1*	*OpenMPI 4.1.0*

Table 1: Compiling environment of the four supercomputers and one desktop computer hosting the YAXT-OASIS compatibility test

The YAXT compatibility test consists in evaluating how much work is required to compile the library and complete a successful execution of the 40 pre-defined tests of the suite.

On the 4 tested supercomputers, YAXT was compiled at first attempt. This seems to confirm the availability and similarity of the MPI derived datatypes included in the different MPI libraries.

Similarly, the test suite was fully and immediately successful on 3 platforms. On the CRAY CSCS machine, a cross compiling makes the executable launching more difficult and the only noticed failure[9] is probably due to a mistake in configuring the testing rather than a porting

4 https://www.top500.org/system/179853/
5 https://documentation.sigma2.no/hpc_machines/fram.html#fram
6 https://www.zib.de/research_services/supercomputing
7 https://www.cscs.ch/computers/piz-daint
8 https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/
9 test_xmap_all2all_fail_run

issue. On the Météo-France facility, we failed in trying to do the test with another compiler (GNU) than the one used for our coupled system. But we probably also would have failed in compiling our reference OASIS coupled system.

This short experience tends to prove the large portability of the YAXT package and surely proves, at least for 4 of them, its compatibility with coupled model compiling environments of our community.

We tried to go further by testing the YAXT package portability on a processor rather made for home automation than supercomputing: a Raspberry Pi. On this kind of personal computer, a dedicated Debian operating system and useful programming tools (such as a gcc compiler and OpenMPI) can be installed without any special expertise. The particularity of this CPU is its simplicity: a small quad-core ARM Cortex-A72, rather included in phone cells than supercomputers, which questions the portability of the YAXT library on an unusual kind of hardware. Compiled without any problem, the install procedure fails at testing the MPI programs that require more than 4 MPI processes. With the OpenMPI option that allows nodes to be oversubscribed[10], the 40 pre-defined tests are successful. This overfulfills our expectation of the YAXT library portability.

Evaluating YAXT performance ?

The popularity of the OASIS software probably relies in its accessibility, but a lack of performance would certainly forbid its use in many cases. This is why, beside the portability check we described, some performance testing of the YAXT communication library would be useful to fully quantify its OASIS compliance.

According to its authors, the most time-consuming part in YAXT is the parallel computation of the Pinar & Hendrickson communication pattern (not used during the weight computations). The rest of YAXT routines are meant to be efficient, as confirmed by the various simulations performed with the ICON climate model.

However, even if the MPI derived datatypes are available in common MPI implementations for several decades, their performance seems to be still implementation dependent. In [Xiong et al. 2018], a benchmark of several kind of datatypes, several compilers and two interconnection networks shows that the performance (measured with a ping pong example) with the same derived datatype can vary by up to 3x. Vectorisation or additional hardware can contribute to reduce the observed gap. But even if MPI derived datatypes are meant to have a positive effect on latency and race conditions on the interconnection network, it has been observed that the memory footprint could be significantly higher than with classical MPI exchanges and reduce in some cases the overall exchanges performance.

As a consequence, we expect that the kind of exchanged data would have an influence on how the YAXT and the MPI derived datatypes structures YAC exchanges. One can expect that (i) a huge amount of small or (ii) large but non continuous messages would be the two ideal

10 --oversubscribe

test cases that could reveal a difference.

At this point, it is necessary to define what "performance" means for the OASIS library. As already observed, the speed of interpolation weight computations is not a bottleneck for most of the community users. The hybrid MPI-OpenMP parallel SCRIP library [Piacentini et al. 2018] proves its efficiency and is supposed to perform more and more effectively with the expected increase of core number per CPU node. This is the coupling field exchange phase that is by far the most demanding part for performance, particularly at scale, with high horizontal resolution configurations (1M cores and 1kB messages) or for 3D exchanges on a small amount of resources (1k cores and 1MB messages).

Unfortunately, no real case is now available to perform such measurements. At the moment, is currently under development a demonstrator of the YAC-Kristikis-et-al interpolation, called in an OASIS like FORTRAN program, for commonly used grids. This demonstrator could serve to test the level of reliability of the MPI derived datatypes and of the whole YAXT-YAC interpolation part and, to some extend, their performance.

Conclusion

Even if additional studies would be necessary to check the level of performance of YAXT as a communication library for the whole OASIS software, it is already possible to conclude that its portability and the portability of the original SCRIP interpolation package, are the same. However, since it have been decided with DKRZ that YAXT and YAC sources distribution will remain independent from the OASIS distribution package, a continuous checking of version compatibility would be necessary during the rest of the OASIS library lifespan.

The author gratefully acknowledges the computing time granted by the Resource Allocation Board and provided on the supercomputer Lise at NHR@ZIB as part of the NHR infrastructure. The calculations for this research were conducted with computing resources (frontend) thanks to Joakim Kjellsson at GEOMAR/Kiel University. He also acknowledges the computing time (frontend) granted by Sigma2 - NRIS - the National Infrastructure for High Performance Computing and Data Storage in Norway thanks to Einar Olason (NERSC), Météo-France (frontend) thanks to CERFACS' base allocation and CSCS Swiss National Supercomputing Center thanks to Edouard Davin (Wyss Academy for Nature). The author also acknowledges the help of Alastair Mc Kinstry (ICHEC) for providing the Debian version of our scientific tools, Guillaume Mercier (INRIA) for his precious MPI derived datatype bibliography, Moritz Hanke (DKRZ) for guiding the YAXT testing and Andrea Piacentini for his advices

References

Couvelard, X., Lemarié, F., Samson, G., Redelsperger, J.-L., Ardhuin, F., Benshila, R., & Madec, G., 2020: Development of a two-way-coupled ocean–wave model: assessment on a global NEMO(v3.6)–WW3(v6.02) coupled configuration, *Geosci. Model Dev.*, **13**, 3067–3090, https://doi.org/10.5194/gmd-13-3067-2020

Craig A., Valcke S., Coquart L., 2017: Development and performance of a new version of the OASIS coupler, OASIS3-MCT_3.0, *Geosci. Model Dev*, **10**, pp. 3297-3308, https://doi.org/10.5194/gmd-10-3297-2017

Crueger, T., Giorgetta, M. A., Brokopf, R., Esch, M., Fiedler, S., Hohenegger, C., et al., 2018: ICON-A, the atmosphere component of the ICON Earth system model: II. Model evaluation. *Journal of Advances in Modeling Earth Systems*, **10**, 1638–1662. https://doi.org/10.1029/2017MS001233

Hanke, M., Redler, R., Holfeld, T. & Yastremsky, M., 2016: YAC 1.2.0: new aspects for coupling software in Earth system modelling, *Geosci. Model Dev*, **9**, 2755-2769. DOI:10.5194/gmd-9-2755-201

Jones, P., 1999: Conservative remapping: First-and second-order conservative remapping, *Mon. Weather Rev.*, **127**, 2204–2210

Kritsikis, E., Aechtner, M., Meurdesoif, Y., & Dubos, T., 2017: Conservative interpolation between general spherical meshes, *Geosci. Model Dev.*, **10**, 425–431, https://doi.org/10.5194/gmd-10-425-2017

Larson, J., R. Jacob, & E. Ong, 2005: The Model Coupling Toolkit: A New Fortran90 Toolkit for Building Multiphysics Parallel Coupled Models. *Int. J. High Perf. Comp. App.*, **19(3)**, 277-292

Maisonnave, E., 2022: A climate community coupler: OASIS , Working Note, **WN/CMGC/22/168**, CECI, UMR CERFACS/CNRS No5318, France

Piacentini, A., Maisonnave, E., Jonville, G., Coquart, L., & Valcke, S., 2018: A parallel SCRIP interpolation library for OASIS , Working Note, **WN/CMGC/18/34**, CECI, UMR CERFACS/CNRS No5318, France

Pinar, A., & Hendrickson, B., 2001 : Communication Support for Adaptive Computation, *Proc. SIAM Conf. on Parallel Processing for Scientific Computing*

Séférian, R., Nabat, P., Michou, M., Saint-Martin, D., Voldoire, A., Colin, J., et al, 2019: Evaluation of CNRM Earth-System model, CNRM-ESM2-1: role of Earth system processes in present-day and future climate. *Journal of Advances in Modeling Earth Systems*, **11**, 4182–4227. https://doi.org/10.1029/2019MS001791

Xiong, Q., Bangalore, P. V., Skjellum, A. & Herbordt., M., 2018: MPI Derived Datatypes: Performance and Portability Issues. In Proceedings of the 25th European MPI Users' Group Meeting (EuroMPI '18). Association for Computing Machinery, New York, NY, USA, Article 15, 1–10. https://doi.org/10.1145/3236367.3236378