PRACE Preparatory Access Type A

Final Report

Peer-Review Office – V0.1 – 22/11/2018

1 General Information

Type of project granted: Preparatory Access Type A – Code scalability and performance.

Tests to obtain the relevant parameters necessary when applying to future PRACE calls for Project Access.

1.1 Proposal ID

2010PA5295

1.2 Period of access to the PRACE facilities

01 April 2020 – 02 June 2020

1.3 Name of the PRACE facility assigned

Joliot-Curie SKL and Joliot-Curie ROME

2 Project information

2.1 Project name to which the tested code corresponds

High Performance Computing in Naval Hydrodynamics (HPCNH)

2.2 Research field

- ☐ Biochemistry, Bioinformatics and Life sciences
- ☐ Chemical Sciences and Materials
- ☐ Earth System Sciences
- ☐ Economics, Finance and Management
- ☒ Engineering
- ☐ Fundamental Physics
- ☐ Linguistics, Cognition and Culture
- ☐ Mathematics and Computer Sciences
- ☐ Physiology and Medicine
- ☐ Universe Science

☐ Fundamental Constituents of Matter

2.3 Institutions and research team members

Riccardo Broglia, Ph.D. CNR-INM, National Research Council of Italy, Institute of Marine Engineering

Antonio Posa, Ph.D., CNR-INM, National Research Council of Italy, Institute of Marine Engineering

2.4 Summary of the project interest

This project aims at testing the scalability performance of an in-house Fortran code with MPI parallelization for simulations in naval hydrodynamics using the Large-Eddy Simulation methodology, coupled with an Immersed-Boundary technique. The code solves the filtered Navier-Stokes equations for incompressible flows, discretized via a numerical methodology achieving optimal conservation properties, well suited for the accurate simulation of turbulent flows. Scalability tests will be performed on a test case relevant to naval hydrodynamic applications, dealing with the DARPA suboff body, which is a notional submarine geometry developed for fundamental studies on underwater vehicles. An Immersed-Boundary methodology will be adopted for handling the body immersed within a cylindrical grid. The same solver was already successfully tested on several parallel clusters and this project is aimed at verifying its performance on Irene SKL and Irene ROME. The results of such tests will be utilized as guidance for selecting the most suitable machine for the adopted solver during the preparation of the proposal for the coming 21st PRACE Call for Project Access, where the same solver is going to be utilized for the simulation of turbulent flow problems in the field of naval hydrodynamics. Those tests will serve as evidence of scalability of the code, demonstrated in the past in HPC environment on different architectures, including resources provided in the framework of past PRACE Calls, as Marconi KNL and Irene KNL.

3 Main features of the code

3.1 Name of the code

Eddy

3.2 Type of the code distribution

Academic in-house Navier-Stokes solver

3.3 Computational problem executed

Filtered Navier-Stokes equations

3.4 Computational method

Finite-Differences on staggered grids; Subgrid scales turbulence modelling; Immersed boundary technique

3.5 Kind of parallelism used

MPI

3.6 Main libraries used, version and language. Did you use the /usr/local one?

Libraries: Fortran BLAS, LAPACK, HDF5.

On Joliot-Curie SKL:

- ✓ blas/mkl/17.0.6.256(default)
- ✓ lapack/mkl/17.0.6.256(default)
- ✓ hdf5/1.8.20(default)

On Joliot-Curie ROME:

- ✓ blas/mkl/19.0.5.281
- ✓ lapack/mkl/19.0.5.281
- ✓ hdf5/1.8.20(default)

The above libraries were those already available on Joliot-Curie.

3.7 Which other software did you use on the PRACE machines? Did you use some post-processing or pre-processing tools?

No additional software was required for performing scaling tests.

4 Compilation step

4.1 How is the program compiled?

Compilation was done using a Makefile.

4.2 Difficulties met to compile, if any, and how they were tackled.

Compilation was straightforward, thanks to the familiarity with the partition KNL of the same Joliot-Curie cluster. Flavors and modules allowed a quick generation of the environment required for compilation. We were able to compile our solver within a few minutes after access to both SKL

and ROME partitions.

4.3 Which version of the compiler and version of the MPI library did you use?

On Joliot-Curie SKL:

- ✓ fortran/intel/17.0.6.256(default)
- ✓ c++/intel/17.0.6.256(default)
- ✓ mpi/openmpi/2.0.4(default)

On Joliot-Curie ROME:

- ✓ fortran/intel/19.0.5.281(default)
- ✓ c++/intel/19.0.5.281(default)
- ✓ mpi/openmpi/4.0.2(default)

4.4 Did you use any tools to study the behaviour of your code?

We utilized timers implemented within our own code.

5 Execution step

5.1 How is the program launched?

Scaling tests were launched using batch scripts.

5.2 Difficulties met to launch the code, if any, and how they were tackled.

No difficulties were encountered in setting up the required batch scripts for both SKL and ROME partitions and launching our Navier-Stokes solver.

6 Communication patterns

If you know which are the main communication patterns used in your code configuration, select the ones from the mentioned below:

- ☒ Few point to point communications
- ☒ Few collective communications
- ☐ Barrier
- ☒ Reduction
- ☒ Broadcast
- ☐ Scatter/gather
- ☐ All to all

7 Results of the scalability testing

7.1 Summary of the obtained results from the scalability testing

Scalability was verified very satisfactory on both SKL and ROME partitions, even outperforming our expectations, at least up to about 4,000 cores, corresponding to the target size of our future production runs. Above this threshold a deterioration of the scaling performance was actually expected, because of the overall size of the computational problem, consisting in a cylindrical computational grid composed of about 3.6 billion points. Those cores count and grid size were designed for the simulations we planned in the framework of the 21st PRACE Call for Project Access. The present Preparatory Access project was indeed aimed at producing scalability results for that Call. A domain decomposition strategy was utilized, dividing the global cylindrical grid into smaller cylindrical subdomains, communicating between each other. For a number of subdomains above 4,000 the cost of communications between them becomes significant, compared to that of the computations performed within each subdomain. As a consequence, a decay of performance was expected for larger cores counts. Of course, increasing the size of the problem would allow pushing scalability to even larger numbers of cores, but this was out of the scope of our future simulations we have designed in the framework of the 21st PRACE Call for Project Access.

7.2 Images or graphics showing results from the scalability testing

Strong and weak scalability tests were conducted on both Joliot-Curie SKL and Joliot-Curie ROME. Results of these tests were also reported in the proposal 2020225357 submitted to PRACE in the framework of the 21st Call for Project Access.

The computational problem considered for the scaling tests deals with the DARPA suboff body. This is a notional submarine geometry, often considered in computations and experiments, for fundamental analyses of the flow over underwater vehicles and for testing the accuracy of the numerical solvers utilized in naval hydrodynamics. The geometry of the DARPA suboff is shown in Figure 1.

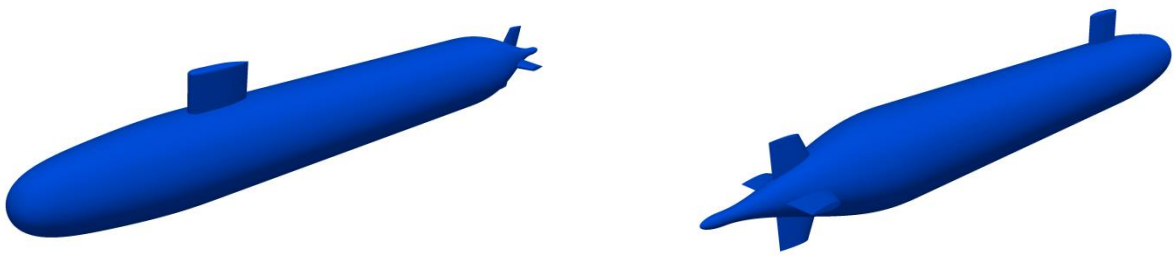

Figure 1. Geometry of the DARPA suboff: views from upstream (left) and from downstream (right)

Scaling tests were carried out using as a reference a cylindrical computational grid composed of about 3.6 billion points. Of course, for weak scaling tests additional grids were generated from the

reference one. Specifically, the number of points of the grid was modified across the streamwise direction, in order to keep the computational burden per core roughly unchanged. About this point, it should be recalled that the parallelization of the computations is achieved in our solver via domain decomposition across the streamwise direction, which is oriented as the axis of the cylindrical computational domain, splitting the overall grid into cylindrical subdomains.

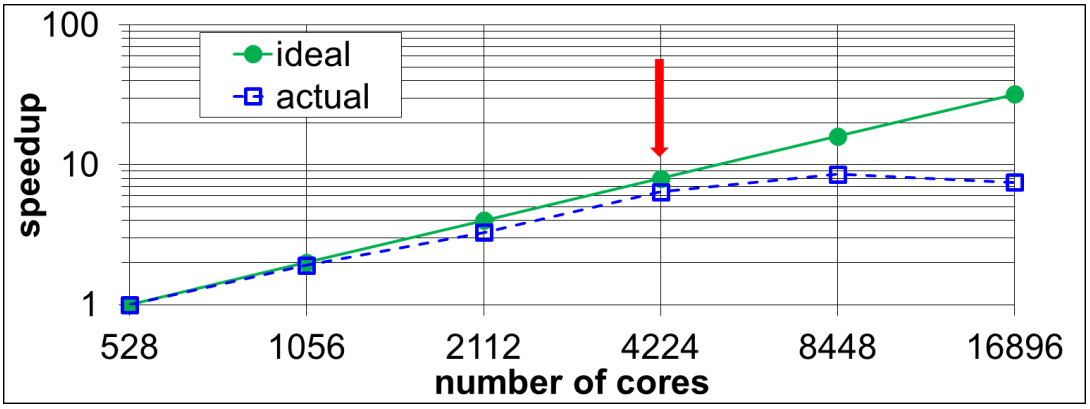

Figure 2. Results of strong scaling tests on Joliot-Curie SKL

Results of strong scaling tests on Joliot-Curie SKL are reported in Figure 2. On the vertical axis the speedup, relative to the computation on the smallest number of cores, which is 528, is reported. Smaller cores counts were not allowed, due to memory requirements. The red arrow points to the target size of the simulations (involving about 4,000 cores) we planned for the 21st Call for Project Access. Scaling is very satisfactory, although above that threshold performance declines, which is an obvious consequence of the size of the problem, which was indeed designed to run on about 4,000 cores. As discussed above, on larger computational grids strong scaling would go even further, but this was not in the scope of the present project and the simulations we devised for the 21st Call for Project Access.

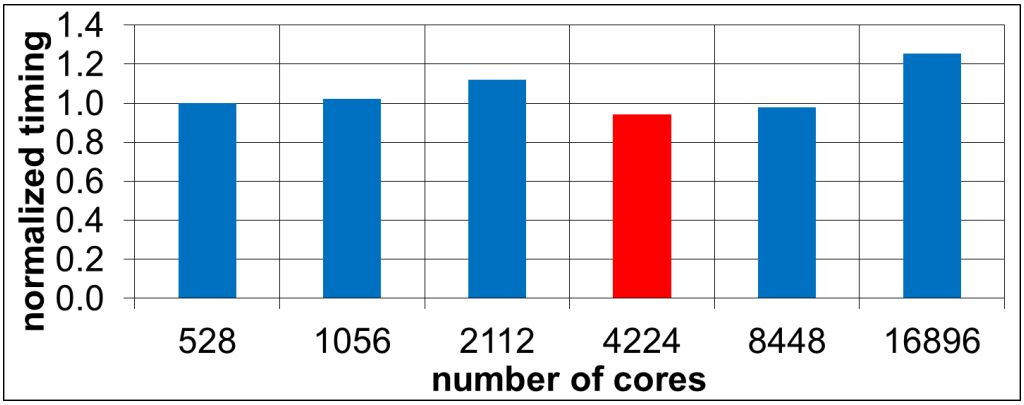

Figure 3. Results of weak scaling tests on Joliot-Curie SKL

Also weak scaling tests on Joliot-Curie SKL were carried out. Their results are illustrated in Figure 3. Again, the case of the reference grid was identified using red colour and all results were scaled based on the timings on the smallest number of cores, which is 528 also in this case. It should be noted that weak scaling tests on smaller cores counts were not allowed by the inherent features of our solver and the reference computational grid, preventing from decreasing further the number of grid points across the axial direction. Again, the scaling performance of our in-house solver on Joliot-Curie SKL was found very satisfactory.

Similar tests were conducted on Joliot-Curie ROME. Results are visualized in Figure 4 for strong scaling and in Figure 5 for weak scaling. It is demonstrated that the scalability of our solver is actually slightly better on Joliot-Curie ROME than on Joliot-Curie SKL. In addition, as reported in the tables below, the time-to-solution by our solver on the ROME partition was found smaller than that on the SKL partition. Therefore, we decided to apply for resources on the the former system within the 21st Call for Project Access.

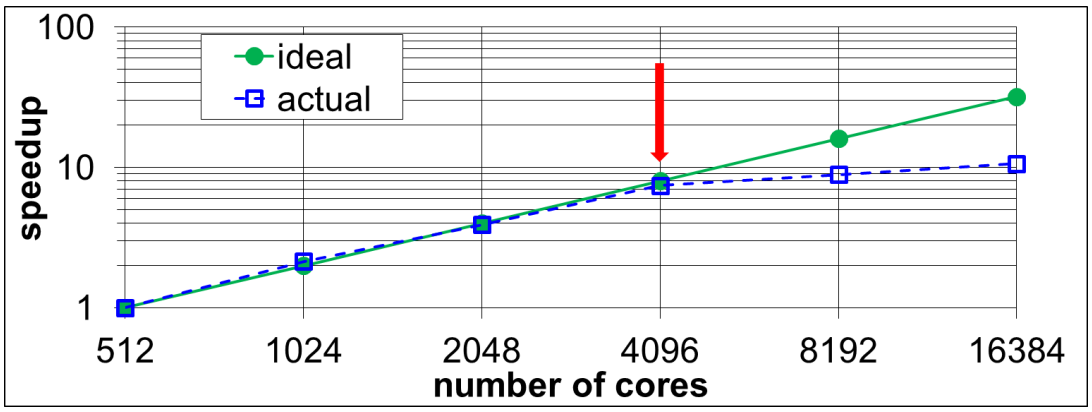

Figure 4. Results of strong scaling tests on Joliot-Curie ROME

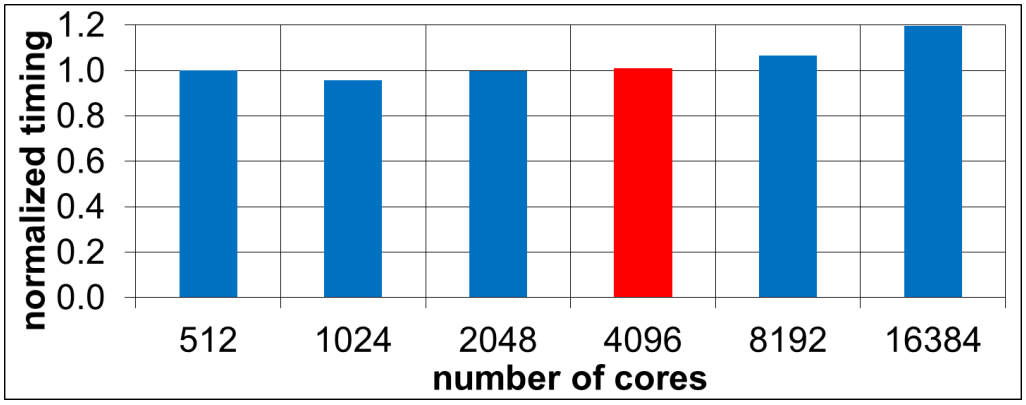

Figure 5. Weak scaling tests on Joliot-Curie ROME

7.3 Data to deploy scalability curves

A) Some typical user test cases

Tests on Joliot-Curie SKL

Number of cores	Wall clock time (s/step)	Speed-up vs the first one	Number of Nodes	Number of process
1056	24.9952	1.0000	22	1056
2112	14.6162	1.7101	44	2112
4224	7.4797	3.3417	88	4224

Tests on Joliot-Curie ROME

Number of cores	Wall clock time (s/step)	Speed-up vs the first one	Number of Nodes	Number of process
1024	20.0933	1.0000	8	1024
2048	11.0011	1.8265	16	2048
4096	5.7684	3.4833	32	4096

B) Strong scaling curve

Tests on Joliot-Curie SKL

Number of cores	Wall clock time (s/step)	Speed-up vs the first one	Number of Nodes	Number of process
528	47.8973	1.0000	11	528
1056	24.9952	1.9163	22	1056
2112	14.6162	3.2770	44	2112
4224	7.4797	6.4036	88	4224
8448	5.5892	8.5696	176	8448
16896	6.3925	7.4927	352	16896

Tests on Joliot-Curie ROME

Number of cores	Wall clock time (s/step)	Speed-up vs the first one	Number of Nodes	Number of process
512	42.9359	1.0000	4	512
1024	20.0933	2.1368	8	1024
2048	11.0011	3.9029	16	2048
4096	5.7684	7.4433	32	4096
8192	4.8258	8.8971	64	8192
16384	4.0425	10.6211	128	16384

C) Weak scaling curve

Tests on Joliot-Curie SKL

Number of cores	Wall clock time (s/step)	Speed-up vs the first one	Number of Nodes	Number of process
528	7.9533	1.0000	11	528
1056	8.1389	1.0233	22	1056
2112	8.9055	1.1197	44	2112

4224	7.4797	0.9405	88	4224
8448	7.7779	0.9780	176	8448
16896	9.9779	1.2546	352	16896

Tests on Joliot-Curie ROME

Number of cores	Wall clock time (s/step)	Speed-up vs the first one	Number of Nodes	Number of process
512	5.7172	1.0000	4	512
1024	5.4734	0.9574	8	1024
2048	5.6943	0.9960	16	2048
4096	5.7684	1.0090	32	4096
8192	6.0869	1.0647	64	8192
16384	6.8408	1.1965	128	16384

7.4 Publications or reports regarding the scalability testing

Results of scalability testing were reported in the proposal we submitted to PRACE in the framework of the 21st Call for Project Access: Broglia R. and Posa A. “WakePropRudd: Characterization of the wake of a propeller-rudder system”. April 2020.

8 Results on Input/Output

8.1 Size of the data and/or the number of files

Although I/O is typically not a computationally expensive component of our simulations, corresponding to only few percent of the overall computational cost, some tests were conducted to verify the I/O subroutines of our solver on both SKL and ROME partitions of Joliot-Curie. Checkpoints files were generated using parallel HDF5 libraries. The size of each of those files was in the order of 30GB. It should be noted that each test generated only 4 checkpoint files, one for each flow variable (velocity components and pressure), since I/O was carried out in parallel, with all processes writing on the same file. Some additional small files were produced during each scaling test for diagnostics, using serial instructions, in the order of a few tens of files with an overall size of a few Megabytes.

8.2 Please, let us know if you used some MPI-IO features.

Output files were generated using parallel HDF5 libraries.

9 Main results

Results were found very satisfactory on both SKL and ROME partitions, for both scalability and time-to-solution, outperforming our expectations. Our target within the present Preparatory Access was demonstrating scalability for the preparation of a proposal to submit in the framework of the 21st PRACE Call for Project Access, but also assessing the most suitable Joliot-Curie partition to our solver, where to ask for computational resources. Although performance was found similar between SKL and ROME architectures, the present results suggested preferring the latter option. In addition, performance of both SKL and ROME partitions in terms of time-to-solution gave us confidence about the possibility to tackle on the Joliot-Curie cluster even larger computational problems in our future studies.

10 Feedback and technical deployment

10.1 Feedback on the centres/PRACE mechanism

We are very satisfied by the opportunity provided by the mechanism of the PRACE Preparatory Access. In the past we were awarded computational resources in the framework of the 15th, 17th and 19th PRACE Calls for Project Access on Marconi KNL, CINECA and Joliot-Curie KNL, CEA. This project allowed us: i) demonstrating the suitability of our solver on different architectures, compared to KNL; ii) selecting the best option for our future studies. We utilized the results of the scalability tests reported in this document for the preparation of our proposal within the 21st Call for Project Access, asking for resources on the Joliot-Curie partition where our solver demonstrated the best performance. Of course, this is beneficial for the productivity of our research and the most efficient exploitation of PRACE resources. The allocations for performing scaling tests were made quickly available, giving us the possibility to produce the data required for the application to Project Access. Compilation and job submission on both SKL and ROME partitions of Joliot-Curie were very straightforward, thanks to the wealth of available compilers and libraries, the proper setup of the environment via flavors and modules and the level of details of the documentation provided by TGCC/CEA. The stability and performance of both partitions were found even better than our expectations and the scheduling of our jobs was very fast. This gave us the possibility of demonstrating within a short timeframe the suitability of our solver to perform major production runs on both Joliot-Curie SKL and Joliot-Curie ROME.

10.2 Explanation of how the computer time was used compared with the work plan presented in the proposal. Justification of discrepancies, especially if the computer time was not completely used.

We were able to perform all needed strong and weak scaling tests, using the allocated computational resources, within a few days. This result was achieved thanks to: i) our earlier familiarity with the KNL partition of the same Joliot-Curie cluster; ii) the lack of major issues for code compilation and in the preparation of submission scripts; iii) the efficiency of the scheduler on both SKL and ROME partitions in processing our queued jobs.

10.3 Please, let us know if you plan to apply for PRACE Project Access in the future? If not, explain us why.

It is very likely we will apply for both PRACE Preparatory and Project Access in the future. Actually, we recently applied for resources within the 21st Call for Project Access, exploiting the results of the scaling tests from the present Preparatory Access project. Overall, we found very appropriate for PRACE keeping a variety of available architectures for computational scientists. Preparatory Access gives indeed the opportunity to each investigator of identifying and selecting the most suitable computational resources for his work, to be exploited via major production runs through Project Access Calls. We think that testing allowed by allocations awarded through Preparatory Access is convenient for the best use of the available computational resources within Project Access and in turn beneficial to both applicant scientists and PRACE. This will allow us in the future to tackle even larger computational problems in the simulation of turbulent flows typical of naval hydrodynamics, whose resolution requirements are exceptionally demanding.