Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

XIOS Integration for ~~Open~~IFS: Computational Aspects and Performance Evaluation

Xavier Yepes-Arbós

Mario C. Acosta

18/05/2018

Lecce, 5th ENES Workshop

1. Introduction

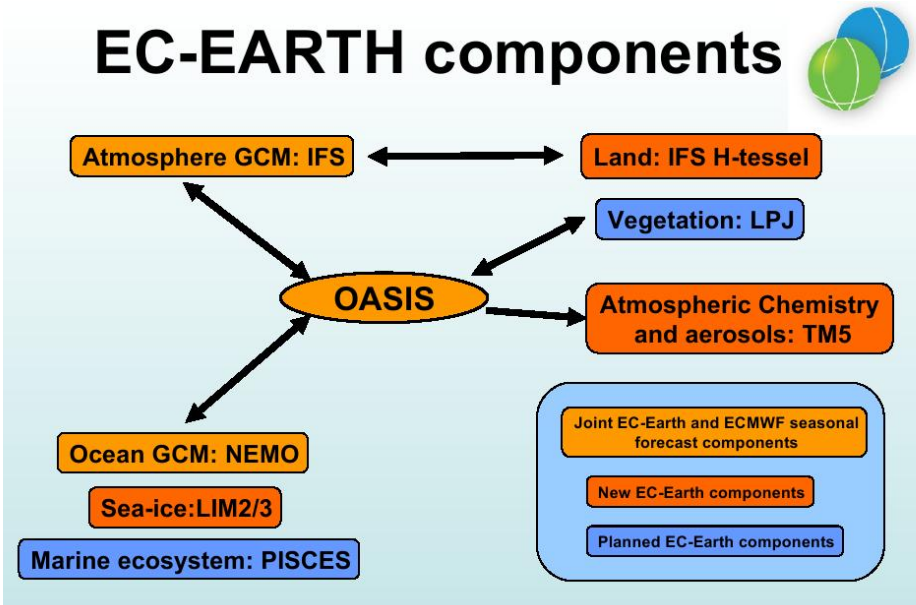
Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

Introduction

- IFS is a global forecasting system developed by ECMWF
- It has two different output schemes:
  - The Météo-France (MF) I/O server (ECMWF only)
  - An inefficient sequential I/O scheme (the rest of users)
- The sequential I/O scheme requires a serial process:
  - Gather all data in the master process of the model
  - Then, the master process sequentially writes all data
- This is not scalable for higher grid resolutions, and even less, for future exascale machines
- IFS is also used in some Earth system models, such as EC-Earth

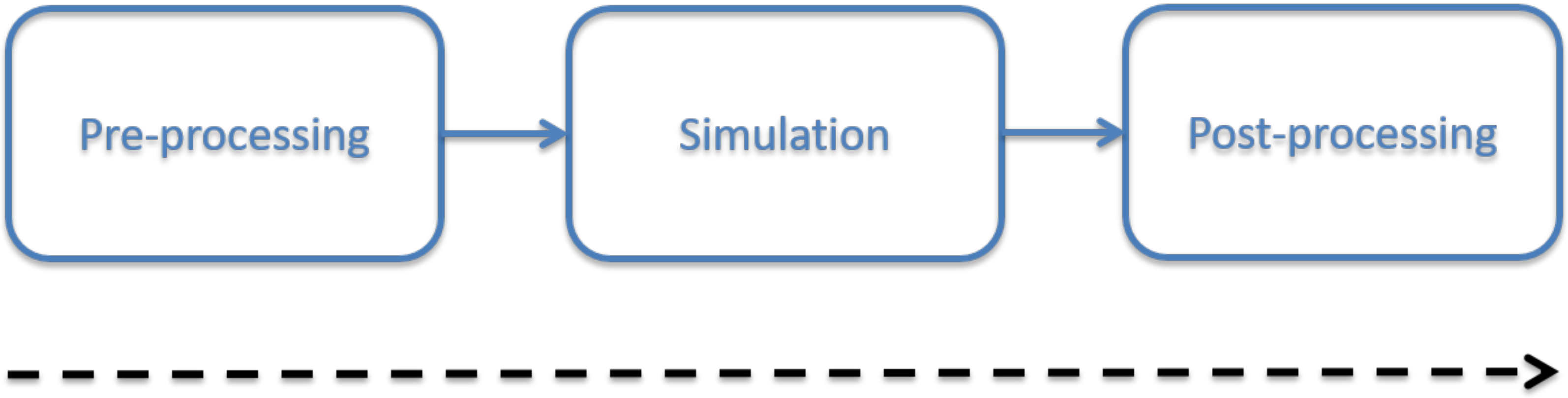
Introduction

- EC-Earth is a global coupled climate model, which integrates a number of component models in order to simulate the Earth system
- The two main components are **IFS as the atmospheric model** and NEMO as the ocean model



Introduction

- In addition, Earth system models such as EC-Earth, run experiments that have other tasks in their workflow
- Post-processing task can perform data format conversion, compression, diagnostics, etc.



Critical path = Pre-processing + Simulation + Post-processing

Introduction

- When IFS is used in EC-Earth for climate modeling, post-processing is needed to:
  - Convert GRIB to netCDF files
  - Transform data to be CMIP-compliant
  - Compute diagnostics
- Post-processing turns into an expensive process

Motivation

- In particular, we are experiencing an I/O bottleneck in the IFS version of EC-Earth
- EC-Earth has been recently used to run ultra-high resolution experiments under the H2020 PRIMAVERA project
- However, it suffers a considerable slowdown, where the I/O in IFS represents about 30% of the total execution time

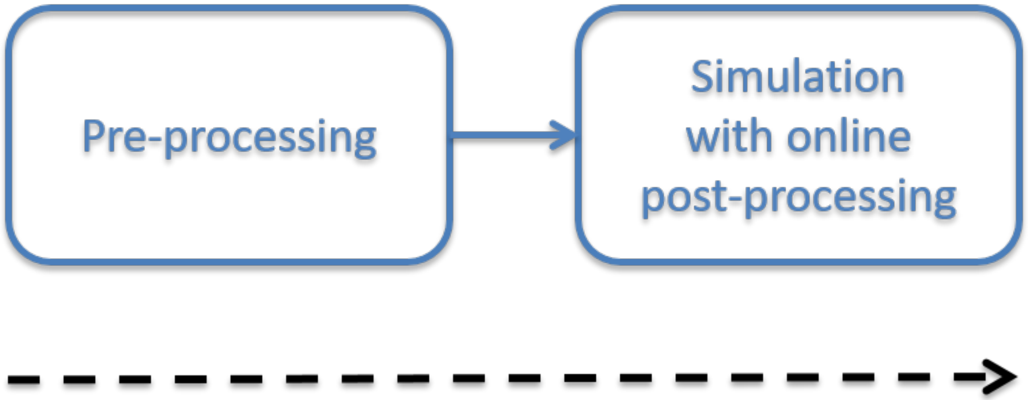
Motivation

- In order to address the I/O issue, we have to select a suitable tool that fulfills a series of needs:
  1. It must be a parallel, efficient and scalable I/O tool
  2. Data must be written using netCDF format (standard in climate modelling) and must follow the CMIP standard
  3. It must perform online post-processing along with the simulation, such as interpolations or data compression
- There is a tool designed to that end: XIOS
- XIOS is an I/O server

Motivation

The use of a tool such as XIOS has a twofold effect:

- Improve the computational performance and efficiency of a model, and thus, reduce the execution time
- Reduce the critical path of its workflow by avoiding the post-processing task



Critical path = Pre-processing + Simulation

European collaboration

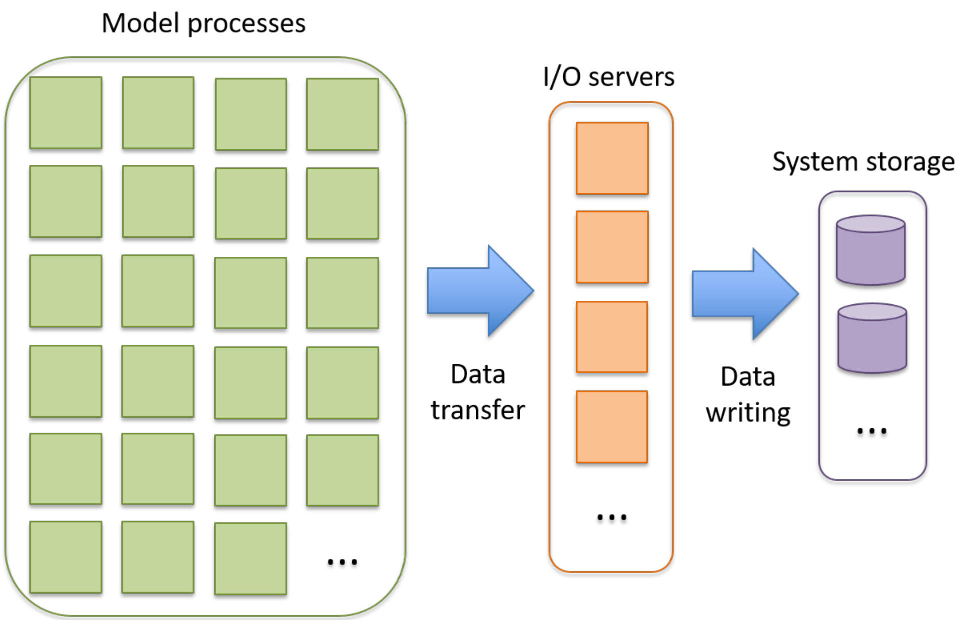
- European Centre for Medium-Range Weather Forecasts (ECMWF)
  - Help in decisions: design, setups, etc.
  - Support with ECMWF enviroment
  - XIOS as an optional I/O scheme of OpenIFS
- Netherlands eScience Center (NLeSC)/Koninklijk Nederlands Meteorologisch Instituut (KNMI)
  - Help in decisions: design, setups, etc.
  - Add FullPos features
  - Ensure a user-friendly output

2. State-of-the-art overview

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

State-of-the-art overview

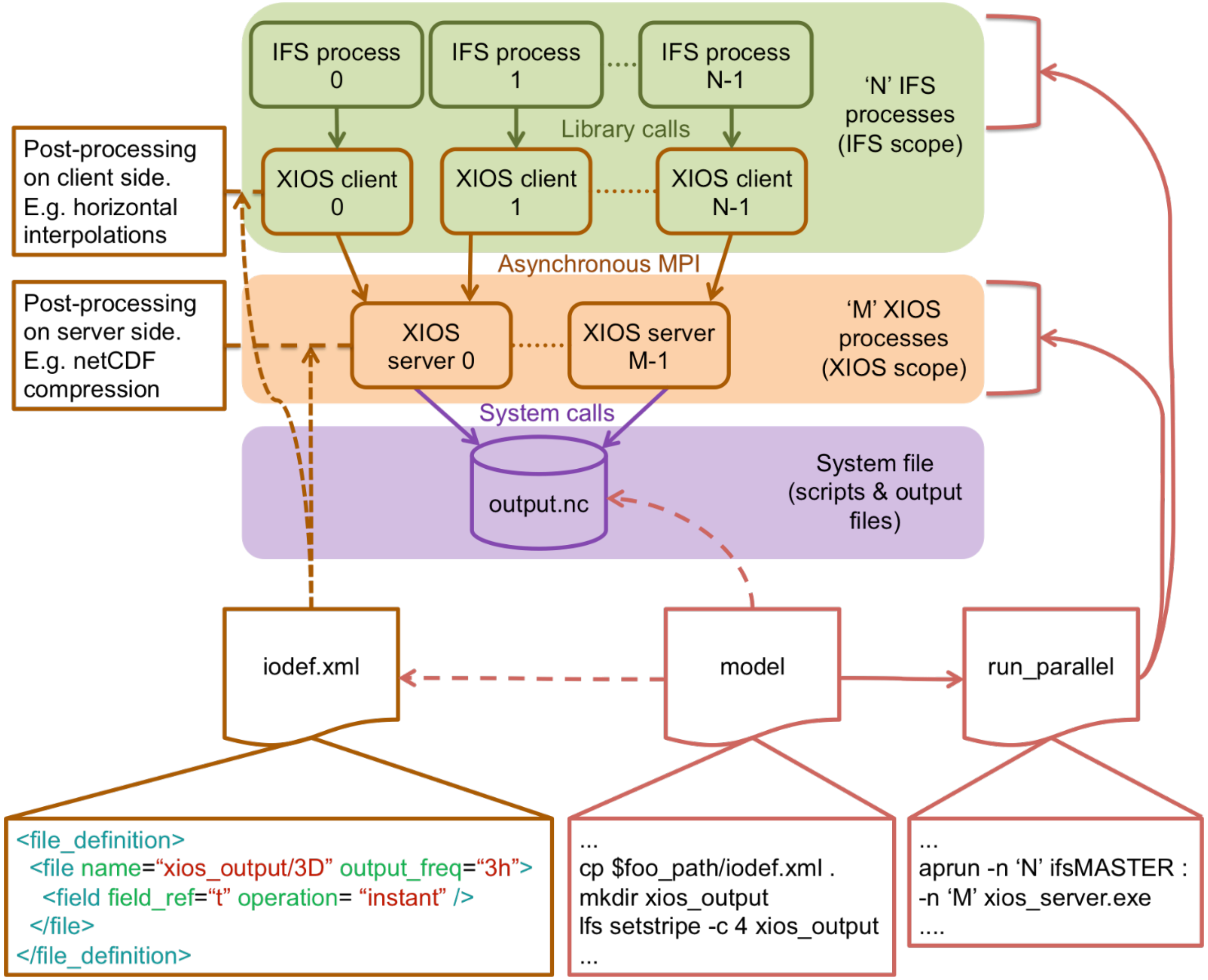
- Sequential I/O
- Parallel I/O libraries: MPI-IO, HDF5 and netCDF
- I/O servers:
  - ADIOS
  - CDI-pio
  - CFIO
  - XIOS



4. IFS-XIOS integration

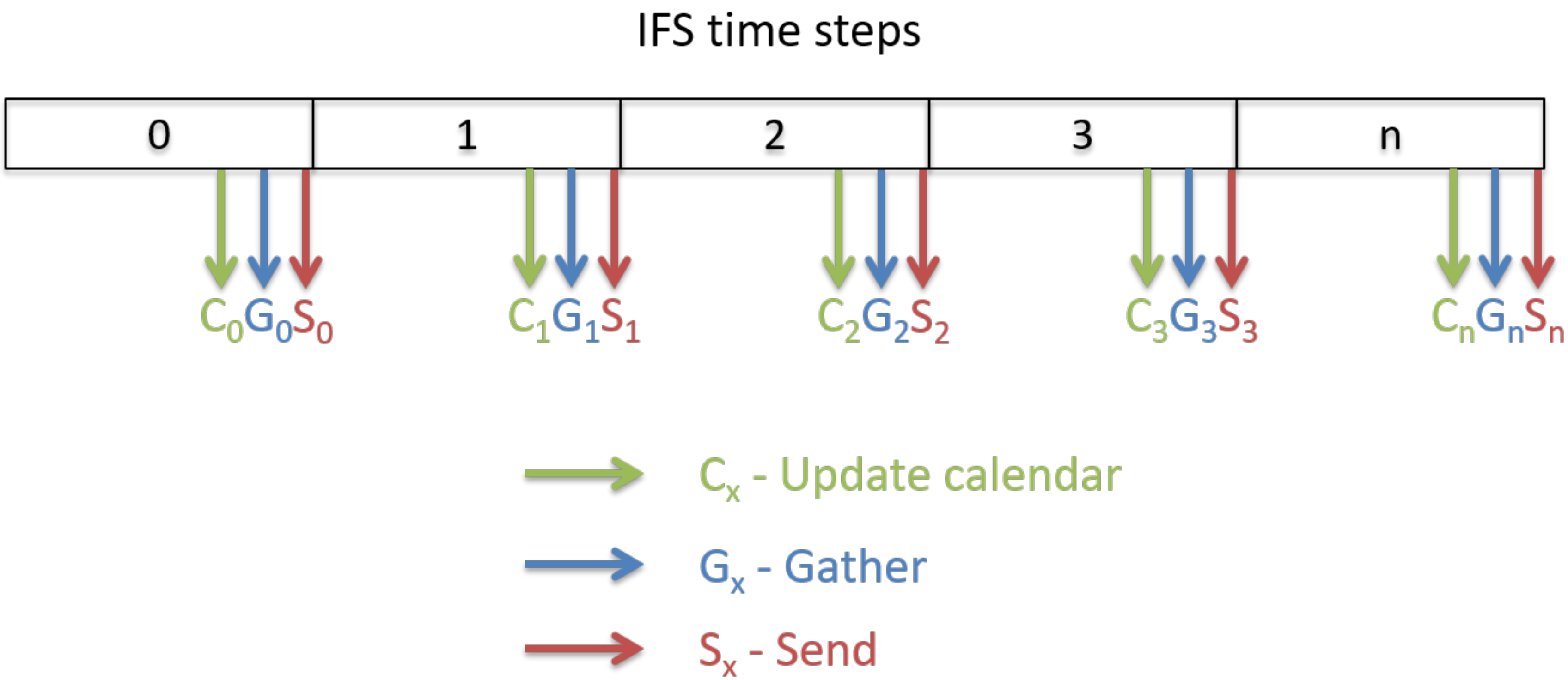
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Scheme of IFS-XIOS integration



Output scheme approach

- If it is an output time step, at the end of it IFS sequentially executes three steps
- Otherwise, IFS only executes the update calendar step



Development steps

- XIOS setup
  - Initialization
  - Finalization
  - Context
    - Calendar
    - Geometry (axis, domain and grid)
  - *Iodef.xml* file
- Grid-point fields transfer
  - NPROMA blocks gather
  - Send fields
- Environment setup
  - XIOS compilation
  - Include and link XIOS, netCDF and HDF5
  - Model script
  - Supporting MPMD mode

Development steps

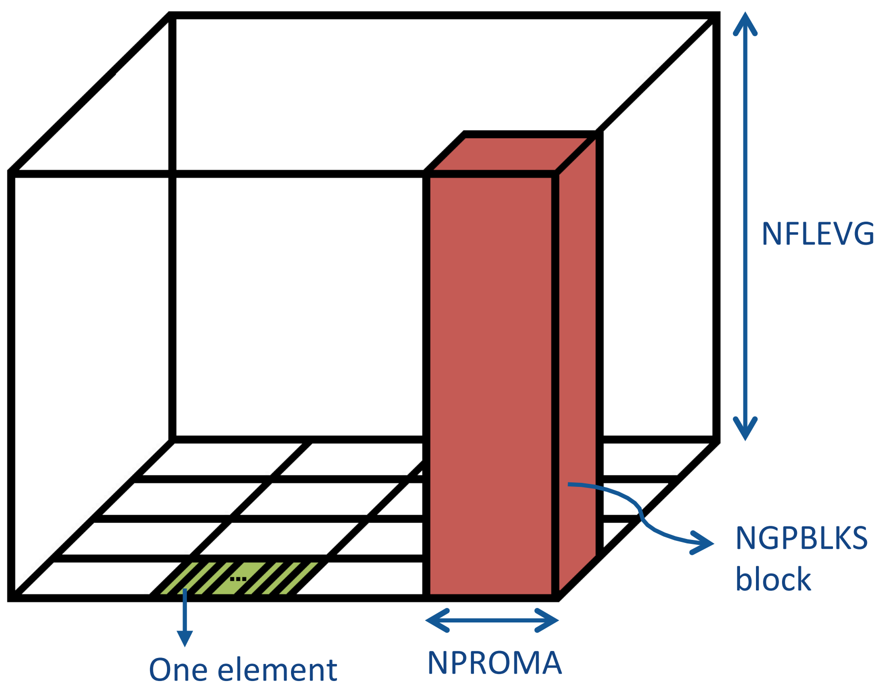
- XIOS setup
  - Initialization
  - Finalization
  - Context
    - Calendar
    - Geometry (axis, domain and grid)
  - *Iodef.xml* file
- Grid-point fields transfer
  - **NPROMA blocks gather**
  - Send fields
- Environment setup
  - XIOS compilation
  - Include and link XIOS, netCDF and HDF5
  - Model script
  - Supporting MPMD mode

NPROMA blocks gather

- The IFS data arrays do not match with the XIOS ones:
  - IFS_data_array(NPROMA, NFLEVG, NFIELDS, NGPBLKS)
  - XIOS_data_array(unidimensional 2D domain, NFLEVG)
- We have to re-shuffle fields data before sending them
- According to the blocking strategy used in IFS, we have to build an XIOS-style array by gathering NPROMA blocks

Subdomain decomposition in IFS

- IFS uses a blocking strategy to efficiently parallelize the manipulation of data arrays using OpenMP
- IFS_data_array(NPROMA, NFLEVG, NFIELDS, NGPBLKS)



5. Performance analysis and optimization

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Execution overview

- 702 MPI processes, each with 6 OpenMP threads
- 10 days of forecast with a time step of 600 seconds
- Output size of netCDF files: 3.2 TB
- Execution times:
  - Sequential output: 9054 seconds
  - MF I/O server: 7535 seconds
  - IFS-XIOS integration: 7773 seconds
  - No output: 7356 seconds

Threading with OpenMP

- Using GSTATS timers, we profiled the NPROMA blocks gather. For instance, the gather of the ciwc field:

ROUTINE	CALLS	SUM(s)	AVE(ms)	STDDEV(ms)	MAX(ms)	SUMB(s)	FRAC(%)
ciwc_GATHER	80	0.5	5.8	0.9	6.5	0.0	0.01

- It does not take too much time. However, it only works the master thread, while the rest are idle
- This is not efficient and could become a bottleneck

Threading with OpenMP

- We used OpenMP to parallelize the NPROMA blocks gather
- In addition, we overlap the send of one field with the gather of the next one

Threading with OpenMP

```
!$OMP PARALLEL PRIVATE(jstglo,icend,ibl,jlev)

! GFL – Specific humidity
IF (q) THEN
  !$OMP DO SCHEDULE(DYNAMIC)
  DO jstglo = 1, YDGEOMETRY%YRGEM%NGPTOT, YDGEOMETRY%YRDIM%
      ↪ NPROMA
    icend = MIN(YDGEOMETRY%YRDIM%NPROMA,YDGEOMETRY%YRGEM%
        ↪ NGPTOT−jstglo+1)
    ibl = (jstglo−1)/YDGEOMETRY%YRDIM%NPROMA + 1
    DO jlev = 1, YDGEOMETRY%YRDIMV%NFLEVG
      xios_gfl(jstglo:jstglo+icend−1,jlev) = YDFIELDS%YRGFL%GFL(1:icend,jlev,
          ↪ YGFL%YQ%MP,ibl)
    END DO
  END DO
  !$OMP END DO
  !$OMP SINGLE
  CALL xios_send_field("q",xios_gfl)
  !$OMP END SINGLE NOWAIT
END IF

! GMV – Temperature
IF (t) THEN
  !$OMP DO SCHEDULE(DYNAMIC)
  DO jstglo = 1, YDGEOMETRY%YRGEM%NGPTOT, YDGEOMETRY%YRDIM%
      ↪ NPROMA
```

Gather

Send

Gather and send overlap among two fields

Threading with OpenMP

- The profiling shows an improvement:

ROUTINE	CALLS	SUM(s)	AVE(ms)	STDDEV(ms)	MAX(ms)	SUMB(s)	FRAC(%)
ciwc_GATHER	80	0.1	1.2	0.2	2.3	0.0	0.00

- The execution time is reduced 68 seconds, from 7773 seconds to 7705 seconds

Optimized compilation of XIOS

- We had a lot of issues to optimally compile XIOS
- For this reason, we used a conservative option: *-O1*
- XIOS reports too much time for just outputting data:

Client

```
-> report :  Performance report : Whole time from XIOS init and
   ↪ finalize: 7681.68 s
-> report :  Performance report : total time spent for XIOS :
   ↪ 132.715 s
-> report :  Performance report : time spent for waiting free
   ↪ buffer : 3.80519 s
-> report :  Performance report : Ratio : 0.0495359 %
```

Server

```
-> report : Performance report : Time spent for XIOS : 7681.68
-> report : Performance report : Time spent in processing events :
   ↪ 3196.4
-> report : Performance report : Ratio : 41.6107%
```

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Optimized compilation of XIOS

- A bug was previously reported in the compilation of XIOS using *-O2* and *-O3* for Cray compilers
- However, it was reported using older Cray compilers, so it might be solved in newer versions
- Certainly, XIOS compiled and tests successfully passed

Optimized compilation of XIOS

- The execution time in both client and server sides is reduced

Client

```
-> report :  Performance report : Whole time from XIOS init and
   ↪ finalize: 7562.36 s
-> report :  Performance report : total time spent for XIOS :
   ↪ 40.3018 s
-> report :  Performance report : time spent for waiting free
   ↪ buffer : 0.463693 s
-> report :  Performance report : Ratio : 0.00613159 %
```

Server

```
-> report : Performance report : Time spent for XIOS : 7562.37
-> report : Performance report : Time spent in processing events :
   ↪  1382.16
-> report : Performance report : Ratio : 18.2768%
```

- The execution time is reduced 76 seconds, from 7705 seconds to 7629 seconds

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Overlapping computation and communication

In an output time step, there is a slight increase in the execution time of the three following time steps

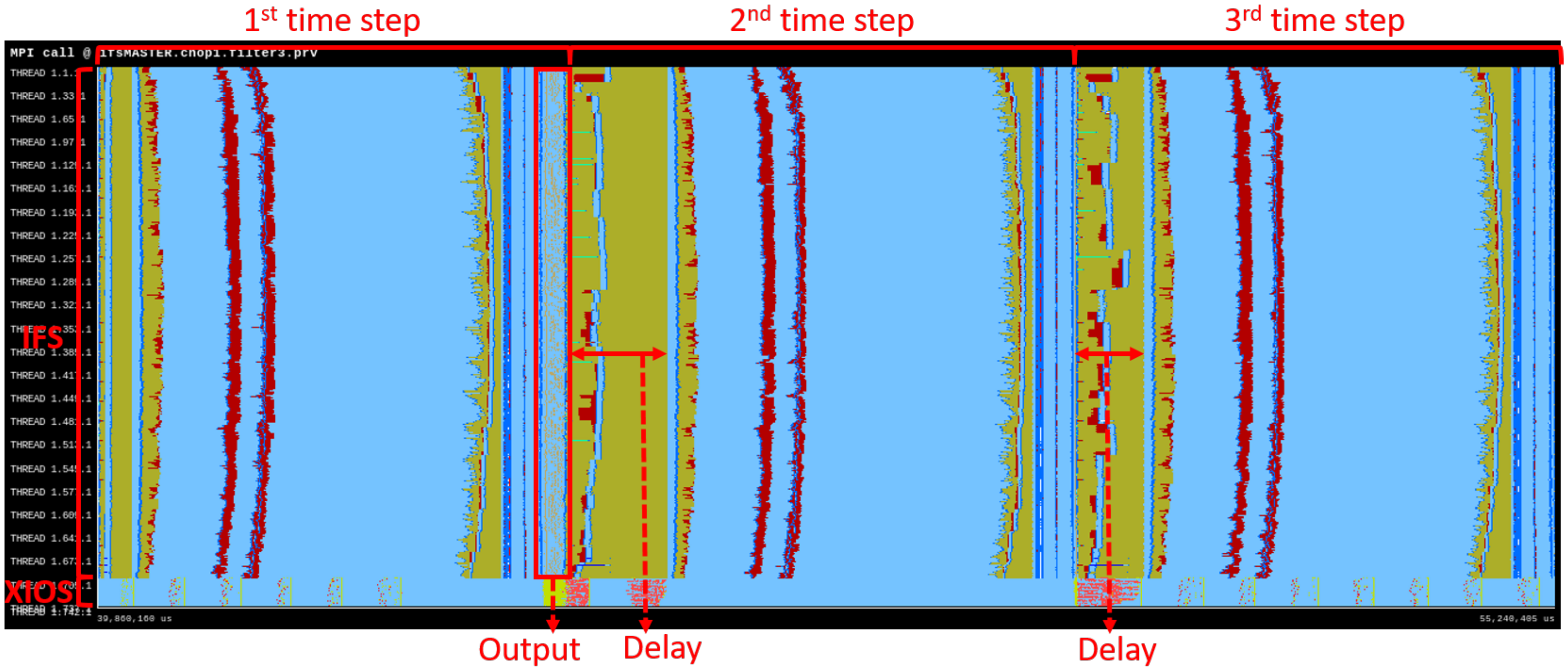
Non-output

Output

12:24:55	0AAA00AAA	STEPO	318	27.370	27.370	4.592	167:53
12:25:02	0AAA00AAA	STEPO	319	39.994	39.994	6.708	168:33
12:25:07	0AAA00AAA	STEPO	320	28.826	28.826	4.826	169:02
12:25:12	0AAA00AAA	STEPO	321	28.034	28.034	4.701	169:30
12:25:16	0AAA00AAA	STEPO	322	27.770	27.770	4.655	169:58
12:25:21	0AAA00AAA	STEPO	323	27.690	27.690	4.654	170:26
12:25:26	0AAA00AAA	STEPO	324	27.854	27.854	4.679	170:53
12:25:33	0AAA00AAA	STEPO	325	42.771	42.771	7.158	171:36
12:25:38	0AAA00AAA	STEPO	326	30.114	30.114	5.044	172:06
12:25:43	0AAA00AAA	STEPO	327	30.870	30.870	5.181	172:37
12:25:48	0AAA00AAA	STEPO	328	27.874	27.874	4.682	173:05

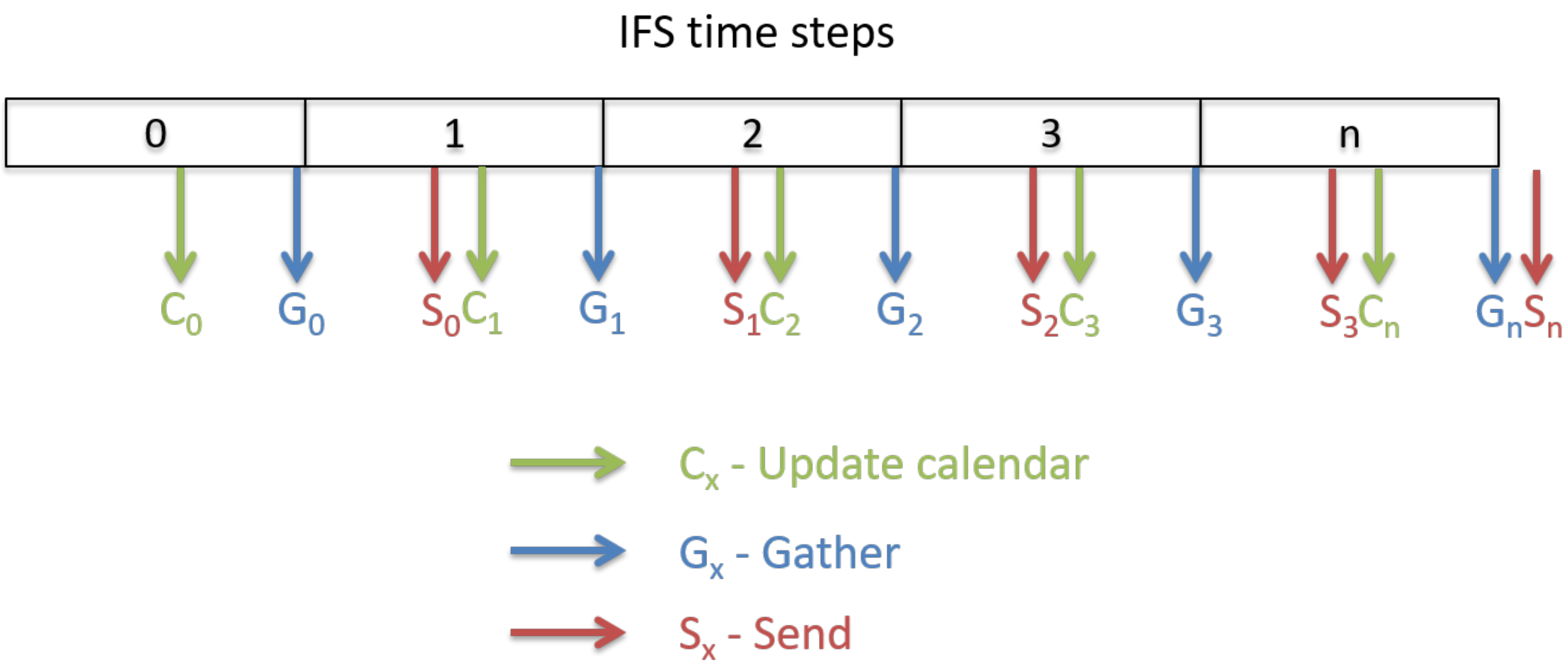
Overlapping computation and communication

- The trace shows that after an output time step, there is a delay in the communication of the next two time steps (*MPI_Waitany* and *MPI_Alltoallv*)
- There is a conflict between intra IFS communication and IFS to XIOS communication



Overlapping computation and communication

- We used a new output scheme to truly overlap XIOS communication with IFS computation
- It splits the three needed steps to output data through XIOS:



Overlapping computation and communication

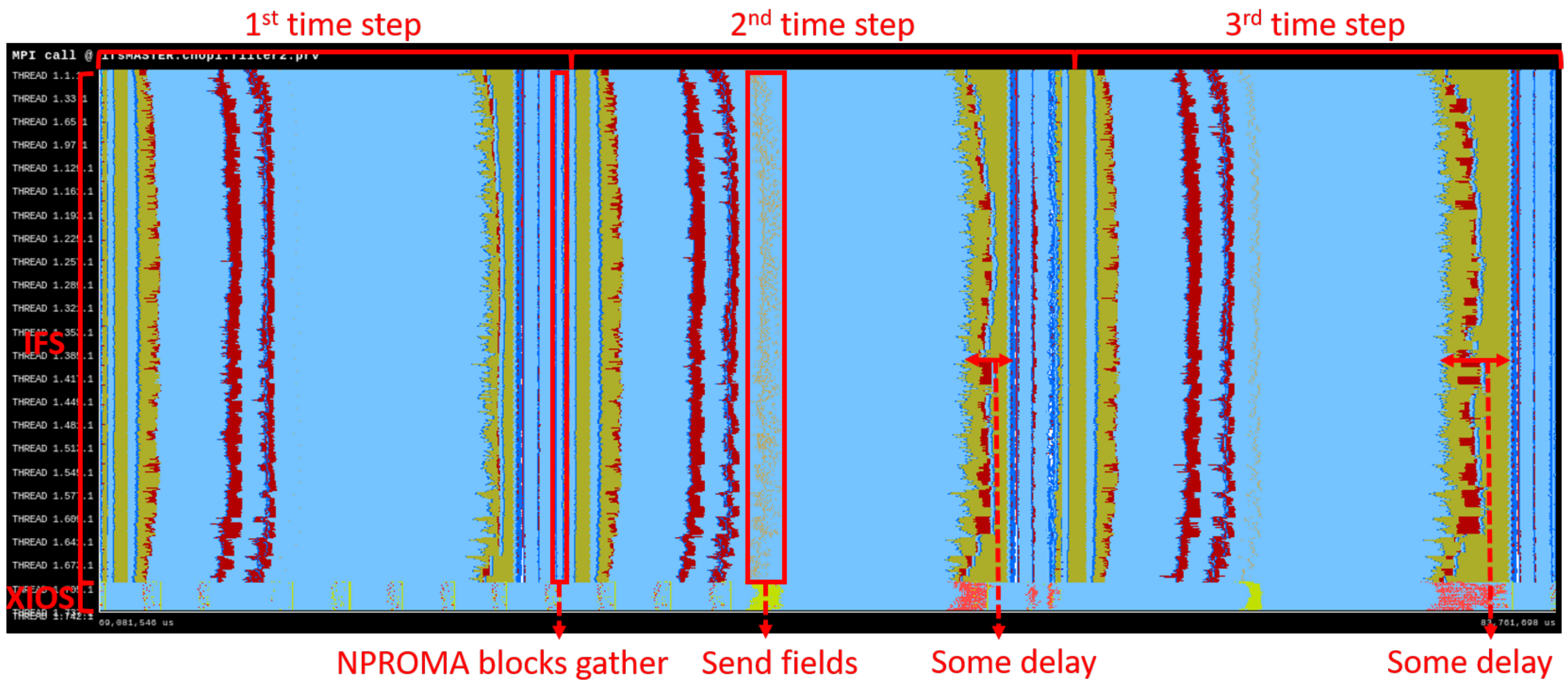
- This new scheme improves the execution time of the three time steps that follow an output time step:

Non-output	12:27:45	0AAA00AAA	STEP0	318	26.926	26.926	4.514	162:23
	12:27:52	0AAA00AAA	STEP0	319	38.414	38.414	6.441	163:01
	12:27:56	0AAA00AAA	STEP0	320	27.054	27.054	4.535	163:28
	12:28:01	0AAA00AAA	STEP0	321	27.030	27.030	4.534	163:55
	12:28:05	0AAA00AAA	STEP0	322	26.882	26.882	4.502	164:22
	12:28:10	0AAA00AAA	STEP0	323	27.394	27.394	4.607	164:50
Output	12:28:15	0AAA00AAA	STEP0	324	27.142	27.142	4.549	165:17
	12:28:21	0AAA00AAA	STEP0	325	39.310	39.310	6.579	165:56
	12:28:26	0AAA00AAA	STEP0	326	28.318	28.318	4.755	166:24
	12:28:31	0AAA00AAA	STEP0	327	28.686	28.686	4.813	166:53
	12:28:35	0AAA00AAA	STEP0	328	26.990	26.990	4.527	167:20

- The execution time is reduced 122 seconds, from 7629 seconds to 7507 seconds

Overlapping computation and communication

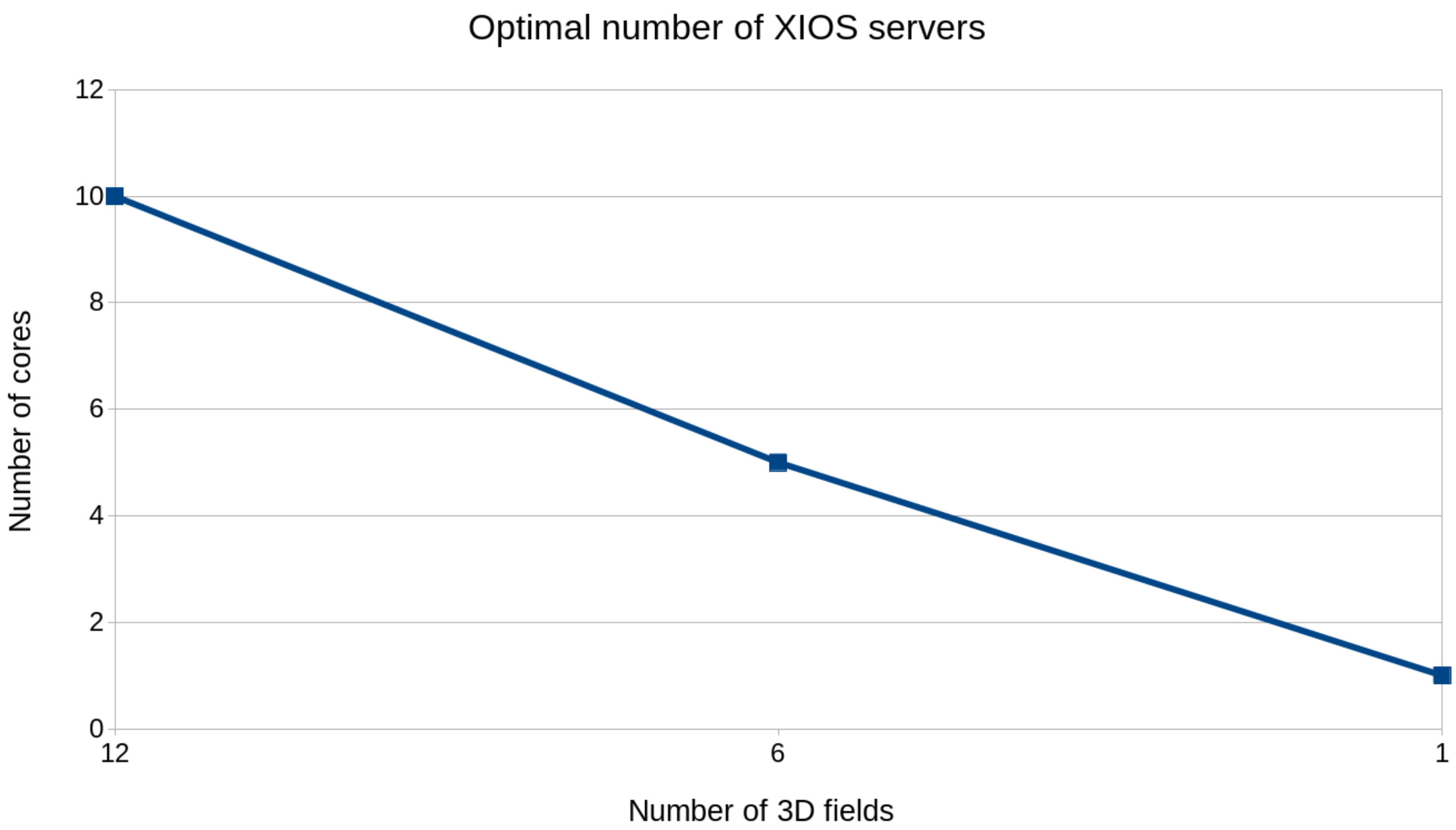
- The trace shows that there is no delay at the beginning of the 2nd and 3rd time steps
- However, there is some delay at the end, but it is less significant



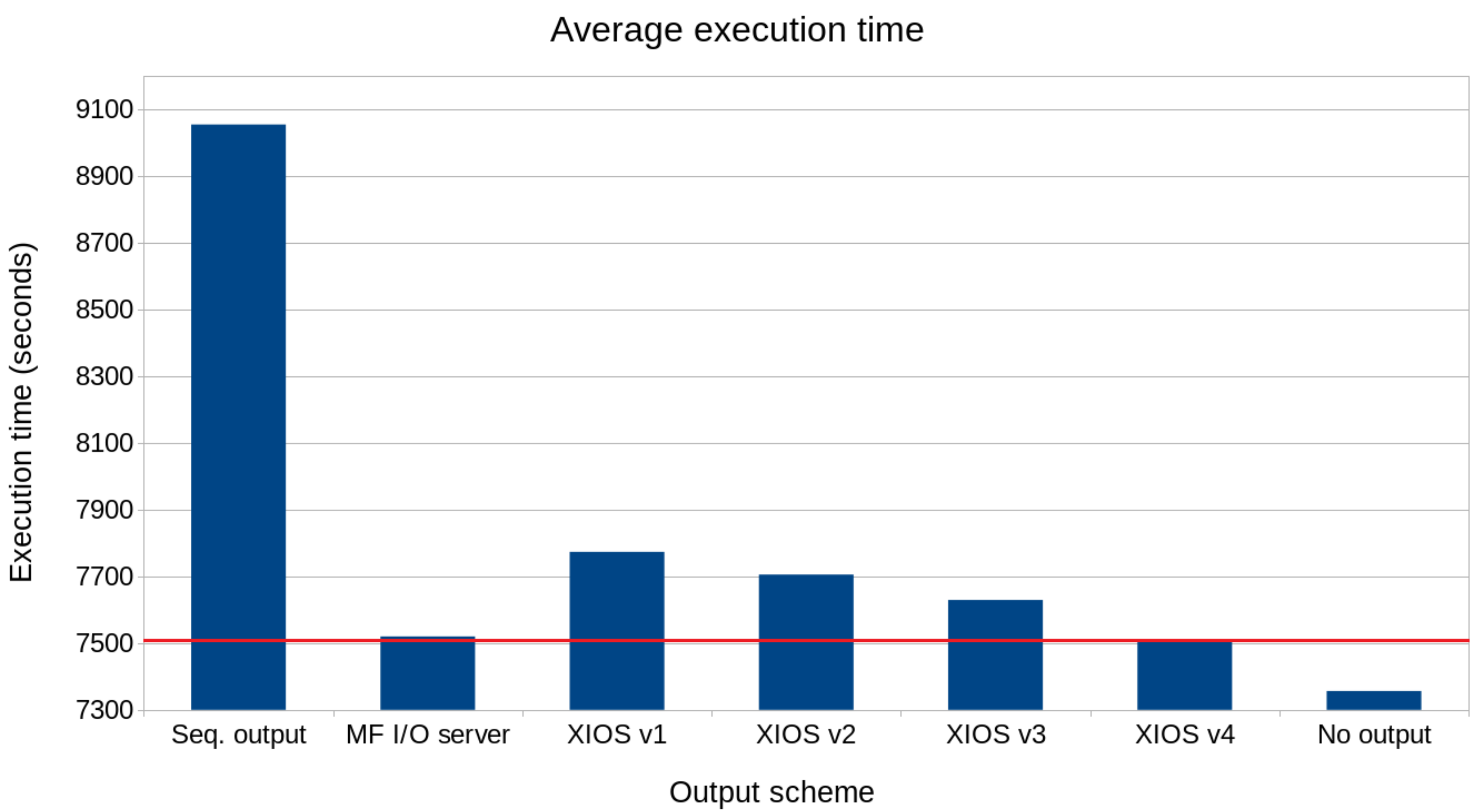
6. Evaluation

Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

Optimal number of XIOS servers

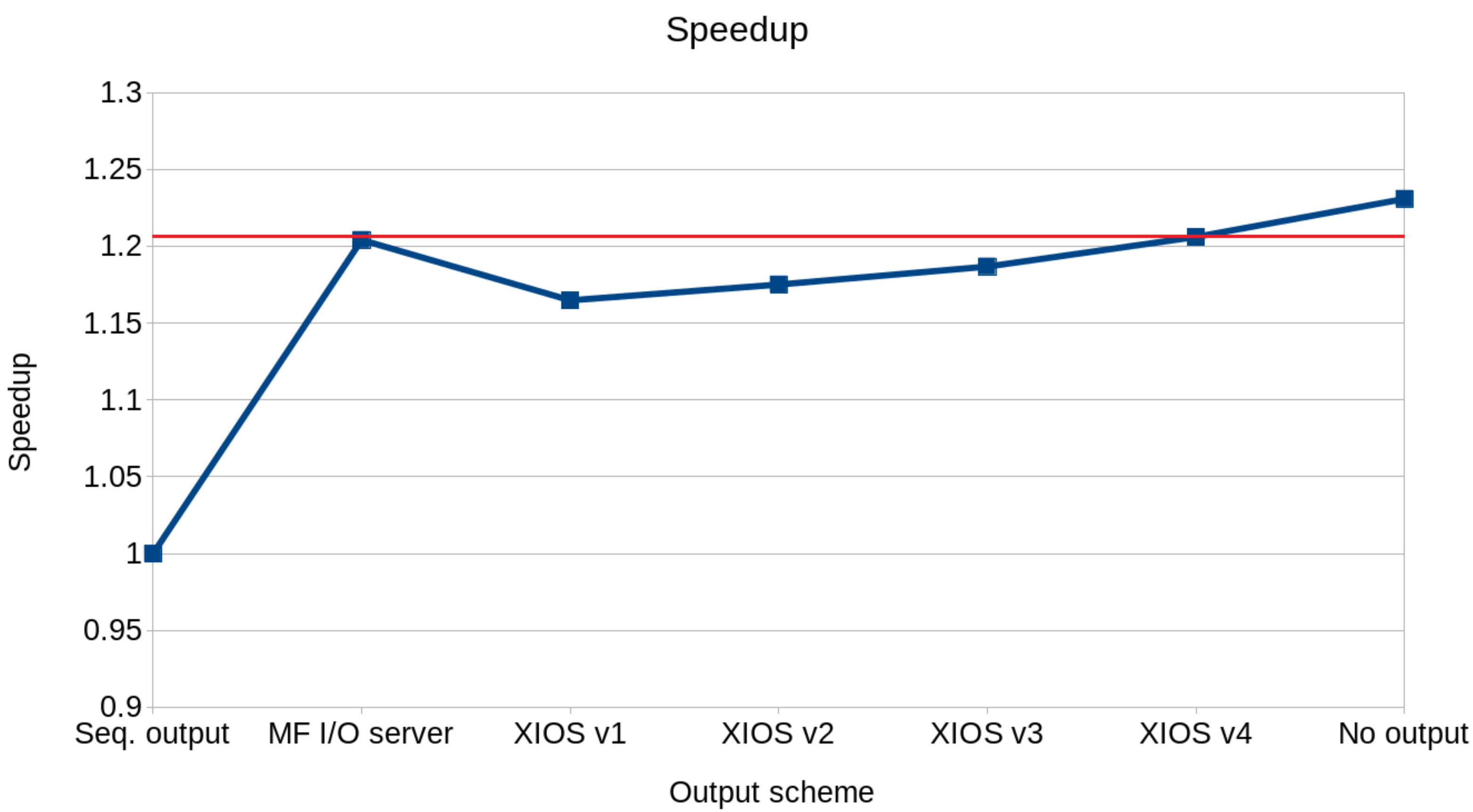


Comparison test



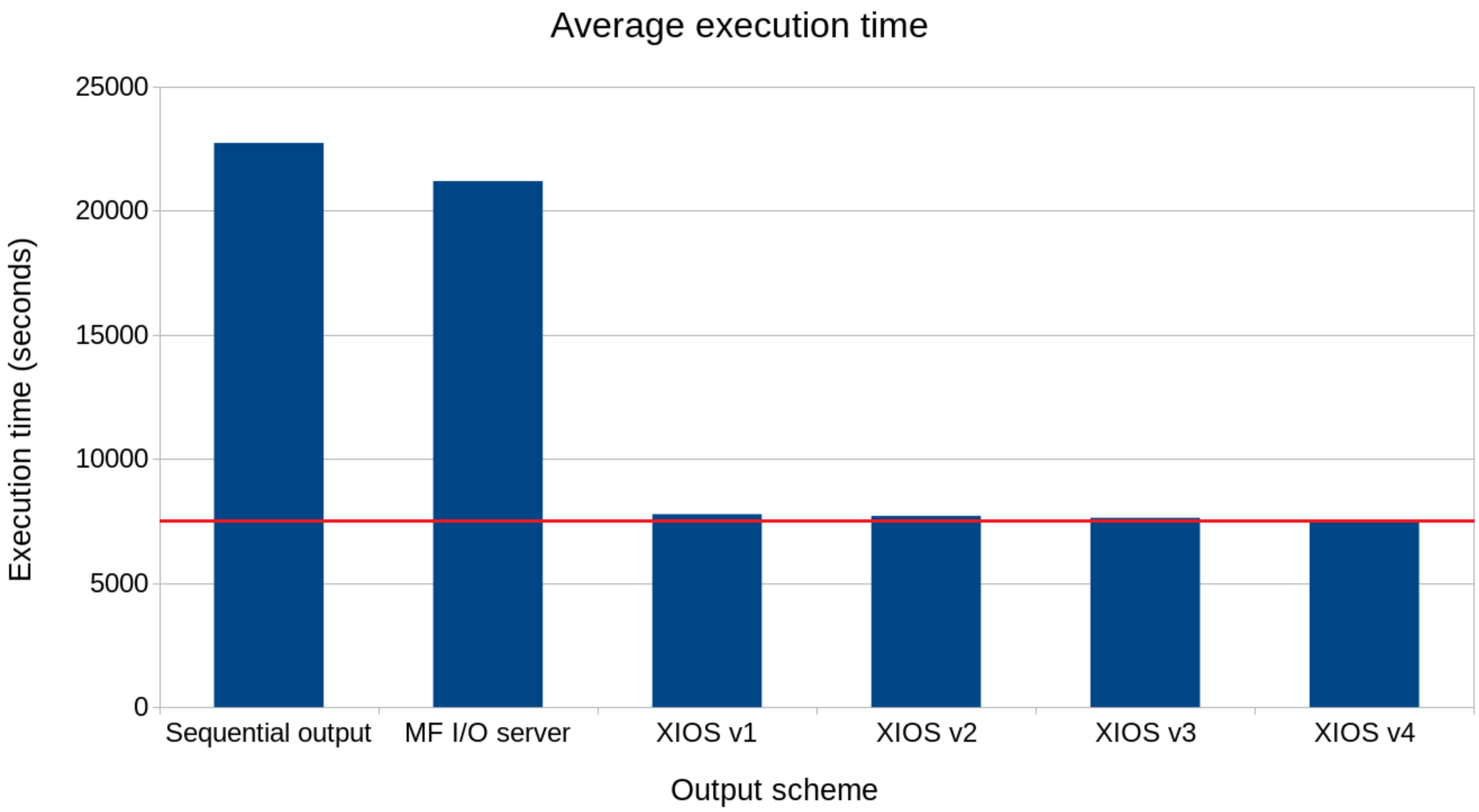
*MF I/O server includes FullPos calls

Comparison test



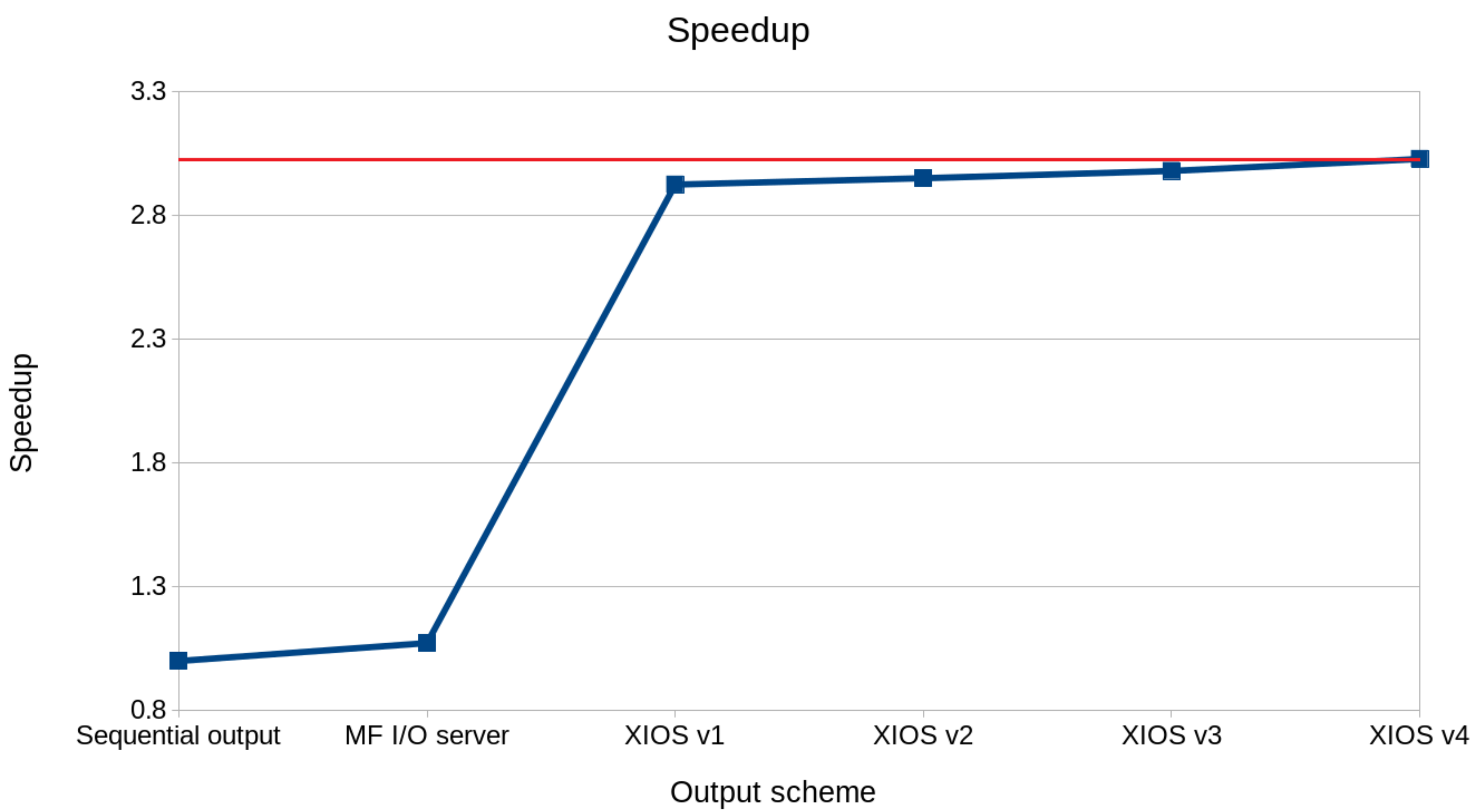
*MF I/O server includes FullPos calls

Comparison test adding GRIB to netCDF post-processing



*MF I/O server includes FullPos calls

Comparison test adding GRIB to netCDF post-processing



*MF I/O server includes FullPos calls

Comparison test with additional computational resources and GRIB to netCDF post-processing

- In this test, we add to the IFS processes of the sequential I/O scheme the equivalent of the computational resources needed to run XIOS
- XIOS uses 10 cores spread along 10 nodes
- Then, the execution times with the additional resources are:
  - XIOS v4 (702 IFS + 10 XIOS) → 7507 seconds
  - Seq. I/O (702 IFS) → 22734 seconds
  - Seq. I/O (702 + 10 cores = 712 IFS) → 22583 seconds
  - Seq. I/O (702 + 120 cores* = 822 IFS) → 21962 seconds

*According to the IFS affinity used, 10 nodes = 120 cores

7. Conclusions

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Conclusions

- The integration with no optimization already improved the execution time:
  - Sequential output 9054 seconds (23% of overhead) → IFS-XIOS integration 7773 seconds (5.6% of overhead)
- Performance highlights of the most optimized version:
  - It is slightly faster than the MF I/O server (but without FullPos calls): 7519 s vs. 7507 s
  - It is only 151 seconds slower than no output (2% of overhead)
  - Within 151 seconds IFS outputs 3.2 TB of data
- When post-processing to convert GRIB to netCDF files is taken into account:
  - The post-processing takes 13680 seconds (3.8 hours)
  - Thus, the most optimized version is a 202% faster than the sequential output and a 182% faster than the MF I/O server

Conclusions

- These numbers denote that we have implemented an scalable and efficient development that will address the I/O issue
- In EC-Earth, this new I/O development will:
  - Increase the performance and efficiency of the whole model
  - Perform online post-processing operations
  - Save thousands of computing hours
  - Save storage space, because it will only store processed data ready to be used

Ongoing work

- Add FullPos features into the IFS-XIOS integration to:
  - Convert spectral fields to grid-point fields before sending them to XIOS
  - Use vertical interpolations such as pressure or PV level

  Ensure a user-friendly output configuration file
- Increase the usability and the robustness of the IFS-XIOS integration by managing XIOS elements in runtime: fields, files, domain, axis, etc.

Future work

- Performance analysis and optimization of the final integration reusing the explained optimization techniques and maybe new ones
- The development done for IFS will be ported to OpenIFS
- Adapt the EC-Earth model to generate online diagnostics from OpenIFS and NEMO components through XIOS

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

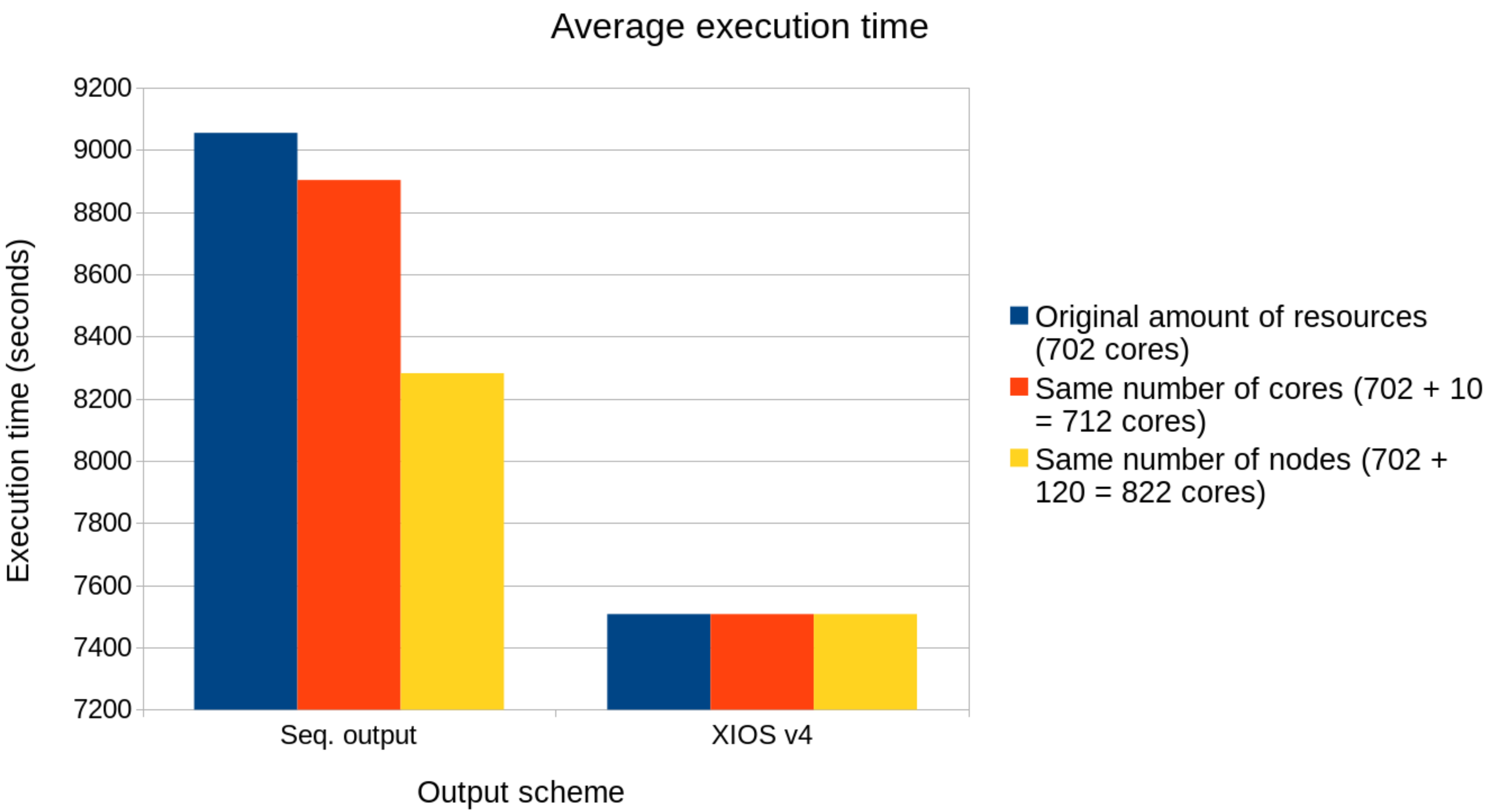
Thank you!

mario.acosta@bsc.es

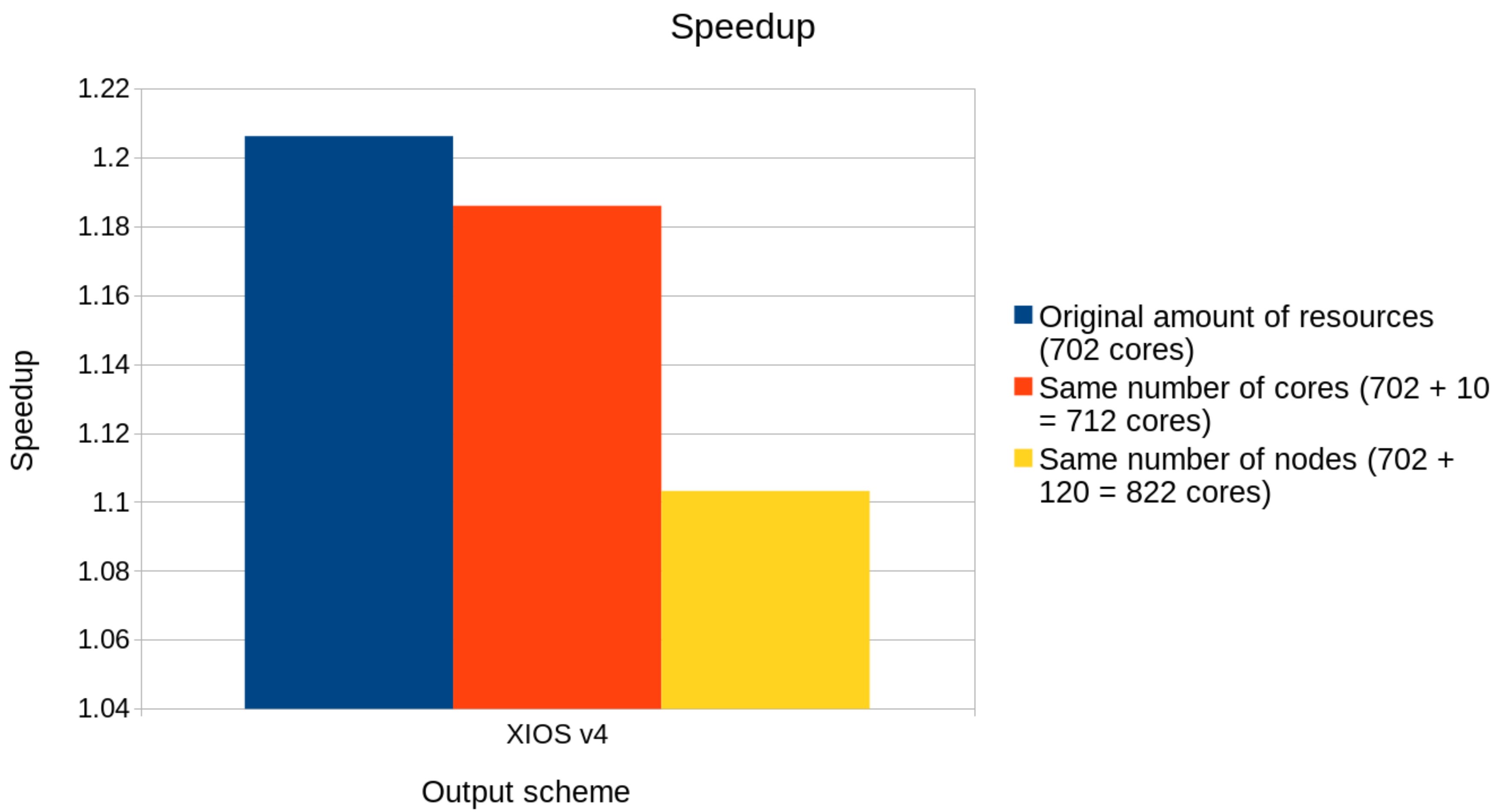
Appendix

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Comparison test with additional computational resources



Comparison test with additional computational resources



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Comparison test with additional computational resources and GRIB to netCDF post-processing

Comparison test with additional computational resources and GRIB to netCDF post-processing