EUROPEAN COMMISSION
Community research

is-enes
INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

IS-ENES2 DELIVERABLE (D -N°: 9.5)

CDO performance analysis

File name: {IS-ENES2_D9.5.pdf}

Authors: **Asela Rajapakse**
Uwe Fladrich

Reviewer: **Eric Maisonnave**

Reporting period: ***01/04/2016 – 31/03/2017***

Release date for review: **10/04/2017**

Final date of issue: **18/04/2017**

Revision table			
Version	**Date**	**Name**	Comments
0	20/02/2017	Asela Rajapakse	First draft
1	23/03/2017	Uwe Fladrich	Reformatting
2	04/04/2017	Asela Rajapakse	Add measurement results
3	05/04/2017	Uwe Fladrich	Add to conclusions
4	07/04/2017	Asela Rajapakse	Final version for review
	11/04/2017	Eric Maisonnave	Internal review
5	18/04/2017	Rajapakse/Fladrich	Final version

Abstract

The computational performance of the CDO (Climate Data Operators) is investigated in the context of the M4HR (Multi-model multi-member high-resolution) demonstrator developed in JRA1. A detailed performance analysis focussed on the disk transfer rates (read and write operations) of the CDO and shows no indication of performance bottlenecks in the post-processing workflow.

Project co-funded by the European Commission's Seventh Framework Programme (FP7; 2007-2013) under the grant agreement n°312979		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants including the Commission Services	
RE	Restricted to a group specified by the partners of the **IS-ENES2** project	
CO	Confidential, only for partners of the **IS-ENES2** project	

Table of contents

Executive Summary

1. Objectives

The M4HR demonstrator developed in JRA1, and described in detail in D9.6, requires the availability of efficient post-processing tools to allow for the analysis of large volumes of data. Both ESMs used in the demonstrator utilise the CDO for their post-processing work flow. Hence, analysis of the computational performance of the CDO adds to the understanding of the performance characteristics of the end-to-end workflow and helps to identify bottlenecks in the data processing. The M4HR components produce tens of gigabyte per simulated year, which have to be read from and written to disk by the CDO during the post-processing step. A thorough analysis of the disk transfer rates of the CDO is therefore a fundamental part of the analysis.

2. Results

To reliably measure the disk transfer rates of the CDO with the *cdiwrite* and *cdiread* operators, a controlled measurement environment has been defined and implemented. This includes the underlying hard- and software as well as the CDO versions and the measuring methodology. Given the fact that the hardware platform used in the M4HR experiments was not available for these tests, an explicit environment description is added to allow for detailed comparison. The disk transfer rates for *cdiwrite* and *cdiread* are measured, ensuring statistical validity to a certain degree. The results are set into the context of the M4HR demonstrator performance metrics, concluding that the CDO post-processing does not constitute a performance bottleneck, as far as disk transfer is concerned.

3. Perspectives

Shortcomings of the measurement methodology are discussed, addressing the limited scope of the analysis. A main issue is the integration of CDO performance analysis with the actual M4HR runs, which was made impossible by the unavailability of the computational resource used in this work package.

An obvious continuation of the performance measurements is the complete analysis of the entire post-processing workflow. However, careful definition of the post-processing operations and environment is needed and validity of such measurements across ESMs has to be investigated.

1. Deliverable context

The subsequent deliverable definition needs to be understood in the context of IS-ENES2 WP 9 *Multi-model, multi-member high resolution (M4 HR) Earth System Models*, which has the stated aim of identifying leading issues that impede M4 HR simulation efficiency. The question guiding WP 9 task 4 titled *Post-processing analysis (PP/DA) efficiency for HR M4 experiments* and its main product, deliverable D9.5 *CDO performance analysis*, then becomes whether the CDO software has seen improvements that contribute to increased efficiency of the overall M4 HR simulation runs or that impede it. Within this context, the CDO software performs well if it does not produce a bottleneck for the model efficiency.

The IS-ENES2 Description of Work defines deliverable D9.5 CDO performance analysis as follows:

This deliverable will document the performance of the CDO system for ensembles of M4 HR data sets. The CDO system, presently used by a number of IS-ENES partners for analysis of ESM data, will be analysed - and typical work flows they are used for - for their optimization potential in terms of performance, on both highly parallel and more regular platforms.

An excerpt from the overarching task description gives additional information on a relevant aspect of CDO software performance:

An assumed key for improvements is the reduction of disk transfer (e.g. disk read/write), which additionally can be advanced by efficient compression algorithms. Therefore, compression algorithms within CDO will be assessed and further developed.

The term performance in the deliverable definition is interpreted as I/O performance, specifically regarding those I/O performance improvements that were implemented in the releases reported in IS-ENES2 milestone MS96 New releases of CDO with improved parallel performance.

2. Aspects of I/O performance to be examined in this document

There are two dimensions to I/O performance analysis that need to be considered in WP9 task 4, but not necessarily in this document; the reduction of overall disk transfer, and the increase of the data rate of write and read operations. This is easy to see; it is preferable to write/read 1 MB of data instead on 100 MB, of which 99 MB are redundant, if the data rate of the write/read operations does not impede this efficiency gain by varying with different file sizes. If it slows down with smaller files to the point when writing/reading 1 MB or 100 MB takes the same amount of time, the process cannot be called more efficient.

The reduction of overall disk transfer during post-processing is mentioned as a key factor in optimizing I/O performance in the task description and thus will have to be examined. The aim is an efficient CDO system that avoids all unnecessary read and write operations. Computationally, for each CDO invocation there exists a lower bound of data that need to be read and written, defining a potential optimization limit. Overall disk transfer in this context can only be assessed by examining the entire post-processing chain of a M4 HR simulation run. Optimizing the post-processing script to minimize overall disk transfer appears to be a daunting task, possibly bordering on infeasible. However, measuring overall disk transfer of a M4 HR simulation run is not in the purview of this document. Neither is assessing the potential for I/O performance optimization as requested in the deliverable definition, both in the CDO software and in the workflows it is typically employed in. Within the context of D9.5, this document is dedicated to the second factor relevant to increasing I/O performance, the increase of the data transfer rate of read and write operations. The data transfer rate is

theoretically limited by the hardware resources. However, the software layers between the CDO application and the hardware layer affect this limit, as well as the CDO application itself.

3. I/O performance measurement

Input and output performance measurements of the CDO software can be taken by employing two operators that were developed at the Max Planck Institute for Meteorology (MPI-M) specifically for such tasks: *cdiread* and *cdiwrite*. They are still undocumented as they are primarily used by CDO developers who are familiar with their functionality. Since these operators are employed by experts on CDO I/O performance measurement, who also lend us their support in preparing the set-up, they are the logical choice for performing the measurements required for our deliverable. The operators have been available since CDO version 1.5.9.

3.1 CDO version comparison

The June 2015 IS-ENES2 milestone MS96 *New releases of CDO with improved parallel performance* lists the CDO versions released from December 2012 to April 2015. This list includes all releases from version 1.5.9 to 1.6.9. As of February 2017, further CDO releases have reached version 1.8.0. In order to assess the I/O performance improvements implemented not only during the period covered by MS96, but also during the time up to January 2017, a comparison of the I/O performance between version 1.8.0 and version 1.5.8 is required. Version 1.5.8 is the latest release before potential I/O performance improvements were implemented in the releases listed in MS96. However, due to the fact that the CDO operators to be used for I/O performance measurement were introduced in CDO version 1.5.9, version 1.5.8 cannot be used for a reliable comparison because the *cdiwrite* and *cdiread* operators are not present. Version 1.5.9 has been used instead, removing the performance gains that may have been implemented in it from our observation. This leaves us with a comparison of CDO versions 1.5.9 and 1.8.0.

3.2 The *cdiwrite* operator

The *cdiwrite* operator is dedicated to measuring the output performance of the CDO software as the mean data rate of a writing operation. An operator call starts a series of consecutive measurement runs, the length of which can be determined by an operator parameter. Each run first generates 3.6 GB[*] of 64-bit values in memory before writing them as 32-bit values into a file. The output file format (NetCDF, e.g.) is determined by the standard CDO option -f. The data rate of this data reduction is measured in million values per second (MVal/s). It leads to a write operation of 1.8 GB of data. The operator measures the overall duration of this write operation in seconds and its mean data rate in Megabytes per second (MB/s). After the last run, the mean value for each metric is calculated and printed. The in-memory data size of 3.6 GB before conversion is preset and has provided good results in the past. The value can be changed if necessary. The reduction from 64-bit to 32-bit values can be switched off by setting the -float option, simplifying the entire process. The operator then generates 32-bit values in memory and writes them into a file, omitting 64-bit values entirely. This corresponds to the way data reduction is handled at MPI-M; it is the responsibility of the model itself to reduce its data to 32-bit values and no longer a post-processing step. This improves the I/O performance of the CDO software by a constant factor. However, the -float

1) This is considerably less data than the output of the M4HR demonstrator (which writes tens of GB per simulated year), but still enough to give sustained rates. See section 6 for a discussion of the results in the context of the M4HR demonstrator.

option is new and only available from CDO version 1.7.2 onwards. So any comparison to an older CDO version will have to include the data reduction step, incurring a loss of I/O performance by a constant factor for both versions.

3.3 The *cdiread* operator

The *cdiread* operator was devised to perform input performance measurements of the CDO software as the mean data rate of its reading operations. It works analogous to the cdiwrite operator and is meant to be used in conjunction with it. An operator call triggers a series of consecutive measurements whose number can be set with the first operator parameter. Its input file is meant to be the output file written by *cdiwrite*. The operator reads data in the form of 32-bit values from the input file and converts them to 64-bit values, reversing the data reduction performed by *cdiwrite*. The data rate of this conversion is measured in million values per second (MVal/s). The operator measures the overall duration and the data rate of the read operation of the input file in seconds and Megabytes per second (MB/s), respectively. After the runs have finished, the operator calculates and prints the mean values for its metrics. The -float option here also switches off the data reduction step with similar consequences as for *cdiwrite*. The option is also still experimental and only available from CDO version 1.7.2 onwards, limiting its use for our comparison.

4. Measurement methodology

It is a fundamental problem of every kind of measurement that it can be influenced by factors inherent in its environment. A controlled environment ideally eliminates all known factors that may affect the measurement results. However, unknown factors may still persist. It is an open question whether we can identify all influencing factors. Even once identified, it is very unlikely that we can eliminate all of them.

4.1 A controlled environment

Although the environment for any I/O performance measurement is artificial, it is not always feasible to exert full control over it. Various constraints may force us to accept only a limited degree of control. Two different environments have been proposed for our measurements:

- A Debian Jessie 64-bit Linux workstation at MPI-M with four Intel Core i5-4690 CPUs with 3.50 GHz clock speed and 8 GB of RAM. This system offers is a high probability that no other users will access the resource during the measurement process and a high degree of control for the experimenter. A known factor here is system administration, which does have access to the computer over the network and may divert resources while the measurements are running and adversely affect I/O performance. This is rather unlikely, though. And it is possible to ask the administrators to refrain from accessing the workstation for the duration of the measurements. Another known factor is memory caching by the operating system. This undoubtedly affects I/O performance and will have to be taken into account when interpreting the data.

- The Mistral HPC cluster at the German Climate Computing Center (DKRZ) with a peak performance of 3.14 PetaFLOPS and consisting of 3,000 compute nodes, 240 Terabytes of memory, and 54 Petabytes of usable disk space. But since the measurement operators do not parallelise their work, only single cores are needed. Mistral provides 5 dedicated nodes for interactive data processing and analysis and 32 nodes for pre- and post-processing of data, each with 2 x 12-core Intel Xeon E5-2680 v3 Haswell processors with a clock speed of 2.50 GHz, 30 MB of cache, and 256 GB

of main memory. This is an environment well-suited for M4 HR simulation runs, but this does not mean that it will produce data rates comparable to those of the actual simulation runs performed on Mare Nostrum III at Barcelona Supercomputing Center (BSC), see D9.6, as it differs in many details known and unknown to us from the environment the original simulation runs were conducted in. Such a multi-user environment is detrimental to sensitive data rate measurements as it cannot be controlled by the experimenter.

CDO I/O performance measurements on Mistral are meant to form the link to M4 HR simulation runs by providing data for different CDO version to identify any bottlenecks in model performance during the post-processing step. However, Mistral and Mare Nostrum III at BSC do not bear much resemblance. Beyond that, the Mistral multi-user environment cannot be considered a controlled environment. It is in heavy use and from a user perspective there is no way around its workload manager. One possible way to avoid this would be to use the maintenance downtime of Mistral to conduct measurements unaffected by multi-user behaviour. This is a perspective for future work and would depend on cooperation from DKRZ. However, to be able to compare directly to the M4 HR simulation runs, the different architectures of the nodes between Mistral and Mare Nostrum III would still have to be taken into account. Ideally though, the same cluster should be used for this comparison.

As a conclusion to these considerations about the experimental setting, the local workstation currently provides a more suitable environment for us than the Mistral HPC cluster to produce measurement data that hold up to scrutiny. Therefore, the CDO write and read data rate measurements are conducted on a local workstation only.

4.2 CDO compilation

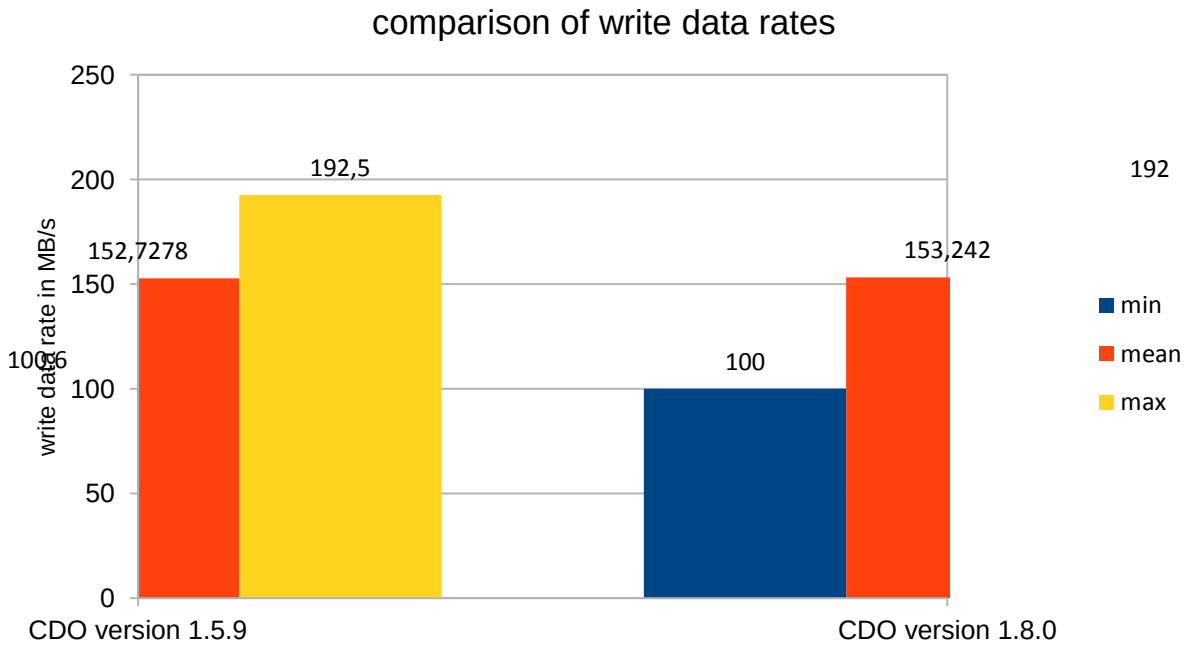
The software environment provided for the workstations available at MPI-M uses environment modules to dynamically load and unload often required software by changing environment variable settings. Since the availability of older CDO versions like 1.5.9 depends on the age of the workstation, a four-year-old CDO version is not available on the fairly new machine that has been used for measuring the data rates. However, this would not have guaranteed that linked CDO versions were compiled in exactly the same manner anyway. Without such a guarantee, the results would have been difficult to compare. So the desired CDO versions have been compiled manually with the same compilers and preset compiler options for the machine we conducted our experiments on.

4.3 Measurement timing

Since the workstation used for data rate measurement is only employed as a single-user machine, the timing of individual measurements is not as crucial as in a multi-user environment where access of multiple users is to be expected. In such an environment, averaging measurements conducted over a longer period of time would be necessary. The only known similar factors in the workstation environment are access and work patterns of the single user and access and work patterns of the system administrators. Both have been eliminated by timing the measurements in a manner that avoids collisions with existing work patterns and by ensuring the cooperation of the system administrators. But in order to discount measurement outliers of any source, averaging of measurement results is still necessary, albeit of measurements taken over a shorter time period. To achieve this, we have conducted the measurements in two-hour intervals during a 48-hour period over the weekend using cron jobs to time them. We assume that the system time ran unaffected during that time period.

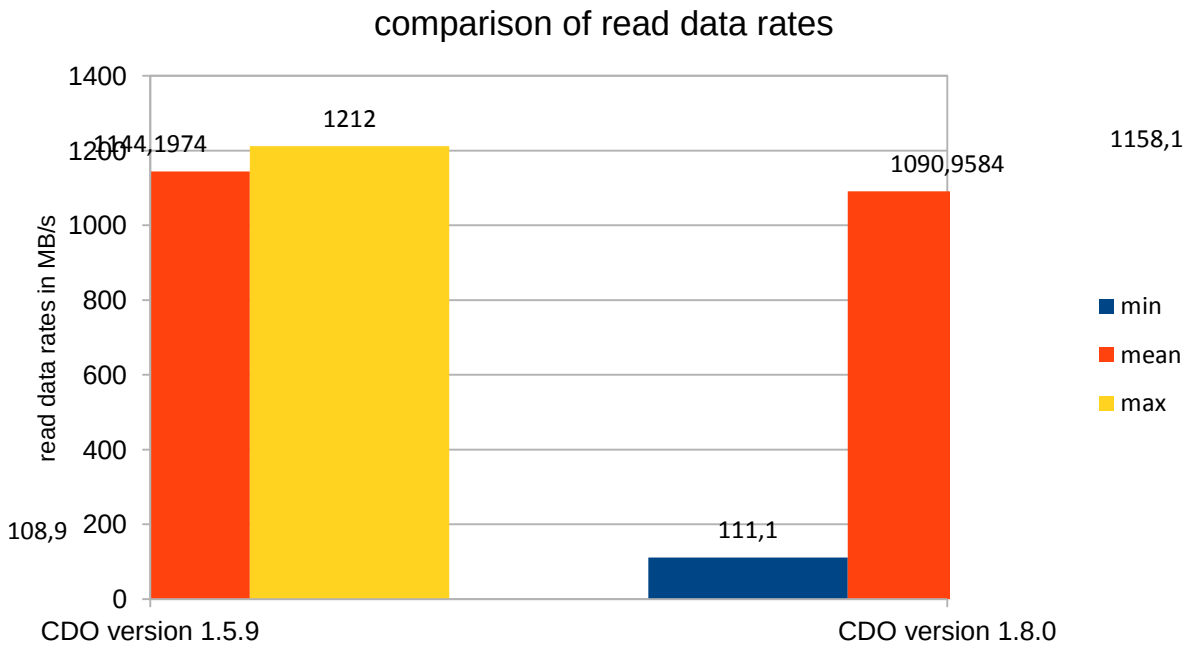
5. Measurement results

The measurement results are presented in two diagrams for write and read operations for both CDO versions. They show the average data rates that were computed from all measurement runs conducted during the 48-hour time period. These are presented alongside the minimal and maximal data rates recorded for each CDO version.



The minimal and maximal write data rates recorded show that during the entire 48-hour time frame, there have been no significant outliers in the write data rate measurements for both CDO versions. With nearly identical values for both minimal and maximal write data rates between the CDO versions, their performance has remained consistent and has not dropped below 100.6 MB/s for 1.5.9 and 100 MB/s for 1.8.0 or climbed higher than 192.5 MB/s for 1.5.9 and 192 MB/s for 1.8.0.

The arithmetic mean over all data rate measurements from CDO version 1.8.0 is slightly higher than the corresponding mean value computed from version 1.5.9. It has risen from 152.73 MB/s to 153.24 MB/s. It remains to be concluded whether this slight increase can be considered significant or is negligible.



The minimal read data rates shown present significant outliers in the measurement runs for both CDO versions. A look at the raw data shows that these low read data rates are always recorded during the first measurement run of each two-hourly operator call. With the beginning of the second measurement run, the performance is consistently close to the arithmetic mean recorded here. This large increase in speed can be attributed to memory caching. Since the same data is read several times during the operator call, reading is sped up by the operating system´s caching mechanism, keeping significant parts of the data in memory and avoiding costly read operations when possible. The consistently high read data rates of 1144.2 MB/s for 1.5.9 and 1090.96 MB/s for 1.8.0 can in all likelihood be attributed to this. Since the first measurement runs of each operator call are unaffected by this, averaging over the read data rates of only the first measurement runs for each CDO version produces average read data rates that are more representative of the IO performance of the CDO system itself. This creates average read data rates of 112.88 MB/s for 1.5.9 and 112.1 MB/s for 1.8.0 which are significantly lower than the average read data rates calculated. The average data rates and the maximum values shown in the diagram are therefore more an indication of the underlying operating system´s caching performance than the CDO system´s IO performance.

6. Conclusion

The data rate measurements conducted here provide the basis for an assessment of CDO I/O performance gains achieved from versions 1.5.9 and 1.8.0. No direct comparison can be made with any model performance bottlenecks that might be revealed during comparative M4 HR simulation runs using the same CDO versions in their post-processing steps. The main reason for this lack of comparability is that the data rate of write and read operations is not the only factor influencing I/O performance. Nevertheless, the data collected here can serve as an indicator and can be used to support or call into question conclusions about I/O performance bottlenecks that may be drawn from the comparative simulation runs.

For D9.5 we chose to perform these data rate measurements on a local workstation as this is the only available environment that allows us to eliminate several known factors that influence the measurement results. The measurement itself was conducted within 48 hours

during the weekend and has gathered the data of 20 consecutive write and read operations, 40 in all, every two hours, generating 1000 data measurements over a 48-hour time period.

Setting the results of the measurements in the context of the performance for the M4HR demonstrator (as documented in D9.6), allows for a (crude) estimate of the post-processing performance with CDO, at least as far as reading and writing the data from/to disk is concerned: The computationally more expensive of the two models used in the M4HR demonstrator (EC-Earth 3.2) has a data intensity of 0.0045 GB/CH and computational costs of 14251 CHPSY (see table 1 in D9.6 and metrics definitions in [1]). Assuming that all the data for one simulated year has to be read and written by CDO once, and taking into account the mean rates for CDO 1.8.0, this yields a post-processing speed of about 173.5 SYPD, compared to the model speed of 1.2 SYPD. This shows that reading and writing the data for post-processing is orders of magnitude faster than the model run itself, thus it does not indicate a performance bottleneck. Moreover, the read/write rates allow for the sequential processing of ensemble member output of M4HR experiments, without slowing the workflow down.

A few comments about assumptions and shortcomings in the above estimate: The assumption that all of the model data needs to be written seems pessimistic, since post-processing usually reduces the data volume. On the other hand, it is not immediately clear that the data has to be read only once, as this depends on the post-processing workflow.

A shortcoming is, of course, that data read and write rates are highly dependent on the hard- and software of the platform under consideration, thus the comparison may seem highly generalising. This is true, and the conclusion should be interpreted qualitatively, not quantitatively. As the platform used in the tests for CDO can not be considered a high-performance I/O architecture, it seems safe to use the data rates as a lower limit.

A serious shortcoming is that the analysis takes only the read and write operations from/to the disk into account, while complete post-processing workflows are computationally expensive (CPU, memory, and possibly the interconnect). Thus, the approach can only yield an estimate for the data reading and writing. Given that limitation, the conclusion remains that no indication of a bottleneck in the post-processing of the M4HR data with CDO has been found.

References

[1] Balaji, V., Maisonnave, E., Zadeh, N., Lawrence, B. N., Biercamp, J., Fladrich, U., Aloisio, G., Benson, R., Caubel, A., Durachta, J., Foujols, M.-A., Lister, G., Mocavero, S., Underwood, S. and Wright, G.: *CPMIP: measurements of real computational performance of Earth system models in CMIP6.* Geosci. Model Dev., 10, 19-34, 2017. doi:10.5194/gmd-10-19-2017