Centro Euro-Mediterraneo sui Cambiamenti Climatici

Research Papers
Issue RP0141
October 2012

Scientific Computing and Operation Division (SCO)

Common Pitfalls Coding a Parallel Model

By **Italo Epicoco**
University of Salento & CMCC
italo.epicoco@unisalento.it

Silvia Mocavero
CMCC
silvia.mocavero@cmcc.it

Alessandro D'Anca
CMCC
alessandro.danca@cmcc.it

and **Giovanni Aloisio**
University of Salento & CMCC
giovanni.aloisio@unisalento.it

SUMMARY The process for developing a climate model often involves a wide community of developers. All of the code releases can be classified into two groups: (i) improvements and updates related to modeling aspects (new parameterizations, new and more detailed equations, remove of model approximations, and so on); (ii) improvement related to the computational aspects (performance enhancement, porting on new computing architectures, fixing of known bugs, and so on). The developing process involves both programmers, scientific experts, and rarely also computer scientists. The new improvements and developments are mainly focused on the scientific aspects and, in second stage, on the computing performance. The developments to improve the physic model often does not care about its impacts on the computational performances. This poses some issue in the developing process; after a new implementation, the code must be revised after new implementation to face out with the performance issues. In this work we analyze 5 different releases starting from the NEMO v3.2 (to be considered as our reference) and evaluate how new developments impact on the computational performances.

The research leading to these results has received funding from the Italian Ministry of Education, University and Research and the Italian Ministry of Environment, Land and Sea under the GEMINA project.

INTRODUCTION

The evolution and reliability of NEMO [1] are organized and controlled by a European Consortium created in 2008 between CNRS (France), Mercator-Ocean (France), NERC (UK), UKMO (UK) and, since 2011, CMCC (Italy) and INGV(Italy). The purpose of the Consortium is to set up appropriate arrangements for the successful and sustainable development of the NEMO System as a well-organized, state-of-the-art ocean model code system suitable for both research and operational work. In order to achieve this goal the Consortium Members have agreed on:

- the resources they will commit each year to the NEMO System Team which will maintain the code;
- the arrangements for managing and coordinating the work of the NEMO System Team and for setting its priorities;
- the arrangements for the Intellectual Property rights over the Software to make the code freely available under an appropriate version of the CeCILL Free Software License with the aim of attracting a critical mass of scientists to use the software and contribute developments to be incorporated into it.

The NEMO System Team is in charge of the maintaing of the reference NEMO code and its distribution through the NEMO System web server. The NEMO System Team shall carry out the Work Plan proposed by the Developers Committee and approved by the Steering Committee. This Work Plan will include:

- incorporation into NEMO System of new developments (scientific or technical);
- reorganization of the code to improve its readability, orthogonality or structure;
- optimization of NEMO System on the architectures available in the Consortium;
- maintenance of the scientific paper and on-line documentation;
- configuration control of the available versions of NEMO System;
- testing and release of new versions (typically once or twice a year);
- making NEMO System readily available to the scientific community and to the Consortium Members;
- providing assistance to new users;
- practical support for user meetings (held typically once a year);
- assistance in scientific development in an area of high priority.

The Work Plan, describing the work which each of the Consortium Members will contribute to the NEMO development, is updated annually according to the following schedule:
31 March: Report of work by the NEMO System Team delivered by the NEMO Project Manager to Steering Committee;
Not later than one month before the Steering Committee: draft of Work-Plan for following year prepared by NEMO Project Manager and approved by NEMO Scientific Leader, following consultation of Developers Committee and discussion with all Consortium Members;
Not later than the 30th of November: Work-Plan for following year agreed by Steering Committee;
Before the end of the year: All schedules to Agreement updated for following year.

The developing process involves both programmers, modeling experts, and rarely also computer scientists. The improvements and new developments are mainly focused on modeling and, in a second stage, on computing performance. This often poses some issue in the developing process where the code must be revised after new implementations to face out with the performance issues. In this work we take 5 different releases starting from the NEMO v3.2 (to be considered as our reference) and evaluate how new modeling developments impact on the computational performances. The NEMO releases are shortly described below:

nemo_v3.2.2 as reference

nemo_v3.3.0 new developments include: (i) merge between TRA and TRC; (ii) introduction of a modified leapfrog-Asselin filter time stepping scheme; (iii) additional scheme for iso-neutral mixing; (iv) addition of a Generic Length Scale vertical mixing scheme; (v) addition of the atmospheric pressure as an external forcing on both ocean and sea-ice dynamics; (vi) addition of a diurnal cycle on solar radiation; (vii) river runoffs added through a non-zero depth, and having its own temperature and salinity; (viii) CORE II normal year forcing set as the default forcing of ORCA2-LIM configuration; (ix) generalization of the use of *fldread* for all input fields; (x) optional application of an assimilation increment.

nemo_v3.3.1 this version matches with 3.3.0 code with dynamic allocation added. The main implications are: (i) the number of processors are now part of the namelist *nam_mpp*; (ii) the definition of the configuration parameters in par_*h90 have to be slightly changed; (iii) the coding rules have been updated.

nemo_v3.4a this is a major release and relevant improvements have been added. From scientific point of view: (i) new pressure gradient suitable for s-coordinate; (ii) completion of Griffies iso-neutral diffusion; (iii) Pacanowski-Philander scheme for computation of Ekman depth added; (iv) a new bulk formulae added; (v) a drag coefficient compute by wave model introduced; (vi) tidal potential forcing added; (vii) Netpune effect parametrization; (viii) point to point MPI communication for north fold; (ix) sub time stepping for biogeochemistry models when using nonlinear free surface allowed; (x) improvement in PISCES.
From coding point of view some simplification have been made: (i) simplification of dynamic allocation; (ii) completion of merging between TRA and TRC; (iii) more flexible definition of BDY input data; (iv) simplification of interfaces toward biogeochemical models; (v) interface with CICE in coupled mode; (vi) use of *fldread* to read/interpolate data for passive tracers and dynamical input data for OFFLINE configurations.

nemo_v3.4b this version corresponds to 3.4a code with some performance improvement on dynamical allocation using the stack: no explicit coding nor POINTER.

DEFINITION OF TESTS

Tests have been done using the GYRE configuration as equivalent to $1/4°$ (cfg=48) and $1/12°$ (cfg=144) resolutions. In our case the GYRE configuration makes no I/O (nor write nor restart), and runs 720 time steps. The CPU time sums CPU time in seconds for time steps between 2 and 719. In the namelist, *nn_bench=1*, *rn_rdt = 180*. Cpp keys: *key_gyre key_dynspg_flt key_ldfslp key_zdftke key_mpp_mpi*.

COMPUTING ENVIRONMENT

Tests have been performed on the IBM Power6 cluster at CMCC. The compiler name, flags and environment variables, chosen to improve the performance, are reported in the following.

Computer used:

Name : Calypso (calypso.cmcc.it)

Architecture : IBM P575 node (32 cores Power6 4.7GHz per node)

Compiler : mpxlf90_r

Compiler flags : -qstrict -qfree=f90 -O3 -qrealsize=8 -qextname -qsource -q64 -qlargepage -qmaxmem=-1 -qarch=pwr6 -qtune=pwr6 -q64

Environment variables :

```
export MP_INSTANCES=4
export MP_WAIT_MODE=poll
export MP_POLLING_INTERVAL=30000000
export MP_SHARED_MEMORY=yes
export MP_EUILIB=us
export MP_EUIDEVICE=sn_all
export LDR_CNTRL=TEXTPSIZE=64K@STACKPSIZE=64K@DATAPSIZE=64K
export MP_TASK_AFFINITY=core
```

RESULTS

The wall clock time increases at each new release except for the last one, as shown in figure 1. The biggest increment appears between release v3.3.1 and v3.4a. The wall clock time for the release v3.4.a is almost double with respect to the wall clock time of the v3.2.2. To evaluate the use of SMT, a decomposition 8x8 has been used. Enabling SMT, all of the 64 processes have been mapped on one node. Disabling SMT, 2 nodes have been allocated. Figure 2 shows that using half number of processors, the CPU time is less than double.

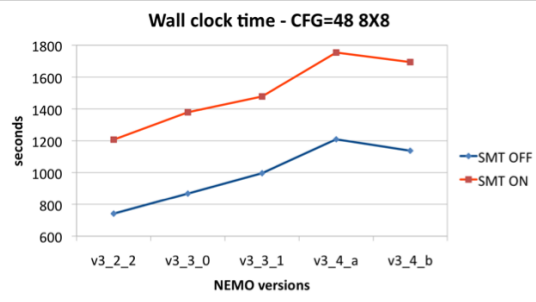

Figure 1:
GYRE wall clock time: resolution given by cfg=48; 64 processes (on one node enabling SMT, on two nodes disabling SMT)

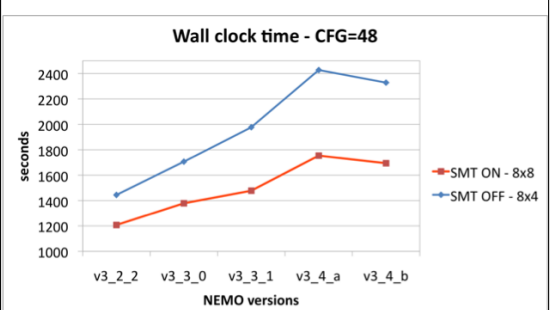

Figure 2:
SMT evaluation. One node has been used either with 32 and 64 processes

In the previous test the number of nodes has been fixed to 1 and the decomposition has been changed. The results show that the a computing node is better exploited when SMT is active. The difference of each new release with respect to the previous one is reported in figure 3.

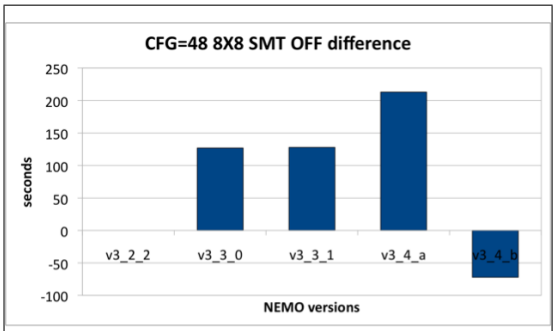

Figure 3:
GYRE wall clock time difference between a NEMO release and the previous one: resolution given by cfg=48; 64 processes on one node enabling SMT

The most significant increment occurs during the transition from v3.3.1 to v3.4.a. The v3.4b reduced the time with respect to the v3.4a.

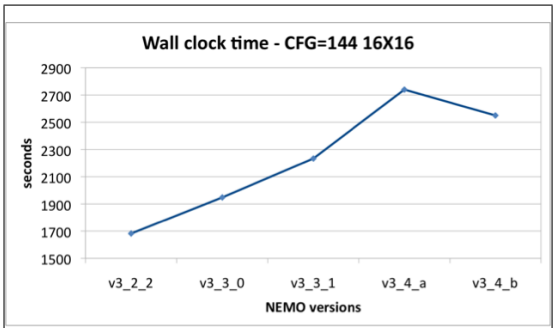

Figure 4:
GYRE wall clock time: resolution given by cfg=144; 256 processes disabling SMT

The wall clock time behaviour for cfg=144 confirms the increment inserted at each new release. In the test reported in figure 4, the SMT is disabled.

The behaviour observed for the configuration with cfg=48 persists when cfg=144, as shown

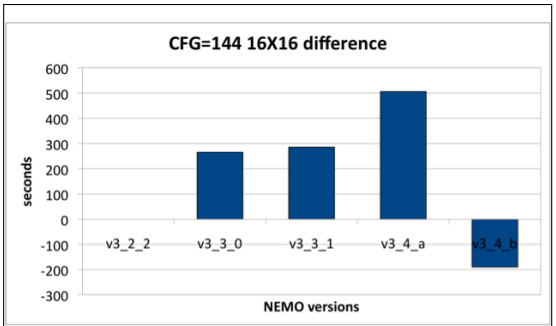

Figure 5:
GYRE wall clock time difference between a NEMO release and the previous one: resolution given by cfg=144; 256 processes disabling SMT

in figure 5. A more detailed analysis has been made. We used the configuration cfg=48, SMT OFF and a domain decomposition of 8x8. We used *gprof* to trace each routine called by the *stp* main routine along the different releases. Table 1 gives the details at low level.

Considering only those routines that have a variation greater than 5% during the transition from one release to the next one, we can focus our attention only on the last 9 routines and namely: *tra_zdf* - *tra_unswap* - *tra_ldf* - *tra_adv* - *zdf_tke* - *ldf_slp* - *dyn_zdf* - *dyn_vor* - *dyn_spg*.

Figure 6 shows the wall clock time spent on those routines and highlights how the time changes during the evolution of the code.

Figure 7 better empathizes which routine is responsible for the increment of the wall clock time during the transition from a release to the next one. Only the difference in time is shown.

The increment of time during the transition from v3.2.2 to v3.3.0 is due to the *tra_ldf* and *zdf_tke* routines. The routines responsible for the increment of time during the transition from v3.3.0 to v3.31 are *tra_ldf* and *tra_adv*. Finally the routine *dyn_zdf* is the solely routine responsible for the increment of time in the release v3.4a.

Table 1
Function tracing along successive releases

Routine	v3.2.2	v3.3.0	v3.3.1	v3.4a	v3.4b
zdf_mxl	3.81	3.80	2.92	2.92	2.93
zdf_evd	11.13	11.43	10.21	10.95	10.29
zdf_bfr	0.43	0.32	0.46	0.38	0.61
tra_sbc	0.13	0.78	0.70	0.75	0.65
tra_qsr	1.31	5.16	5.11	5.11	5.16
tra_nxt	6.48	6.21	5.58	6.45	6.22
stp_ctl	2.20	2.49	2.47	2.35	2.42
ssh_wzv	16.30	15.88	15.57	16.46	16.09
ssh_nxt	0.05	0.08	0.08	0.12	0.10
sbc	3.98	4.75	4.65	4.46	4.33
iom_setkt	0	0	0	0	0
iom_close	0	0	0	0	0
eos_insitu	3.78	3.73	3.68	3.83	3.76
eos_bn2	10.41	9.89	8.38	8.08	8.20
dyn_nxt	6.64	6.21	6.33	6.53	6.48
dyn_ldf	9.64	9.74	10.09	10.14	9.76
dyn_hpg	7.09	7.25	5.60	8.98	6.09
dyn_bfr	0.63	0.59	0.38	0	0
dyn_adv	18.09	18.19	17.13	17.74	18.94
day	0	0	0.01	0.01	0
tra_zdf	32.97	43.26	28.36	48.79	29.67
tra_unsw.	0	14.00	13.97	0	0
tra_ldf	53.44	92.96	161.11	156.81	160.36
tra_adv	203.71	222.03	281.84	299.61	280.56
zdf_tke	74.73	105.87	113.59	115.36	123.78
ldf_slp	99.68	90.32	97.45	104.19	100.52
dyn_zdf	37.27	37.14	37.98	214.11	186.08
dyn_vor	11.14	11.05	8.22	20.16	7.97
dyn_spg	22.90	22.20	34.20	32.99	33.52

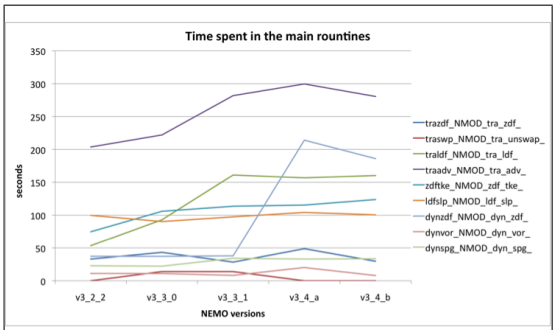

Figure 6:
Wall clock time for each main routine

MFS CONFIGURATION

The performance comparison between NEMO releases has been applied also to the MFS16 configuration [2]. This is a production configu-

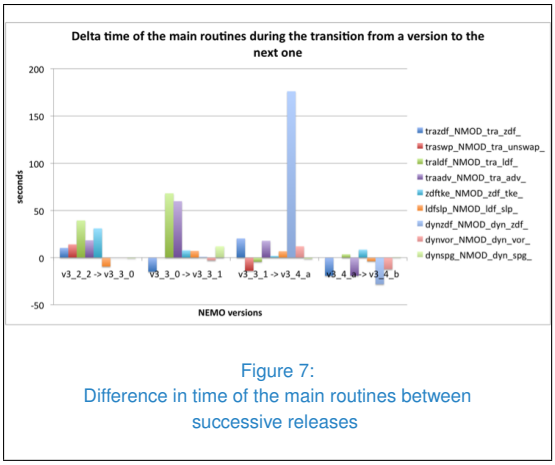

Figure 7:
Difference in time of the main routines between successive releases

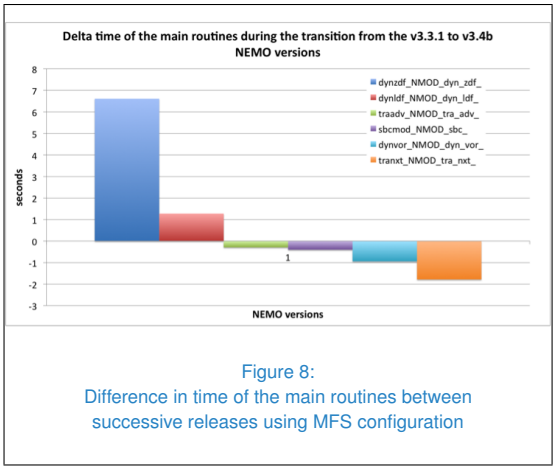

Figure 8:
Difference in time of the main routines between successive releases using MFS configuration

ration with a scientific interest for CMCC. The tests have been carried out with the following data:

- Domain decomposition: 8x8
- SMT: disabled (32 core per node)
- Total number of nodes: 2
- Number of time steps: 144
- I/O: creation of output and restart file only at the end of the simulation

The wall clock time between v3.3.1 and v3.4a releases differs only for 5.7%. Having a look at each routine we can note that the increment of the execution time of the *dyn_zdf* routine is partially balanced by a reduction of the execution time of the *tra_nxt* and other routines. Figure 8 reports the timing only for those routines with a difference greater then 5% of the execution time measured in release v3.3.1

Table 2
Execution time

v3.3.1	v3.4b
104 sec	110 sec

Also for this configuration the inefficiencies previously described for *dyn_zdf* have a deep impact on the performance. The examined configuration does not activate any other routine with relevant performance inefficiency. The detailed values for each routine are reported in the table 3.

Table 3
Delta Time from v3.3.1 to v3.4b for MFS configuration

Routine	Time (secs)
dyn_zdf	6.61
dyn_ldf	1.28
obc_rad	0.53
zdf_tke	0.45
tra_adv	-0.29
sbc	-0.40
dyn_vor	-0.94
tra_nxt	-1.78

ANALYSIS

To identify exactly which is the reason of this increment of time in *dyn_zdf* routine, we focused the attention only on the releases v3.3.1 and v3.4.a and only on the *dyn_zdf* routine. Actually the *dyn_zdf* calls the *dyn_zdf_imp*. We have instrumented a region within the *dyn_zdf_imp* for both releases. The instrumented region excludes the calls to the *wrk_array* routines (that has been introduced in v3.4a).

```
Total execution time of instrumented code (wall time): 102.08002 seconds

 ########  Resource Usage Statistics  ########

 Total amount of time in user mode            : 101.119562 seconds
 Total amount of time in system mode          : 0.403783 seconds
 Maximum resident set size                    : 440040 Kbytes
 Average shared memory use in text segment    : 309698 Kbytes*sec
 Average unshared memory use in data segment  : 43993006 Kbytes*sec
 Number of page faults without I/O activity   : 7468
 Number of page faults with I/O activity      : 0
 Number of times process was swapped out      : 0
 Number of times file system performed INPUT  : 0
 Number of times file system performed OUTPUT : 0
 Number of IPC messages sent                  : 0
 Number of IPC messages received              : 0
 Number of signals delivered                  : 0
 Number of voluntary context switches         : 933
 Number of involuntary context switches       : 111

 #######  End of Resource Statistics  ########

 Instrumented section: 1 - Label: dyn_zdf_imp - process: 0
 file: dynzdf_imp.F90, lines: 82 <--> 266
  Count: 72
  Wall Clock Time: 3.894087 seconds
  Total time in user mode: 3.87232864051871 seconds
  Average duration: 0.0540845
  Standard deviation: 5.23728e-315

 Set: 1
 Counting duration: 3.890726357 seconds
  PM_FPU_1FLOP (FPU executed one flop instruction )          :     1035763200
  PM_FPU_FMA (FPU executed multiply-add instruction)         :      273715200
  PM_FPU_FSQRT_FDIV (FPU executed FSQRT or FDIV instruction) :      556761672
  PM_CYC (Processor cycles)                                  :    18215433925
  PM_RUN_INST_CMPL (Run instructions completed)              :     7225690646
  PM_RUN_CYC (Run cycles)                                    :    18295004672


  Utilization rate                              :       99.441 %
  Flop                                          :     3810.240 Mflop
  Flop rate (flops / WCT)                       :      978.468 Mflop/s
  Flops / user time                             :      983.966 Mflop/s
  FMA percentage                                :       41.805 %
```

(a) v3.3.1

```
Total execution time of instrumented code (wall time): 125.444905 seconds

 ########  Resource Usage Statistics  ########

 Total amount of time in user mode            : 124.353644 seconds
 Total amount of time in system mode          : 0.423365 seconds
 Maximum resident set size                    : 404712 Kbytes
 Average shared memory use in text segment    : 412879 Kbytes*sec
 Average unshared memory use in data segment  : 49748308 Kbytes*sec
 Number of page faults without I/O activity   : 6843
 Number of page faults with I/O activity      : 81
 Number of times process was swapped out      : 0
 Number of times file system performed INPUT  : 0
 Number of times file system performed OUTPUT : 0
 Number of IPC messages sent                  : 0
 Number of IPC messages received              : 0
 Number of signals delivered                  : 0
 Number of voluntary context switches         : 1710
 Number of involuntary context switches       : 142

 #######  End of Resource Statistics  ########

 Instrumented section: 1 - Label: dyn_zdf_imp - process: 0
 file: dynzdf_imp.F90, lines: 94 <--> 261
  Count: 72
  Wall Clock Time: 21.293118 seconds
  Total time in user mode: 21.2582213690476 seconds
  Average duration: 0.295738
  Standard deviation: 5.24284e-315

 Set: 1
 Counting duration: 4.511261185 seconds
  PM_FPU_1FLOP (FPU executed one flop instruction )          :      855381600
  PM_FPU_FMA (FPU executed multiply-add instruction)         :      276825600
  PM_FPU_FSQRT_FDIV (FPU executed FSQRT or FDIV instruction) :      559872072
  PM_CYC (Processor cycles)                                  :    99998673320
  PM_RUN_INST_CMPL (Run instructions completed)              :     7112912225
  PM_RUN_CYC (Run cycles)                                    :   100126202339


  Utilization rate                              :       99.836 %
  Flop                                          :     3648.521 Mflop
  Flop rate (flops / WCT)                       :      171.347 Mflop/s
  Flops / user time                             :      171.629 Mflop/s
  FMA percentage                                :       48.900 %
```

(b) v3.4a

```
Instrumented section: 1 - Label: dyn_zdf_imp - process: 0
file: dynzdf_imp.F90, lines: 82 <--> 266
 Count: 72
 Wall Clock Time: 3.899114 seconds
 Total time in user mode: 3.87651308205782 seconds
 Average duration: 0.0541544
 Standard deviation: 5.23892e-315

Set: 3
Counting duration: 3.895389757 seconds
 PM_CYC (Processor cycles)                      :    18235117538
 PM_LD_MISS_L1 (L1 D cache load misses)         :        5007385
 PM_ST_MISS_L1 (L1 D cache store misses)        :      284925782
 PM_INST_CMPL (Instructions completed)          :     7223792619
 PM_RUN_INST_CMPL (Run instructions completed)  :     7225864323
 PM_RUN_CYC (Run cycles)                        :    18316854462


 Utilization rate                              :       99.420 %
 MIPS                                          :     1852.675 MIPS
 Instructions per cycle                        :        0.396
 Total L2 data cache accesses                  :      289.933 M
 % accesses from L2 per cycle                  :        1.590 %
 L2 traffic                                    :    35392.232 MBytes
 L2 bandwidth per processor                    :     9076.993 MBytes/s
```

(c) v3.3.1

```
Instrumented section: 1 - Label: dyn_zdf_imp - process: 0
file: dynzdf_imp.F90, lines: 94 <--> 261
 Count: 72
 Wall Clock Time: 21.194461 seconds
 Total time in user mode: 21.1442108726616 seconds
 Average duration: 0.294368
 Standard deviation: 5.24193e-315

Set: 3
Counting duration: 4.413331605 seconds
 PM_CYC (Processor cycles)                      :    99462367945
 PM_LD_MISS_L1 (L1 D cache load misses)         :      317388531
 PM_ST_MISS_L1 (L1 D cache store misses)        :      143627523
 PM_INST_CMPL (Instructions completed)          :     7102544202
 PM_RUN_INST_CMPL (Run instructions completed)  :     7112770907
 PM_RUN_CYC (Run cycles)                        :    99662106966


 Utilization rate                              :       99.763 %
 MIPS                                          :      335.113 MIPS
 Instructions per cycle                        :        0.071
 Total L2 data cache accesses                  :      461.016 M
 % accesses from L2 per cycle                  :        0.464 %
 L2 traffic                                    :    56276.374 MBytes
 L2 bandwidth per processor                    :     2655.240 MBytes/s
```

(d) v3.4a

Figure 9:
Floating point operations (a and b) and L1 cache misses (c and d): MFS configuration

The configuration used for the instrumented executable is characterized by cfg=48, a domain decomposition 8x8, SMT OFF and 72 total timesteps. We first checked if the increment of time was due to new operations introduced in the new release, we checked the total number of floating point operations (Flop counter) using the HPM (High Performance Monitoring) and we got the result in figure 9(a) and 9(b).
The total execution time in release v3.4a is about 23 seconds greater than v3.3.1 and the time spent within the instrumented section in release v3.4a reached 21.3 seconds against 3.9 seconds on the same region in release v3.3.1. This confirms what has been observed with *gprof*. The result says also that the management of the work arrays does not have a deep impact on the computing performance. The most part of the time increment in the v3.4a is due to *dyn_zdf_imp* routine. Having a look at the total number of floating point operations, the v3.4a executes less operations than v3.3.1 (3648.5Mflop against 3810Mflop). Hence the new release reduces the number of flops and in some way tries to optimize the execution. But the execution rate of the floating point operations is drastically low with respect to the v3.3.1 (171.3Mflop/s against 978.5 Mflop/s). This is typically due to a bad use of the memory hierarchy.

We used the hardware counters to check the L1 data cache misses (see figure 9(c) and 9(d)). The results highlight that in the new release an inefficiency due to the bad use of L1 and L2 caches has been introduced. The number of L1 misses of the v3.4a is 2 order of magnitude greater than the v3.3.1 (317,388,531 against 5,007,385); also the L2 data cache accesses is doubled in the v3.4a (461M against 290M). Having a look at the code, the lose of performance due to the bad cache usage is quite evident. The implementation of the *dyn_zdf_imp* in the release v3.3.1 accesses the 3D array sweeping the data following contiguous memory locations: as example see the snapshot in figure 10 where the elements of the arrays *avmu* and *umask* are accessed following the column wise order, indeed the first index (*ji*) iterates before the second index (*jj*) that iterates before the third index (*jk*).
In release v3.4a the loops have been restructured and the ordered access to the memory has been lost (see figure 11). In this case, the innermost loop, iterates on *jk* that is not the first index of the arrays.

Regarding the transition from v3.3.0 to v3.3.1, that corresponds to the introduction of the dynamical memory allocation, the major impact on performance can be seen on routines *tra_ldf* and *tra_adv*. But, contrary to what happened in the previous case where the code has been restructured. In this case both routines are the

```
DO jk = 1, jpkm1          ! Matrix
   DO jj = 2, jpjm1
      DO ji = fs_2, fs_jpim1   ! vector opt.
         zcoef = - p2dt / fse3u(ji,jj,jk)
         zzwi          = zcoef * avmu (ji,jj,jk  ) / fse3uw(ji,jj,jk  )
         zwi(ji,jj,jk) = zzwi  * umask(ji,jj,jk)
         zzws          = zcoef * avmu (ji,jj,jk+1) / fse3uw(ji,jj,jk+1)
         zws(ji,jj,jk) = zzws  * umask(ji,jj,jk+1)
         zwd(ji,jj,jk) = 1._wp - zwi(ji,jj,jk) - zzws
      END DO
   END DO
END DO
. . .
. . .
```

Figure 10:
Nested loops in release v3.3.1

```
DO jj = 2, jpjm1
DO ji = fs_2, fs_jpim1   ! vector opt.
   ikbum1 = mbku(ji,jj)
      zbfru = bfrua(ji,jj)

   DO jk = 1, ikbum1
      zcoef = - p2dt / fse3u(ji,jj,jk)
      zwi(ji,jj,jk) = zcoef * avmu(ji,jj,jk  ) / fse3uw(ji,jj,jk  )
      zws(ji,jj,jk) = zcoef * avmu(ji,jj,jk+1) / fse3uw(ji,jj,jk+1)
      zwd(ji,jj,jk) = 1._wp - zwi(ji,jj,jk) - zws(ji,jj,jk)
   END DO
. . .
. . .
```

Figure 11:
Nested loops in release v3.4a

same in v3.3.0 and v3.3.1 (apart for the calls to *wrk_in_use* and to *wrk_not_released* functions in v3.3.1). We focused on the *tra_ldf* that actually has only one call to the *tra_ldf_iso* routine (this is a leaf routine). We instrumented a region that excludes the calls to the work arrays functions. Figure 12(a) and 12(b) reports the values of the FLOP counters. The total number of floating point operations is exactly the same for both releases. This implies that the compiler performs the same optimizations regarding to the floating point operations. However the number of completed instructions in the v3.3.1 is almost twice the number of instructions of the v3.3.0. This implies that the compiler introduces more instructions (such as integer or load/store operation).

Taking into account the counters related to the cache misses, we can notice that there is not an evident difference. The L1D cache misses are of the same order of magnitude (see figure 12(c) and 12(d)). Once again we notice that the number of instructions in v3.3.1 is almost double even if the instruction rate (MIPS) for the v3.3.1 is greater than v3.3.0.

Analyzing the list file produced by the compiler, the transformations performed by the compiler and the assembler code, we can notice that, the access to an allocatable array (that is stored in the heap memory instead of the stack memory) produces a lot of register spilling that implies an increase number of load and store instructions. To confirm this observation we can see the counters related to the total number operations of the LSU (Load Store Unit): in v3.3.1 we have 11398M operations against 7651M (see figure 12(e) and 12(f)).

To conclude, the main reason for the performance decrement during the transition from v3.3.0 to the v3.3.1 release is due to the compiler that is not able to make the same optimizations when the code uses allocatable arrays and when the number of iterations of the loops are not known at compile time. The lack of performance observed during the transition to v3.4a release is mainly due to a worst use of the cache. The management of the work arrays, introduced in v3.4a, has not an evident impact on the performance.

```
Instrumented section: 3 - Label: trg_ldf_iso - process: 0
file: trgldf_iso.F90, lines: 126 <--> 299
 Count: 72
 Wall Clock Time: 9.430277 seconds
 Total time in user mode: 9.39108283545918 seconds
 Average duration: 0.130976
 Standard deviation: 5.24181e-315

Set: 1
Counting duration: 1.039553796 seconds
 PM_FPU_1FLOP (FPU executed one flop instruction )          :      6788490192
 PM_FPU_FMA (FPU executed multiply-add instruction)         :       279426240
 PM_FPU_FSQRT_FDIV (FPU executed FSQRT or FDIV instruction) :       745476480
 PM_CYC (Processor cycles)                                  :     44175653658
 PM_RUN_INST_CMPL (Run instructions completed)              :     20555471413
 PM_RUN_CYC (Run cycles)                                    :     44337529573


 Utilization rate                                :        99.584 %
 Flop                                            :     10329.249 Mflop
 Flop rate (flops / WCT)                         :      1095.328 Mflop/s
 Flops / user time                               :      1099.900 Mflop/s
 FMA percentage                                  :         7.907 %
```

(a) v3.3.0

```
Instrumented section: 3 - Label: trg_ldf_iso - process: 0
file: trgldf_iso.F90, lines: 129 <--> 301
 Count: 72
 Wall Clock Time: 15.469546 seconds
 Total time in user mode: 15.4158412872024 seconds
 Average duration: 0.214855
 Standard deviation: 5.2498e-315

Set: 1
Counting duration: 7.075332222 seconds
 PM_FPU_1FLOP (FPU executed one flop instruction )          :      6788494368
 PM_FPU_FMA (FPU executed multiply-add instruction)         :       279426240
 PM_FPU_FSQRT_FDIV (FPU executed FSQRT or FDIV instruction) :       745476480
 PM_CYC (Processor cycles)                                  :     72516117415
 PM_RUN_INST_CMPL (Run instructions completed)              :     38116579793
 PM_RUN_CYC (Run cycles)                                    :     72723619886


 Utilization rate                                :        99.653 %
 Flop                                            :     10329.253 Mflop
 Flop rate (flops / WCT)                         :       667.715 Mflop/s
 Flops / user time                               :       670.041 Mflop/s
 FMA percentage                                  :         7.907 %
```

(b) v3.3.1

```
Instrumented section: 3 - Label: trg_ldf_iso - process: 0
file: trgldf_iso.F90, lines: 126 <--> 299
 Count: 72
 Wall Clock Time: 9.368282 seconds
 Total time in user mode: 9.34484859396258 seconds
 Average duration: 0.130115
 Standard deviation: 5.23979e-315

Set: 3
Counting duration: 0.977843166 seconds
 PM_CYC (Processor cycles)                  :     43958167786
 PM_LD_MISS_L1 (L1 D cache load misses)     :        12237231
 PM_ST_MISS_L1 (L1 D cache store misses)    :       631162639
 PM_INST_CMPL (Instructions completed)      :     20549971597
 PM_RUN_INST_CMPL (Run instructions completed) :     20554549684
 PM_RUN_CYC (Run cycles)                    :     44052518810


 Utilization rate                                :        99.750 %
 MIPS                                            :      2193.569 MIPS
 Instructions per cycle                          :         0.467
 Total L2 data cache accesses                    :       643.400 M
 % accesses from L2 per cycle                    :         1.464 %
 L2 traffic                                      :     78540.023 MBytes
 L2 bandwidth per processor                      :      8383.610 MBytes/s
```

(c) v3.3.0

```
Instrumented section: 3 - Label: trg_ldf_iso - process: 0
file: trgldf_iso.F90, lines: 129 <--> 301
 Count: 72
 Wall Clock Time: 15.564379 seconds
 Total time in user mode: 15.5147778346088 seconds
 Average duration: 0.216172
 Standard deviation: 5.2474e-315

Set: 3
Counting duration: 7.174136792 seconds
 PM_CYC (Processor cycles)                  :     72981514934
 PM_LD_MISS_L1 (L1 D cache load misses)     :        15438471
 PM_ST_MISS_L1 (L1 D cache store misses)    :       636196507
 PM_INST_CMPL (Instructions completed)      :     38105317914
 PM_RUN_INST_CMPL (Run instructions completed) :     38118354642
 PM_RUN_CYC (Run cycles)                    :     73184223703


 Utilization rate                                :        99.681 %
 MIPS                                            :      2448.239 MIPS
 Instructions per cycle                          :         0.522
 Total L2 data cache accesses                    :       651.635 M
 % accesses from L2 per cycle                    :         0.893 %
 L2 traffic                                      :     79545.285 MBytes
 L2 bandwidth per processor                      :      5110.727 MBytes/s
```

(d) v3.3.1

```
Instrumented section: 3 - Label: trg_ldf_iso - process: 0
file: trgldf_iso.F90, lines: 126 <--> 329
 Count: 72
 Wall Clock Time: 9.326323 seconds
 Total time in user mode: 9.26628380463435 seconds
 Average duration: 0.129532
 Standard deviation: 5.24483e-315
 Exclusive duration: 0.060868 seconds

Set: 2
Counting duration: 0.920509060 seconds
 PM_INST_CMPL (Instructions completed)                  :     20552281216
 PM_LSU_LDF (LSU executed Floating Point load instruction) :      6794002768
 PM_FPU_STF (FPU executed store instruction)            :       856780440
 PM_CYC (Processor cycles)                              :     43588599017
 PM_RUN_INST_CMPL (Run instructions completed)          :     20558409852
 PM_RUN_CYC (Run cycles)                                :     43778740295


 Utilization rate                                :        99.356 %
 MIPS                                            :      2203.685 MIPS
 Instructions per cycle                          :         0.472
 Floating point load and store operations        :      7650.783 M
 Instructions per floating point load/store      :         2.686
```

(e) v3.3.0

```
Instrumented section: 3 - Label: trg_ldf_iso - process: 0
file: trgldf_iso.F90, lines: 129 <--> 331
 Count: 72
 Wall Clock Time: 15.007106 seconds
 Total time in user mode: 14.9584778420493 seconds
 Average duration: 0.208432
 Standard deviation: 5.24507e-315
 Exclusive duration: 0.062128 seconds

Set: 2
Counting duration: 6.601828792 seconds
 PM_INST_CMPL (Instructions completed)                  :     38019010436
 PM_LSU_LDF (LSU executed Floating Point load instruction) :      8670442390
 PM_FPU_STF (FPU executed store instruction)            :      2727081312
 PM_CYC (Processor cycles)                              :     70364679769
 PM_RUN_INST_CMPL (Run instructions completed)          :     38030961269
 PM_RUN_CYC (Run cycles)                                :     70505159457


 Utilization rate                                :        99.676 %
 MIPS                                            :      2533.401 MIPS
 Instructions per cycle                          :         0.540
 Floating point load and store operations        :     11397.524 M
 Instructions per floating point load/store      :         3.336
```

(f) v3.3.1

Figure 12:
Floating point operations (a and b), L1 cache misses (c and d) and LSU operations (e and f): MFS configuration

Bibliography

[1] G. Madec. Nemo ocean engine. Technical Report Technical Report 27 ISSN No 1288-1619, Institut Pierre-Simon Laplace (IPSL), 2008.

[2] P. Oddo, M. Adani, N. Pinardi, C. Fratianni, M. Tonani, and D. Pettenuzzo. A nested atlantic-mediterranean sea general circulation model for operational forecasting. *Ocean Sci.*, 5(4):461–473, 2009.

Visit **www.cmcc.it** for information on our activities and publications.

The Euro-Mediteranean Centre on Climate Change is a Ltd Company with its registered office and administration in Lecce and local units in Bologna, Venice, Capua, Sassari, Viterbo, Benevento and Milan. The society doesn't pursue profitable ends and aims to realize and manage the Centre, its promotion, and research coordination and different scientific and applied activities in the field of climate change study.