AROME Forecast migration & optimization at FMI

April 13-16, 2010

All Staff Meeting (ASM)

Krakow, Poland

Sami Niemelä, FMI

Niko Sokka, FMI

Sami Saarinen, CSC

Outline

- **Introduction**
- **Fundamental performance problems**
- **Migration status & some results**
- **Issues requiring urgent attention**
- **Good news for benchmarking**
- **Conclusions**

Introduction

- **FMI has acquired two Cray XT5m clusters with 1968 cores in each**
- **AROME Forecast migration**
  - Harmonie version **35h1.2** (tag 6935) copied to the local FMI-branch
  - Over 3.3 million lines of code (Fortran+C), over 9000 source objects
- **CSC – IT Center for Science Ltd.,(Espoo, FI) was responsible for migrating & optimizing AROME to FMI's Cray XT5**

Introduction *(cont'd)*

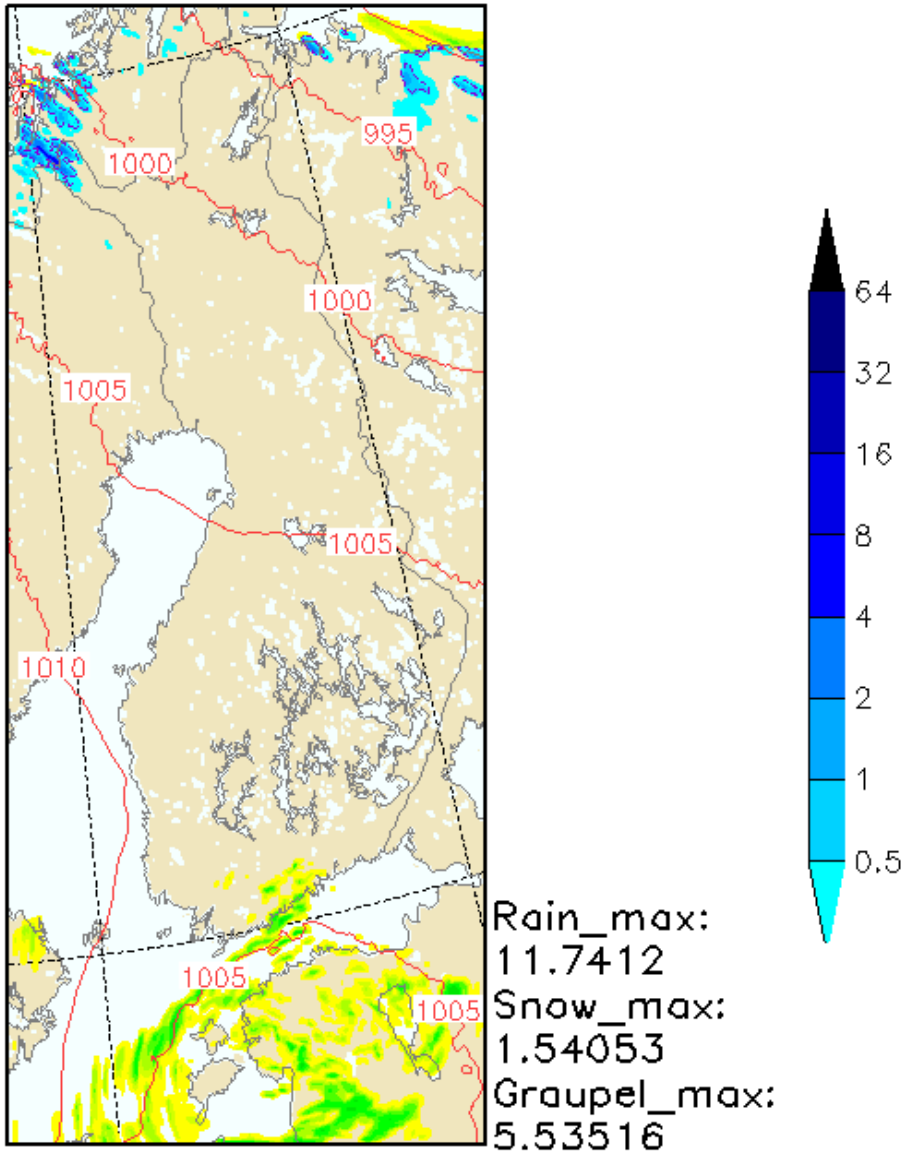
- **AROME model resolution over Finland : 300 x 600 x L40 (2,5km)**
  - 24h forecast with radiation calculation every 15th step (every 15 min)
  - Full output every 15th step (both Atmos & SURFEX data)
  - Boundary input every 60th step
- **24h AROME FC *previously* took 2-3hours on SGI Altix ~ 100 cores**
  - Was run twice a day in a non time-critical manner
- **The near future targets for FMI's AROME on Cray XT5m are**
  - **Repeat** 24h Forecast **every 3hrs** in a production mode
  - **Increase** computational area & levels to ca. **600 x 600 x L60 (2,5km)**

AROME 09OCT2009 00 UTC Forecast. Precipitation [mm 1h⁻¹]
09OCT2009 07:00 UTC (aro33h1,2.5km)

Now :
NLON x NLAT x NLEV

300 x 600 x L40

Later most likely ca.
600 x 600 x L60
i.e. double the width



About FMI's Cray XT5m

- **Introduction to the system**
  - **2 identical clusters, 17.3 TFlop/s peak for each, ca. 35TF total**
  - **Hex-core AMD Opteron 2.2GHz Istanbul chip**

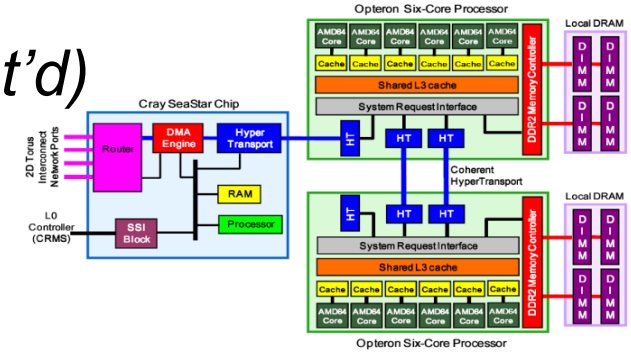
    12 (= 2 x 6) cores in a shared memory node

    8.8 GFlop/s peak per core, 105.6 Gflop/s peak per node

    164 nodes x 12 cores = 1968 cores per each cluster

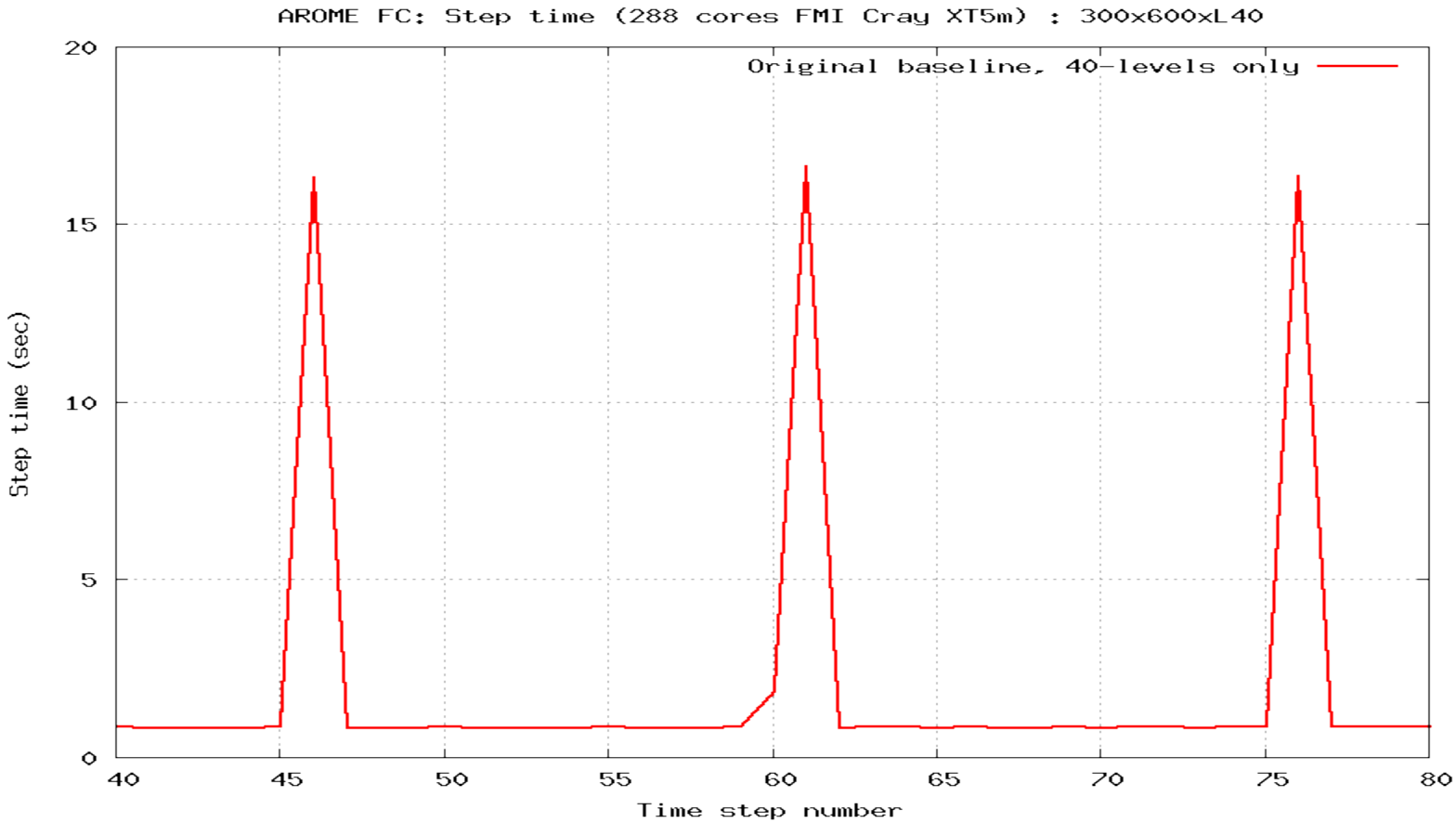
    16 GB shared memory per node (~1.3GB per core)
  - **2D-torus SeaStar-1 interconnection network**

About FMI's Cray XT5m (cont'd)

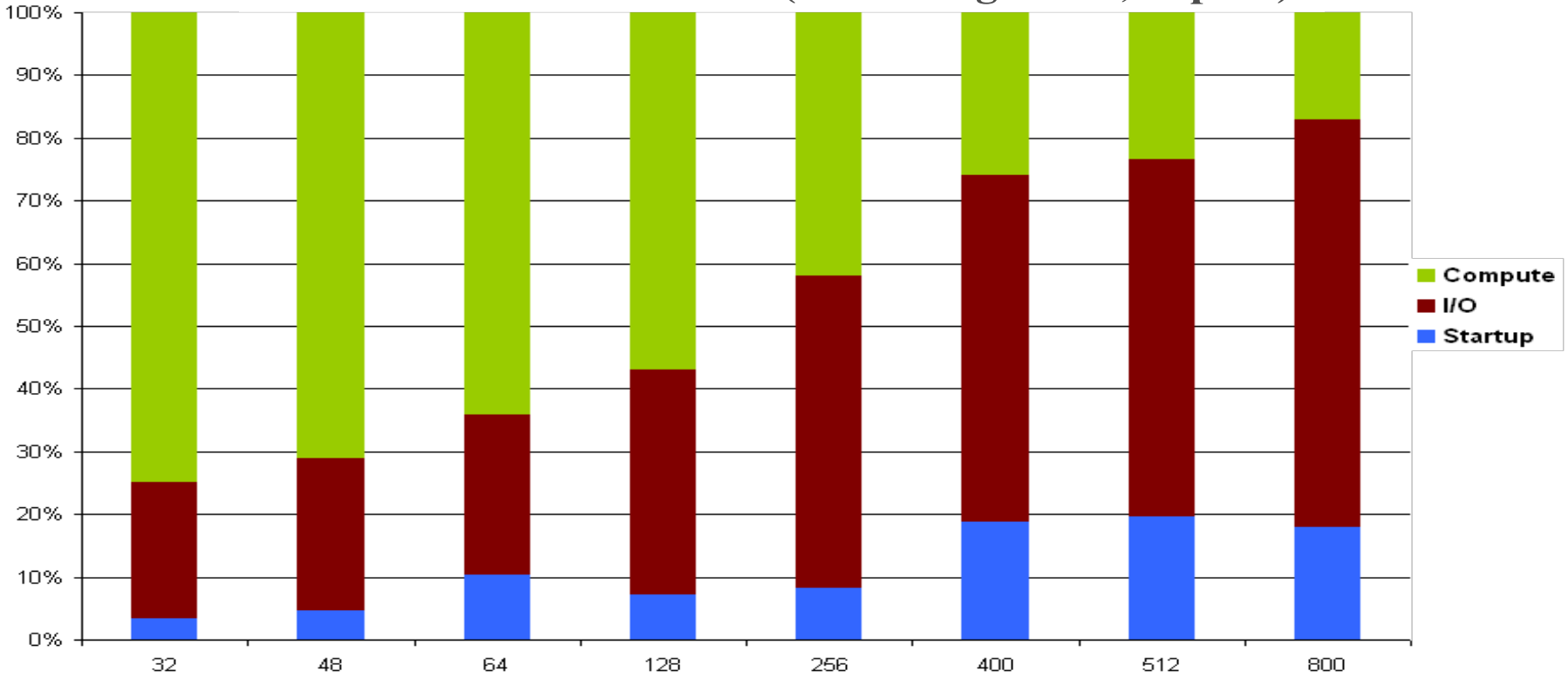


- **Local Lustre file-system on each cluster**
  - 2 X 60TB raw == 2 X 43TB formatted
- **1st cluster was installed in Sep'09, and 2nd late Oct'2009**
- **Performance acceptance completed on 4th of Nov'2009**

The fundamental performance problem in AROME



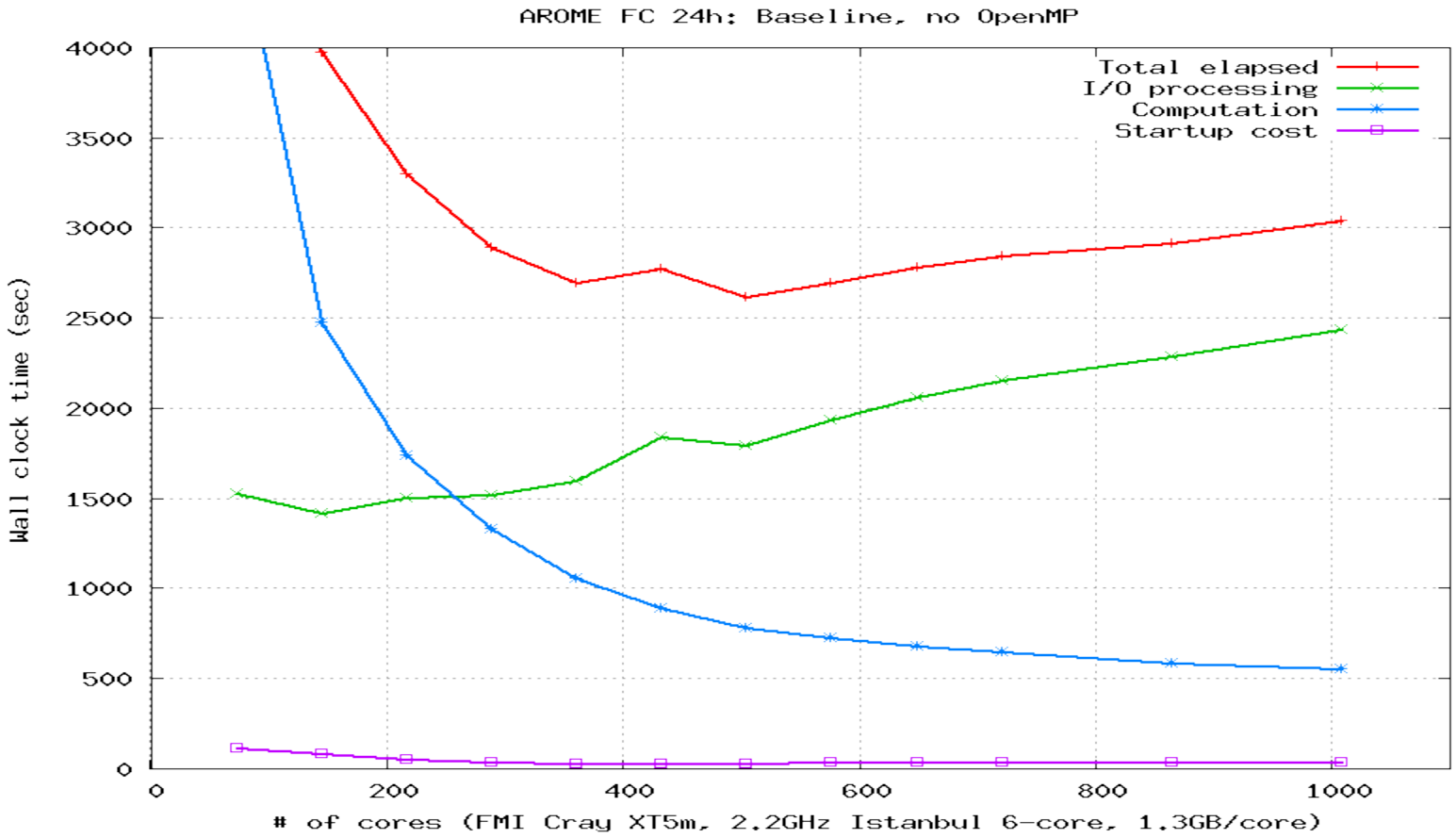
Relative time distribution (initial migration, Sep'09)



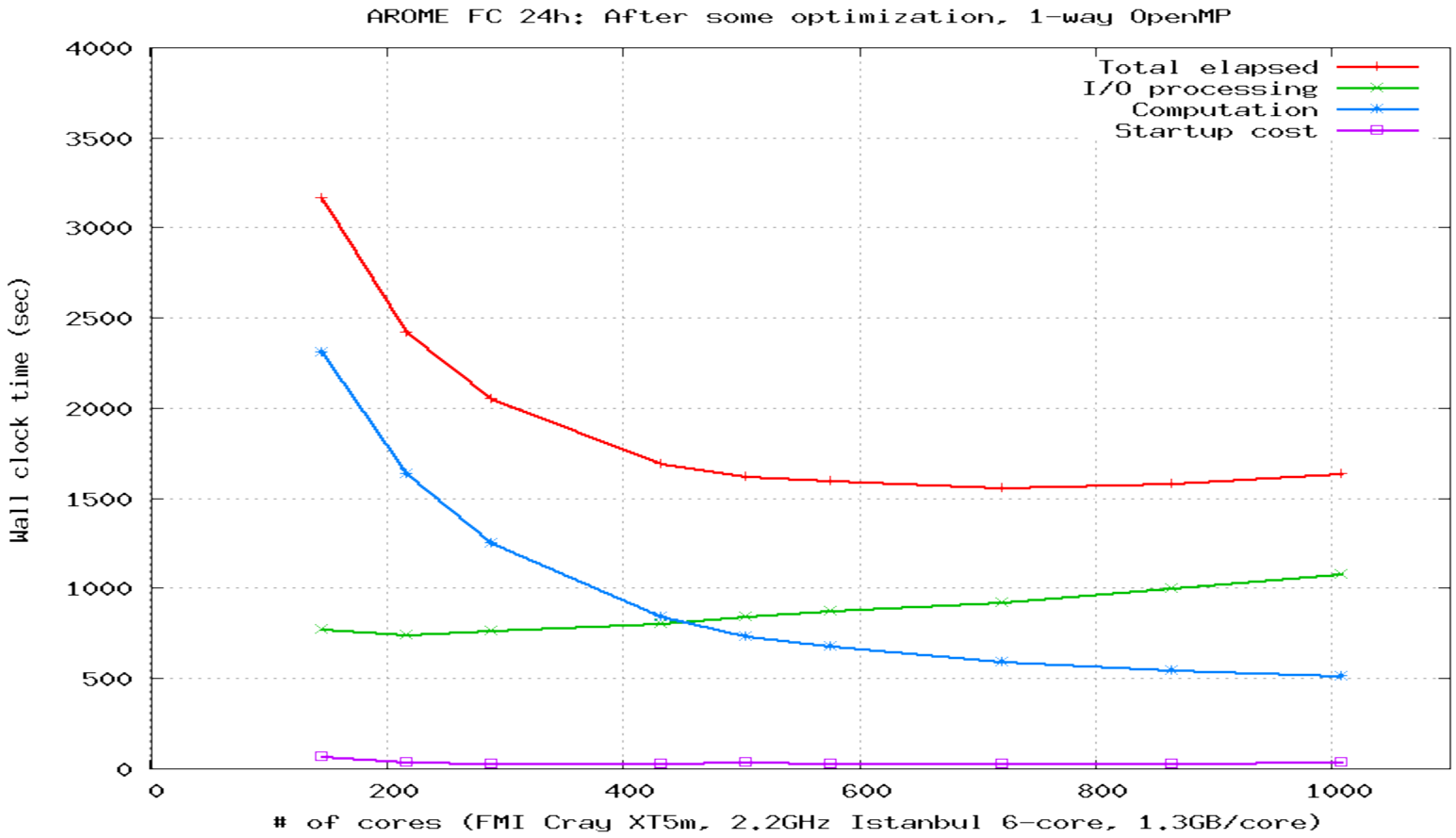
Number of Cores (MPI-only)

Courtesy & thanks to: Markku Kangas, FMI

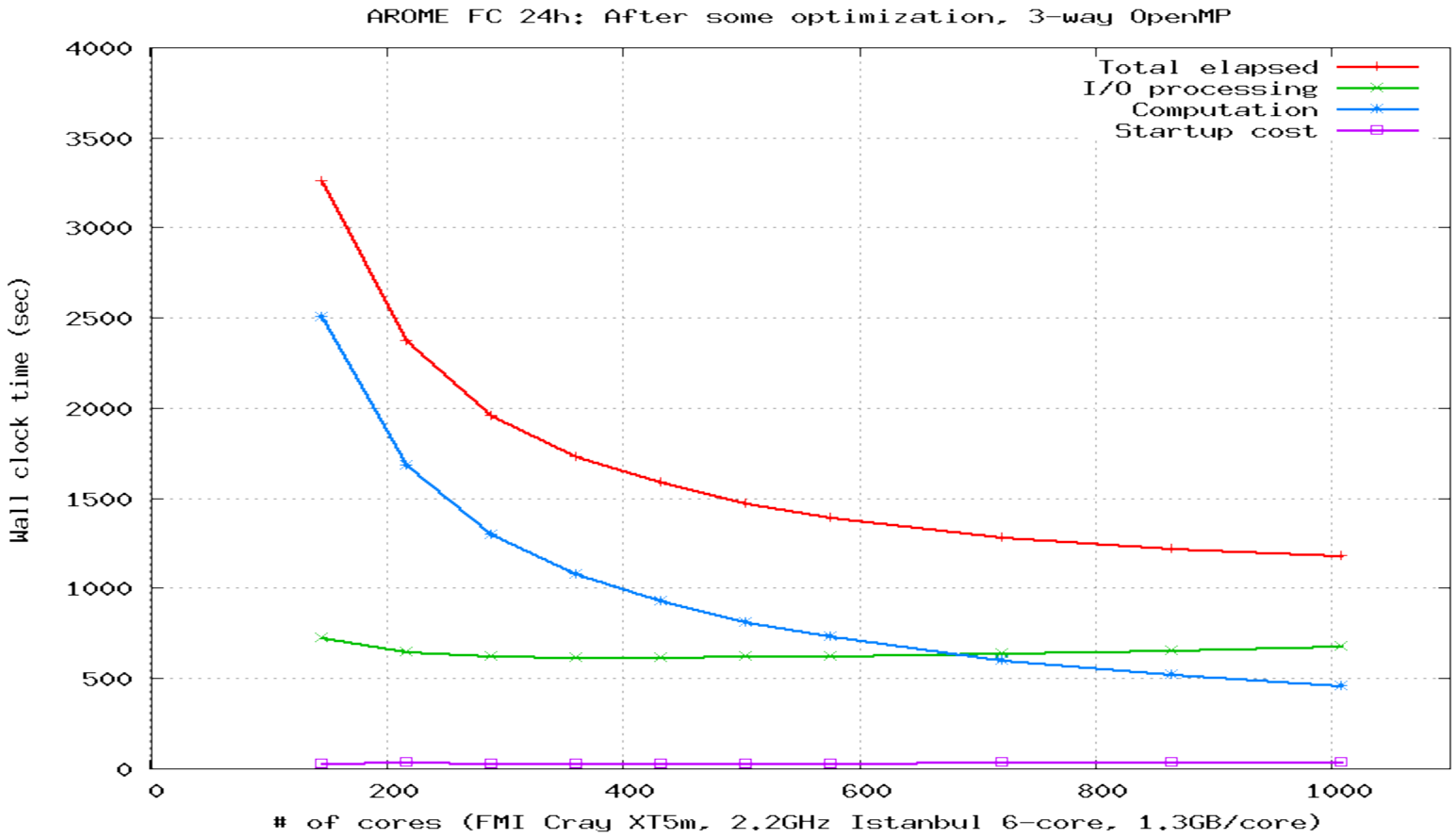
Total baseline time = Startup + Comp + I/O



After some optimization : 1-way OpenMP mode



*After some optimization : **3-way** OpenMP mode*



Status of AROME migration to Cray XT5m

- **AROME Forecast code**
  - Was *difficult* to run with > 64 cores due to msg flooding

    Using synchronized MPI_Ssend and later MPI_gatherv in diwrgrid*.F90
  - Now *works well* with MPI + OpenMP **> 1700** cores
  - Output interval 15min in 24h forecast : 53GB / 24h FC

    I/O processing costs are huge (data gathering, not the I/O itself as such)

Status of AROME migration …

- **AROME Forecast code** *(cont'd)*
  - Optimization by looking at the DrHook profiles

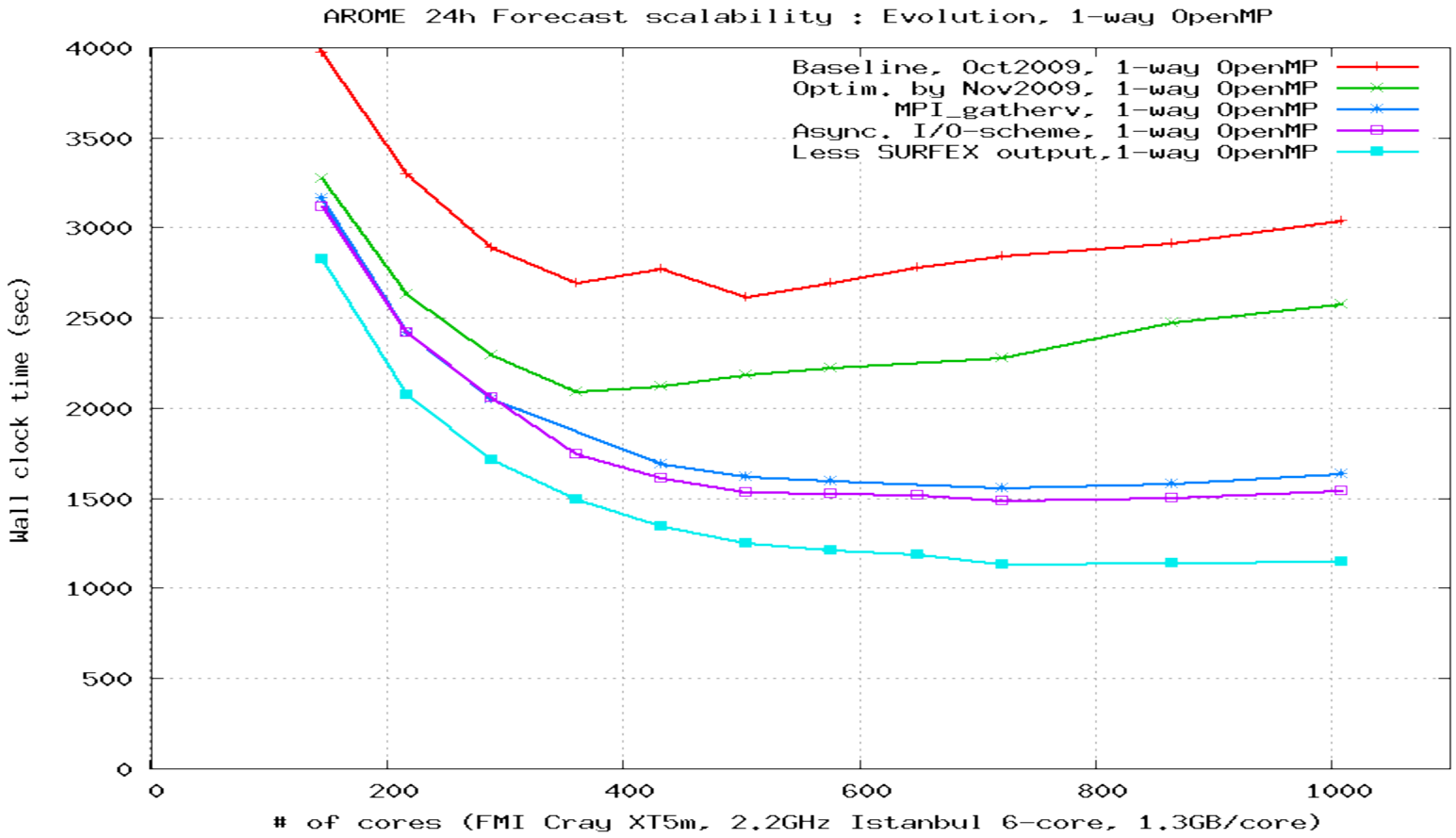
    Savings of 600MB of memory per MPI-task : message passing buffers

    Weak async I/O scheme (SAMIO) : Comp + I/O cores separated

    Increase output interval of SURFEX-data from 15min to 1h
- **This part was about 2 person months of work**
  - Now *runs comfortably* using 1000 cores with around 10X faster than the reference : **runtime under 15-20 minutes**

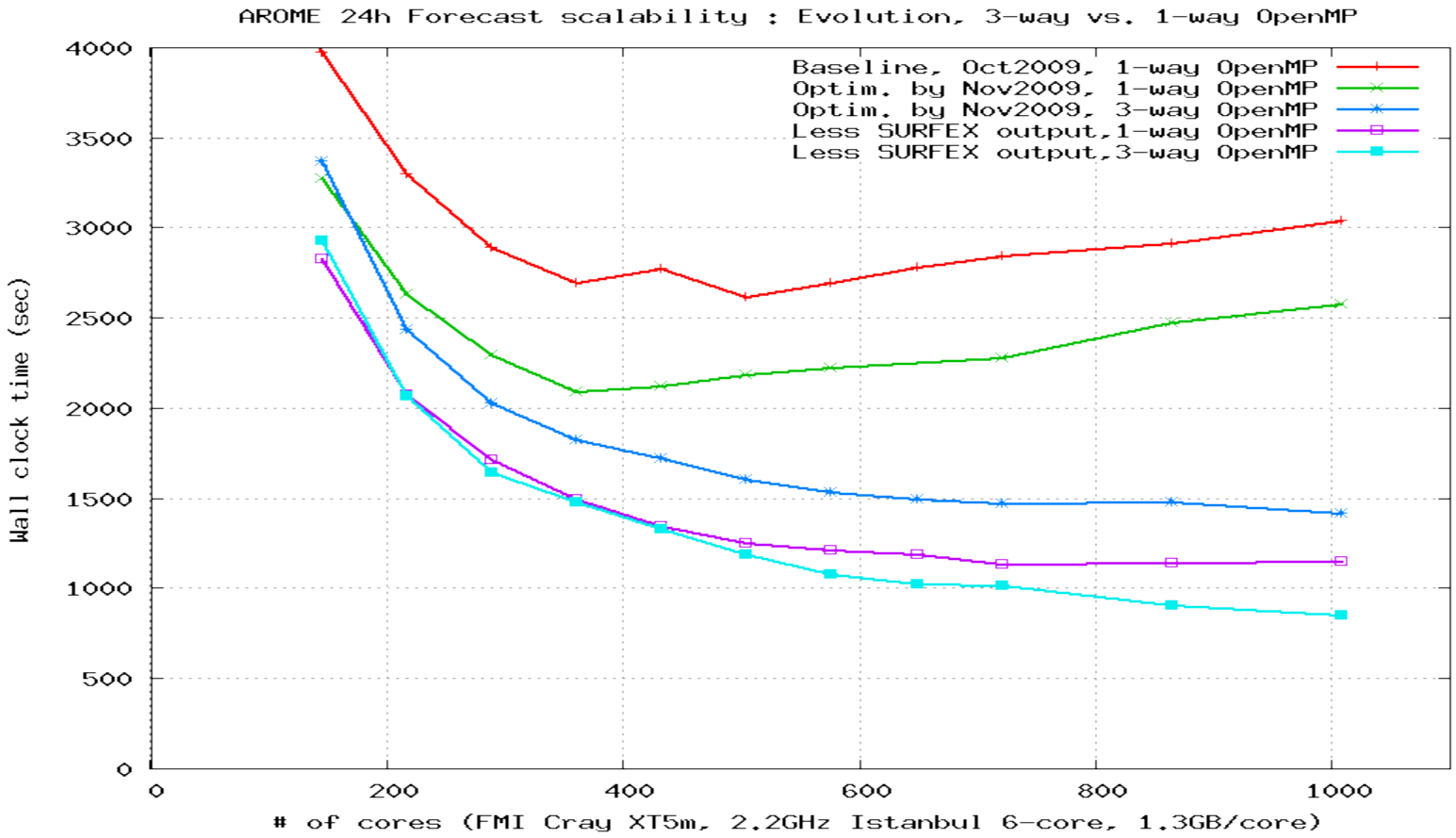
Ref.time : SGI Altix 72cores : 8820 sec

Model: 300 x 600 x L40

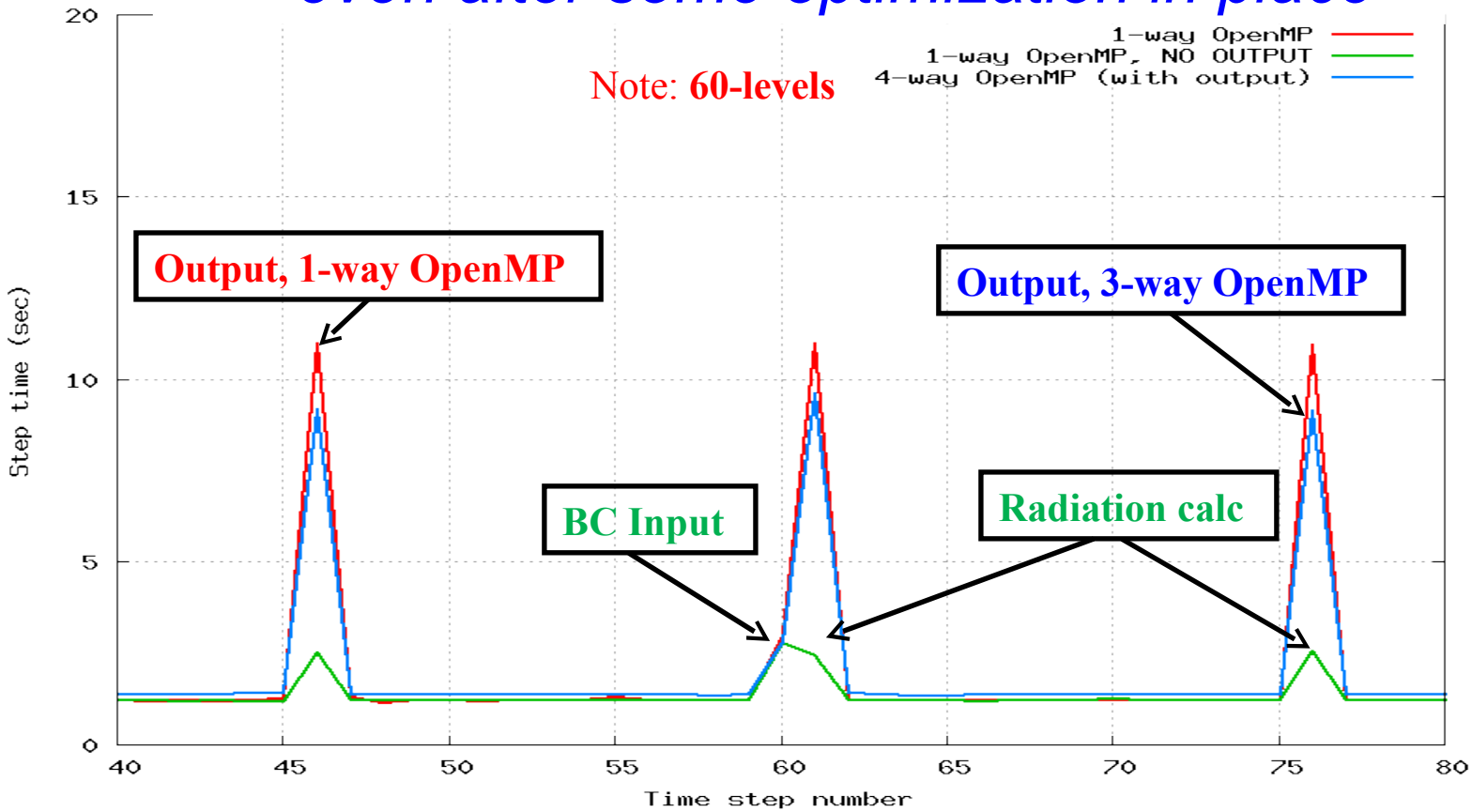


Ref.time : SGI Altix 72cores : 8820 sec

Model: 300 x 600 x L40



The performance problem still persists … even after some optimization in place



Issues requiring urgent attention

- **SURFEX & Atmospheric field I/O**
  - Coded *very inefficiently* , with parallel computers NOT in mind
  - Seriously limits scalability of the whole AROME run
  - The I/O logic (data distribution + I/O) needs to be rewritten
  - Gains ought to be comparable to Hirlam/HGS (40X speedup in I/O)

Issues requiring …

- **Startup cost can be excessive**
  - 200-300 secs for larger models
  - The same BIG files read by all MPI-tasks
  - NAMELIST-file processing is pretty slow :

    Should ”HIRLAM technique” be used ?

Issues requiring …

- **OpenMP shared memory parallelization part**
  - Current APL_AROME coding is a suboptimal use of OpenMP
  - Prevents exploitation of multicore-technologies

Issues requiring …

- **The last, but not least :**
  - AROME hardly works with other compilers than

    Pathscale, IBM XL and Intel ifort v11 (and of course on NEC)
  - GNU Fortran, Portland (PGI), Cray's new FTN still problematic
  - Are there ***genuine problems*** in the AROME code ?

    There seems to be lots of potentially *uninitialized* variables !!

    Cray X1E at AEMET finds 285 uninitialized variables !!!
  - Can be a ***show-stopper*** in supercomputer ITT & benchmarking

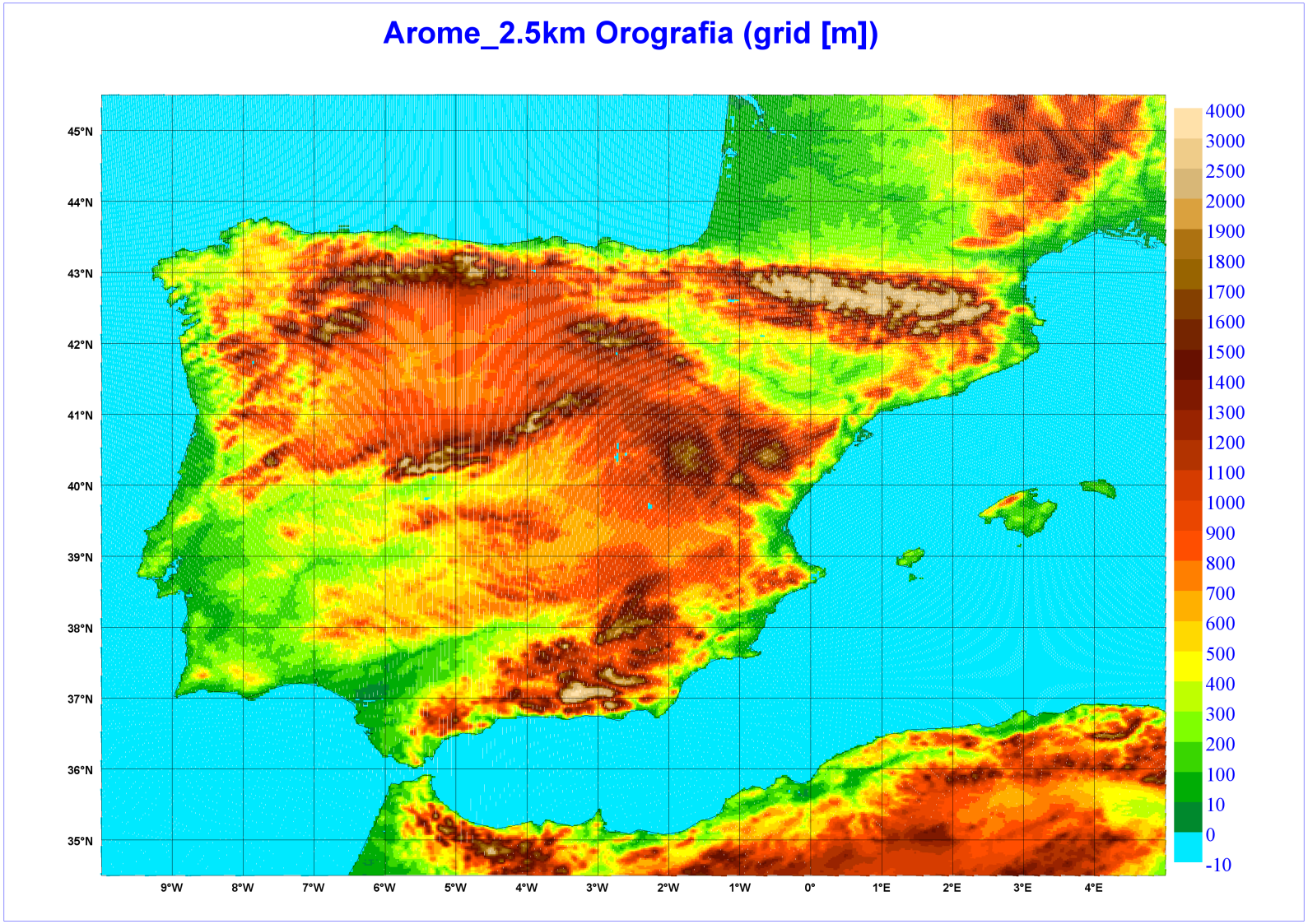
Good news for benchmarking

- **New gmake-based robust build mechanism : MAKEUP**
  - Standard gmake-based & simple to use build system for AROME
  - Also builds : Auxlibs (GRIBEX, EMOS), GL-tools, Monitor & Oulan
  - Handles ODB builds correctly & in a streamlined manner
  - Generates dummy-objects automagically
  - Tested so far at FMI, AEMET, SMHI, MetNo, DMI & ECMWF
  - One of the January/2010 Harmonie working week topics at FMI

Good news for benchmarking …

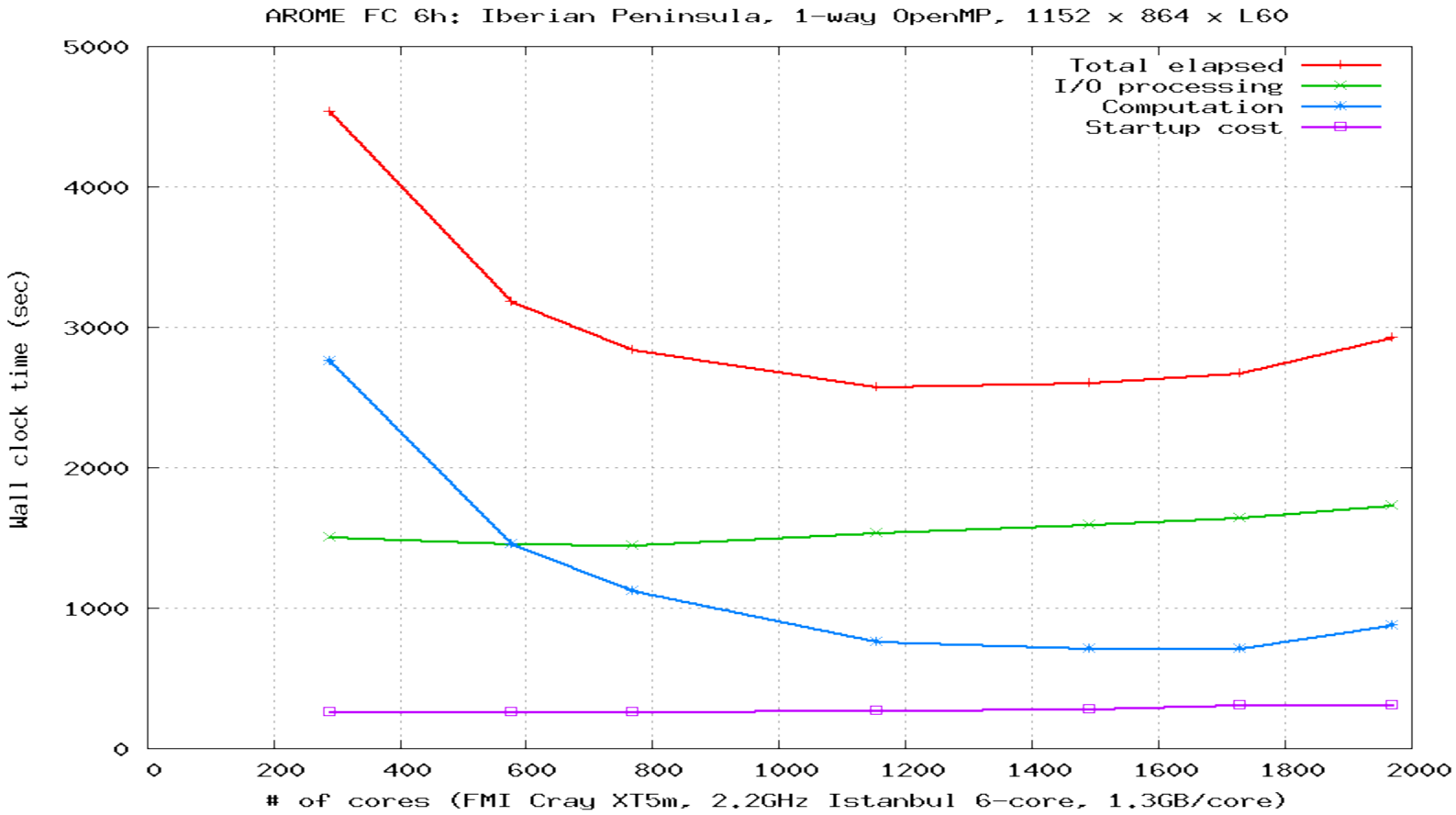
- **Traceback & performance profile extraction with DrHook**
  - Useful in debugging & finding locations for crashes
  - Produces wall-clock performance profiles cheaply, per MPI-task
  - Also supports OpenMP-threads
- **Weak asyncronous I/O by using SAMIO**
  - SAMIO stands for **Storage Access with Multiprocessing I/O**
  - A proper solution should be sought after in a collaboration with MF

Benchmark with Iberian peninsula (AEMET)

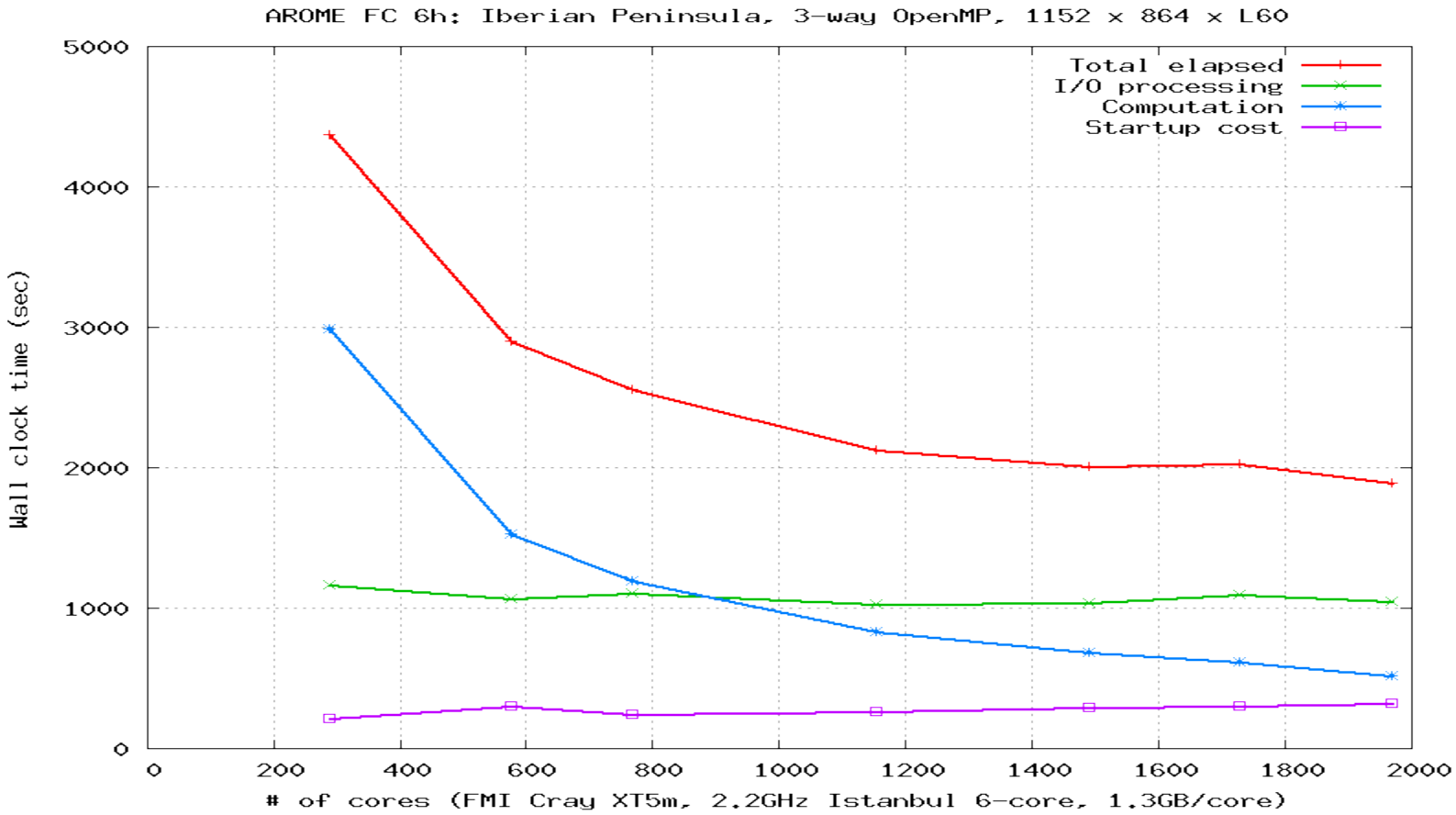


Courtesy & thanks to: Carlos Santos & Jose Antonio Garcia-Moya, AEMET

Iberian peninsula : ***1152 x 864 x L60***, *1-way OpenMP*, ***6h FC***



Iberian peninsula : ***1152 x 864 x L60***, *3-way* OpenMP, ***6h FC***



CSC Performance Monitor report example

Wall clock time in sec

Only 102 Mflop/s per core ca. 1% of peak

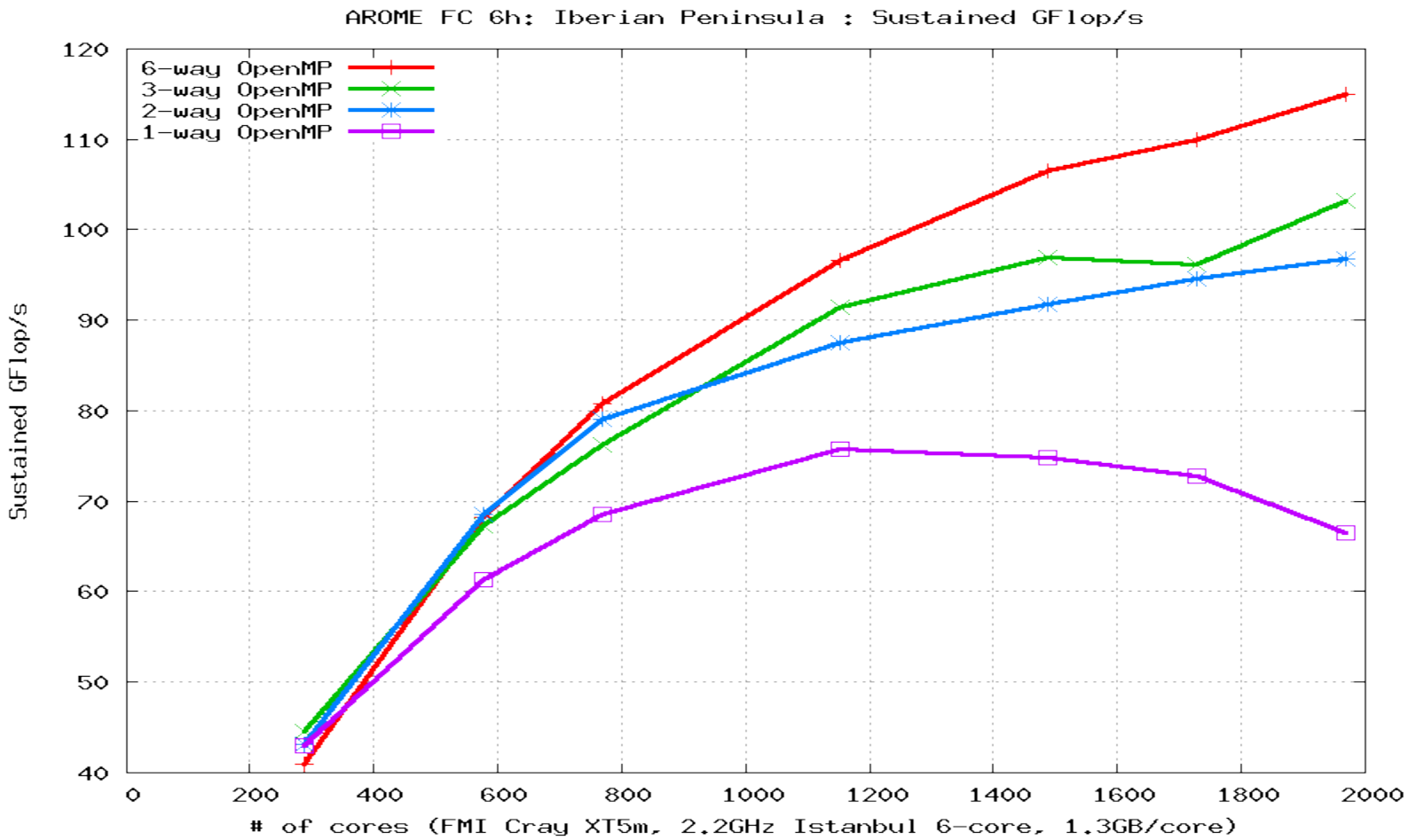
```
# MPI tasks    576
#                          aggregated     average     min(rank/val)   max(rank/val)
#       Real time (s)     1917415.789    3328.847    57   3328.787    0    3329.028
#    Process time (s)     1747722.650    3034.241     2   2925.400  132    3219.240
#     Flops (GFlop/s)          58.621       0.102   564      0.073  251       0.127
#    Flp-opers (10^9)      195141.505     338.787   564    241.635  251     422.195
#    L1 hit ratio (%)                      98.742   178     97.996    7      99.250
# Peak mem size (MB)                    2448.146   208   2416.768    0    2624.564
# Peak resident (MB)                     838.046   565    804.304    3    1123.104
```

195 Trillion floating point operations for 6h FC

Total Gflop/s rate over 576 tasks

Iberian peninsula 1152 x 864 x L60, 6h forecast
576 MPI-tasks , 1-way OpenMP

Iberian peninsula : ***Sustained Gflop/s, 6h FC***



Conclusions

- **AROME FC migration at FMI has demostrated that using hundreds of cores in a mixed MPI +OpenMP parallel environment is a feasible approach**
- **New MAKEUP build process helps significantly in**
  - **Program development**
  - **Standard benchmarking**

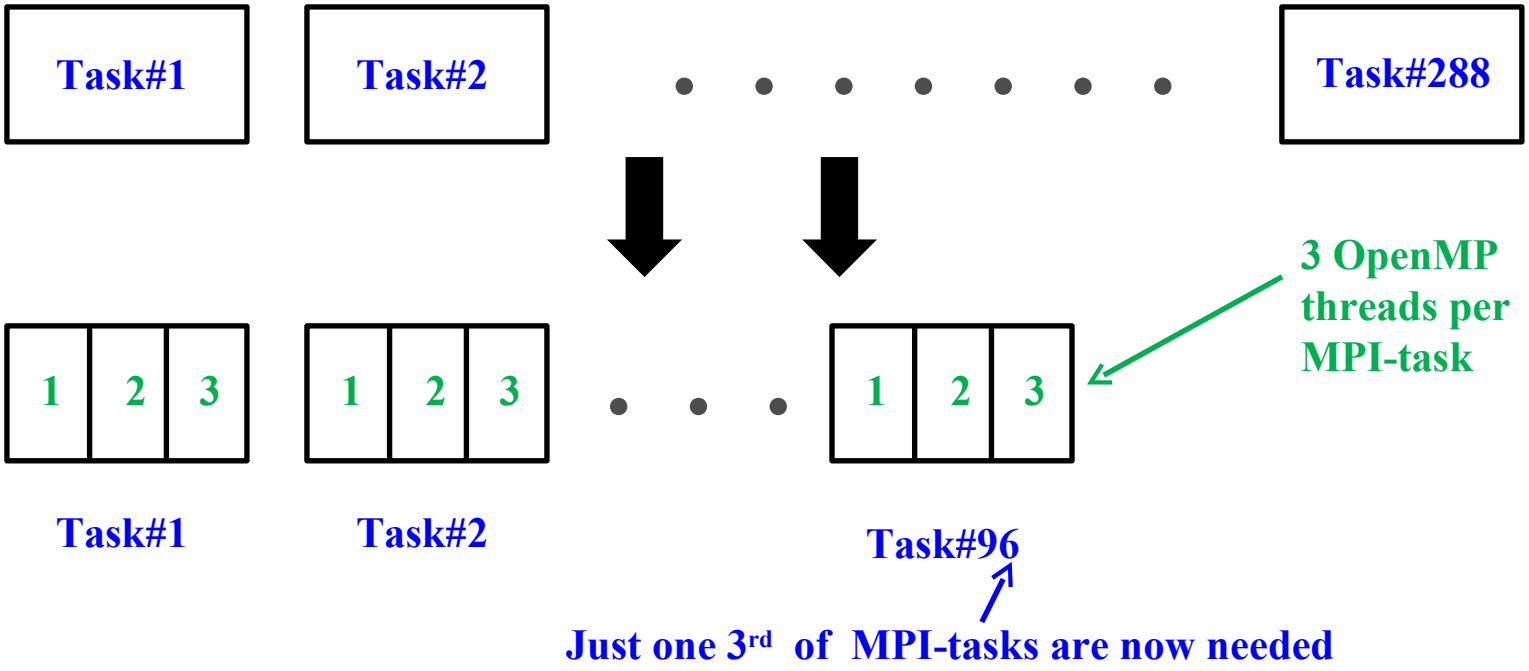
- **CSC wonders if AROME could be run further 5X faster ??!**
  - **If I/O problems were fixed**
  - **The causes for high startup cost were fixed**
  - **If OpenMP implementation in APL_AROME was fixed**

THANK YOU FOR YOUR ATTENTION !

Extra slides …

Using n-way OpenMP for AROME FC

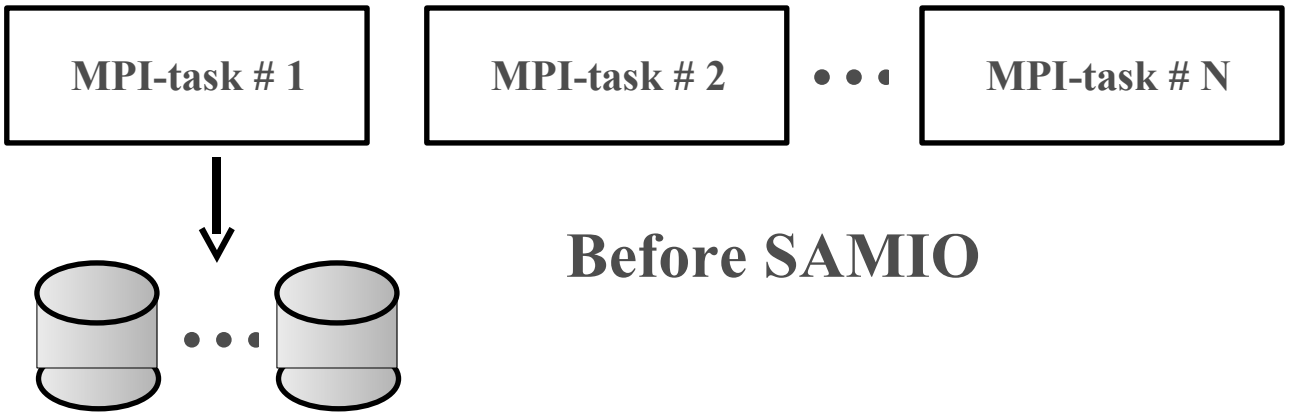
- On FMI's Cray XT5m we can use n-way OpenMP for each MPI-task (where n = 1,2,3,4,6 or 12 to keep full cores / node occupancy)
- Use of OpenMP is preferred and gives more bandwidth for AROME
- For example going from 1-way to 3-way OpenMP :



SAMIO

- **Storage Access with Multiprocessing I/O**
- **A "recipe" to replace Fortran I/O with asynchronous I/O with minimal code changes**
- **Actual I/O performed by separate MPI-tasks, which are not participating in computation at all**
- **Goals, principles, details …**
  - Let computation continue without waiting I/O to complete
  - I/O continues in the "background" by I/O-tasks
  - Upon next output requests currently free I/O-tasks take over
  - An I/O-task responsible for I/O on a given file
  - No MPI I/O, nor split files – just low level Unix I/O

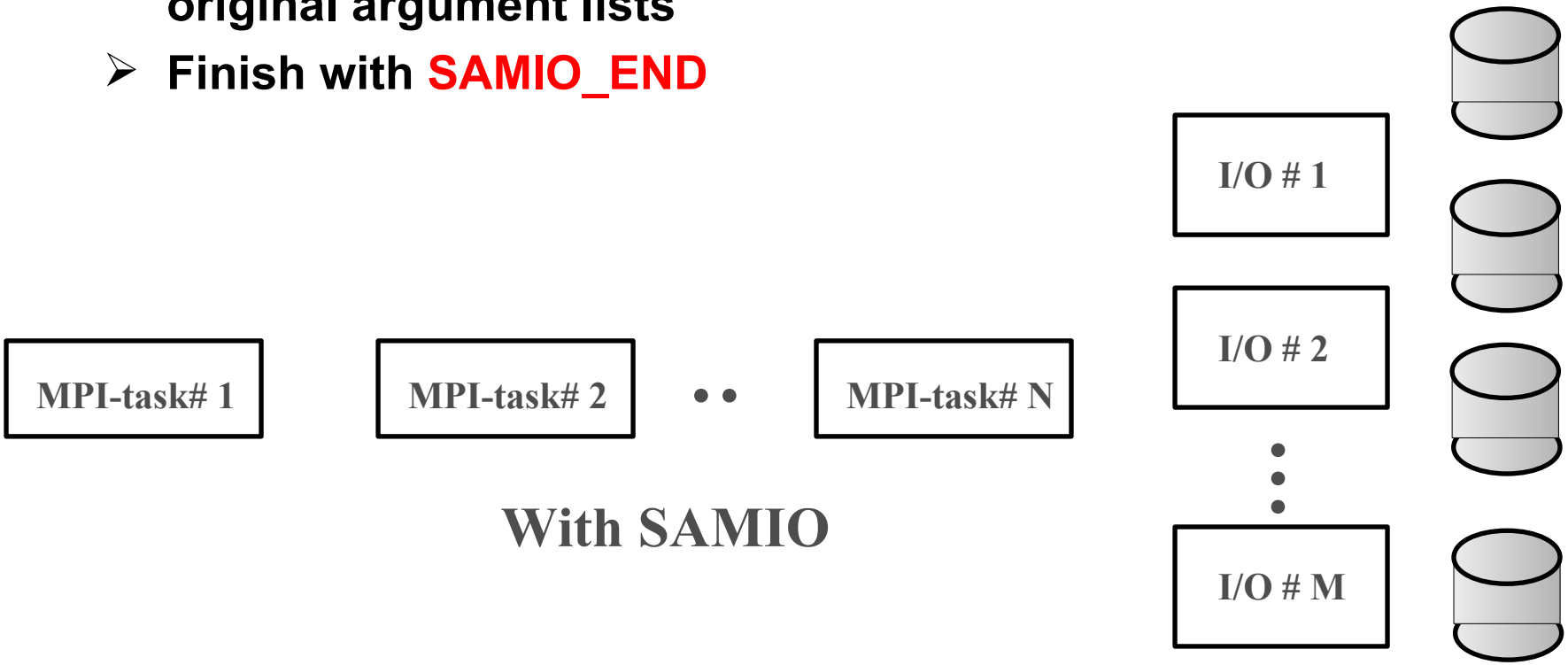
Schematic view of SAMIO in AROME



Before SAMIO

Schematic view of SAMIO in AROME

- **Initialize MPI with SAMIO_INIT**
- **Replace OPEN/READ/WRITE/CLOSE with SAMIO_OPEN/READ/WRITE/CLOSE using (near) original argument lists**
- **Finish with SAMIO_END**



With SAMIO

Some code fractions : OPEN-file

```
      SUBROUTINE OPEN_FILE(KNUMER, CDNOMF, CDSTAT, KRECL, ..)
       USE SAMIO_MOD
      IMPLICIT NONE
      ..
      LOGICAL LLSAMIO
      LLSAMIO = (SAMIO_MYPE == 1)
      IF (LLSAMIO) THEN
#ifdef USE_SAMIO
           CALL SAMIO_OPEN (UNIT=KNUMER,FILE=CDNOMF,STATUS=CDSTAT,
     &        ERR=902,FORM='UNFORMATTED',ACCESS='DIRECT',RECL=KRECL,
     &        IOSTAT=IREP)
           IF (IREP /= 0) GOTO 902
#endif
      ELSE
           OPEN (UNIT=KNUMER,FILE=CDNOMF,STATUS=CDSTAT,
     &        ERR=902,FORM='UNFORMATTED',ACCESS='DIRECT',RECL=KRECL,
     &        IOSTAT=IREP)
      ENDIF

      END SUBROUTINE OPEN_FILE
```

Some code fractions : WRITE data

```
      SUBROUTINE WRITE_CHAR_DATA(KNUMER, KREC, KREOP, CDTAB, ..)
       USE SAMIO_MOD
      IMPLICIT NONE
      ..
      CHARACTER CDTAB (JPNXNA*KFACTM)*(JPNCPN)
      LOGICAL LLSAMIO
      LLSAMIO = SAMIO_HAS_OPENED(KNUMER)
      IF (LLSAMIO) THEN
#ifdef USE_SAMIO
         CALL SAMIO_WRITE
(UNIT=KNUMER,REC=KREC,ERR=901,IOSTAT=KREP,
     &        ARRAY=CDTAB)
         IF (KREP /= 0) GOTO 901
#endif
      ELSE
         WRITE (UNIT=KNUMER,REC=KREC,ERR=901,IOSTAT=KREP) CDTAB
      ENDIF
      ..
      END SUBROUTINE WRITE_CHAR_DATA
```

What is going on under the hood of SAMIO ?

- **The next free I/O-task is chosen during the SAMIO_OPEN and file credentials are sent to that task**
- **Each SAMIO_WRITE buffers data locally until some user defined buffer size (e.g. 25MB) has been reached**
- **When cached buffers are full, they are sent to I/O-task using non-blocking (asynchronous) MPI_isend**
- **SAMIO_READ are processed immediately by requesting read and issuing MPI_recv from I/O-task …**
- **… unless direct access file, in which case this data is available locally – in AROME code at least**
- **Only SAMIO_CLOSE enforces disk I/O before which an optional (say) big-endian conversion is performed first**

Why SAMIO is not the final solution ?

- **There are still two memory copies that cost**
  - **SAMIO_WRITE** buffering to local cache
  - Local cache is **MPI_Isend** to the remote I/O-task
- **Compared to Unix/Linux write caching, there is not much progress – in fact !**
- **What ought to be done is to send data directly to the I/O-processes without intermediary (the 1st computational task currently) and let I/O-tasks to decide how and when and by whom to do the I/O**
- **Thus there is a need to parallelize properly the data collection steps in AROME I/O, not only the I/O itself**