



Collaborative Research into Exascale
Systemware, Tools and Applications

George.Mozdzynski@ecmwf.int

Acknowledgements

Mats Hamrud

ECMWF

Nils Wedi

ECMWF

Jens Doleschal

Technische Universität Dresden

Harvey Richardson

Cray UK

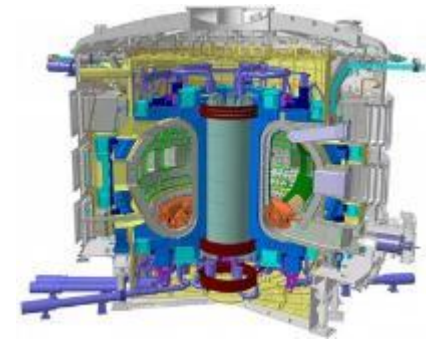
And my other partners in the CRESTA Project

The CRESTA project has received funding from the EU Seventh Framework Programme (ICT-2011.9.13)

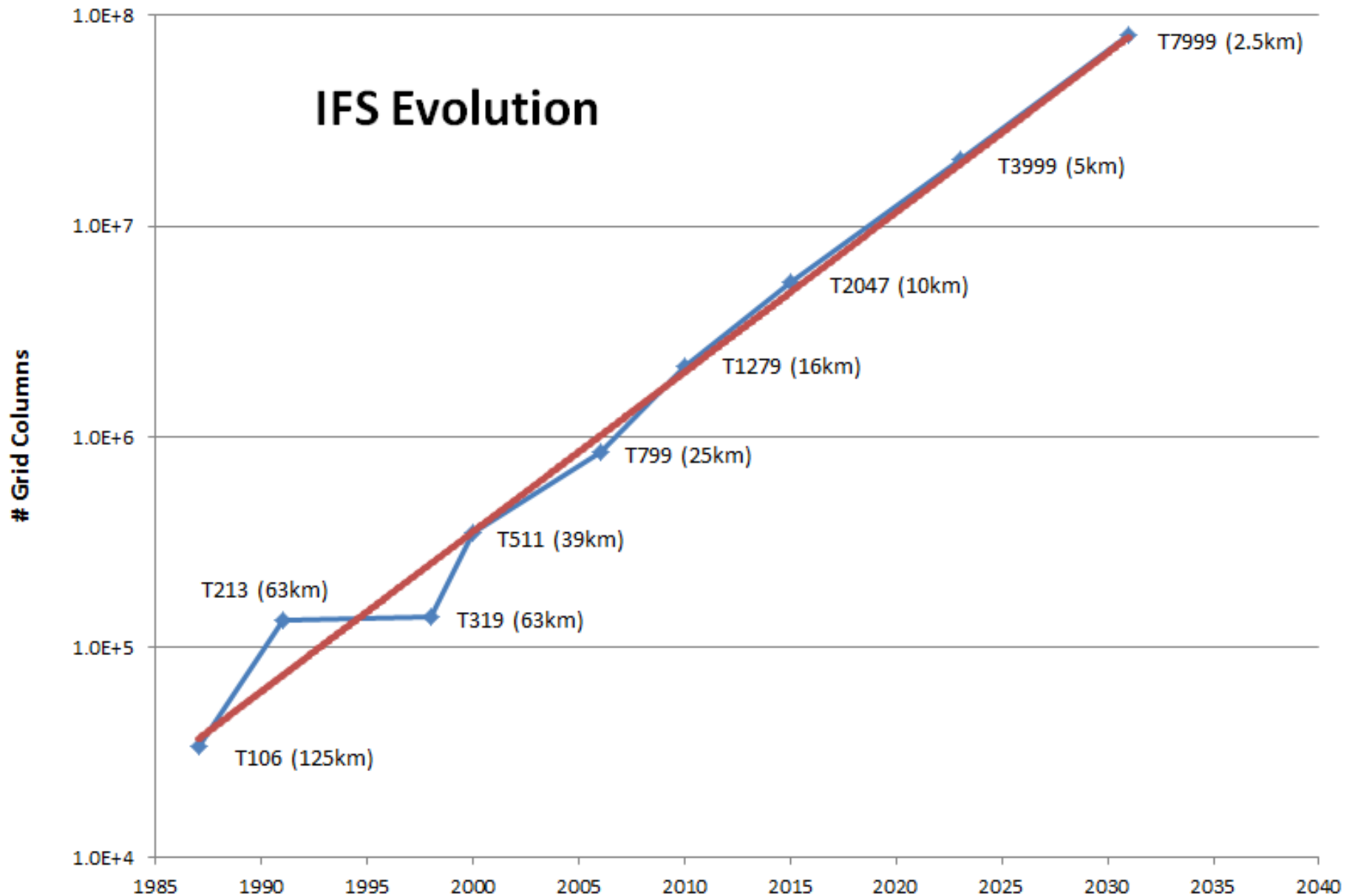


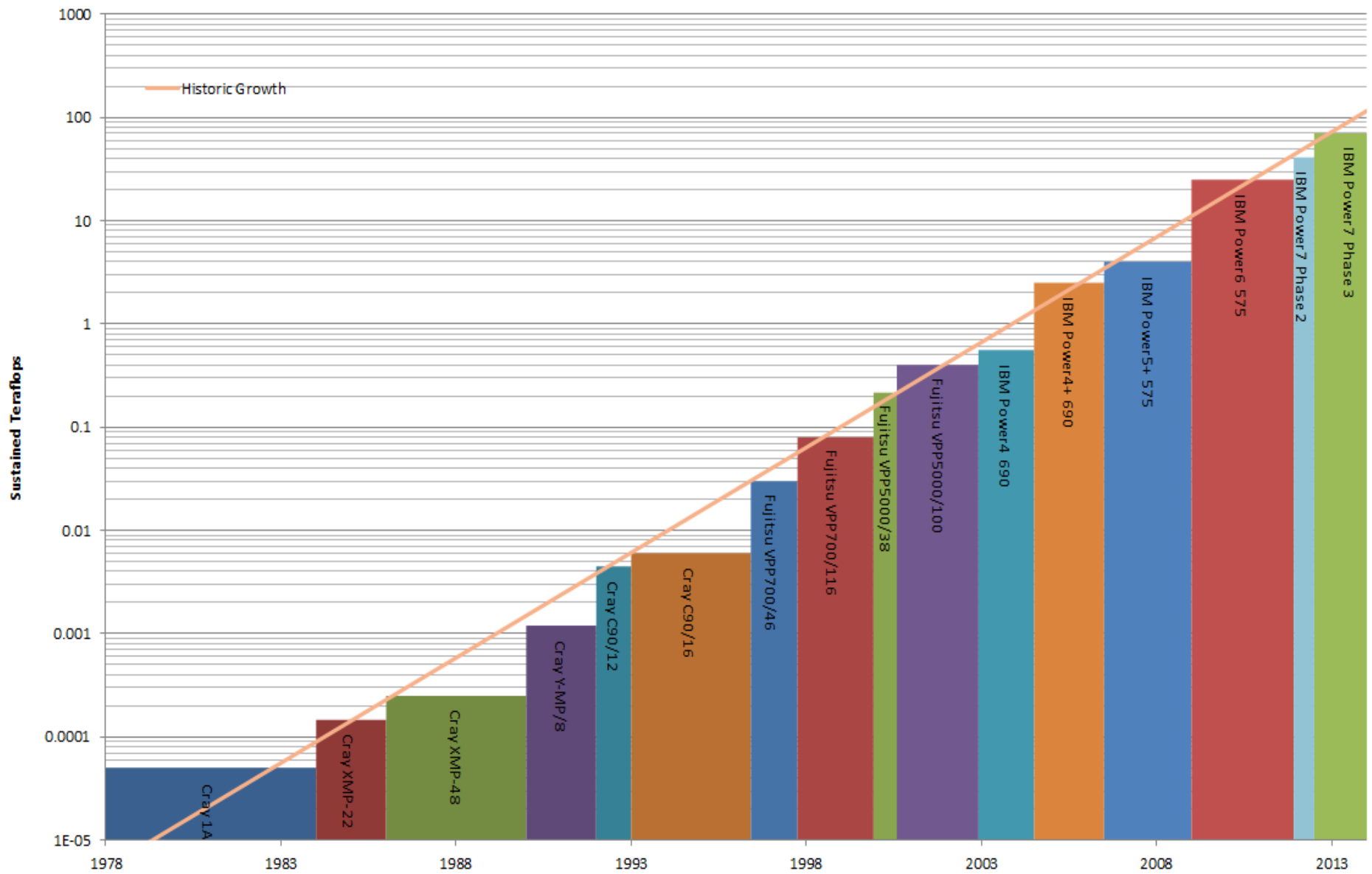
What is CRESTA - see <http://cresta-project.eu/>

- Collaborative Research into Exascale Systemware, Tools and Applications
- EU funded project, 3 years (started Oct 2011), ~ 50 scientists
- Six co-design vehicles (aka applications)
 - ELMFIRE (CSC, ABO, UEDIN) - fusion plasma
 - GROMACS (KTH) - molecular dynamics
 - HEMELB (UCL) - biomedical
 - IFS (ECMWF) - weather
 - NEK5000 (KTH) & OPENFOAM (USTUTT, UEDIN) - comp. fluid dynamics
- Two tool suppliers
 - ALLINEA (ddt : debugger) & TUD (vampir : performance analysis)
- Technology and system supplier – CRAY UK
- Many Others (mostly universities)
 - ABO, CRSA, CSC, DLR, JYU, KTH, UCL, UEDIN-EPCC, USTUTT-HRLS

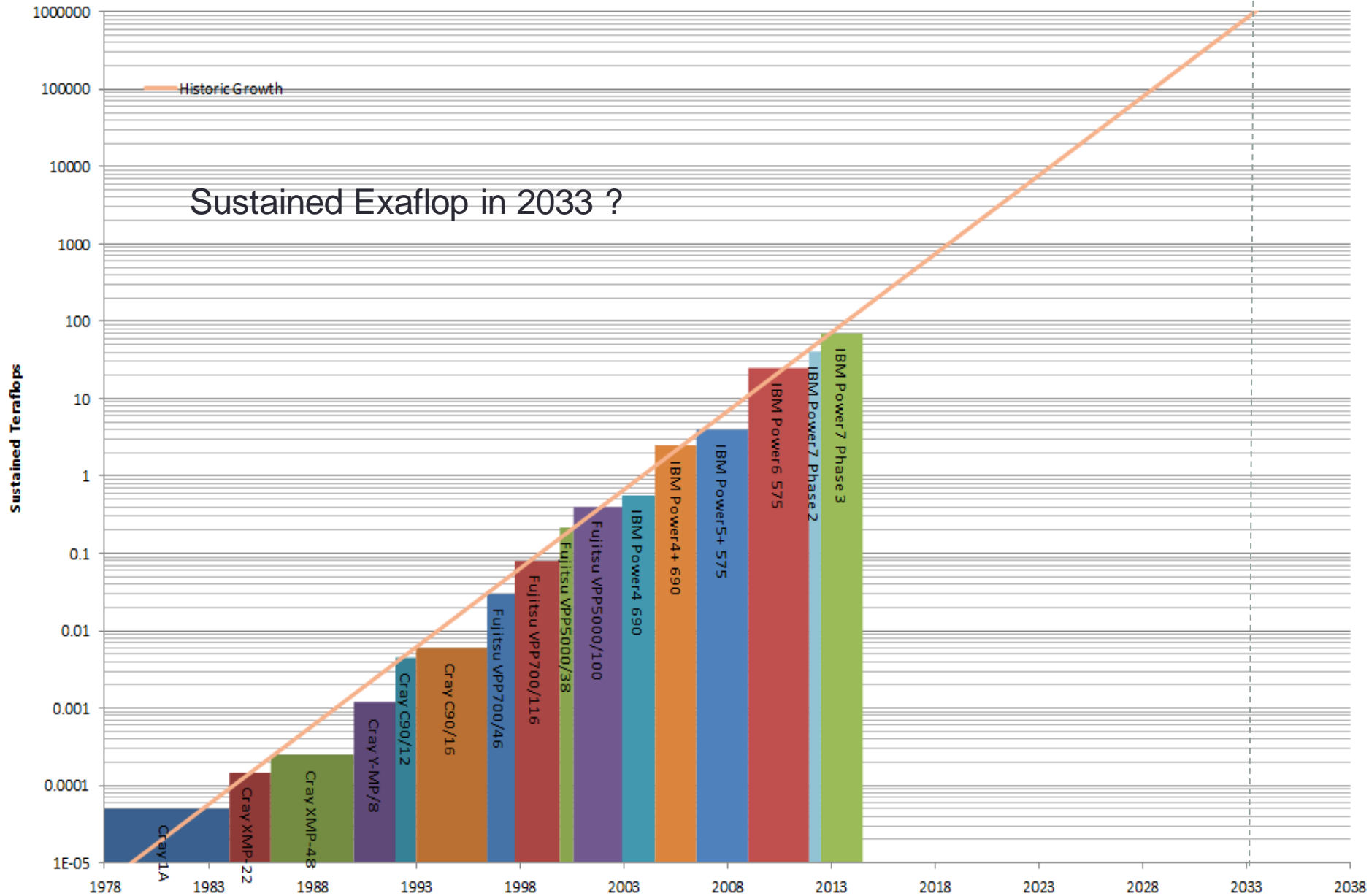


IFS Evolution





Sustained Exaflop in 2033 ?



IFS model: current and future model resolutions

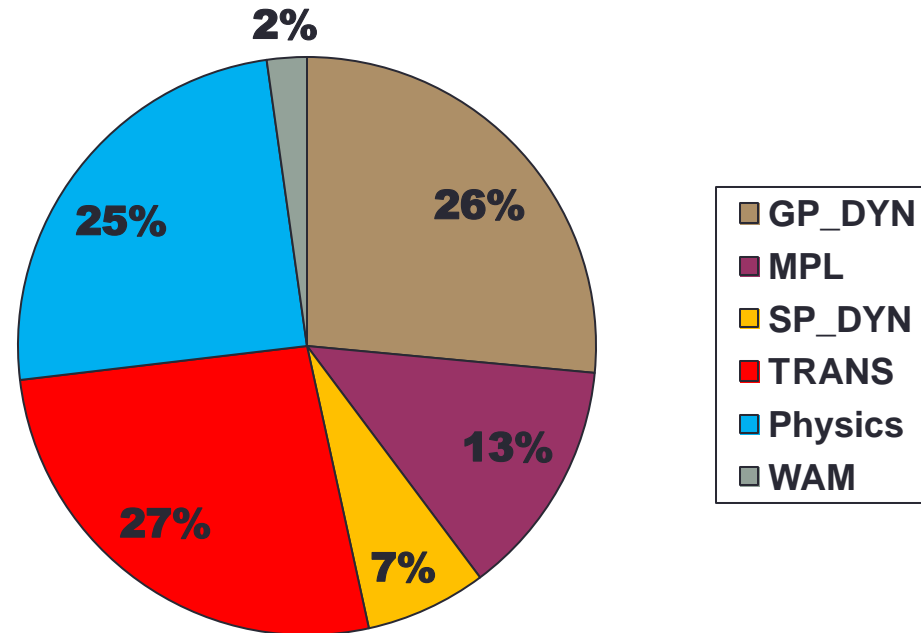
IFS model resolution	Envisaged Operational Implementation	Grid point spacing (km)	Time-step (seconds)	Estimated number of cores ¹
T1279 H²	2010 (L91)	16	600	1100
	2013 (L137)			1600
T2047 H	2014-2015	10	450	6K
T3999 NH³	2023-2024	5	240	80K
T7999 NH	2031-2032	2.5	30-120	1-4M

1 – a gross estimate for the number of 'IBM Power7' equivalent cores needed to achieve a 10 day model forecast in under 1 hour (~240 FD/D), system size would normally be ~10 times this number.

2 – Hydrostatic Dynamics

3 – Non-Hydrostatic Dynamics

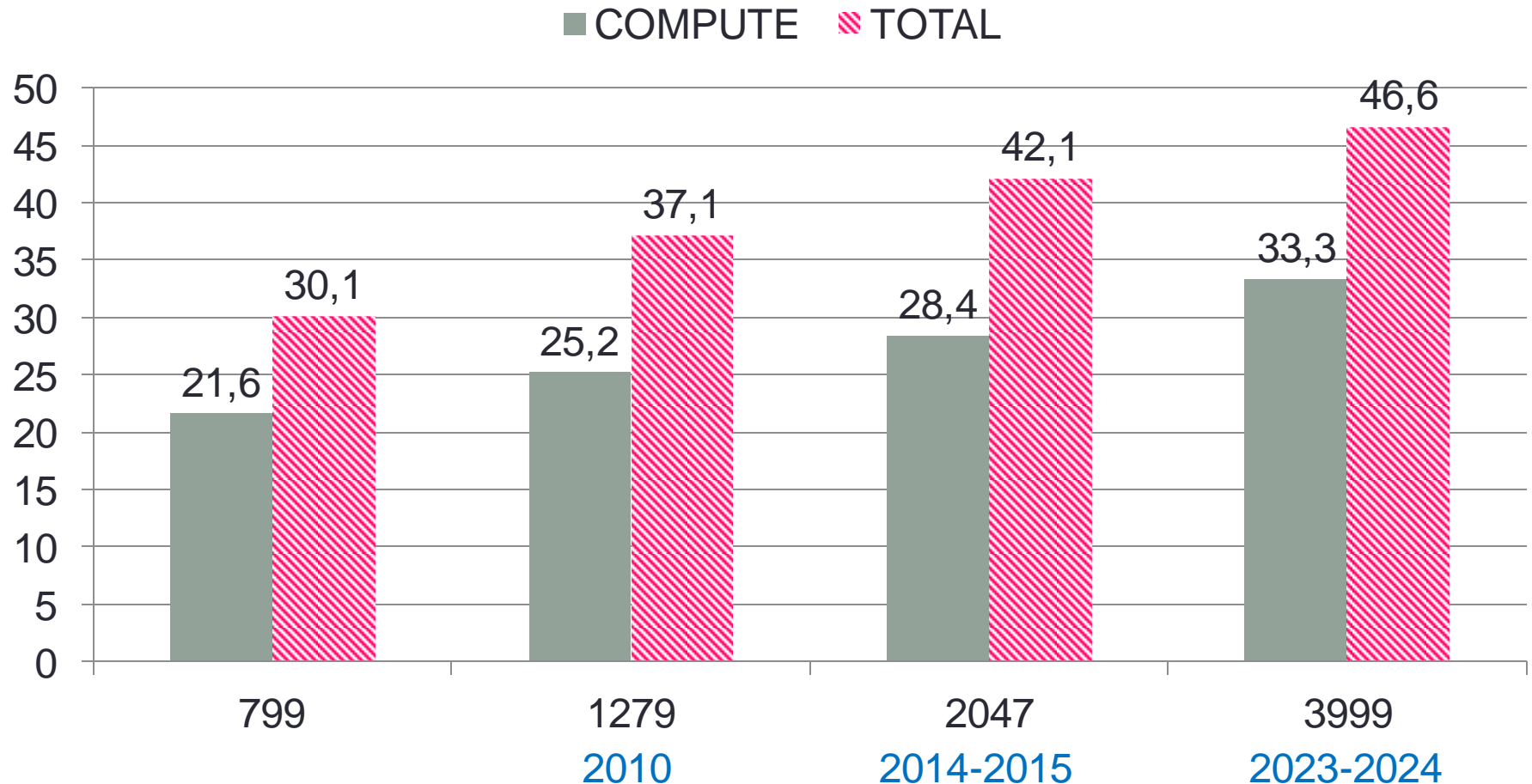
NH IFS T_L3999 L91 (5 km) on IBM Power7 with FLT



TSTEP=180s, 3.1s/iteration
Using 1024 tasks x16 OpenMP threads
10 day forecast ~ 4 hours for this config

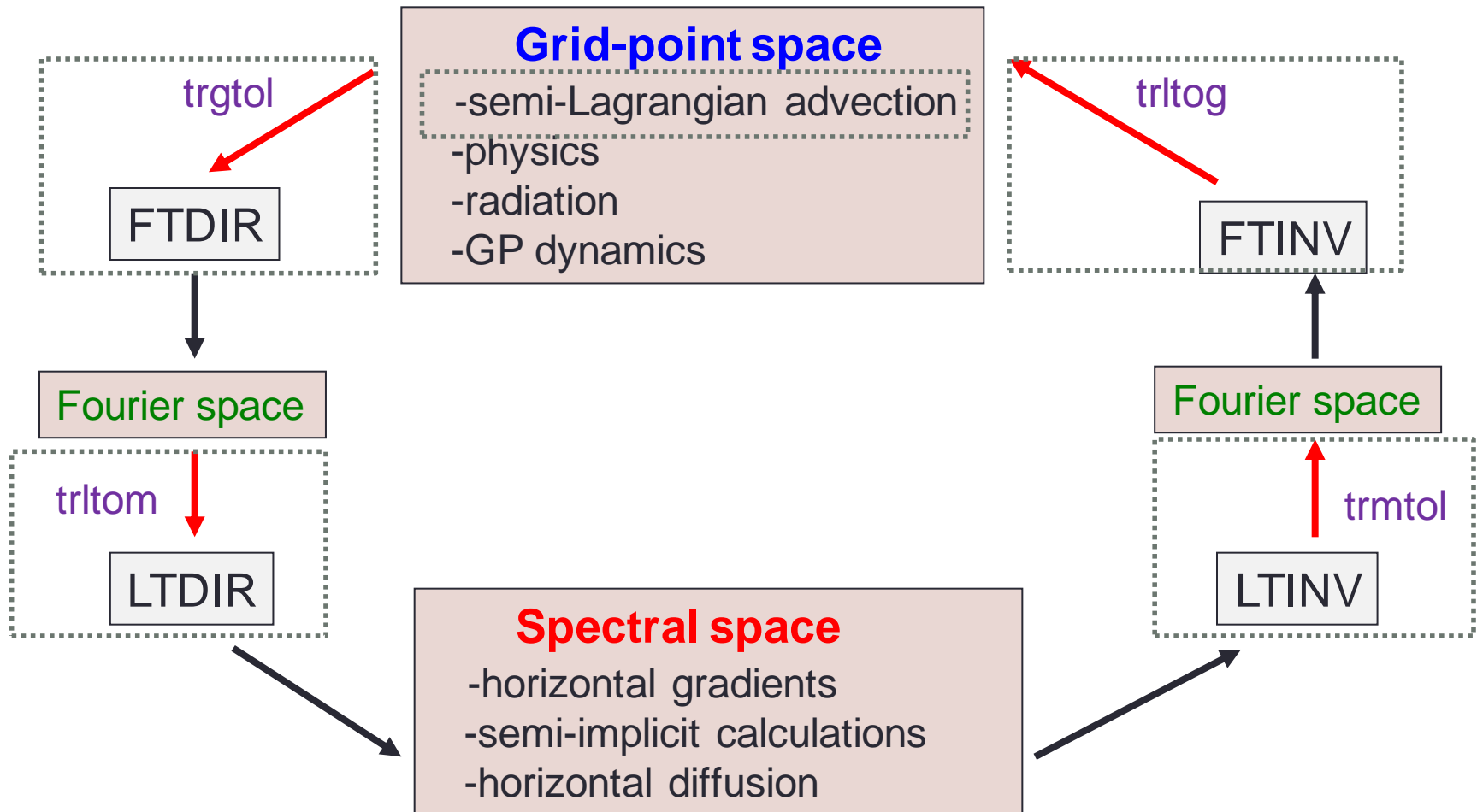
SP_DYN was 23 percent for this model configuration, and is now 7 percent. Improvement due to exposing 'greater OpenMP parallelism' from 4K threads to a maximum of 4K * 91 threads ; in this case 16K threads.

% cost of Spectral Transforms on IBM Power7 (all L91, all NH for comparison)



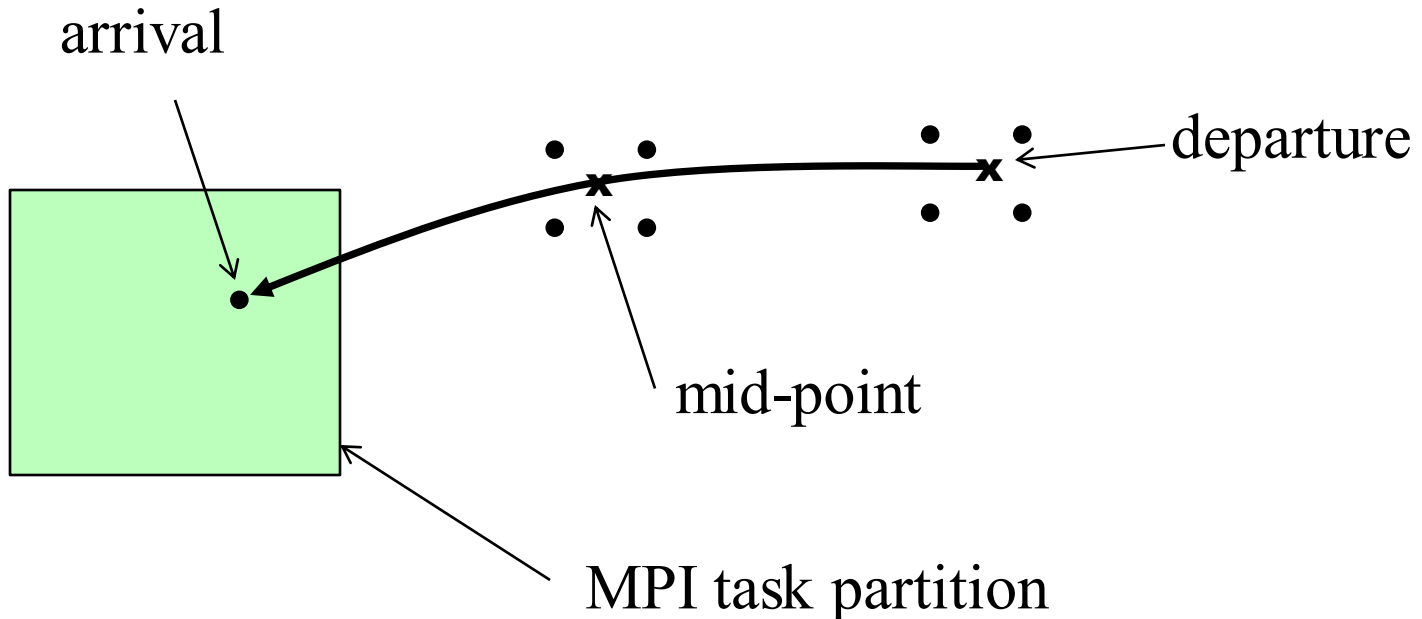
Expect significant reductions in future cores -> vector instr. / GPU

IFS optimizations for [Tera,Peta,Exa]scale

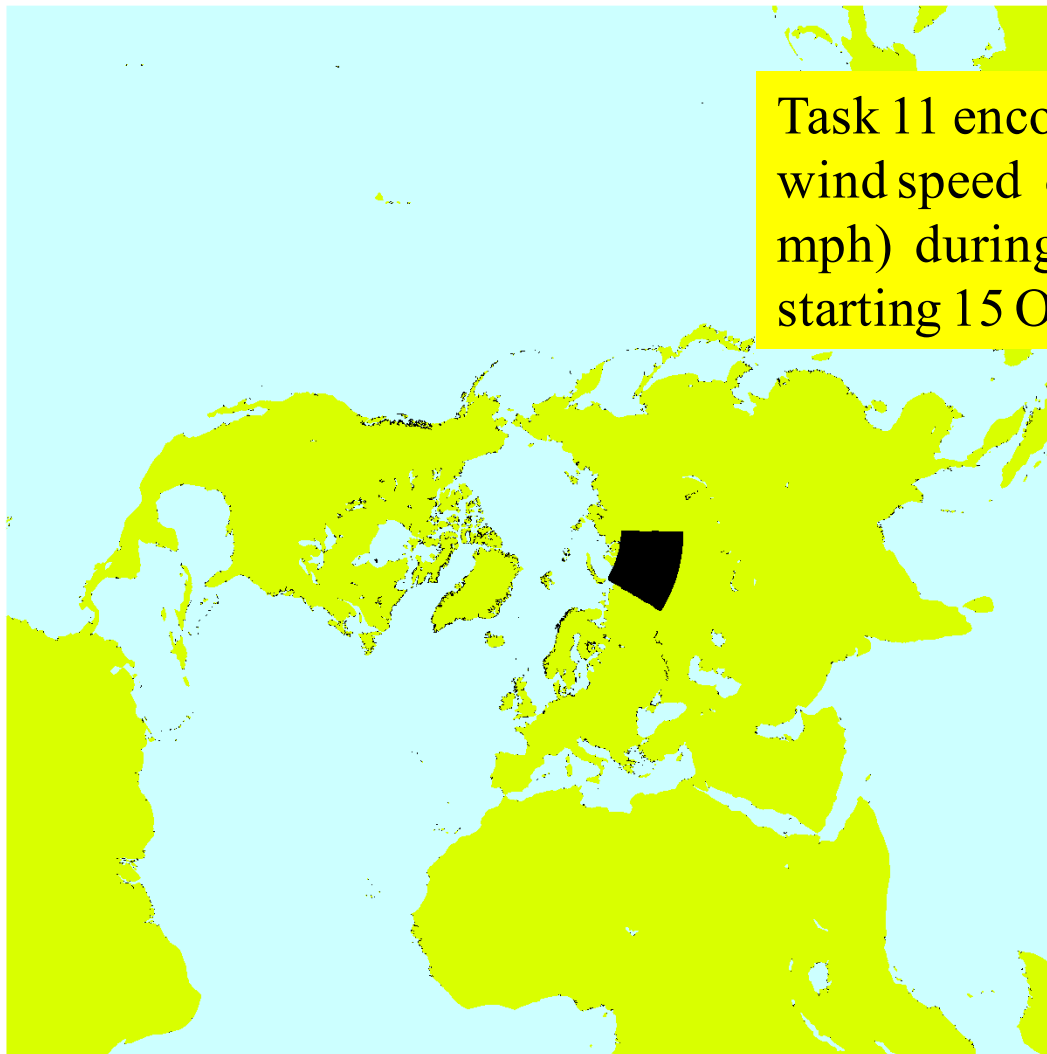


Semi-Lagrangian Transport

- Computation of a trajectory from each grid-point backwards in time, and
- Interpolation of various quantities at the departure and at the mid-point of the trajectory

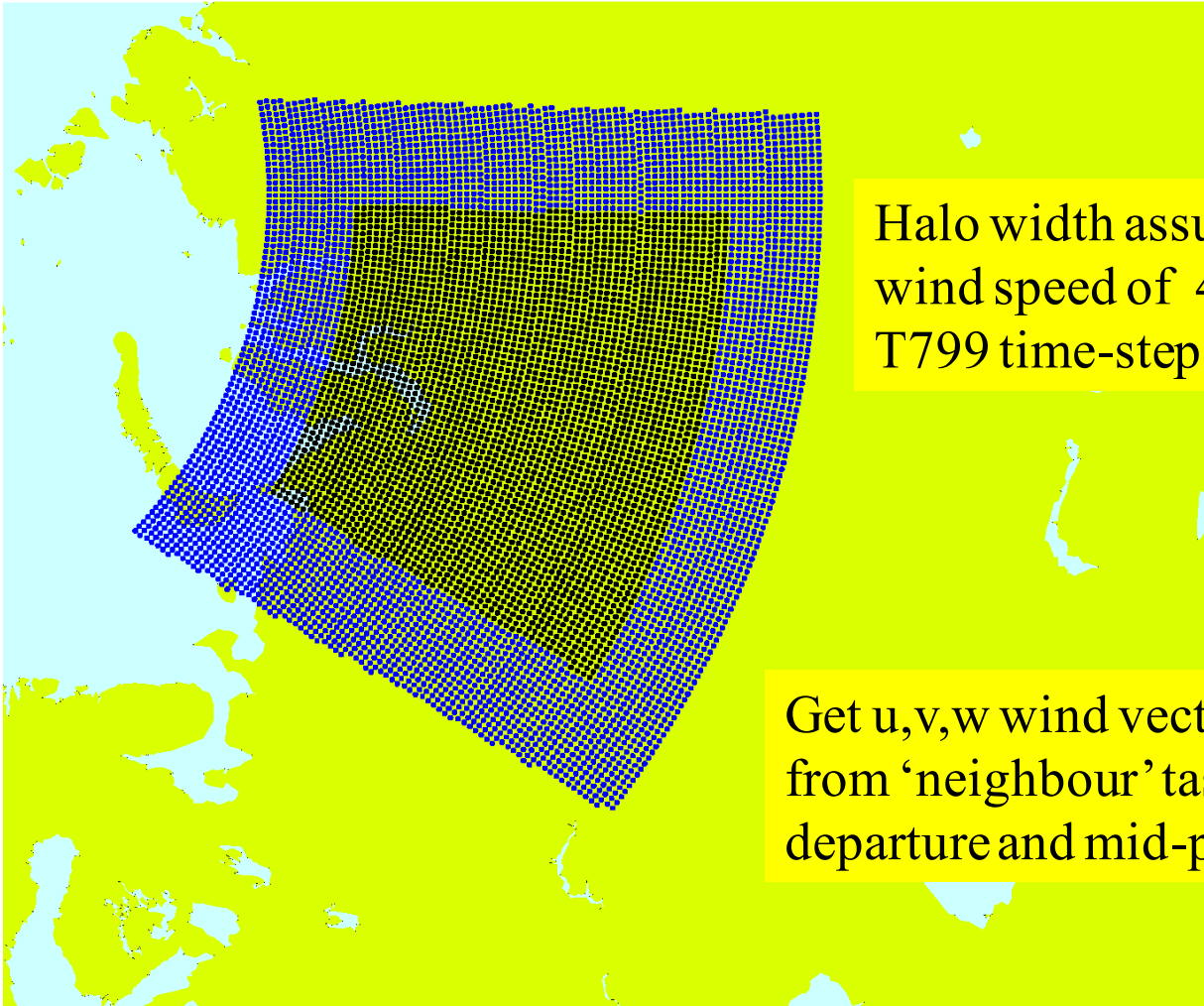


Semi-Lagrangian Transport: T799 model, 256 tasks



Task 11 encountered the highest wind speed of 120 m/s (268 mph) during a 10 day forecast starting 15 Oct 2004

blue: halo area



Halo width assumes a maximum wind speed of $400 \text{ m/s} \times 720 \text{ s}$ T799 time-step (288 km)

Get u, v, w wind vector variables (3) from 'neighbour' tasks to determine departure and mid-point of trajectory

red: halo points actually used

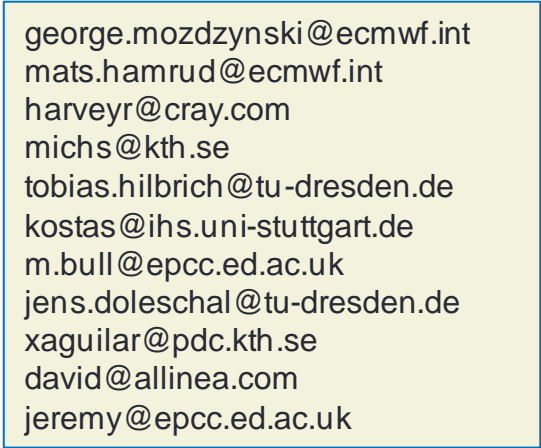


Get rest of the variables (26) from the red halo area and perform interpolations

Note that volume of halo data communicated is dependent on wind speed and direction in locality of each task

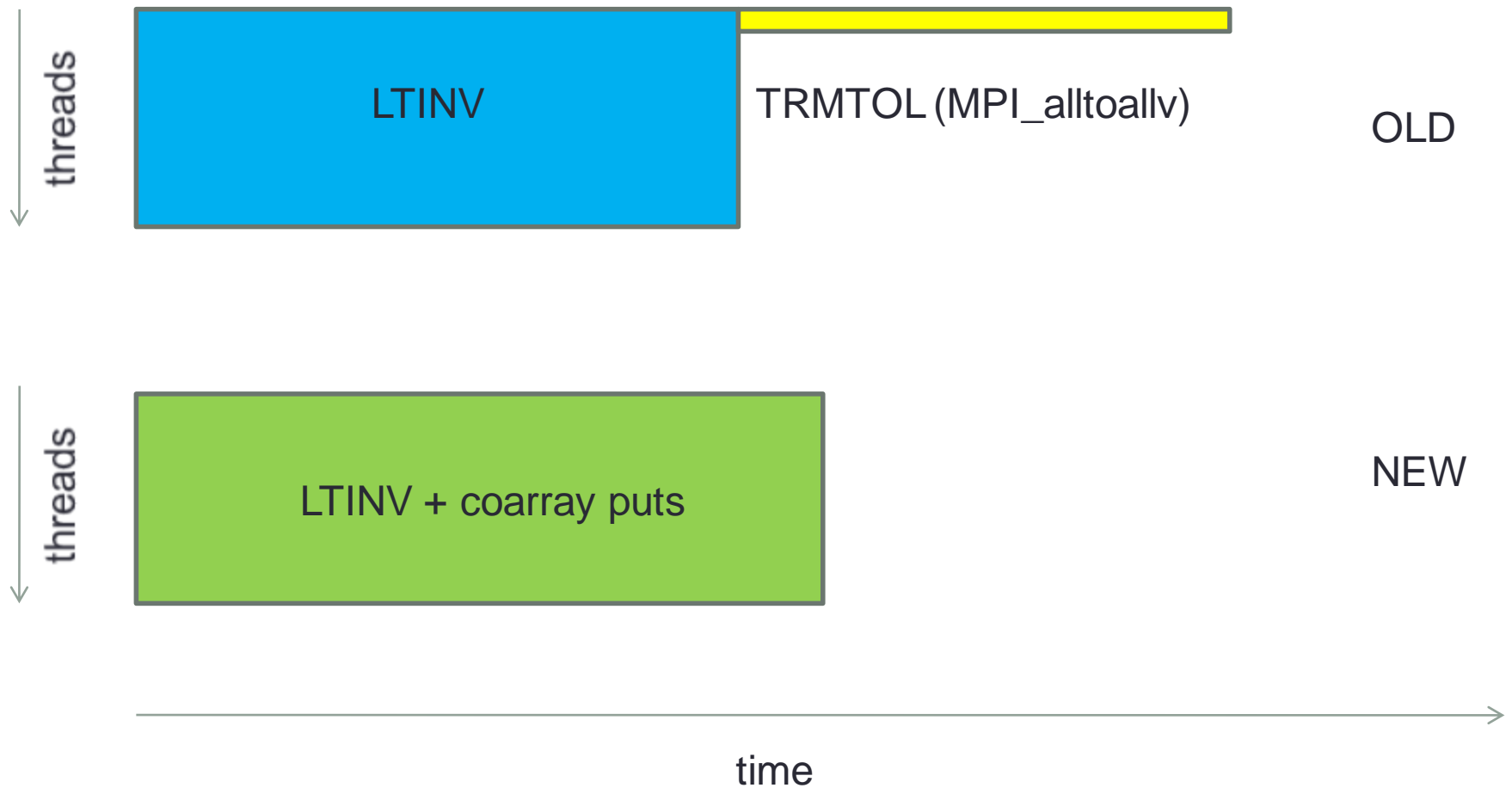
IFS Optimisations for ExaScale & Co-design

- All IFS optimisations in the CRESTA project
 - Involve use of Fortran2008 coarrays (CAF)
 - Used within context of OpenMP parallel regions
- Overlap Legendre transforms with associated transpositions
- Overlap Fourier transforms with associated transpositions
- Rework semi-Lagrangian communications
 - To substantially reduce communicated halo data
 - To overlap halo communications with SL interpolations
- CAF co-design team
 - caf-co-design@cresta-project.eu
 - ECMWF – optimise IFS as described above
 - CRAY – optimize DMAPP to be thread safe
 - TUD – visualize CAF operations in IFS with vampir
 - ALLINEA – debug IFS at scale with ddt (MPI/OMP/CAF)

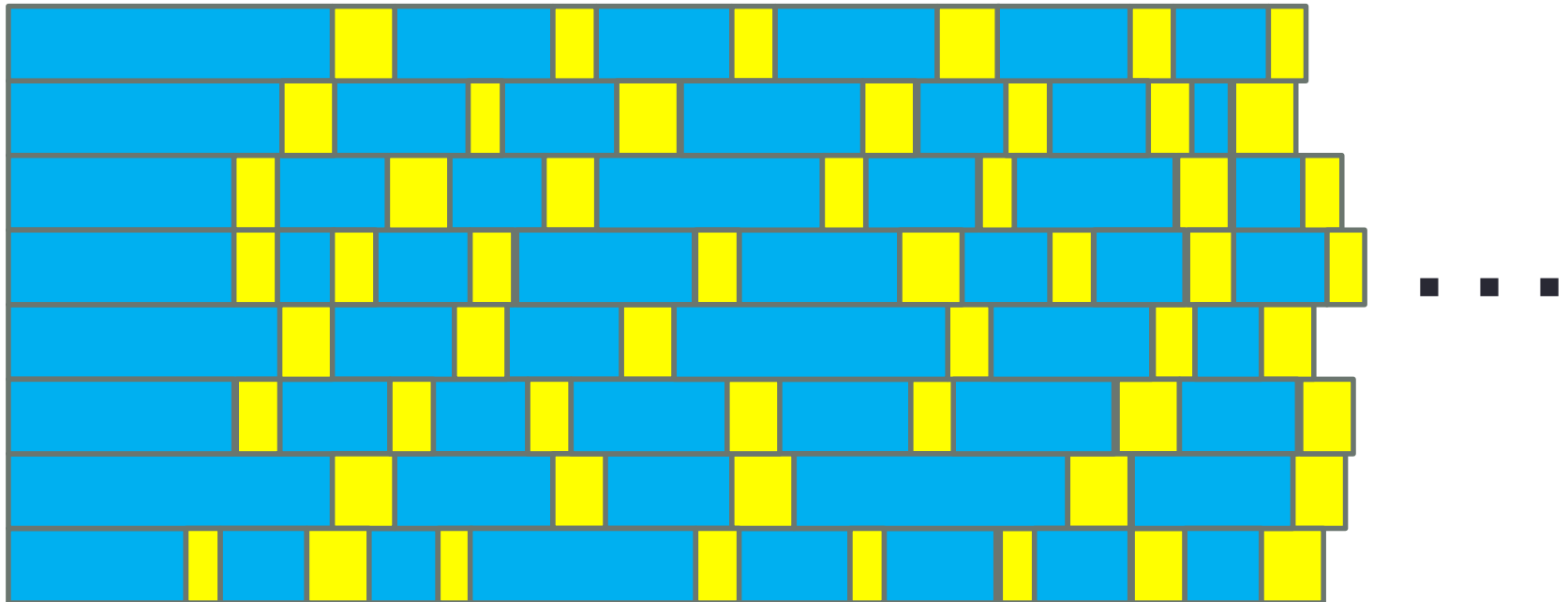


george.mozdzynski@ecmwf.int
mats.hamrud@ecmwf.int
harveyr@cray.com
michs@kth.se
tobias.hilbrich@tu-dresden.de
kostas@ihs.uni-stuttgart.de
m.bull@epcc.ed.ac.uk
jens.doleschal@tu-dresden.de
xaguiar@pdc.kth.se
david@allinea.com
jeremy@epcc.ed.ac.uk

Overlap Legendre transforms with associated transpositions



Overlap Legendre transforms with associated transpositions/3 (LTINV + coarray puts)



Expectation is that compute (LTINV-blue) and communication (coarray puts-yellow) overlap in time. We can now see this with an extension to vampir developed in CRESTA

Semi-Lagrangian – coarray implementation

red: only the halo points that are used are communicated

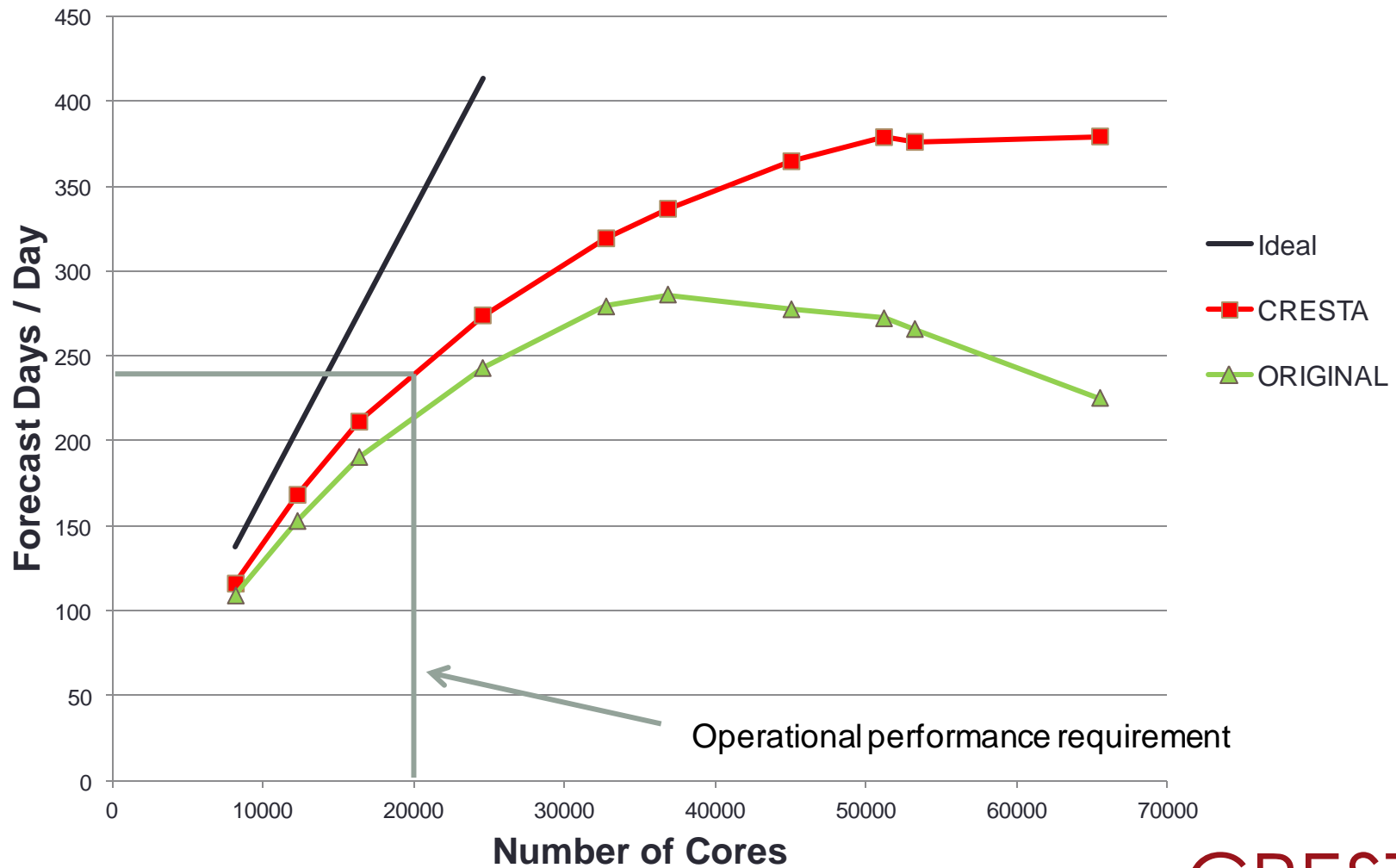


Note no more blue area (max wind halo) and associated overhead.

Also, halo coarray transfers take place in same OpenMP loop as the interpolations.

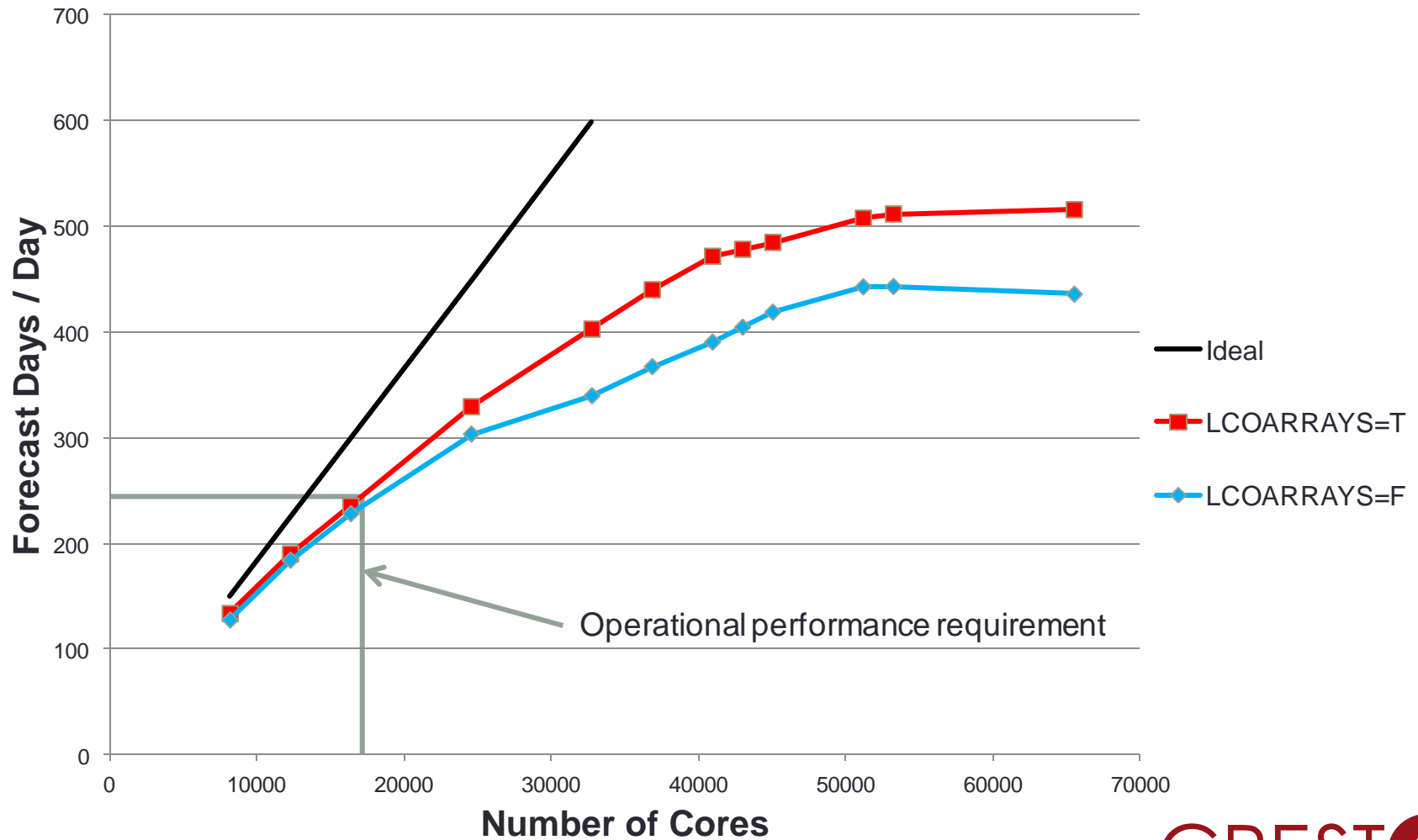
T2047L137 model performance on HECToR (CRAY XE6) RAPS12 IFS (CY37R3), cce=7.4.4

APRIL 2012

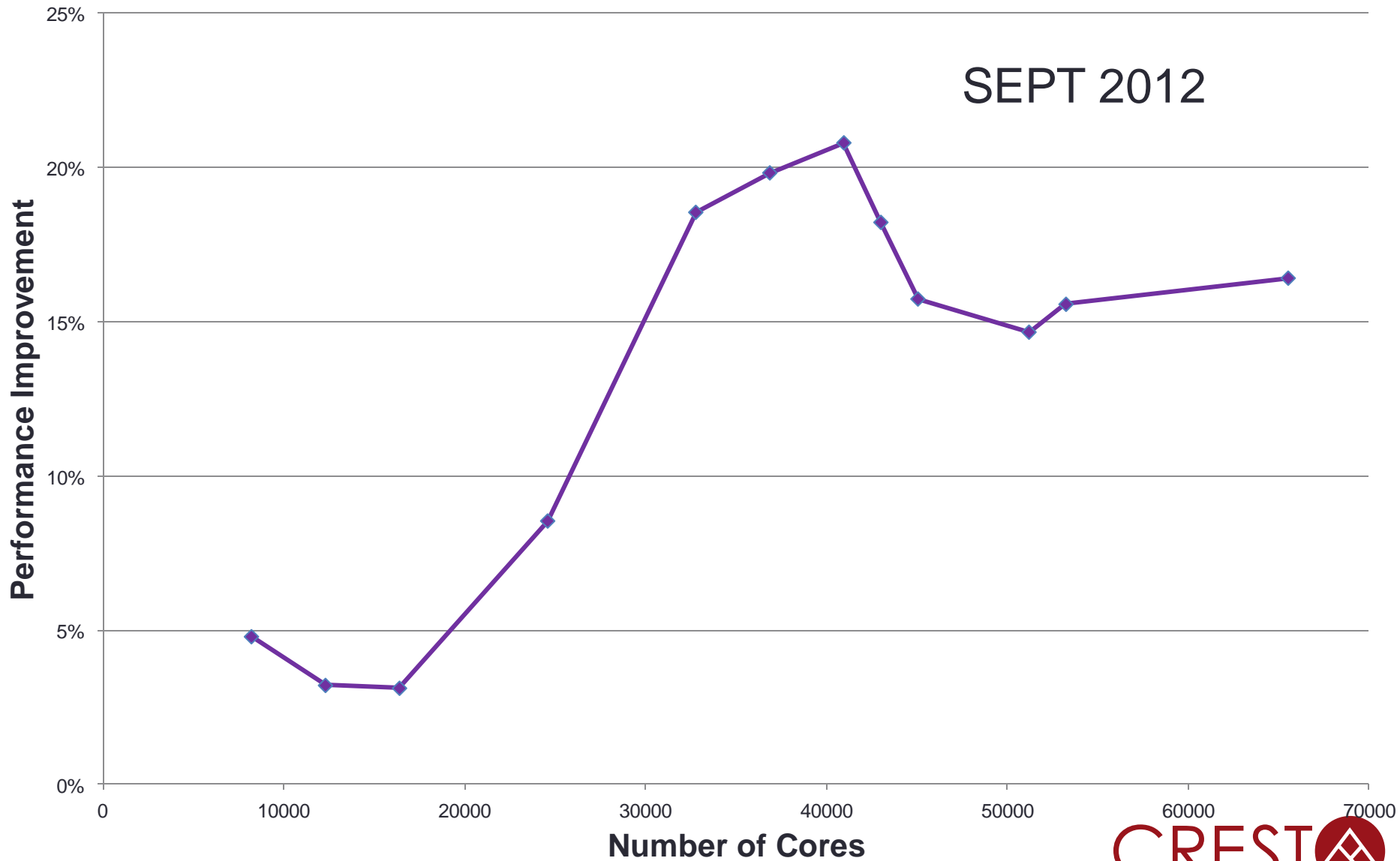


T2047L137 model performance on HECToR (CRAY XE6) RAPS12 IFS (CY37R3), cce=8.0.6 -hflex_mp=intolerant

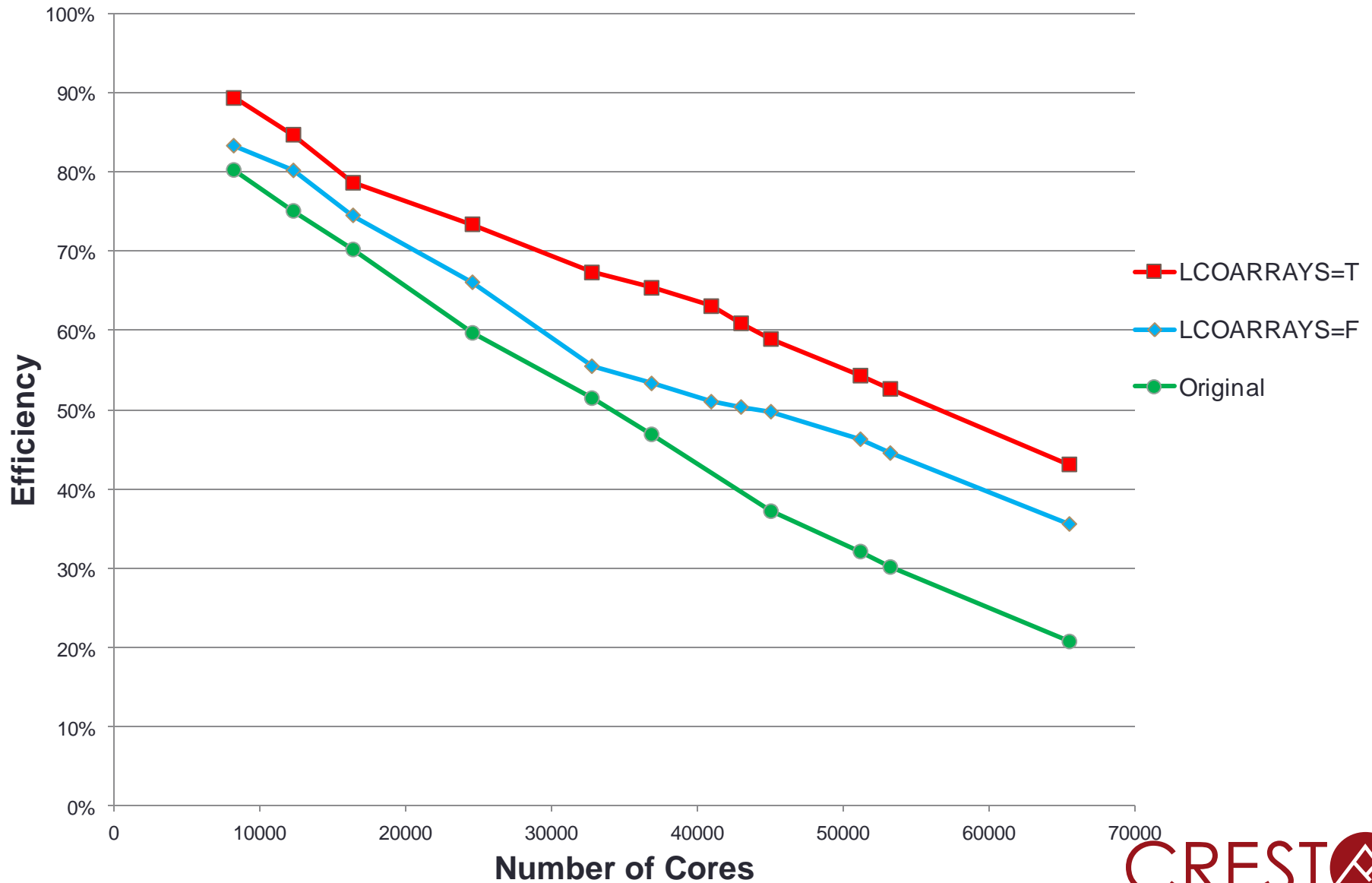
SEPT 2012



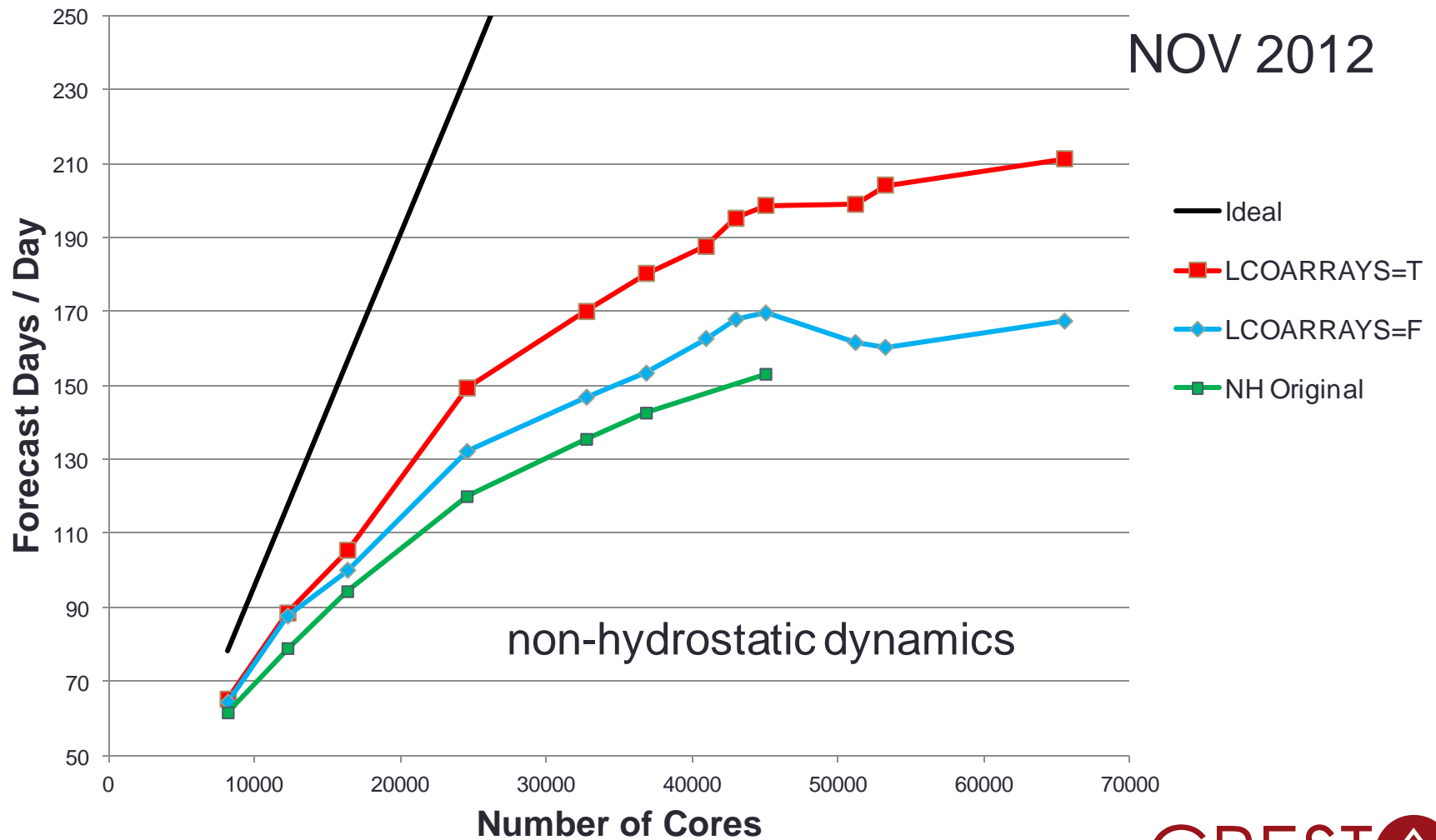
T2047L137 IFS model performance improvement by using Fortran2008 coarrays on HECToR (CRAY XE6)



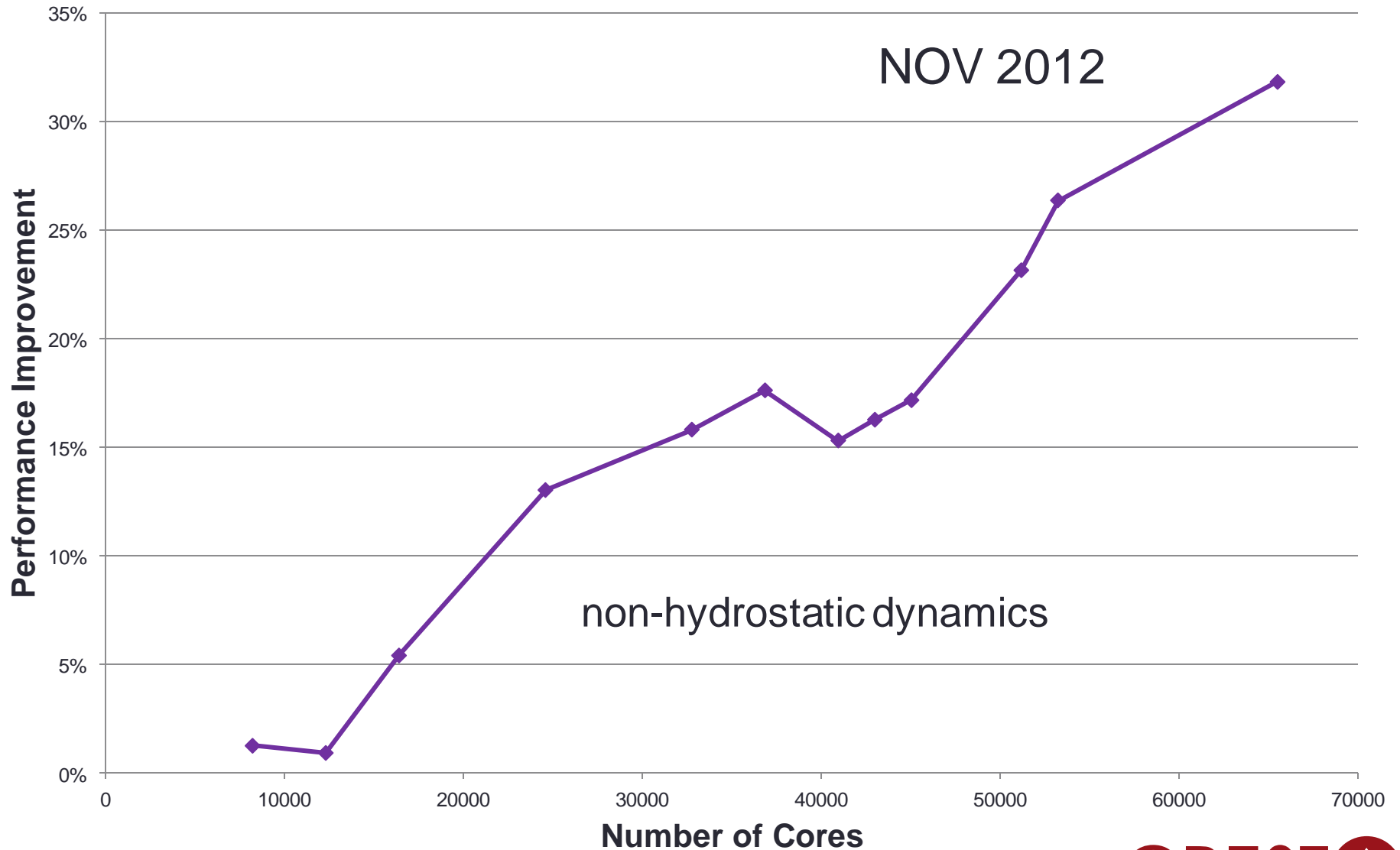
T2047L137 IFS model Efficiency on HECToR RAPS12 IFS (CY37R3)



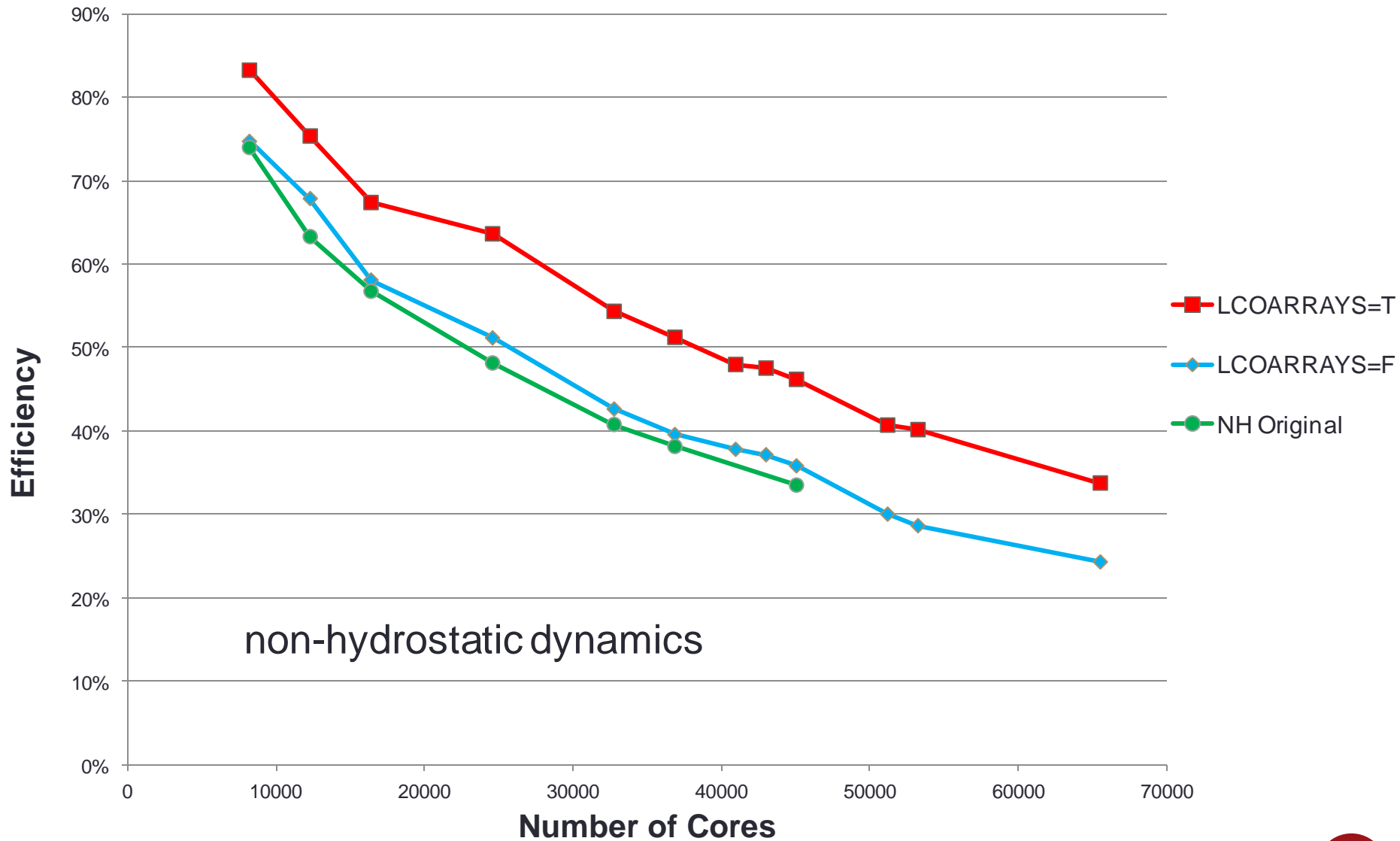
T2047L137 NH model performance on HECToR (CRAY XE6) RAPS12 IFS (CY37R3), cce=8.0.6 -hflex_mp=intolerant

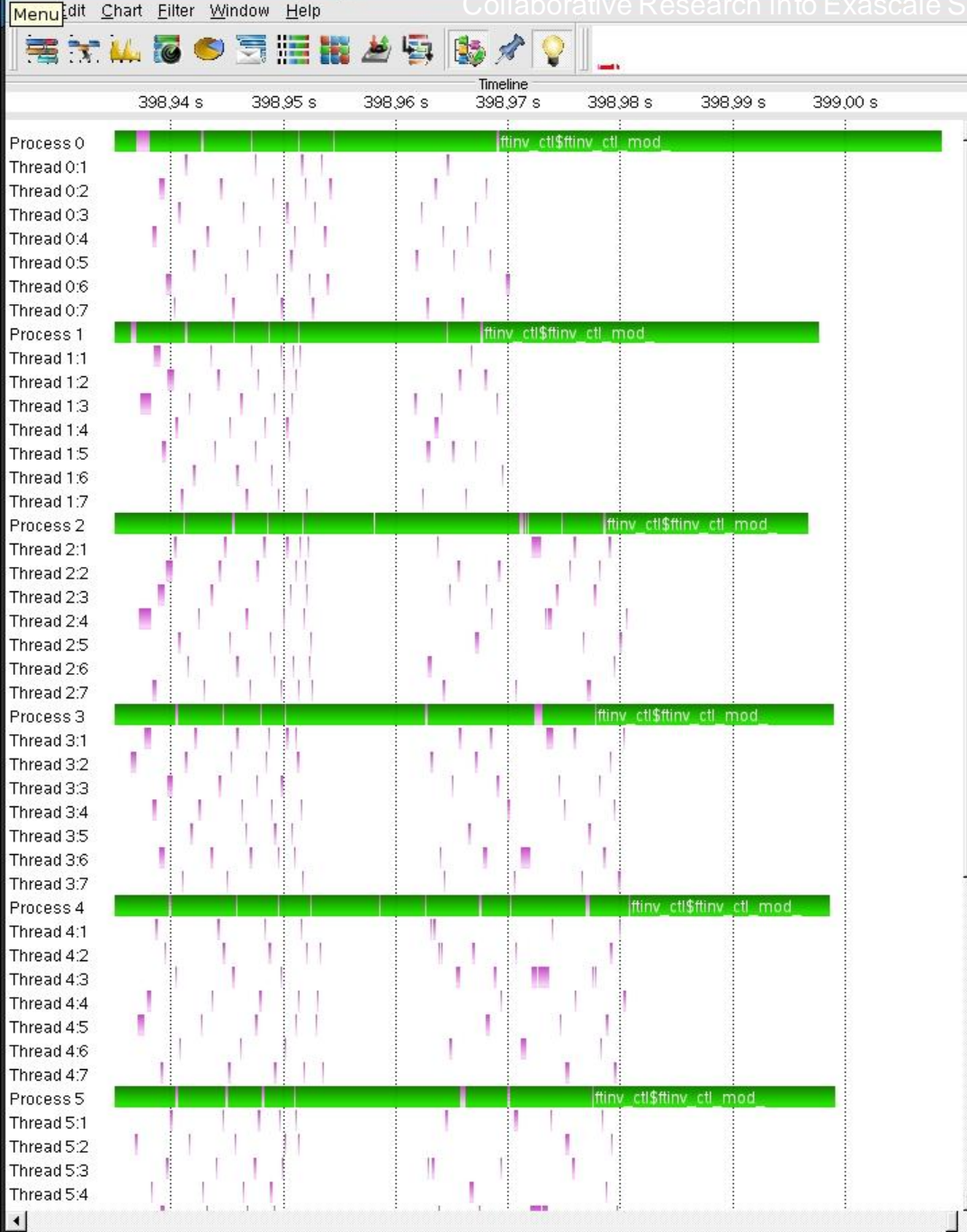


T2047L137 NH IFS model performance improvement by using Fortran2008 coarrays on HECToR (CRAY XE6)



T2047L137 NH model Efficiency on HECToR (CRAY XE6) RAPS12 IFS (CY37R3)





Function Summary

All Processes, Accumulated Exclusive Time per Function Group

Function Group	Accumulated Exclusive Time
Application	0.514 s
DMAPP	0.1 s

Context View

Property	Value
Display	Master Timeline
Type	Function
Location	Process 0
Function	ftinv_ctl\$ftinv_ctl_mod
Function Group	Application
Interval Begin	398.938043 s
Interval End	398.942646 s
Duration	4.6023 ms
Source Loca	ftinv_ctl_mod.F90:3

Process Summary

Similar Processes, Accumulated Exclusive Time per Function

Process ID	Function	Accumulated Exclusive Time
15	ftinv_ctl\$ftinv_ctl_mod	~15 ms
14	ftinv_ctl\$ftinv_ctl_mod	~14 ms
10	ftinv_ctl\$ftinv_ctl_mod	~10 ms
9	ftinv_ctl\$ftinv_ctl_mod	~9 ms
8	ftinv_ctl\$ftinv_ctl_mod	~8 ms
6	ftinv_ctl\$ftinv_ctl_mod	~6 ms
2	ftinv_ctl\$ftinv_ctl_mod	~2 ms

Call Tree

All Processes

Function	Min Inclusive Time	Max Inclusive Time
ftinv_ctl\$ftinv_ctl_mod	0.000 s	26.442 ms
ftinv_ctl\$ftinv_ctl_mod	0.000 s	52.853 ms
dmapp syncid wait	0.000 s	5.590 µs
dmapp set rma attrs	0.000 s	5.777 µs
dmapp put	0.000 s	2.995 ms
dmapp get rma attrs	0.000 s	12.411 µs
dmapp get nb	0.000 s	84.420 µs

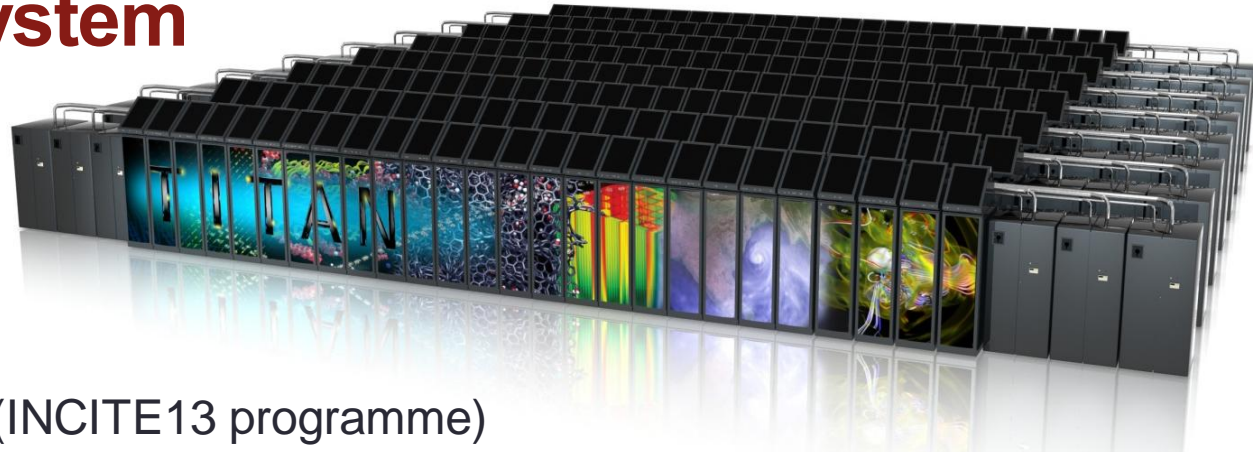
Function Legend

- Application (Green)
- DMAPP (Purple)
- MPI (Red)

Schedule for IFS optimizations in CRESTA (planned)

When	Activity
1Q2013	<p>RAPS13 IFS CY38R2 preparation and port to HECToR</p> <ul style="list-style-type: none"> • including Fast Legendre Transform <p>Run T3999 model on HECToR</p> <p>Initial use of GPUs for IFS (targeting costly LTINV/LTDIR dgemm's)</p> <ul style="list-style-type: none"> • Expose additional parallelism by performing dgemms in each FLT butterfly stage in parallel (good for HYPER-Q performance)
1H2013	<p>Scaling runs of T3999 IFS model on TITAN (CRESTA INCITE award)</p> <ul style="list-style-type: none"> • T3999 is targeted for ECMWF operations in 2023-24
2013-2014	<p>Explore use of DAG parallelization (like DPLASMA)</p> <ul style="list-style-type: none"> • Expect 12+ months effort, possibly more depending on scope of DAG'd areas <p>Test with IBM F2008 'technology preview' compiler on Power7 at ECMWF</p> <p>Other IFS scalability optimizations</p> <ul style="list-style-type: none"> • transpose SL data, physics load balancing, +++ <p>Development & testing of a future solver for IFS (Plan B) – NA section</p> <ul style="list-style-type: none"> • Following closely developments in GungHO! project (MetOffice, NERC, STFC) • GungHO=<u>G</u>lobally <u>U</u>niform <u>N</u>ext <u>G</u>eneration <u>H</u>ighly <u>O</u>ptimized

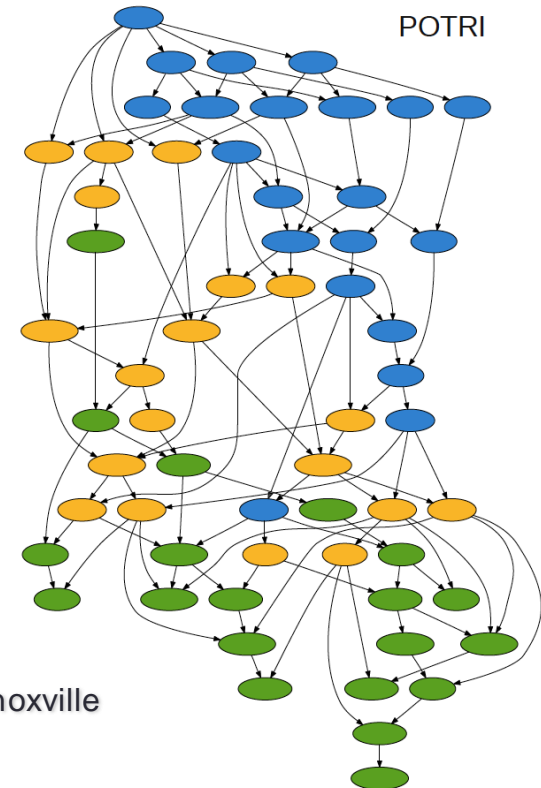
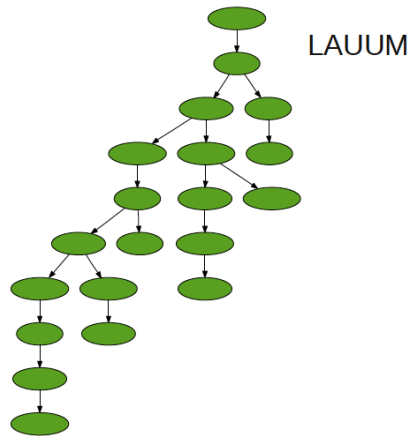
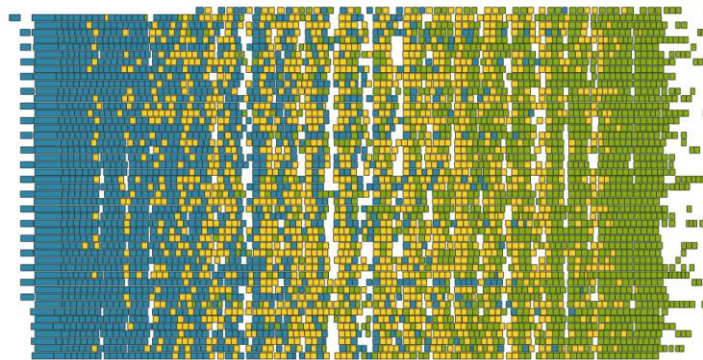
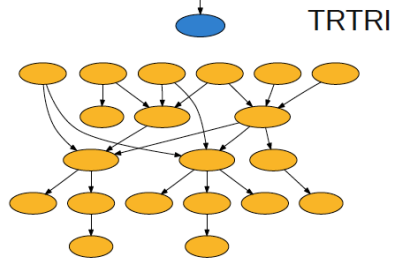
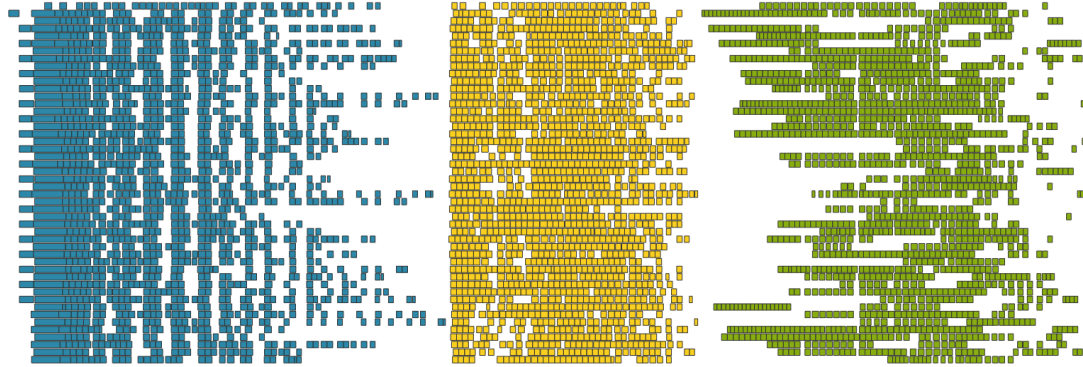
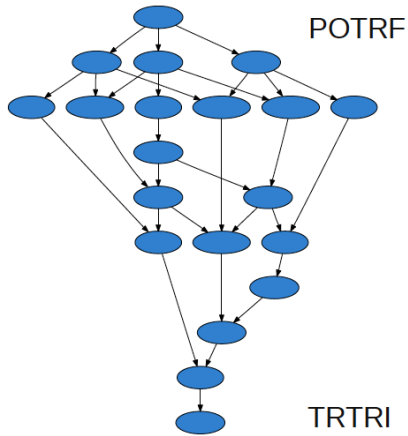
ORNL's "Titan" System



- #1 in Nov 2012 Top500 list
- CRESTA awarded access (INCITE13 programme)
- 18X peak perf. of ECMWF's P7 clusters (C2A+C2B=1.5 Petaflops)
- Upgrade of Jaguar from Cray XT5 to XK6
- Cray Linux Environment operating system
- Gemini interconnect
 - 3-D Torus
 - Globally addressable memory
- AMD Interlagos cores (16 cores per node)
- New accelerated node design using NVIDIA K20 "Kepler" multi-core accelerators
- 600 TB DDR3 mem. + 88 TB GDDR5 mem

Titan Specs	
Compute Nodes	18,688
Login & I/O Nodes	512
Memory per node	32 GB + 6 GB
# of NVIDIA K20 "Kepler" processors	14,592
Total System Memory	688 TB
Total System Peak Performance	27 Petaflops

DAG example: Cholesky Inversion



DAG = Directed Acyclic Graph

Can IFS use this technology?

Source: Stan Tomov, ICL, University of Tennessee, Knoxville

A wide-angle photograph of a snowy mountain range under a clear blue sky. The central focus is a sharp, snow-covered mountain peak. In the foreground, a cable car station is partially visible, surrounded by a fence and snow-covered ground. The overall scene is bright and clear.

Thank you for your
attention

QUESTIONS?

IFS model coarray developments

Compile with `-DCOARRAYS`

for compilers that support Fortran2008 coarray syntax

Run with,

`&NAMPAR1`

`LCOARRAYS=true,`

to use coarray optimizations

`&NAMPAR1`

`LCOARRAYS=false,`

to use original MPI implementation

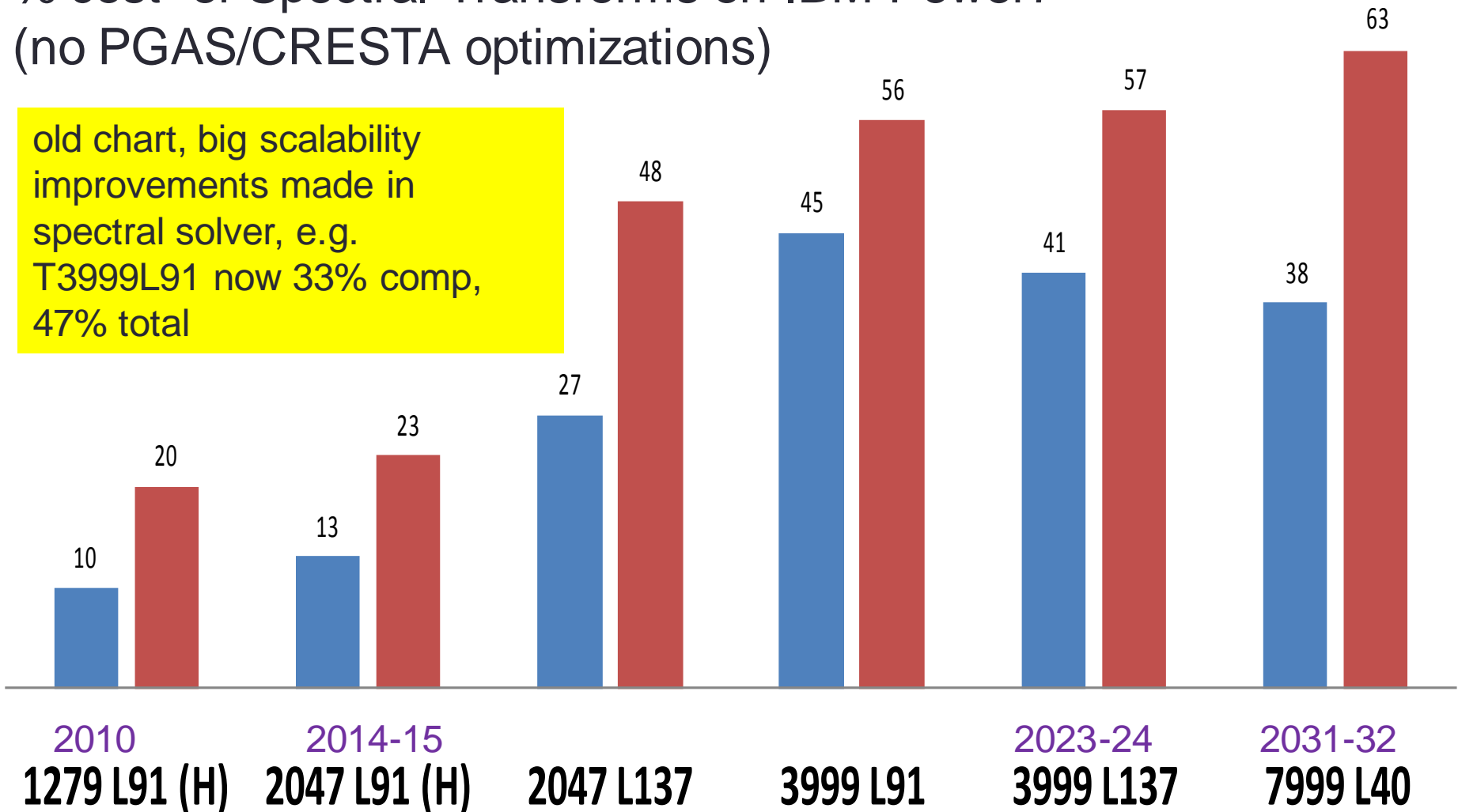
Motivation – T2047 and T3999 costs on IBM Power7 (percentage of wall clock time)

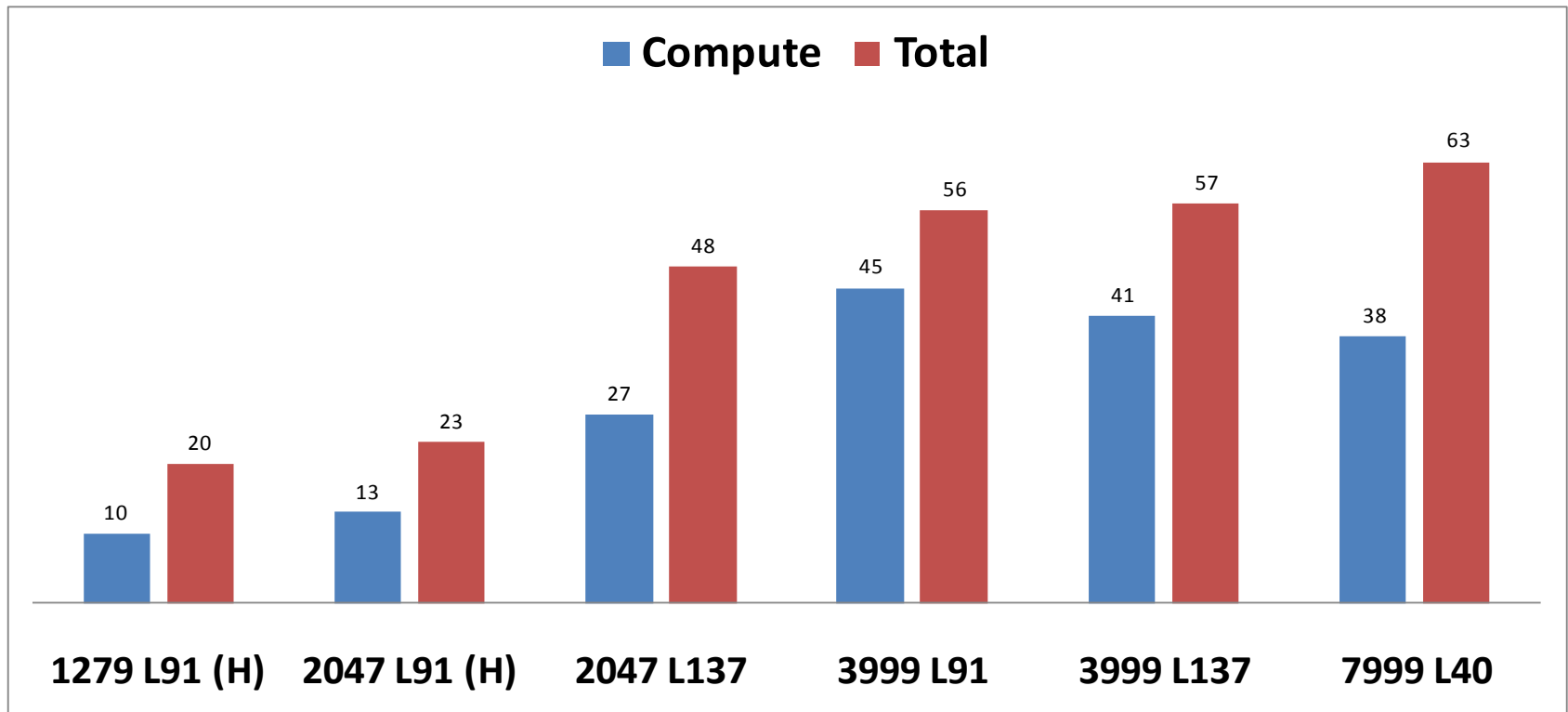
		T2047 (%) 2014-15	T3999 (%) 2020-21
LTINV_CTL	- INVERSE LEGENDRE TRANSFORM	3.30	8.40
LTINV_CTL	- M TO L TRANSPOSITION	5.37	5.24
LTDIR_CTL	- DIRECT LEGENDRE TRANSFORM	3.56	5.30
LTDIR_CTL	- L TO M TRANSPOSITION	2.84	3.14
FTDIR_CTL	- DIRECT FOURIER TRANSFORM	0.20	1.07
FTDIR_CTL	- G TO L TRANSPOSITION	2.85	2.21
FTINV_CTL	- INVERSE FOURIER TRANSFORM	0.72	3.76
FTINV_CTL	- L TO G TRANSPOSITION	4.47	7.36
	SUM (%)	23.4	36.5
		L137/LT	L91/FLT
		4224Tx8t	1024Tx16t
		528 Nodes	256 Nodes
		470 FD/D	28 FD/D

■ Compute ■ Total

% cost of Spectral Transforms on IBM Power7
(no PGAS/CRESTA optimizations)

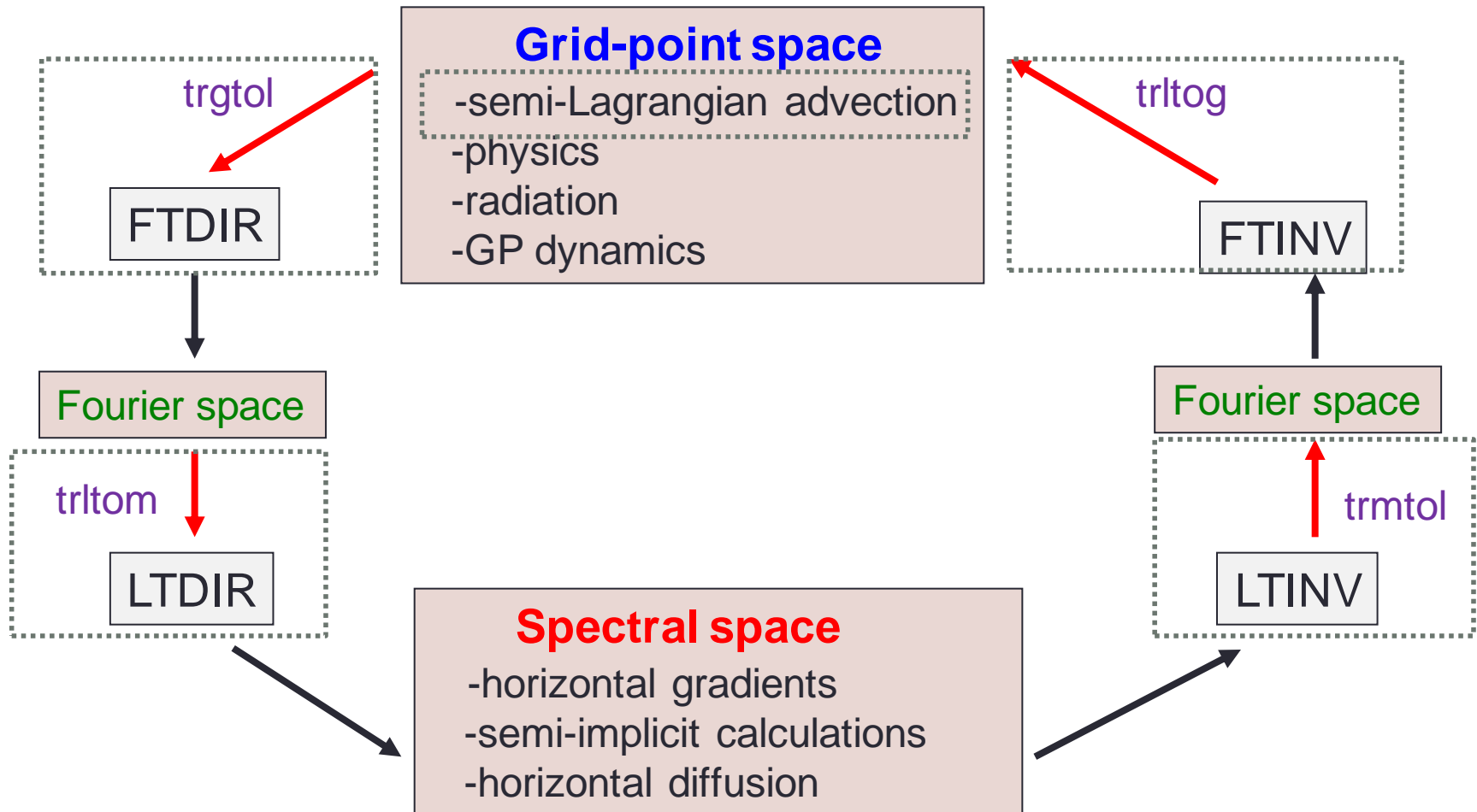
old chart, big scalability
improvements made in
spectral solver, e.g.
T3999L91 now 33% comp,
47% total

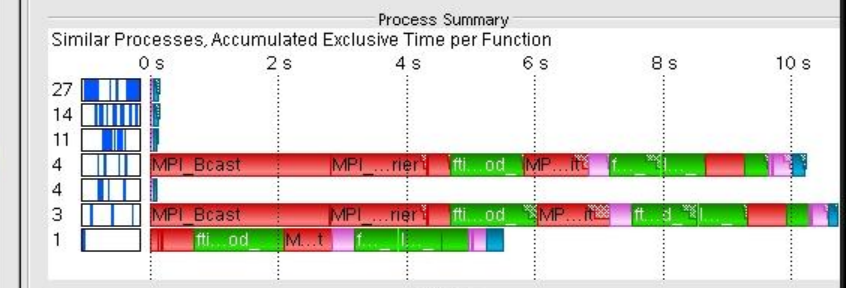
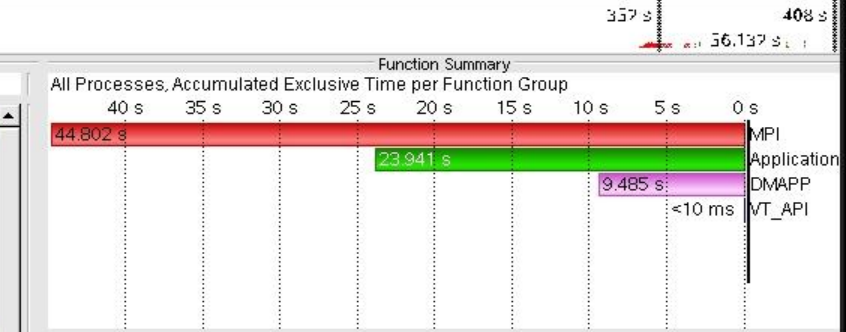
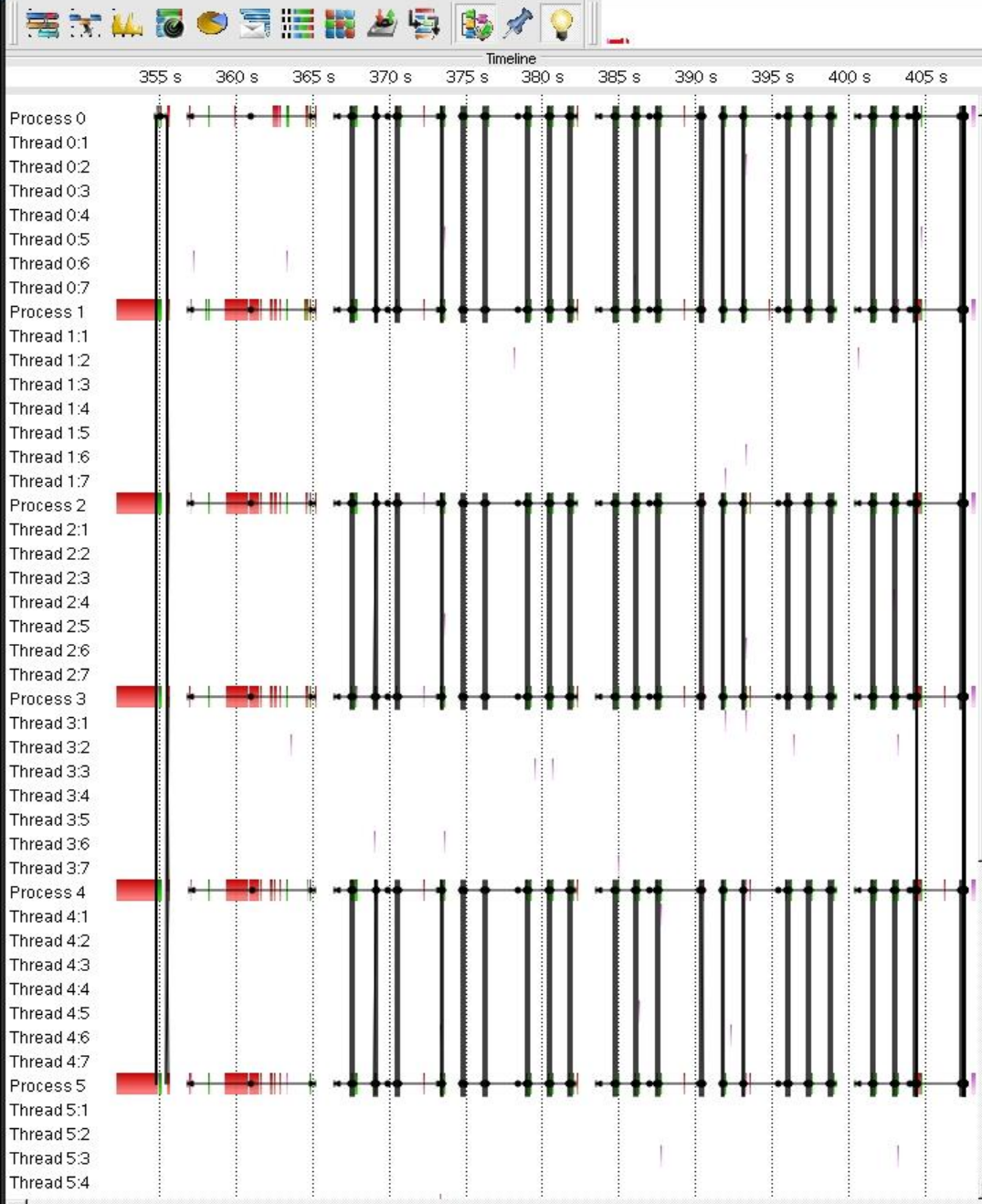




Relative **computational** cost of the spherical harmonics transforms plus the spectral computations (solving the Helmholtz equation) as a percentage of the overall model cost for various configurations. Red bars indicate the total cost **including** the global communications involved. Percentages have been derived considering all gridpoint dynamics and physics computations but without considering IO, synchronization costs (barriers), and any other ancillary costs. All runs are non-hydrostatic unless indicated with (H). All runs further show that the communications cost is less than or equal to the compute cost on the IBM Power7 and have good potential for “hiding” this overhead. However, communication cost is likely to increase with the number of cores.

Planned IFS optimisations for [Tera,Peta,Exa]scale

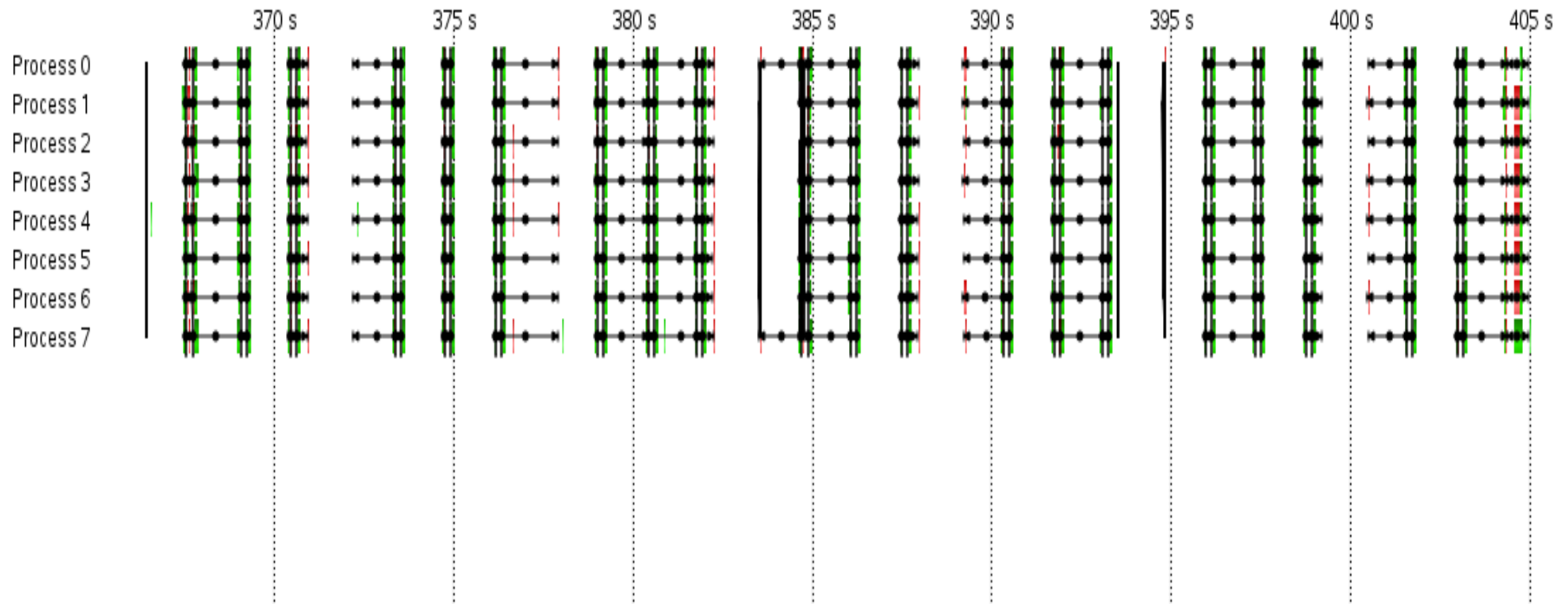




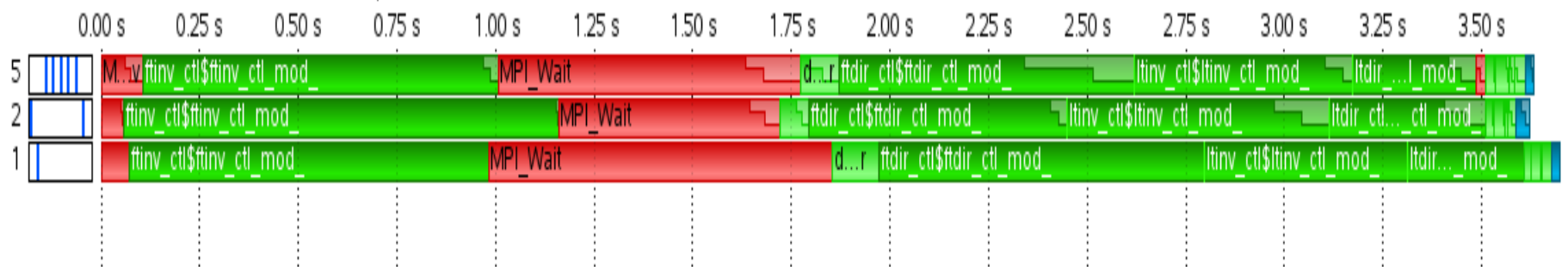
All Processes

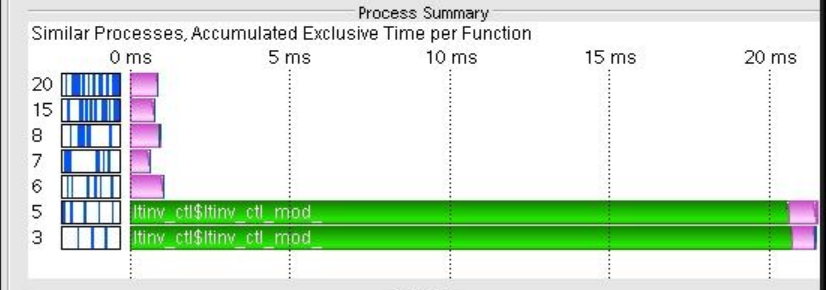
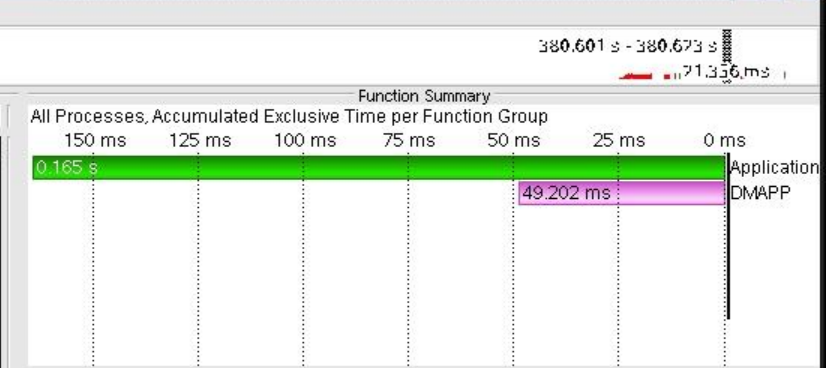
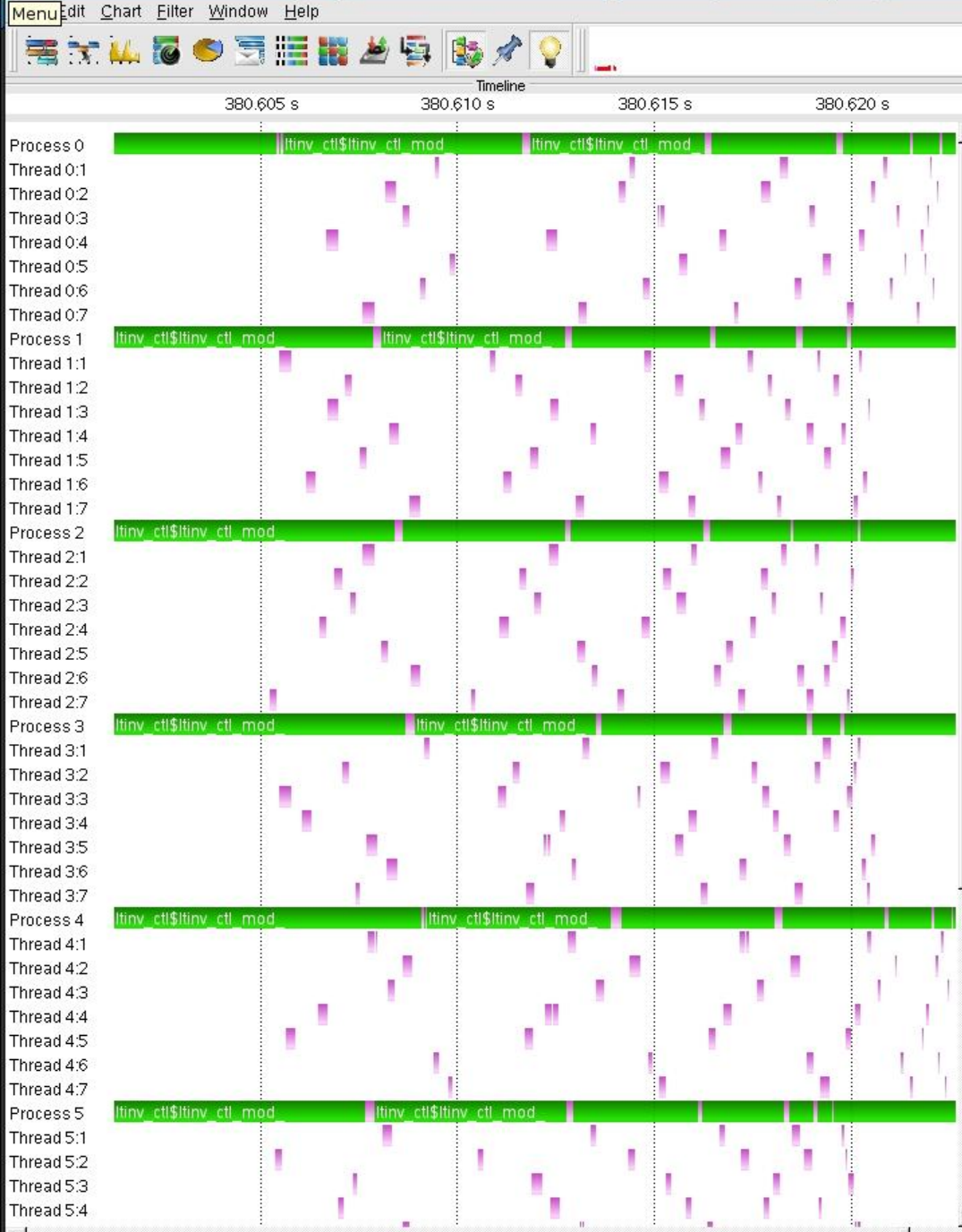
Function	Min Inclusive Time	Max Inclusive Time
ltnv ctl\$ltinv ctl mod	0.000 s	0.927 s
ltdir ctl\$ltdir ctl mod	0.000 s	0.466 s
ftinv ctl\$ftinv ctl mod	0.000 s	1.504 s
ftdir ctl\$ftdir ctl mod	0.000 s	1.087 s
dmapp syncid wait	0.000 s	341.146 μs
dmapp sheap malloc	0.000 s	2.197 μs
dmapp sheap free	0.000 s	3.279 μs
dmapp set rma attrs	407.629 μs	3.488 ms
dmapp put ixpe	0.000 s	35.447 μs





Similar Processes, Accumulated Exclusive Time per Function



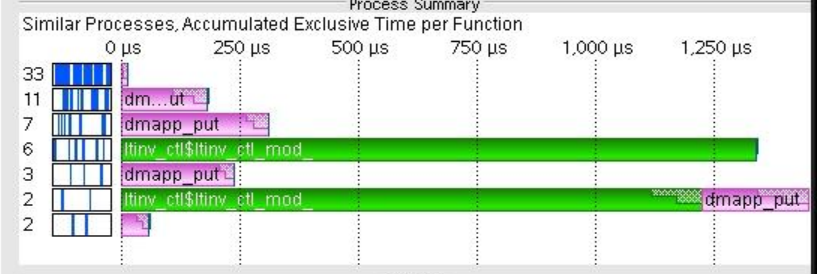
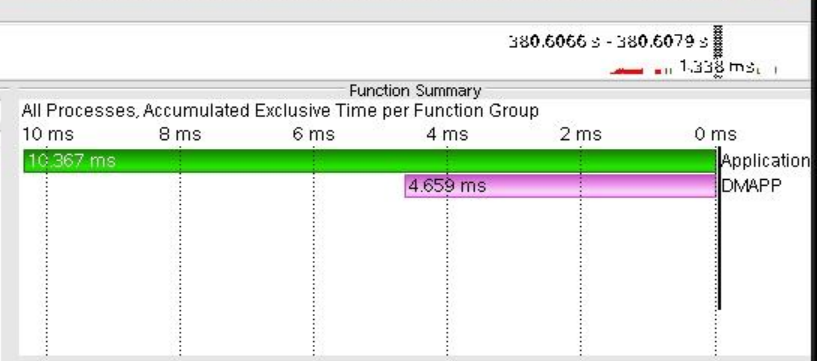


Call Tree

All Processes

Function	Min Inclusive Time	Max Inclusive Time
ltinv_ctl\$ltinv_ctl_mod	0.000 s	25.866 ms
dmapp put	0.000 s	1.013 ms

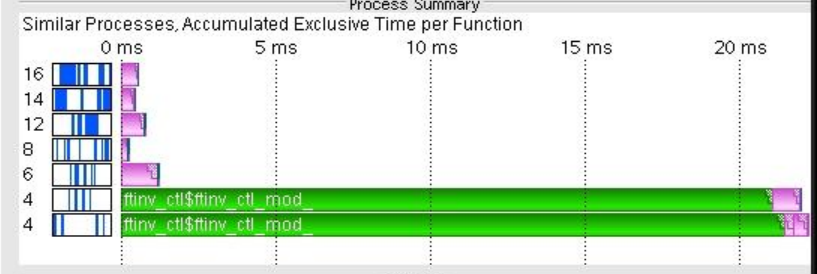
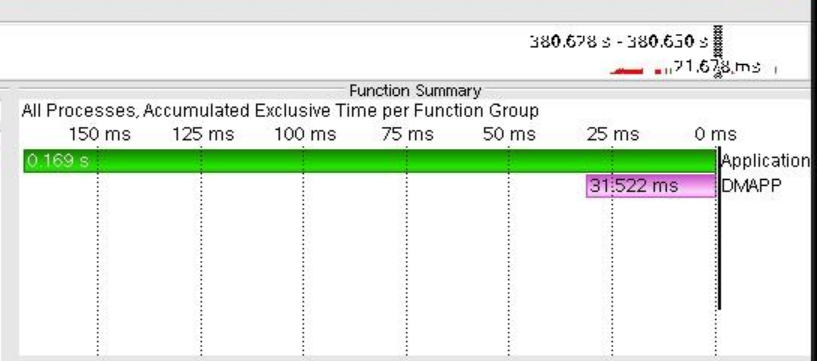
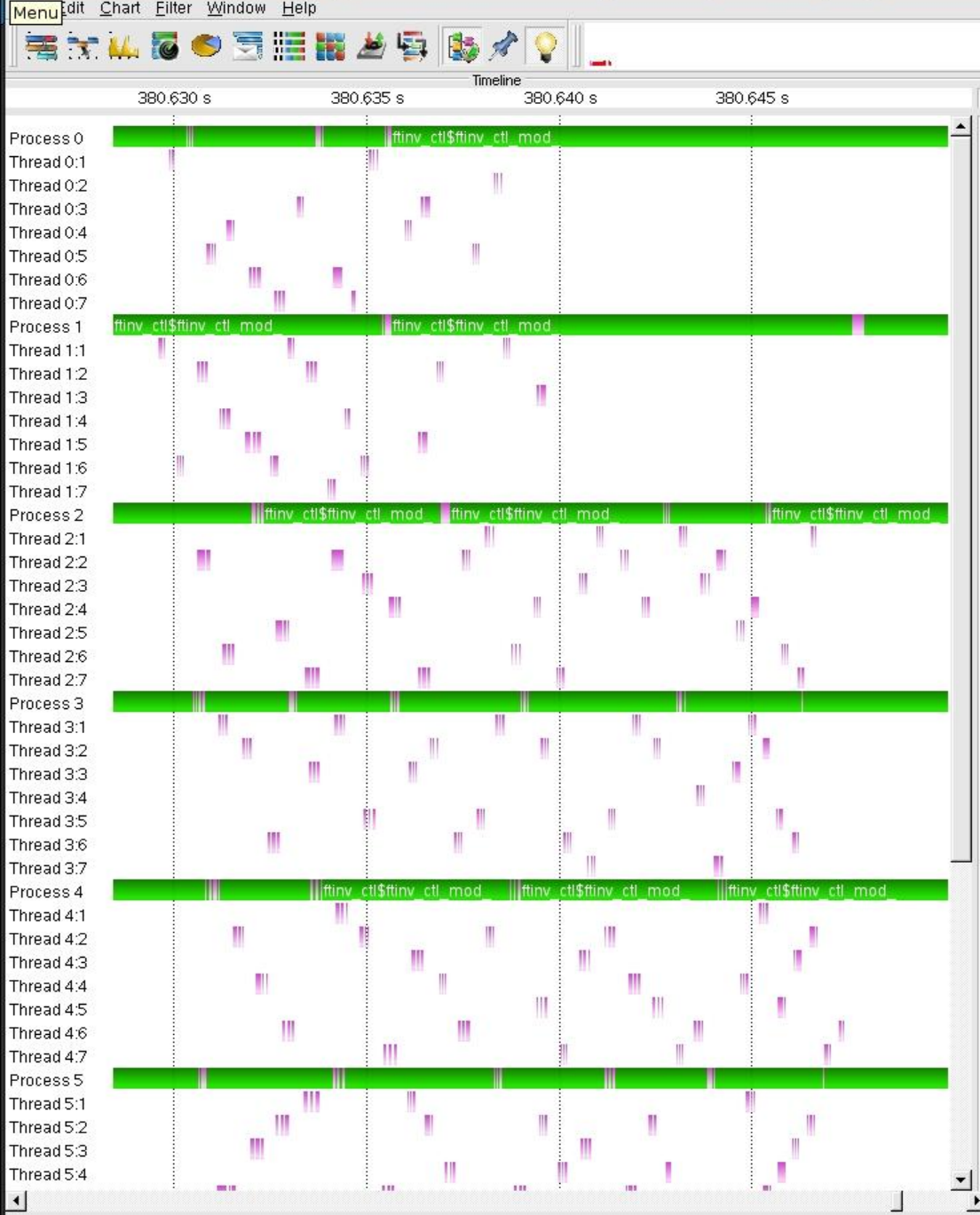




All Processes

Function	Min Inclusive Time	Max Inclusive Time
Application	0.000 s	12.183 ms
DMAPP	0.000 s	308.858 μs



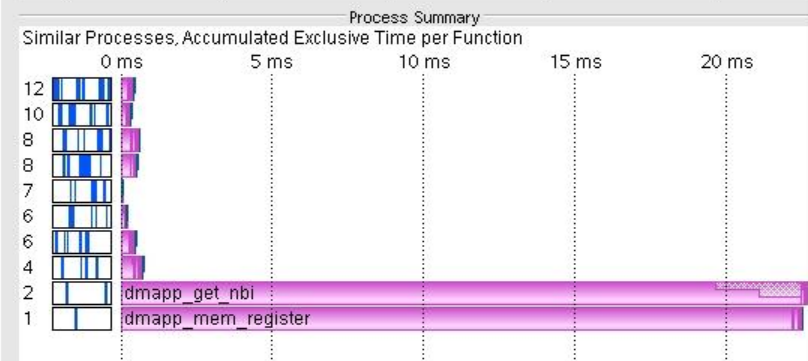
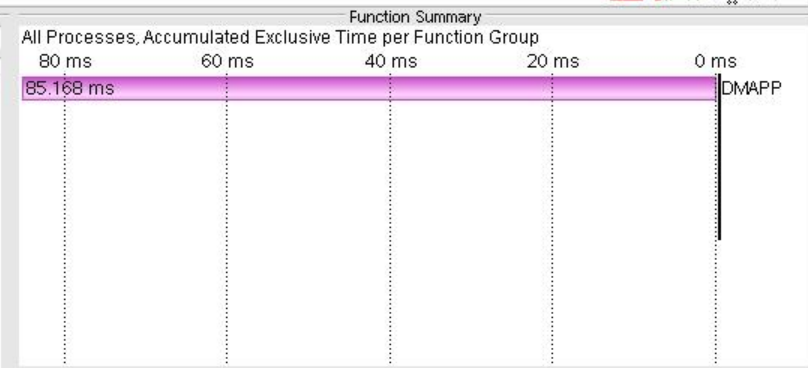
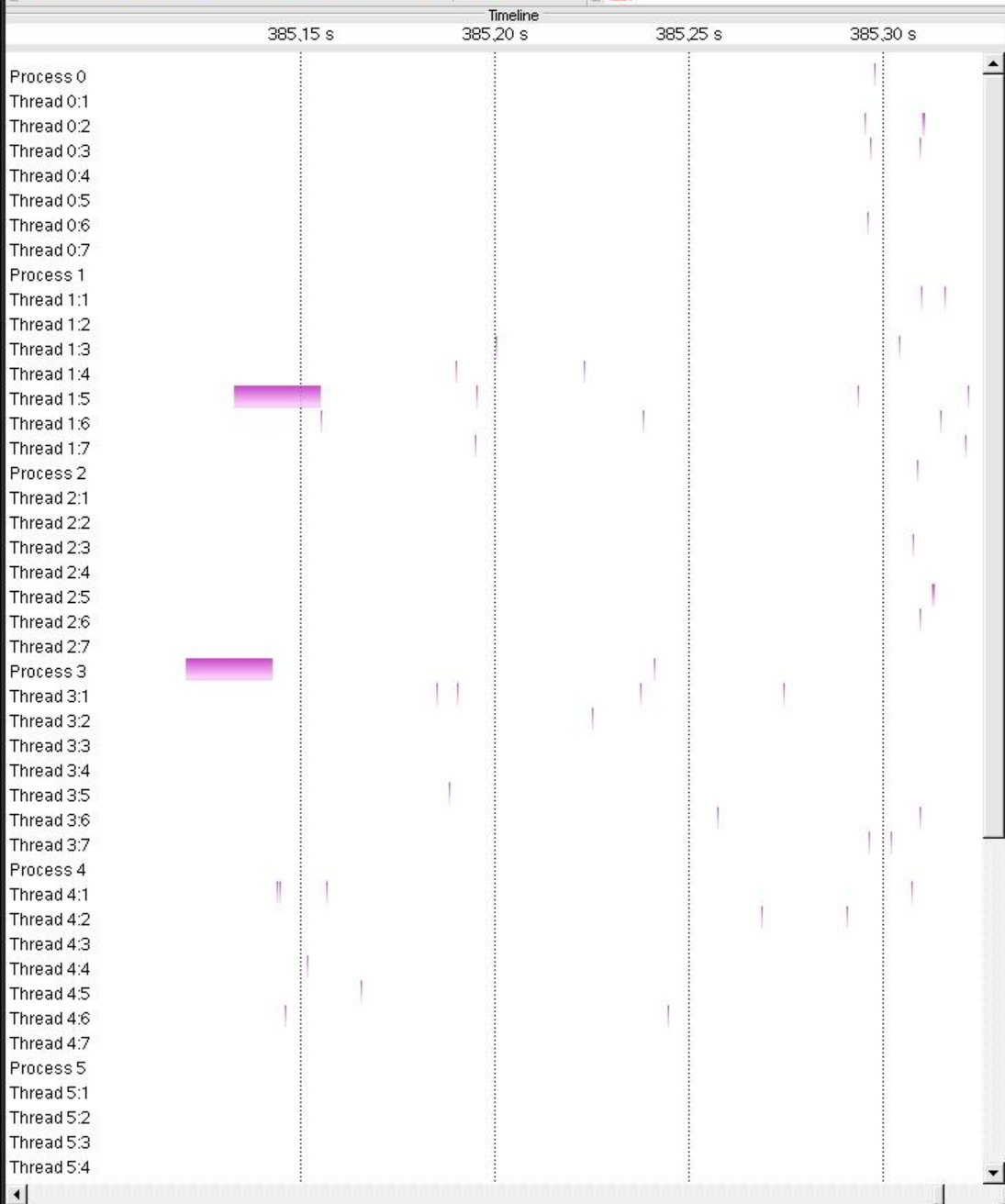


Call Tree

All Processes

Function	Min Inclusive Time	Max Inclusive Time
ftinv_ctl\$ftinv_ctl_mod	0.000 s	50.984 ms
dmapp syncid wait	0.000 s	5.377 μs
dmapp set rma attrs	0.000 s	6.327 μs
dmapp put	0.000 s	1.158 ms
dmapp get rma attrs	0.000 s	11.596 μs
dmapp get nb	0.000 s	10.102 μs

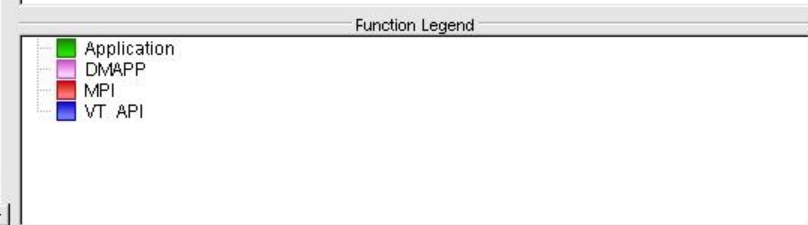




Call Tree

All Processes

Function	Min Inclusive Time	Max Inclusive Time
dmapp set rma attrs	0.000 s	109.670 μs
dmapp mem register	0.000 s	22.190 ms
dmapp get rma attrs	0.000 s	175.998 μs
dmapp get_nbi	0.000 s	22.464 ms
dmapp afadd qw_nbi	0.000 s	16.133 μs



Other Fortran 2008 compilers

- License finally agreed with IBM
 - ECMWF will install xlf v14 compiler on Power7
 - Only took 1 year from first inquiry (pre-CRESTA)
 - Subject to non-disclosure
 - Am sure we will be granted permission to present and publish results if they are good
 - Plan is first to test IFS RAPS12 with this compiler
- Promoting need for Fortran 2008 to vendors is important
- Intel ?
- Fujitsu ?
- gfortran ?
- PGI ?

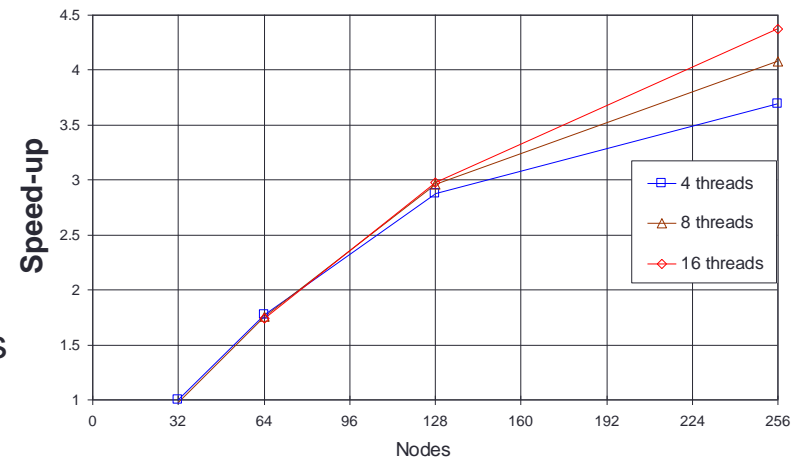
Using HECToR

- Moved IFS from cce=8.0.3 to cce=8.0.6
 - To pick up fix to random hangs at start of job
 - Job would run to cp time limit without executing a single application statement
 - Refunded lost KAu's
 - 8.0.6 also fixed a couple of random coarray runtime failures
 - Thanks to CRAY for providing a good compiler release
- Multiple aprun's in high core count jobs (10K to 64K cores)
 - To improve overall system resource utilization
 - Small, medium and large batched jobs
 - Some waste due to unused cores in each job
 - Promise of refund (more KAUs) at some time in future

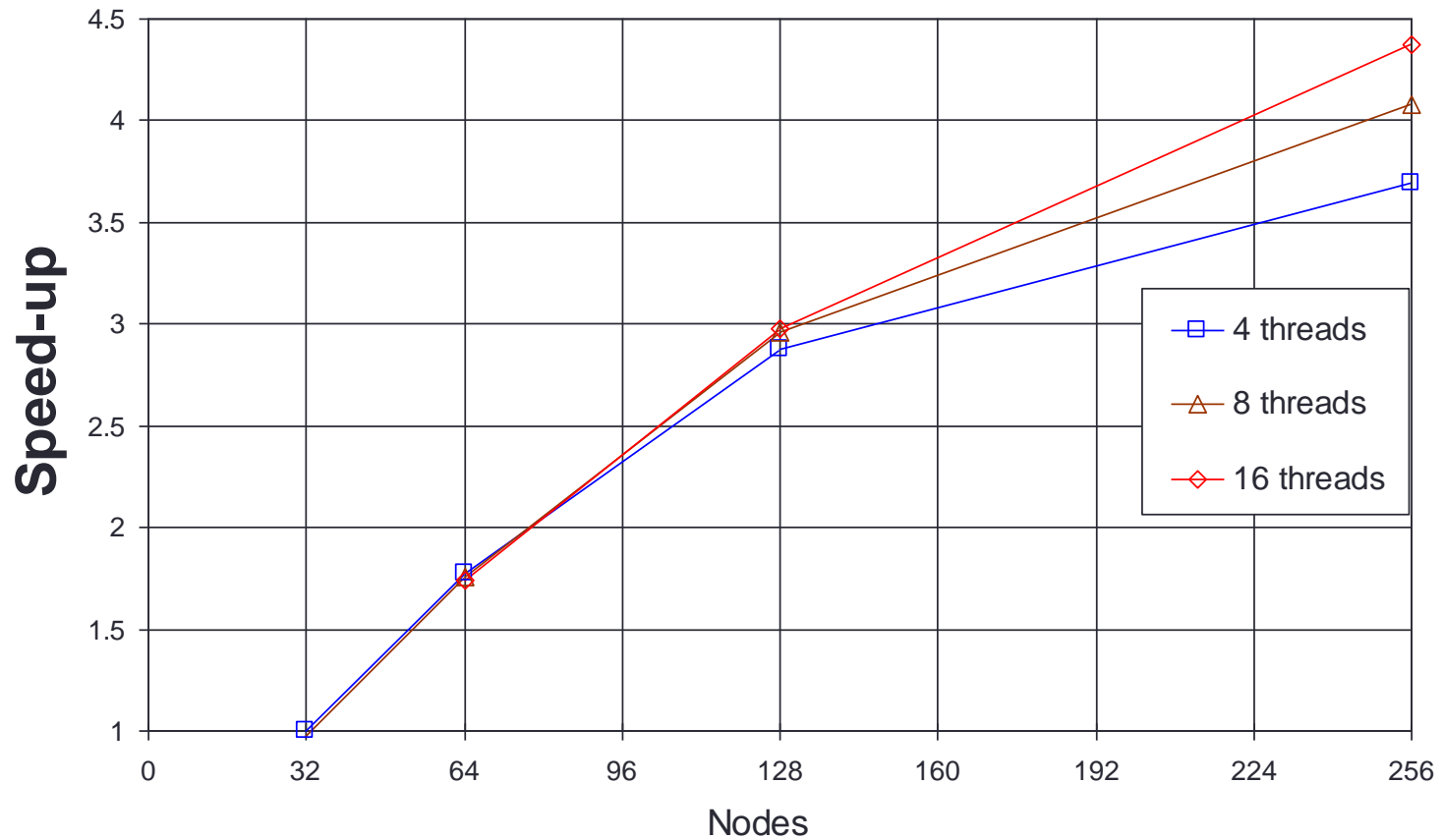
Hybrid runtime support - IFS

- Initial IFS MPI implementation 1994-1996
- Hybrid MPI/OpenMP implementation ~1999
 - OpenMP implementation at highest level
 - Single parallel regions for each of physics, radiation scheme, dynamics, Legendre transforms, Fourier transforms and Fourier space computations
 - Schedule dynamic used in most parallel regions
- Hybrid implementation benefits
 - About 20 percent performance improvement at scale
 - Huge memory savings , memory use reduces linearly with number of OpenMP threads
- Next evolutionary step: use of Fortran 2008 coarrays to
 - Overlap computation with communication in transpositions
 - Fourier space <-> Spectral space comms, overlapped with Legendre transforms
 - grid point space <-> Fourier space comms, overlapped with FFTs and Fourier space computations
 - Reduce total halo communication in semi-Lagrangian scheme
 - Dominant coarray communications in OpenMP parallel regions

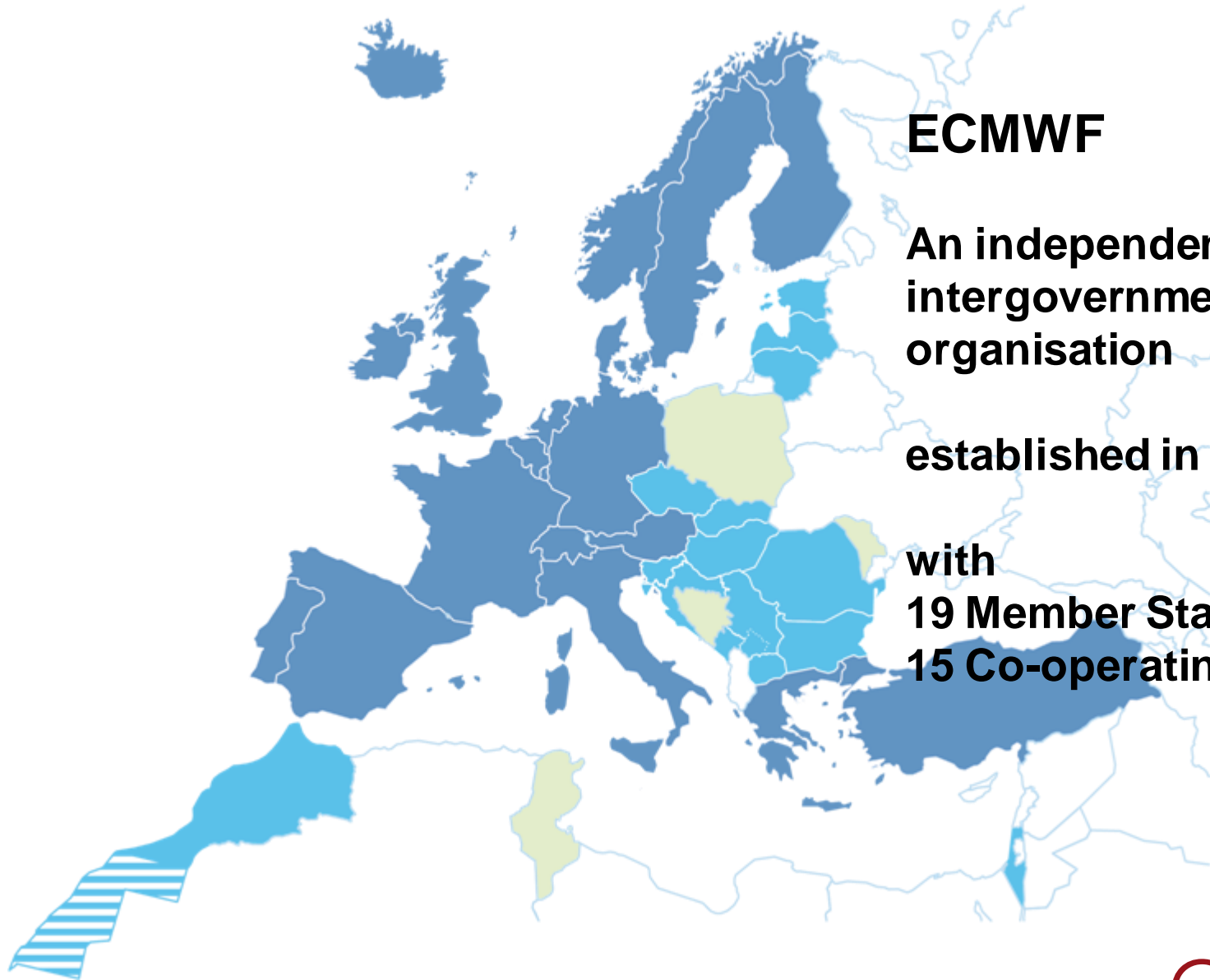
OpenMP for IFS T1279L91 model
on IBM Power6 (~2009)



OpenMP for IFS T1279L91 model on IBM Power6 (~2009)



■ Member States ■ Co-operating States ■ Under negotiation



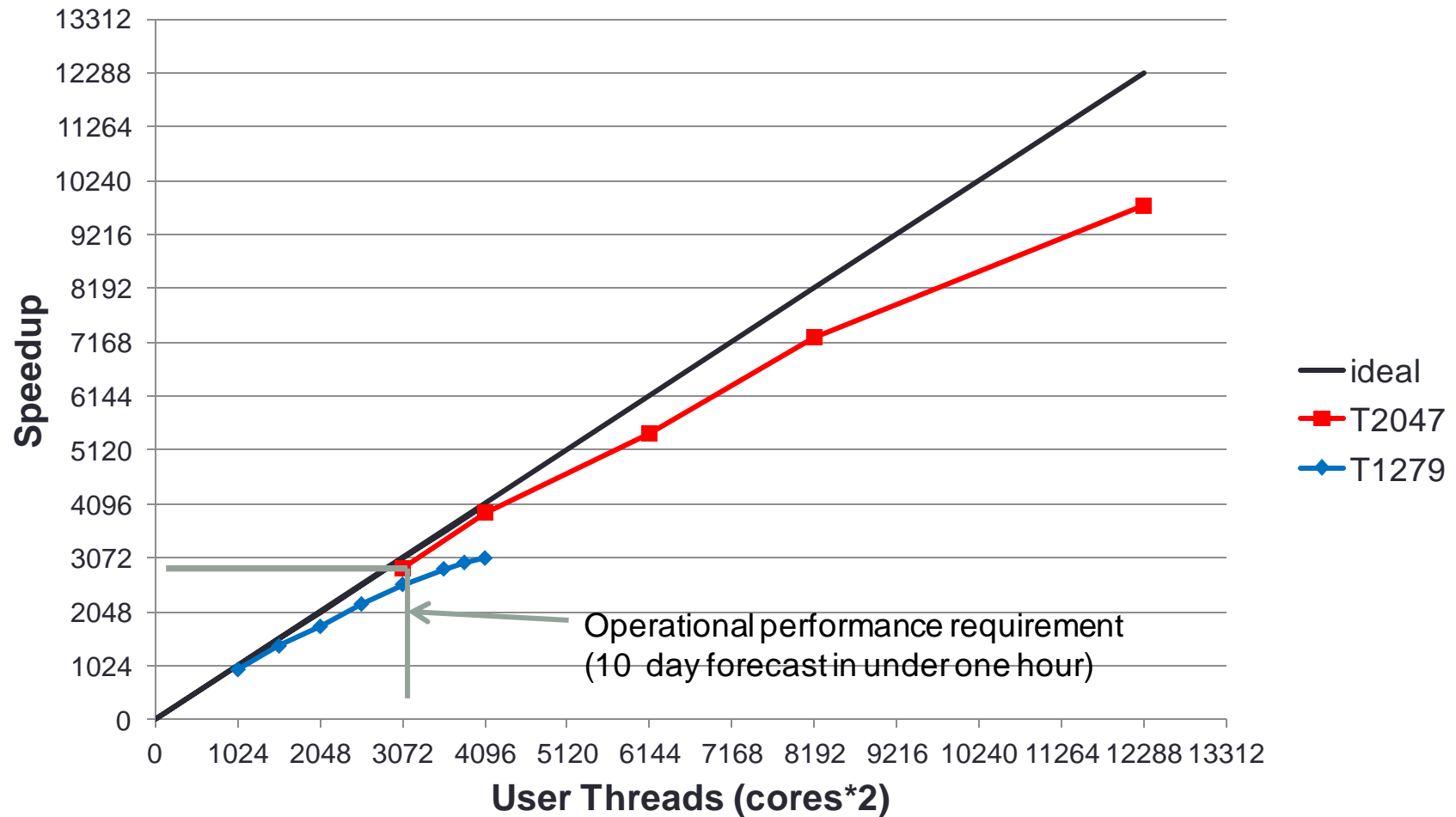
ECMWF

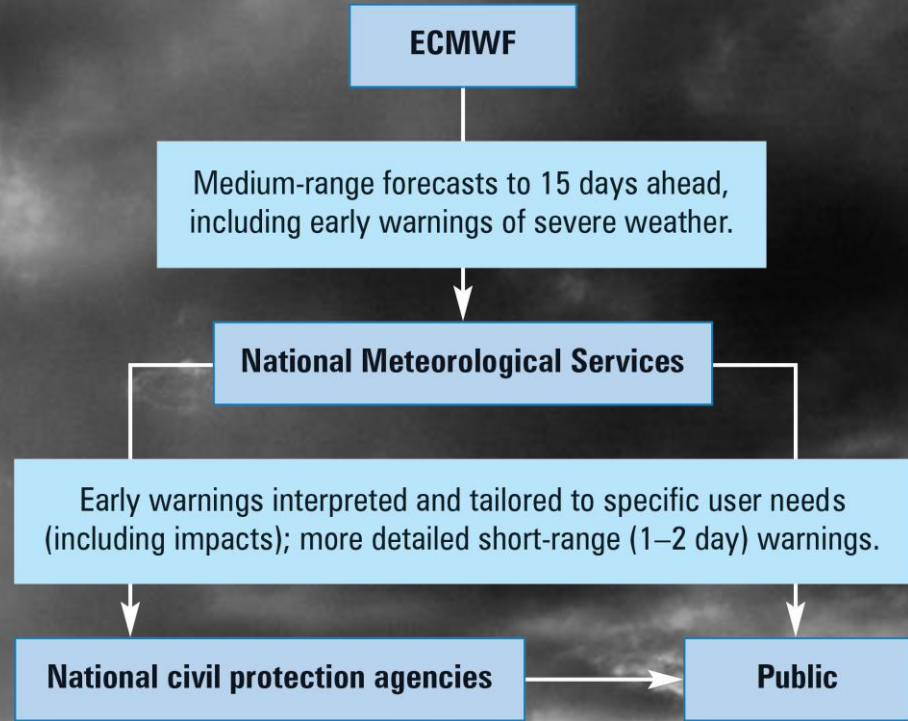
An independent
intergovernmental
organisation

established in 1975

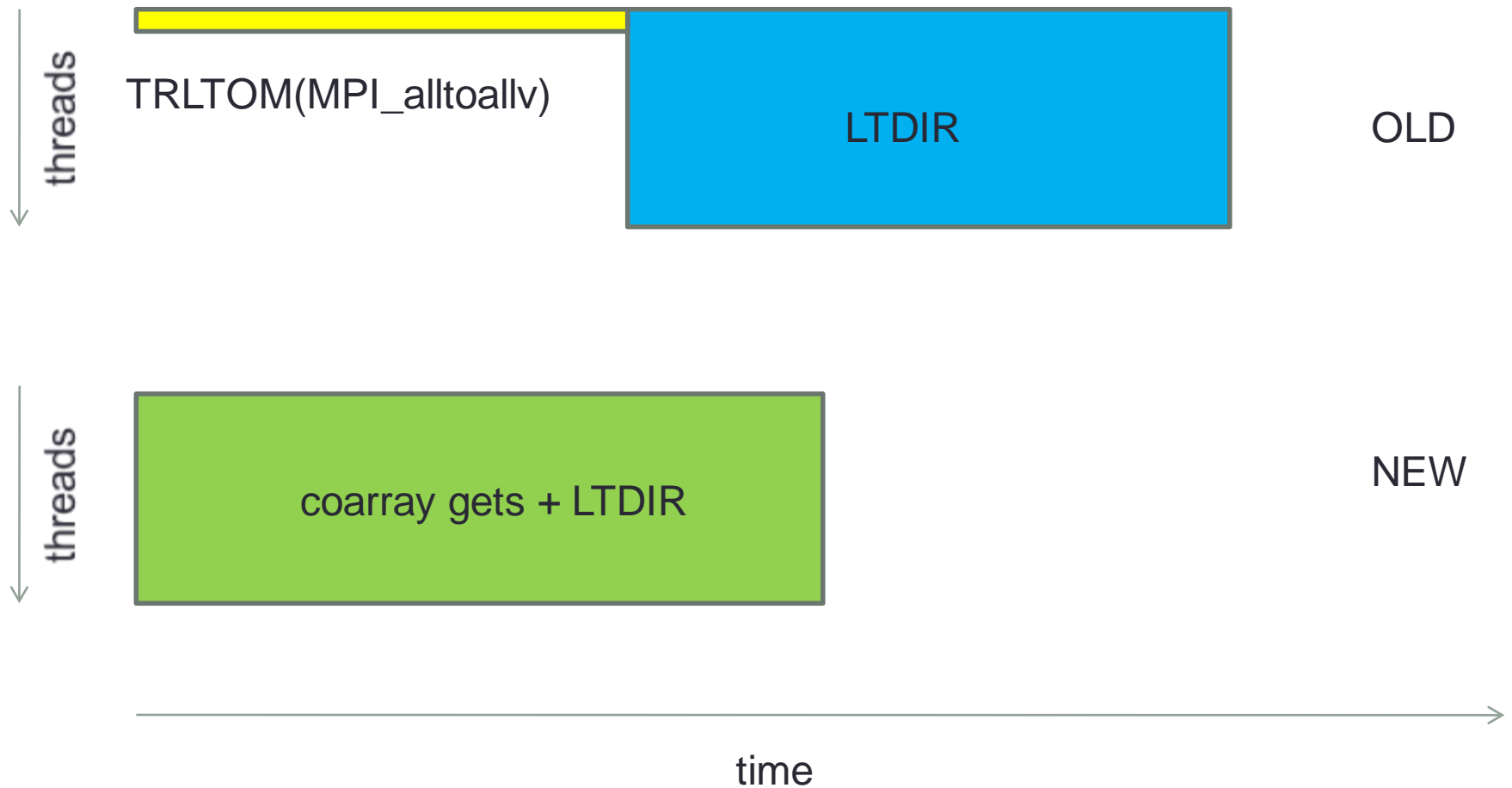
with
19 Member States
15 Co-operating States

IFS model speedup on IBM Power6 (~2010)



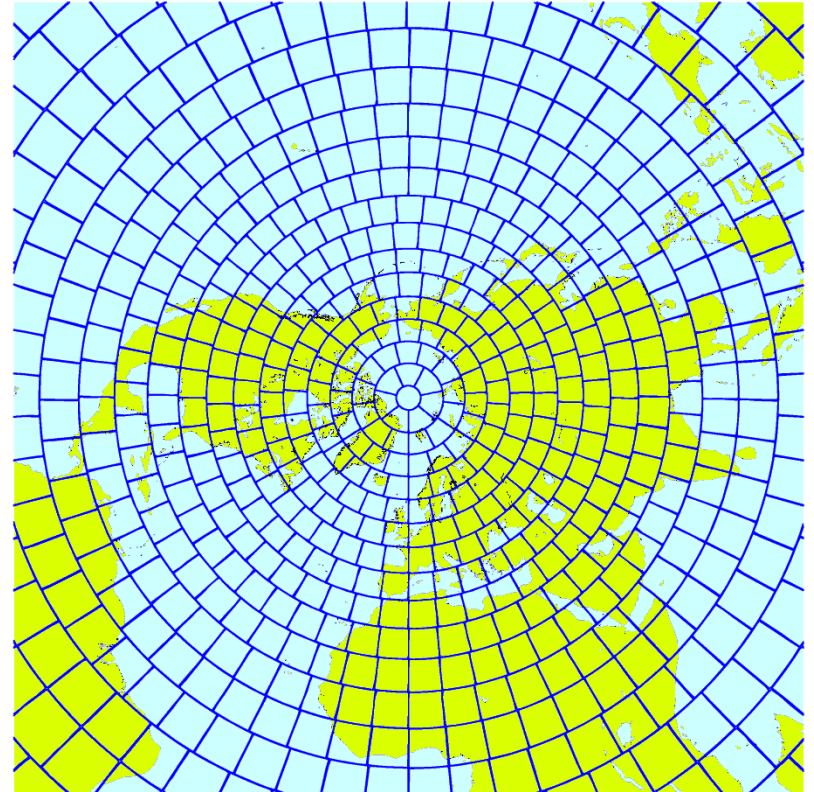
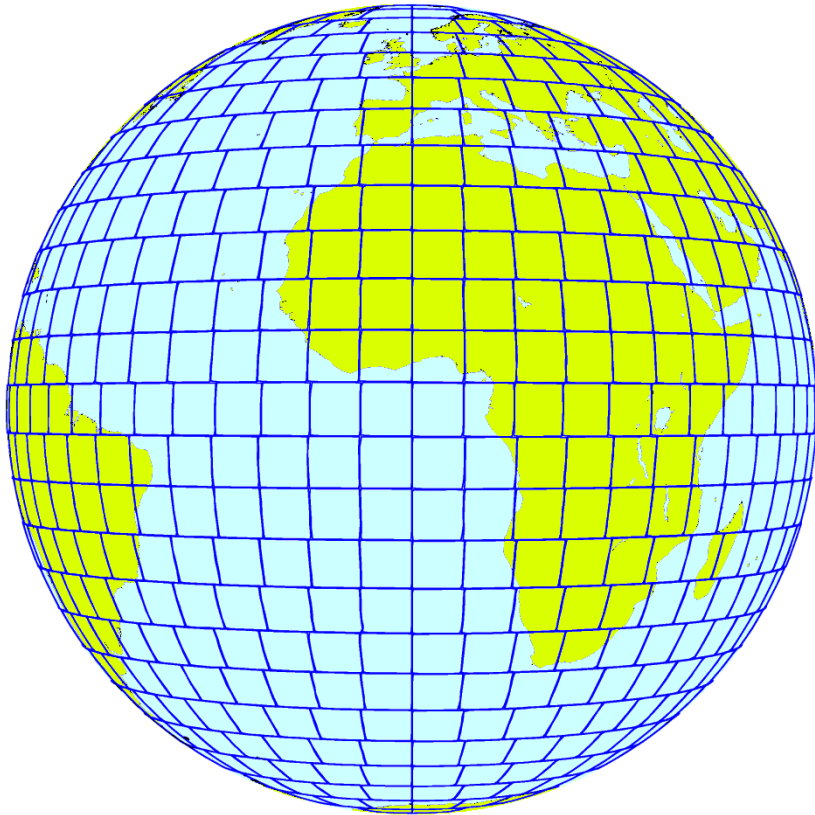


Overlap Legendre transforms with associated transpositions/2



IFS grid point space: “EQ_REGIONS” partitioning for 1024 MPI tasks

Each MPI task has an equal number of grid points



LTINV recoding

COMPUTE COMMUNICATION

```

!$OMP PARALLEL DO SCHEDULE(DYNAMIC,1) PRIVATE(JM,IM,JW,IPE,ILEN,ILENS,IOFFS,IOFFR)
DO JM=1,D%NUMP
  IM = D%MYMS(JM)
  CALL LTINV(IM,JM,KF_OUT_LT,KF_UV,KF_SCALARS,KF_SCDERS,ILEI2,IDIM1,&
    & PSPVOR,PSPDIV,PSPSCALAR ,&
    & PSPSC3A,PSPSC3B,PSPSC2 ,&
    & KFLDPTRUV,KFLDPTRSC,FSPGL_PROC)
ENDDO
!$OMP END PARALLEL DO
DO J=1,NPRTRW
  ILENS(J) = D%NLTSEFTB(J)*IFIELD
  IOFFS(J) = D%NSTAGT0B(J)*IFIELD
  ILENR(J) = D%NLTSGTB(J)*IFIELD
  IOFFR(J) = D%NSTAGT0B(D%MSTABF(J))*IFIELD
ENDDO
CALL MPL_ALLTOALLV(PSENDBUF=FOUBUF_IN,KSEND COUNTS=ILENS,&
  & PRECVBUF=FOUBUF,KRECV COUNTS=ILENR,&
  & KSEND DISPL=IOFFS,KRECV DISPL=IOFFR,&
  & KCOMM=MPL_ALL_MS_COMM,CDSTRING='TRMTOL:')

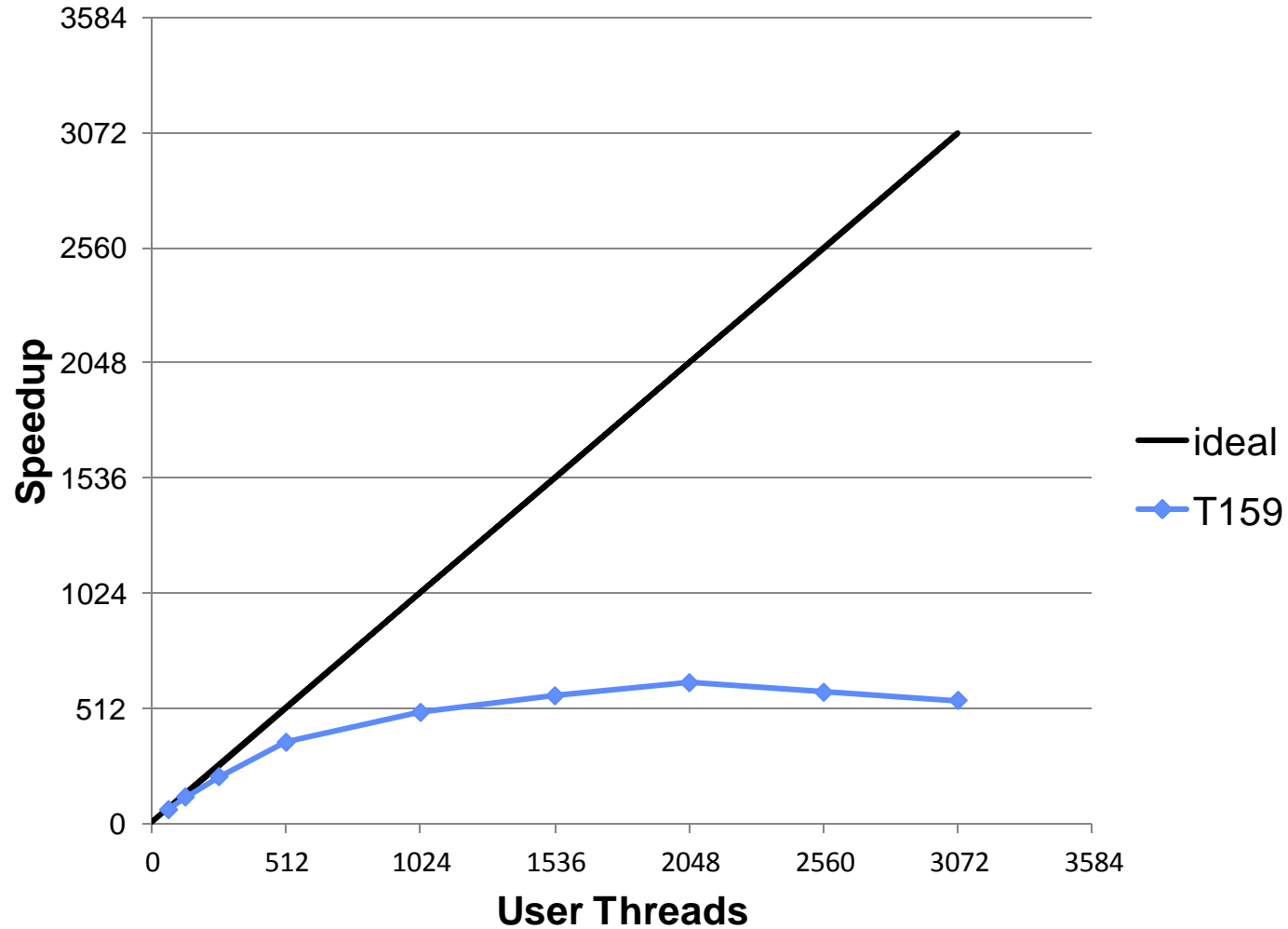
!$OMP PARALLEL DO SCHEDULE(DYNAMIC,1) PRIVATE(JM,IM,JW,IPE,ILEN,ILENS,IOFFS,IOFFR)
DO JM=1,D%NUMP
  IM = D%MYMS(JM)
  CALL LTINV(IM,JM,KF_OUT_LT,KF_UV,KF_SCALARS,KF_SCDERS,ILEI2,IDIM1,&
    & PSPVOR,PSPDIV,PSPSCALAR ,&
    & PSPSC3A,PSPSC3B,PSPSC2 ,&
    & KFLDPTRUV,KFLDPTRSC,FSPGL_PROC)
DO JW=1,NPRTRW
  CALL SET2PE(IPE,0,0,JW,MYSETV)
  ILEN = D%NLEN_M(JW,1,JM)*IFIELD
  IF(ILEN > 0) THEN
    IOFFS = (D%NSTAGT0B(JW)+D%NOFF_M(JW,1,JM))*IFIELD
    IOFFR = (D%NSTAGT0BW(JW,MYSETV)+D%NOFF_M(JW,1,JM))*IFIELD
    FOUBUF_C(IOFFR+1:IOFFR+ILEN)[IPE]=FOUBUF_IN(IOFFS+1:IOFFS+ILEN)
  ENDIF
  ILENS = D%NLEN_M(JW,2,JM)*IFIELD
  IF(ILENS > 0) THEN
    IOFFS = (D%NSTAGT0B(JW)+D%NOFF_M(JW,2,JM))*IFIELD
    IOFFR = (D%NSTAGT0BW(JW,MYSETV)+D%NOFF_M(JW,2,JM))*IFIELD
    FOUBUF_C(IOFFR+1:IOFFR+ILENS)[IPE]=FOUBUF_IN(IOFFS+1:IOFFS+ILENS)
  ENDIF
ENDDO
ENDDO
!$OMP END PARALLEL DO
SYNC IMAGES(D%NMYSETV)
FOUBUF(1:IBLEN)=FOUBUF_C(1:IBLEN)[MYPROC]

```

ORIGINAL
code

NEW
code

T159 model scaling: small model with 'large' number of user threads (4 threads per task)



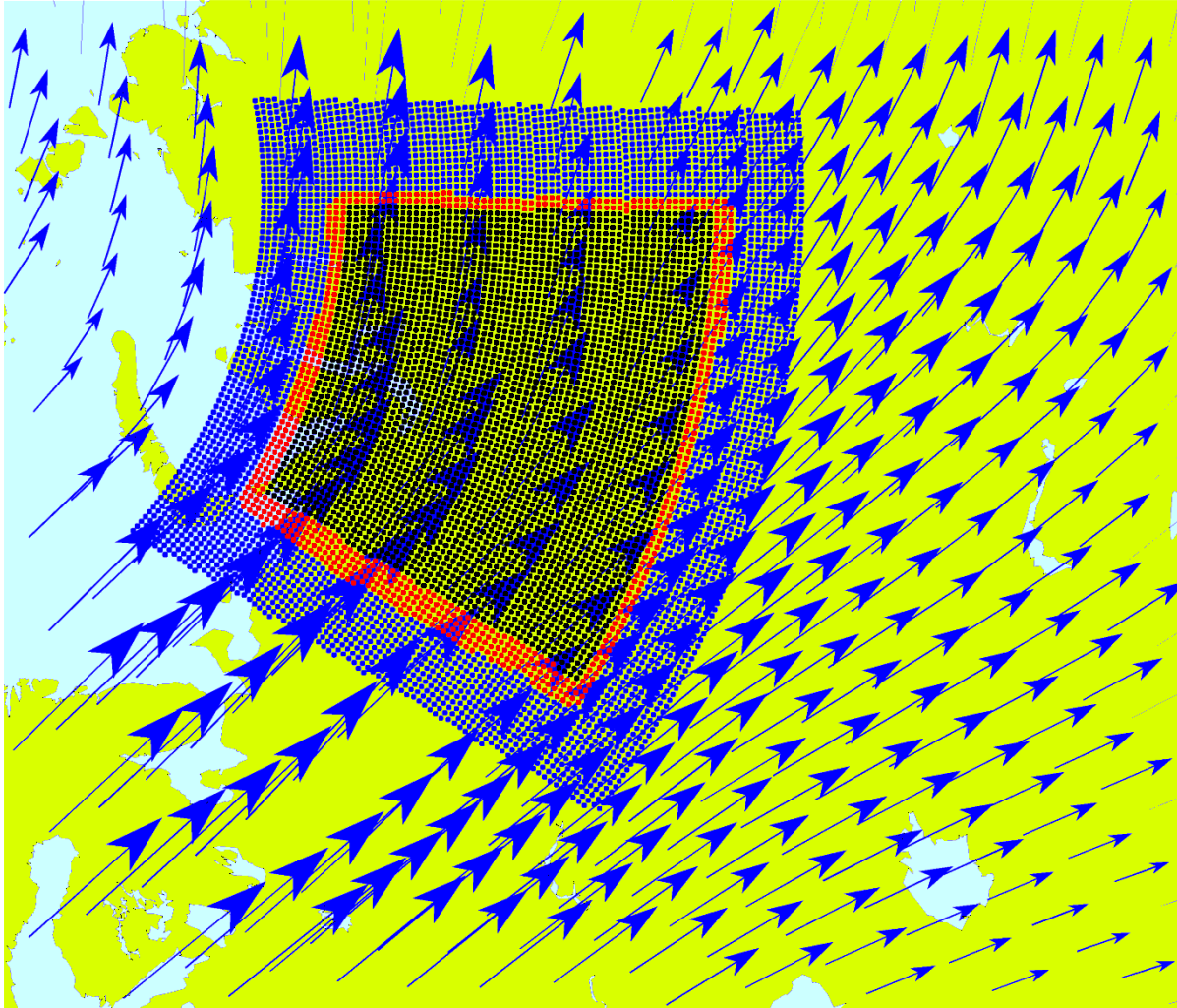
IFS Semi-Lagrangian Comms

- **SL comms scaling limited by**
 - constant width halo for u,v,w (400 m/s x time step)
 - Halo volume communicated, which is a function of wind speed and direction in locality of each task
- **'Halo-lite' approach tested (2010)**
 - Only get (using MPI) grid columns from neighbouring tasks that your task needs, i.e. only the red points
 - Requires more MPI communication steps (e.g. mid-point, departure point)
 - No faster than original approach due to overheads of above
- **CRESTA optimisation using F2008 coarrays (2012)**
 - Only get grid columns from neighbouring tasks that your task needs, i.e. only the red points
 - Do the above in the context of an OpenMP parallel region; overlapping interpolations for determining the departure point & mid-point and interpolations at these points

wind plot

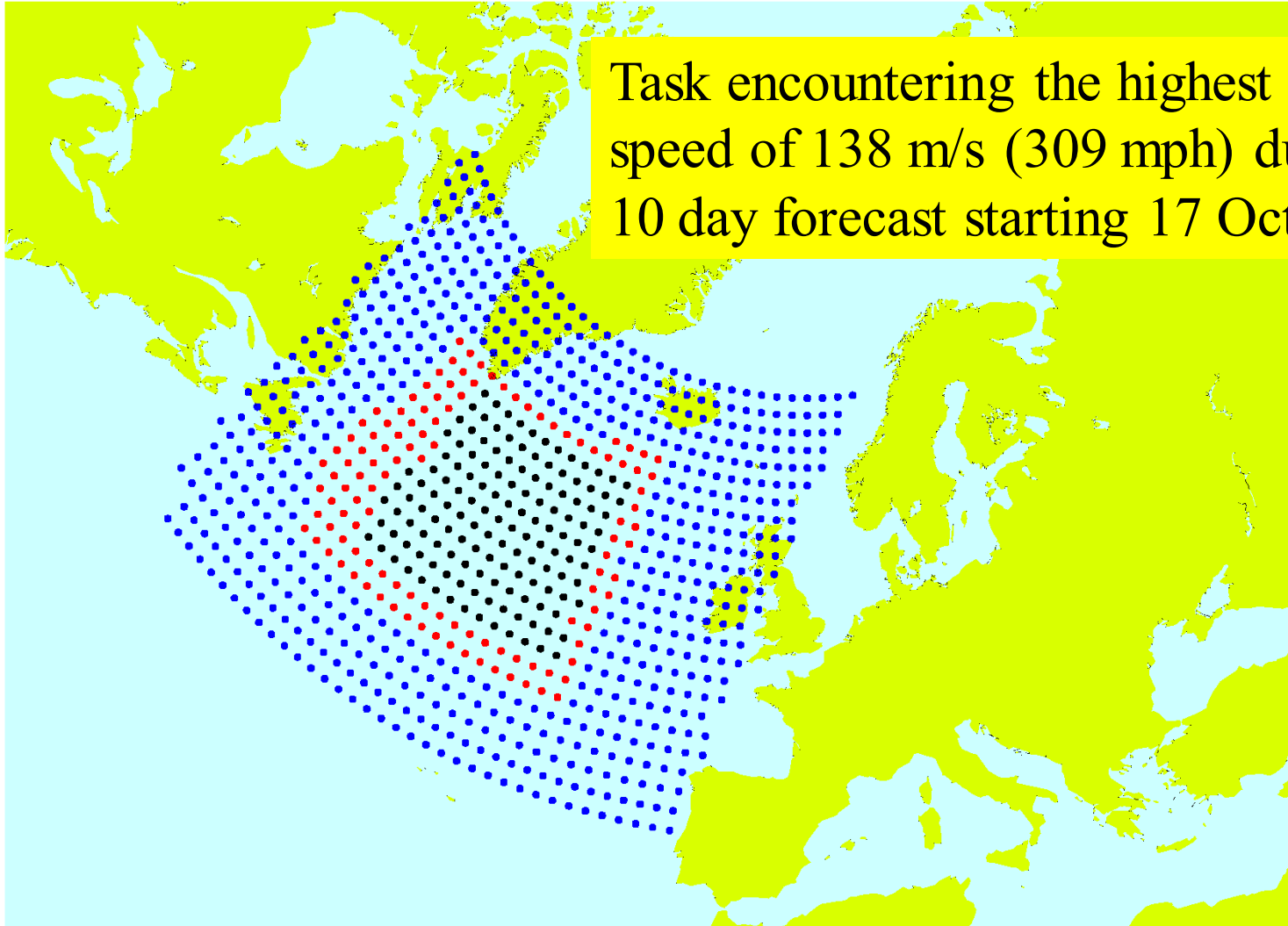
Friday 15 October 2004 12UTC ECMWF Forecast t+0 VT: Friday 15 October 2004 12UTC Model Level 1 U velocity/V velocity

25.0m/s

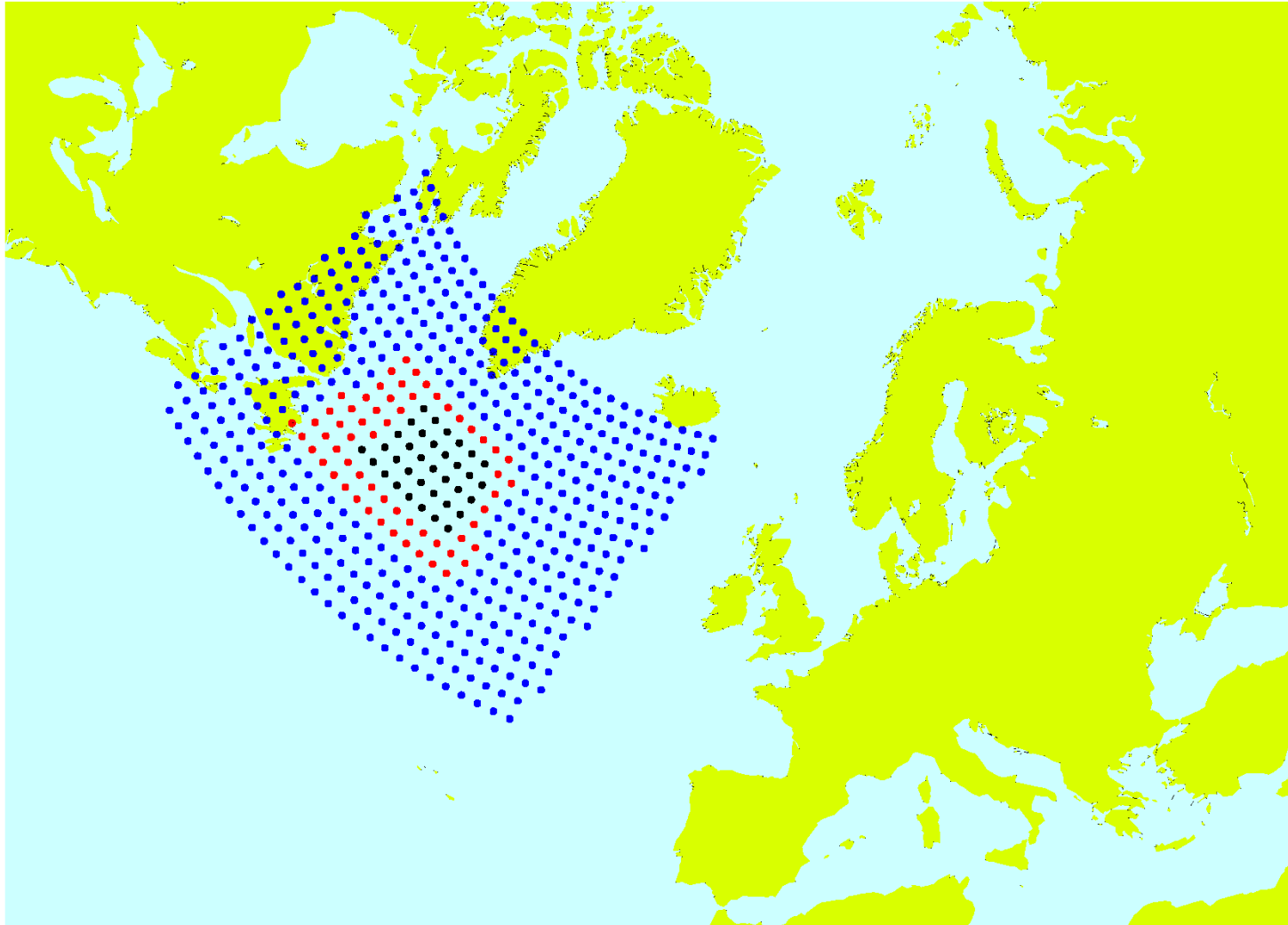


T159 model task 37 of 256 tasks

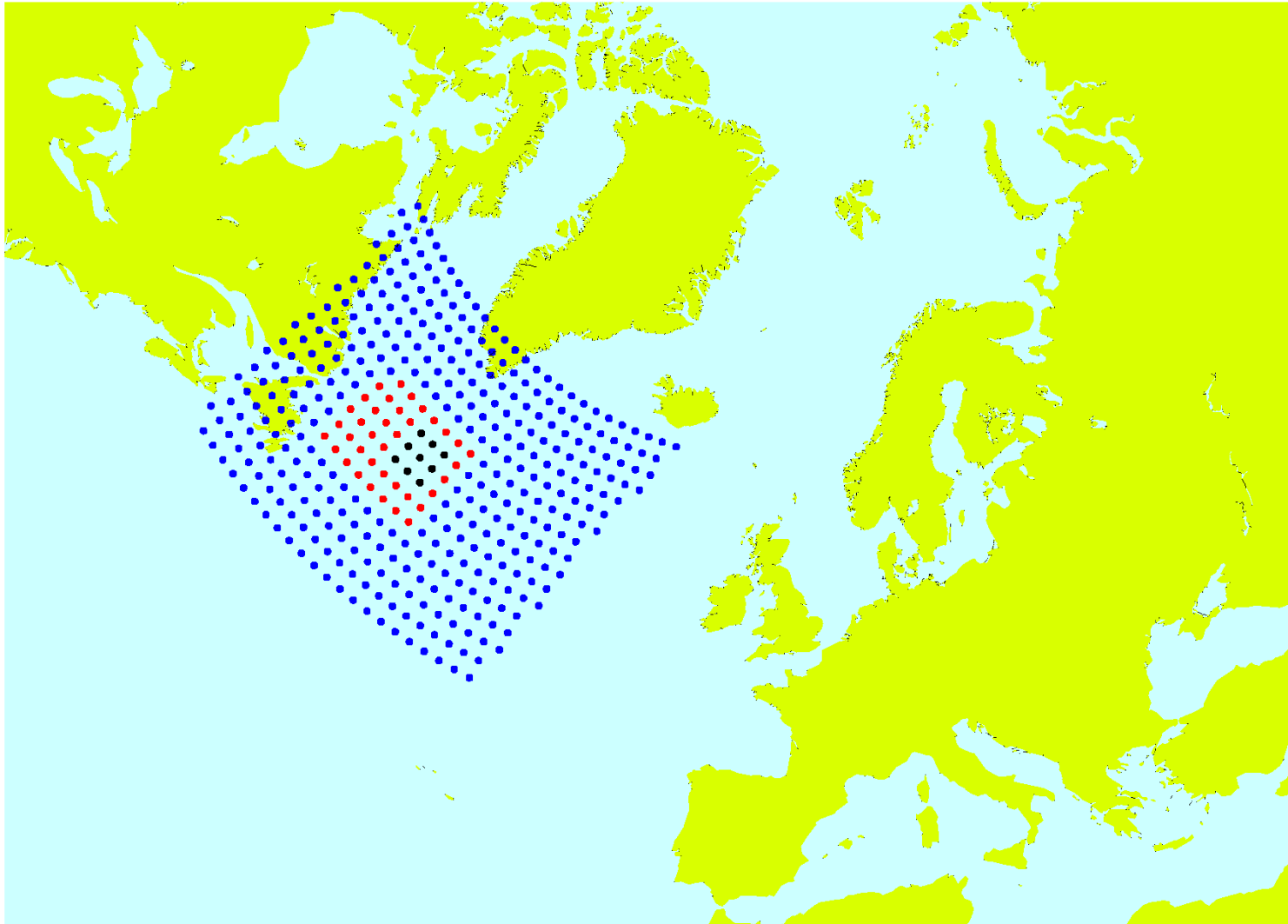
Task encountering the highest wind speed of 138 m/s (309 mph) during a 10 day forecast starting 17 Oct 2010



T159 model task 128 of 1024 tasks



T159 model task 462 of 4096 tasks



IFS Introduction – A history

- Resolution increases of the deterministic 10-day medium-range Integrated Forecast System (IFS) over ~25 years at ECMWF:
 - 1987: T 106 (~125km)
 - 1991: T 213 (~63km)
 - 1998: T_L 319 (~63km)
 - 2000: T_L 511 (~39km)
 - 2006: T_L 799 (~25km)
 - 2010: T_L 1279 (~16km)

Introduction – A history

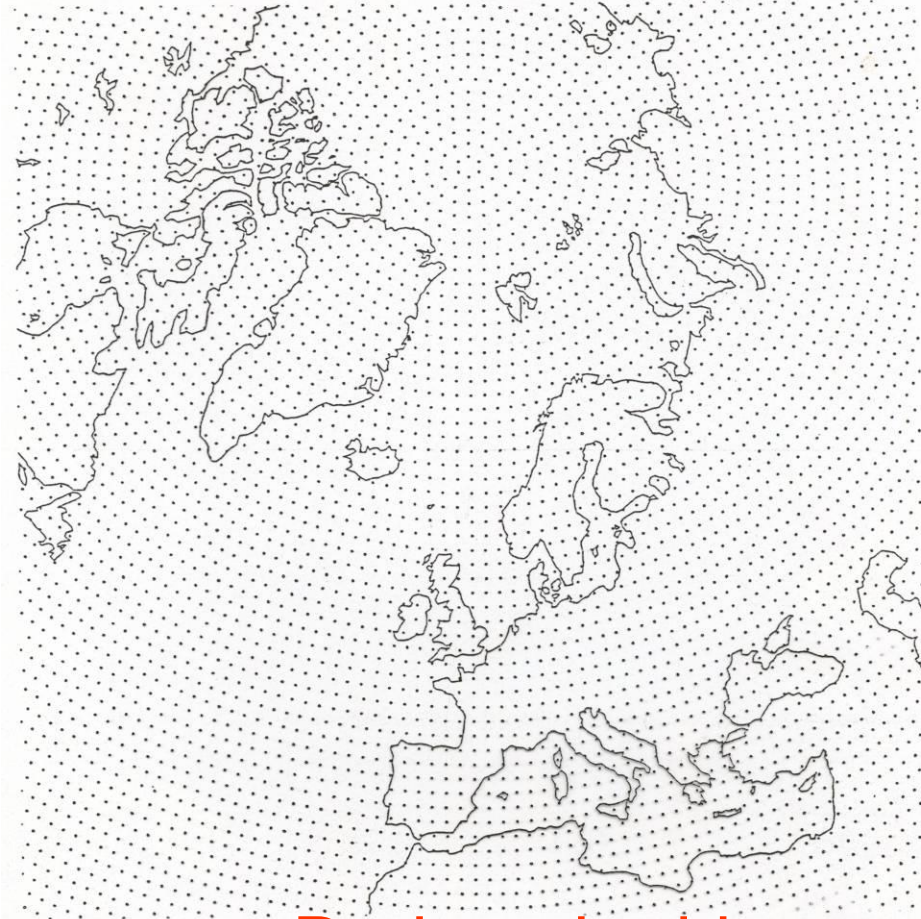
- Resolution increases of the deterministic 10-day medium-range Integrated Forecast System (IFS) over ~25 years at ECMWF:
 - 1987: T 106 (~125km)
 - 1991: T 213 (~63km)
 - 1998: T_L 319 (~63km)
 - 2000: T_L 511 (~39km)
 - 2006: T_L 799 (~25km)
 - 2010: T_L 1279 (~16km)
 - **2015?**: T_L 2047 (~10km)
 - **2020-???**: (~1-10km) Non-hydrostatic, cloud-permitting, substantially different cloud-microphysics and turbulence parametrization, substantially different dynamics-physics interaction ?

The Gaussian grid

About 30% reduction in number of points



Full grid



Reduced grid

Reduction in the number of Fourier points at high latitudes is possible because associated Legendre functions are very small near the poles for large m .



(Adaptive) Mesh Refinement

- The IFS model is **inherently based on a fixed structured mesh** due to the link between the spectral representation and the position of the grid-points (zero's of the ordinary Legendre polynomials), which makes selective mesh refinement (adaptive or not) difficult to achieve.
- “AMR” possibilities: coexisting global multigrids, physics/ dynamics on different grids, wavelet-collocation methods, ...: Costly investment both in RD and computational cost
- Hence it is of strategic importance to **understand the added-value of adaptive or static mesh refinement** for multiscale global NWP and climate prediction !

Nonhydrostatic IFS (NH-IFS)

Bubnová et al. (1995); Bénard et al. (2004), Bénard et al. (2005), Bénard et al. (2010), Wedi et al. (2009), Yessad and Wedi (2011)

- **Arpégé/ALADIN/Arome/HIRLAM/ECMWF nonhydrostatic dynamical core, which was developed by Météo-France and their ALADIN partners and later incorporated into the ECMWF model and also adopted by HIRLAM.**

Numerical solution

- Two-time-level, semi-implicit, semi-Lagrangian.
- Semi-implicit procedure with two reference states, with respect to gravity and acoustic waves, respectively.
- The resulting **Helmholtz equation** can be solved (subject to some constraints on the vertical discretization) with a **direct spectral method**, that is, a mathematical separation of the horizontal and vertical part of the linear problem in spectral space, with the remainder representing at most a pentadiagonal problem of dimension $NLEV^2$. Non-linear residuals are treated explicitly (or iteratively implicitly)!

(Robert, 1972; Bénard et al 2004,2005,2010)

The spectral transform method

Eliassen et. al (1970), Orszaag (1970)

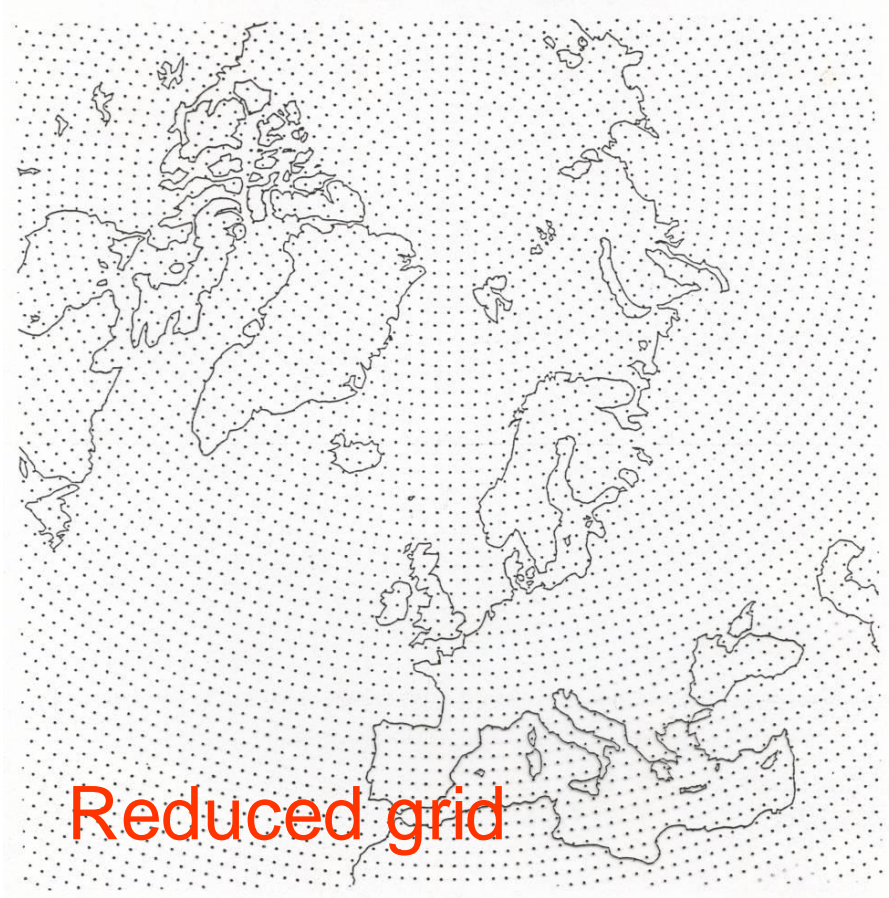
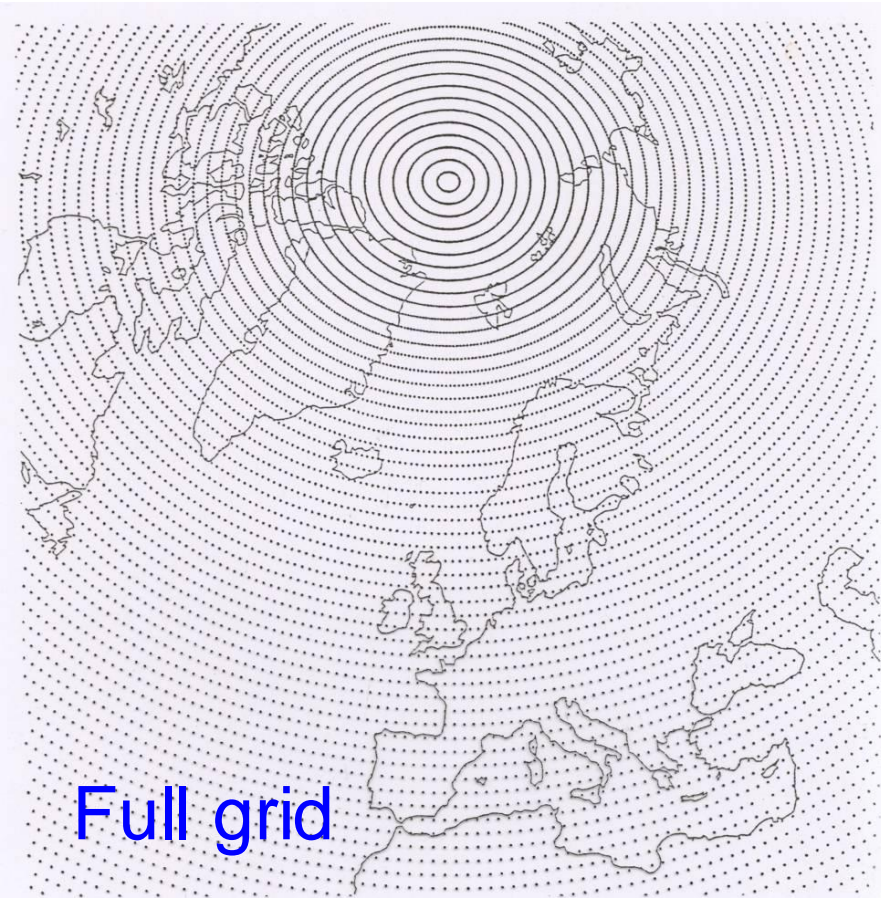
Applied at ECMWF for the last 30 years ...

Spectral semi-Lagrangian semi-implicit
(compressible) a viable option ?

- Computational efficiency on future MPP architectures ?
- Accuracy at cloud-resolving scales ?
- Suitability for the likely mixture of medium and high resolution ensembles and ultra-high resolution forecasts ?

The Gaussian grid

About 30% reduction in number of points

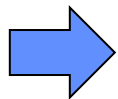


Reduction in the number of Fourier points at high latitudes is possible because associated Legendre polynomials are very small near the poles for large m .

Note: number of points nearly equivalent to quasi-uniform icosahedral grid cells of the ICON model.

Cost of the spectral transform method

- FFT can be computed as $C*N*\log(N)$ where C is a small positive number and N is the cut-off wave number in the triangular truncation.
- Ordinary Legendre transform is $O(N^2)$ but can be combined with the fields/levels such that the arising matrix-matrix multiplies make use of the highly optimized BLAS routine DGEMM.
- But overall cost is $O(N^3)$ for both memory and CPU time requirements.



Desire to use a fast Legendre transform where the cost is proportional to $C*N*\log(N)$ with $C \ll N$

and thus overall cost $N^2*\log(N)$

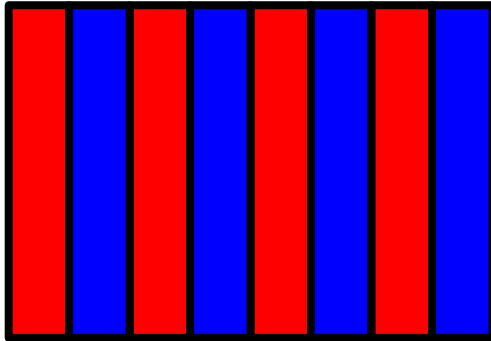
Fast Legendre Transform (FLT)

- The algorithm proposed in *(Tygert, 2008,2010)* suitably fits into the IFS transform library by simply replacing the matrix-matrix multiply DGEMM call with a BUTTERFLY_MATRIX_MULT call plus slightly more expensive pre-computations.
- (1) Instead of the recursive *Cuppen divide-and-conquer algorithm (Tygert, 2008)* we use the so called *butterfly algorithm (O'Neil et al, 2009; Tygert, 2010)* based on a matrix compression technique via rank reduction with a specified accuracy to accelerate the arising *matrix-vector/matrix multiplies (sub-problems still use DGEMM)*.
- (2) We apply the matrix compression directly on the matrix of the associated polynomials, which reduces the required precomputations and **eliminates the need** to apply *FMM (fast multipole method)* accelerated interpolations. Notably, the latter were an essential part of the proposed FLT in *Suda and Takami (2001)*.

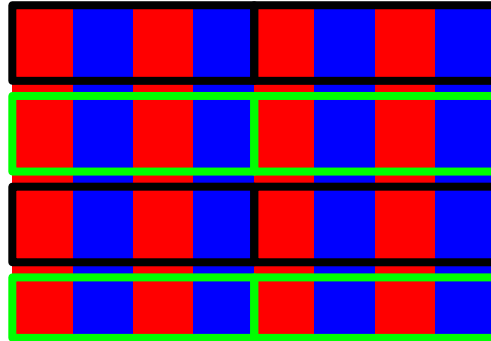
The butterfly compression

(O'Neil, Woolfe, Rokhlin, 2009; Tygert 2010)

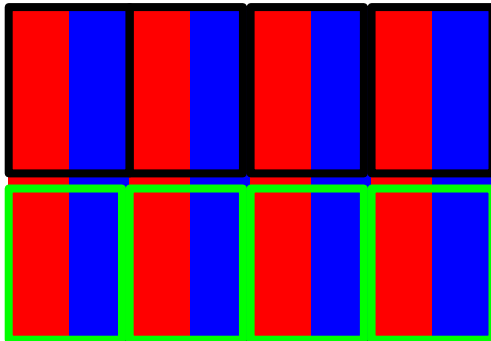
With each level l ,
double the columns
and half the rows



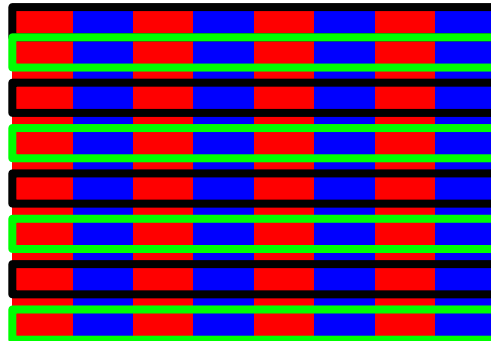
$l=0$



$l=2$



$l=1$

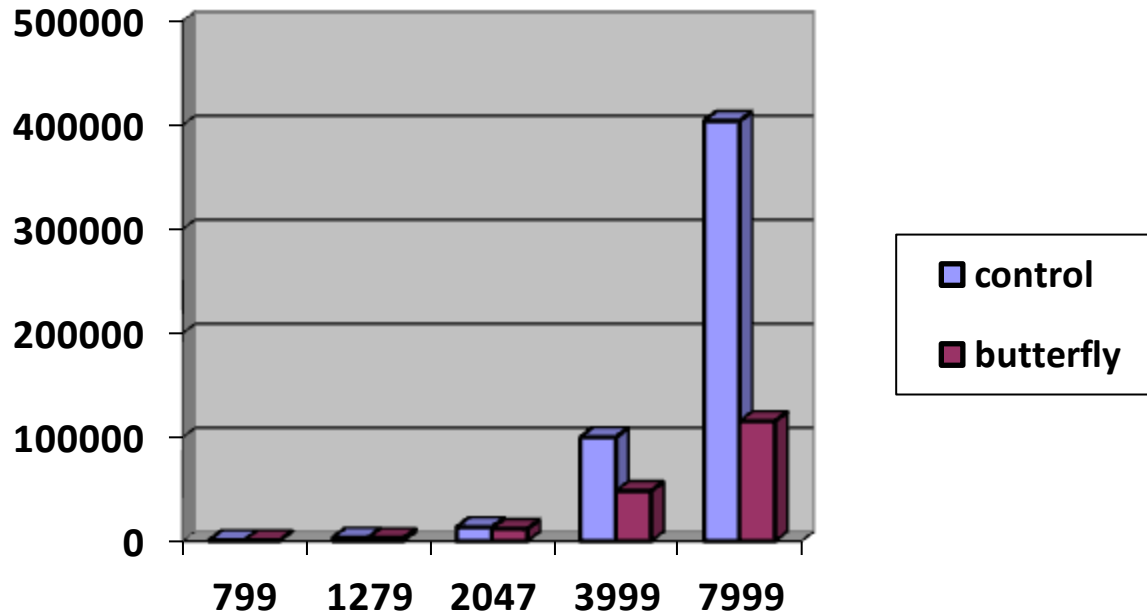


$l=3$



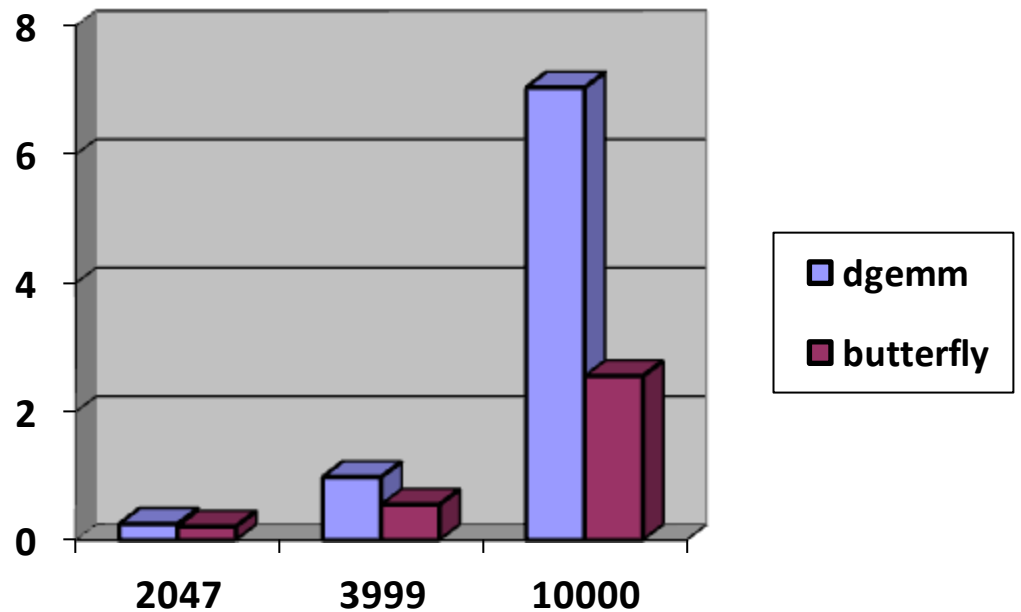
Floating point operations
per time-step in Gflop

**Inverse transform of
single field/level**



Wallclock time in seconds

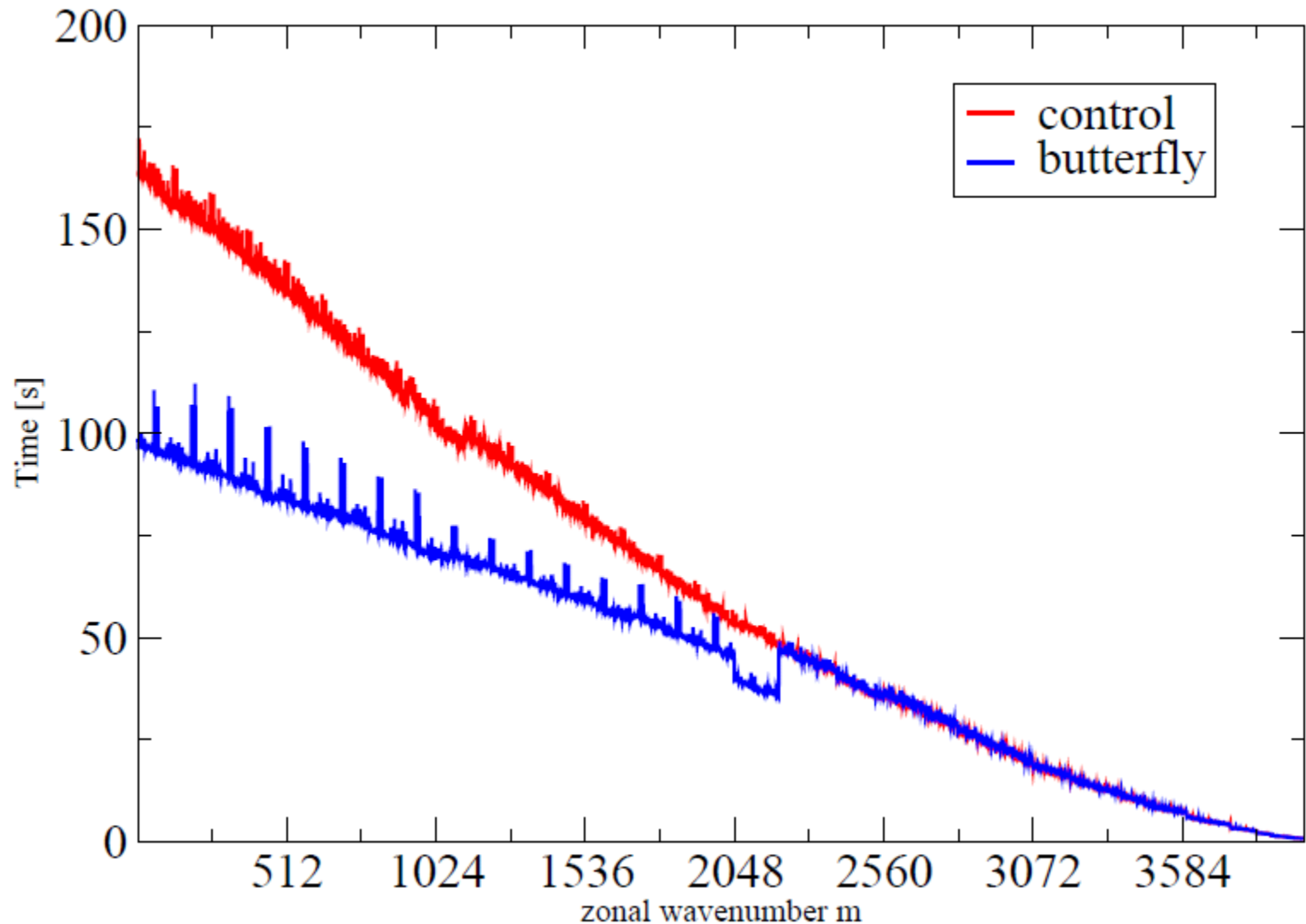
**inverse transform of 10 fields,
offline test environment**



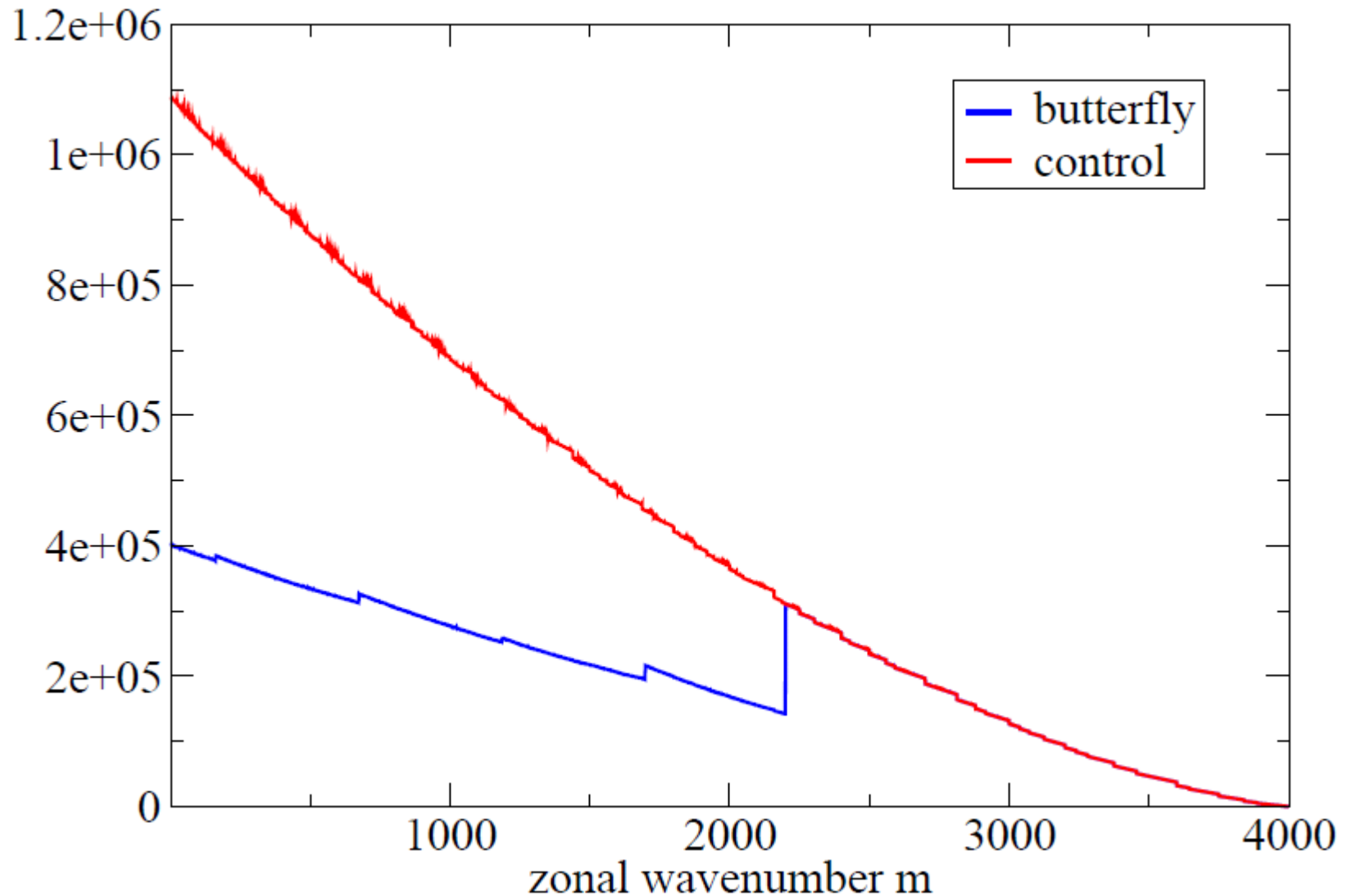
Selected projects to prepare for exascale computing in Meteorology (NWP)

- ***ICOMEX*** – *ICOsahedral* grid Models for *EXascale* Earth system simulations (2011-2014)
- **Gung-Ho** – Development of the Next Generation Dynamical Core for the UK MetOffice (2 phases, 2011-2013, 2013-2016)
- **CRESTA** – Collaborative *Research* into *Exascale* Systemware, *Tools & Applications* (2011-2014)

T3999 6h forecast - inverse transforms: CPU time vs. wave number

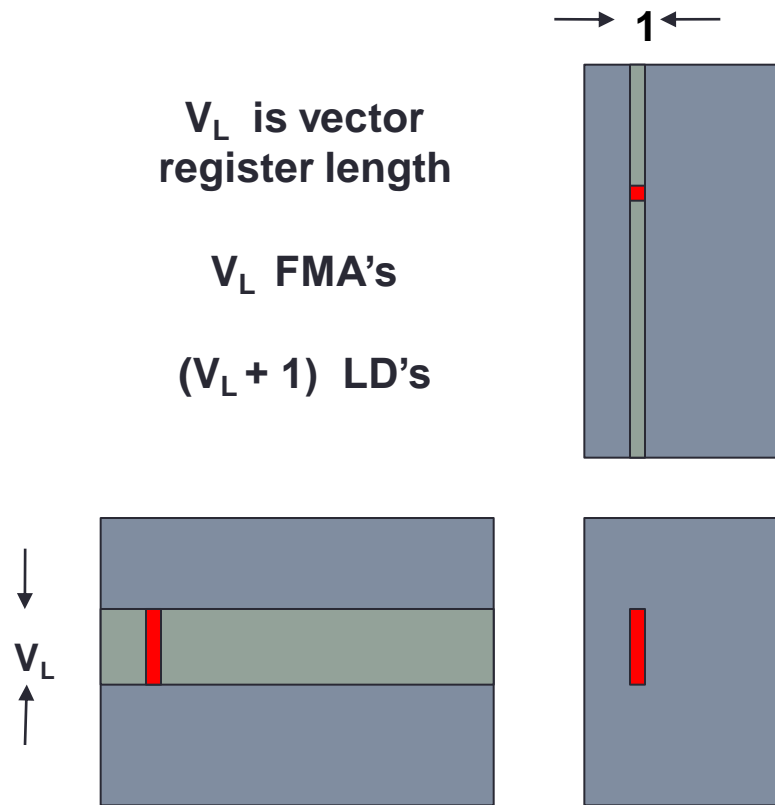


T3999 6h forecast - inverse transforms: Floating point operations vs. wave



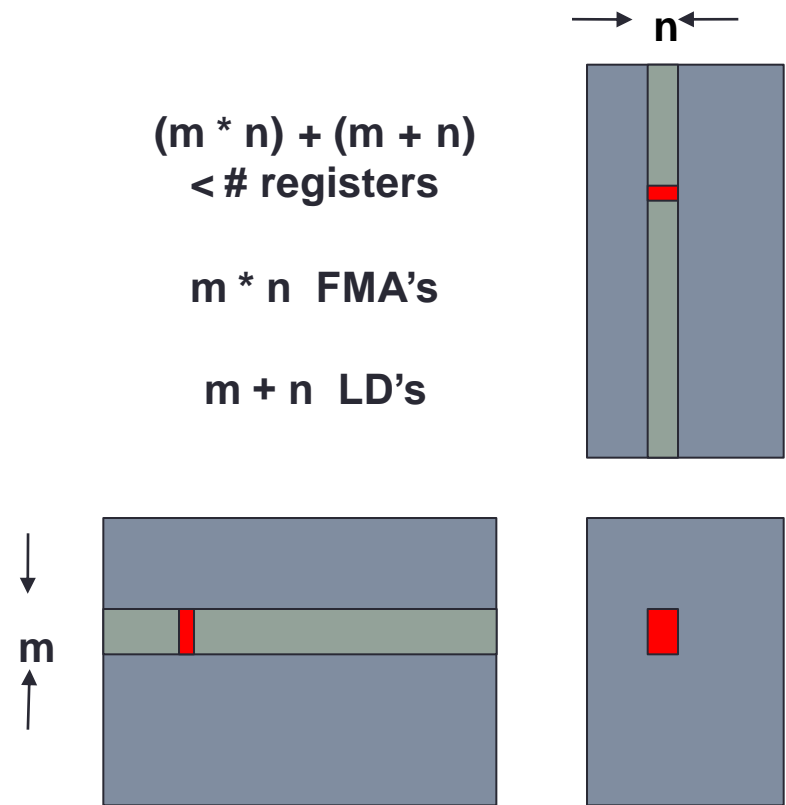
Why is Matrix Multiply (DGEMM) so efficient?

VECTOR



FMA's \approx LD's

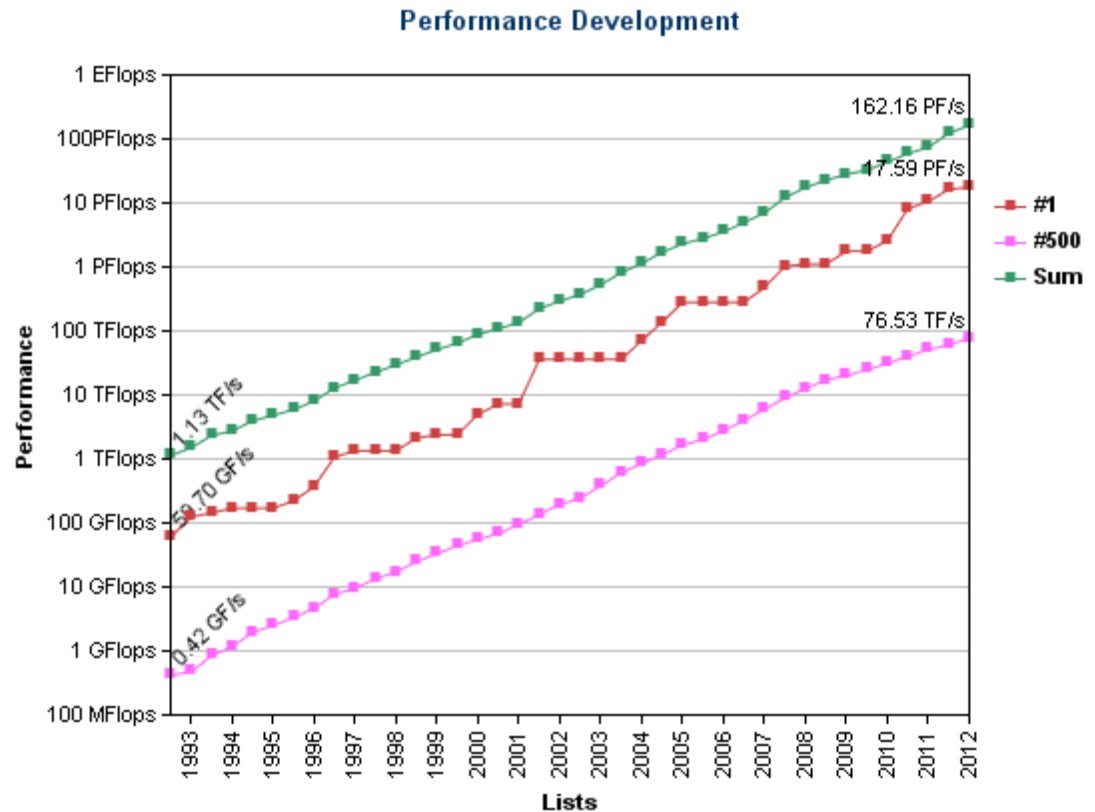
SCALAR / CACHE



FMA's \gg LD's

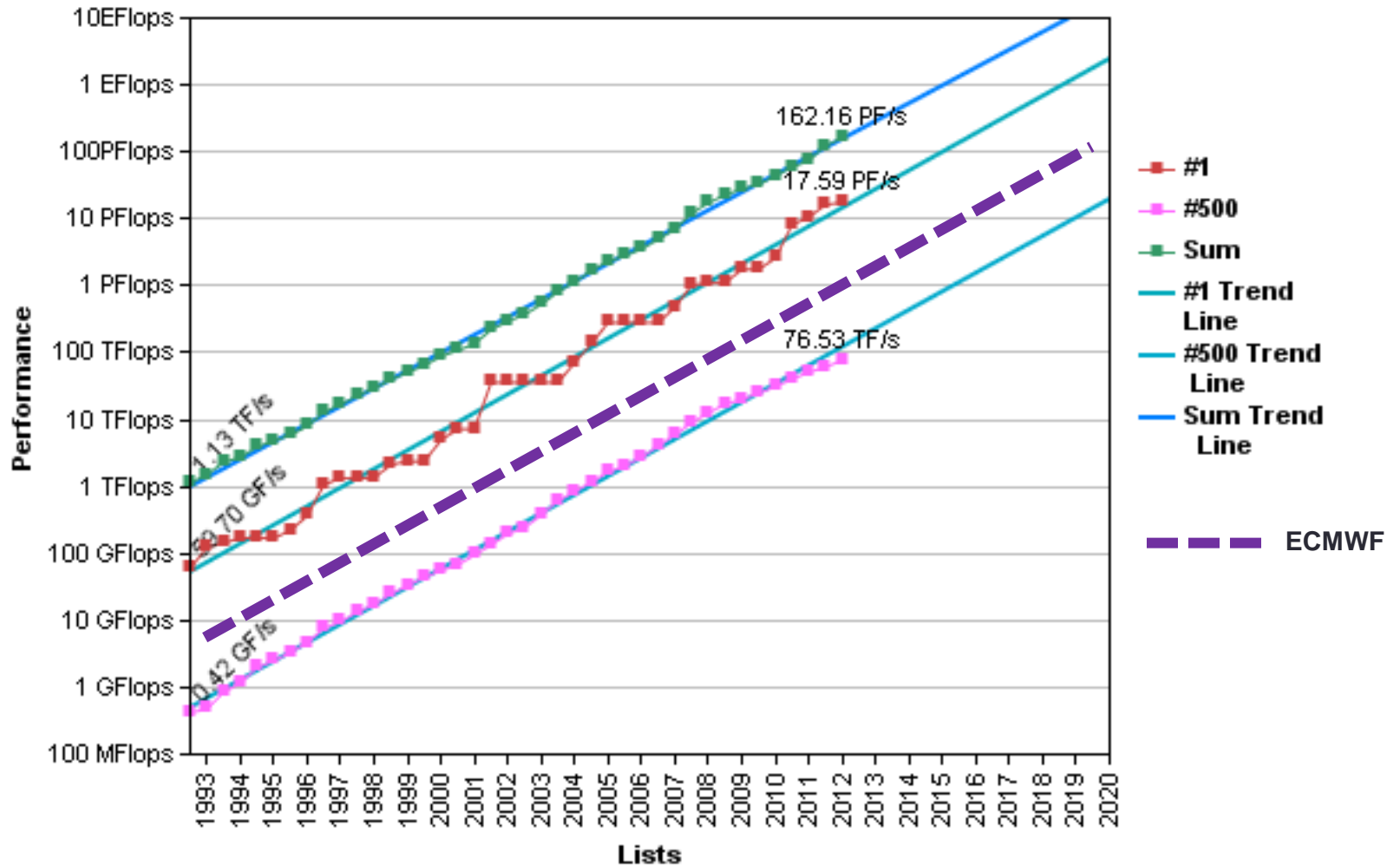
What is Exascale?

Mega	10^6
Giga	10^9
Tera	10^{12}
Peta	10^{15}
Exa	10^{18}
Zetta	10^{21}
Yotta	10^{24}



Source: www.top500.org

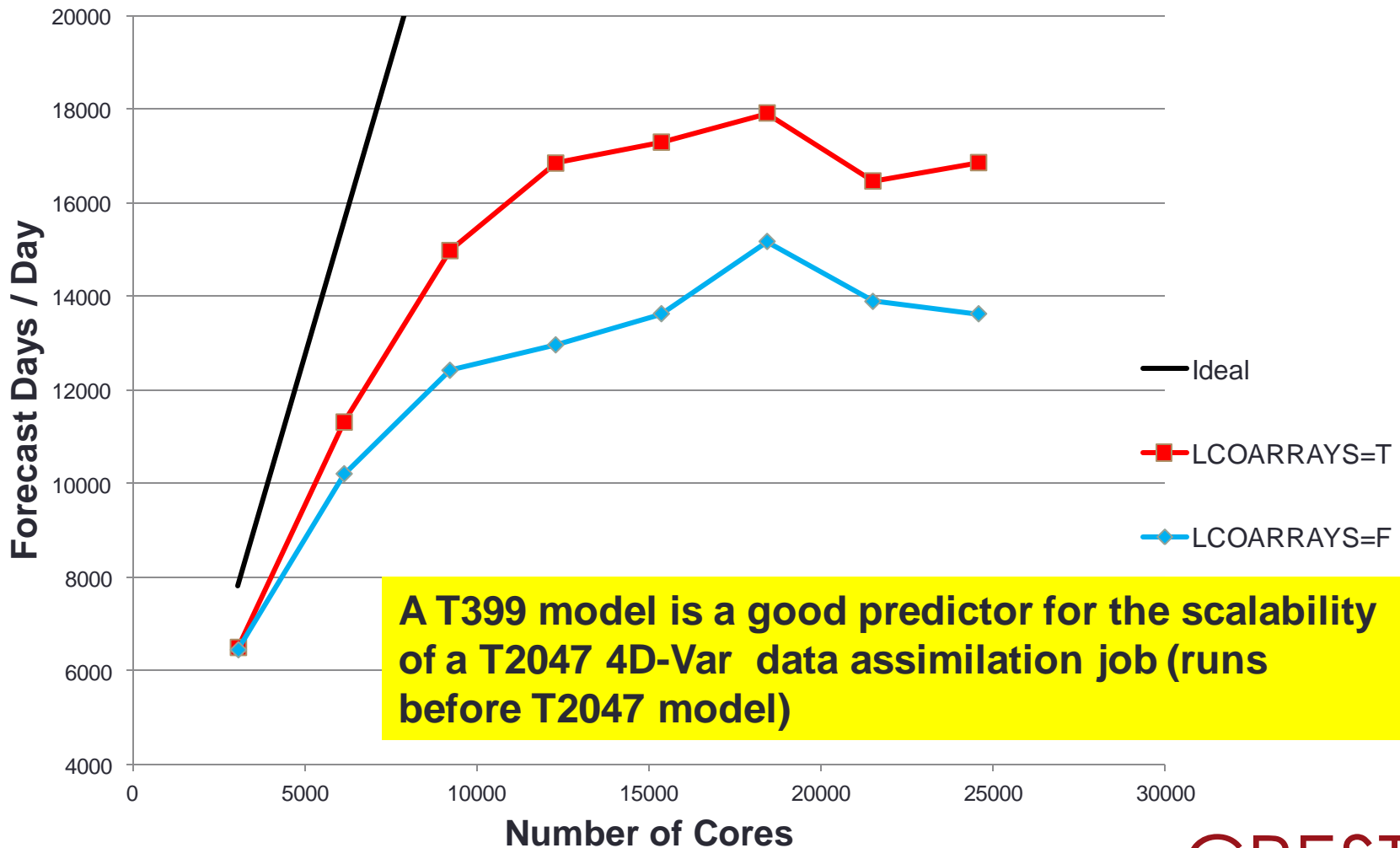
Projected Performance Development



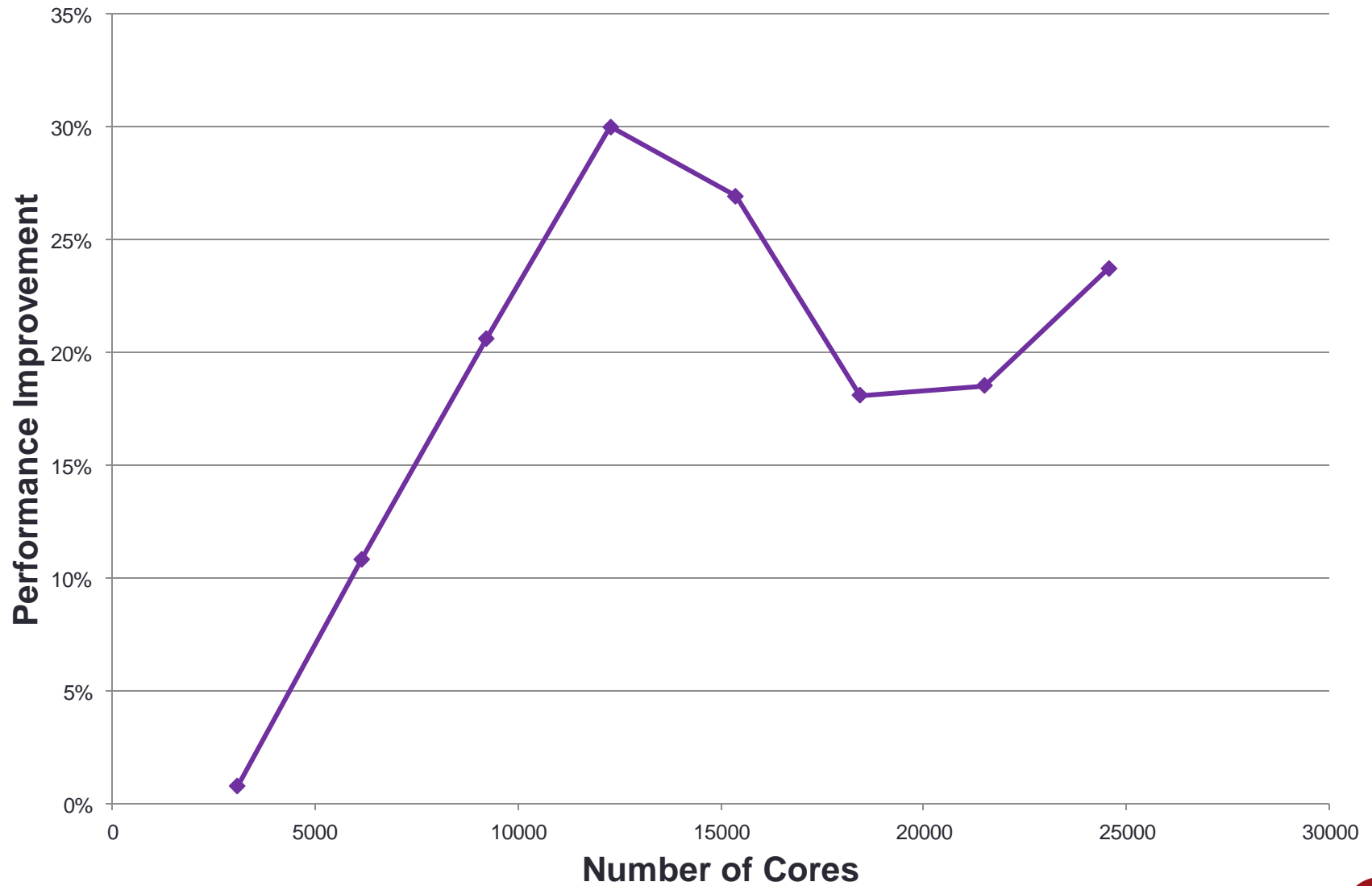
Some of the issues at the Exascale

- Power
 - An Exascale computer today would require about a gigawatt (\$1B per year)
 - 20 megawatt seen as a limit for governments with deep pockets
 - We expect engineers will solve this problem
- Processors are not getting faster
 - They are getting slower
 - But this is more than compensated by their number (e.g. GPGPUs)
- Reliability
 - Uptime for single system ~ 1 day
 - Implies redundancy of nodes, network, filesystem, no single point of failure
- Scalability of applications
 - Incremental / disruptive solutions / new algorithms / I/O
 - More ensemble methods?

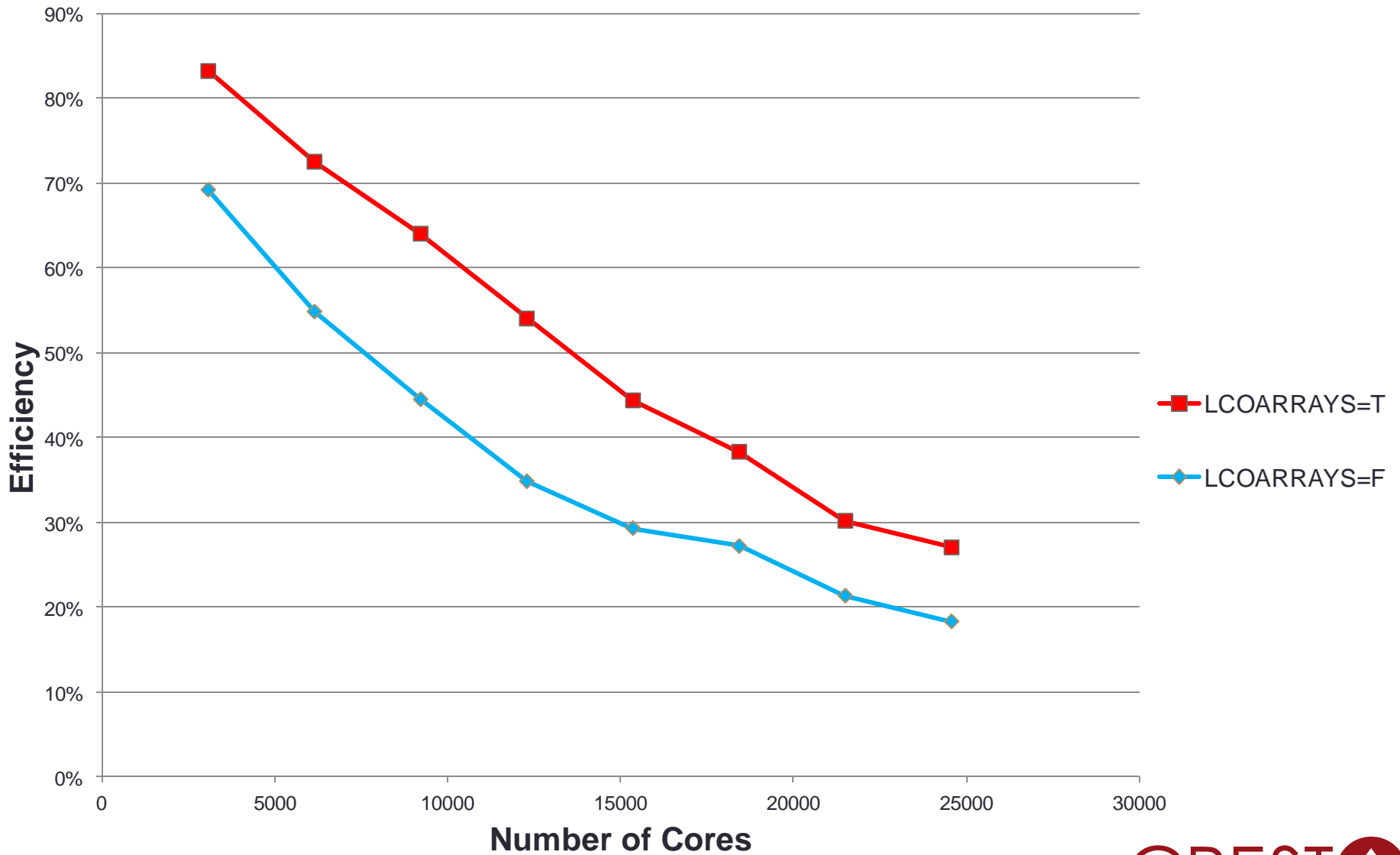
T399L91 model performance on HECToR (CRAY XE6) RAPS12 IFS (CY37R3), cce=8.0.6 -hflex_mp=intolerant



T399L91 IFS model performance improvement by using Fortran2008 coarrays on HECToR (CRAY XE6)



T399L91 model Efficiency on HECToR (CRAY XE6) RAPS12 IFS (CY37R3)



Schedule for IFS optimizations in CRESTA (completed)

When	Activity
4Q2011	Coarray Kernel
1Q2012	IFS CY37R3 port to HECToR (Cray XE6) Run T2047 model at scale and analyze performance
2Q2012	Scalability improvements arising from T2047 analyses (“low hanging fruit”) Overlap Legendre transform computations with associated TRMTOL & TRLTOM transpositions
3Q2012	Semi-Lagrangian optimization Overlap TRGTOL & TRLTOG transpositions with associated Fourier transforms
4Q2012	Optimization of spectral semi-implicit computations for non-hydrostatic model