Challenges of getting ECMWF's weather forecast model (IFS) to the Exascale

George Mozdzynski, Willem Deconinck and Mats Hamrud

Acknowledgements

Mats Hamrud	ECMWF
Nils Wedi	ECMWF
Willem Deconinck	ECMWF
Jens Doleschal	Technische Universität Dresden
Harvey Richardson	Cray UK
Alistair Hart	Cray UK
John Levesque	Cray USA
Peter Messmer	Nvidia
Jesus Labarta	BSC

And my other partners in the CRESTA Project

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

Acknowledgements/2

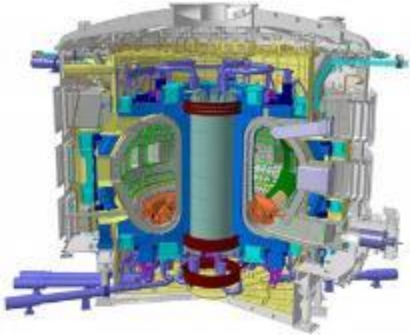
An award of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program. This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

Outline

- CRESTA?
- IFS model focus
- IFS model evolution
- Overlapping computation and communication
  - one-sided Fortran2008 coarray communications
  - DAG scheduling (OmpSs) study
- Radiation in parallel development
- OpenACC port of spectral transform test
- Alternative dynamical core option

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

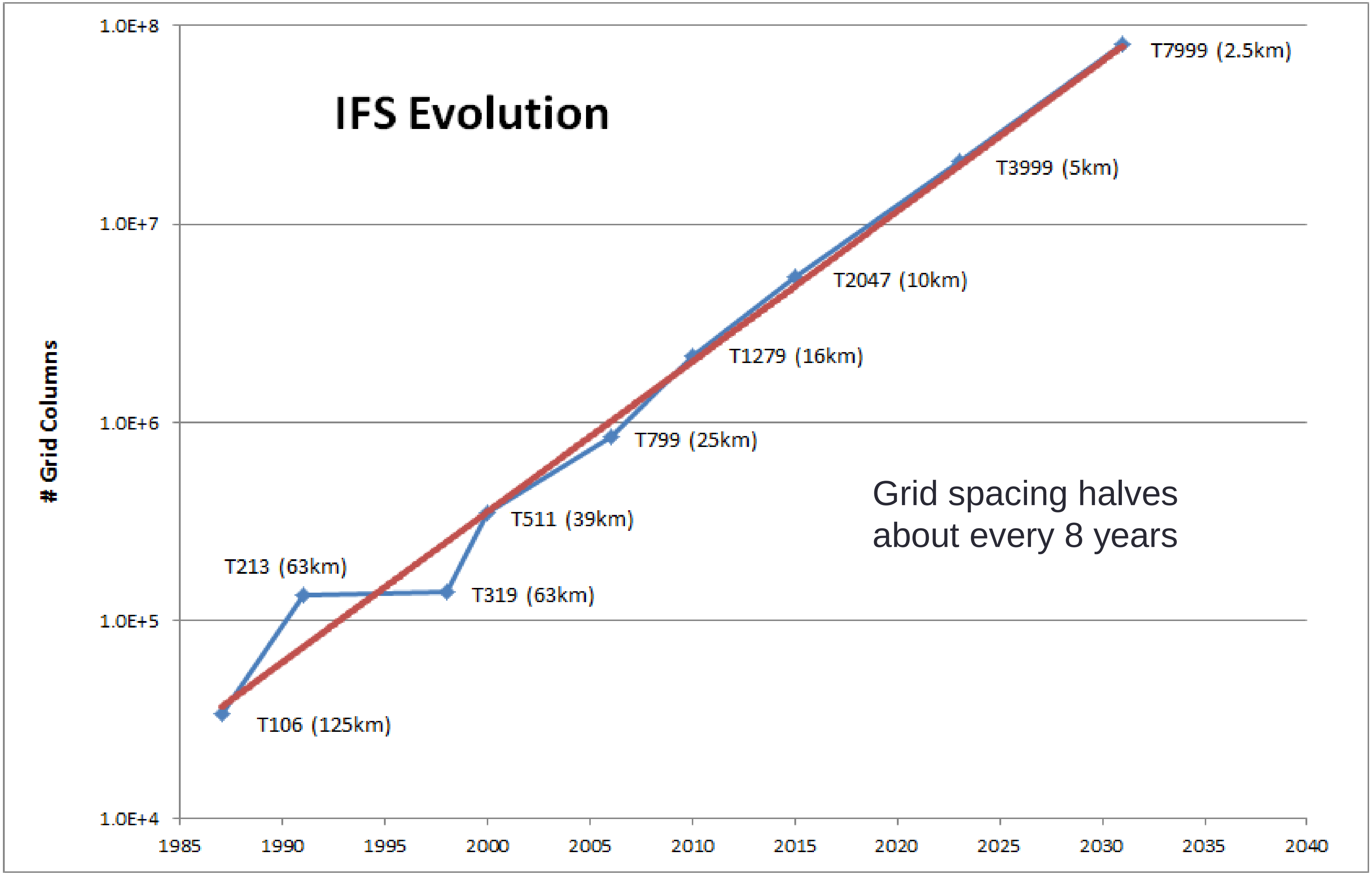


- Two tool suppliers
  - ALLINEA (ddt : debugger) & TUD (vampir : performance analysis)
- Technology and system supplier – CRAY UK
- Many Others (mostly universities)
  - ABO, CRSA, CSC, DLR, JYU, KTH, UCL, UEDIN-EPCC, USTUTT-HRLS

CRESTA

IFS model: some background

- 10-15 day forecasts, Hi-Resolution and Ensembles
- Spectral, semi-implicit, semi-Lagrangian
- Long time step (600 seconds today for operational T_L1279L137)
- Joint development between ECMWF and Météo France
- MPI+OpenMP parallelisation
- Operational Hi-Res model 10-day forecast to complete in UNDER one hour
- Current and Future Goals
  - Improving forecast skill & initial conditions from data assimilation
  - Performance, Scalability
  - Portability, Reliability and Low Power
  - We all want these!!!!


IFS Evolution
Grid Columns
1.0E+8
1.0E+7
1.0E+6
1.0E+5
1.0E+4
1985
1990
1995
2000
2005
2010
2015
2020
2025
2030
2035
2040
T106 (125km)
T213 (63km)
T319 (63km)
T511 (39km)
T799 (25km)
T1279 (16km)
T2047 (10km)
T3999 (5km)
T7999 (2.5km)
Grid spacing halves
about every 8 years

IFS model: current and future model resolutions (at start of CRESTA project)

IFS model resolution	Envisaged Operational Implementation	Grid point spacing (km)	Time-step (seconds)	Estimated number of cores[1]
T1279 H[2]	2013 (L137)	16	600	2K
T2047 H	2014-2015	10	450	6K
T3999 NH[3]	2023-2024	5	240	80K
T7999 NH	2031-2032	2.5	30-120	1-4M

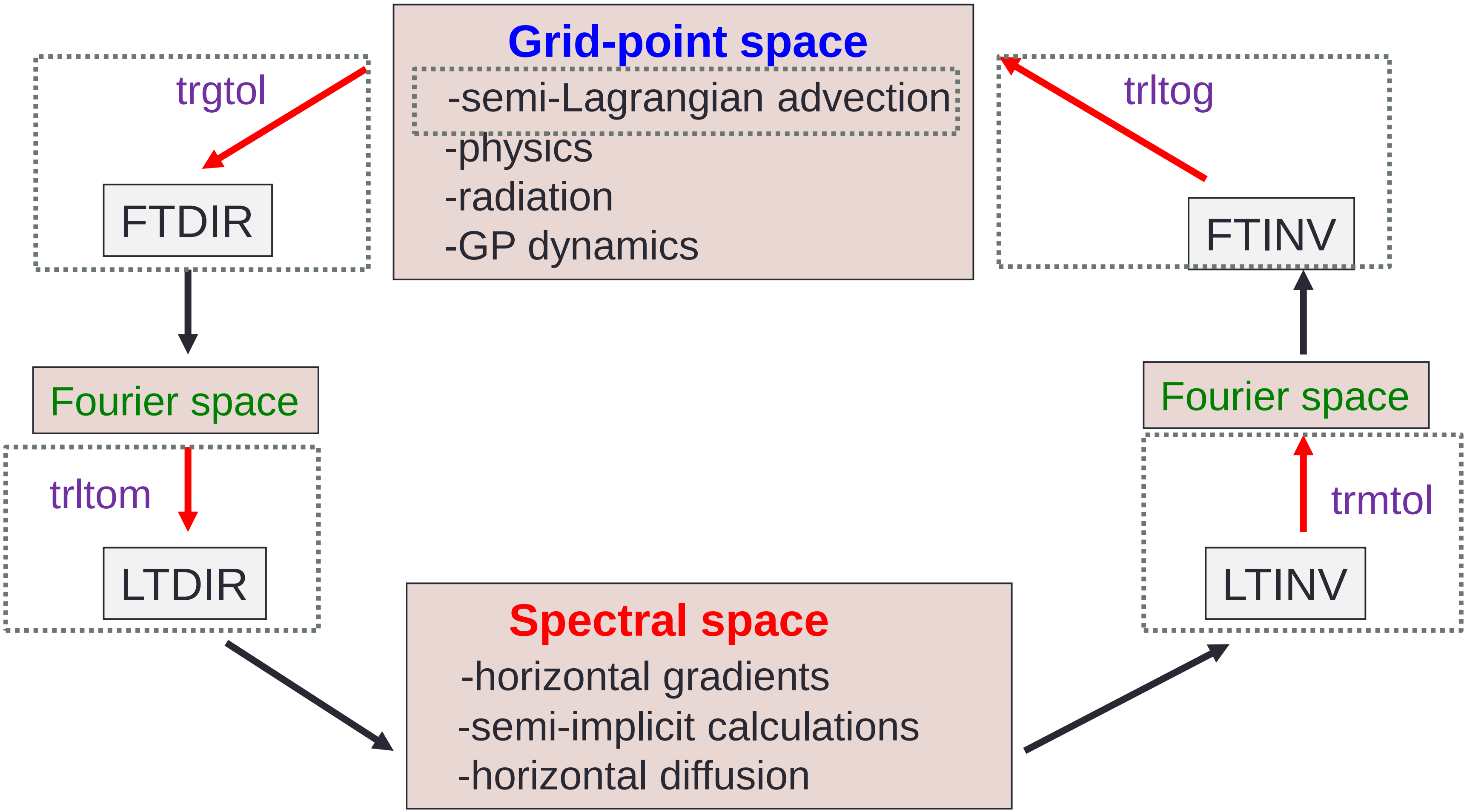
1 – a gross estimate for the number of ‘IBM Power7’ equivalent cores needed to achieve a 10 day model forecast in under 1 hour (~240 FD/D), system size would normally be ~10 times this number.
2 – Hydrostatic Dynamics
3 – Non-Hydrostatic Dynamics

Compute/Communication Overlap strategies (for hi-res IFS model)

- OpenMP parallel loops containing,
  - Expensive computations
  - PGAS one-sided communications
  - Implemented in IFS model using Fortran2008 coarrays
  - Need for coarray teams (in next Fortran standard)
  - GASPI/GPI library calls could be used as an alternative to coarrays
- Graph based approaches (DAG), e.g. OmpSs from BSC
  - Create computation tasks and communication tasks including the dependencies e.g. !$OMP TASK IN(…) OUT(…)
  - Run time system to process the graph as it evolves
  - Explored with an IFS model kernel (collaboration with BSC)
  - Extrae, Paraver, Mercurium & Nanos installed and used on XC-30

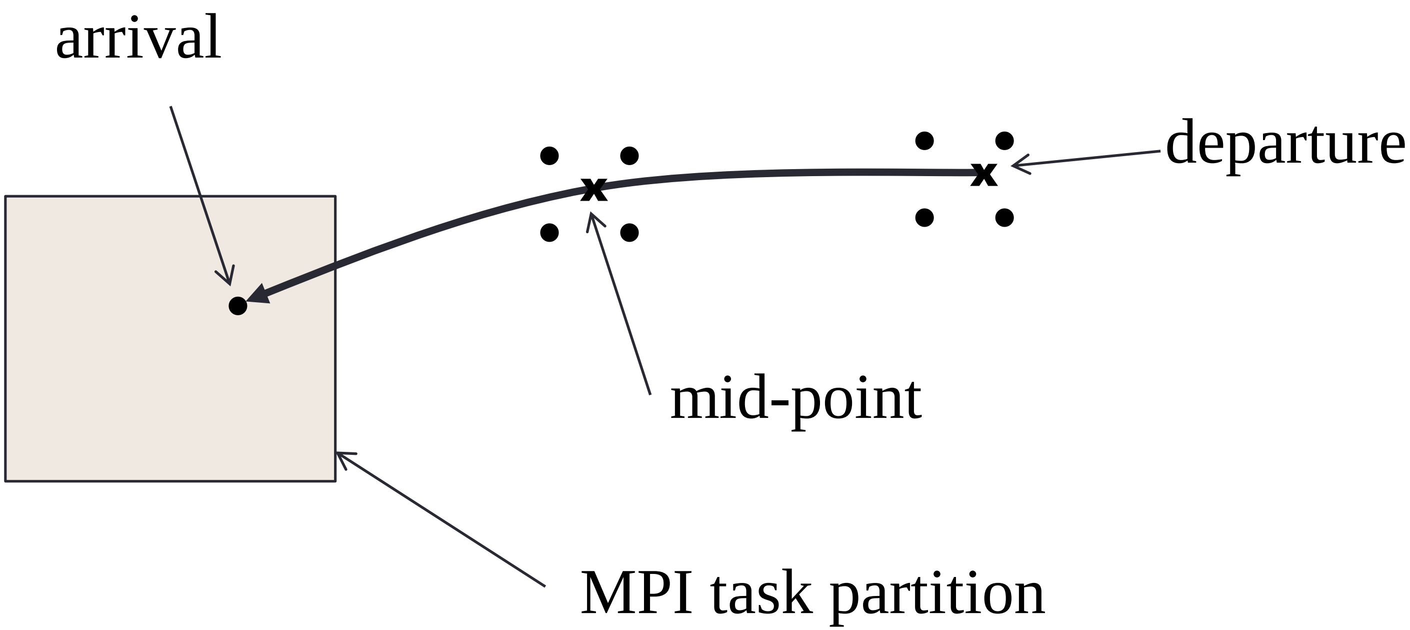
CRESTA

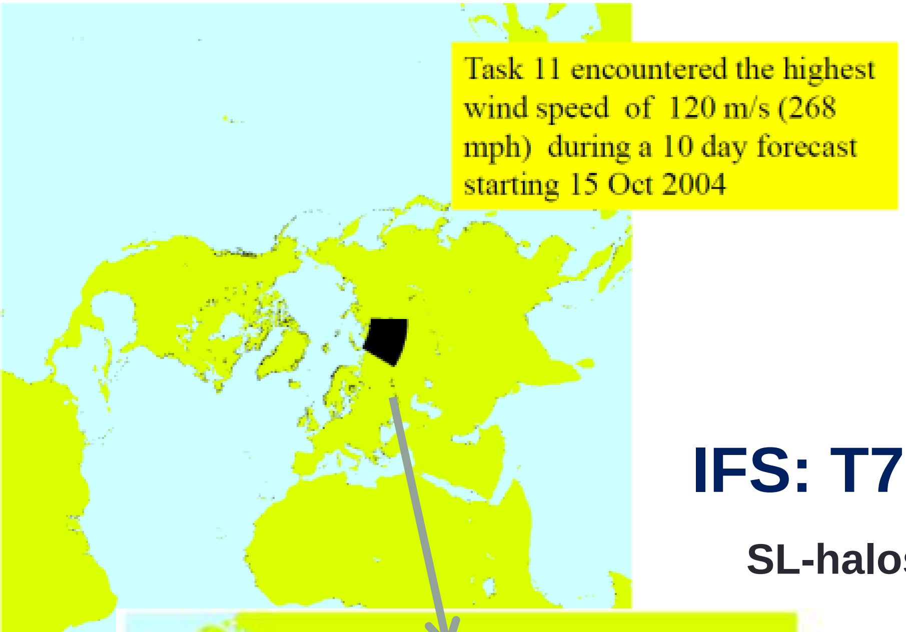
IFS coarray optimizations for [Tera,Peta,Exa]scale

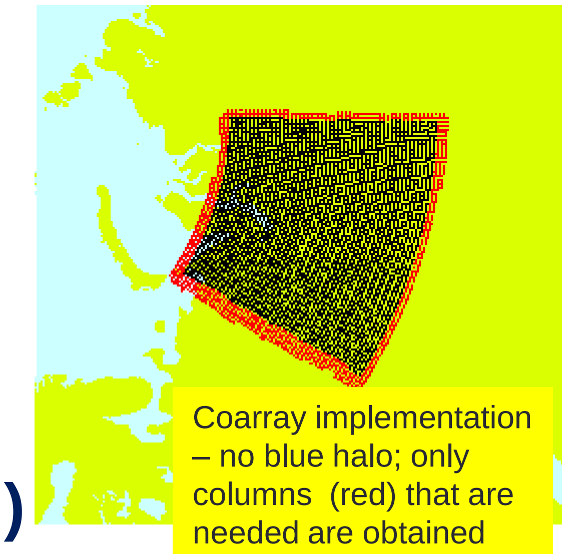


Semi-Lagrangian Transport

- 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

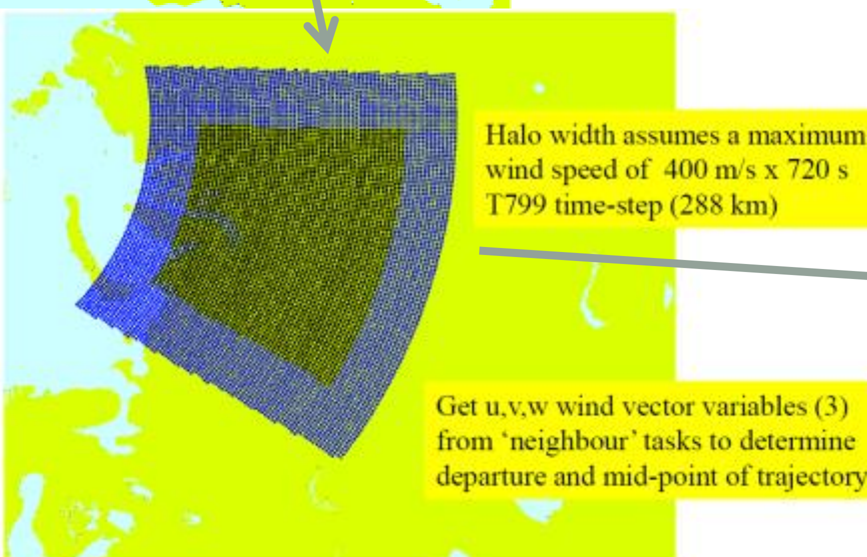


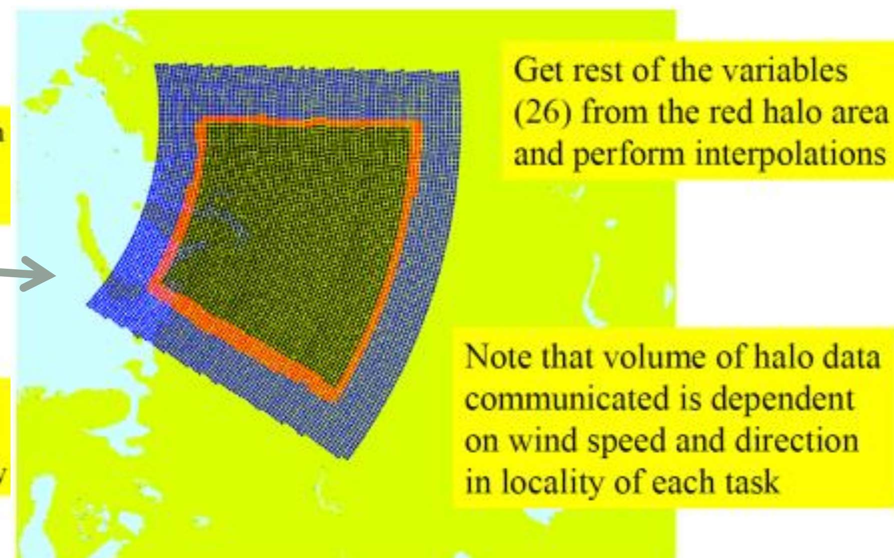




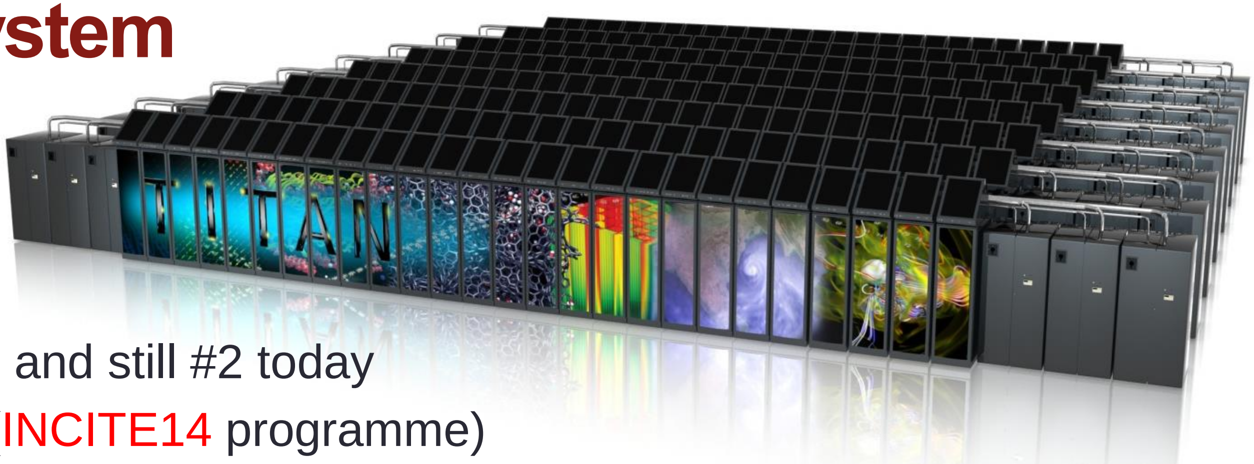
IFS: T799L91 (25 km)

SL-halos for task 11 / 256





ORNL's "Titan" System



- #1 in Nov 2012 Top500 list, and still #2 today
- CRESTA awarded access (INCITE14 programme)
- 7.5X peak perf. of ECMWF's XC-30 clusters (CCA+CCB=3.6 Petaflops)
- 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

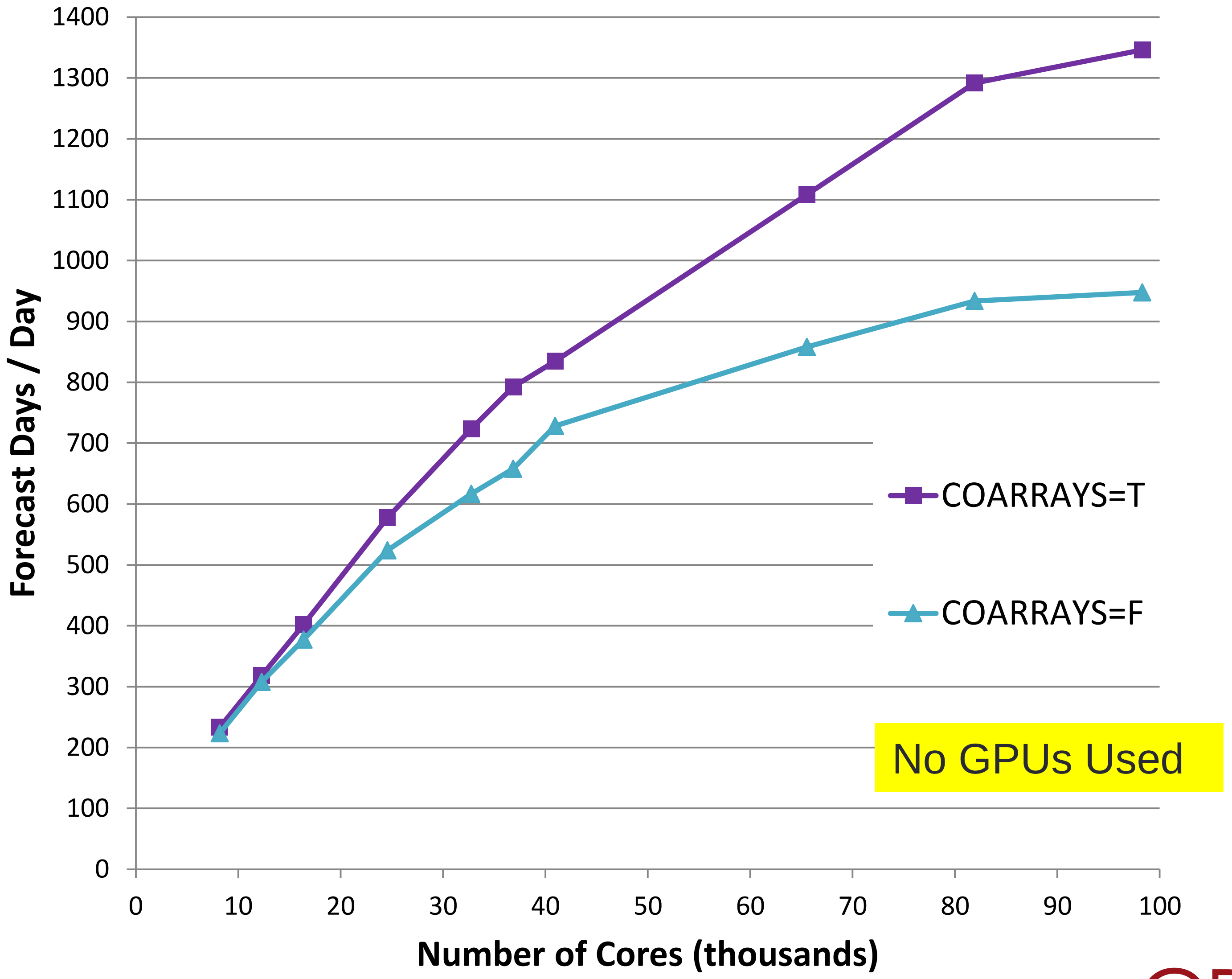
Titan Specs	
Compute Nodes	18,688
Login & I/O Nodes	512
Memory per node	32 GB + 6 GB
# of NVIDIA K20 "Kepler" processors	14,592
Total System Memory	688 TB
Total System Peak Performance	27 Petaflops

Source (edited): *James J. Hack, Director,* Oak Ridge National Laboratory

CRESTA

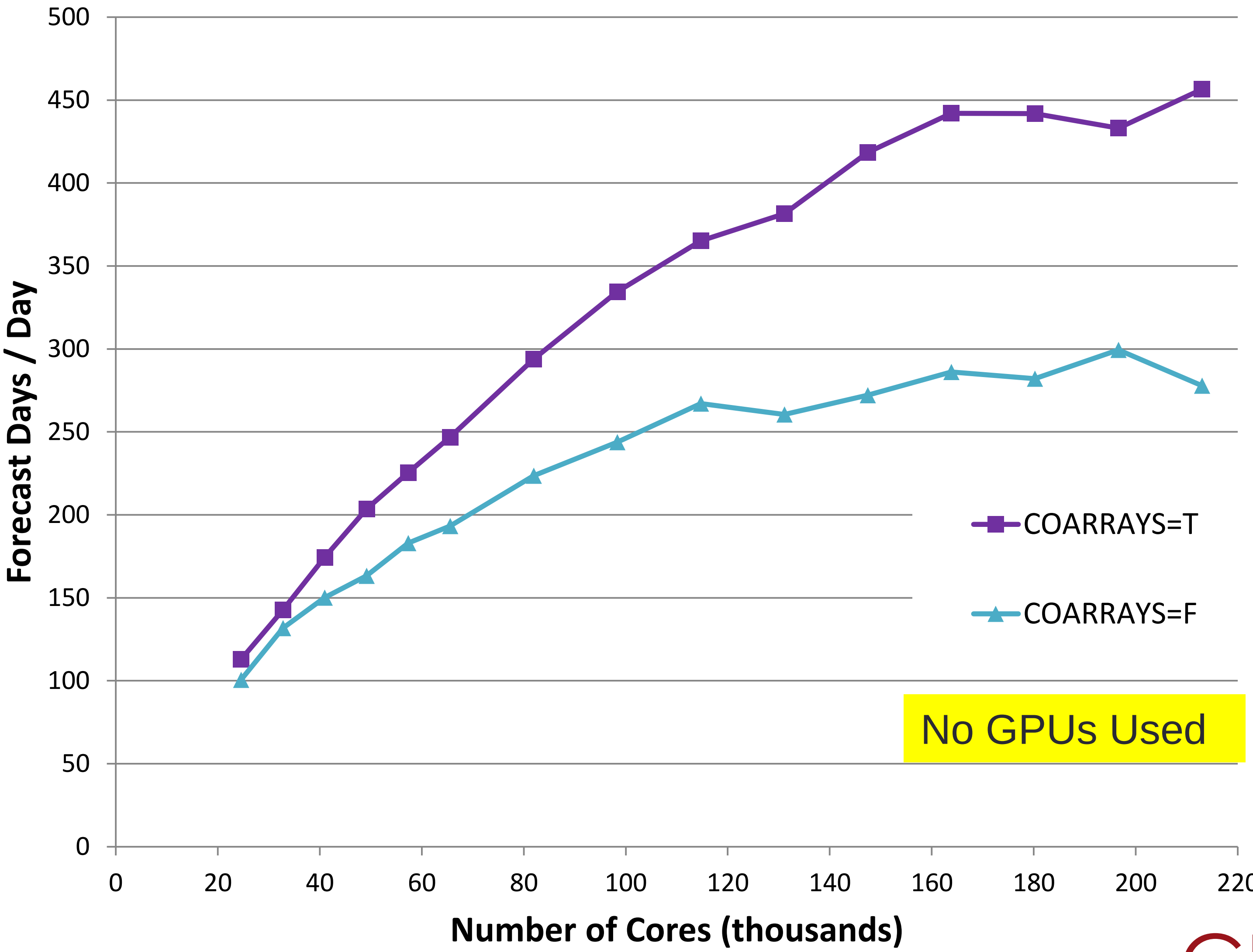
"At the start of the CRESTA project the maximum number of cores that an IFS model had run on was less than 10,000"

T_c1023L137 10 km (~2015) IFS model scaling on TITAN



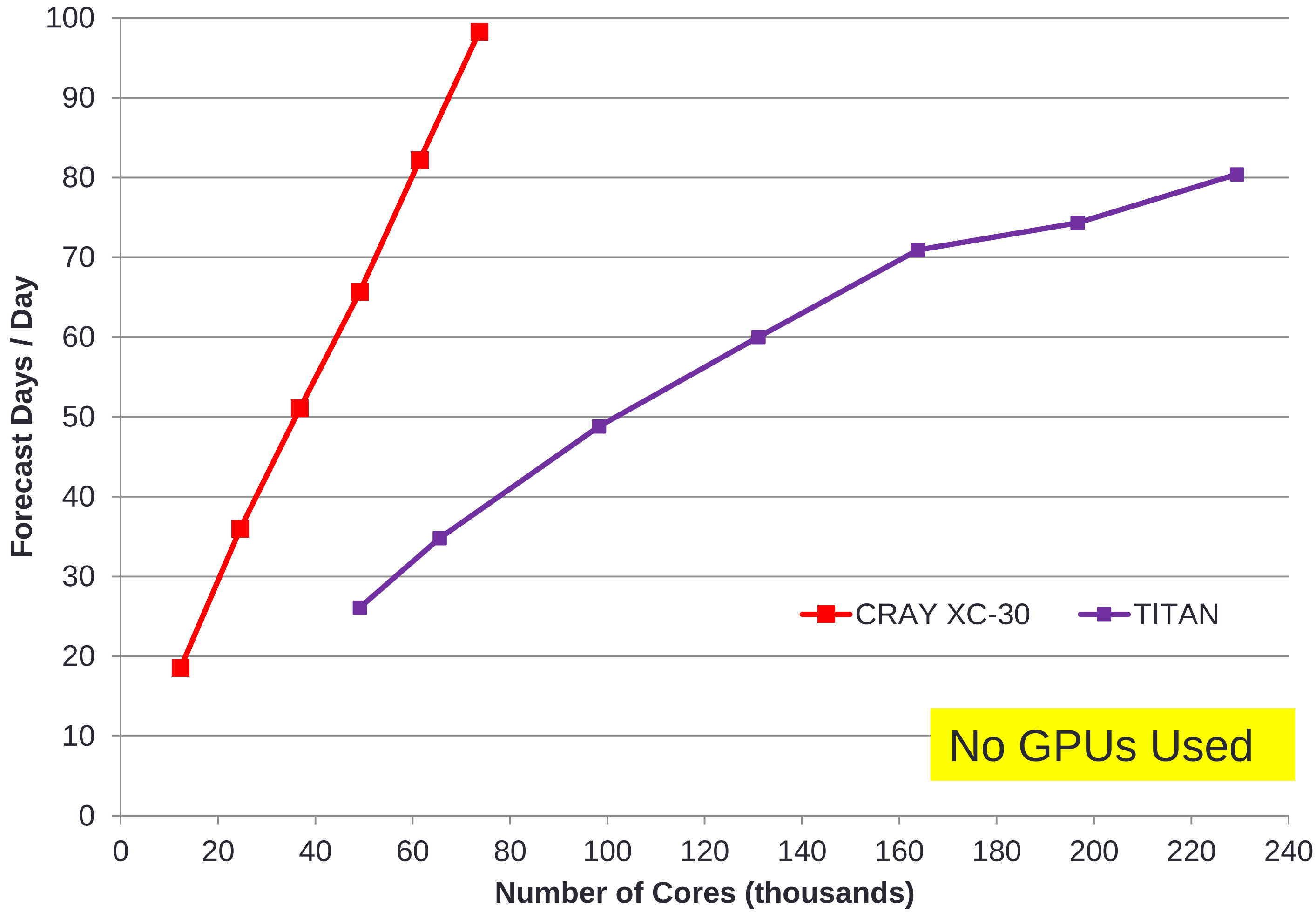
CREST

T_c1999L137 5 km (~2024) IFS model scaling on TITAN



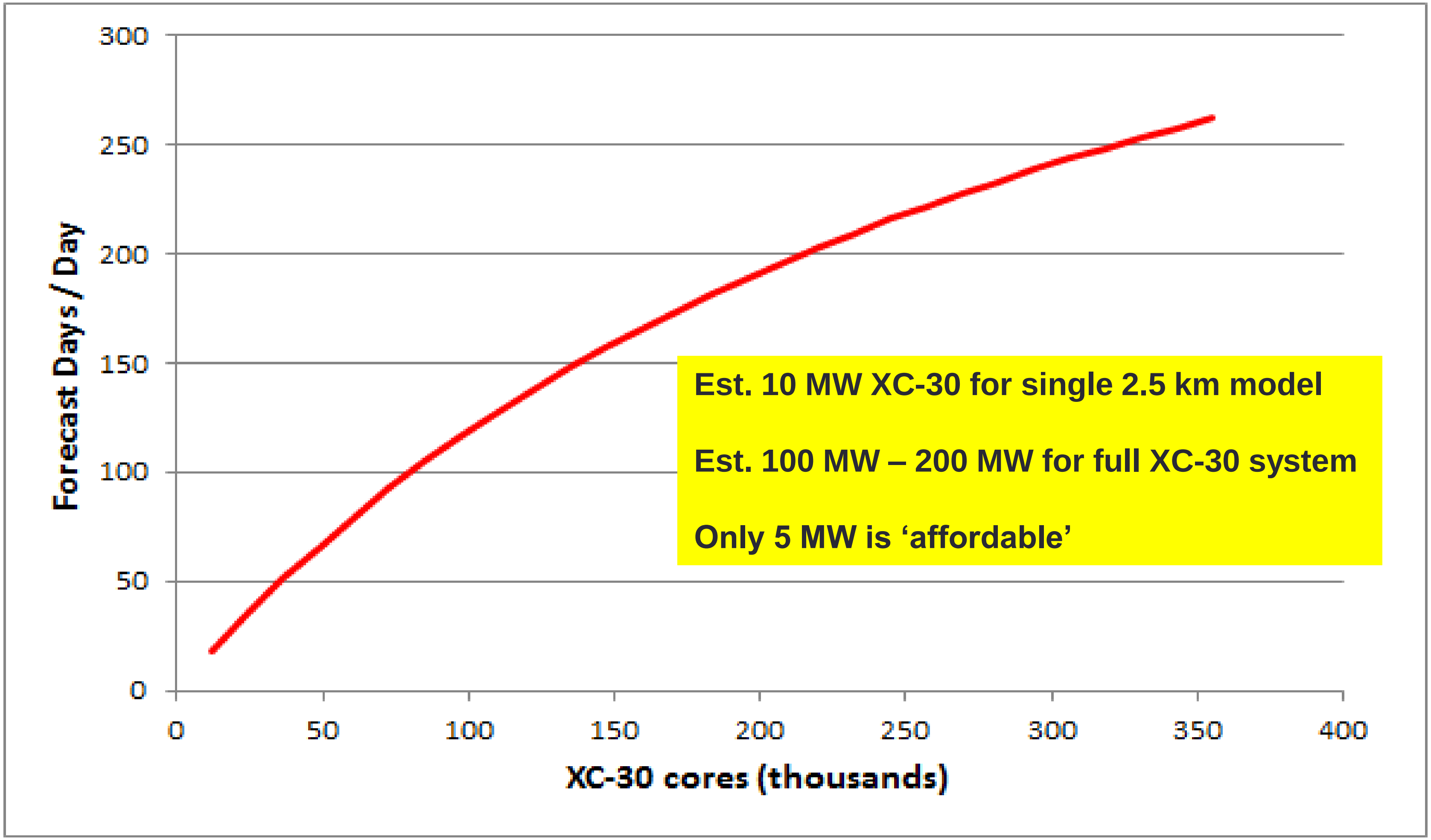
CREST

T_c3999L137 2.5 km (~2032) IFS model scaling



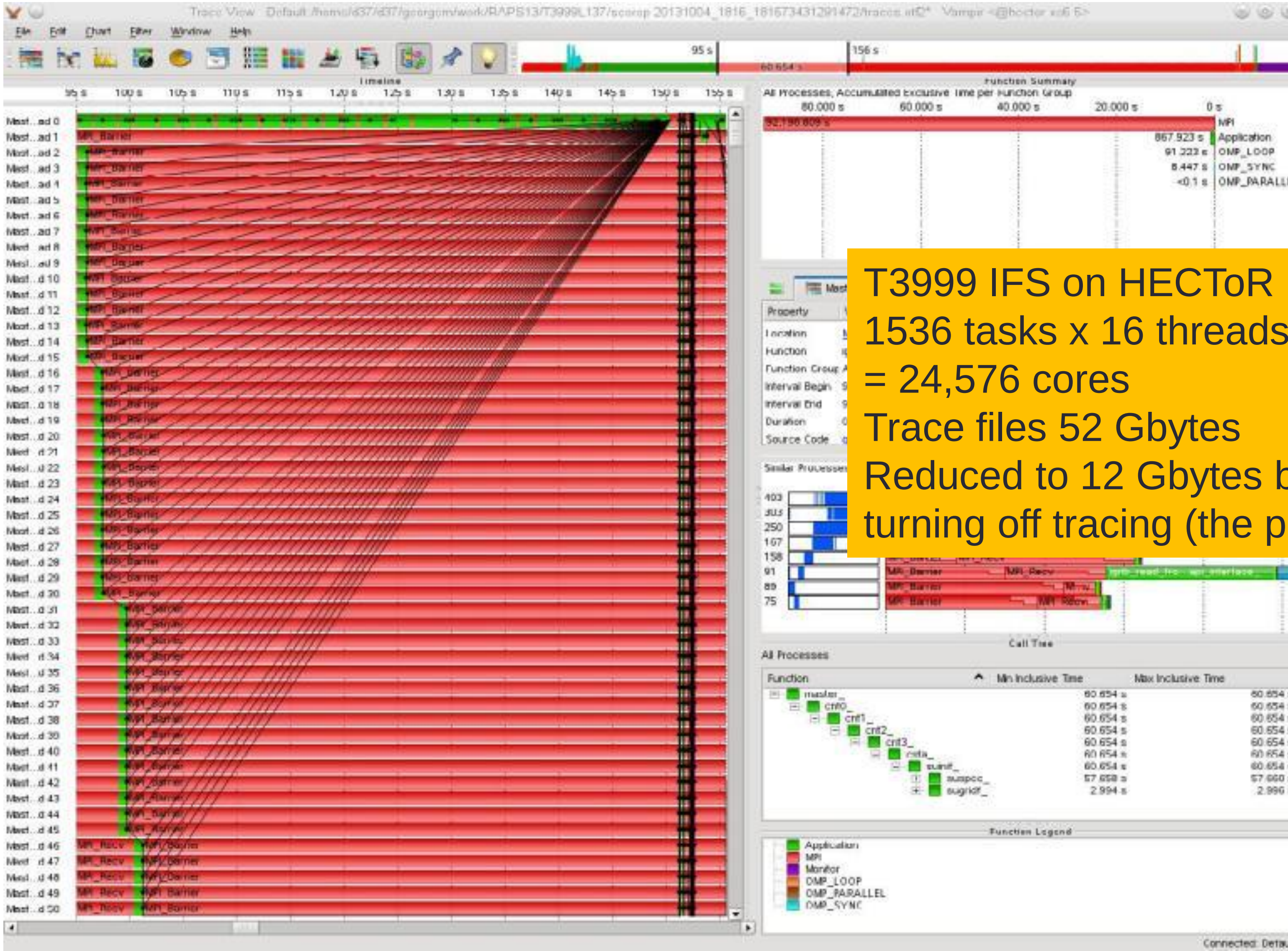
CREST

2.5 km (~2032) global IFS model EXTRAPOLATION



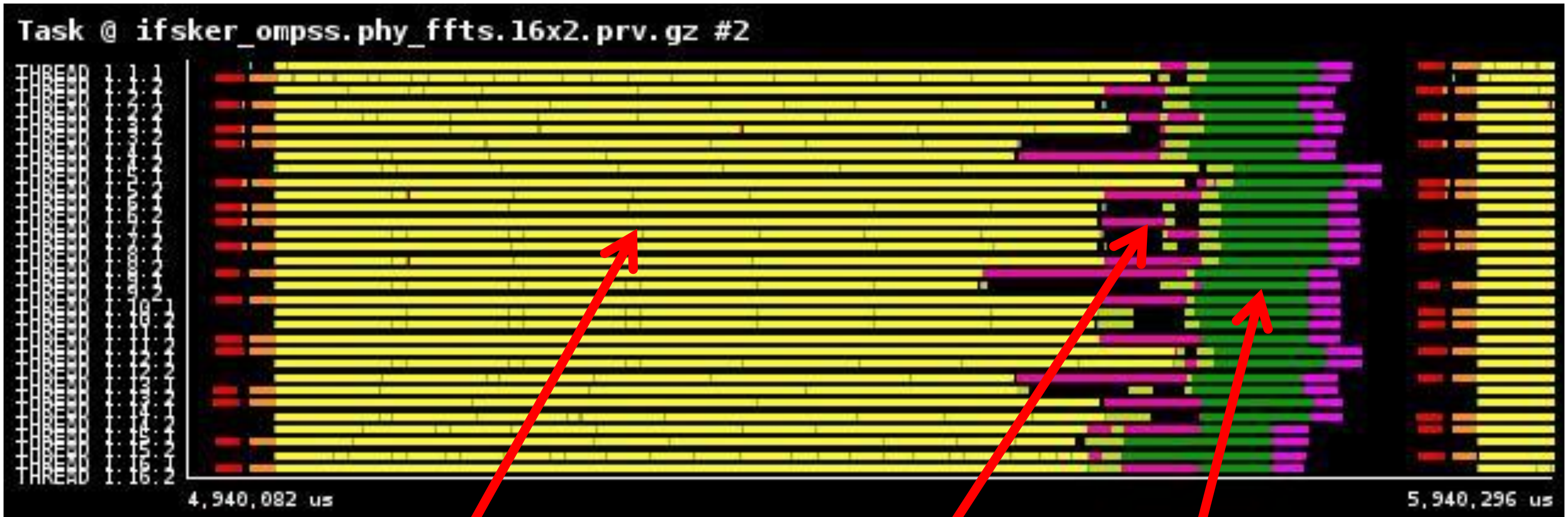
CREST

Co-design: IFS initialization study with vampir (& IFS gstats)



T3999 IFS on HECToR using
1536 tasks x 16 threads
= 24,576 cores
Trace files 52 Gbytes
Reduced to 12 Gbytes by
turning off tracing (the purple bit)

IFS kernel using OmpSs (on single node)



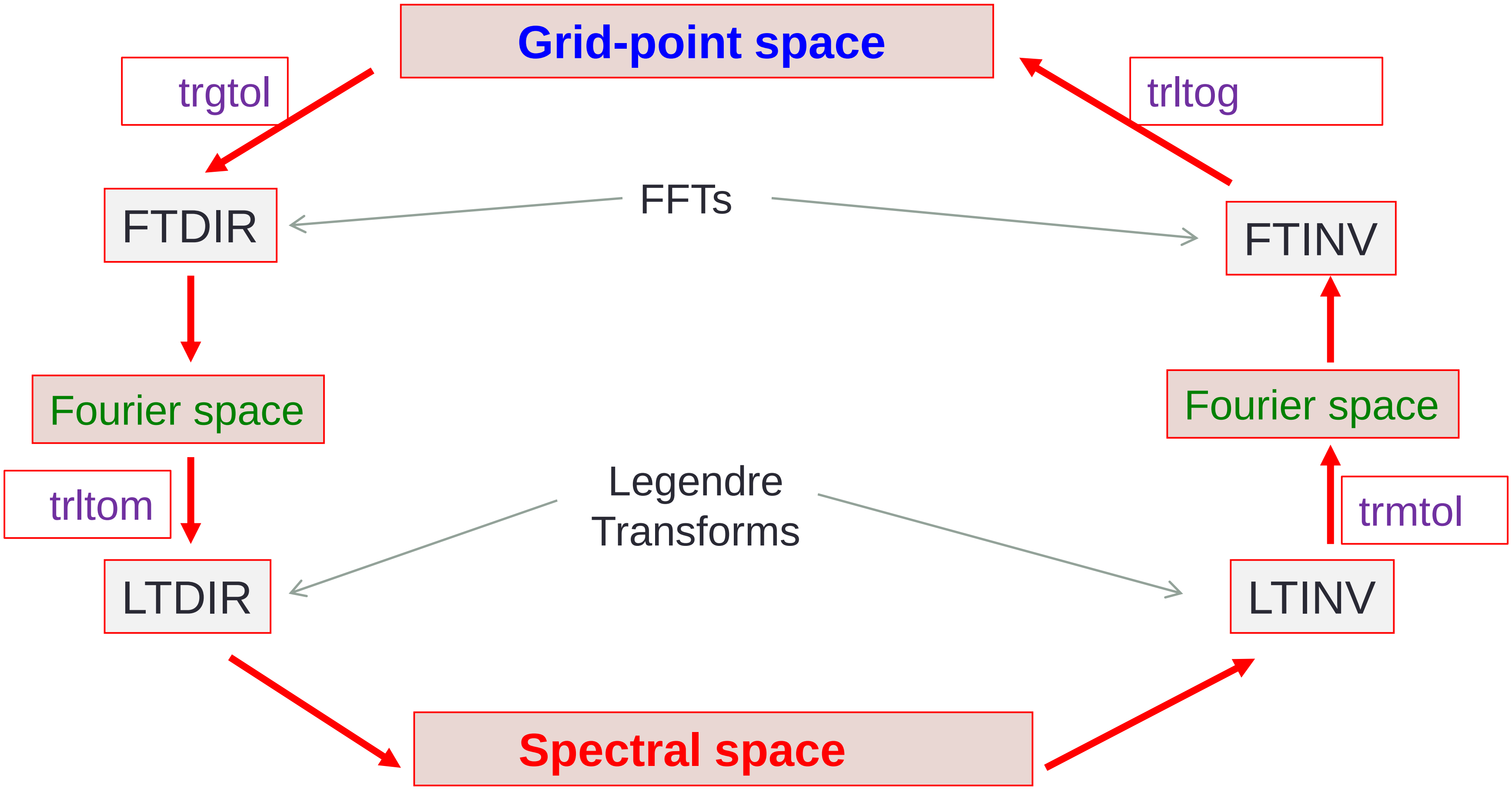
Physics

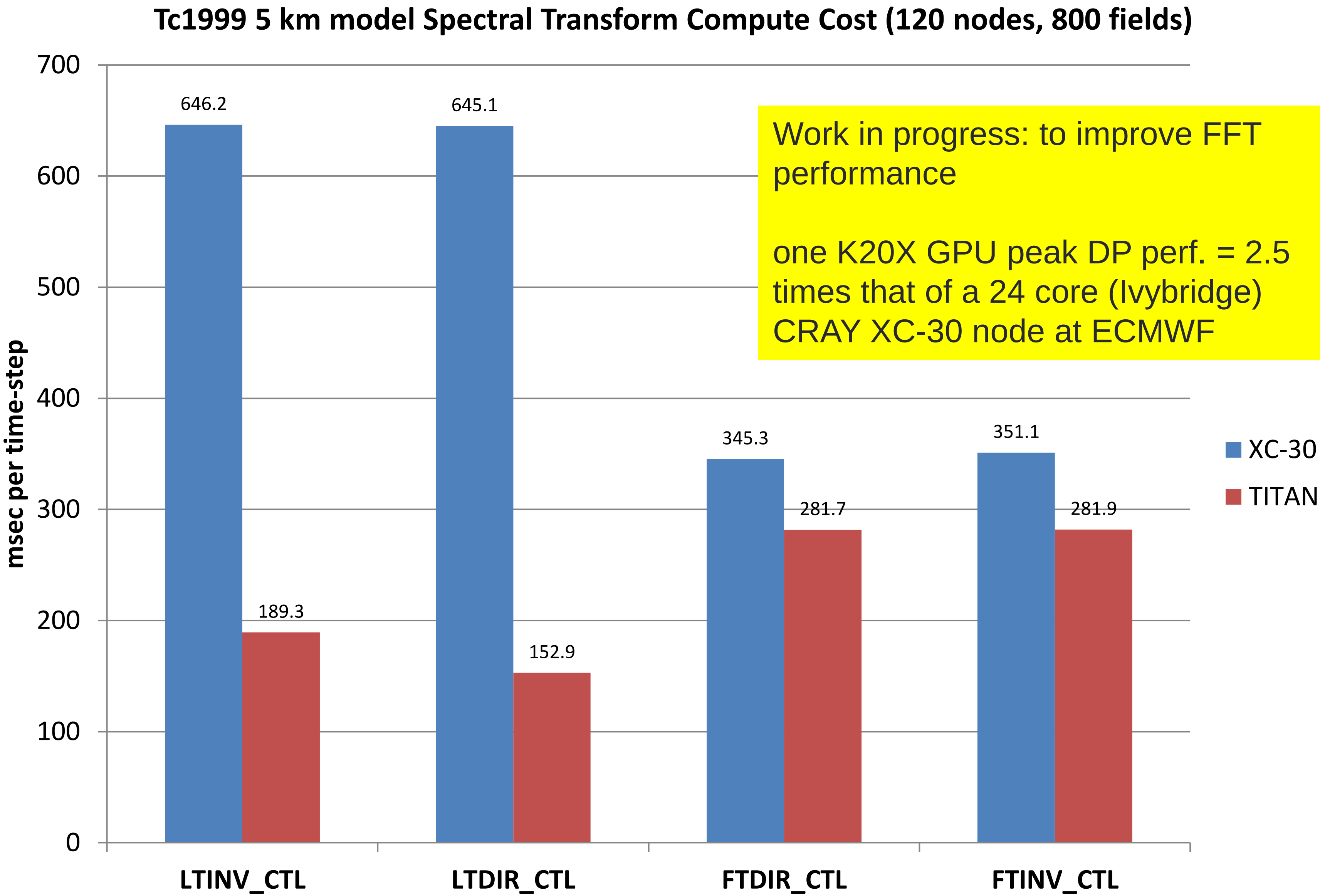
Comms

FFTs

collaboration with Prof Jesus Labarta @ BSC

Spectral Transform test : OpenACC K20X GPU port





Work in progress: to improve FFT performance

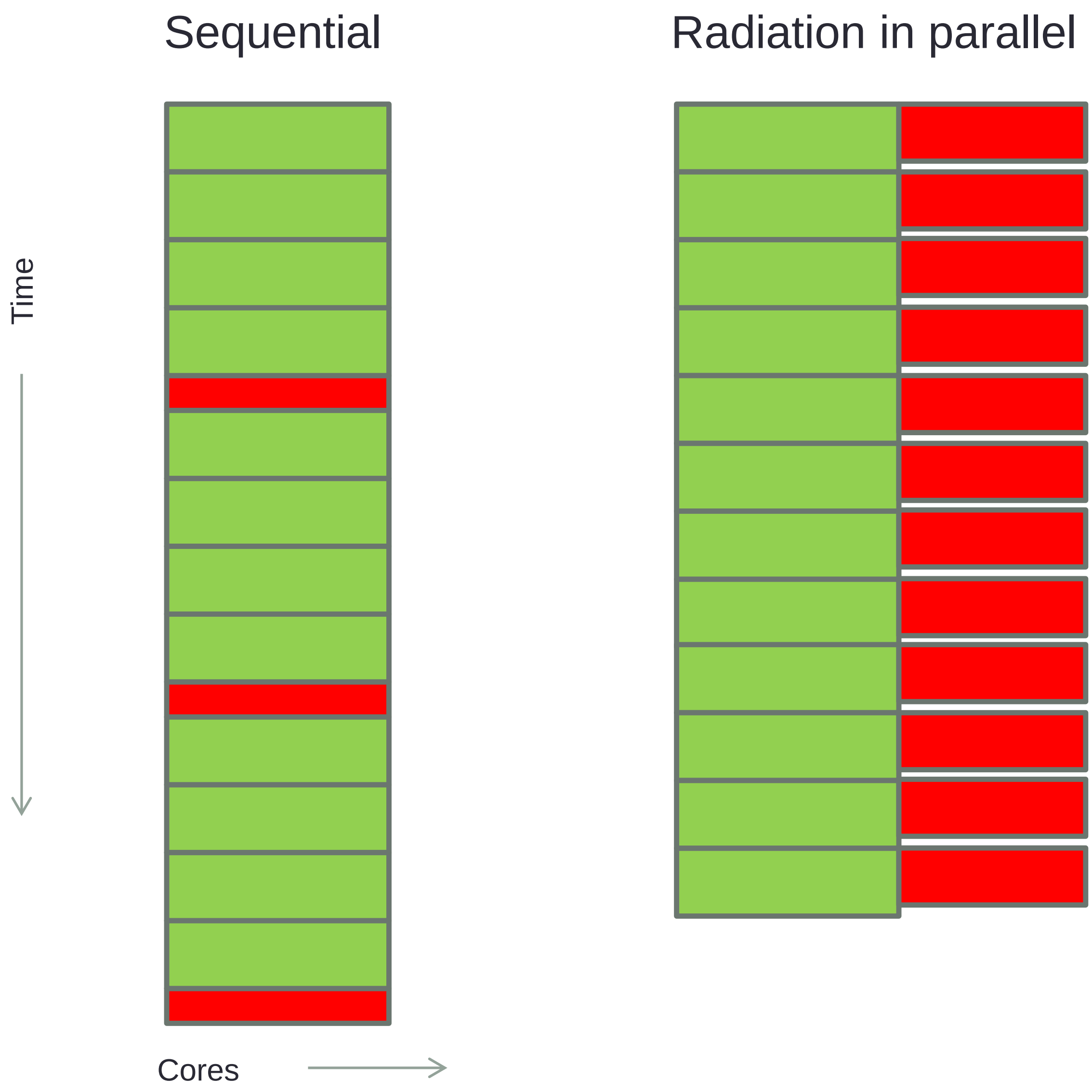
one K20X GPU peak DP perf. = 2.5 times that of a 24 core (Ivybridge) CRAY XC-30 node at ECMWF

Lessons Learnt from transform test OpenACC port

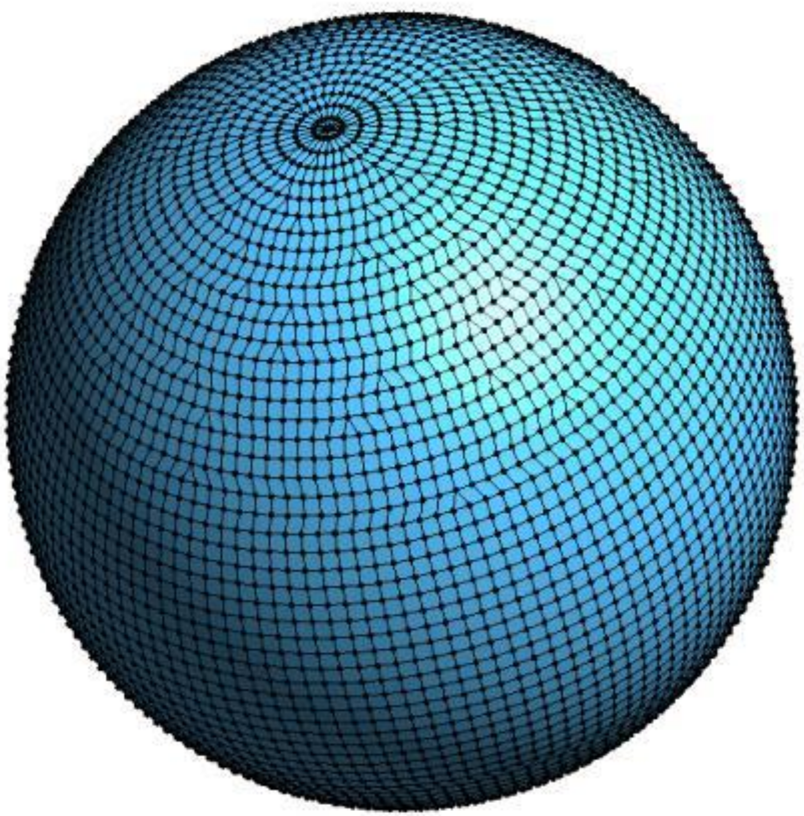
- OpenACC programming effort
  - Replaced ~20 OpenMP directives (high-level parallelisation)
  - By ~280 OpenACC directives (low-level parallelisation)
- Most of the porting time spent on,
  - Strategy for porting existing IFS FFT interface
  - Replaced by calls to new cuda interface
    - Calls to NVIDIA cuFFT library routines
- Performance issues
  - GPU/OpenACC: on each GPU, FFTs execute sequentially over latitudes (all fields)
  - XC-30/OpenMP: on each node, FFTs execute in parallel over latitudes (all fields)
  - FFT data layout important on GPU (fields,latitudes) v (latitudes,fields)

CREST

Radiation computations in parallel with model



IFS alternative dynamical core option



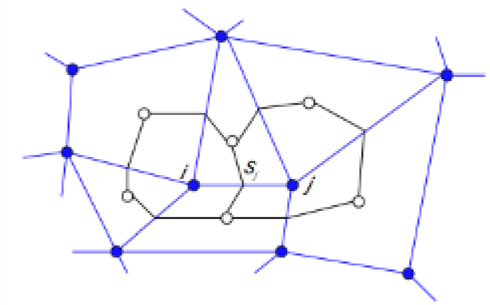
IFS T63 mesh
Nodes (points) are existing T63 grid

Existing IFS EQ_REGIONS partitioning

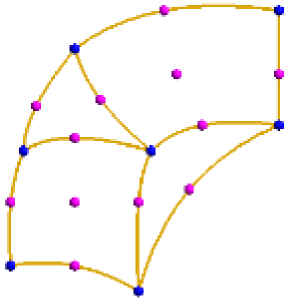
What do we want to support?

Nearest neighbour communication algorithms, compact stencils

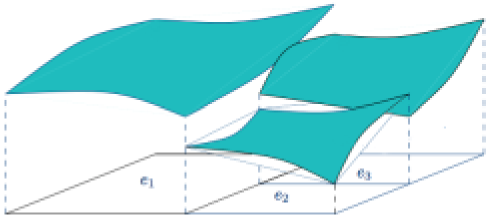
Edge-based Finite-Volume scheme (MPDATA)



Element-based Finite-Element scheme



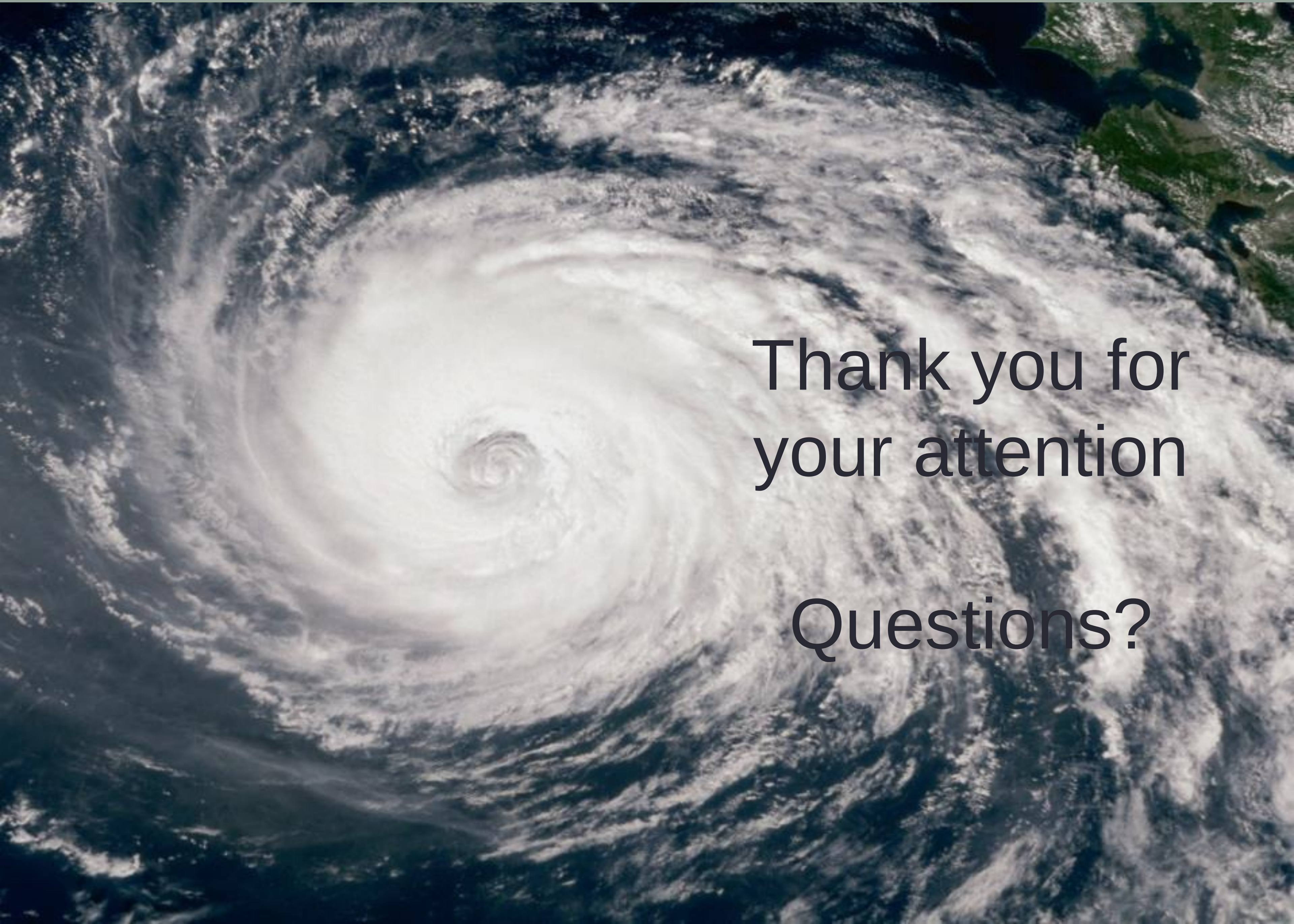
Element-based Discontinuous Higher-Order schemes



Summary

- Many challenges exist for IFS **applications** to run at the Exascale
- First of these is for hardware vendors to build Exascale computers that are both affordable (cost + power) and reliable
- Overlapping computations and communications must be considered at the Petascale/Exascale (F2008 coarrays v GASPI/GPI or DAG?)
- MPI/OpenMP (high level parallelisation) is easier to program/maintain than MPI/OpenACC (low level parallelisation)
- Ease of programming GPU technology should be easier in the future when there is a single address space for GPU and conventional cores
- IFS model initialisation expensive at scale (single reader, grib_api)
- IFS **applications** (not just the model) will require substantial development in the years to come to run efficiently on Petascale and more so on Exascale computers

CREST



Thank you for your attention

Questions?

Fortran2008 coarray (PGAS) example

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

CREST

T_c3999 transform test performance comparison, including computation **and** communication (simple 1D parallel, IFS uses 2D)

Tc3999	XC-30	GPU	XC-30+ GPU prediction
LTINV_CTL	1024.9	324.3	324.3
LTDIR_CTL	1178.6	279.8	279.8
FTDIR_CTL	428.3	342.3	342.3
FTINV_CTL	424.6	341.8	341.8
TRMTOL	752.5	**4763.0**	752.5
TRLTOM	407.9	**4782.9**	407.9
TRLTOG	1225.9	**1541.9**	1225.9
TRGTOL	401.5	**1658.4**	401.5
HOST2GPU	n/a	655.4	655.4
GPU2HOST	n/a	650.0	650.0
Total	5844.2	14034.4	5381.4

Using 400 nodes on both XC-30 and TITAN (GPU)
single time-step cost in millisecs

CREST