ECMWF's IFS: Parallelization and exascale computing challenges

George Mozdzynski, Mats Hamrud, Nils Wedi

European Centre for Medium Range Weather Forecasts
Shinfield Park, Reading RG2 9AX, UK
{George.Mozdzynski, Mats.Hamrud, Nils.Wedi}@ecmwf.int

Abstract— ECMWF is a partner in the Collaborative Research into Exascale Systemware, Tools and Applications (CRESTA) project, funded by a EU call for projects in Exascale computing, software and simulation (ICT-2011.9.13). The significance of the research carried out within the CRESTA project is that it will demonstrate techniques required to scale the current generation of Petascale capable simulation codes towards the performance levels required for running on future Exascale systems. Today the European Centre for Medium-Range Weather Forecasts (ECMWF) runs a 16 km global T1279 operational weather forecast model using 1,600 cores of an IBM Power7. Following the historical evolution in resolution upgrades, ECMWF could expect to be running a 2.5 km global forecast model by 2030 on an Exascale system that should be available and hopefully affordable by then. To achieve this would require IFS to run efficiently on about 1000 times the number of cores it uses today. This is a significant challenge, one that we are addressing within the CRESTA project. Two years into the CRESTA project, ECMWF has now demonstrated IFS running a 5 km global model on over 200,000 AMD Interlagos cores of TITAN a Cray XK7 at the Oak Ridge National Laboratory. Of course, to get to multi Petaflop sustained performance on TITAN IFS must utilize the NVIDIA Kepler K20 GPUs that are available on each node. This remains a formidable challenge, as are other scalability improvements that have yet to be implemented. Within CRESTA, ECMWF has explored the use of Fortran2008 coarrays; in particular it is possibly the first time that coarrays have been used in a world leading production application within the context of OpenMP parallel regions. The purpose of these optimizations is primarily to allow the overlap of computation and communication, and further, in the semi-Lagrangian advection scheme, to reduce the volume of data communicated. The importance of this research is such that if these and other planned developments are successful then the IFS model may continue to use the spectral method to 2030 and beyond on an Exascale sized system. In addition, ECMWF is considering alternative grid meshes and time-stepping schemes that could be used in this period. The current status of the coarray scalability developments to IFS will be presented in this paper, including an outline of planned developments.

1. Introduction

Weather-related natural disasters have major consequences for society and the economy. Over the last two decades, many thousands of lives have been lost as a result of windstorms, tropical cyclones, floods, drought, cold outbreaks and heat waves. In Europe alone, overall losses from severe weather total billions of Euros annually. The storms of Lothar and Martin of December 1999, which affected France and Germany, resulted in catastrophic damage to property, forests and electricity distribution networks. Lothar alone killed 125 people; and total losses due to windstorms in 1999 exceeded 13 billion Euros. The exceptional heat wave that affected most of Western Europe in August 2003 killed over 20,000 across Europe and caused economic losses of 8 billion Euros. Forecasts of severe weather events are vital to warn authorities and the public, and to allow appropriate mitigating action to be taken. Early warnings, made a few days ahead of potential events, are of significant benefit, giving additional time to allow contingency plans to be put in place. ECMWF is an application partner in CRESTA bringing the IFS numerical weather prediction application to the project. IFS is a production application used to provide medium-range weather forecast products up to 10 to 15 days ahead to its Member States and Co-operating States. At shorter range, national weather services use products from ECMWF to provide boundary data for their own regional and local short-range forecast models. In

figure 1 we show the evolution of the IFS model from the mid-1980's to the current T1279 operational model and extrapolated out to 2030. This figure shows that halving the horizontal grid spacing has occurred about every 8 years, and provides an estimate for the dates when the T3999 and T7999 models could be introduced into operations. It is clear that this simplistic extrapolation (given the number of grid columns and slope from T106 through T1279) does not take into account the many architectural and technology changes that are needed to get to the Exascale. For IFS the initial focus of developments in CRESTA is primarily to use Fortran2008 coarrays within OpenMP parallel regions to overlap computation with communication and thereby improve performance and scalability. This research is significant as the techniques used should be applicable to other hybrid MPI/OpenMP codes with the potential to overlap computation and communication. Of course, using coarrays is not a panacea to get codes to achieve scaling at the Exascale. What it does allow us to do is to experiment with exposing parallelism at the fine scale for both computation and communication. It is expected that future developments to IFS will build on this work. One of the challenges of the CRESTA project is to think more disruptively and we may consider much more innovative approaches; for instance, to run component models in parallel and explore Direct Acyclic Graph (DAG) scheduling. IFS is a spectral, semi-implicit, semi-Lagrangian weather prediction code, where model data exists in three spaces, namely, grid-point, Fourier and spectral space. In a single time-step (figure 2) data is transposed between these spaces so that the respective grid-point, Fourier and spectral computations are independent over two of the three co-ordinate directions in each space.

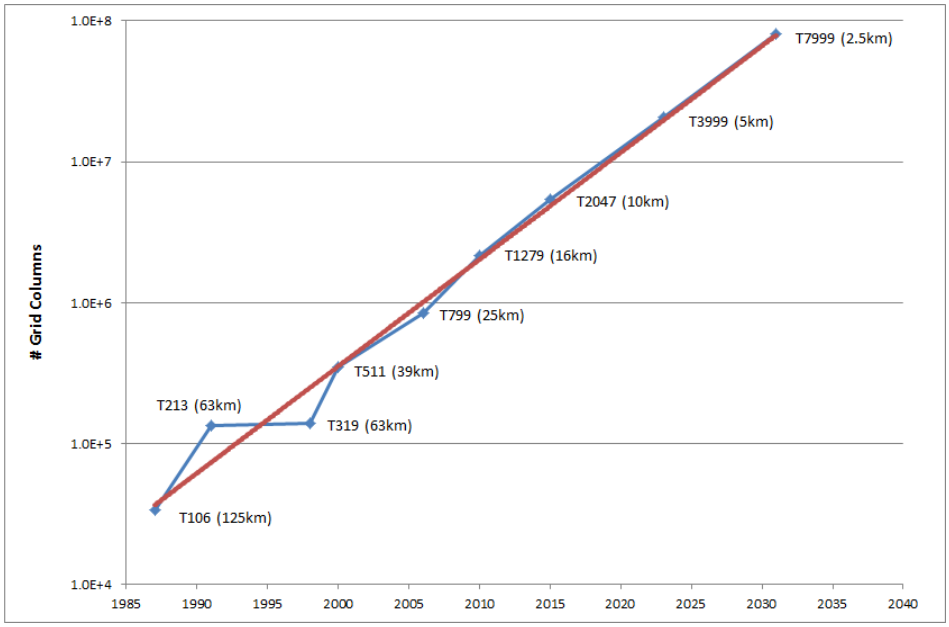

Figure 1: Evolution of the IFS model

Fourier transforms are performed between grid-point and Fourier space, and Legendre transforms are performed between Fourier and spectral space. A full description of the above IFS parallelization scheme is contained in [1]. At ECMWF, this same model is used in an Ensemble Prediction System (EPS) suite where today 51 models are run at lower resolution with perturbed input conditions to provide probabilistic information to complement the accuracy of the high resolution forecast. The EPS suite is a perfect candidate to run on future Exascale systems, with each ensemble member being

independent. By increasing the number of members and their resolution it is relatively easy to fill a future Exascale system. However, at the same time there will be a need to perform model simulations at convection-resolving resolutions of O(1km), which is more challenging to scale and the reason for ECMWF's focus in the CRESTA project. Today ECMWF uses a 16 km global grid for its operational high resolution model, and plans to scale up to a 10 km grid in 2014-15, followed by further resolution increases as shown in Table I. These model resolution increases will require IFS to run efficiently on over a million cores by around 2030. The current status of the coarray scalability developments to IFS will be presented in this paper, including an outline of planned developments. IFS comprises several component suites, namely, a 10-day high resolution forecast, a four dimension variational analysis (4D-Var), an ensemble prediction system (EPS) and an ensemble data assimilation system (ENDA). However good ensemble methods are for HPC systems, they are only part of the IFS production suite and the high resolution model (referred to as 'IFS model' from now on) and 4D-Var analysis applications are equally important in providing forecasts to ECMWF member states up to 10 to 15 days ahead.

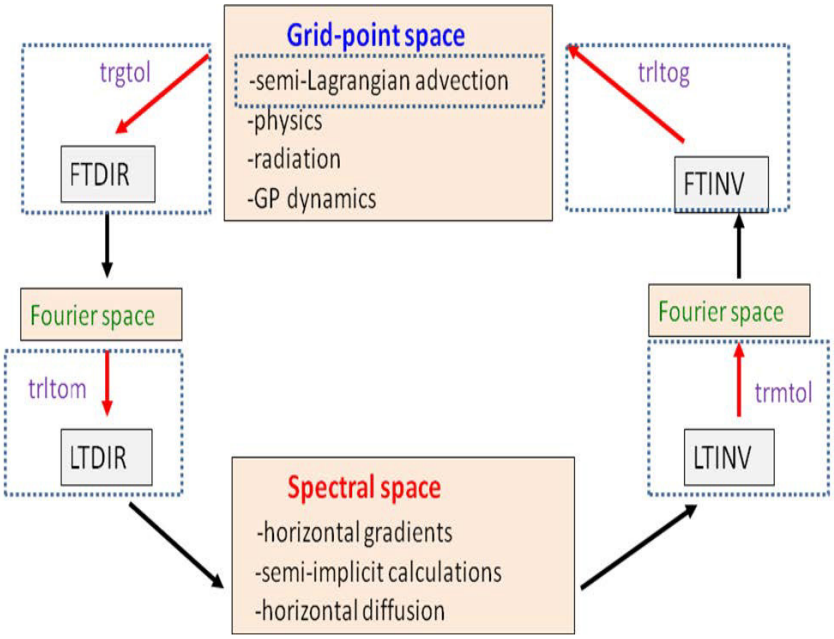

Figure 2: IFS time step: Current areas of IFS optimisation showing where coarray optimisations have been applied (the dotted boxes)

For the CRESTA project it has been decided to focus on the IFS model to understand its present limitations and to explore approaches to get it to scale well on future Exascale systems. While the focus is on the IFS model, it is expected that developments to the model should also improve the performance of the other IFS suites (EPS, 4D-Var and ENDA) mentioned above. The resolution of the operational IFS model today is T1279L137 (1279 spectral waves and 137 levels in the atmosphere). For the IFS model, it is paramount that it completes a 10-day forecast in less than one hour so that forecast products can be delivered on time to ECMWF member states. The IFS model is expected to be increased in resolution over time as shown in Table I. As can be seen in this table, the time-step reduces as the model resolution increases. For IFS, halving the grid spacing typically increases the computational cost by 12, a doubling of cost for each of the horizontal coordinate directions plus the time-step and only 50 percent more for the vertical. However, in reality the cost can be greater than

this, when some non-linear items are included such as the Legendre transforms and Fourier transforms. It is clear from this that the IFS model from a computational complexity viewpoint can utilize future supercomputers at Exascale and beyond. What is less clear is whether the IFS model can continue to run efficiently on such systems and continue to meet the operational target of one hour when running on millions of cores which it would have to do. The performance of the IFS model has been well documented over the past 20 years, with many developments to improve performance, with more recent examples described in [2], [3], [4], [5], [6] and [8]. In recent years focus has turned to the cost of the Legendre transform, where the computational cost is O(N**3) for the global model, where N denotes the cut-off wave number in the triangular truncation of the spherical harmonics expansion.

Table 1. IFS model: current and future resolutions

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

This has been addressed by a Fast Legendre Transform (FLT) development, where the computational cost is reduced to C_L*N**2*log(N) where C_L is a constant and C_L<<N. The relative performance improvement is shown in Figure 3, where dgemm is the cost of the original algorithm and FLT the cost of the new algorithm. It should be noted that the FLT algorithm is only used for wave number beyond 1279, even within higher resolution models. The FLT algorithm is described in [9], [10], [11] and [12]. While the cost of the Legendre transforms has been addressed, the associated TRMTOM and TRMTOL transpositions (also shown in figure 1) between Fourier and spectral space remain relatively expensive at T3999 (>10 per cent of wall time). Today, these transpositions are implemented using efficient MPI_alltoallv collective calls in separate communicator groups, which can be considered the state of the art for MPI communications. In addition, the cost of transpositions between grid-point space and Fourier space are also relatively expensive, these being TRGTOL and TRLTOG, and are implemented today by non-blocking MPI_send, MPI_recv and MPI_wait calls as no MPI collective calls are applicable here. Figure 4 shows the relative cost of the spectral method for current and planned IFS resolutions as run on ECMWF's current IBM Power7 systems. Some of the costs clearly come from increased resolution and the need to use greater numbers of cores. In practice for resolutions under T7999 (2.5 km grid) and possibly T3999 (5 km grid), the costs are half those shown, due to a hydrostatic formulation being used.

In figure 4, the red bars indicate the total cost including the MPI communications involved. Percentages have been derived considering all grid-point dynamics and physics computations but without considering IO, synchronization costs (barriers), and any other ancillary costs. All runs show

that the MPI communications cost (the difference between the two columns) 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.

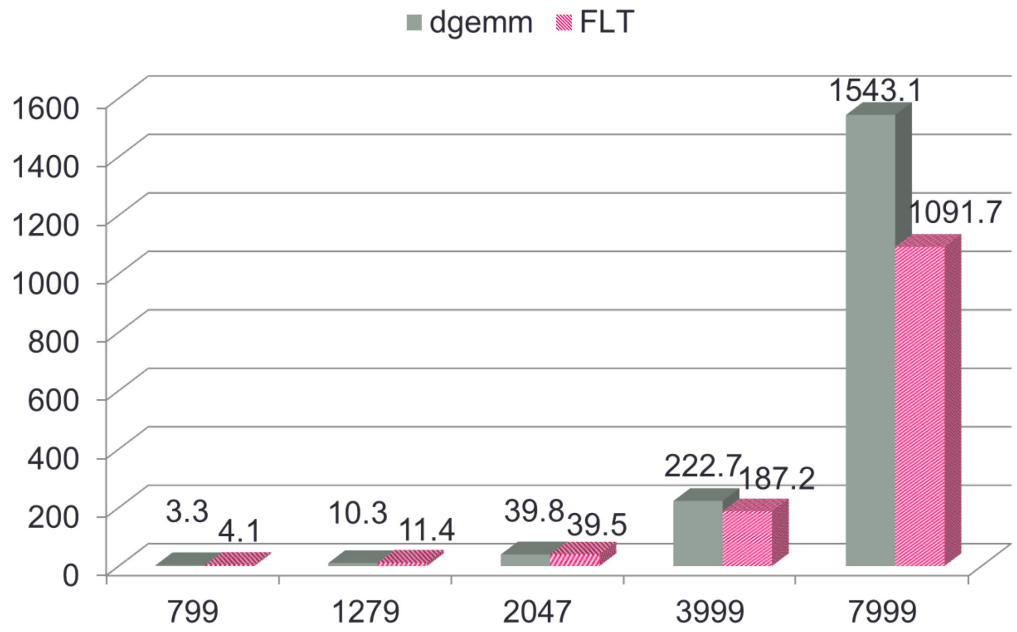

Figure 3: Average wall-clock time compute cost [milli-seconds] per spectral transform, where dgemm represents the old algorithm and FLT the new algorithm

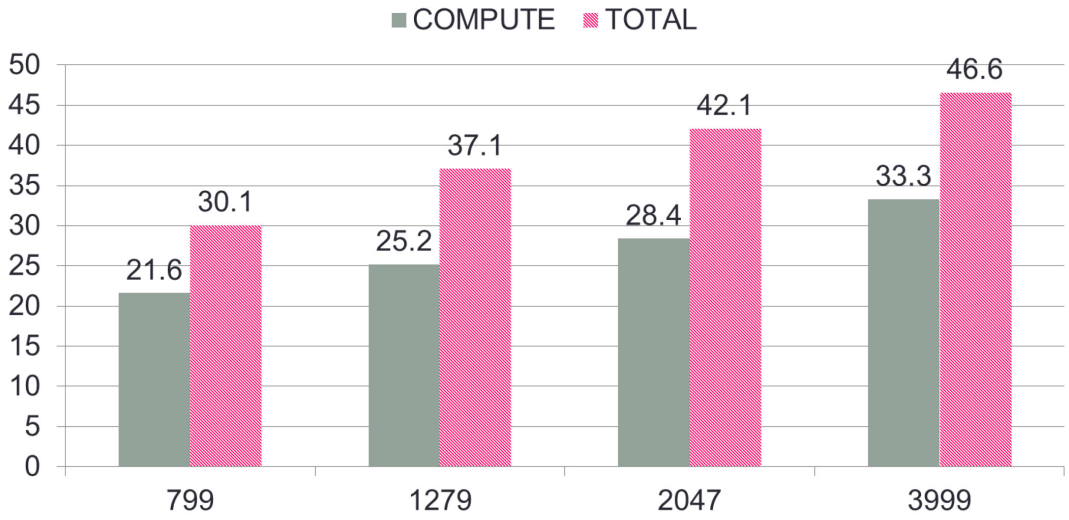

Figure 4: Relative cost of the IFS spectral method (all 91 levels, all non-hydrostatic for comparison)

2. Legendre and Fourier transforms

Within the CRESTA project we are addressing the performance issues described above by using Fortran2008 coarrays to overlap these communications with the computations in the Legendre and Fourier transforms. For the Legendre transforms this is being done per wave number within an

OpenMP parallel region. Figure 5 shows the original and new approaches for the inverse Legendre transform. In the original (OLD) approach the computation (LTINV) and communication (TRMTOL) are done sequentially, with no overlap. In the NEW scheme (using coarrays) each thread is computing and then communicating its computed data to the respective tasks of its 'communicator' group. While Fortran2008 has no coarray groups/teams construct it is nevertheless trivial to compute a mapping to a set of image numbers. The expectation here is that compute (LTINV-blue) and communication (coarray puts-yellow) overlap in time. Experience has shown that the Cray DMAPP library is not thread safe with the CCE compiler version 8.1.5 and earlier releases and a workaround has been used to locate coarray transfers in OMP CRITICAL SECTIONs with a small performance penalty for doing so today. The coarray puts are expected to be non-blocking, and only waited on for completion on a subsequent SYNC IMAGES statement. For the direct Legendre transforms a similar approach is used, the original (OLD) approach and new coarray approach being shown in figure 7.

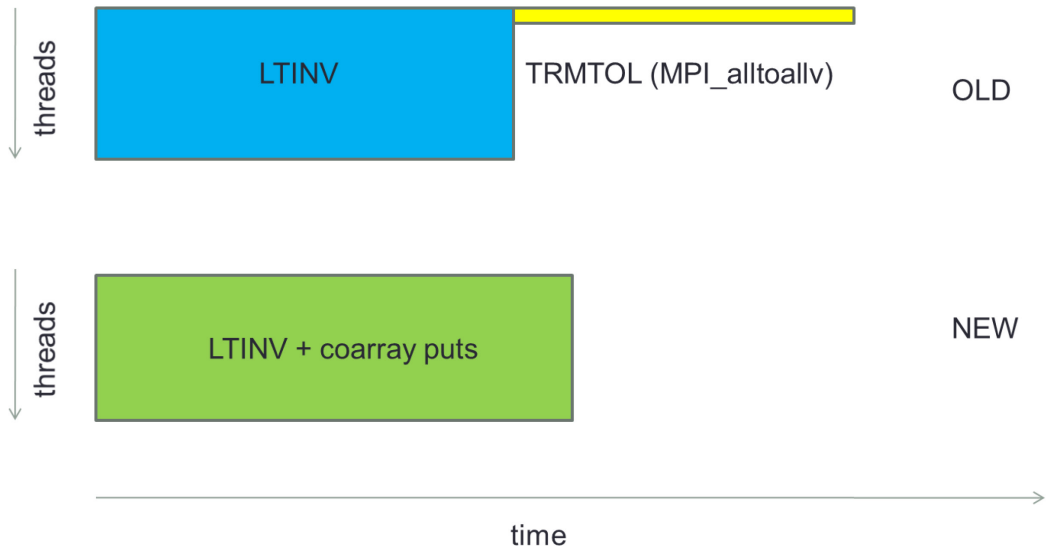

Figure 5: Schematic showing the overlap of LTINV computations with the associated TRMTOL transposition.(blue + yellow = green)

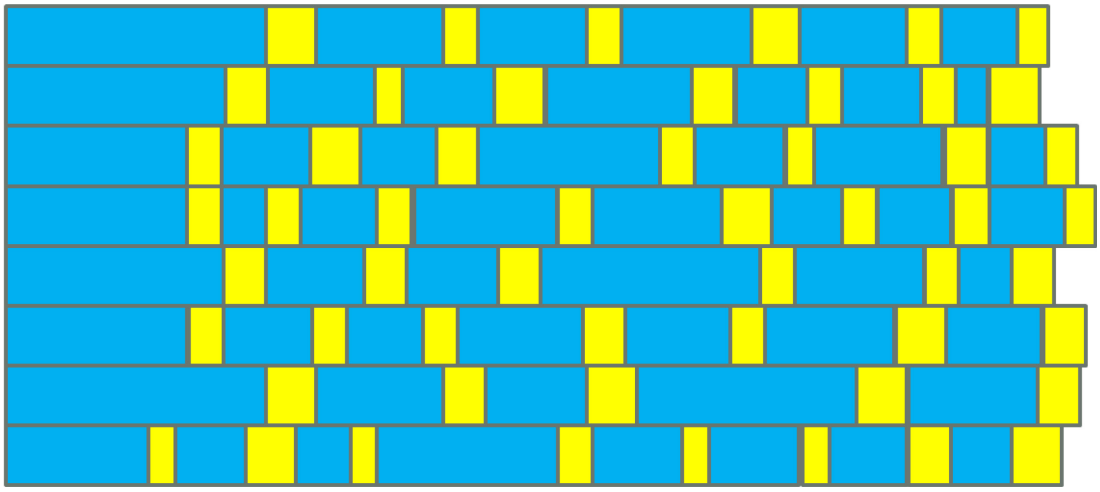

Figure 6: Schematic showing the new IFS approach where Legendre transform computations (blue) are overlapped with associated transpositions (coarray puts-yellow)

Here the coarray gets in each thread are clearly blocking until data arrives and then progress onto computation. In the above description we have focused on the Legendre transform and the PGAS approach to overlap computation with communication, by performing these in a single OpenMP parallel region which operates over spectral wave numbers. For the Fourier transforms a similar scheme is employed, but instead of spectral wave numbers, the Fourier transforms operate over latitudes, where tasks in Fourier space have a subset of latitudes and a subset of atmospheric levels. The TRGTOL and TRLTOG transpositions move data from grid point space which has a EQ_REGIONS partitioning [6][7] shown in Figure 8, which show partitioning for 1024 tasks.

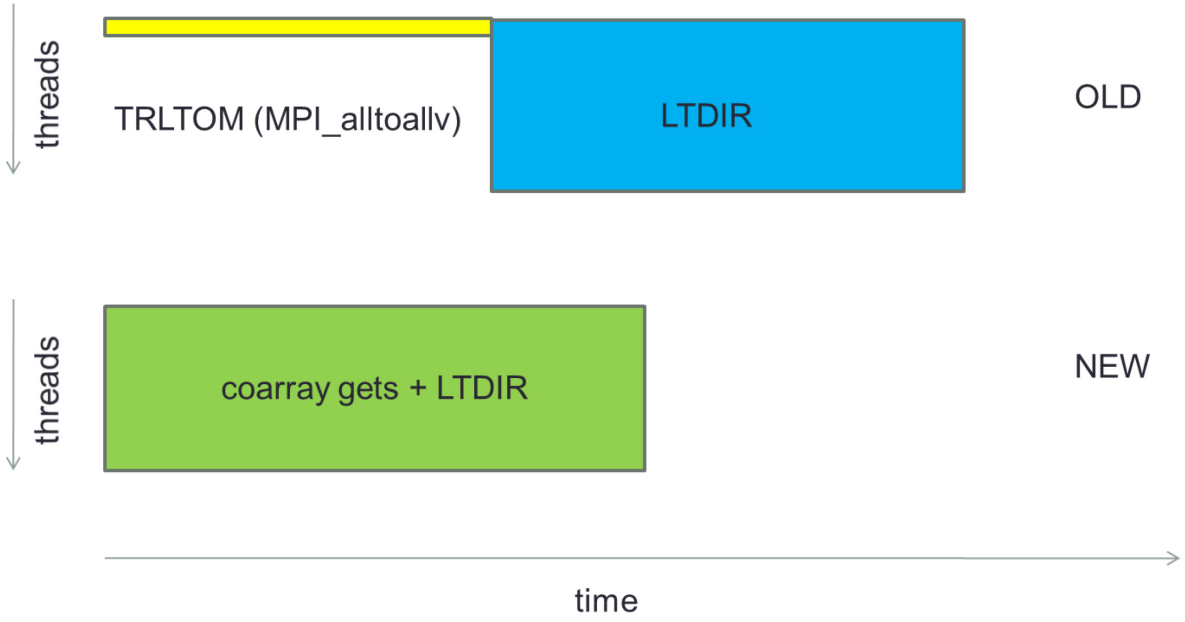

Figure 7: Schematic showing the overlap of LTDIR computations with the associated TRLTOM transposition

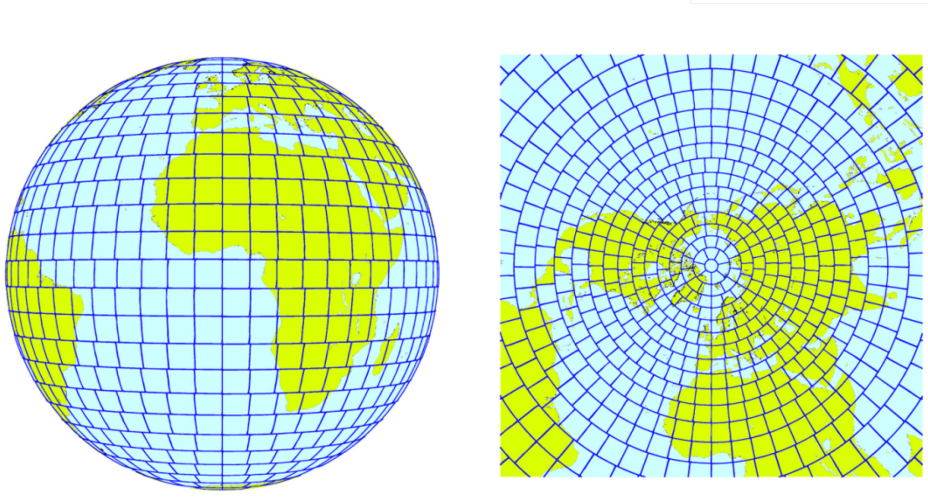

Figure 8: EQ_REGIONS partitioning of grid-point space

3. Semi-Lagrangian Transport

3.1. Original semi-Lagrangian scheme

The semi-implicit semi-Lagrangian (SL) scheme in IFS allows the use of a relatively long time-step as compared with an Eulerian time-stepping scheme. This SL scheme (as shown in Figure 9) involves the use of a halo of data from neighboring MPI tasks which is needed to compute the departure-point and mid-point of the wind trajectory for each grid-point ('arrival' point) in a task's MPI partition. While the communications in the SL scheme are relatively local the downside is that the location of the departure point is only known at run-time and therefore the IFS must assume a worst case geographic distance for the halo extent computed from a maximum assumed wind speed of 400 m/s and the time-step. Today, each task must perform MPI communications for this halo of data before the iterative scheme can execute to determine the departure-point and mid-point of the wind trajectory. This approach is clearly non-scaling as the same halo of data must be communicated, even if a task only has one grid-point (a rather extreme example). To address this non-scaling issue, the SL scheme has been optimised to use Fortran2008 coarrays to only get grid-columns from neighboring tasks as and when they are required in the iterative scheme to compute the departure-point and mid-point of the trajectory (using a 8 point stencil), and also for any other grid-columns needed for the subsequent interpolations (using a 32-point stencil).

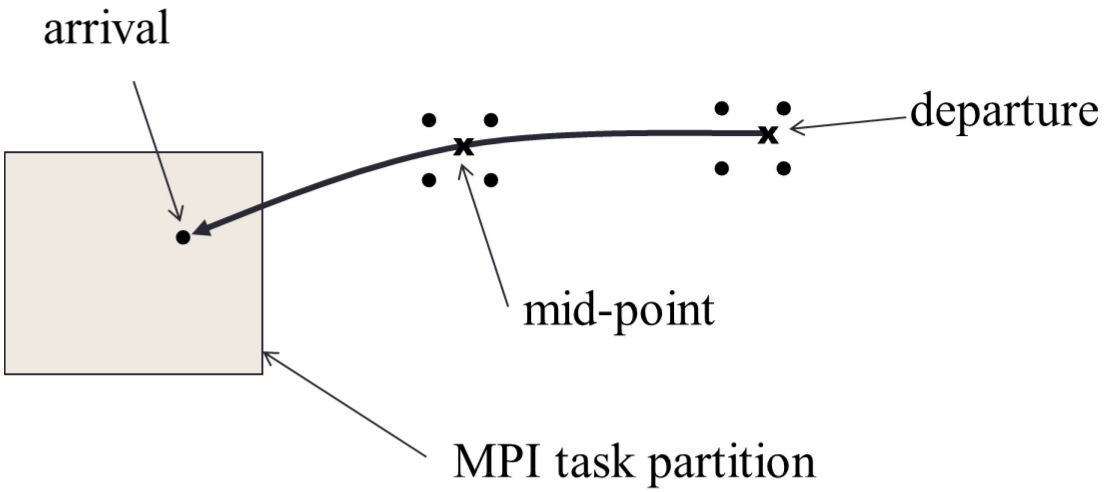

Figure 9: Semi-Lagrangian transport

Figure 10 highlights (in black) the grid-points owned by MPI task 11 which encountered the highest wind speed of 120 metres/s (268 mph) during a 10 day forecast starting 15 Oct 2005. Figure 12 shows a closer view of MPI task 11 (black grid-points) as before, now including a halo of grid-points (marked blue) whose width is determined by a maximum wind speed of 400 meters/sec x the time-step which is 720 seconds (or 288 kilometers). Only the 3 wind vector variables u, v, w are obtained from neighbouring tasks for computing the trajectory. The rest of the variables (26) are obtained from neighbouring tasks, but only for the grid points (marked red in figure 12) that have been identified during the process of computing the trajectory; this is called the on-demand scheme. The semi-Lagrangian interpolations can now be performed. Note that the volume of halo data communicated is dependent on the wind speed and direction in the locality of each task.

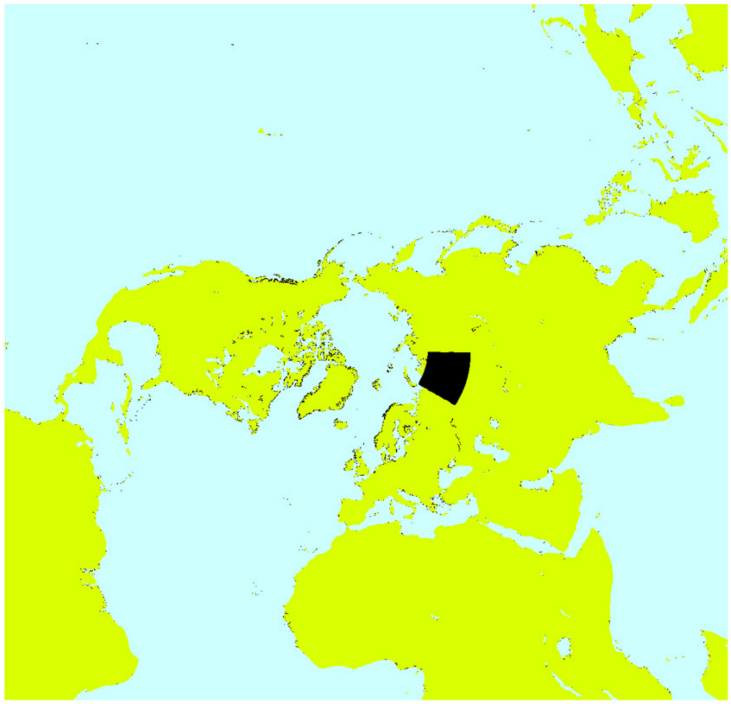

Figure 10: Semi-Lagrangian transport example, for a T799 model run with 256 MPI tasks

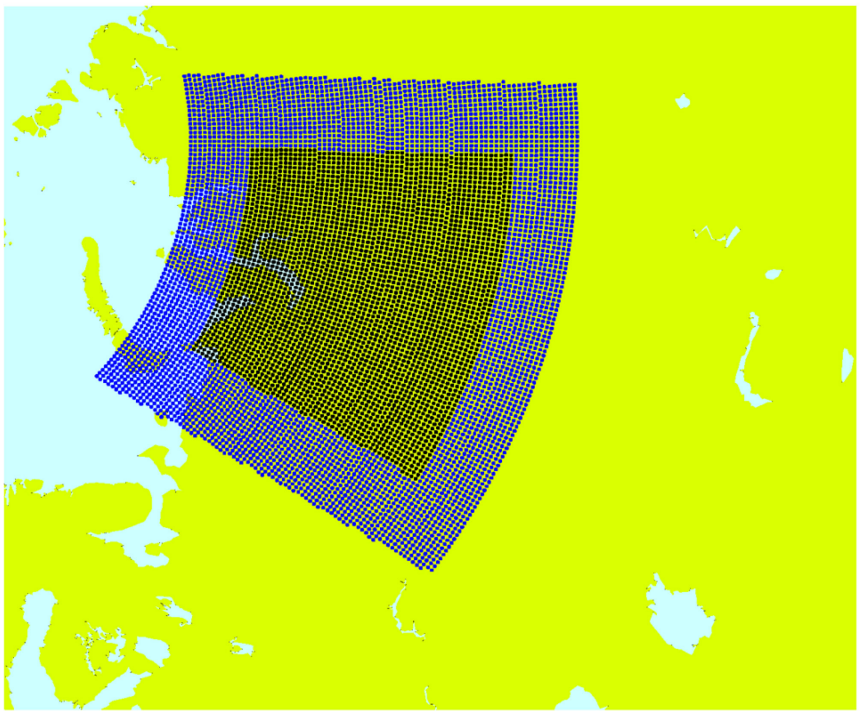

Figure 11: Original semi-Lagrangian transport, showing max wind halo

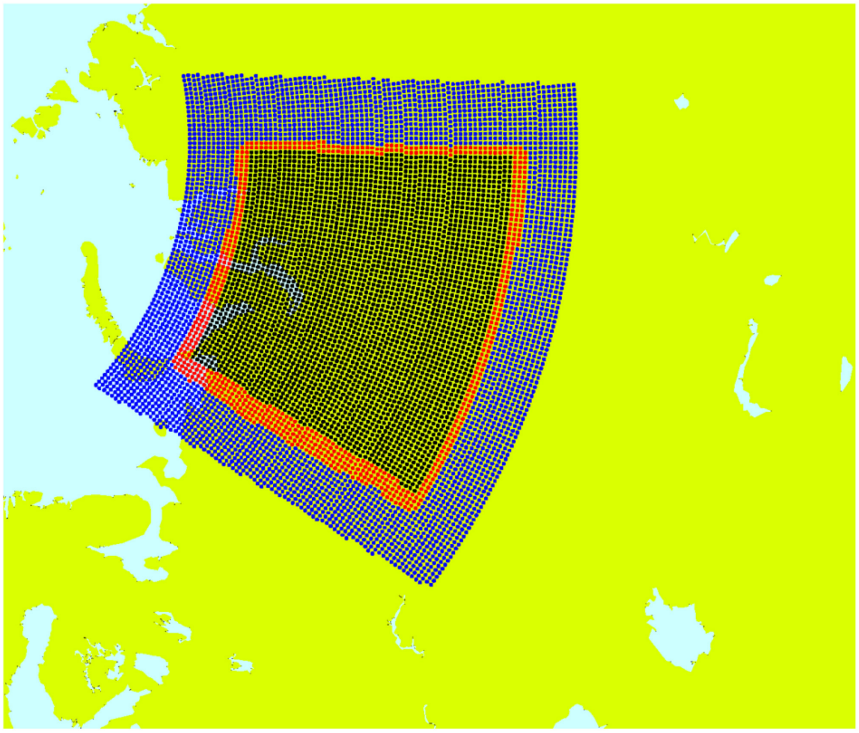

Figure 12: Original semi-Lagrangian transport, showing the halo grid-points that are actually used

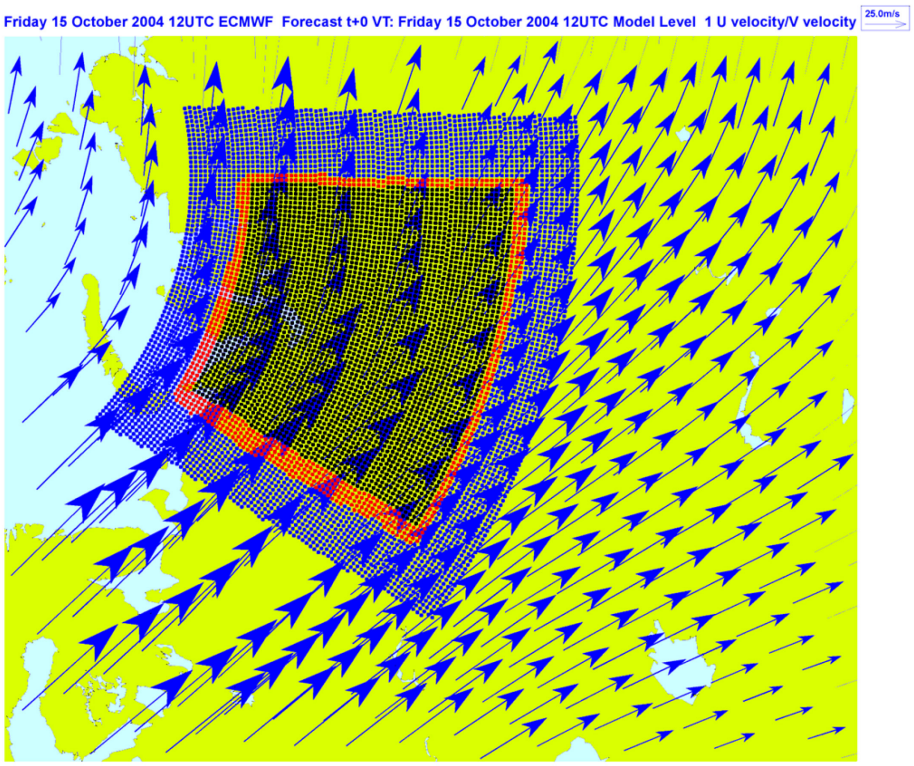

Figure 13: Original semi-Lagrangian transport, showing a wind plot, and greater number of red grid points upwind of MPI task 11

3.2. PGAS semi-Lagrangian scheme

Only the halo grid points (marked red) that are used are communicated by Fortran2008 coarray transfers, as shown in figure 14. No MPI communication is done at all here. Also no max wind blue halo is needed with this approach, with a big saving on the volume of data communicated. A further advantage of this scheme is that all the coarray transfers are done in the same OpenMP parallel region as the computation of the trajectory and subsequent interpolations.

Figure 14: New PGAS (Fortran2008 coarray) semi-Lagrangian transport

4. IFS performance measurements

For an IFS model execution it is crucial that an operational 10 day forecast is completed in under one hour wall time, which is equivalent to 240 forecast days per day (FD/D). Figure 15 shows the evolution of IFS T2047L137 model performance in the CRESTA project. The original unmodified runs used a benchmark release of IFS called RAPS12 (corresponding to an ECMWF internal source cycle 37R3). These runs were performed on HECToR, a Cray XE6, at EPCC, Edinburgh. HECToR and TITAN systems have the same Gemini interconnect and Interlagos AMD cores (32 per node on HECToR and 16 per node on TITAN). The Cray compiler environment (CCE) version was 7.4.4 at the time of the original runs. The performance of the T2047 model was first measured without any source modifications on up to 64K cores and this reached an asymptotic performance of 280 FD/D at around 30K cores. With the CRESTA optimizations, performance was significantly improved with an asymptotic performance of 500 FD/D at around 50K cores. These optimizations included both MPI optimizations mainly to the wave model and the coarray optimizations described in sections 2 and 3. The RAPS12 CRESTA runs in figure 15 were performed using an updated CCE=8.0.6 release. [13] contains more details on the performance gains on HECToR due the individual optimizations (MPI,

coarray and compiler versions used). On TITAN, CCE=8.1.5 was used, in addition to an updated benchmark version called RAPS13 (corresponding to an ECMWF internal source cycle 38R2). It should be noted that the achieved level of performance is in excess of 2 times the requirement for a T2047L137 operational forecast of 240 FD/D which is clearly encouraging. The T2047 model runs on TITAN were about 5 percent slower than on HECToR for comparable numbers of cores. This was mainly due to a new de-aliasing feature in RAPS13 which included 2 extra transforms per time step. Even with this extra cost, it can be seen that this model case continued to show improved performance at higher core counts. The most probable reason for this is the increased communication bandwidth on TITAN as it has fewer Interlagos cores per node compared with HECToR.

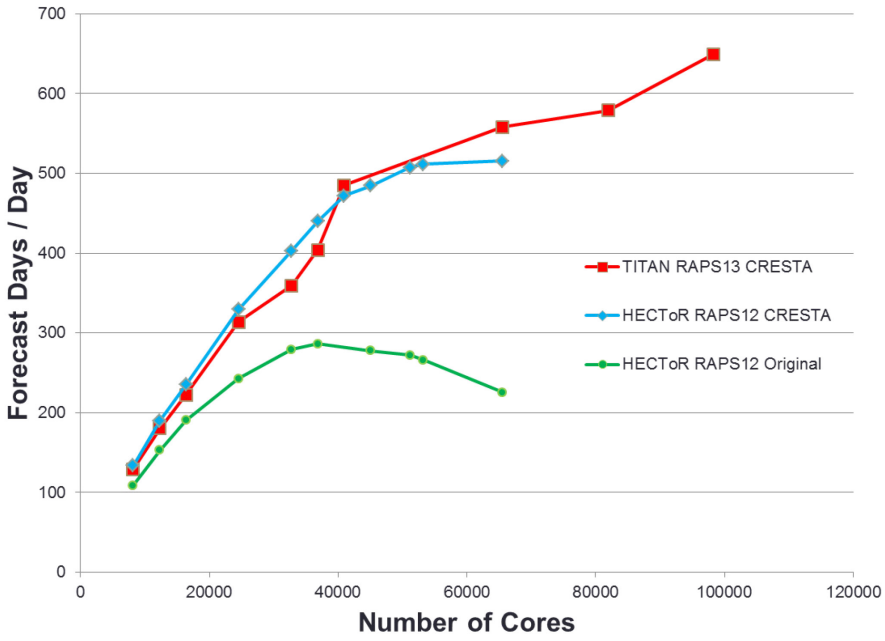

Figure 15: T2047L137 IFS forecast model performance RAPS12 (CY37R3, on HECToR), RAPS13 (CY38R2, on TITAN)

A T3999L137 5 km global model was then run on TITAN, by only using the 16 AMD Interlagos cores available on each node. This model was run with radiation grid of T2047 and radiation transfer calculations computed at every timestep (NRADFR=1). No use was made of the GPGPUs on each node; this was simply due to the code complexity of IFS and its 2 million source lines. This case was run from 24,576 cores up to 212,992 cores, using 16 OpenMP threads per MPI task. It is clear from these runs that this model case did scale but it could do much better. If we assume that a GPGPU port could be done, then we would hope for a 3-4 times improvement in performance given the experience of other NWP codes that have been ported to GPGPUs such as COSMO at MeteoSwiss.

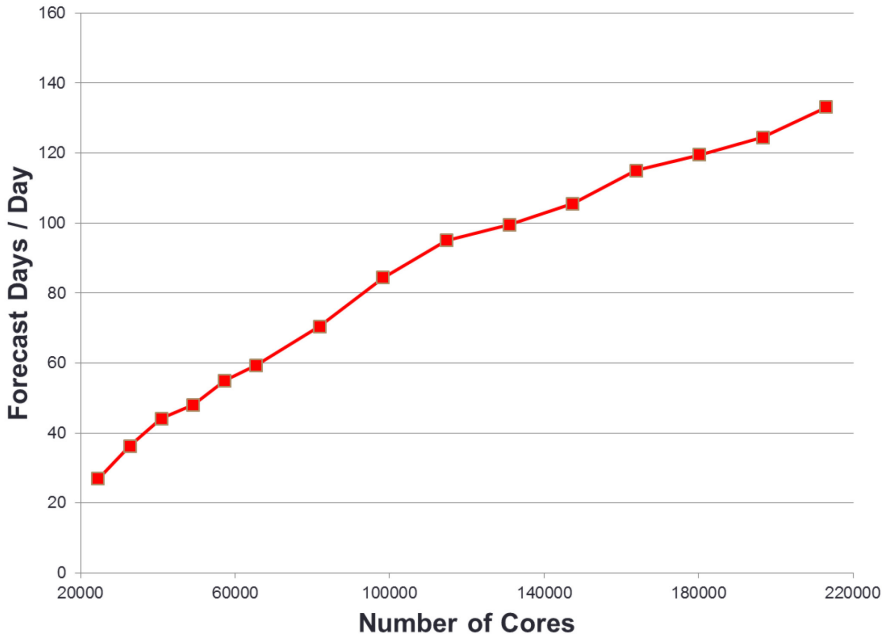

Figure 16: T3999L137 RAPS13 IFS (CY38R2) hydrostatic forecast model performance on TITAN (Cray XK7)

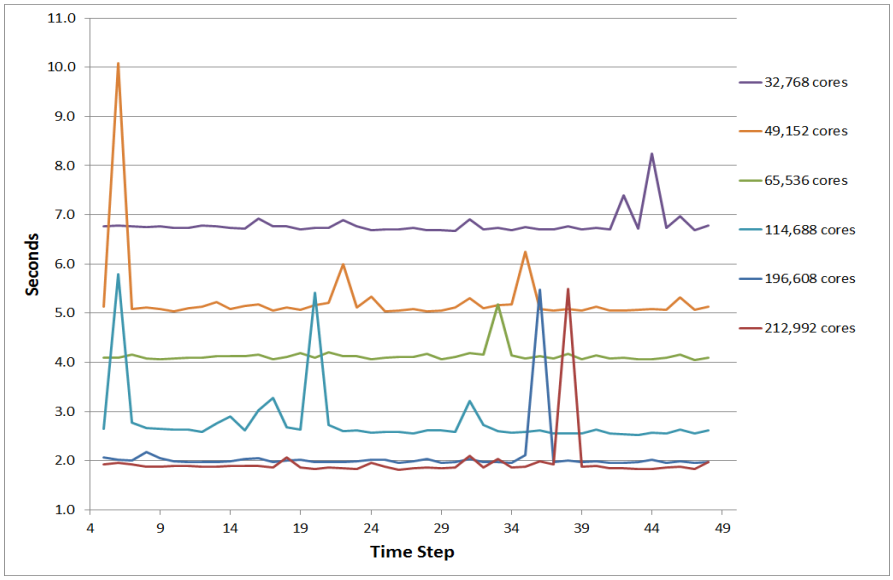

Figure 17: T3999L137 RAPS13 IFS (CY38R2) hydrostatic forecast model on TITAN (Cray XK7), cost per timestep

If we look at the cost of individual time steps we see that TITAN exhibits some large spikes at shown in Figure 17. While we expect a few percent in wall clock variability this is rather excessive. Others have speculated that this is due to the Lustre ping effect, used to check file server health. As RAPS13 IFS benchmark does no I/O beyond initialization this is clearly not something that IFS is causing. Recognising that this effect exists, the performance shown in Figure 16 excludes these peaks.

5. Future work

5.1. Radiation Reorganisation

In the proposed scheme the radiation transfer calculations will execute in parallel with the rest of the model using separate processor cores as shown in Figure 18. From an MPI view, the radiation transfer and model will have separate MPI communicators, while both will clearly be part of MPI_COMM_WORLD. Data between the model and the radiation transfer calculations will be sent asynchronously. An important and necessary requirement for this configuration is that the product of the radiation transfer calculations are returned to the model shifted by one time step. This shift is necessary to allow the radiation transfer calculations to execute completely independently during the time the model executes a full time step. It is expected that both the present and proposed configurations be supported and selected by a runtime namelist variable. It is estimated that the number of processor cores for the proposed configuration will be about double the present configuration for the current default reduced radiation grids, with a reduction in total wall time due to the radiation transfer calculations executing in parallel with the rest of the model. This proposed scheme has the further advantage that it is possible to run the radiation calculations on the same grid as the model grid with only a small increase in total wall time required for the extra MPI communications. Obviously running in this radiation on model grid configuration would require substantially more processor cores for the radiation component to complete within the time-step window. An initial meteorological evaluation has been conducted simulating the proposed configuration, while still running sequentially as in the current configuration. This evaluation has

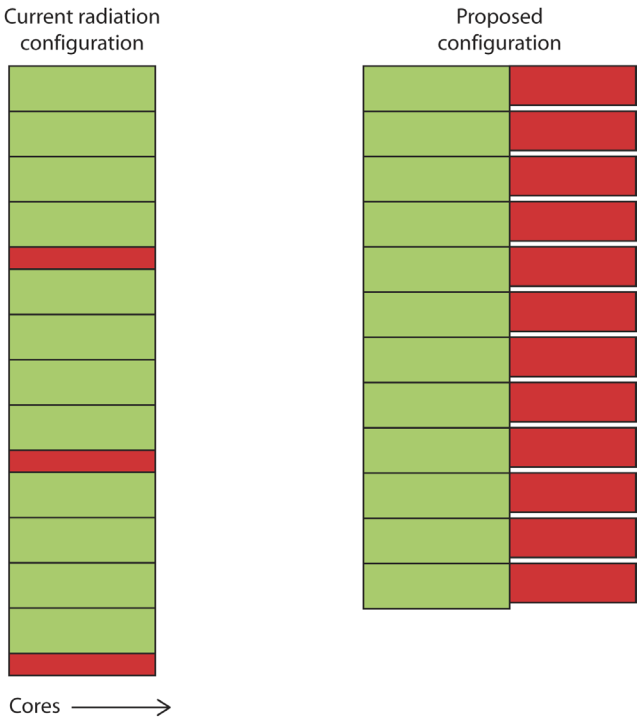

Figure 18: Schematic showing current and proposed radiation configurations. In the current configuration, non-radiation model timesteps are shown in green, and radiation computations are shown in red.

concluded that results are acceptable to allow for a full implementation of the proposed configuration and a more thorough meteorological evaluation ahead of introduction into production. This scheme is further seen as a prototype for running the wave model and land surface schemes in parallel, although each of these must be tested individually in simulation mode, ahead of a full implementation as per the radiation scheme.

5.2. Early work with GPUs

We would like to conduct some experiments with GPUs where we simply offload DGEMMs beyond some size threshold to the GPU and measure any performance improvement. We have started to do this by linking with libsci_acc, and this gave a small improvement of 3 percent for the T3999L137 model case, however, we expect there is more performance to be gained here. As the DGEMMs used in the Legendre transforms involve use of a constant matrix for one of the inputs, then these can be pre-loaded or loaded on-the-fly and left in the GPU memory for the duration of the model execution. These tests should be done with and without the FLT enabled. The purpose of these tests is to see if it is more efficient to use larger more expensive DGEMMs or many smaller DGEMMs used in the FLT butterfly [11] scheme. Also it would be interesting to see if the parallelism in the butterfly scheme can be further exploited on the GPU, i.e. to several DGEMMs executing at the same time in each stage of the butterfly. It is further of interest to explore use of OpenACC to run expensive parts of the radiation transfer calculations on the GPU. This optimisation if successful could allow IFS to run these calculations with the radiation grid at the same resolution as the model, rather the present courser half resolution. It is expected that this would deliver an improvement in forecast skill.

5.3. Use of Coarray Teams

The next Fortran standard (beyond Fortran2008) is expected to support a feature called coarray teams. An issue with the present Fortran2008 standard is that coarrays when allocated must be the same size on each image and the allocation takes place on all images. This behaves like a global operation, something that we want to avoid as we progress to the Exascale. In the next standard we will be able to allocate a coarray team, where the size of images across teams can be different and also the allocation of coarrays in teams is independent of such allocations on other teams. With respect to IFS, this will be more consistent with the IFS parallelisation scheme in Fourier and spectral spaces, and permit the use of dynamic allocation of coarrays in these spaces. Today we minimise the cost of a global coarray allocation by only allocating when a current coarray is too small, and never deallocating it, which is clearly not satisfactory.

5.4. Reorganisation of semi-Lagrangian data structures

The present dominant data structure (SLBUF) in the IFS semi-Lagrangian scheme is organised such that the leading dimension is a slab; a number of consecutive grid-points of full or part latitudes which include a halo of points obtained from near neighbour tasks. The other non-leading dimension of SLBUF is fields, which is a number of variables and their levels. This is not ideal for two reasons, firstly, the 'max-wind by time-step' halo can be relatively large at scale and only partially used leading to poor memory scaling, Secondly, for an efficient coarray implementation we need to obtain a full column (fields) of data (many thousands of words) with a single coarray transfer, so the coarray used today in the semi-Lagrangian scheme requires a transpose of SLBUF to be efficient. Both these

issues can be resolved by reorganising SLBUF with the leading dimension as columns, which will both improve memory scalability and the need for a transpose of SLBUF.

5.5. Wave model (WAM) blocking

Wam uses OpenMP, however it only partitions the leading dimension of the dominant arrays. This is sub-optimal at scale as this approach leads to cache line ping-pong effect and is therefore inefficient. A better approach would be to block these arrays like IFS does for its grid point data structures, the so-called NPROMA blocking where the first array dimension is NPROMA and last array dimension is the number of blocks, these two dimensions replacing the current first dimension. This optimisation will result in improved cache use, performance and scalability.

5.6. Use of Direct Acyclic Graph (DAG) technology

DAG technology is becoming more widely used in high performance computing [14]. To use DAG scheduling in IFS would be a major development effort and we are initially planning to explore use of OMPSs [15] in a toy code that is representative of IFS. Our hope is that OMPSs will permit a more dynamic scheduling of computational and communication tasks across a single node and also to allow some load balancing of computational tasks across nodes. If this is successful in the toy code we will then consider a IFS model implementation.

6. Summary

The ECMWF IFS model has been enhanced to use Fortran2008 coarrays to overlap computation and communication in the context of OpenMP parallel regions. This has been applied to the Legendre transforms, Fourier transforms and the semi-Lagrangian scheme. This approach has resulted in an overall 20 percent performance improvement for a T2047L137 10 km model at 40K cores on HECToR, a Cray XE6. It is expected that the work completed in the first year of three in the CRESTA project has enabled IFS to run at the Petascale. A greater challenge remains to enable IFS for the Exascale. At this point in time it is not clear if implementing all the planned optimisations in section 5 will permit a single IFS forecast model to run at the Exascale. This will become evident as optimisations are applied over time and in particular with access to future HPC architectures. Consequently, continuous investment into the scalability of the IFS is paramount for the future success of ECMWF.

Acknowledgments

The authors would like to thank Bob Numrich, John Reid and Kathy Yelick, for their inspirational talks given on visits to ECMWF in recent years promoting the use of PGAS techniques. Further, without Bob Numrich's ground breaking work in the 1990's (while working for Cray Research) we would not have Fortran2008 coarrays today. Finally, we would like to thank all our colleagues in the CRESTA project for the many fruitful discussions we have had on how we can get IFS and the other CRESTA applications to the Exascale. This work has been supported by the CRESTA project that has received funding from the European Community's Seventh Framework Programme (ICT-2011.9.13) under Grant Agreement no. 287703. In addition, time on TITAN has been provided within the Innovative and Novel Computational Impact on Theory and Experiment (INCITE13) program.

References

[1] Barros, S. R. M., Dent, D., Isaksen, L., Robinson, G., Mozdzynski, G. and Wollenweber, F., 1995: The IFS Model: A parallel production weather code, *Parallel Computing*, **21** 1621-1638.

[2] Hamrud, M., 2010: Report from IFS scalability project. ECMWF Technical Memorandum No. 616, www.ecmwf.int/publications/library/ecpublications/_pdf/tm /601700/tm616_rev.pdf

[3] Salmond, D. and Hamrud, M., 2010: IFS scalability and computational efficiency. www.ecmwf.int/newsevents/meetings/workshops/2010/high_performance_computing_14th /presentations/Salmond_Hamrud.pdf

[4] Mozdzynski, G., 2010: IFS: RAPS11 and model scaling, www.ecmwf.int/newsevents/meetings/workshops/2010/high_performance_computing_14th /presentations/Mozdzynski_scaling.pdf

[5] Salmond, D., 2008: IFS performance on the new IBM Power6 systems at ECMWF. www.ecmwf.int/newsevents/meetings/workshops/2008/high_performance_computing_13th /presentations/Salmond.pdf

[6] Mozdzynski,G., 2008: IFS scaling. www.ecmwf.int/newsevents/meetings/workshops/2008/high_performance_computing_13th /presentations/Mozdzynski.pdf

[7] Leopardi , P., 2006: A Partition of the Unit Sphere of Equal Area and Small Diameter, *Electronic Transactions on Numerical Analysis*, **25**, 309-327, 2006. http://www.maths.unsw.edu.au/applied/files/2005/amr05_18.pdf

[8] Mozdzynski, G., 2006: A new partitioning approach for ECMWF's Integrated Forecasting System (IFS), www.ecmwf.int/newsevents/meetings/workshops/2006/high_performance_computing-12th/pdf/George_Mozdzynski.pdf

[9] Rokhlin, V. and Tygert, M., 2006: Fast algorithms for spherical harmonic expansions, *SIAM Journal on Scientific Computing*, **27** (6): 1903-1928. See: http://cims.nyu.edu/~tygert/sph2.pdf

[10] Tygert, M., 2008: Fast algorithms for spherical harmonic expansions, II, *Journal of Computational Physics*, **227** (8): 4260-4279. http://cims.nyu.edu/~tygert/spharmonic.pdf

[11] Tygert, M., 2010: Fast algorithms for spherical harmonic expansions, III, J*ournal of Computational Physics*, **229** (18): 6181-6192. http://cims.nyu.edu/~tygert/butterfly.pdf

[12] Wedi, N., M. Hamrud, and G. Mozdzynski, 2013: A fast spherical harmonics transform for global NWP and climate models. *Mon. Wea. Rev*. doi:10.1175/MWR-D-13-00016.1, in press.

[13] George Mozdzynski, Mats Hamrud, Nils Wedi, Jens Doleschal, Harvey Richardson, 2012: A PGAS Implementation by Co-design of the ECMWF Integrated Forecasting System (IFS), *High Performance Computing, Networking, Storage and Analysis* (SCC), SC Companion, 652-661, doi: 10.1109/SC.Companion.2012.90

[14] G. Bosilca, A. Bouteiller, A. Danalis, T. Herault, P. Lemarinier, J. Dongarra, 2012: DAGuE: A generic distributed DAG engine for high performance computing, *Parallel Computing*, **38** (12), 37 – 51, 2012.

[15] Duran et al, 2011: OmpSs: A Proposal For Programming Heterogeneous Multi-Core Architectures, *Parallel Process. Lett.* **21**, 173. DOI: 10.1142/S0129626411000151.