Geosci. Model Dev., 8, 769–779, 2015
www.geosci-model-dev.net/8/769/2015/
doi:10.5194/gmd-8-769-2015

Geoscientific Model Development
Open Access

Efficient performance of the Met Office Unified Model v8.2 on Intel Xeon partially used nodes

I. Bermous and P. Steinle

Centre for Australian Weather and Climate Research, the Australian Bureau of Meteorology, Melbourne, Australia

Correspondence to: I. Bermous (i.bermous@bom.gov.au)

Received: 9 September 2014 – Published in Geosci. Model Dev. Discuss.: 6 November 2014
Revised: 5 March 2015 – Accepted: 5 March 2015 – Published: 24 March 2015

Abstract. The atmospheric Unified Model (UM) developed at the UK Met Office is used for weather and climate prediction by forecast teams at a number of international meteorological centres and research institutes on a wide variety of hardware and software environments. Over its 25 year history the UM sources have been optimised for better application performance on a number of High Performance Computing (HPC) systems including NEC SX vector architecture systems and recently the IBM Power6/Power7 platforms. Understanding the influence of the compiler flags, Message Passing Interface (MPI) libraries and run configurations is crucial to achieving the shortest elapsed times for a UM application on any particular HPC system. These aspects are very important for applications that must run within operational time frames. Driving the current study is the HPC industry trend since 1980 for processor arithmetic performance to increase at a faster rate than memory bandwidth. This gap has been growing especially fast for multicore processors in the past 10 years and it can have significant implication for the performance and performance scaling of memory bandwidth intensive applications, such as the UM. Analysis of partially used nodes on Intel Xeon clusters is provided in this paper for short- and medium-range weather forecasting systems using global and limited-area configurations. It is shown that on the Intel Xeon-based clusters the fastest elapsed times and the most efficient system usage can be achieved using partially committed nodes.

1 Introduction

The Unified Model (UM) numerical modelling system (Brown et al., 2012) is used for short- and medium-range weather forecasting, for both high-resolution weather modelling and for relatively coarser climate modelling. Such modelling software requires relatively powerful High Performance Computing (HPC) systems to support operational forecast production. Typically the computing systems have a peak performance comparable to the computer systems included in the TOP500 list released every 6 months. Since September 2009 the UM has been used in the Numerical Weather Prediction (NWP) component of the Australian Community Climate and Earth System Simulator (ACCESS; Puri et al., 2010) at the Australian Bureau of Meteorology (BoM).

Current operational systems at BoM are based on the UM version 7.5 (vn7.5) and the next operational systems upgrade will be based on UM vn8.2. The latter version therefore was used for the work described here.

UM versions include both science and performance upgrades, and extensive evaluation of both types of changes in development mode is required prior to operational implementation. In addition, changes to these systems have major consequences for many downstream applications. For these reasons changing UM versions for operational systems is only done every 1–2 years at the BoM.

Leading HPC systems have from tens of thousands to several million very powerful cores. Since 1980 the trend in HPC development has been for the available processor performance to increase at a greater rate than the available memory bandwidth (Graham et al., 2005, pp. 106–108). The authors concluded that a growing gap between processor and memory performance could become a serious constraint in

performance scaling for memory-bound applications. The gap between processor performance and memory bandwidth has been growing especially quickly for multicore processors in the past 10 years (Wellein et al., 2012). This gap forces the cores on a node to compete for the same memory causing resource contention, which can become a major problem for memory-intensive applications such as the UM.

Increasing the resolution of numerical models is one of the key approaches to improving forecast accuracy. However, in an operational setting these models are constrained to run within a fixed elapsed time on available computing resources. Increasing resolution requires increased computation and therefore the performance efficiency (simply referred to as efficiency in what follows) as measured by the run time on a given number of cores.

Finding the most efficient usage of the system for a particular application and the shortest elapsed times varies depending on whether the application is run on all node cores (fully committed case) or on a subset of the cores available on each node (partially committed case). The placement of the threads and/or Message Passing Interface (MPI) processes across partially committed nodes (sockets) also needs to be done carefully taking into consideration all shared resources available to these cores.

Another practical aspect of the performance analysis discussed in the paper is to estimate whether the coming upgrade for the BoM's operational models will be feasible given the operational time windows and available HPC resources.

The performance analysis described here shows that on some modern HPC systems the shortest run times can be achieved with the usage of partially committed nodes. The concept of using partial nodes for UM applications allows reduced resource contention and improves application performance (Bermous et al., 2013).

2 Description of the models

Regular upgrades to operational NWP systems are driven by the improvements made in both the NWP software and the underlying science. The BoM is currently planning for the next APS2 (Australian Parallel Suite 2) upgrade. The operational suite combines a number of short- and medium-range weather forecasting systems based on the UM software. These systems include the global NWP system (ACCESS-G), the regional NWP system (ACCESS-R, 12 km), the tropical-cyclone forecast system (ACCESS-TC) and several city forecast-only systems (ACCESS-C).

The current APS1 weather forecasting operational systems are based on UM vn7.5 for the global (BoM, 2012) and vn7.6 for the city systems (BoM, 2013). At this stage it is planned that the operational weather forecasting software in APS2 will be upgraded to at least UM vn8.2. This software includes improvements to physical parameterisations, computational performance and performance scaling. Most of the scaling improvement is due to the introduction of asynchronous I/O. With this upgrade the model resolutions for the Global and City models will be increased. An increase in the model resolutions presents a challenge to fit the model runs into the required operational time windows. As a result an initial analysis of the performance measurements of the models is needed. In the current paper we will consider two types of the weather forecasting models: a medium-range N512L70 Global model and a short-range, limited-area "City" model.

2.1 Global N512L70 model

The resolution of the currently operational Global N320 (40 km) model with 70 vertical levels in APS1 will be upgraded to N512 (25 km), 70 levels in APS2. With a finite difference discretisation of the UM mathematical model the latest Global model has a horizontal grid of West–East $\times$ South–North of 1024×769. The existing "New Dynamics" dynamical core with semi-implicit and semi-Lagrangian time integration (Davies et al., 2005) was used.

The operational systems run 4 times daily with two runs for 3 days and two runs for 10 days. In this paper performance scaling analysis for a 3 model day simulation with a 10 min time step was used. With the operational model settings this system produces 137 GB of output data. With relatively large amount of I/O, performance of the N512 global model and especially its performance scalability is significantly affected by the I/O cost at high core counts. Improvements in the scalability can be achieved with the usage of the asynchronous I/O feature (Selwood, 2012) introduced into the model sources from UM release vn7.8. The I/O servers functionality has been continually improving since then.

2.2 UKV city model

APS1 ACCESS-C operational systems nested in the ACCESS-R cover five Australian major city domains: Adelaide, Brisbane (Southeast Queensland), Perth, Sydney and Victoria/Tasmania (Melbourne, Hobart). Each domain in the APS1 ACCESS-C has a horizontal resolution of approximately 4 km with 70 vertical levels. The corresponding short-range city models are set to run 36 h forecasts 4 times daily. A significant horizontal resolution increase is planned for the APS2 upgrade by reducing the horizontal resolution from 4 km to either 1.5 or 2.2 km. As a result some initial performance analysis for the city models is required to find the most efficient run configurations and arrangement within the operational run time schedule.

In this paper an example of a short-range limited-area forecasting is taken from a 1.5 km high-resolution system for Sydney domain (Steinle et al., 2012). The experimental system was created in 2012 and it is currently running 24 times per day. The range of the forecasts provided by the system varies between 12 and 39 h. The corresponding atmospheric

model is nested within the N512L70 global model and based on the UM vn8.2 sources using its variable resolution version (the UKV) with the "New Dynamics" dynamical core.

The UKV modelling concept includes a high resolution of 1.5 km in the inner domain, a relatively coarse resolution of 4 km near the boundary of the main domain and a transition zone of using a variable grid size connecting the inner domain of 1.5 km with the "outer" domain of 4 km.

The Sydney UKV model had a horizontal grid of E–W × N–S of 648 × 720 with 70 vertical levels. The related forecast job was set to run a 25 h simulation with a time step of 50 s giving in total 1800 time steps for a run. The I/O in the job producing only 18 GB per run of the output data is relatively small in comparison to the size of I/O in the global model job. Therefore the usage of I/O servers does not have any major impact on the job performance, even when a large number of cores are utilised.

UKV decomposition constraints

The MPI decomposition in the Unified Model is based on horizontal domain decomposition where each subdomain (MPI process) includes a full set of vertical levels. Within the computational core of the UM, OpenMP is generally used to parallelise loops over vertical dimension.

Due to semi-Lagrangian dynamics implementation, the halo size for each sub-domain limits the maximum number of sub-domains in each direction. With the halo size of 10 grid points used in this study and the horizontal grid size of 648 × 720, the corresponding limits for the MPI decomposition sizes were 42 in the West–East direction and 48 in the South–North direction. Another constraint in the UM model implementation is that the decomposition size in the West–East direction must be an even number.

3 Description of HPC clusters and software used

This section includes hardware specifications and details of the software environment for the HPC clusters used.

3.1 Hardware: specifications of HPC clusters

Numerical results have been obtained on three HPC clusters with Intel® Xeon® processors. The first Constellation Cluster (Solar) with Intel Nehalem processors was installed by Oracle (Sun) in 2009. This system was upgraded by Oracle to a new cluster (Ngamai) with Sandy Bridge processors in 2013.

In addition to the Ngamai system, the BoM also has access to a Fujitsu system (Raijin) installed at the National Computational Infrastructure (NCI) at the Australian National University (ANU) in Canberra, and it is also based on Intel Sandy Bridge processors. The NCI system at 1.2 Pflops was the fastest HPC system in Australia in 2013. Technical characteristics of these three systems are provided in Table 1.

A node Byte/Flop value in the table was calculated as a ratio of the node maximum memory bandwidth and the node peak performance. For the newer Ngamai and Raijin systems this ratio is less than half of that for Solar. All three clusters have Lustre file systems.

It should be noted that turbo boost was enabled in Basic Input/Output System (BIOS) on Raijin only. As per Intel turbo boost 2.0 technology the processor can run at above its base operating frequency which is provided in Table 1 ("Node processor cores" line). Having idle cores on the processor, power that would have been consumed by these idle cores can be redirected to the utilised cores allowing them to run at higher frequencies. Based on the turbo boost additional multipliers a Base Clock Rate (BCLK) can be calculated. For example, we have

```
BCLK = (3.3GHz-2.6GHz)/7 = 100MHz
```

and utilised cores operate at $2.6\,\mathrm{GHz} + 5 \times \mathrm{BCLK} = 3.1\,\mathrm{GHz}$ if an application is run on six cores from the eight cores available on each Raijin processor.

3.2 Software: compiler, MPI library

With the UM vn8.2 sources separate executables were required for each model type: global and limited-area. The Intel compiler version 12.1.8.273 was used to produce UKV executables. In order for the global N512L70 system to use the UM async I/O features the Intel 14.0.1.106 compiler release was needed to avoid crashes due to interaction between the older compiler and the new I/O code.

On the BoM HPC systems an open source implementation of MPI (OpenMPI) was the only library available. The Intel MPI library was available on Raijin; however, testing showed a 10–20 % degradation in performance in comparison with OpenMPI. For this reason and to maintain compatibility between the NCI and BoM systems, OpenMPI was used for all comparisons presented below.

The UKV executable was built with OpenMPI 1.6.5. The usage of a UM async I/O feature requires at least SERIALIZED thread safety level. Therefore the OpenMPI 1.7.4 library was needed to enable the async I/O feature of UM vn8.2.

3.3 Intel compiler options

The Fortran and C sources of the UM were compiled on Sandy Bridge systems Raijin and Ngamai with the following Intel compiler options:

```
-g -traceback -xavx -O3 \
       -fp-model precise                (1)
```

Option `-O3` specifies the highest optimisation level with the Intel compiler. Combination of "`-g -traceback`" options was required in order to get information on a failed subroutine call sequence in the case of a run time crash problem.

Table 1. Intel Xeon Compute System Comparison.

	Solar, BoM	Ngamai, BoM	Raijin, NCI (ANU)
Processor type	Intel Xeon X5570	Intel Xeon E5-2640	Intel Xeon E5-2670
Number of compute nodes	576	576	3592
Total number of cores	4608	6912	57472
InfiniBand interconnect	QDR	QDR	FDR
Memory size per node	24 GB	64 GB	32 GB on 2395 nodes (67 %) 64 GB on 1125 nodes (31 %) 128 GB on 72 nodes (2 %)
Node processor cores	2 × (2.93 GHz, 4-core)	2 × (2.5 GHz, 6-core)	2 × (2.6 GHz, 8-core)
Max turbo frequency	3.333 GHz	3 GHz	3.3 GHz
Node cache size	2 × 8 MB	2 × 15 MB	2 × 20 MB
Memory type	DDR3-1333 MHz	DDR3-1333 MHz	DDR3-1600 MHz
Number of memory channels	3	4	4
Node peak performance (base)	85 GFlops	240 GFlops	332.8 GFlops
Node max memory bandwidth	64 GB s^{-1}	85.3 GB s^{-1}	102.4 GB s^{-1}
Byte/Flop	0.753	0.355	0.308
Turbo boost additional multipliers	2/2/3/3	3/3/4/4/5/5	4/4/5/5/6/6/7/7
Turbo boost	OFF	OFF	ON
Usage of hyper threading	No	No	No

The usage of these two options had no impact on the application performance. Bit reproducibility of the numerical results on a rerun is a critical requirement for the BoM operational systems. For this purpose compilation flag "`-fp-model precise`" (Corden and Kreitzer, 2012) was used in spite of causing a 5–10 % penalty in the UM model performance. An additional pair of compilation options "`-i8 -r8`" was used to compile Fortran sources of the UM. These options make integer, logical, real and complex variables 8 bytes long. Option `-openmp` was specified to compile all model sources and to link the corresponding object files to produce executables used for MPI/OpenMP hybrid parallelism.

Due to a very limited capacity for non-operational jobs on the BoM operational system, Ngamai, the testing and evaluation of the forecast systems prior to their operational implementation is predominantly performed on the relatively larger Raijin system. The BoM share on Raijin is 18.9 %.

Code generated by the compiler option `-xHost` targets the processor type on which it is compiled. On the old Solar Nehalem chip based system, this was equivalent to compiling with `-xsse4.2`. During the porting stage of our executables to the new Sandy Bridge Ngamai and Raijin systems, in order to ensure compatibility of executables across these machines, the `-xHost` compiler option used on Solar was replaced with the compilation flag `-xavx` for advanced vector extensions supporting up to 256-bit vector data and available for Intel Xeon Processor E5 family. It was confirmed, as expected, that adding the `-xHost` compiler option to the set of compilation options (1) did not have any impact on either the numerical results or the model performance.

Compatibility of binaries across systems was achieved by having the same Intel compiler revisions and OpenMPI library versions on both the Ngamai and Raijin systems as well as system libraries dynamically linked to the executables at run time. Note that the usage of Intel compilers and MPI libraries on all systems is via the environment modules package (http://modules.sourceforge.net). It was found empirically that using Intel compiler options (1) as described above provided both compatibility of the executables between the systems and reproducibility of the numerical results between Ngamai and Raijin.

General COMmunications (GCOM) library which provides the interface to MPI communication libraries is supplied with the UM sources. GCOM version 4.2 used with UM vn8.2 was compiled with a lower `-O2` optimisation level.

4 Description of the performance results

Jobs were submitted from login nodes to compute nodes, and job scheduling was done via Sun Grid Engine (SGE) software on Ngamai (earlier on Solar) and Portable Batch Systems Professional Edition (PBS Pro) job scheduler on Raijin.

All model runs were made on very busy multi-user systems using the standard normal queue and a shared Lustre file system for I/O with a potential impact of the I/O contention on the performance results. At the same time each model run used exclusive nodes without being affected by suspend and resume functionality on the systems.

Fluctuations in the elapsed times were usually around 3–5 %, but they were up to 50 % in 3–5 % of the runs. This was particularly noticeable on the Raijin system which had consistent utilisation above 95 %. Support staff at NCI (D. Roberts) investigated this problem. It was found that if cached memory is not being freed when a Non-Uniform

Memory Access (NUMA) node has run out of memory, any new memory used by a program is allocated on the incorrect NUMA node, as a result slowing down access. The UM is particularly sensitive to this issue. An environment setting of

```
OMPI_MCA_hwloc_base_mem_alloc_policy =
  local_only                                    (2)
```

was recommended to include in the batch jobs running UM applications. Setting (2) forces all MPI processes to allocate memory on the correct NUMA node, but if the NUMA node is filled, the page file will be used. As a result the usage of setting (2) greatly improved the stability of the run times on Raijin. Addressing these findings on Ngamai, it appeared that (2) was a default setting on that system. The best run times were initially taken from 3 or 4 runs. If this initial estimate appeared to be an outlier from the estimated performance scaling curve, further runs were made. It is noteworthy that the fluctuations in elapsed times were much higher on all systems when more than 2000 cores were used. The cause of these large fluctuations was not investigated.

Choosing the best timing from a number of runs has been shown to provide reliable estimates of timings under operational conditions – use of the highest priority queue and dedicated file systems to avoid I/O contention and reserved and exclusive use of sufficient nodes to run operational systems. These arrangements result in variations in elapsed times of a few percent.

Starting from the old Solar system it was found that for I/O performance improvement especially for applications with relatively heavy I/O of order at least tens of gigabytes Lustre striping had to be used. Based on the experimentation done on all three systems, Lustre striping with a stripe count of 8 and a stripe size of 4 M in a form of

```
lfs setstripe -s 4M \
    -c 8 < run_directory >
```

was used to optimise I/O performance. Here the `< run_directory >` is a directory where all output files are produced during a model run. One of the criteria on whether striping parameters were set to near-optimal values was based on the consistency of the produced elapsed times for an application running on a relatively busy system.

4.1 UKV performance results

Due to an earlier development of the Unified Model system at the end of the 1980s and beginning of the 1990s on a massively parallel (MPP) system on which each CPU had its own memory, initially the model had only a single level of parallelism using MPI. With the appearance of symmetric multi-processor (SMP) systems in the 1990s and Open Multiprocessing (OpenMP) software the hybrid MPI/OpenMP parallel programming concept which combines MPI across the system nodes and multi-threading with OpenMP within a single node was introduced. This concept uses the shared address space within a single node. From the mid-2000s starting from release 7.0 the hybrid parallel programming paradigm was introduced in the UM code and since then the OpenMP implementation has been consistently improving in the model. Recent studies (Sivalingam, 2014) have shown that even with UM vn8.6 the OpenMP code coverage is limited. Furthermore, the efficiency of pure MPI versus the hybrid parallelism depends on the implementation, the nature of a given problem, the hardware components of the cluster, the network and the available software (compilers, libraries) and the number of used cores. As a result, there is no guarantee hybrid parallelism will improve performance for every model configuration.

4.1.1 Pure MPI vs. MPI/OpenMP hybrid

Comparison of the best elapsed times produced by running the UKV model with the usage of pure MPI and MPI/OpenMP hybrid parallelism on Raijin and Ngamai is given in Figs. 1 and 2 respectively. For simplicity the elapsed times are provided for 4 different decompositions starting from the usage of 384 cores with a stride of 384. The run decompositions were 16×24, 24×32, 32×36 and 32×48 with pure MPI usage. In the case of the hybrid parallelism, two OpenMP threads were used and the related run configurations using the same number of cores as in the pure MPI case were $2 \times 6 \times 32$, $2 \times 12 \times 32$, $2 \times 16 \times 36$ and $2 \times 16 \times 48$, where the first value is the number of threads used. Figures 1 and 2 include results for two cases: fully and partially committed nodes. With partially committed nodes the application was running with the same decomposition as in the fully committed node case, but only a part of each node was used: 8 cores from 12 on Ngamai and 12 cores from 16 cores on Raijin.

With the use of partially committed nodes, the placement/binding of cores to the nodes/sockets should be done in a symmetrical way to give the same number of free cores on each socket. This allows for a better usage of the shared L3 cache on each socket.

Based on the performance results plotted in Fig. 2, the usage of pure MPI gives shorter elapsed times than with the usage of the hybrid parallelism for all decompositions on Ngamai. Figure 1 shows that a similar conclusion can be made for the elapsed times obtained on Raijin, excluding the last point with the usage of 1536 cores. At the same time the shortest elapsed times on Raijin are achieved using pure MPI on partially committed nodes (12 cores-per-node) for the whole range of the used cores. Due to the limited proportion of the UM code that can exploit OpenMP, the use of more than 2 threads (3 or 4) showed no improvement in the performance efficiency.

Comparing the actual set of the obtained elapsed times between the two systems with the usage of pure MPI on fully committed nodes of (2604; 1480; 1103; 942) s on Nga-

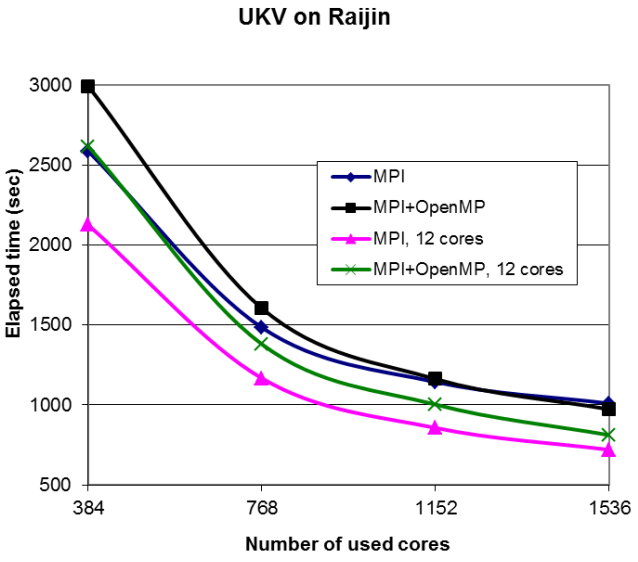

Figure 1. Elapsed times for the UKV model runs on Raijin versus the number of cores actually used in each run.

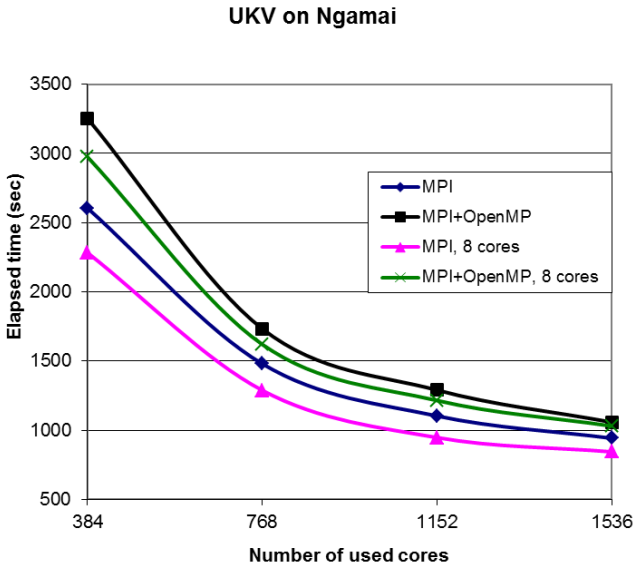

Figure 2. Elapsed times for the UKV model runs on Ngamai versus the number of cores actually used in each run.

mai and (2587; 1484; 1131; 1010) s on Raijin shows that the model performance on Raijin is slightly worse than on Ngamai. At the same time comparing the corresponding elapsed times of (2282; 1288; 947; 844) s on Ngamai and (2125; 1167; 857; 720) s on Raijin with the usage of partially committed nodes for pure MPI, performance and especially performance scaling is better on Raijin. For the same decomposition of 32×48 the elapsed time of 720 s on 2048 reserved cores on Raijin is 14.7 % better than the corresponding elapsed time of 844 s obtained on Ngamai on 2304 reserved cores. This improvement reduces with the number of used cores, with only a 6.9 % faster time for a decomposition of 16×24. Contributing factors to this include the Raijin cores being slightly faster than Ngamai cores and turbo boost not being enabled in BIOS on Ngamai. With the use of partially committed nodes, memory contention between the processes/threads running on the same node is reduced, improving performance for memory-intensive applications. At the same time, enabling turbo boost can increase processor performance substantially, reaching peak speeds of up to 3.1 GHz on Raijin using 12 cores-per-node.

On Raijin the usage of 8 out of 16 cores with hybrid parallelism and 2 OpenMP threads showed between 16.6 % (for high core counts) to 31.0 % (for low core counts) slower run times in the 768–3072 reserved core range in comparison with the corresponding results obtained using pure MPI. On Ngamai the usage of half-committed (6 from 12 cores) nodes with the hybrid parallelism gives two possible configurations for running an application. Firstly there is the symmetrical case with 1 MPI process and 3 OpenMP threads running on each socket. Another option is the non-symmetrical case with 2 MPI processes running on one socket and 1 MPI process running on another socket with 2 threads per each MPI process. Taking into account the limited OpenMP coverage in the UM vn8.2 sources, tests using hybrid parallelism with a symmetrical case of 3 threads or an asymmetrical case with 2 threads on half-committed nodes were not performed on Ngamai.

4.1.2 Fully committed nodes vs. partially committed nodes

Elapsed times for the UKV model with pure MPI usage on partially committed nodes on all three systems are provided in Figs. 3–8. Each pair of figures (Figs. 3–4 for Raijin; Figs. 5–6 for Solar and Figs. 7–8 for Ngamai) shows speedup as a function of the number of cores actually used as well as a function of the reserved cores (i.e. total number of cores allocated to the run, both used and unused). The performance relative to the number of reserved cores is the most important metric, however performance relative to the "Number of used cores" provides additional information on the value of reducing the number of active cores per node. This extra information is particularly relevant to circumstances where the elapsed time is more important than using nodes as efficiently as possible. Examples include climate runs and cases where other restrictions mean that the number of nodes available is not a significant constraint on an application's run time. The related performance information cannot be easily seen on the graph using the "Number of reserved cores" metric.

For example, a 12 cores-per-node case on Raijin and a 6 cores-per-node case on Solar each reserved full nodes (16 and 8 cores respectively), but left a quarter of unused cores. This indicates a requirement of specifying by 1/3 of more cores using `-npersocket` or `-npernode` option of the mpirun command in comparison with the fully committed

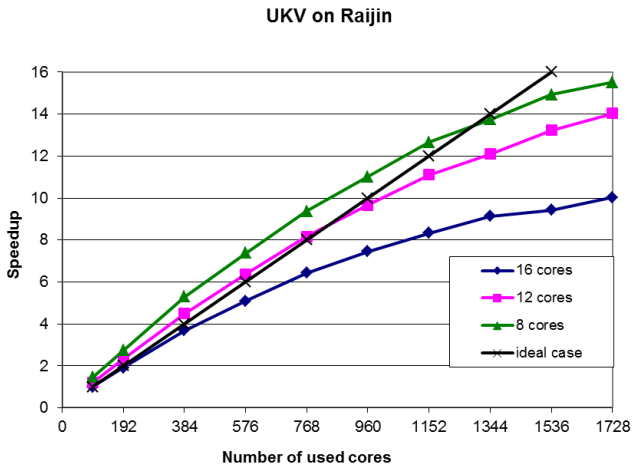

Figure 3. Speedup as a function of number of used cores on Raijin. Speedup was calculated in relation to the elapsed time of 9523 s obtained for a 96-core run on fully committed nodes.

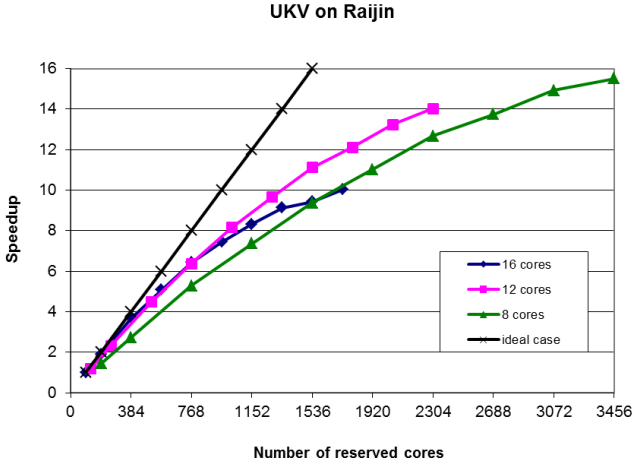

Figure 4. Speedup as a function of number of reserved cores on Raijin. Speedup was calculated in relation to the elapsed time of 9523 s obtained for a 96-core run on fully committed nodes.

case using the same run configuration. In the example of running the model with pure MPI on Raijin and using 12 cores per node the following options:

```
mpirun -npersocket 6 \
    -mca orte_num_sockets 2 \
    -mca orte_num_cores 8 ...
```

were used in the mpirun command with OpenMPI 1.6.5. The last two options specify the number of sockets on a node and the number of cores on each socket. These options were required to avoid bugs found in the OpenMPI 1.6.5 software.

Figure 3 shows the value of partially committed nodes on Raijin where using 12 cores-per-node significantly improves the model scaling. This improvement generally increases as the number of active cores increases. The improvement reaches 28.5 % the performance of using 1728 fully committed nodes. The usage of 8 cores-per-node on Raijin gives an additional performance improvement in comparison with the 12 cores-per-node case varying from 16.8 % at 96 cores to 9.6 % at 1728 cores. Examining the same performance results on the reserved cores basis as in Fig. 4 shows that it is more efficient to use 12 cores-per-node than fully committed nodes just with over 768 cores. On 768 reserved cores the 12 cores-per-node case has value of 1495 s for a 24×24 decomposition and the fully committed node case has value of 1484 s for a 24×32 decomposition.

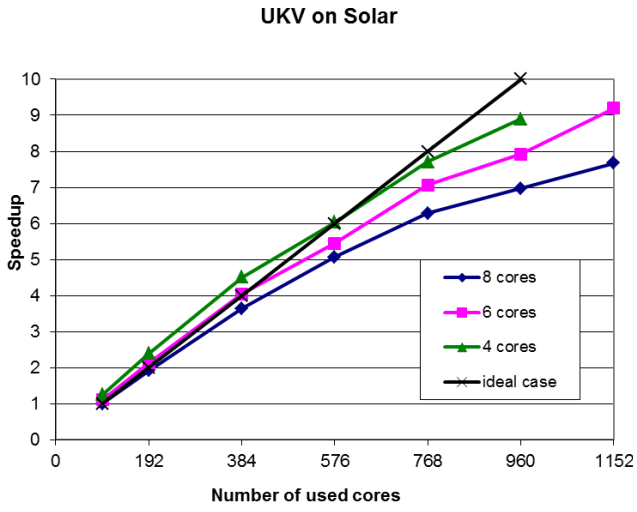

Figure 5. Speedup as a function of number of used cores on Solar. Speedup was calculated in relation to the elapsed time of 11 488 s obtained for a 96-core run on fully committed nodes.

Figure 5 shows that the usage of partially committed nodes on Solar improves the run times with 6 cores-per-node by 6.9–16.5 % and a further reduction of 8.2–11.0 % is achieved with the usage of 4 cores-per-node.

The speedup curves as a function of used cores on Ngamai shown in Fig. 7 indicate that the model runs 10.4–14.6 % faster with 8 cores-per-node. Unlike the other two systems (Raijin and Solar), the use of half-utilised nodes with 6 cores-per-node on Ngamai gives only a very modest reduction of no more than 5.3 %. These latter results indicate that a reduction in memory contention with the 6 cores-per-node case has almost no impact over using 8 cores-per-node.

Speedup curves as functions of the reserved cores for Solar (Fig. 6) and Ngamai (Fig. 8) show that unlike Raijin, the efficiency gains on partial nodes were not achieved on up to 1152 reserved cores on Solar and 1728 on Ngamai. A relatively poor UKV performance on partial nodes especially on Ngamai system in comparison with Raijin was due to the unavailability of turbo boost on that system. Turbo boost would allow active cores to run at up to 16 % higher clock speeds with the usage of 8 cores-per-node on Ngamai.

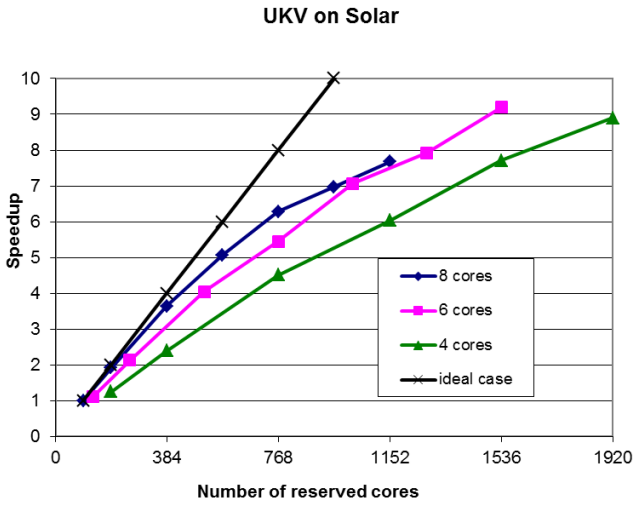

Figure 6. Speedup as a function of number of reserved cores on Solar. Speedup was calculated in relation to the elapsed time of 11 488 s obtained for a 96-core run on fully committed nodes.

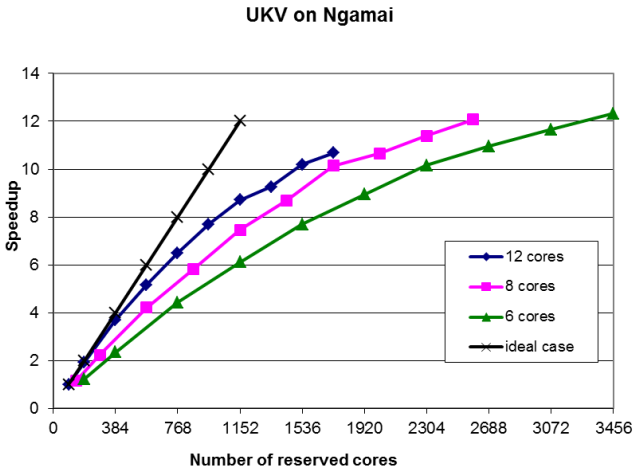

Figure 8. Speedup as a function of number of reserved cores on Ngamai. Speedup was calculated in relation to the elapsed time of 9608 s obtained for a 96-core run on fully committed nodes.

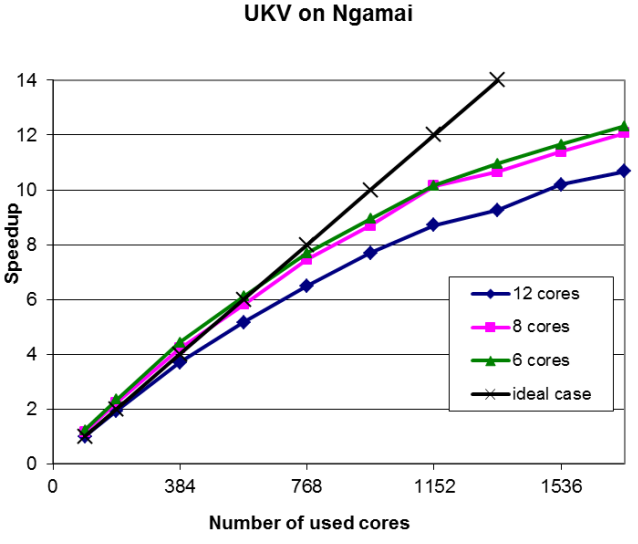

Figure 7. Speedup as a function of number of used cores on Ngamai. Speedup was calculated in relation to the elapsed time of 9608 s obtained for a 96-core run on fully committed nodes.

The shortest run times using fully committed nodes on Ngamai and Raijin with the usage of up to 1728 cores were achieved on a decomposition of 36×48 with pure MPI. The largest allowable decomposition under the constraints provided in Sect. "UKV decomposition constraints" is 42×48 on 2016 cores. Increasing the number of cores from 1728 to 2016 provides further improvements in the elapsed times of 1.7 % on Ngamai and 2.2 % on Raijin. This indicates that the corresponding performance scaling has not begun to level off at the largest allowed decomposition of 42×48. At the same time, with the use of partially committed nodes on Raijin, an elapsed time of 950 s obtained on fully committed nodes for a 36×48 decomposition can be improved by: 28.5 % (679 s) on 2304 reserved cores with 12 cores-per-node or 35.4 % (614 s) on 3456 reserved cores with 8 cores-per-node.

The above-mentioned constraint of 2016 cores on fully committed nodes is applied for pure MPI only. With the usage of hybrid parallelism and 2 OpenMP threads on fully committed nodes, performance of the model is still improving when the number of cores is increasing from 1536 for decomposition of 16×48 to 3072 for a decomposition of 32×48 but the corresponding run times are still greater than the run times obtained with pure MPI on partial nodes using the same number of reserved cores. For example, the use of multi-threading with 2 OpenMP threads and decomposition of 30×48 on 3072 cores gives an elapsed time of 669 s. This run time is improved by 9.1 % (608 s) with the use of pure MPI for a decomposition of 40×48 run on 10 cores-per-node with the same 3072 cores.

4.2 N512L70 performance results

As mentioned earlier, the UM async I/O feature (Selwood, 2012) was used to obtain good performance and especially performance scaling when running the model on more than 1000 cores. Using the UM multi-threaded I/O servers feature, all model runs were made with 2 OpenMP threads. The usage of 3 or even 4 threads showed no improvement in model performance.

Elapsed times from runs without I/O were used as a target for the async I/O case. The run times with full I/O in the fully committed node case were within 5–10 % of those without I/O. Note that on a very busy system such as Raijin some improvement in the run times for a few cases were achieved using different from setting (2) Lustre striping parameters, namely

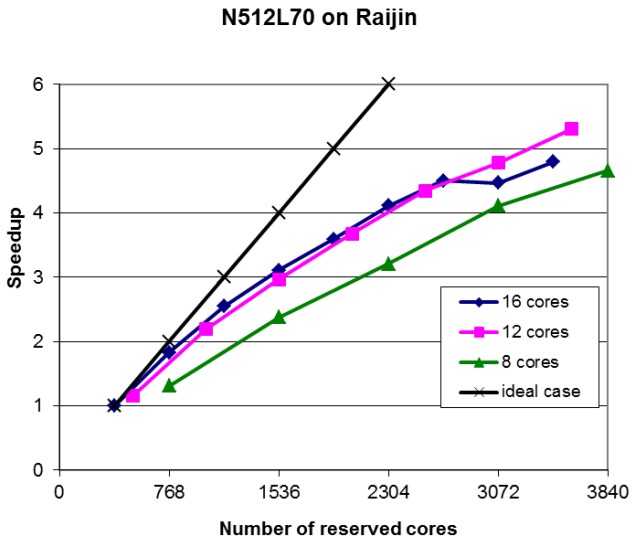

Figure 9. Speedup as a function of number of reserved cores on Raijin. Speedup was calculated in relation to the elapsed time of 2881 s obtained for a 384-core run on fully committed nodes.

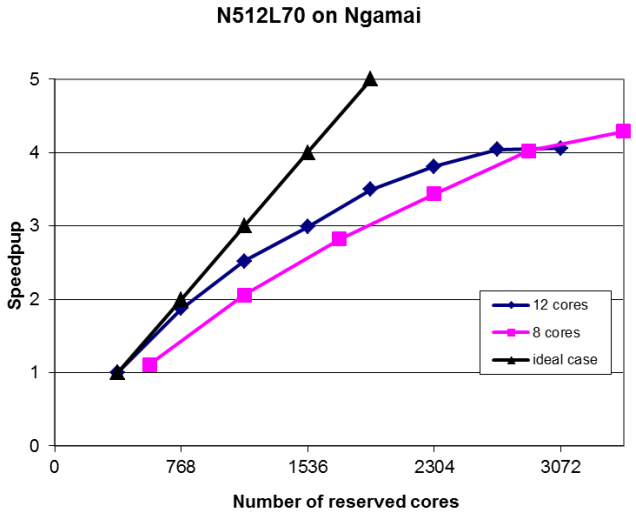

Figure 10. Speedup as a function of number of reserved cores on Ngamai. Speedup was calculated in relation to the elapsed time of 3068 s obtained for a 384-core run on fully committed nodes.

```
lfs setstripe -s 8M \
   -c < number_of_IO_servers > \
   < run_directory >
```

As in the previous case of (2) the values for the Lustre stripe count and the stripe size were found experimentally. Performance results for the model were obtained with up to 3500 cores on Ngamai. Due to large variations in the run times with over 4500 cores on Raijin, the related model performance results are provided for only up to 3840 cores.

The best elapsed times obtained on Raijin are provided in Table 2, where the I/O server configurations of form $m \times n$ included in the third column of the table have the following meaning: m is a number of the I/O server groups, n is a number of I/O tasks per server. For up to 2688 cores the best performance was achieved on fully committed nodes. For 3072 or more cores the best performance results were achieved using partially committed nodes.

Performance scaling of the model as a function of a number of the reserved cores for fully committed node case, 12 cores-per-node and 8 cores-per-node cases is shown in Fig. 9. The curves clearly show that the most efficient system usage with 3072 cores or higher is achieved running the application on partially committed nodes with 12 cores on each node from 16 available. The curves corresponding to 12 cores-per-node and 8 cores-per-node cases show a reasonably good scaling of the model with the usage of up to 4000 cores. Note that using partially committed nodes the model performance is slightly worse when core usage is in the range 384 to 2688.

The best elapsed times obtained on Ngamai are provided in Table 3. On this system in contrast with Raijin, the most efficient usage is achieved using fully committed nodes. Performance scaling of the model as a function of a number of the reserved cores for fully committed node case and with the usage of 8 cores-per-node case is provided in Fig. 10. For the fully committed node case a relatively good performance scaling is achieved with the usage of up to 1920 cores – after that performance scaling degrades slowly with the usage of 2304 and 2688 cores and levels out by 3072 cores. Based on the elapsed times produced with the usage of up to 2688 cores, the most efficient usage of the system is with fully committed nodes. At the same time the usage of 8 cores-per-node for up to 3456 reserved cores has relatively good performance scaling and from the efficiency point of view runs with 3072 reserved cores and higher should use partially committed nodes. Our expectations are that this efficiency in partially used nodes could be improved if turbo boost was enabled.

Performance results for a 6 cores-per-node case are not presented in Fig. 10. As discussed at the end of Sect. 4.1.2 there are two possible run configurations for this case. With symmetrical usage of 3 threads per MPI process run on each socket, the model performance was even worse in comparison with the fully committed node case. At the same time the non-symmetrical usage with 3 MPI processes and 2 threads gave similar performance results as in the 8 cores-per-node case.

5 Conclusions

With a trend in the HPC industry of decreasing the Byte / Flop ratio especially on multicore processors and increasing the number of cores per CPU, the most efficient system usage by memory-intensive applications can be achieved with the usage of partially committed nodes. In other words,

Table 2. The best elapsed times for N512L70 on Raijin using 2 threads.

Total reserved cores	Decomposition	I/O server configuration	Cores used per node	Elapsed time (s)
384	8×23	2×4	16	2881
768	12×31	2×6	16	1575
1152	14×40	2×8	16	1131
1536	16×47	2×8	16	927
1920	16×59	2×8	16	802
2304	18×63	2×8	16	701
2688	20×66	2×12	16	640
3072	18×63	2×8	12	603
3584	20×66	2×12	12	543

Table 3. The best elapsed times for N512L70 on Ngamai using 2 threads on fully committed nodes.

Total reserved cores	Decomposition	I/O server configuration	Elapsed time (s)
384	8×23	2×4	3068
768	12×31	2×6	1639
1152	14×40	2×8	1217
1536	16×47	2×8	1026
1920	18×52	2×12	877
2304	18×63	2×9	805
2688	20×66	2×12	759
3072	22×69	2×9	756

the following factors such as increasing memory bandwidth per active core, reduction in the communication time using less MPI processes and active cores running at higher clock speeds with turbo boost can more than compensate for the reduced number of cores in action. This approach can improve an application performance and most importantly the application performance scaling. A conclusion on whether a specific application should be running on fully or partially committed nodes depends on the application itself as well as on the base operating frequency of the processor and memory bandwidth available per core. Other factors such as availability of turbo boost, hyper-threading and type of node interconnect on the system can also influence the best choice. This study showed that both the regional and global models can run faster if partially committed nodes are used on Raijin. At the same time taking into account the similarities between Raijin and Ngamai systems, there is a reasonable expectation that a similar effect would have been achieved on Ngamai if turbo boost would be available on this system.

The usage of partially committed nodes can further reduce elapsed times for an application when the corresponding performance scaling curve has flattened.

Another example when the use of partially committed nodes can reduce run times is when the performance scaling has not flattened but the number of used cores cannot be increased due to other constraints in the application. This case was illustrated by the UKV model example in Sect. 4.1.2. This approach can be used for climate models based on the UM sources and run at a relatively low horizontal resolution. As per the results of Sect. 4.1.2 the usage of partial nodes can reduce elapsed times significantly. This has a very important practical value for climate research experiments that require many months to complete.

The approach of using partially committed nodes for memory bandwidth-bound applications can have a significant practical value for efficient HPC system usage. In addition, this can also ensure the lowest elapsed times for production runs of time-critical systems. This approach is a very quick method for providing major performance improvements. In contrast, achieving similar improvements through code-related optimisation can be very time consuming and may not even be as productive.

Code availability

The Met Office Unified Model is available for use under licence. The Australian Bureau of Meteorology and CSIRO are licensed to use the UM in collaboration with the Met Office to undertake basic atmospheric process research, produce forecasts, develop the UM code and build and evaluate Earth System models. For further information on how to apply for a licence see http://www.metoffice.gov.uk/research/collaboration/um-collaboration.

The Supplement related to this article is available online at doi:10.5194/gmd-8-769-2015-supplement.

Acknowledgements. The authors thank Yi Xiao (BoM) for providing the UM job settings used in the paper; Robert Bell (CSIRO), Paul Selwood (UK Met Office), Gary Dietachmayer (BoM) and Martyn Corden (Intel) for their useful comments; Jörg Henrichs (Oracle, Australia), Dale Roberts (NCI, ANU) and Ben Menadue (NCI, ANU) for continuous application support provided on all systems. A useful discussion with Dale Roberts on the Lustre striping parameter settings helped improve elapsed times for a few

configurations where the original settings gave small anomalies (a few percent) in scaling results. We thank two anonymous reviewers for the recommendations to improve the original version of the paper.

Edited by: D. Lunt

References

Bermous, I., Henrichs, J., and Naughton, M.: Application performance improvement by use of partial nodes to reduce memory contention, CAWCR Research Letters, 9, 19–22, 2013.

BoM (the Australian Bureau of Meteorology): NMOC Operations Bulletin Number 93, available at: http://www.bom.gov.au/australia/charts/bulletins/apob93.pdf (last access: 3 September 2014), 2012.

BoM (the Australian Bureau of Meteorology): NMOC Operations Bulletin Number 99, available at: http://www.bom.gov.au/australia/charts/bulletins/apob99.pdf (last access: 3 September 2014), 2013.

Brown, A. R., Milton, S., Cullen, M., Golding, B., Mitchell, J., and Shelly, A.: Unified modelling and prediction of weather and climate: a 25 year journey, B. Am. Meteorol. Soc., 93, 1865–1877, 2012.

Corden, M. and Kreitzer, D.: Consistency of Floating-Point Results using the Intel® Compiler or Why doesn't my application always give the same answer?, available at: https://software.intel.com/en-us/articles/consistency-of-floating-point-results-using-the-intel-compiler (last access: 3 September 2014), 2012.

Davies, T., Cullen, M. J. P., Malcolm, A. J., Mawson, M. H., Staniforth, A., White, A. A., and Wood, N.: A new dynamical core for the Met Office's global and regional modelling of the atmosphere, Q. J. Roy. Meteorol. Soc., 131, 1759–1782, 2005.

Graham, S. L., Snir, M., and Patterson, C. A.: Getting Up To Speed: The Future Of Supercomputing. National Academies Press, 289 pp., 2005.

Puri, K., Xiao, Y., Sun, X., Lee, J., Engel, C., Steinle, P., Le, T., Bermous, I., Logan, L., Bowen, R., Sun, Z., Naughton, M., Roff, G., Dietachmayer, G., Sulaiman, A., Dix, M., Rikus, L., Zhu, H., Barras, V., Sims, H., Tingwell, C., Harris, B., Glowacki, T., Chattopadhyay, M., Deschamps, L., and Le Marshall, J.: Preliminary results from Numerical Weather Prediction implementation of ACCESS, CAWCR Research Letters, 5, 15–22, 2010.

Selwood, P.: The Met Office Unified Model I/O Server, ENES Workshop: Scalable IO in climate models, available at: https://verc.enes.org/computing/hpc-collaborations/parallel-i-o/workshop-scalable-io-in-climate-models/presentations/Unified_Model_IO_Server_Paul_Selwood.ppt/view (last access: 3 September 2014), 2012.

Sivalingam, K.: Porting and optimisation of MetUM on ARCHER, The 16th ECMWF Workshop on High Performance Computing in Meteorology, available at: http://www.ecmwf.int/sites/default/files/HPC-WS-Sivalingam.pdf (last access: 28 January 2015), 2014.

Steinle, P., Xiao, Y., Rennie, S., and Wang, X.: SREP Overview: Meso-scale Modelling and Data Assimilation, available at: http://cawcr.gov.au/research/esm/Modelling_WS/10_MEW/Steinle_P.pdf (last access: 3 September 2014), 2012.

Wellein, G., Hager, G., and Kreutzer, M.: Programming Techniques for Supercomputers: Modern processors, available at: http://moodle.rrze.uni-erlangen.de/pluginfile.php/10724/mod_resource/content/2/04_04-23-2014_PTfS.pdf (last access: 22 March 2015), 2014.