Geosci. Model Dev., 7, 2531–2543, 2014
www.geosci-model-dev.net/7/2531/2014/
doi:10.5194/gmd-7-2531-2014

Geoscientific Model Development
Open Access

Implementation and scaling of the fully coupled Terrestrial Systems Modeling Platform (TerrSysMP v1.0) in a massively parallel supercomputing environment – a case study on JUQUEEN (IBM Blue Gene/Q)

F. Gasper[1,2], K. Goergen[2,3,4], P. Shrestha[3], M. Sulis[3], J. Rihani[3], M. Geimer[4], and S. Kollet[1,2]

[1]Agrosphere (IBG-3), Forschungszentrum Jülich GmbH, Jülich, Germany
[2]Centre for High-Performance Scientific Computing in Terrestrial Systems (HPSC TerrSys), ABC/J Geoverbund, Jülich, Germany
[3]Meteorological Institute, University of Bonn, Bonn, Germany
[4]Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Jülich, Germany

Correspondence to: F. Gasper (f.gasper@fz-juelich.de)

Received: 17 April 2014 – Published in Geosci. Model Dev. Discuss.: 3 June 2014
Revised: 11 September 2014 – Accepted: 15 September 2014 – Published: 29 October 2014

Abstract. Continental-scale hyper-resolution simulations constitute a grand challenge in characterizing nonlinear feedbacks of states and fluxes of the coupled water, energy, and biogeochemical cycles of terrestrial systems. Tackling this challenge requires advanced coupling and supercomputing technologies for earth system models that are discussed in this study, utilizing the example of the implementation of the newly developed Terrestrial Systems Modeling Platform (TerrSysMP v1.0) on JUQUEEN (IBM Blue Gene/Q) of the Jülich Supercomputing Centre, Germany. The applied coupling strategies rely on the Multiple Program Multiple Data (MPMD) paradigm using the OASIS suite of external couplers, and require memory and load balancing considerations in the exchange of the coupling fields between different component models and the allocation of computational resources, respectively. Using the advanced profiling and tracing tool Scalasca to determine an optimum load balancing leads to a 19 % speedup. In massively parallel supercomputer environments, the coupler OASIS-MCT is recommended, which resolves memory limitations that may be significant in case of very large computational domains and exchange fields as they occur in these specific test cases and in many applications in terrestrial research. However, model I/O and initialization in the petascale range still require major attention, as they constitute true big data challenges in light of future exascale computing resources. Based on a factor-two speedup due to compiler optimizations, a refactored coupling interface using OASIS-MCT and an optimum load balancing, the problem size in a weak scaling study can be increased by a factor of 64 from 512 to 32 768 processes while maintaining parallel efficiencies above 80 % for the component models.

1 Introduction

In studies of the terrestrial hydrologic, energy and biogeochemical cycles, integrated multi-physics simulation platforms take a central role in characterizing nonlinear interactions, variances and uncertainties of system states and fluxes in reciprocity with observations. Recently developed integrated simulation platforms attempt to honor the complexity of the terrestrial system across multiple time and space scales from the deeper subsurface including groundwater dynamics into the atmosphere (Anyah et al., 2008; Fersch et al., 2013; Keyes et al., 2013; Maxwell et al., 2007, 2011; Shrestha et al., 2014). Technically, the application of these new generations of terrestrial modeling systems over regional climate scale or microscale (e.g., large eddy simulation) requires porting of the system to supercomputing environments, while ensuring ideally a high degree of efficiency in the utilization of, for example, standard Linux clusters and

massively parallel resources alike. With such complex applications, a systematic scaling study and performance analysis including profiling and tracing is crucial for understanding the runtime behavior, to identify optimal model settings, and an efficient identification of bottlenecks in the program's parallelism. On sophisticated leadership-class supercomputers, such as the 28-rack 5.0 Petaflops (Linpack performance) IBM Blue Gene/Q JUQUEEN of the Jülich Supercomputing Centre (JSC) (Germany) used in this study, this is a challenging task, in particular, when a coupled model system consisting of an external coupler integrated with different component models is to be analyzed.

There exist a number of studies dealing with the detailed strong and weak scaling behavior of various simulation platforms in hydrology and reactive solute transport, such as Hammond et al. (2014), Kollet et al. (2010), and Mills et al. (2007). In these studies the focus has been placed on the parallel efficiency of solution algorithms including preconditioners for various classes and systems of partial differential equations in global implicit and explicit solution approaches. In the presented study, the focus is shifted from the analysis of parallel solver and preconditioner performance toward the challenges and parallel efficiency of coupling different component models externally as part of the development of (regional) earth system models.

The challenges and intricacies of coupling technologies of earth system models were reviewed by Valcke et al. (2012), who focused on the central features of different established systems consisting of data transfers, re-gridding, time step management, and parallel efficiency. Prominent examples of coupled modeling systems are the Community Climate System Model, CCSM (Gent, 2006), and the Earth System Modeling Framework, ESMF (Hill et al., 2006), which have also been shown to scale to processor numbers on the order of 10^4. As a matter of fact Dennis et al. (2007) explicitly discuss the application of ultra-high-resolution CCSM on the Blue Gene platform and the required preparations with regard to, for example, memory allocations and parallel I/O due to this unique supercomputer architecture.

The need for high- or hyper-resolution (e.g., 1 km lateral grid spacing over continental computational domains), coupled simulations of the terrestrial system originates from the multi-scale, nonlinear processes and feedbacks of the water, energy, and biogeochemical cycles in and between the subsurface, land surface, and atmosphere (Wood et al., 2011). As a matter of fact, ab initio simulations would require spatial resolutions in the sub-millimeter and sub-second ranges, in order to resolve, for example, non-local reactive transport process in porous media (Yang et al., 2013) and turbulent exchange between the land surface and the atmosphere (Shao et al., 2013). Additionally, heterogeneity of the terrestrial system exists at all spatial scales resulting in variances and residence time distributions of system's states and fluxes spanning orders of magnitude (Kirchner et al., 2000). Thus, resolving all pertinent processes at their respective support scales and adequately honoring cross-scale heterogeneity of the terrestrial system constitutes a grand challenge that may be tackled by efficiently utilizing massively parallel supercomputing environments (Kollet et al., 2010).

The issue that subsurface hydrologic models usually run on a relatively small scale with high resolution, while atmospheric models operate on a very big/continental scale, leads to unsolved questions regarding the coupling of those models. A solution by upscaling the hydrology model to a continental scale lacks adequate scaling laws for the continuity equations of variably saturated subsurface flow (e.g., Richards' equation). Also, the downscaling of the atmospheric model to a regional scale remains challenging due to the representation of turbulence and the lower boundary condition in atmospheric models, that is, the land surface. A straightforward way to combine both models in a soil–vegetation–atmosphere system is to increase the size of the hydrology model to a continental scale, but leaving the resolution high. This requires computational resources only massively parallel supercomputers like JSC's JUQUEEN can provide.

In this study, we present our experiences from porting, tuning, and scaling the parallel Terrestrial Systems Modeling Platform (TerrSysMP) (Shrestha et al., 2014) from commodity Linux clusters to the massively parallel supercomputing environment JUQUEEN, the IBM Blue Gene/Q system of JSC. We aim at addressing and highlighting general technical aspects that have to be considered in designing, porting, or refactoring fully coupled geoscience models to highly scalable high performance computing (HPC) architectures. The study also demonstrates how an optimal resource allocation may be achieved for such a complex modeling system with heterogeneous computing loads between the different component models, and gives an example for a weak scaling study of the highly scalable model system TerrSysMP.

2 TerrSysMP, computer environment, and experiment design

In this section, the modeling platform consisting of the different component models and coupling technologies is introduced, followed by a description of the hardware characteristics of the JUQUEEN (IBM Blue Gene/Q) supercomputer environment used in this study. The modeling platform was instrumented with performance analysis tools, which are also outlined here. The design of the numerical experiments for the ensuing scaling, profiling, and tracing analyses is detailed, including remarks on an ad hoc a priori load balancing of the different component models.

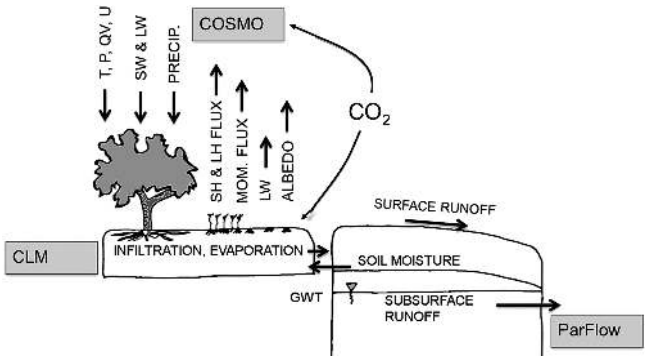

Figure 1. Schematic of interaction processes between TerrSysMP component models.

2.1 The Terrestrial Systems Modeling Platform, TerrSysMP

The parallel Terrestrial Systems Modeling Platform (v1.0) consists of the numerical weather prediction system (COSMO, v4.11) of the German Weather Service (Baldauf et al., 2011), the Community Land Model (CLM, v3.5) (Oleson et al., 2008), and the variably saturated surface–subsurface flow code ParFlow (v3.1) (Jones and Woodward, 2001; Kollet and Maxwell, 2006). For details with regard to the different component models, the reader is referred to the aforementioned publications. In TerrSysMP, these component models were integrated in a scale-consistent way conserving moisture and energy from the subsurface across the land surface into the atmosphere (Fig. 1). The interested reader is referred to Shrestha et al. (2014) for a detailed description of the modeling system. Each component model is itself parallel and has been demonstrated to scale efficiently to a large number of parallel tasks (e.g., Kollet et al., 2010).

In order to couple differently structured component models to simulate complex systems, it is necessary to match a specified interface to exchange fluxes and states. Tailoring this interface exclusively for a certain model environment does not provide the flexibility and compatibility that is needed for various scientific modeling platforms. The obvious solution is a coupling strategy that abstracts that interface via synchronous data exchange, time step management, grid transformation and interpolation methods, and I/O with a low cost and strong stability on different computing environments.

In TerrSysMP, the interface abstraction relies on the Multiple Program Multiple Data (MPMD) execution model, which forms the basis of the external Ocean-Atmosphere-Sea-Ice-Soil coupler, OASIS (Valcke, 2013). With the MPMD functionality, which is offered by most Message Passing Interface (MPI) implementations, it is possible to run several executables within the same global MPI_COMM_WORLD communicator. This functionality enables a coupler that has an external "view" of all component models reflecting the key requirement of high modularity and is especially useful in coupling of component models with fast development cycles and heterogeneous computation loads (Chang et al., 1997). The implementation of the coupler is almost non-invasive. Therefore component models remain independent, which allows for interchangeable executables as a major advantage. Thus, OASIS links the aforementioned component models as independent executables, and can be implemented in two different versions: OASIS3 and OASIS3-MCT (OASIS3 including the Model Coupling Toolkit libraries). In case of OASIS3, the coupler is implemented as an additional independent executable, while in case of OASIS3-MCT the coupler is attached to each individual component model as a library. The impact of coupling with OASIS-3 or OASIS3-MCT in massively parallel computer environments is discussed in detail in following sections.

It is important to note that coupling independent executables based on the MPMD paradigm may confront the developer and user with basic technical drawbacks that need to be considered in the initial design of the modeling platform. For example, the MPMD functionality might not be available or well supported on every machine, especially in case of customized MPI implementations. Additionally, the assigned computational resources, that is, the number of parallel tasks per component executable, are fixed at runtime; thus, load balancing between them has to be performed a priori. Moreover, component models with relatively small computational load, even after load balancing, are constantly blocking resources and use up allocated core hours that cannot be made available to other users.

2.2 Characteristics of JUQUEEN Blue Gene/Q

JUQUEEN is an IBM Blue Gene/Q system with 458 752 cores and 448 TB main memory with a Linpack performance of 5.0 Petaflops. This makes JUQUEEN (November 2013) the eighth fastest supercomputer in the world (Top500.org, 2013).

Supercomputers like JUQUEEN have very special characteristics. Most remarkable is the trade-off in clock rate (1.6 GHz) for smaller power/cooling requirements and improved system reliability. This trade-off in clock rate is compensated by the large number of cores and also the four-way simultaneous multithreading (SMT) of the 64 bit PowerPC A2 processors. The IBM Blue Gene/Q architecture is based on nodes which contain one Central Processing Unit (CPU) with 16 cores and 16 GB main memory; 32 of those nodes are assembled in one (water cooled) node board, which is also the smallest allocation unit for jobs. One rack consists of 8 I/O nodes and 2 midplanes containing 16 node boards each. Compared to standard Linux clusters, the IBM Blue Gene/Q series is an architecture with very low memory per core. The 16 GB RAM per node are distributed to 16 (64 with SMT4) cores and have a static mapping. Thus, each MPI process can only access 1 GB (256 MB with SMT4). While there

is a workaround to enable more memory per core, described later in the text, this is the most challenging constraint and discussed in following sections.

An important feature of Blue Gene/Q is the very fast interconnect, which links all nodes via a 5-D torus (electrical signaling within a midplane, optical signaling beyond midplanes). The 512 nodes of a midplane are connected in a $4 \times 4 \times 4 \times 4 \times 2$ configuration and allow for a very high peak bandwidth ($40\,\mathrm{GB\,s^{-1}}$ per node). The mapping of requested hardware allocations is left to the LoadLeveler job scheduling system, which generally prioritizes large jobs (with maximum wall clock time), but smaller jobs can be placed in the gaps. The mapping to the 5-D torus can be a critical task for communication intensive programs; however, requesting a certain configuration (shape) can result in increased queuing times.

2.3 Performance analysis

Scalasca 1.4.3 was used as a profiling and tracing tool to analyze the runtime behavior of TerrSysMP, identify performance bottlenecks and determine the optimum (static) load balance (i.e., resources allocation for each experimental setup). Scalasca (Geimer et al., 2012) is a portable open-source toolset which can be used to analyze the performance behavior of parallel applications written in C, C++ and Fortran, which are based on the parallel programming interfaces MPI and/or OpenMP. It has been specifically designed for use on large-scale HPC systems such as the IBM Blue Gene series, but is also well-suited for small- and medium-scale systems. Scalasca supports an incremental performance-analysis procedure, combining runtime summaries (profiles) suitable to obtain a performance overview with in-depth studies of concurrent behavior via event tracing. A distinctive feature of Scalasca is its scalable automatic trace analysis (Geimer et al., 2010), which scans event traces of parallel applications for wait states that occur, for example, as the result of unevenly distributed workloads. Such wait states can present major obstacles to achieving good performance.

The typical Scalasca workflow is as follows: before any performance data can be collected, the target application is instrumented; that is, probes are inserted into the application to intercept important events. Scalasca supports various ways to accomplish this task, for example, using automatic compiler-based instrumentation, library interposition, or via source-to-source transformation. At runtime, these probes trigger the collection of performance events to – by default – generate a profile measurement providing a performance overview. Based on the initial profile results, the measurement configuration can be optimized to reduce measurement perturbation, for example, by filtering small but frequently executed functions. In-depth analyses of the performance behavior can then be performed by collecting and automatically analyzing event traces, which allow one to distinguish between wait states and actual communication or synchronization time as well as to determine their root causes and activities on the critical path (Böhme et al., 2010, 2012).

To obtain information about the allocated memory, only an interface provided by IBM can be used (#include <spi/include/kernel/memory.h>). This is due to the fact that the compute nodes of JUQUEEN use a specific compute node kernel with reduced functionality that does not offer generic memory interfaces, making the use of conventional memory tools impossible.

2.4 Scaling study experimental design

To identify scalability and performance limitations of TerrSysMP when going to very large model domains either by increasing the spatial resolution or expanding the model domain to, for example, continental scales, a weak scaling study with an idealized test case was developed. In the scaling study, the two-dimensional horizontal extent of the model domain (nx, ny) was increased by a factor of 4 for each scaling step (doubling every dimension). The number of cells in vertical dimension, nz, remained constant for every scaling step with ParFlow $nz = 30$, CLM $nz = 10$, and COSMO $nz = 40$. All models use a two-dimensional processor topology, and in the first scaling step one Blue Gene/Q node board with 32 nodes and 512 physical CPU cores was used. The allocated resources are doubled in each dimension as well, and thus the patch size (grid cells per task) for every MPI rank remains constant throughout the scaling experiment.

Time stepping remains constant across all scaling steps and is based on the physical processes simulated and applied solution algorithms of the different component models. In the atmospheric model COSMO, the time step size, Δt, is strongly determined by the spatial discretization and was fixed at $\Delta t = 10\,\mathrm{s}$. Time integration of the relevant exchange fluxes with the land surface and subsurface model CLM and ParFlow is performed by OASIS over a 900 s interval, which simultaneously constitutes the constant time step size of CLM and ParFlow. Note that, in the presented scaling study, file I/O is disabled as far as possible. The reason for this is the missing parallel file I/O in some component models and memory limitations in case of large domain sizes.

The scaling study is performed with two different setups in terms of grid size and processor allocation (Table 1).

1. In the first setup, a grid size, n, is used that is closely related to real-data test cases used by Shrestha et al. (2014) for development and testing of TerrSysMP. The initial scaling step consists of $nx = ny = 288$ grid cells for CLM and ParFlow with a lateral spatial discretization of $\Delta x = \Delta y = 0.5\,\mathrm{km}$ and $nx = ny = 144$ for COSMO with a lateral spatial discretization of $\Delta x = \Delta y = 1\,\mathrm{km}$. An optimal hardware distribution was used, which was predicted with profiles from the analysis tool Scalasca and the method described in Sect. 3.2. The profiling showed minimal wait states (critical path) with a

Table 1. Summary of experimental design setup for scaling studies.

(a)				
Design 1				
Scaling step	1	2	3	4
#grid cells per dimension (COSMO/CLM/ParFlow)	144/288/288	288/576/576	576/1152/1152	1152/2304/2304
#processors (COSMO/CLM/ParFlow)	$24 \times 16/8 \times 8/8 \times 8$	$48 \times 32/16 \times 16/16 \times 16$	$96 \times 64/32 \times 32/32 \times 32$	$192 \times 128/64 \times 64/64 \times 64$
cores	512	2048	8192	32 768
node boards	1	4	16	64
midplanes	1/16	1/4	1	4 (two racks)
(b)				
Design 2				
Scaling step	1	2	3	4
#grid cells per dimension (COSMO/CLM/ParFlow)	128/256/256	256/512/512	512/1024/1024	1024/2048/2048
#processors (COSMO/CLM/ParFlow)	$16 \times 16/8 \times 16/8 \times 16$	$32 \times 32/32 \times 16/32 \times 16$	$64 \times 64/64 \times 32/64 \times 32$	$128 \times 128/128 \times 64/128 \times 64$
cores	512	2048	8192	32 768
node boards	1	4	16	64
midplanes	1/16	1/4	1	4 (two racks)

processor allocation (starting with one node board/512 MPI ranks) of $8 \times 8 = 64$ for CLM and ParFlow and $24 \times 16 = 384$ for COSMO. This results in patch sizes of $(288 \times 288 \times 30)/64 = 38\,880$ grid cells for ParFlow, $(288 \times 288 \times 10)/64 = 12\,960$ for CLM and $(144 \times 144 \times 40)/384 = 2160$ for COSMO.

2. In the second scaling setup, the grid sizes, n, and number of processors, np, are expressed as a power of 2 to provide a more standardized experiment for better comparability. In this setup, the computational resource allocation is not possible in an optimal sense, since the load of the component models is roughly distributed as follows: 75 %/12.5 %/12.5 %, which does not follow powers of 2. The first step has grid sizes of 256×256 for ParFlow and CLM and 128×128 for COSMO. The 512 MPI ranks (one node board) are distributed as $16 \times 8 = 128$ for ParFlow and CLM and $16 \times 16 = 256$ for COSMO. This results in patch sizes of $(256 \times 256 \times 30)/128 = 15\,360$ grid cells for ParFlow, $(256 \times 256 \times 10)/128 = 5120$ for CLM and $(128 \times 128 \times 40)/256 = 2560$ for COSMO.

In both setups, the parallel efficiency $E_{\mathrm{nb}}(n)$ [%] in our study is defined as

$$E_{\mathrm{nb}}(n) = \frac{1}{\mathrm{nb}} \cdot \frac{T_1(n)}{T_{\mathrm{nb}}(n)} \cdot 100, \quad (1)$$

where $T_1(n)$ is the runtime with one node board and $T_{\mathrm{nb}}(n)$ the runtime with nb node boards and the problem size n. Thus, in case of perfect parallel weak scaling without communication overhead, the simulation platform would exhibit an efficiency of $E_{\mathrm{nb}}(n) = 100\,\%$ for nb node boards and the problem size n.

3 Results

In this section, the implementation and building process of TerrSysMP is described, followed by an introduction of an ad hoc load balancing approach for MPMD programs with the usage of performance analysis tools. The execution of the designed scaling study and the reason why first attempts failed due to memory restrictions are also presented in this section. This is followed by the advancements with the new OASIS version with results and discussion.

3.1 TerrSysMP implementation

For coupled systems with independently developed model codes, it is unlikely that all components are initially ready and efficient for various computing sites, compilers and libraries. In order to reach an optimum single-node and component-model performance, TerrSysMP was initially ported to use IBM XL compilers that may produce executables with the most efficient hardware utilization. To improve the usability of the complete model system, which is developed in a standard Linux cluster environment, fully automatized script-based install procedures allow for a very efficient and fast application deployment. The most current release version of the TerrSysMP system is retrieved from a master GIT repository and adjusted for the build environment for the machine, in our case JUQUEEN, that is, little/big endianness,

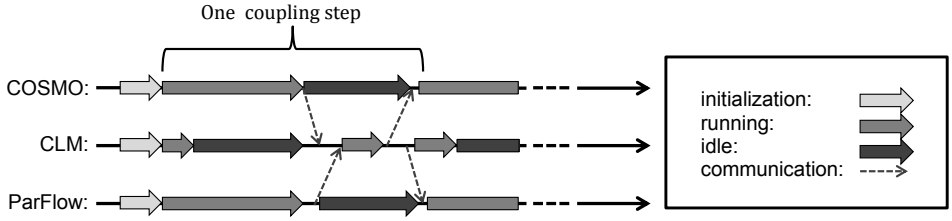

Figure 2. Schematic of the synchronization and communication structure. CLM receives first from COSMO before receiving from ParFlow; thus, wait states in ParFlow are indicating an overloaded COSMO. CLM calculation is very fast, but COSMO and ParFlow are idle during this time. CLM sends first to COSMO before sending to ParFlow. CLM is idle during COSMO and ParFlow computation.

library paths and data structures, similar to the GNU autoconf software configuration package. Optional/experimental features (e.g., OASIS3-MCT) are also available for integration during this procedure. In a second step, the complete model system is built and the runtime environment (model settings, forcing data and job scripts, etc.) is set up. In order to preserve portability and legacy code, TerrSysMP does not make use of hardware intrinsics or interfaces to IBM APIs (i.e., L1P prefetcher, atomic operations, etc.). However, there are compiler options, which guide the compiler to make use of architecture-specific benefits and help with constraints, in our case: $-O3\ -qhot\ -qarch=qp\ -qtune=qp$. The usage of these options enables a speedup of roughly a factor of 2 for TerrSysMP. To allow for easy regression testing during model development and for first-time users familiarizing themselves with the system, forcing data and model settings for well-defined real data and idealized test cases as well as reference results are provided.

3.2 Optimum resource allocations for MPMD

As already briefly mentioned in the explanation of TerrSysMP's coupling scheme in Sect. 2.1, in most MPMD implementations, the resource allocation or association of hardware nodes to a certain application is fixed during runtime. Usually, in many MPI implementations a different number of executables is started through the invocation of the MPI parallel job launcher; processes are then mapped onto the computational resources allocated by the job scheduler. On IBM Blue Gene/Q, a mapfile has to be used in conjunction with MPMD to explicitly assign MPI ranks to the actual CPU cores. This mapfile may either be set up before job submission to optimize the communication pattern on the 5-D torus network topology of the BG/Q, or the resources are assigned automatically by the scheduler. The latter was used to define the mapfiles. In order to allocate the resources in a performant way, an algorithm is used that first queries the assigned shape and then arranges the resources in a way that an executable is distributed to adjacent nodes. This usually ensures low latencies within the 5-D torus interconnect.

This setup combined with CPU affinity means that a load balancing between the component models during runtime is not possible and assigned resources are fixed. Thus, no dynamic load-balancing algorithms are applicable. Since simulations may run for several hours, unbalanced resource assignments have a strong impact on the parallel efficiency. Therefore, determining an approximate load for every component model and applying a static load balancing in advance is a necessary condition for an efficient utilization of resources. For TerrSysMP, using a profiling tool (on JUQUEEN for example Scalasca) in conjunction with a graphical tool to visualize the profile (here CUBE-QT; Song and Wolf, 2004) provides a complete picture of the time spent within the individual models and routines. With detailed knowledge of the synchronization and communication structure (Fig. 2) of the coupled system (or a critical-path analysis available in the newest Scalasca implementation), one can identify which models are waiting for completion of others and, thus, are under- or overloaded. For example, if ParFlow has 30 % LateSender waiting time in the corresponding receive call from CLM and CLM is also waiting, it is clear that COSMO needs about 30 % more resources from, for example, ParFlow. This might have to be iterated a few times, especially if the speedup saturates.

Figure 3 is a showcase for this workflow and shows two CUBE-QT screenshots of the fully coupled TerrSysMP. In Fig. 3a, the load is not ideally balanced and the topology view (right) shows more cores with higher load in the relevant functions than in the optimized balancing of Fig. 3b. In both screenshots, the metric LateSender was chosen and, thus, the displayed (accumulated) timings are equivalent to this particular wait state (receiver waits for sender).

With this complete picture of TerrSysMP, it was possible to determine an improved load balance for the test setup 1 in Sect. 2.4 and also characteristic real-data test cases reacting positively to this approach. For example, compared to established balancing methodologies based on component intrinsic timing routines a 19 % speedup was reached in this example. However, this method is only precise if the actual setup is traced/profiled. In order to determine the distribution for our test setup 1, 24 h were traced in scaling step 1. Since we are simulating an idealized test case (flat geometry with homogeneous vegetation), we assumed negligible influence on the load distribution with increasing domain sizes.

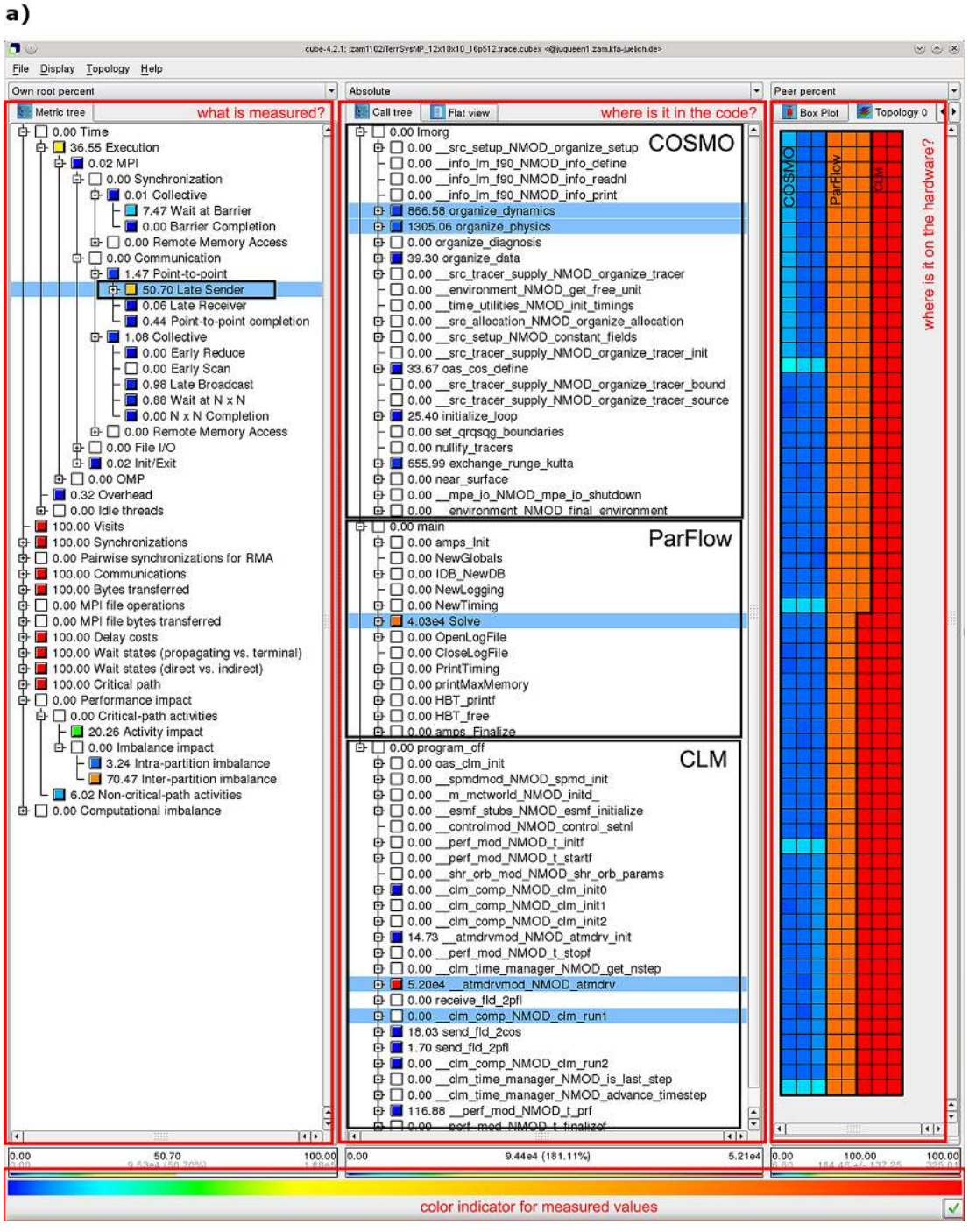

Figure 3a. CUBE screenshots of the fully coupled TerrSysMP after 6 h simulation time. Each component model is naively distributed to one-third of the resources (processor distribution: 192 COSMO, 160 ParFlow, 160 CLM).

3.3 Advanced coupling interface

Nowadays, parallel scientific software applications are targeted mostly at architectures such as commodity Linux clusters with fast interconnects, which are used regularly without major problems. However, utilizing massively parallel supercomputers requires different approaches, not only because of the architecture, but also because of complicated communication patterns, data structures and distinct optimization that may be possible or necessary. The individual component models, which are used in TerrSysMP, are well tested at many different supercomputing sites, but coupling them

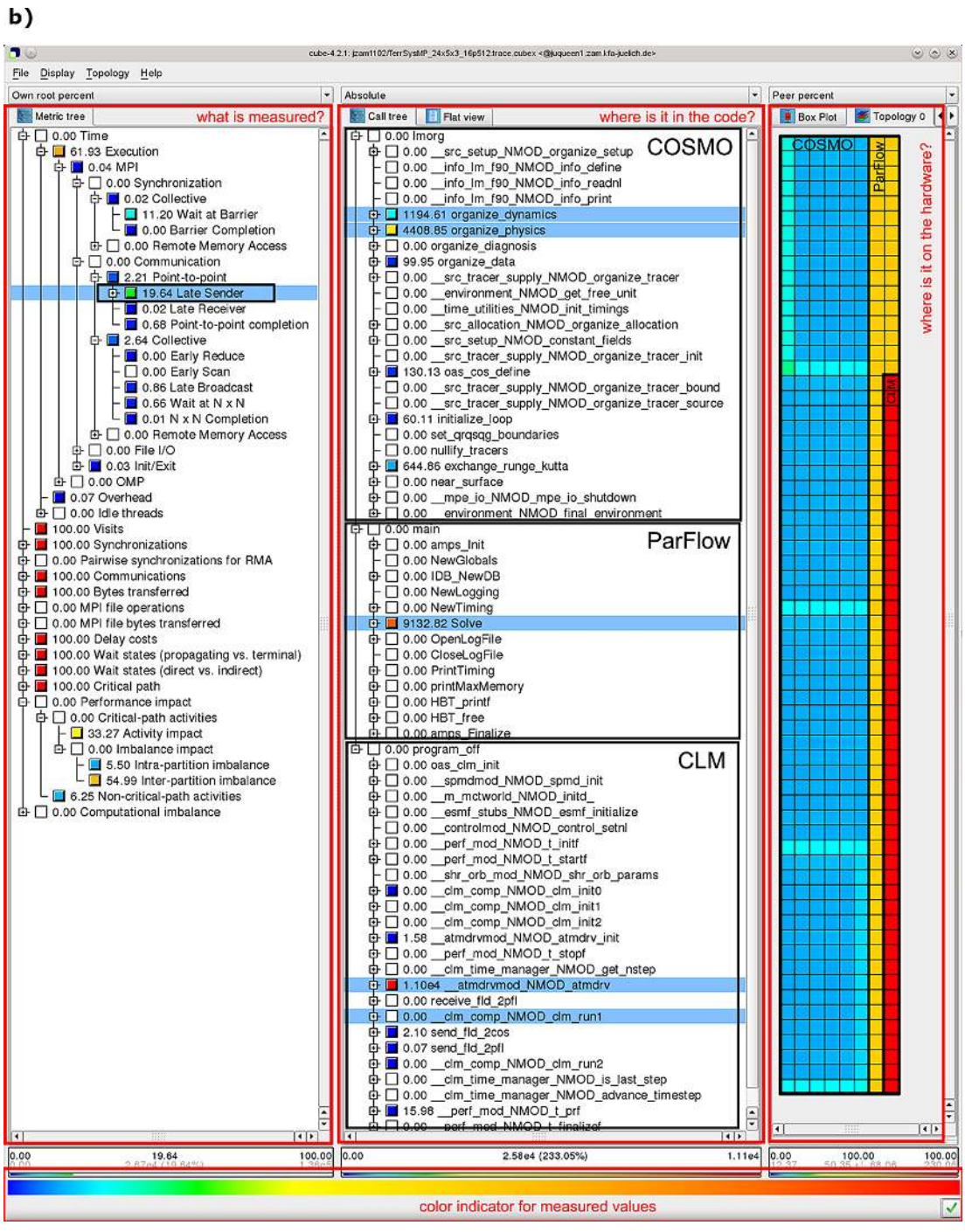

Figure 3b. The resources are distributed according to load; thus, the LateSender wait state is significantly reduced (processor distribution: 384 COSMO, 80 ParFlow, 48 CLM). The topology view shows fewer cores with LateSender wait states where receivers are waiting for senders in the relevant functions. The unit of the middle view is LateSender waiting time (accumulated over all CPUs). The units in the left and right view are percent.

especially with a highly resolved hydrologic model based on an external coupler adds an additional level of complexity.

TerrSysMP was first developed for a standard Linux cluster and then ported to JSC's IBM Blue Gene/Q supercomputer JUQUEEN. A comparably small reference test case scaled reasonably well. However, in order to use TerrSysMP as a model for large-scale, hyper-resolution simulations, the applicability for much bigger

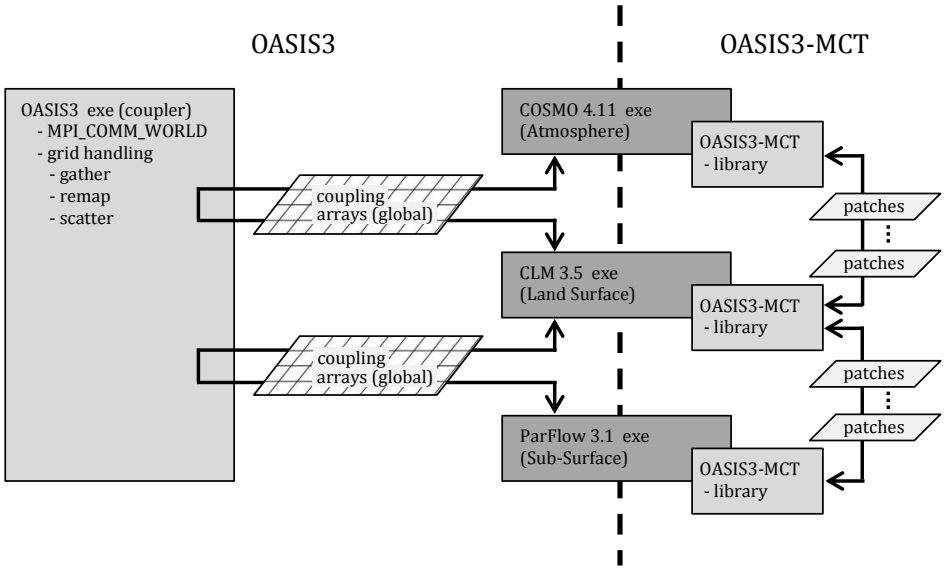

Figure 4. Schematic of the coupling in TerrSysMP with OASIS3 (left) and OASIS3-MCT (right). OASIS3 is a separate executable, and coupling arrays are repartitioned to the full domain by OASIS. OASIS3-MCT is part of each component model, and coupling arrays only consist of the local fraction of the full domain and are routed by OASIS to the destination processor.

domain sizes had to be explored. Scaling studies as described in Sect. 2.4 with resolutions from $nx = ny = 288$ (CLM, ParFlow)/$nx = ny = 144$ (COSMO) ideally up to $nx = ny = 9216$ (CLM, ParFlow)/$nx = ny = 4608$ (COSMO) were planned, while $nx = ny = 2304$ (CLM, ParFlow)/$nx = ny = 1152$ (COSMO) were actually reached.

During initial scaling tests, an increase in problem size by a factor of 4 in the second scaling step led to stalled simulations due to insufficient main memory. In contrast to most standard Linux clusters, the IBM Blue Gene/Q uses a static memory map, which means that the nodes' memory is equally distributed across the processes running on that node in MPI parallel setup (see also Sect. 2.2).This configuration is fixed and cannot change during a simulation. Since the stand-alone external coupler OASIS3 is only running with a single process, it can only use 1/16th of the RAM of an individual node if all 16 CPU cores per node are to be used, which results in 1 GB using OASIS3 as the coupler, although the rest of the node is unused (only one and the same executable may run on an individual node). A workaround for enabling more memory to one CPU is to reduce the number of processes per node (nppn), with the side effect that this configuration obviously decreases the parallel efficiency of the modeling system, especially because this process count also applies to all CPUs and, thus, also to all other component models. For OASIS3, reducing nppn to 4 and using only one-fourth of the nodes CPUs results in 4 GB of RAM which are available per process. Thus, for applications with large memory requirements, such as TerrSysMP, the resource usage when coupling with OASIS3 may be inefficient in non-standard supercomputer environments.

Investigating the memory problems further with JUQUEEN's memory-tracking interface, which provides information on the actually allocated amount of memory, showed that, in each coupling time step, OASIS3 receives several arrays from each sending process of a certain component model. It then repartitions all these local parts from the domain decomposition of each individual component model into the full domain. In subsequent steps, re-gridding and also weighting algorithms are performed. Then, the global domain is partitioned again into local parts and sent forward to the receiving component model processes. The aforementioned memory transgression occurred due to the use of arrays with the size of the complete model domain. This usually does not pose problems for smaller domain sizes, especially on general-purpose Linux clusters, which usually provide more than 2 GB RAM per core including dynamic memory allocations. However, on JUQUEEN the allocation of global domain sizes prohibits an extensive weak scaling. For example, if one needed to use just one of JUQUEEN's racks, each process would be allowed to store only 8192 double values as a local partition in order to enable one node to gather a global domain. This limitation of the single-threaded concept of OASIS3 indicates that it is (at least with regard to massively parallel supercomputers) only applicable to medium grid sizes and processor counts.

In September 2012 CNRS/CERFACS released a new version of OASIS, namely OASIS3-MCT (since May 2013 OASIS3-MCT_2.0), which now relies on the Model Coupling Toolkit, MCT (Larson et al., 2005). In the new version, OASIS is not a stand-alone coupler, but a library that is included in the different component models. The actual

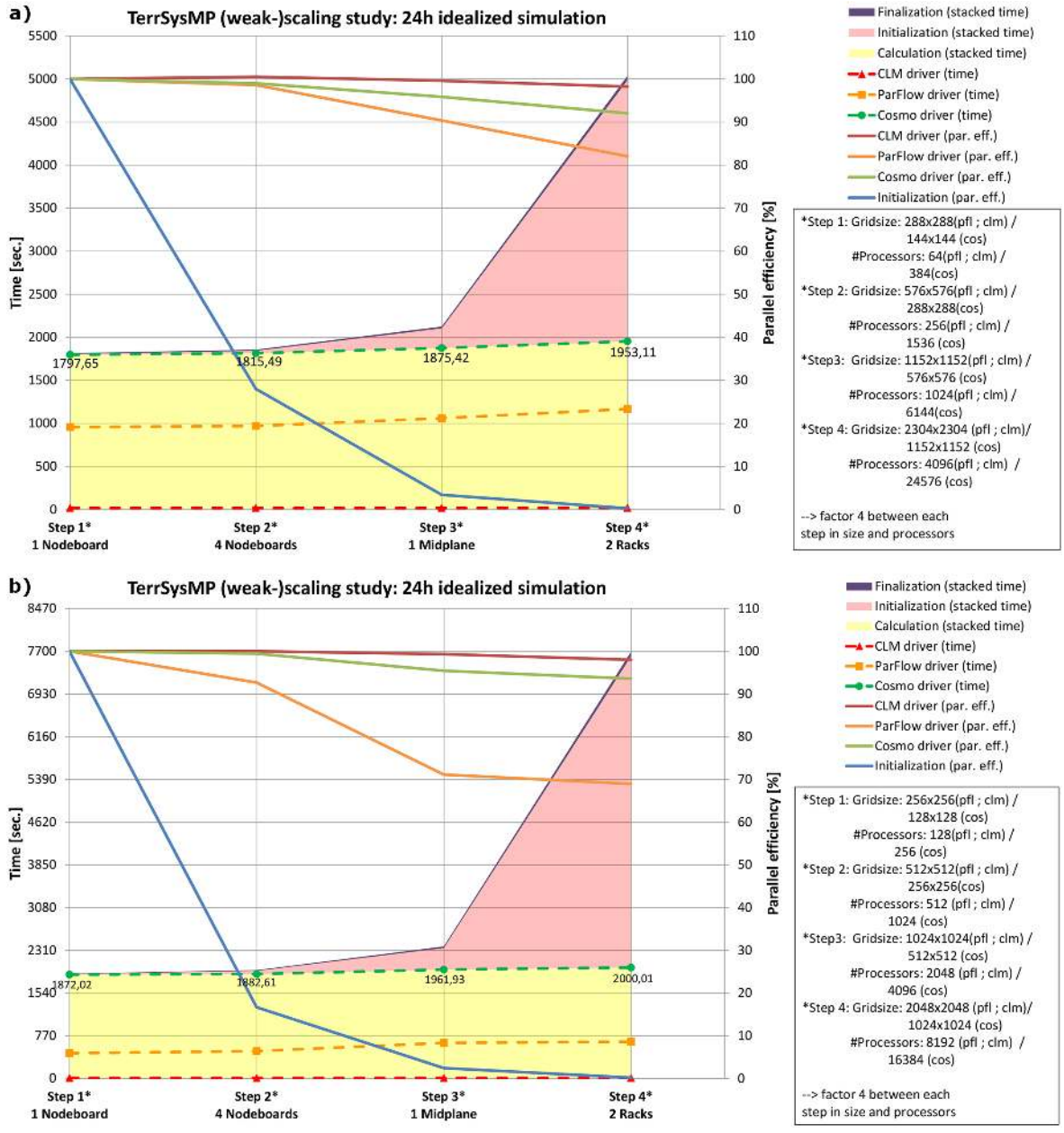

Figure 5. Idealized TerrSysMP weak-scaling study results with **(a)** setup design 1 ($nx = ny = 288$, 288, and 144 for ParFlow, CLM and COSMO, respectively) and **(b)** setup design 2 ($nx = ny = 256$, 256, and 128 for ParFlow, CLM and COSMO, respectively). The dotted lines show the absolute timings of the individual component models (green/COSMO is bounding the calculation time). The colored areas show the stacked absolute timings of the calculation, initialization and finalization time. The solid lines show the parallel efficiency of the relevant components on the secondary axis. The computational problem size, n, as well as the assigned CPU cores, np, is increasing by a factor of 4 between each step.

interface basically remains the same, which makes porting to this new version straightforward. Implementing the coupling within a library leads to a parallel OASIS, since the library is part of each process, which overcomes computational as well as bandwidth bottlenecks. But most importantly, each process can send its data to the targeted processes without the need for repartitioning a global array. This renders the coupling thinner and consumes only few extra resources. Figure 4 shows an illustration of the coupling with (a) OASIS3 and (b) OASIS3-MCT. With this newly designed coupling interface, scaling to very large model domains is possible.

3.4 Weak scaling study

By using OASIS3-MCT, the model system allows for domain sizes up to a resolution of $nx = ny = 2304$ (CLM, ParFlow) and $nx = ny = 1152$ (COSMO) grid points, which constitutes an increase in the problem size by a factor of 64 as compared to the unit reference test cases applying the original OASIS3 coupling. A further scaling was not possible at this point because, also in the component model CLM3.5, arrays with global domain size are used. It appears that in newer CLM versions this bottleneck has been removed. Further scaling steps might be possible after a newer CLM version has been implemented into TerrSysMP.

The scaling plot (Fig. 5a) of setup design 1 (Table 1a) shows that the dynamic model kernels, here called driver routines, scale well, which is essential for extended hyper-resolution runs in the context of large-scale integrated terrestrial simulations. CLM has a parallel efficiency of almost 100 % (98 % in the largest run) due to its 1-D isolated column physics with no communication overhead. The driver takes only a couple of seconds even in the larger runs. The COSMO driver has a parallel efficiency of slightly above 92 % (largest run; see dotted lines in Fig. 5a for driver efficiencies), but is the component with the heaviest computational load, therefore dictating the total calculation time. The ParFlow driver scales less well with about 82 % parallel efficiency (largest run).

Figure 5 shows which bottlenecks eventually arise in the larger scaling steps preventing the coupled system from efficient scaling. The initialization time of CLM increases drastically with each step. An analysis of the code revealed that, during initialization, the load-balancing algorithm is redundantly done by every rank and dependent on the global grid size n and the number of processors np. Since both grow by a factor of 4 between each scaling step, the initialization time in theory increases by a factor of 16. The actual increase of the initialization time is a factor of 14.41 between the last two steps. The scaling plot (Fig. 5b) of setup design 2 (Table 1b) shows a similar behavior. Only ParFlow shows a decrease in parallel efficiency (68 % in the largest run), which indicates a higher sensitivity to communication with a larger number of MPI ranks (Kollet et al., 2010). Additionally, the initialization time determined by CLM is higher because of the larger number of CLM ranks. The overall calculation time is slightly higher than in setup approach 1, since the patch size of the limiting component model COSMO is larger.

4 Summary and conclusions

TerrSysMP was successfully ported to the massive parallel IBM BG/Q system JUQUEEN of the Jülich Supercomputing Centre. In comparison to the domain sizes that could be run using the initial coupling with OASIS3, the problem size could be increased by a factor of 64 while still maintaining very good scaling factors and hence a high parallel efficiency using OASIS3-MCT. The study demonstrated that an in-depth consideration of the hardware features and software environment is necessary to efficiently operate fully coupled model systems based on the MPMD paradigm on massively parallel architectures such as JUQUEEN. This is irrespective of the individual component model's performance, as the coupling process adds significant additional complexity. Applying OASIS3 in standard Linux cluster environments for external coupling is appropriate for medium domain sizes on the order of 256 MPI ranks. Beyond medium domain sizes, OASIS3-MCT enables efficient coupling in standard and massively parallel computer environments by overcoming mainly RAM-dependent limitations. MPMD load balancing can be performed efficiently with profiling tools, such as Scalasca, to optimize MPMD resource allocation and solve configuration restrictions, such as static resource mapping. However, despite TerrSysMP's encouraging weak-scaling performance of the dynamic kernels of the different components models, initialization and I/O need to be reconciled for processor counts beyond one BG/Q midplane (8192 cores), which are required for large-scale hyper-resolution simulations. Currently, the applicability of TerrSysMP is explored for fully coupled terrestrial simulations over the pan-European continent and simulations of a regional-scale *virtual reality*.

Code availability

TerrSysMP is freely available for academic, non-profit research and application after registration at https://git.meteo.uni-bonn.de.

Acknowledgements. We would like to thank the John von Neumann Institute for Computing and the Forschungszentrum Jülich for providing the required compute time for the projects JICG43 ("Fractal Scaling of Hydrodynamics at the Catchment Scale") and HBN24 ("Development, testing, and application of an integrated soil-vegetation-atmosphere model") that were utilized in this study. The financial support by the SFB/TR 32 "Pattern in Soil-Vegetation-Atmosphere Systems: Monitoring, Modeling, and Data Assimilation" funded by the Deutsche Forschungsgemeinschaft (DFG) and the Geoverbund ABC/J (http://www.geoverbund-abcj.de) is gratefully acknowledged.

The service charges for this open access publication have been covered by a Research Centre of the Helmholtz Association.

Edited by: S. Unterstrasser

References

Anyah, R. O., Weaver, C. P., Miguez-Macho, G., Fan, Y., and Robock, A.: Incorporating water table dynamics in climate modeling: 3. Simulated groundwater influence on coupled land-atmosphere variability, J. Geophys. Res.-Atmos., 113, D07103, doi:10.1029/2007jd009087, 2008.

Baldauf, M., Seifert, A., Forstner, J., Majewski, D., Raschendorfer, M., and Reinhardt, T.: Operational Convective-Scale Numerical Weather Prediction with the COSMO Model: Description and Sensitivities, Mon. Weather Rev., 139, 3887–3905, doi:10.1175/Mwr-D-10-05013.1, 2011.

Böhme, D., Geimer, M., Wolf, F., and Arnold, L.: Identifying the Root Causes of Wait States in Large-Scale Parallel Applications, 39th International Conference on Parallel Processing (ICPP) 2010, 90–100, 2010.

Böhme, D., Wolf, F., De Supinski, B. R., Schulz, M., and Geimer, M.: Scalable Critical-Path Based Performance Analysis, IEEE 26th International Parallel & Distributed Processing Symposium (IPDPS) 2012, 1330–1340, 2012.

Chang, C.-C., Czajkowski, G., Eicken, T. V., and Kesselman, C.: Evaluating the performance limitations of MPMD communication, Proceedings of the 1997 ACM/IEEE conference on Supercomputing, San Jose, CA, 1997.

Dennis, J. M., Jacob, R., Vertenstein, M., Craig, T., and Loy, R.: Toward an ultra-high resolution community climate system model for the BlueGene platform, SciDac 2007: Scientific Discovery Through Advanced Computing, 78, U247–U251, doi:10.1088/1742-6596/78/1/012030, 2007.

Fersch, B., Wagner, S., Rummler, T., Gochis, D., and Kunstmann, H.: Impact of groundwater dynamics and soil-type on modelling coupled water exchange processes between land and atmosphere, Clim. Land Surf. Chang Hydrol., 359, 140–145, 2013.

Geimer, M., Wolf, F., Wylie, B. J. N., Abraham, E., Becker, D., and Mohr, B.: The Scalasca performance toolset architecture, Concurr. Comput.-Pract. E., 22, 702–719, doi:10.1002/Cpe.1556, 2010.

Geimer, M., Saviankou, P., Strube, A., Szebenyi, Z., Wolf, F., and Wylie, B. J. N.: Further Improving the Scalability of the Scalasca Toolset, Applied Parallel and Scientific Computing, Part II, 7134, 463–473, 2012.

Gent, P. R.: Preface to special issue on Community Climate System Model (CCSM), J. Climate, 19, 2121–2121, doi:10.1175/Jcli9020.1, 2006.

Hammond, G. E., Lichtner, P. C., and Mills, R. T.: Evaluating the performance of parallel subsurface simulators: An illustrative example with PFLOTRAN, Water Resour. Res., 50, 208–228, doi:10.1002/2012WR013483, 2014.

Hill, C., DeLuca, C., Balaji, V., Suarez, M., da Silva, A., Sawyer, W., Cruz, C., Trayanov, A., Zaslavsky, L., Hallberg, R., Boville, B., Craig, A., Collins, N., Kluzek, E., Michalakes, J., Neckels, D., Schwab, E., Smithline, S., Wolfe, J., Iredell, M., Yang, W. Y., Jacob, R., and Larson, J.: Implementing applications with the Earth System Modeling Framework, Lect. Notes Comput. Sc., 3732, 563–572, 2006.

Jones, J. E. and Woodward, C. S.: Newton-Krylov-multigrid solvers for large-scale, highly heterogeneous, variably saturated flow problems, Adv. Water Resour., 24, 763–774, doi:10.1016/S0309-1708(00)00075-0, 2001.

Keyes, D. E., McInnes, L. C., Woodward, C., Gropp, W., Myra, E., Pernice, M., Bell, J., Brown, J., Clo, A., Connors, J., Constantinescu, E., Estep, D., Evans, K., Farhat, C., Hakim, A., Hammond, G., Hansen, G., Hill, J., Isaac, T., Jiao, X. M., Jordan, K., Kaushik, D., Kaxiras, E., Koniges, A., Lee, K., Lott, A., Lu, Q. M., Magerlein, J., Maxwell, R., McCourt, M., Mehl, M., Pawlowski, R., Randles, A. P., Reynolds, D., Riviere, B., Rude, U., Scheibe, T., Shadid, J., Sheehan, B., Shephard, M., Siegel, A., Smith, B., Tang, X. Z., Wilson, C., and Wohlmuth, B.: Multiphysics simulations: Challenges and opportunities, Int. J. High Perform. C., 27, 4–83, doi:10.1177/1094342012468181, 2013.

Kirchner, J. W., Feng, X. H., and Neal, C.: Fractal stream chemistry and its implications for contaminant transport in catchments, Nature, 403, 524–527, doi:10.1038/35000537, 2000.

Kollet, S. J. and Maxwell, R. M.: Integrated surface-groundwater flow modeling: A free-surface overland flow boundary condition in a parallel groundwater flow model, Adv. Water Resour., 29, 945–958, doi:10.1016/J.Advwatres.2005.08.006, 2006.

Kollet, S. J., Maxwell, R. M., Woodward, C. S., Smith, S., Vanderborght, J., Vereecken, H., and Simmer, C.: Proof of concept of regional scale hydrologic simulations at hydrologic resolution utilizing massively parallel computer resources, Water Resour. Res., 46, W04201, doi:10.1029/2009wr008730, 2010.

Larson, J., Jacob, R., and Ong, E.: The Model Coupling Toolkit: A new fortran90 toolkit for building multiphysics parallel coupled models, Int. J. High Perform. C., 19, 277–292, doi:10.1177/1094342005056115, 2005.

Maxwell, R. M., Chow, F. K., and Kollet, S. J.: The groundwater-land-surface-atmosphere connection: Soil moisture effects on the atmospheric boundary layer in fully-coupled simulations, Adv. Water Resour., 30, 2447–2466, doi:10.1016/J.Advwatres.2007.05.018, 2007.

Maxwell, R. M., Lundquist, J. K., Mirocha, J. D., Smith, S. G., Woodward, C. S., and Tompson, A. F. B.: Development of a Coupled Groundwater-Atmosphere Model, Mon. Weather Rev., 139, 96–116, doi:10.1175/2010mwr3392.1, 2011.

Mills, R. T., Lu, C., Lichtner, P. C., and Hammond, G. E.: Simulating subsurface flow and transport on ultrascale computers using PFLOTRAN, SciDac 2007: Scientific Discovery Through Advanced Computing, 78, U387–U393, doi:10.1088/1742-6596/78/1/012051, 2007.

Oleson, K. W., Niu, G. Y., Yang, Z. L., Lawrence, D. M., Thornton, P. E., Lawrence, P. J., Stockli, R., Dickinson, R. E., Bonan, G. B., Levis, S., Dai, A., and Qian, T.: Improvements to the Community Land Model and their impact on the hydrological cycle, J. Geophys. Res.-Biogeosci., 113, G01021, doi:10.1029/2007jg000563, 2008.

Shao, Y. P., Liu, S. F., Schween, J. H., and Crewell, S.: Large-Eddy Atmosphere-Land-Surface Modelling over Heterogeneous Surfaces: Model Development and Comparison with Measurements, Bound.-Lay. Meteorol., 148, 333–356, doi:10.1007/S10546-013-9823-0, 2013.

Shrestha, P., Sulis, M., Masbou, M., Kollet, S., and Simmer, C.: A scale-consistent Terrestrial Systems Modeling Platform based on COSMO, CLM and ParFlow, Mon. Weather Rev., 142, 3466–3483, doi:10.1175/MWR-D-14-00029.1, 2014.

Song, F. and Wolf, F.: CUBE User Manual, University of Tennessee, Innovative Computing Laboratory, 2004.

Top500.org: Top 500 Supercomputer Sites, available at: http://www.top500.org/ (last access: November 2013), 2013.

Valcke, S., Balaji, V., Craig, A., DeLuca, C., Dunlap, R., Ford, R. W., Jacob, R., Larson, J., O'Kuinghttons, R., Riley, G. D., and Vertenstein, M.: Coupling technologies for Earth System Modelling, Geosci. Model Dev., 5, 1589–1596, doi:10.5194/gmd-5-1589-2012, 2012.

Valcke, S.: The OASIS3 coupler: a European climate modelling community software, Geosci. Model Dev., 6, 373–388, doi:10.5194/gmd-6-373-2013, 2013.

Wood, E. F., Roundy, J. K., Troy, T. J., van Beek, L. P. H., Bierkens, M. F. P., Blyth, E., de Roo, A., Doll, P., Ek, M., Famiglietti, J., Gochis, D., van de Giesen, N., Houser, P., Jaffe, P. R., Kollet, S., Lehner, B., Lettenmaier, D. P., Peters-Lidard, C., Sivapalan, M., Sheffield, J., Wade, A., and Whitehead, P.: Hyperresolution global land surface modeling: Meeting a grand challenge for monitoring Earth's terrestrial water, Water Resour. Res., 47, W05301, doi:10.1029/2010wr010090, 2011.

Yang, X. F., Scheibe, T. D., Richmond, M. C., Perkins, W. A., Vogt, S. J., Codd, S. L., Seymour, J. D., and McKinley, M. I.: Direct numerical simulation of pore-scale flow in a bead pack: Comparison with magnetic resonance imaging observations, Adv. Water Resour., 54, 228–241, doi:10.1016/J.Advwatres.2013.01.009, 2013.