FD4: A Framework for Highly Scalable Load Balancing and Coupling of Multiphase Models

Matthias Lieber*, Verena Grützun†, Ralf Wolke**, Matthias S. Müller* and Wolfgang E. Nagel*

*Center for Information Services and High Performance Computing, TU Dresden, Germany
†Max Planck Institute for Meteorology, Hamburg, Germany
**Leibniz Institute for Tropospheric Research, Leipzig, Germany

Abstract. More and more detailed simulation codes are developed promoted by the growing capability of high performance computers and the increasing knowledge about the underlying processes. This includes multiphase and multiphysics simulations as well as the coupling of multidisciplinary models. This paper introduces the framework FD4 (Four-Dimensional Distributed Dynamic Data structures), which enables highly scalable implementations of multiphase models. The separation of the data structures for the single phases allows the use of different partitionings and also dynamic load balancing, which is essential to efficiently utilize high performance computers. The application of FD4 for an atmospheric modeling system with a detailed description of cloud processes successfully eliminates the load imbalance with a moderate overhead of 5% at 32k processors.

Keywords: High performance computing, Dynamic load balancing, Model coupling, Atmospheric modeling, Multiphase modeling
PACS: 07.05.Tp, 07.05.Bx, 92.60.Nv

INTRODUCTION AND RELATED WORK

Today, complex numerical simulations are a fundamental research tool in nearly all scientific areas. This includes multiphase/multiphysics simulations like volcanic eruptions [1] and reactive geochemical transport [2] as well as the coupling of multidisciplinary models like earth system models [3] and air-quality models [4]. The strongly increasing parallelism in current supercomputer hardware makes the efficient use of such complex model systems a complicated task. Various frameworks and tools have been developed to assist with distributed data management and dynamic load balancing [5, 6], adaptive mesh refinement [7, 8], and model coupling [9, 10].

In this paper, we present the framework FD4 (Four-Dimensional Distributed Dynamic Data structures [11]). FD4 has been originally developed for the parallelization of spectral bin cloud models and their coupling to atmospheric models. However, FD4 is not restricted to this kind of application and can be used in other areas which benefit from the key features of the framework:

- *Dynamic load balancing*: FD4 provides dynamic load balancing of a three-dimensional block-decomposed regular grid using either the Hilbert space-filling curve or graph partitioning based on ParMETIS.
- *Model coupling*: The simulation model based on FD4 can be coupled to external models. An external model could be, for example, any CFD model or weather model.
- *Adaptive block mode*: If data and computations are only required for a specific spatial subset of the domain, FD4 saves memory by allocating only required blocks. Combined with model coupling, this feature is particularly suitable for many multiphase problems like drops, clouds, combustion processes, and flame fronts.
- *4th dimension*: In addition to the three spatial dimensions, a variable may have an extra dimension which is not decomposed. It may act, for example, as array of gas phase tracers or as a size spectrum for size-resolving particle models. In particular, FD4 is optimized to manage large numbers of values per grid cell.

FD4 integrates dynamic data management, load balancing, and coupling into a single framework to operate on the same data structures, which allows more performance optimizations compared to the utilization of separate software for these tasks. Though, frameworks specialized for, e.g., coupling offer more functionality than FD4, like grid interpolation and *concurrent* coupling. In contrast to FD4, many applications handle space-time heterogeneity of the simulation problem with adaptive mesh refinement methods. The adaptive block mode of FD4 offers a different kind of adaptivity which is suited for certain multiphase problems only. A special feature of FD4 is the optimization to large number of values per grid cell.

The rest of the paper is organized as follows: In the next section we give a description of the framework FD4 and show some details about its functionality. Then we show that the usage of FD4 for the atmospheric modeling system COSMO-SPECS reduces the run time of the simulation about 52% at 32k processor cores of a BlueGene/P system with an overhead of less then 5% for dynamic load balancing and coupling. Finally, we conclude the paper and give an outlook to future developments.

THE FRAMEWORK FD4

FD4 manages a three-dimensional regular Cartesian grid without local refinement on distributed memory systems. It consists of several Fortran 95 modules each providing different data structures and services. The parallelization is based on version 2.0 of the Message Passing Interface (MPI-2).

Basic Data Structure. The basic data structure of FD4 consists of the *variable table* and the *blocks*. The variable table is a user-provided table of all variables which should be managed in the numerical grid. This table contains information about the location in the grid cells (cell-centered or face-centered), the number of time steps to allocate, the size of a 4th (non-spatial) dimension, and the default variable value. FD4 allocates the data fields in the blocks based on the variable table. The blocks are rectangular subsections of the numerical grid and provide a decomposition which is used for parallelization and dynamic load balancing. Each process can own an arbitrary number of blocks. To access the blocks, FD4 offers an *iterator* to traverse through the list of blocks.

Ghost Blocks. The blocks in FD4 do not contain a ghost area to store values from neighbor blocks. Instead, additional *ghost blocks* are allocated only at the processor boundaries of the distributed block structure. This reduces the memory usage when using blocks containing a small number of grid cells. For our target applications, using small blocks is crucial for two reasons: Firstly, small blocks enable a fine-grained load balancing and increase the efficiency of the adaptive block mode. Secondly, FD4 is focused on models which require a high number of values per grid cell (4th dimension in addition to the three spatial dimensions). With respect to cache efficiency, this means that only blocks with a small number of cells will not exceed the processor cache. To access the neighbor cells in a uniform way, FD4 copies the values from the current block and the boundaries of the six neighbor blocks (or ghost blocks) to a work array. The ghost blocks are updated via MPI communication using FD4's *ghost communicator*.

Adaptive Block Mode. FD4 allows the dynamic adaption of the block allocation to spatial structures. It is useful for multiphase applications when neither computations nor data are required for certain regions of the numerical grid. In this case, memory can be saved by not allocating the unused (*empty*) blocks. A block is considered empty if the value of selected variables is less or equal than a given threshold, i.e. it does not contain any quantities of a certain phase. Based on this definition, FD4 ensures that all blocks are allocated which are (possibly) modified by stencil computations near non-empty cells. That means that an empty block will not be removed if the phase front is located in a neighbor block near the boundary and new blocks will be allocated as the phase front propagates towards a block boundary, see Fig. 1a.

Dynamic Load Balancing. Dynamic load balancing in FD4 is carried out when the load balance decreases below a certain limit or when blocks are added or removed from the block structure (adaptive block mode). A weight can be assigned for each block, e.g. by measuring the computation time. Two partitioning algorithms are available in FD4: Graph partitioning using ParMETIS [12] and Hilbert space-filling curve (SFC [13]) partitioning. However, our measurements have shown that the graph partitioning calculation has a higher overhead compared to the SFC [11]. The Hilbert curve traverses each block with high locality and provides a linearization of the block structure. This reduces the three-dimensional partitioning problem to the partitioning of a one-dimensional array of block weights with the goal to minimize the maximum load (*bottleneck value*) among all partitions. Several heuristics and exact algorithms exist for this problem [14]. FD4 uses the following parallel heuristic: After determining the lower and upper bound of the bottleneck value, each process checks for a different bottleneck value within this interval whether a partitioning exists for it. Then, the minimum of the valid bottleneck values is identified and each process determines its own partitioning based on this value. As shown in Fig. 1b, the SFC load balancing method also works in adaptive block mode by skipping the not allocated blocks.

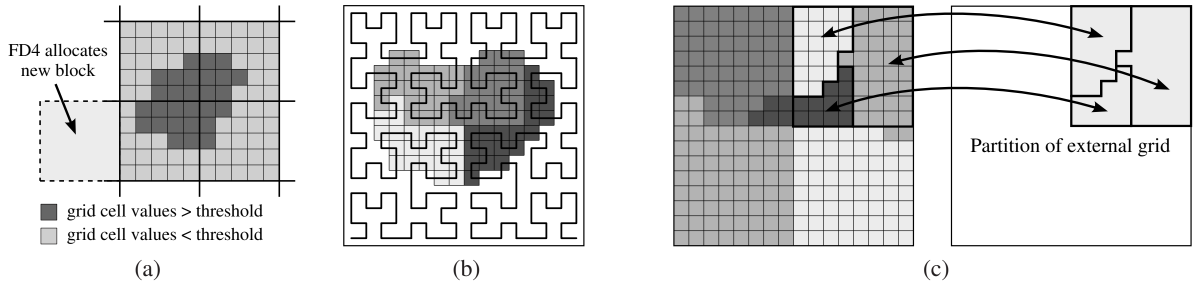
FD4 allocates new block
grid cell values > threshold
grid cell values < threshold
(a)
(b)
(c)
Partition of external grid

FIGURE 1. Illustration of FD4 concepts for a two-dimensional grid: (a) adaptive block mode: a new block is allocated to guarantee appropriate data for stencil computations around non-empty grid cells, (b) Hilbert space-filling curve partitioning in adaptive block mode, (c) model coupling between FD4 and an external partition.

Performing load balancing at every time step of the simulation generates noticeable overhead and limits scalability. To reduce the number of load balancing invocations, FD4 decides automatically if load balancing is beneficial. Therefore, the time lost due to imbalance is weighed against the time required for load balancing.

Model Coupling. FD4 allows coupling models based on FD4 to external models which have a different grid partitioning. The coupling interface assumes that the models are coupled *sequentially in the time domain*, i.e. both models work on the same set of parallel processes and all processes perform computations for these models alternately. Additionally, the models should use the same numerical grid, or at least the external model provides its coupling data matching the grid of FD4. Based on a description of the external model's partitions, FD4 computes the overlaps with the FD4 block structure and transmits the variables directly between the processes, see Fig. 1c. Coupling data can be exchanged in both directions between FD4 and the external models.

FD4 APPLICATION STUDY: DETAILED CLOUD MODELING

FD4 is applied to improve the performance of the atmospheric modeling system COSMO-SPECS [15], which has been developed to study the interaction between aerosols, clouds, and precipitation in a high level of detail. It consists of the COSMO model, a non-hydrostatic limited-area atmospheric model, and SPECS, the SPECtral bin cloud microphysicS model [16]. SPECS is integrated into the COSMO model and describes cloud processes in a very high level of detail, which requires large memory and computing time resources. Since the workload varies strongly depending on the presence and type of clouds, load imbalances occur in the static partitioning of the COSMO model. To enable dynamic load balancing for SPECS, the domain decompositions of COSMO and SPECS are separated by using FD4 for the data management of SPECS. FD4's coupling interface is used to exchange data between the different partitionings, resulting in the model system COSMO-SPECS+FD4 [17].

We executed a weak scaling benchmark to compare the performance of the original model system with the optimized version and to measure the additional overhead of FD4 for dynamic load balancing and coupling. The basic test case consists of a spherical heat perturbation which is placed in the center of the horizontal 32×32 grid and results in the growth of a precipitating mixed-phase cumulus cloud during a period of 30 min. The horizontal grid has a resolution of 1 km and periodic boundary conditions. The domain height of 18 km is discretized with 48 nonuniform height levels. This test is executed with 256 processes. To scale up the problem size exactly in the same proportion as the number of processes, we are using the *replication scaling* approach. When doubling the number of processes, the horizontal grid size is also doubled and virtually subdivided into tiles of 32×32 cells, each initialized with exactly the same conditions for the growth of the cloud. This way, the grid is scaled up to 512×256 cells containing 128 clouds. FD4 has to balance 393 216 blocks dynamically over 32k processes in this configuration. Note, that neither the model nor FD4 take advantage about the replication in the simulation. Figure 2a compares the measured run times of both model versions (without initialization and model output time). Both scale almost perfectly, but the FD4 version executes more than twice as fast due to the dynamic load balancing which minimizes waiting times. The slight increase of run time is due to the overhead of FD4 for load balancing and coupling. Figure 2b shows the run time percentage of this overhead. It is growing from 1% at 256 processes to nearly 5% at 32k processes, which is a good result considering the 128-fold increase of the number of blocks and processes that have to be managed.

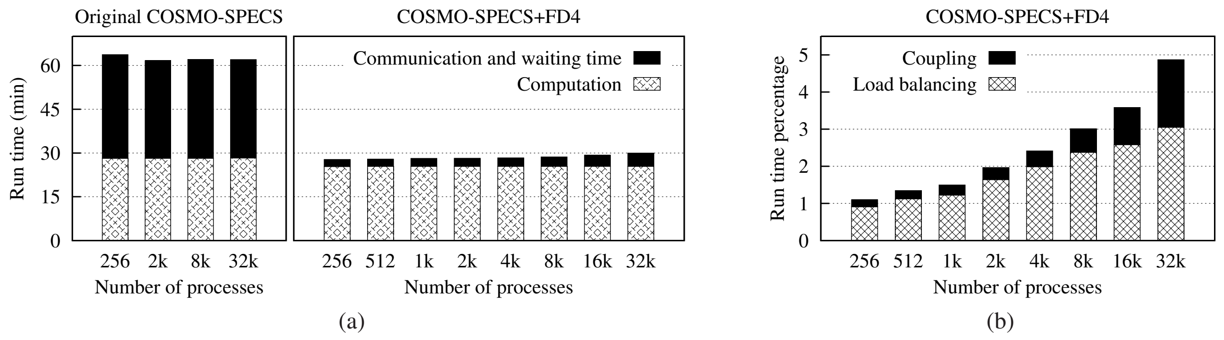

FIGURE 2. Weak scaling benchmark results on a BlueGene/P system: (a) Run time comparison between the original COSMO-SPECS and the load balanced version COSMO-SPECS+FD4, (b) Run time percentage of FD4 overhead.

CONCLUSION AND OUTLOOK

This paper presents the framework FD4 for dynamic load balancing and coupling of multiphase models. Benchmark measurements with the atmospheric modeling system COSMO-SPECS+FD4 show that FD4 can efficiently balance more than 10^5 blocks over more than 10^4 processes. FD4 is available as open source software at `http://www.tu-dresden.de/zih/clouds`. Future developments of FD4 include the development of fast parallel I/O based on the NetCDF, which is necessary to efficiently simulate large-scale problems. Moreover, the amount of metadata about the dynamic partitioning and coupling to external models becomes a critical issue for highly parallel systems. For COSMO-SPECS+FD4 we plan to implement multirate Runge-Kutta time integration methods to account for the heterogeneous time scales of cloud processes and enable further performance improvements.

ACKNOWLEDGMENTS

We thank the Deutscher Wetterdienst (German Weather Service) for providing the COSMO model. Furthermore, we thank Jülich Supercomputing Centre, Germany, for access to the JUGENE BlueGene/P supercomputer. This work was funded by the German Research Foundation (DFG), grant No. NA 711/2-1.

REFERENCES

1. T. E. Ongaro, C. Cavazzoni, G. Erbacci, A. Neri, and M. Salvetti, *Parallel Comput.* **33**, 541–560 (2007).
2. G. Hammond, P. Lichtner, and C. Lu, *J. Phys. Conf. Ser.* **78**, 012025 (2007).
3. W. M. Washington, L. Buja, and A. Craig, *Phil. Trans. R. Soc. A* **367**, 833–846 (2009).
4. M. Lieber, and R. Wolke, *Environ. Modell. Softw.* **23**, 235–243 (2008).
5. K. Devine, E. Boman, R. Heaphy, B. Hendrickson, and C. Vaughan, *Comput. Sci. Eng.* **4**, 90–97 (2002).
6. J. D. Teresco, K. D. Devine, and J. E. Flaherty, "Partitioning and Dynamic Load Balancing for the Numerical Solution of Partial Differential Equations," in *Numerical Solution of Partial Differential Equations on Parallel Computers*, Springer, 2006, pp. 55–88.
7. B. Van Straalen, J. Shalf, T. Ligocki, N. Keen, and W.-S. Yang, "Scalability challenges for massively parallel AMR applications," in *IPDPS '09*, IEEE Computer Society, 2009, pp. 1–12.
8. C. Burstedde, M. Burtscher, O. Ghattas, G. Stadler, T. Tu, and L. C. Wilcox, *J. Phys. Conf. Ser.* **180**, 012009 (2009).
9. J. Larson, R. Jacob, and E. Ong, *Int. J. High Perf. Comput. Appl.* **19**, 277–292 (2005).
10. R. Redler, S. Valcke, and H. Ritzdorf, *Geosci. Model Dev.* **3**, 87–104 (2010).
11. M. Lieber, R. Wolke, V. Grützun, M. S. Müller, and W. E. Nagel, "A framework for detailed multiphase cloud modeling on HPC systems," in *Parallel Computing*, IOS Press, 2010, vol. 19, pp. 281–288.
12. K. Schloegel, G. Karypis, and V. Kumar, *Concurr. Comp. Pract. E.* **14**, 219–240 (2002).
13. H. Sagan, *Space-filling curves*, Springer, 1994.
14. A. Pinar, and C. Aykanat, *J. Parallel Distrib. Comput.* **64**, 974–996 (2004).
15. V. Grützun, O. Knoth, and M. Simmel, *Atmos. Res.* **90**, 233–242 (2008).
16. M. Simmel, and S. Wurzler, *Atmos. Res.* **80**, 218–236 (2006).
17. M. Lieber, V. Grützun, R. Wolke, M. S. Müller, and W. E. Nagel, "Highly Scalable Dynamic Load Balancing in the Atmospheric Modeling System COSMO-SPECS+FD4," 2010, accepted for Proc. of PARA 2010.