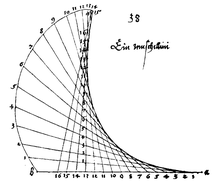
Large Scale Simulations in the Software Toolbox UG4: Preliminary Results for the Project FWUG on the Hawk supercomputer

Ruben Buijse, Arne Nägel*, Martin Parnet
Goethe-Center for Scientific Computing, Goethe Universität Frankfurt am Main, Kettenhofweg 139, D-60325 Frankfurt a.M., Germany
*naegel@gcsc.uni-frankfurt.de



Goethe-Center for Scientific Computing

Abstract

This work presents a first performance analysis of the software framework UG4 [1, 2] on the Hawk Apollo supercomputer. The software demonstrated excellent scaling properties before. It has now been demonstrated that this also holds true for HLRS's most recent machine and its architecture. Three aspects are emphasized: (i) classic weak scaling of a multi-grid solver for the Poisson equation, (ii) strong scaling for the heat equation using multi-grid-in-time, and (iii) application to a thermo-haline-flow problem in a fully-coupled fashion.

Numerical Methods

Weak Scaling of the Multilevel Solver

Classic setup, e.g. [2]:

- Poisson's equation on the unit cube with Dirichlet boundary conditions
- Finite elements on a structured hexagonal grid
- Solver: Geometric multigrid, damped Jacobi smoother, V(1,1)-cycle
- Wall clock times for a total $8, 64 \ldots, 32768$ cores have been measured.
- Roughly $250,000$ degrees of freedom per core, number of degrees of freedom and number of cores grow simultaneously ($\times 8$).

Using **default process assignment** onto n nodes, the maximum number of processes per node was limited using the switch `select=n:node_type=rome:mpiprocs=m`. Three different phases of the algorithm are considered:

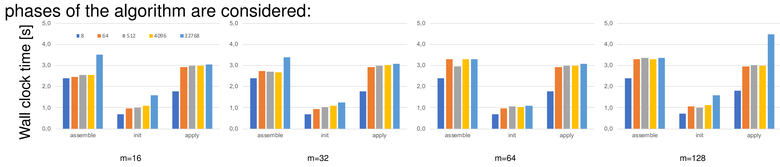


Figure 1: Wall clock times using a default process assignment with at most m processes per node.

- The `assemble`-phase computes the finite element stiffness matrix. Dominated by computation, with memory access for reading element information and writing coefficients into the sparse-matrix format. No communication via MPI.
- The `init`-phase refers to the setup of the multilevel solver. Communication interfaces are constructed. Data exchange for a proper application of matrix-vector operations.
- The `apply`-phase corresponds to timings are for a fixed number of five BiCGStab-sweeps. This phase features many memory accesses to vector and sparse-matrix data structures. Moreover, the network is used heavily for all-reduce operations (e.g., for stopping criteria) as well as nearest neighbor communication (e.g., master-slave-exchange along the process boundaries).

In order to match the node topology, the experiment has been repeated for **process assignment with stride**. Only every fourth and second core have been used respectively:

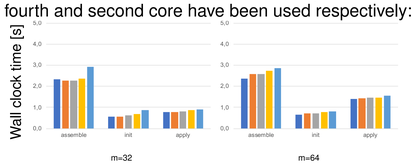


Figure 2: Wall clock times for process allocation with stride (i.e., one and two core per CCX respectively).

This yields various performance improvements:

- For the `assemble`-phase, a small acceleration of about 10% is observed. This is propably due to improved read/write memory access.
- In the `init`-phase communication becomes more important. In this case a slightly more pronounced acceleration is observed, and run times reduce by 20-30%.
- The biggest gains are achieved in the `apply`-phase. Compared to the default assignment, the acceleration approaches the optimal factors, i.e., 2 when using 2 out of 4 cores per CCX, and 4 when using only a single core per CCX.

Strong Scaling for Multigrid Reduction in Time

Classic time stepping by one-step or multi-step-methods is a serial procedure. On modern architectures, researchers may thus face a serial time-integration bottleneck, which can be mitigated suing parallel-in-time integration. One example of this class is multigrid reduction in time (MGRIT) [3]. This method has been implemented in form of the `XBraid` library, and coupled with UG4 succesfully [4].

As an example, we consider the heat equation

$$\partial_t u(\vec{x},t) - \alpha \triangle u(\vec{x},t) = f(\vec{x},t)$$

for $(\vec{x},t) \in (0,1)^3 \times (0,4\pi)$ and $\alpha = 0.1$. Initial value, Dirichlet boundary conditions and right hand side f are chosen such that the solution $u(\vec{x},t) = \sin(\pi x_1) \cdot \sin(\pi x_2) \cdot \sin(\pi x_3) \cdot \cos(t)$ is obtained. The equation is discretized using an implicit Euler method in time and Q_1 finite elements in space.

Assuming that p MPI processes are available, two setups are compared:

- For **serial time integration**, all available processes are dedicated to the spatial domain. We consider the sequence $p \in \{8, 64, 512\}$. The time domain is split into 16384 equidistant intervals that are treated sequentially.
- The strong scaling w.r.t the spatial variables is limited: From eight to 64 processes, a speedup of ≈ 5.5 is observed. For 512 processes the run time even deteriorates. In this case, the number of degrees of freedom per process is small, and communication becomes the dominant factor.

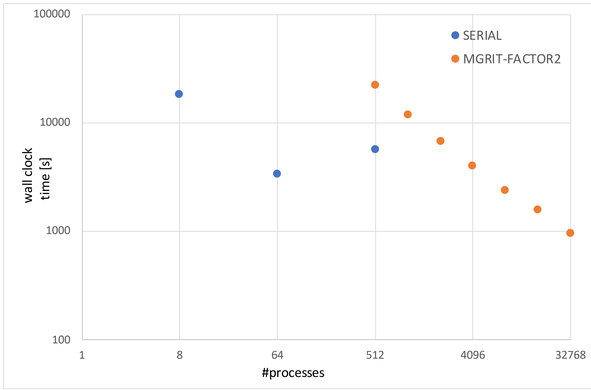


Figure 3: Strong scaling for heat equation. Comparison of a classic serial time integration (SERIAL) and multigrid in time (MGRIT-FACTOR2) with factor of 2 coarsening. On the finest level, the unit cube $\Omega = (0,1)^3$ is split into $128 \times 128 \times 128$ elements, with $2,146,689$ degrees of freedom. Splitting this onto eight processes, this means that workload is in a similar order of magnitude as in the previous example for the steady state. Note, however, that due to the fixed time step size, assembling the operator is only required once.

- The **parallel-in-time** integration using MGRIT uses $p = p_s \times p_t \in \{512, \ldots, 32768\}$ processes. A fixed number of $p_s = 64$ processes is dedicated to spatial domain. For the the time domain a variable number $p_t \in \{8, \ldots, 256\}$ is used. Each process then owns an interval with $32768/p_t$ equidistant time points.
- MGRIT initially suffers from some computational overhead, as several iterations are required to achieve the discretization error of the serial method. To that end, observe that using 512 processes for $p_t = 8 = 512/64$ equidistant time intervals roughly takes the same time as the serial computation using $p = 8$ processes. However, from that point on, adding more processes in the time dimension always lead to a speedup.
- The results are in agreement with results [3, 4], where problems of the size investigated here were not computed, but predicted.

Applications

Thermohaline flow

- Thermohaline flow, e.g. [5], is a special instance of density-driven flow. Fluid density depends on both temperature and fluid composition. Systems of this type are important, e.g., for modeling transport of CO2 or NaCl in repositories and deep geological layers.
- Hot fluids have a lower density than cold fluids (↗); high solute concentrations yields a higher density (↘). For a brine parcel in porous medium, both positive and negative buoyancy can occur.
- The following images provides preliminary results for the parcel benchmark with negative buoyancy. Figures 4 and 5 show the evolution of the parcel at an early and late stage. Simulations for different grid resolutions using 128, 1024, 8192 cores. Implementation provided by the d^3f-framework [6].

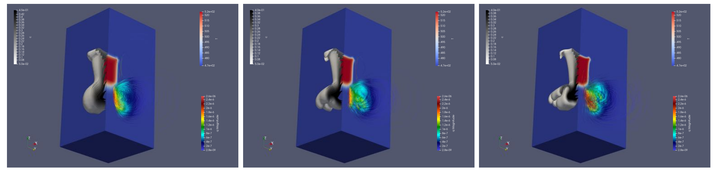


Figure 4: Early stage (three different levels of spatial refinement from top to bottom): The parcel sinks down. Depending on the spatial resolution, a branching of the central finger becomes visible.

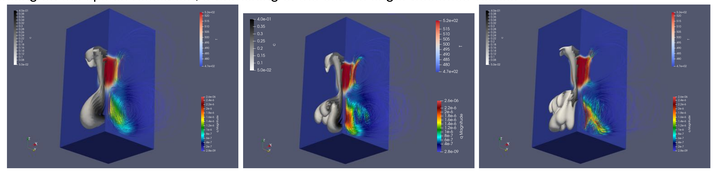


Figure 5: Late stage (three different levels of refinement): A layered fingering evolves. In the highest resolution, three layers of fingers are visible.

- All fields shown in a single plot (symmetry w.r.t the center axis): *temperature field* is shown in the background (blue to red); ten isosurfaces of the *salt mass fraction* in greyscale on the left; streamlines indicating the *velocity field* are shown on the right (rainbow colors).
- Proper control of the discretisation error is madatory an can be achieved using novel time integration schemes [7].
- Negative buoyancy obviously leads to a fingering effect. The structure of the convective cells, and/or, number of fingers is subject to further research.

References

[1] Andreas Vogel et al. UG 4: A novel flexible software system for simulating PDE based models on high performance computers. *Comp. Vis. Sci.*, 16(4):165–179, 2013.

[2] Sebastian Reiter et al. A massively parallel geometric multigrid solver on hierarchically distributed grids. *Comp. Vis. Sci.*, 16(4):151–164, 2013.

[3] Robert D. Falgout et al. Parallel time integration with multigrid. *SIAM J. Sci. Comput.*, 36(6):C635–C661, 2014. LLNL-JRNL-645325.

[4] Martin Parnet. Zeitparallele Lösungsverfahren für die Wärmeleitungsgleichung, 2020.

[5] Alfio Grillo, Michael Lampe, and Gabriel Wittum. Three-dimensional simulation of the thermohaline-driven buoyancy of a brine parcel. *Comp. Vis. Sci.*, 13(6):287–297, Aug 2010.

[6] Anke Schneider, editor. *Modeling of Data Uncertainties on Hybrid Computers.* GRS Bericht 392, 2016.

[7] Ruben Buijse. Numerische Verfahren zur Simulation thermohaliner Strömungen in der Software d3f++, 2018.

Acknowledgements

The authors gratefully acknowledge the support of Björn Dick, Bernd Krischok, and many other HLRS staff members.