Generic Programming for Flexible, High Performance Numerical Software

Bård Skaflestad*, Atgeirr F. Rasmussen†

*SINTEF ICT, Applied Mathematics
Oslo, Norway
e-mail: Bard.Skaflestad@sintef.no

†SINTEF ICT, Applied Mathematics
Oslo, Norway
e-mail: atgeirr@sintef.no

ABSTRACT

Implementing software for numerical solution of partial differential equations has traditionally been a trade-off between high computational efficiency and generality of the resulting software system. In particular, extensible and interchangeable components based on dynamic polymorphism have an inherent abstraction cost which often leads to software which is not sufficiently efficient. Recently, generic programming [1] has emerged as a promising and popular approach to reducing or eliminating this cost. Such techniques consequently enable writing separate families of, e.g., solvers and grid representations within which individual family members may be exchanged at zero additional cost.

The Dune [2] project uses generic programming to provide interfaces for grids and linear algebra routines which are independent of the actual, underlying implementations. Numerical software built on top of these interfaces in turn becomes highly adaptable whilst not incurring significant run-time performance penalties as a result of the added flexibility.

We have implemented a new grid class in the Dune framework that is capable of representing corner-point grids. Such grids are popular for modeling porous media flow in subsurface reservoirs. We have also written solvers that work with the same interface. To demonstrate the flexibility of the generic programming approach, we have solved several example cases multiple times, each time with a different combination of grid and solver. We compare the results with each other, and give some advice for choosing the optimal combination for a given problem. We also compare the results to older, less flexible code, and find that no performance is sacrificed for the gain in flexibility.

REFERENCES

[1] D. Vandevoorde and N.M. Josuttis, C++ Templates: The Complete Guide. Addison–Wesley Professional, 2002.

[2] P. Bastian, M. Blatt, C. Engwer, A. Dedner, R. Klöfkorn, S.P. Kuttanikkad, M. Ohlberger and O. Sander, The Distributed and Unified Numerics Environment (DUNE), *Proc. of the 19th Symposium on Simulation Techniques in Hannover*, September 12–14, 2006