THE `deal.II` LIBRARY, VERSION 8.0

WOLFGANG BANGERTH[*], TIMO HEISTER[†], LUCA HELTAI[‡], GUIDO KANSCHAT[§], MARTIN KRONBICHLER[¶], MATTHIAS MAIER[‖], BRUNO TURCKSIN[**], AND TOBY D. YOUNG[††]

Abstract. This paper provides an overview of the new features of the finite element library `deal.II` version 8.0.

1. Overview. `deal.II` version 8.0 was released July 24, 2013. This paper provides an overview of the new features of this release and serves as a citable web reference for the `deal.II` software library version 8.0. `deal.II` is an object-oriented finite element library used around the world in the development of finite element solvers. It is available for free under the GNU Lesser General Public License (LGPL) from the `deal.II` homepage at http://www.dealii.org/.

Version 8.0 is a major release. It has numerous significant features along with the usual set of bug fixes and documentation updates. In particular, it has the following noteworthy large changes:

- The configuration and build system has been switched to `CMake`, providing better support for a wide variety of platforms, better integration with IDEs such as Eclipse, and many other advantages. See Section 2.1.
- `deal.II` now supports 64-bit integers for degrees of freedom indices for problems with more than 2 billion unknowns and has been tested on problems of up to 27 billion unknowns. See Section 2.2.
- `deal.II` is now licensed under the GNU Lesser General Public License version 2.1 or later (LGPL-2.1+), see Section 2.3 for details.
- There is the usual set of dozens or hundreds of new small features and bugfixes, some of which are described in 2.4.

Information on how to cite `deal.II` is provided in Section 3.

2. Important Changes.

2.1. Build System.

With the release of version 8.0 and after more than 12 years with an `autoconf` and custom makefiles setup, `deal.II` switched its build system over to `CMake`. This is a major rewrite which changes over $25,000$ lines of code (10,000 insertions and 15,000 deletions) together with a major reorganization and cleanup of

[*]Department of Mathematics, Texas A&M University, College Station, TX 77843, USA, bangerth@math.tamu.edu

[†]Mathematical Sciences, O-110 Martin Hall. Clemson University. Clemson, SC 29634, USA, heister@clemson.edu

[‡]SISSA - International School for Advanced Studies, Via Bonomea 265, 34136 Trieste, Italy, luca.heltai@sissa.it

[§]Interdisciplinary Center for Scientific Computing, Heidelberg University, Im Neuenheimer Feld 368, 69120 Heidelberg, Germany, kanschat@uni-heidelberg.de

[¶]Institute for Computational Mechanics, Technische Universität München, Boltzmannstr. 15, 85748 Garching b. München, Germany, kronbichler@lnm.mw.tum.de

[‖]Institute of Applied Mathematics, Heidelberg University, Im Neuenheimer Feld 293/294, 69120 Heidelberg, Germany, matthias.maier@iwr.uni-heidelberg.de

[**]Department of Mathematics, Texas A&M University, College Station, TX 77843, USA, turcksin@math.tamu.edu

[††]Institute of Fundamental Technological Research of the Polish Academy of Sciences, ul. Pawińskiego 5b, Warsaw 02-106, Poland, tyoung@ippt.pan.pl

the code base. The rewrite addressed quite a number of shortcomings with the old build system, that made a rewrite highly desirable (if not necessary):

- Over time, with the addition of an increasing number of external libraries, `deal.II` can optionally interface with, the configuration happened to be highly heterogeneous—almost every external dependency had to be set up with different command line options with completely different internal setup logic in the `autoconf` files.
- Nowadays, `deal.II` is used in highly different environments ranging from installations on laptops to computing clusters. The old build system was not suited very well for quite a number of them, e. g. distributional needs or static linkage.
- The old build system strongly depended on the availability of a UNIX or GNU/Linux environment due to the fact that GNU Make, Bash and Perl were mandatory build time dependencies, which made support for other platforms (and especially Microsoft Windows) difficult.

Arguably, it would have been possible to address all this shortcomings in the current build system, or by changing to another alternative, but there are a number of reasons that make `CMake` a particularly good choice as build system for `deal.II`:

- `CMake` readily provides elaborate support for dependency resolution, incremental rebuilds and parallel builds, which were previously implemented mainly by hand.
- `CMake` is a meta build system. It provides support for *native toolchains* for Linux, BSD, Darwin, and Windows platforms with minimal external dependencies, which drastically increases portability.
- `CMake` supports a variety of commonly used IDEs for different platforms such as Eclipse, KDevelop, Xcode, CodeBlocks or MSVC where it can directly produce a corresponding project.

The fact that the new build system was rewritten from scratch, offered the possibility to reformulate and respect some major design criteria. Amongst them are

- Full support for all 16 external dependencies in various setups—either compiled and/or installed by the user, or provided by software distributions such as Linux distributions or Mac Ports.
- Feature auto detection to ease the setup for the average user, while at the same time providing full user override of configuration options to support almost any setup case.
- Be (almost) readily packageable for major distributions (e. g. Debian Linux, or Ubuntu).
- Provide a project configuration so that user projects based on `deal.II` only need a minimalistic CMakeLists.txt file on client side.

2.2. 64-bit Indices. Up to version 7.3 of `deal.II`, the indices of degrees of freedom (which are uniquely defined) were stored using the *unsigned int* datatype. This limited the total number of degrees of freedom in a computation to about four billion. When `deal.II` was used in conjunction with `PETSc` or `Trilinos`, this number was in fact only half as large since both of these libraries use *signed integer* indices (leaving only 31 bits for the indices). Furthermore, because an individual processor needs at least on the order of 10^5 unknowns to have a suitable amount of work, `deal.II` could not scale to more than about $\frac{2 \cdot 10^9}{10^5} = 20,000$ processors. Even though this number may seem large, an increasing number of supercomputers offer a much larger number of processors – the fastest supercomputers today are almost one hundred times larger. It is also important to note that using simulations with more than two billions of unknowns can be run using only a couple of thousand processors. This number is easily reached by clusters at typical universities in the West.

To increase the number of unknowns that can be solved using `deal.II`, the degrees of freedom can now be stored using *unsigned long long int* (a data type typically 64 bit wide) and the new 64 bit capabilities of Epetra (Trilinos 11.2 and later) are employed (PETSc could already be compiled to use 64 bit indices for a long time). The new version of `deal.II` is completely backward compatible with the previous version and can be compiled to use 32 or 64 bit indices indifferently. This was done by using a typedef *types::global_dof_index* to switch between *unsigned int* and *unsigned long long int* indices. The switch between 32 and 64 bit indices is done using the cmake option `-DDEAL_II_WITH_64BIT_INDICES=OFF/ON`. By default, this option is turned off to keep memory consumption at moderate levels.

Using 64 bit indices only makes sense for parallel computations for which `deal.II` relies on either `PETSc` or `Trilinos`. To use the 64 bit indices with the former, `PETSc` must be configured with `--with-64-bit-indices`. For the latter, `Trilinos` 11.2 or later must be used (by the default `Trilinos` is compiled with 32 and 64 bit capabilities) and uses the facilities laid out in [8].

To write a code compatible for 32 and 64 bit indices, *types::global_dof_index* needs to be used in all places where the global indices of degrees of freedom are referenced. There are of course many of these places: the library contained some 20,000 occurrences of the text `unsigned int`, all of which needed to be examined, and some 1,600 places had to be modified. However, the library now compiles and runs with 64 bit indices and the adapted version of the step-40 tutorial program was run on 3600 processors; this computation had 1.47 billions of active cells, 5.898 billions of degrees of freedom, and was solved in 771 seconds.

2.3. License Change. `deal.II` used to be distributed under the Q Public License (QPL), a license that was popular in the late 1990s mainly due to its utilization by the Qt Application Framework.

The QPL, as many other open source licenses, puts restrictions on any use; in particular, programs that are developed to use `deal.II` needed to be open source as well. In practice, this puts some restrictions on commercial use because companies are unwilling to engage in a model where they are required to deliver their source code to customers who may then distribute it further.

This license was chosen in the early days of `deal.II` out of a feeling that whoever uses the library should be required to show the same open source spirit. Abstractly, many will probably agree with this sentiment. However, in more than 10 years of reality, it has only led to companies walking away from using `deal.II` for commercial products. As a consequence, the project has seen no gain, only losses from this choice of license.

We have therefore decided to change the license under which `deal.II` is distributed to the GNU Lesser General Public License (LGPL), version 2.1 or later at the discretion of the user. Among the advantages of this choice are:

– The LGPL allows proprietary software to dynamically link against `deal.II` without the need to redistribute this software under a restrictive license. On the other hand, the LGPL allows us to leave the full “copyleft” of the library itself intact.
– It avoids licensing incompatibilities with (external) dependencies that are either licensed under (L)GPL or a BSD style license—which would otherwise

make any binary distribution of `deal.II` impossible, for example for inclusion in the package management system of many operating systems. This is especially important given the large number of external packages `deal.II` interacts with these days.

A further account of these considerations can be found in [3].

2.4. Other Changes. The `deal.II` release 8.0 also includes improvements in the following areas:

– Improvements to `deal.II` threading support: The WorkStream class used in `deal.II` to parallelize assembly loops now uses thread local memory, which gives considerably better utilization of caches on multi-core architectures. In addition, a few more linear algebra functions have been parallelized with threads. The new version also allows to directly limit the number of threads used by the task scheduler in Threading Building Blocks at run time.
– `deal.II`'s own iterative solvers have been revised. CG and BiCGStab now use less auxiliary vectors, and GMRES avoids unnecessary work due to reorthogonalization when it is not necessary. This makes behavior more similar to external linear algebra in PETSc and Trilinos (but those remain better adapted to large-scale parallel computers by reducing global communication).
– The interfaces to parallel linear algebra objects of PETSc and Trilinos objects have been further unified to allow for easily exchanging one variant with the other.
– The generic template instantiation mechanism in `deal.II` has been applied to several new places to pre-compile code for all vector types in ErrorEstimator, VectorTools, FETools, ConstraintMatrix.

3. How to cite `deal.II`. In order to justify the work the developers of `deal.II` put into this software, we ask that papers using the library reference one of the `deal.II` papers. This helps us justify the effort we put into it.

There are various ways to reference `deal.II`. To acknowledge the use of a particular version of the library, reference the present document as follows:

```
@article{dealII80,
  title = {The {\tt deal.{I}{I}} Library, Version 8.0},
  author = {W. Bangerth and T. Heister and L. Heltai and G. Kanschat
   and M. Kronbichler and M. Maier and B. Turcksin and T. D. Young},
  journal = {arXiv preprint \url{http://arxiv.org/abs/1312.2266v3}},
  year = {2013},
}
```

The original `deal.II` paper containing an overview of its architecture is [2]. If you rely on specific features of the library, please consider citing any of the following:

– For geometric multigrid: [7, 9];
– For distributed parallel computing: [1];
– For *hp* adaptivity: [5];
– For matrix-free and fast assembly techniques: [10];
– For computations on lower-dimensional manifolds: [6].

4. Acknowledgements. `deal.II` is a world-wide project with dozens of contributors around the globe. Other than the authors of this paper, the following people contributed code to this release: Fahad Alrashed, Juan Carlos Araujo Cabarcas, Denis Davydov, Ashkan Dorostkar, Jörg Frohne, Felix Gruber, Scott Miller, Spencer Patty, Martin Steigemann, Kainan Wang, Christian Wülker. Their contributions are much appreciated!

deal.II and its developers are financially supported through a variety of funding sources. W. Bangerth and B. Turcksin were partially supported by the National Science Foundation under award OCI-1148116 as part of the Software Infrastructure for Sustained Innovation (SI2) program; by the Computational Infrastructure in Geodynamics initiative (CIG), through the National Science Foundation under Award No. EAR-0949446 and The University of California – Davis; and through Award No. KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).

L. Heltai was partially supported by the project OpenSHIP, "Simulazioni di fluidodinamica computazionale (CFD) di alta qualita per le previsioni di prestazioni idrodinamiche del sistema carena-elica in ambiente OpenSOURCE", financed by Regione FVG - POR FESR 2007–2013 Obiettivo competitivita regionale e occupazione.

T. Heister was partially supported by the Computational Infrastructure in Geodynamics initiative (CIG), through the National Science Foundation under Award No. EAR-0949446 and The University of California – Davis and through Award No. KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST).

REFERENCES

[1] W. Bangerth, C. Burstedde, T. Heister, and M. Kronbichler. Algorithms and data structures for massively parallel generic adaptive finite element codes. *ACM Trans. Math. Softw.*, 38:14/1–28, 2011.

[2] W. Bangerth, R. Hartmann, and G. Kanschat. deal.II — a general purpose object oriented finite element library. *ACM Trans. Math. Softw.*, 33(4), 2007.

[3] W. Bangerth and T. Heister. What makes computational open source software libraries successful? *Accepted for publication in Computational Science & Discovery*, 2013.

[4] W. Bangerth and G. Kanschat. Concepts for object-oriented finite element software – the deal.II library. Preprint 1999-43, SFB 359, Heidelberg, 1999.

[5] W. Bangerth and O. Kayser-Herold. Data structures and requirements for *hp* finite element software. *ACM Trans. Math. Softw.*, 36(1):4/1–4/31, 2009.

[6] A. DeSimone, L. Heltai, and C. Manigrasso. Tools for the solution of PDEs defined on curved manifolds with deal.II. Technical Report 42/2009/M, SISSA, 2009.

[7] B. Janssen and G. Kanschat. Adaptive multilevel methods with local smoothing for H^1- and H^{curl}-conforming high order finite element methods. *SIAM J. Sci. Comput.*, 33(4):2095–2114, 2011.

[8] C. Jhurani, T. M. Austin, M. A. Heroux, and J. M. Willenbring. Supporting 64-bit global indices in Epetra and other Trilinos packages – Techniques used and lessons learned. *arXiv*, 1307.6638, 2013.

[9] G. Kanschat. Multi-level methods for discontinuous Galerkin FEM on locally refined meshes. *Comput. & Struct.*, 82(28):2437–2445, 2004.

[10] M. Kronbichler and K. Kormann. A generic interface for parallel cell-based finite element operator application. *Comput. Fluids*, 63:135–147, 2012.