The Distributed and Unified Numerics Environment (DUNE)

Oliver Sander

joint work with a lot of people

ICME Barcelona, 12. 4. 2016

TECHNISCHE
UNIVERSITÄT
DRESDEN

Dune

Distributed and Unified Numerics Environment

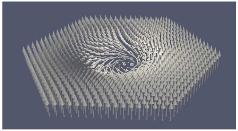
Who I am

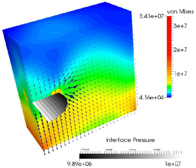
Oliver Sander

- Professor for Numerics of Partial Differential Equations
- TU Dresden, Germany

Fields of research:

- Nonlinear finite element methods
- Computational mechanics
- Materials with orientation
- Nonsmooth problems
- Design and development of simulation software





Standardization in finite element code development

The case for standardization

- Very many finite element codes
- Good reasons to have more than one
- Lots of wheels reinvented

Goals

- Agree on low-level components
- Have a set of separate libraries for basic things
- UNIX philosophy: do one thing only, but do it right

Distributed and Unified Numerics Environment (Dune)

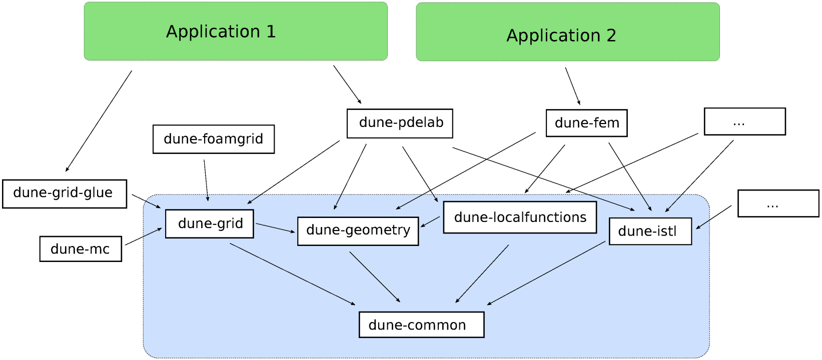
DUNE: Distributed and Unified Numerics Environment

- Set of libraries for grid-based numerical methods
  - Grids
  - Shape functions
  - Linear algebra
  - etc.
- Open source C++ code (License: GPLv2 with linking exception)
- `www.dune-project.org`
- Distributed development

Very short history of DUNE:

- 2003: Started by Peter Bastian (Heidelberg)
- 2006: Split monolithic code into separate modules
- 2011: First run on the entire Jülich supercomputer
- 2012: Starts to appear in Linux Distributions
- 2016: Release 2.4.1

Modular structure

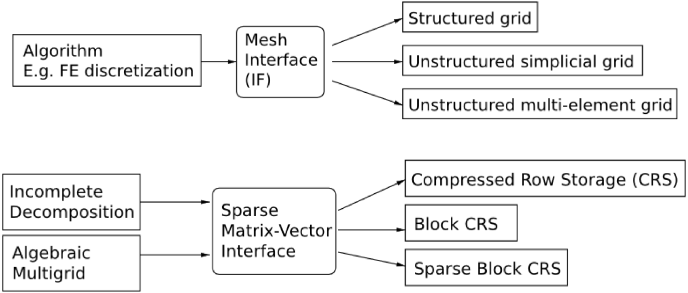


- Collection of separate libraries ("modules")
- Well-defined inter-module dependencies
- Package manager tracks and resolves dependencies
- CMake build system for each module

Abstract interfaces

Separate data structure and algorithms

- Determine what algorithms require from a data structure (`abstract interface')
- Formulate algorithms based in this interface
- Provide different implementations of the interface



Algorithm
E.g. FE discretization

Mesh
Interface
(IF)

Structured grid

Unstructured simplicial grid

Unstructured multi-element grid

Incomplete
Decomposition

Algebraic
Multigrid

Sparse
Matrix-Vector
Interface

Compressed Row Storage (CRS)

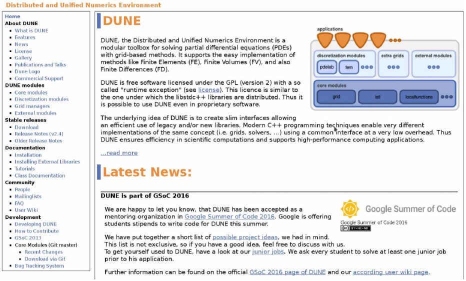
Block CRS

Sparse Block CRS

Development and support

Standard open-source development model

- Project homepage:
  www.dune-project.org
- Gitlab server:
  gitlab.dune-project.org
- Automated testing system
- Active mailing lists
- Yearly developer and user meetings
- Yearly Dune courses



Distributed and Unified Numerics Environment

Home

About DUNE
- What is DUNE
- Features
- News
- License
- Gallery
- Publications and Talks
- Dune Logo
- Commercial Support

DUNE modules
- Core modules
- Discretization modules
- Grid managers
- External modules

Stable releases
- Download
- Release Notes (v2.4)
- Older Release Notes

Documentation
- Installation
- Installing External Libraries
- Tutorials
- Class Documentation

Community
- People
- Mailinglists
- FAQ
- User Wiki

Development
- Developing DUNE
- How to Contribute
- GSoC 2013
- Core Modules (Git master)
  - Recent Changes
  - Download via Git
- Bug Tracking System

DUNE

DUNE, the Distributed and Unified Numerics Environment is a modular toolbox for solving partial differential equations (PDEs) with grid-based methods. It supports the easy implementation of methods like Finite Elements (FE), Finite Volumes (FV), and also Finite Differences (FD).

DUNE is free software licensed under the GPL (version 2) with a so called "runtime exception" (see license). This licence is similar to the one under which the libstdc++ libraries are distributed. Thus it is possible to use DUNE even in proprietary software.

The underlying idea of DUNE is to create slim interfaces allowing an efficient use of legacy and/or new libraries. Modern C++ programming techniques enable very different implementations of the same concept (i.e. grids, solvers, ...) using a common interface at a very low overhead. Thus DUNE ensures efficiency in scientific computations and supports high-performance computing applications.

...read more

Latest News:

DUNE is part of GSoC 2016

We are happy to let you know, that DUNE has been accepted as a mentoring organization in Google Summer of Code 2016. Google is offering students stipends to write code for DUNE this summer.

We have put together a short list of possible project ideas, we had in mind.
This list is not exclusive, so if you have a good idea, feel free to discuss with us.
To get yourself used to DUNE, have a look at our junior jobs. We ask every student to solve at least one junior job prior to his application.

Further information can be found on the official GSoC 2016 page of DUNE and our according user wiki page.

Google Summer of Code Participating Organization 2013, 2016

Commercial support:

- HPC-Simulation-Software & Services (Heidelberg)

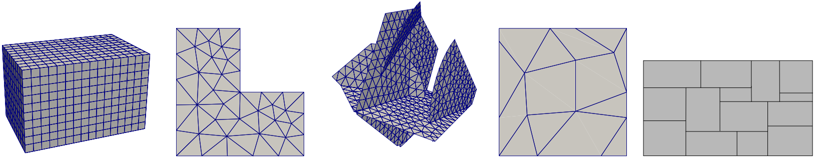
Latest stable release: 2.4.1

- Released on Feb. 29. 2016
- Available from Debian, Ubuntu, OpenSuse, etc.
- Merchandising articles available on request :-)

dune-grid: Finite element grid data structures

Mission statement

Make an abstract interface general enough for anything that people might recognize as a grid. . .



. . . while keeping top performance.

Key features

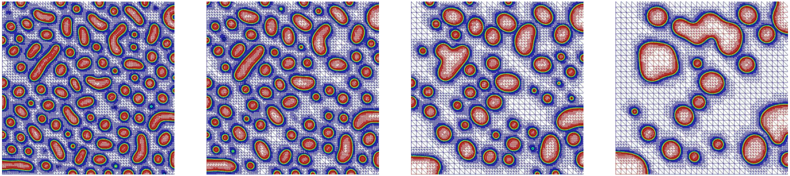
- Grids are completely separate from numerical data
- Use any linear algebra library you want!
- Grids are independent of any particular file format

dune-grid: Features

Local adaptivity

- Grid interface supports wide range of local grid adaptivity strategies
- Grid data structures may or may not implement them
  - Red–green refinement
  - Bisection refinement
  - Nonconforming refinement
  - Anisotropic refinement
  - ...

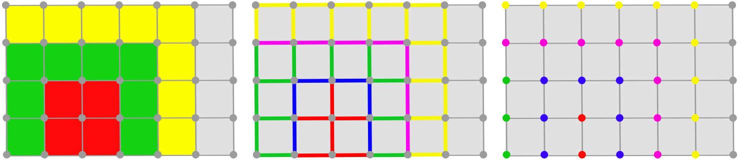
Example: Binary Allen–Cahn equation [Simulations by Carsten Gräser]



dune-grid: Features

Distributed Computing

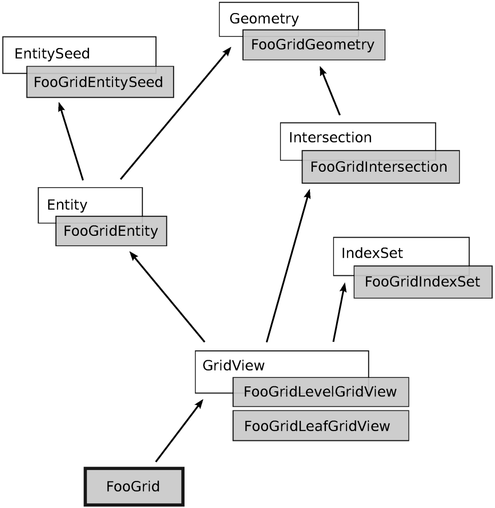
- Grids can be distributed
- Grids implement communication
- Grids implement load balancing



Shared Memory and Vectorization

- Being worked on in the exa-dune project

Dune grid interface classes



Code example (yes, this is real C++!)

Integrate a function `f` over the entire grid:

```
double result = 0.0;

for (const auto& element : elements(gridView))
{
  const auto& quadRule = Dune::QuadratureRules<double,dim>::rule(element.type(),
                                                                 order );
  for (const auto& qp : quadRule)
  {
    auto geometry = element.geometry();

    // Determinant of the Jacobian matrix
    auto det = geometry.integrationElement(qp. position ());

    // global position of the quadrature point
    auto x = geometry.global(qp. position ());

    double result += f(x) * det * qp.weight ();
  }
}
```

Grid implementations

Dedicated DUNE grid implementations

- `YaspGrid`: structured grid
- `OneDGrid`: fully adaptive one-dimensional grid
- `FoamGrid`: 1d and 2d networks in $\mathbb{R}^n$
- `CpGrid`: corner-point grid [from Rasmussen et al., Sintef]

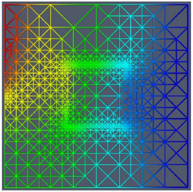
Using external libraries

- `UGGrid`: hybrid grids in 2d/3d, red–green refinement
- `AlbertaGrid`: simplex grids with bisection refinement
- `ALUGrid`: simplex and cube grids with non-conforming refinement
- `P4estGrid`: highly scalable hexahedral grids

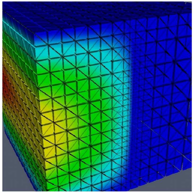
Meta grids: grids parametrized by other grids

- `SubGrid`: select element subset and treat it like a new grid
- `GeometryGrid`: deform any grid into a different shape
- `PrismGrid`: turn any grid into a prism grid one dimension higher

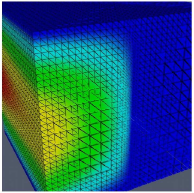
Example: Poisson Problem



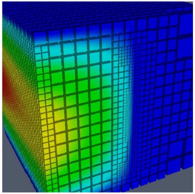
AlbertaGrid, 2d



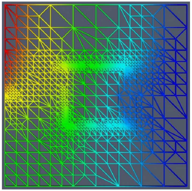
AlbertaGrid, 3d



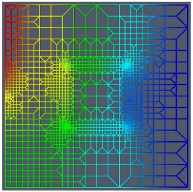
AluSimplexGrid, 3d



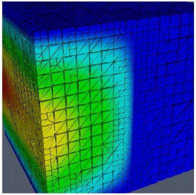
AluCubeGrid, 3d



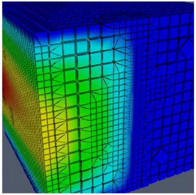
UGGrid, 2d, simplices



UGGrid, 2d, cubes



UGGrid, 3d, simplices



UGGrid, 3d, cubes

Further DUNE modules

dune-localfunctions

- Collection of finite element implementations, with a common interface

dune-functions

- Abstractions for grid functions and bases of grid function spaces

dune-grid-glue

- Compute the geometric intersections between two arbitrary DUNE grids

dune-mc

- Support for level-set methods

dune-pdelab

- Discretizations for many common partial differential equations

[...]

- ...

Thank you for your attention!

Dune

Distributed and Unified Numerics Environment

Questions?

- Ask me now!
- Project homepage: `www.dune-project.org`
- Mailing list: `dune@dune-project.org`
- Ask me after the talk!