EXTENDING PEANO: COUPLING SOLVERS FOR SYSTEMS OF PDES

Dmitry Nikolaenko

Durham University

Introduction

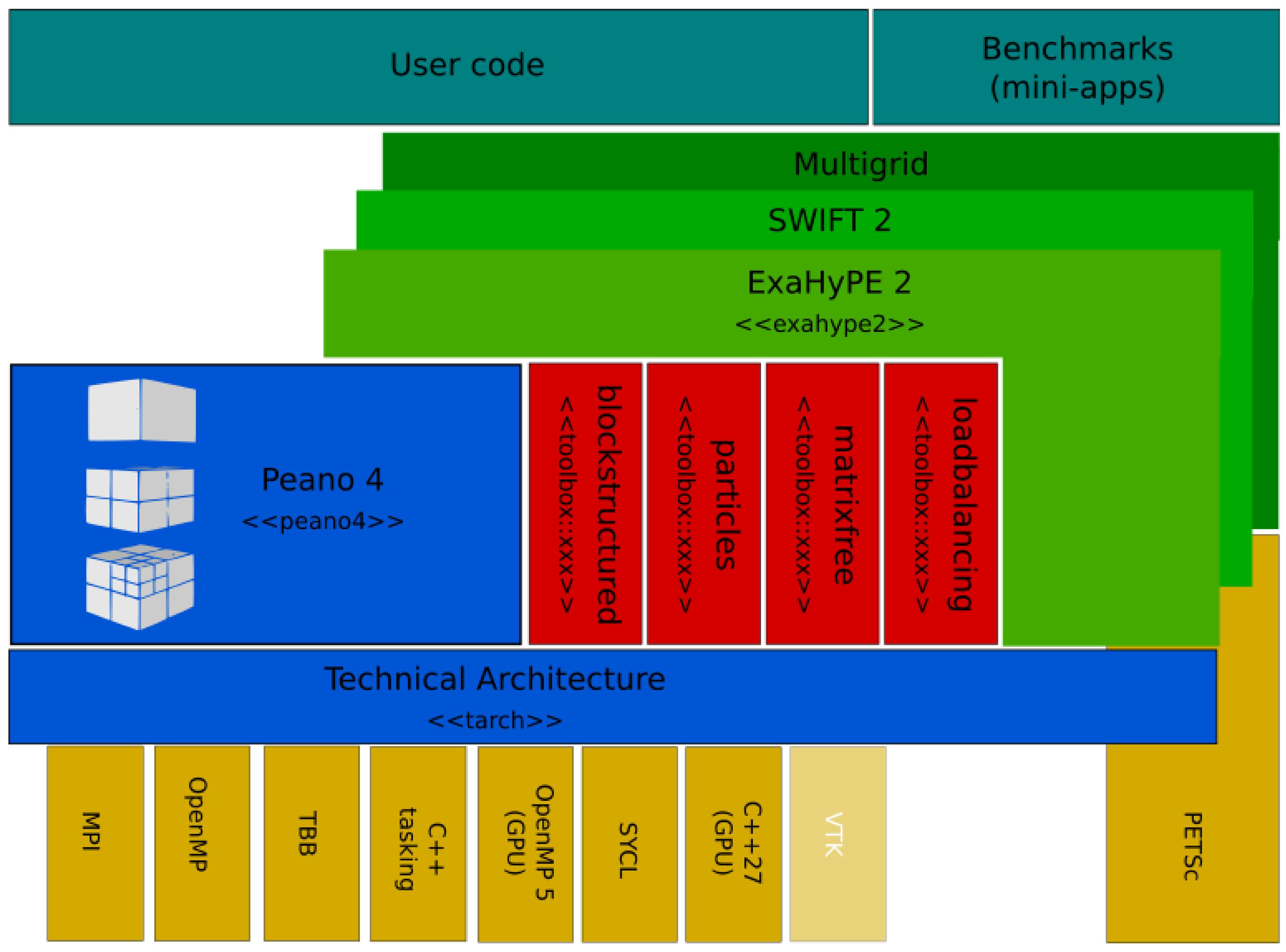
Peano is a *framework* for PDE solvers on hierarchical meshes with adaptive mesh refinement [2] and provides the mesh management, data storage, distribution and mesh traversal on the Peano *space-filling curve*. Peano is massively parallelised and capable of exploiting the full resources of supercomputers. Its current version 4 is shipped with several *specialised extensions* following Peano architecture as well as applications and benchmarks to demonstrate its work and assess its performance.

ExaHyPE2 is one of extensions for solving systems of first-order hyperbolic partial differential equations (PDEs) [1] (e.g., in problems of seismology and astrophysics). It provides a generic engine and collection of solvers (such as finite volume method and higher order ADER discontinuous Galerkin schemes) to solve systems of PDEs.

Multigrid is another extension being developed in collaboration with mathematicians from Bath University. It implements elliptic solvers based on multigrid methods using a hierarchy of discretisations.

Every Peano application, including ones based on the above extensions, is a C++ code which follows the unified Peano architecture consisting of several layers:

- Technical Architecture «tarch»
- Peano core «peano4»
- Various toolboxes, on top of which extensions are built, e.g.:
  - «toolbox::blockstructured» for blockstructured meshes
  - «toolbox::particles» for particle management
  - «toolbox::loadbalancing» for dynamic load balancing



General workflow

ExaHyPE extension:

1. Create an «exahype2» Project
2. Implement flux, eigenvalue, source terms
3. Instantiate solvers and add them to the Project
4. Configure and generate a «peano4» Project

Multigrid extension:

1. Create an «mghype» Project
2. Construct matrices
3. Instantiate solvers and add them to the Project
4. Configure and generate a «peano4» Project

Motivation

The Python API simplifies creating applications by writing python scripts within the framework of one Peano extensions.

- **Old approach** Pick one extension suitable for the given problem (reformulate the problem if necessary) and write within it
- **New approach** Identify parts of the given complex problem suitable for solving by different extensions, implement them each within its extension (e.g. solve hyperbolic equations in *ExaHyPE* and elliptic in *Multigrid*) and generate an application coupling solvers from both

Example problem formulation [3]

Hyperbolic problem for Euler equation:

$$\begin{cases} \partial_t Q + \nabla \cdot F(Q) = S(Q), \\ Q = (\rho, v, E) \in \mathbb{R}^5 \text{ subject to modified RHS,} \end{cases}$$

Dirichlet problem for Poisson equation:

$$\begin{cases} \Delta u = -f(x,y) & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases}$$

Problem

Implement a coupling between a hyperbolic solver (provided by *ExaHyPE*) for the Euler equation and an elliptic solver (provided by *Multigrid*) for the Poisson equation.

Solution

1. Generate the 1st *Peano* application from one extension, e.g. *Multigrid*, following the same workflow as above
2. Generate the 2nd *Peano* application from another extension, e.g. *ExaHyPE*, in a similar fashion
3. (new feature) Merge the above 2 «peano4» Projects coupling solvers from both

Discussion

In line with the theme proposed for *RSECon24*, this work tries to follow some guiding principles of FAIR for Research Software:

- New ways of coupling solvers for different PDE systems are realised which weren't possible before, thus improving *interoperability*. Various Peano extensions, such as *ExaHyPE* and *Multigrid*, are integrated.
- In turn, *reusability* of the extensions as building blocks is enhanced.

Demonstration/Results

Preliminary results

...

...

...

...

References

[1] Anne Reinarz et al. "ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems". In: *Computer Physics Communications* 254 (2020). ISSN: 0010-4655. DOI: `https://doi.org/10.1016/j.cpc.2020.107251`. URL: `https://www.sciencedirect.com/science/article/pii/S001046552030076X`.

[2] T. Weinzierl. "The Peano software—parallel, automaton-based, dynamically adaptive grid traversals". In: TOMS, 45(2), 2019.

[3] Han Zhang et al. "Spherical accretion of collisional gas in modified gravity I: self-similar solutions and a new cosmological hydrodynamical code". In: *Monthly Notices of the Royal Astronomical Society* 515.2 (2022), pp. 2464–2482.

Contact:
Dmitry Nikolaenko
dmitry.nikolaenkodurham.ac.uk