

Peano4/ExaHyPE2

Holger Schulz holger.schulz@durham.ac.uk
Marion Weinzierl marion.weinzierl@durham.ac.uk
Tobias Weinzierl tobias.weinzierl@durham.ac.uk

<http://www.peano-framework.org/>

User code

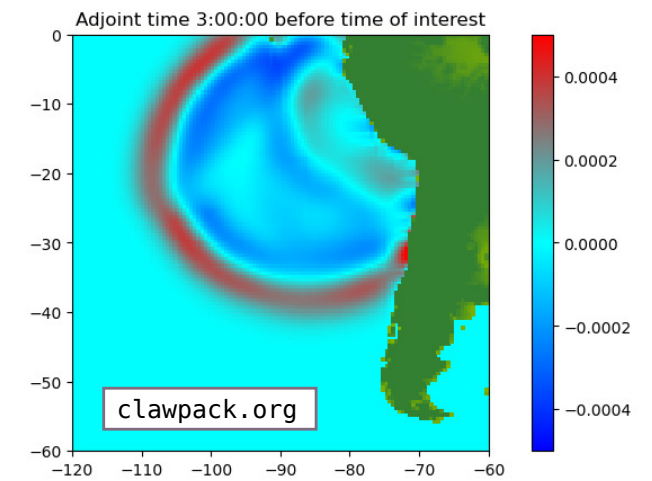
```
jupyter Euler (autosaved)
File Edit View Insert Cell Kernel Help Not Trusted Python 3
In [3]: project = exahype2.Project( ["examples", "exahype2", "euler"], "finitevolumes" )

A Finite Volumes solver
Our first step is to use a Finite Volumes solver. So we add this solver to the project. An ExaHyPE mesh can carry multiple solvers at the same time, but we only use one here.

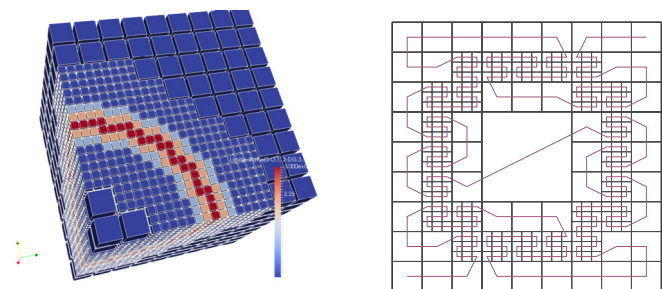
In [4]: patch_size = 9
unknowns = 5
auxiliary_variables = 0 # This could be something alike material parameters. Not required
time_step_size = 0.0001
max_h = 0.2
min_h = max_h
solver = exahype2.solvers.fv.GenericRusanovFixedTimeStepSize(
    "Euler", patch_size, unknowns, auxiliary_variables, min_h, max_h, time_step_size,
    flux=exahype2.solvers.fv.PDETerms.User_Defined_Implementation
)
project.add_solver( solver )
```

Clawpack support

- Fortran adapter + python glue code couple Clawpack to peano
- Shallow water simulation done
- Next: Read in bathymetry and simulate Chile tsunami

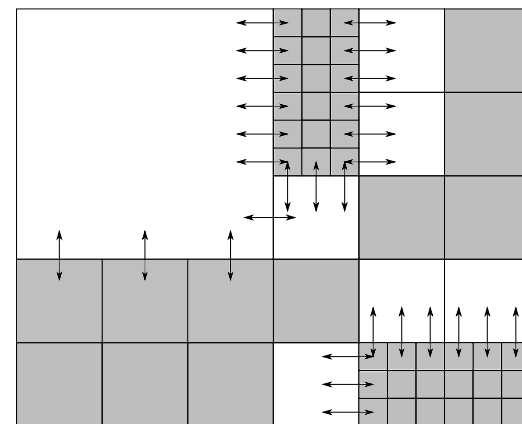


Peano



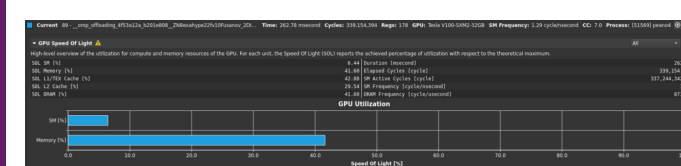
- Dynamically adaptive cartesian meshes
- Block structured AMR
- Space filling curves for domain decomposition

Enclave tasking



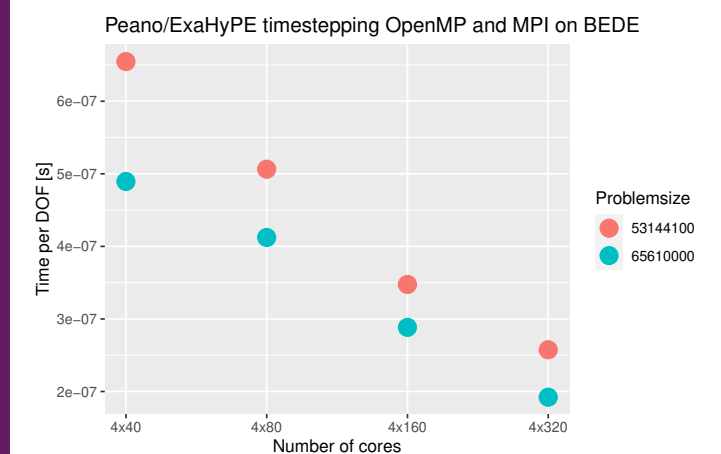
- High priority Skeleton cells (white):
- at AMR boundaries
 - take part in MPI exchanges
 - spawn tasks
- Low priority Enclave cells (gray):
- computationally intense
 - scheduled by (OpenMP) runtime
 - run on host or device
 - overlap with Skeleton tasks

GPU offloading



- OpenMP + LLVM (+ CUDA)
- Restructuring of functions to increase concurrency
- Offload expensive low priority ("Enclave") tasks
- Routinely millions of threads on V100 (Bede)
- Todo: OpenMP + GNU, Intel compilers

Scaling



- CPU only scaling on Bede
- Computationally cheap Euler

ExaHyPE

$$\frac{\partial Q}{\partial t} + \nabla \cdot F(Q) + \sum_{i=1}^d B_i(Q) \frac{\partial Q}{\partial x_i} = S(Q) \quad \text{with } Q : \mathbb{R}^{d+1} \mapsto \mathbb{R}^m$$

- Engine for solving hyperbolic PDEs (e.g. seismic, gravitational waves)
- Scalable infrastructure
- ADER-DG + FV
- C++, MPI, OpenMP
- User: thin Python layer for code generation



References

Clawpack Development Team (2020), Clawpack Version 5.7.1, <http://www.clawpack.org>, doi: 10.5281/zenodo.4025432

Weinzierl, T. "The Peano Software—Parallel, Automaton-based, Dynamically Adaptive Grid Traversals." ACM Transactions on Mathematical Software (TOMS) 45 (2019): 1 - 41.

Enclave Tasking for DG Methods on Dynamically Adaptive Meshes Dominic Etienne Charrier, Benjamin Hazelwood, and Tobias Weinzierl SIAM Journal on Scientific Computing 2020 42:3, C69-C96

ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems, Reinartz et. al., Computer Physics Communications 2020, V254, 107251