Towards an Exascale-Ready Mini-App for Smooth Particle Hydrodynamics

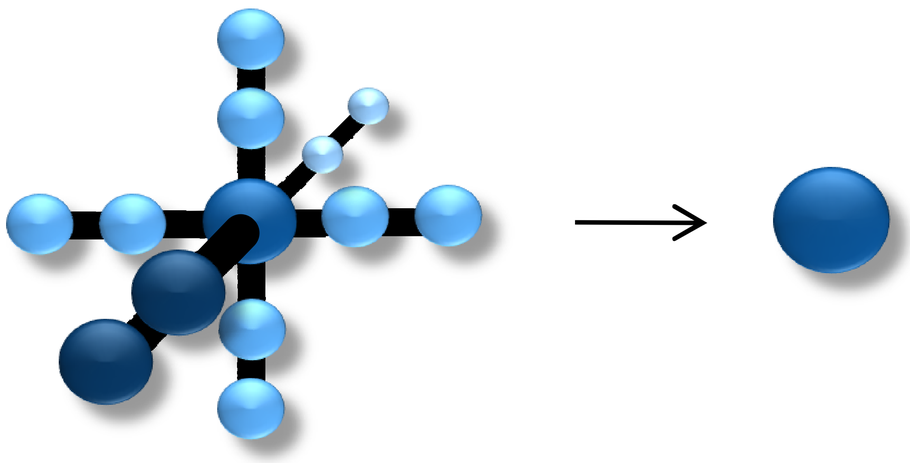
University of Basel

Florina M. Ciorba, Lucio Mayer, Rubén Cabezon,David Imbert,
Danilo Guerrera, Aurélien Cavelan, Darren S. Reed, Jean-Guillaume Piccinali,
Ioana Banicescu, Domingo Garciá-Senz, and Thomas R. Quinn

1. Vision

- Evaluate Smooth Particle Hydrodynamics codes in terms of performance
- Understand the limits of the actual implementations
- Derive a mini-app that synthesizes their characteristics
- Provide a optimized implementations of basics SPH operands

2. Smooth Particle Hydrodynamics codes

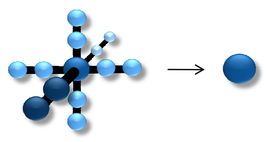


SPH Characteristics

- purely Lagrangian method (meshless)
- used in numerical simulations of fluids in astrophysics and computational fluid dynamics
- computationally demanding

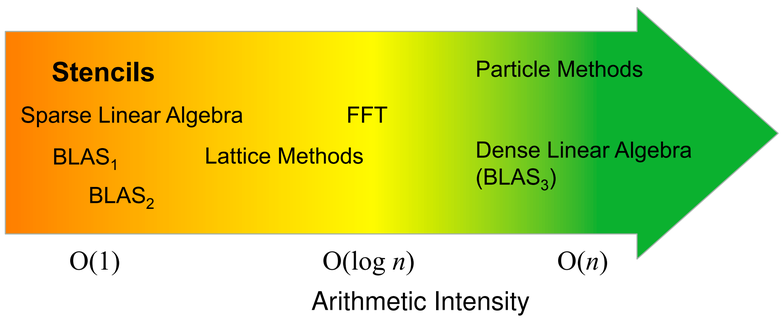
3. Rotating Square Patch

The stencil motif has manifold applications in science, ranging from weather forecast to image processing.



13 Points Stencil

Stencil computations are characterized by low arithmetic intensity: they are memory-bound.

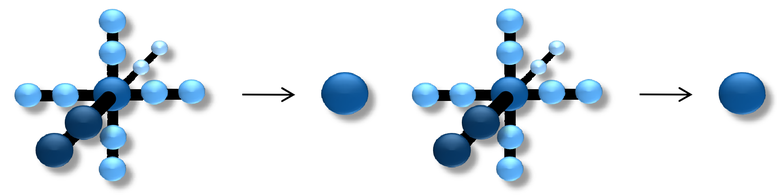


Sophisticated parallelization techniques are required in order to arrive at scalable solutions: several stencil compilers are available.

4. Experimental results

We use two stencil compilers:

- PATUS [3] defines a DSL to express stencil and exploits auto-tuning
- PLUTO [4], a source to source compiler that uses the polyhedral model approach for compiler optimization



Mint, University of Basel LiMa, University of Erlangen

Our reference is a naive implementation of the classical 3-D wave equation, with NUMA-aware initialization, parallelized with OpenMP.

University of Basel 2016

Problem	Calculate a 3-D wave equation of 200^3 elements (IEEE single precision arithmetic) in 100 timesteps
System	**SW:** OpenMP 4.0, GCC 4.9.2, PATUS 0.1.4, PLUTO 0.10 **HW:** 1 node • CPU: 2x AMD Opteron 6274 "Bulldozer" 16-Core, 2.2 GHz, 12 MiB L3 cache, 4 NUMA domains • RAM: 256 GiB • OS: Ubuntu 14.04.4, Kernel 3.8.0-38
Method	1. Naive OpenMP implementation with NUMA aware initialization (16 FLOPS) 2. DSL + auto-tuning with PATUS (20 FLOPS) 3. Polyhedral model with PLUTO (16 FLOPS)

University of Erlangen 2016

Problem	Calculate a 3-D wave equation of 200^3 elements (IEEE single precision arithmetic) in 100 timesteps
System	**SW:** OpenMP 4.0, GCC 4.9.2, PATUS 0.1.4, PLUTO 0.10 **HW:** 1 node • CPU: 2x Xeon 5650 "Westmere" 6 cores + SMT, 2.66 GHz, 12 MiB Shared Cache per chip, 2 NUMA domains • RAM: 24 GB (DDR3-1333) • OS: CentOS 6.7, Kernel 2.6.32-573.7.1.el6
Method	1. Naive OpenMP implementation with NUMA aware initialization (16 FLOPS) 2. DSL + auto-tuning with PATUS (20 FLOPS) 3. Polyhedral model with PLUTO (16 FLOPS)

University of Erlangen 2016

Problem	Calculate a 3-D wave equation of 200^3 elements (IEEE single precision arithmetic) in 100 timesteps
System	**SW:** OpenMP 4.0, GCC 4.9.2, PATUS 0.1.4, PLUTO 0.10 **HW:** 1 node • CPU: 2x Xeon 5650 "Westmere" 6 cores + SMT, 2.66 GHz, 12 MiB Shared Cache per chip, 2 NUMA domains • RAM: 24 GB (DDR3-1333) • OS: CentOS 6.7, Kernel 2.6.32-573.7.1.el6
Method	1. Naive OpenMP implementation with NUMA aware initialization (16 FLOPS) 2. DSL + auto-tuning with PATUS (20 FLOPS) 3. Polyhedral model with PLUTO (16 FLOPS)

5. SPH-EXA Mini-app

What we have:

- extrapolate common features
- provide a reference optimized implementation (MPI+X)
- provide a library for SPH simulations

What we pursue:

- Integration of likwid [2] into PROVA!
- Evaluation of new compilers
- Stencil applications tuning
- Develop a performance model

Interested?

- https://prova.io

6. References

[1] Antonio Maffia, Helmar Burkhart and Danilo Guerrera. Reproducibility in Practice: Lessons Learned from Research and Teaching Experiments. In *Euro-Par 2015: Parallel Processing Workshops*, 2015.

[2] Jan Treibig, Georg Hager, and Gerhard Wellein. LIKWID: A lightweight performance-oriented tool suite for x86 multicore environments. In *Proceedings of PSTI2010, the First International Workshop on Parallel Software Tools and Tool Infrastructures*, 2010.

[3] Matthias Christen, Olaf Schenk, and Helmar Burkhart. PATUS: A Code Generation and Autotuning Framework for Parallel Iterative Stencil Computations on Modern Microarchitectures. In *Proceedings of the 2011 IEEE International Parallel & Distributed Processing Symposium*, 2011.