Recent developments in HPX and Octo-Tiger

Patrick Diehl
Joint work with: Gregor Daiß, Sagiv Schieber, Dominic Marcello, Kevin Huck, Hartmut Kaiser, Juhan Frank, Geoffery Clayton, Patrick Motl, Dirk Pflüger, Orsola DeMarco, Mikael Simberg, John Biddiscombe, and many more

Center for Computation & Technology, Louisiana State University

patrickdiehl@lsu.edu

October 2021

Motivation

At peak brightness, the rare 2002 red nova V838 Monocerotis briefly rivalled the most powerful stars in the Galaxy. Credit: NASA/ESA/H. E. Bond (STScI)

Goal

Simulate the merger and obtain the light curve to understand the observations better:

Multi-physic is need:

- Hydro
- Gravity
- Radiation

Reference

- Tylenda, R., et al. "V1309 Scorpii: merger of a contact binary." Astronomy & Astrophysics 528 (2011): A114.

Outline

Software framework

Octo-Tiger

Astrophysics open source program[1] simulating the evolution of star systems based on the fast multipole method on adaptive Octrees.

Modules

- Hydro
- Gravity
- Radiation (benchmarking)

Supports

- Communication: MPI/libfabric
- Backends: CUDA, HIP, Kokkos

Reference

- Marcello, Dominic C., et al. "octo-tiger: a new, 3D hydrodynamic code for stellar mergers that uses hpx parallelization." Monthly Notices of the Royal Astronomical Society 504.4 (2021): 5345-5382.

[1] https://github.com/STEllAR-GROUP/octotiger

Example of a merger simulation

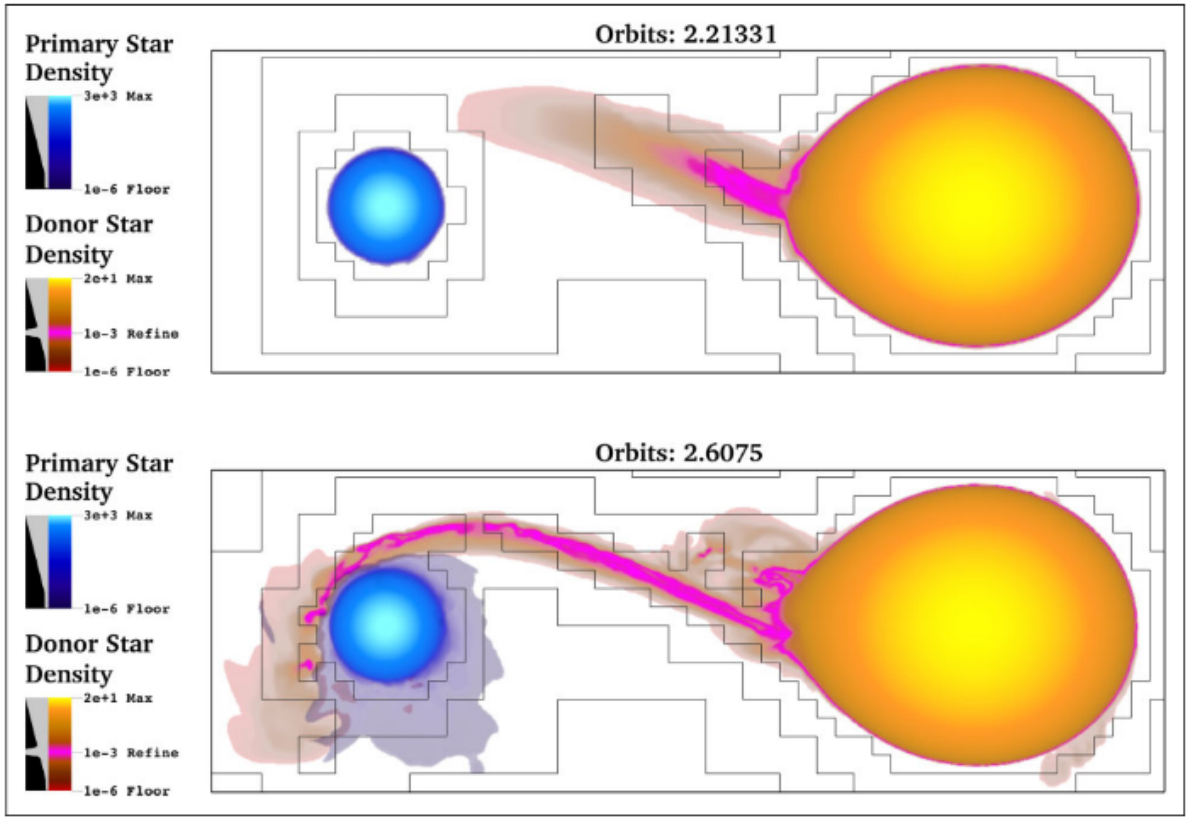


Figure 2. The early stages of mass transfer in a binary star system. The accreting star is five times more massive than the donor star.

Reference

- Heller, Thomas, et al. "Harnessing billions of tasks for a scalable portable hydrodynamic simulation of the merger of two stars." The International Journal of High Performance Computing Applications 33.4 (2019): 699-715.

Example of a merger simulation

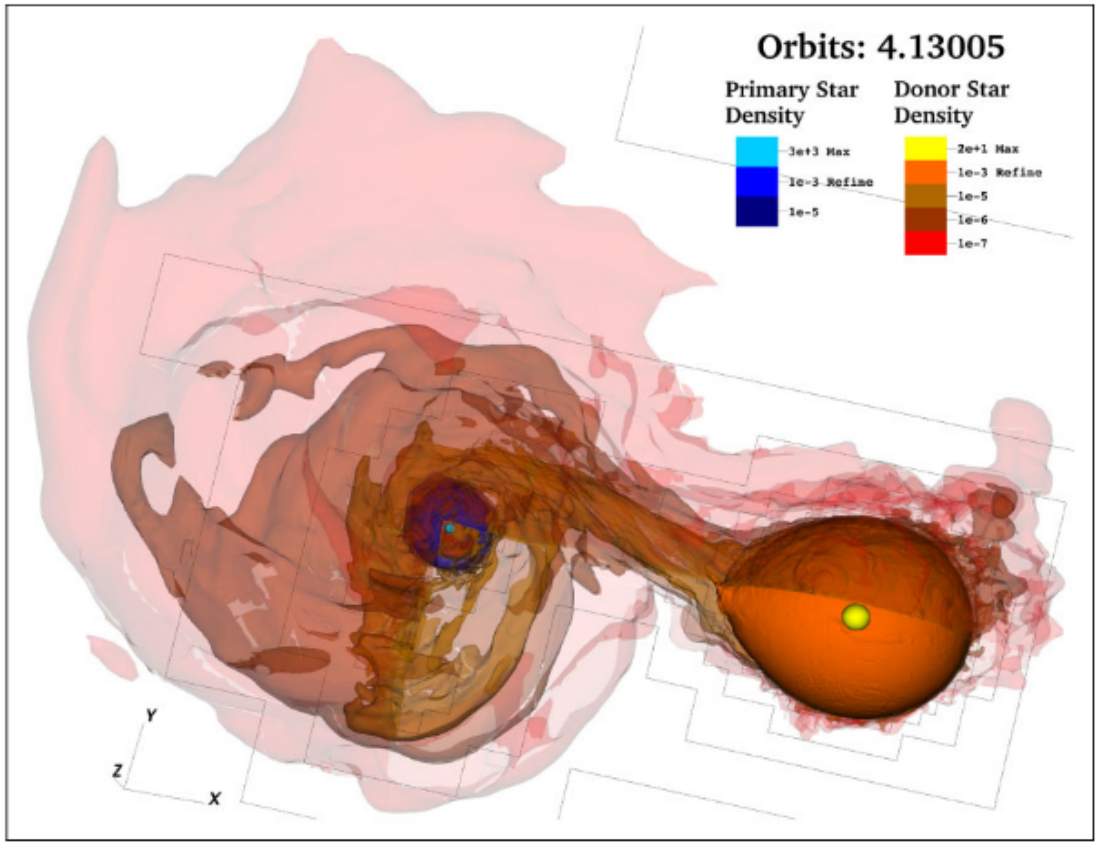


Figure 3. A 3-D contour plot of the system in Figure 2 after an accretion disc begins to form. 3-D: three-dimensional.

Reference

- Heller, Thomas, et al. "Harnessing billions of tasks for a scalable portable hydrodynamic simulation of the merger of two stars." The International Journal of High Performance Computing Applications 33.4 (2019): 699-715.

HPX

HPX is a open source C++ Standard Library for Concurrency and Parallelism[2].

Features

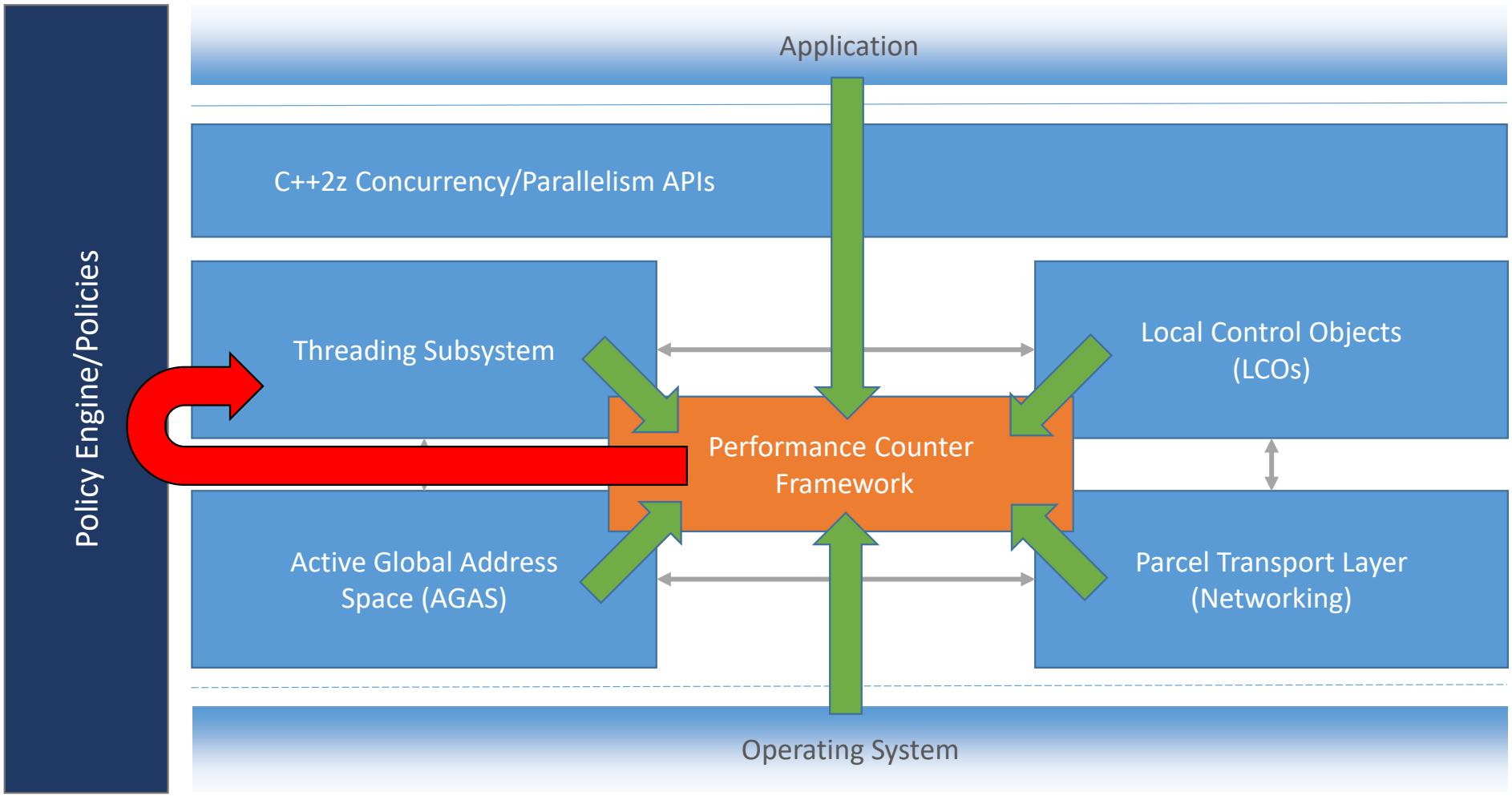
- HPX exposes a uniform, standards-oriented API for ease of programming parallel and distributed applications.
- HPX provides unified syntax and semantics for local and remote operations.
- HPX exposes a uniform, flexible, and extendable performance counter framework which can enable runtime adaptivity.

Reference

- Kaiser, Hartmut, et al. "Hpx-the c++ standard library for parallelism and concurrency." Journal of Open Source Software 5.53 (2020): 2352.

[2] https://github.com/STEllAR-GROUP/hpx

HPX's architecture

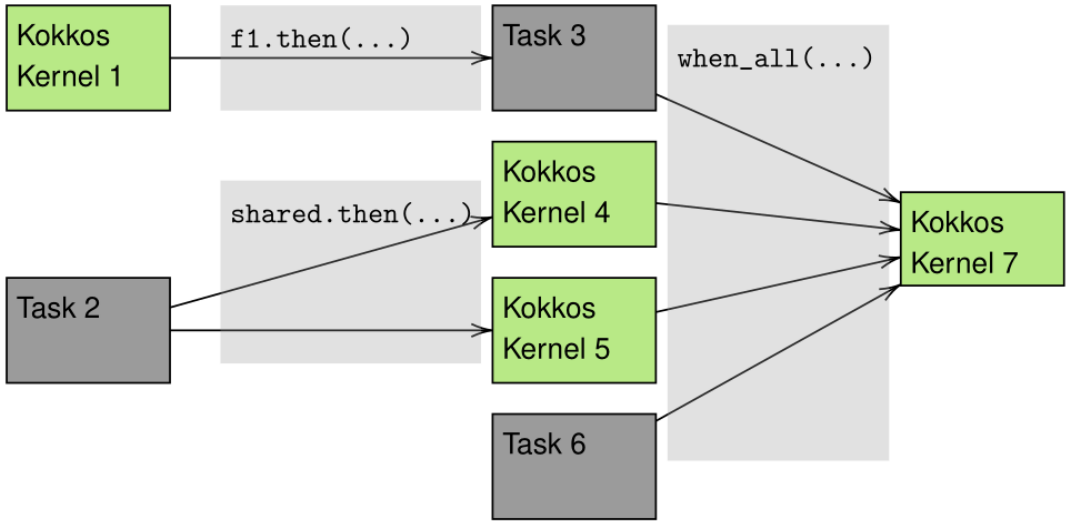


Reference

- Kaiser, Hartmut, et al. "Hpx-the c++ standard library for parallelism and concurrency." Journal of Open Source Software 5.53 (2020): 2352.

HPX support in Kokkos

- Combine Tasks via futures in different ways
- HPX-Kokkos integration needs to work for both host-side and device-side execution
- Device-Side execution: Kernels need to be futurized to be integrated into the DAG
- → Use uderlying CUDA/HIP Api with callbacks or events to set hpx futures ready automatically

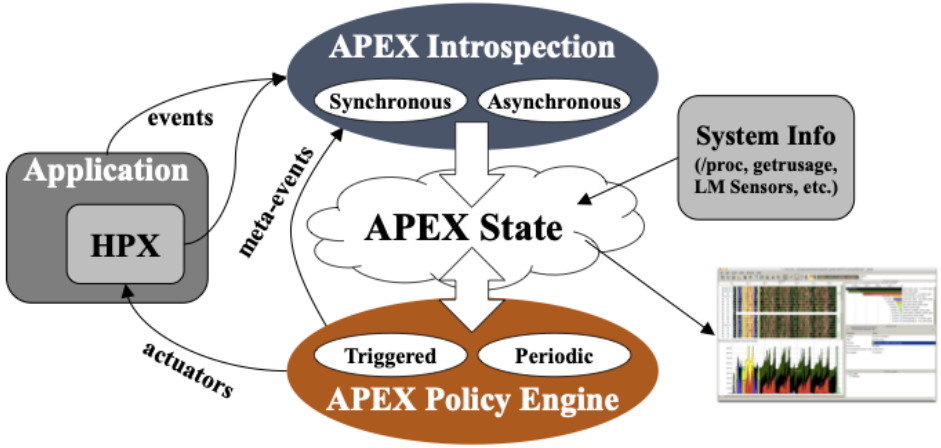


Reference

- Edwards, H. Carter, Christian R. Trott, and Daniel Sunderland. "Kokkos: Enabling manycore performance portability through polymorphic memory access patterns." Journal of parallel and distributed computing 74.12 (2014): 3202-3216.
- Daiß, Gregor, et al. "Beyond Fork-Join: Integration of Performance Portable Kokkos Kernels with HPX." 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE, 2021.

APEX

- APEX: Autonomous Performance Environment for Exascale: Performance measurement library for distributed, asynchronous multitasking systems.



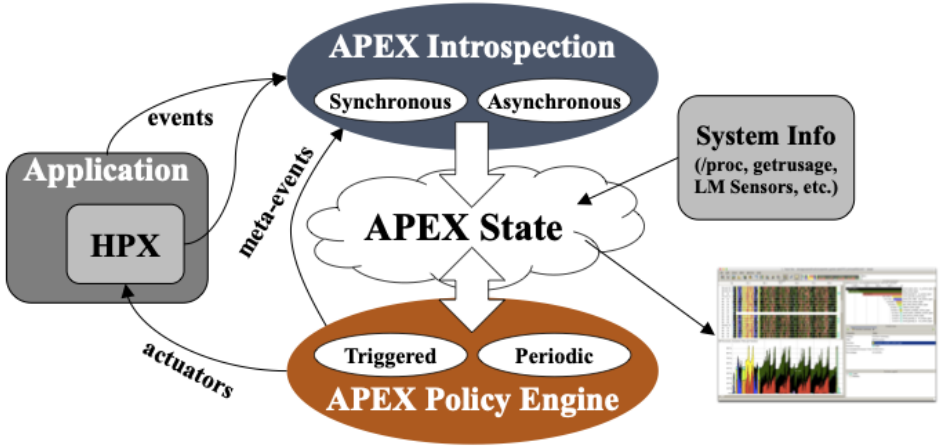
- CUPTI used to capture CUDA events
- NVML used to monitor the GPU
- OTF2 and Google Trace Events trace output
- Task Graphs and Trees
- Scatterplots of timers and counters

Reference

- Huck, Kevin A., et al. "An autonomic performance environment for exascale." Supercomputing frontiers and innovations 2.3 (2015): 49-66.

APEX

- To support performance measurement in systems that employ user-level threading, APEX uses a dependency chain in addition to the call stack to produce traces and task dependency graphs.



- CUPTI used to capture CUDA events
- NVML used to monitor the GPU
- OTF2 and Google Trace Events trace output
- Task Graphs and Trees
- Scatterplots of timers and counters

Reference

- Huck, Kevin A., et al. "An autonomic performance environment for exascale." Supercomputing frontiers and innovations 2.3 (2015): 49-66.

Scaling

Synchronous (MPI) vs asynchronous communication (libfabric)

Configuration

	Piz Daint
CPU	1 × Intel®Xeon™E5-2690 v3, 2.60GHz, 12 cores
GPU	1 × NVIDIA®Tesla®P100
RAM	64 GB
IC	Cray Aries routing and communications ASIC

Table 3: Configuration of Piz Daint.

Level of refinement	sub-grids	memory usage (GB)
13	5,417	8
14	10,928	16.37
15	42,947	56.92
16	$2.24 \cdot 10^5$	271.94
17	$1.5 \cdot 10^6$	2,305.92

Table 4: Number of tree nodes (sub-grids) per level of refinement (LoR) and the memory usage of the corresponding level.

Reference

- Daiß, Gregor, et al. "From piz daint to the stars: Simulation of stellar mergers using high-level abstractions." Proceedings of the international conference for high performance computing, networking, storage and analysis. 2019.

Synchronous vs asynchronous communication

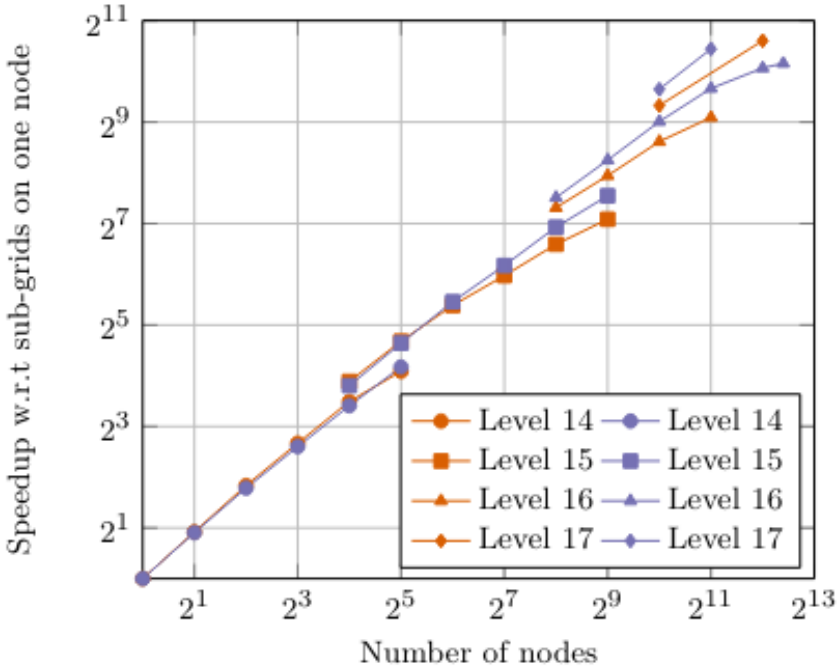


Figure 2: Relative speedup with respect to the processed sub-grids on one node for level 14. The red lines show the results using HPX's MPI parcelport and the blue lines using HPX's libfabric parcelport, respectively. Note that for level 16 and level 17 some data points are missing due to restricted node hours for development projects.

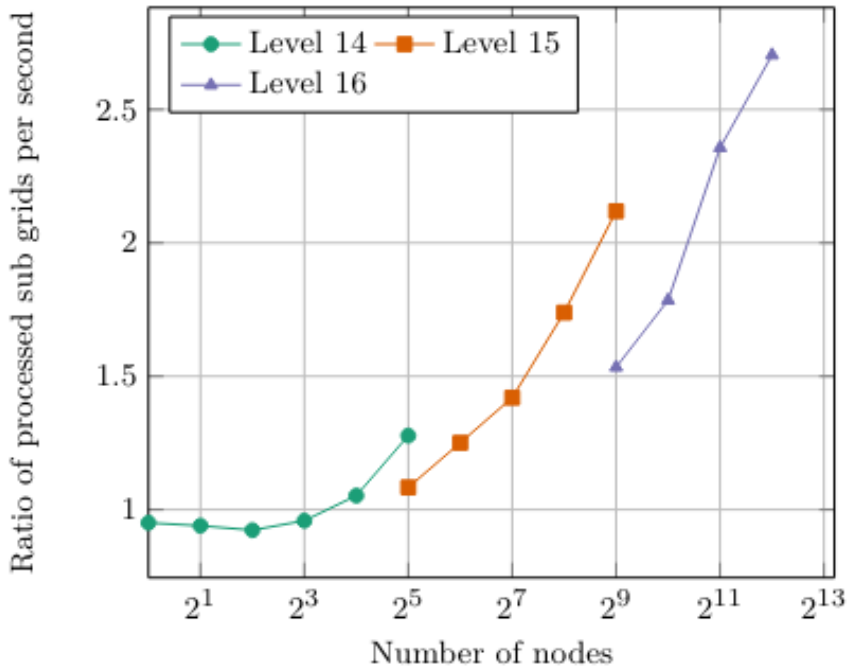


Figure 3: Ratio of processed sub grids per second between HPXs libfabric and MPI Parcelport on Piz Daint (higher numbers mean libfabric is faster).

Reference

- Daiß, Gregor, et al. "From piz daint to the stars: Simulation of stellar mergers using high-level abstractions." Proceedings of the international conference for high performance computing, networking, storage and analysis. 2019.

Scaling on ORNL's Summit

Node level scaling: Hydro

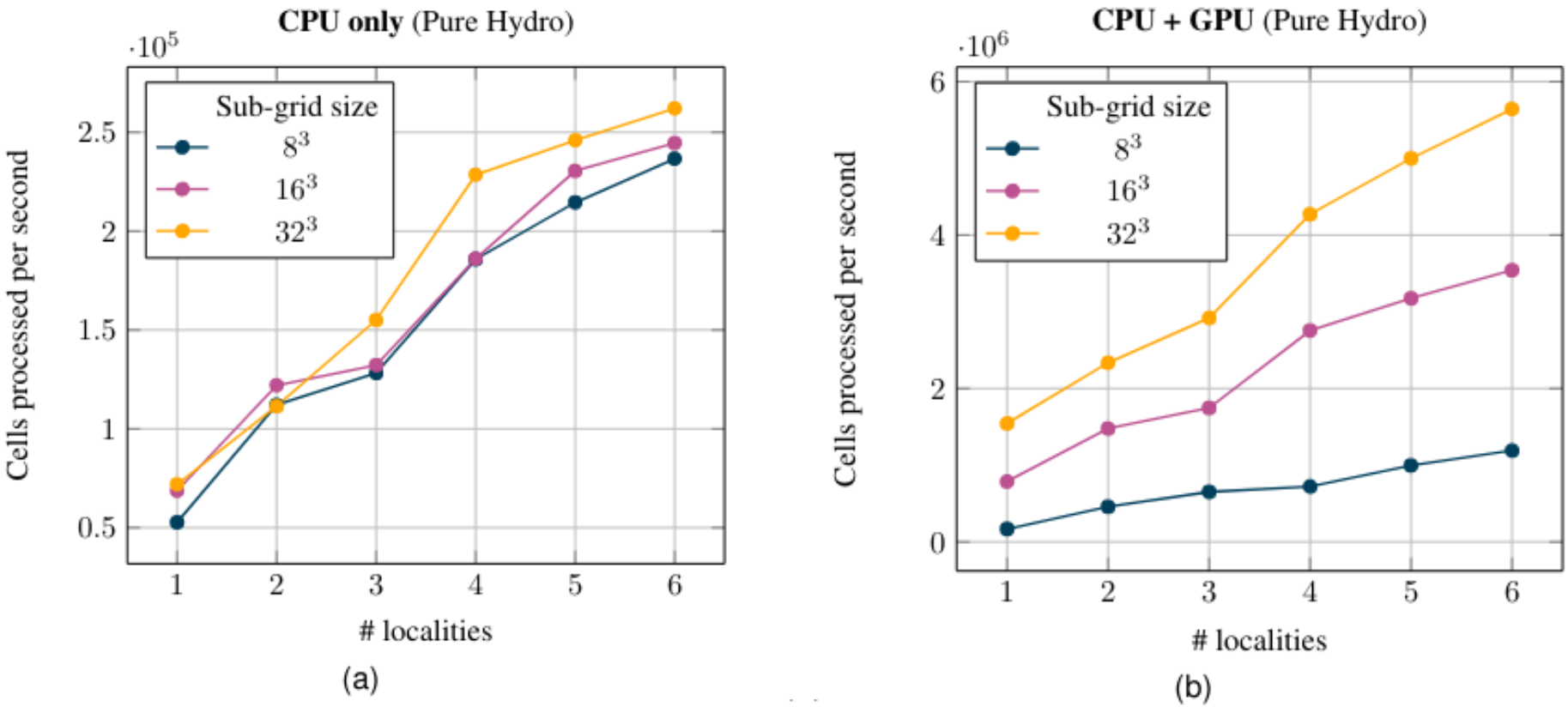


Figure 2: Cells processed per second for the node level scaling. For one up to 6 localities on one Summit node. One locality was assigned to seven CPUs and one NVIDIA® V100 GPU.

Reference

- Diehl, Patrick, et al. "Octo-Tiger's New Hydro Module and Performance Using HPX+ CUDA on ORNL's Summit." arXiv:2107.10987 (2021). (Accepted IEEE Cluster 21)

Distributes scaling: Hydro

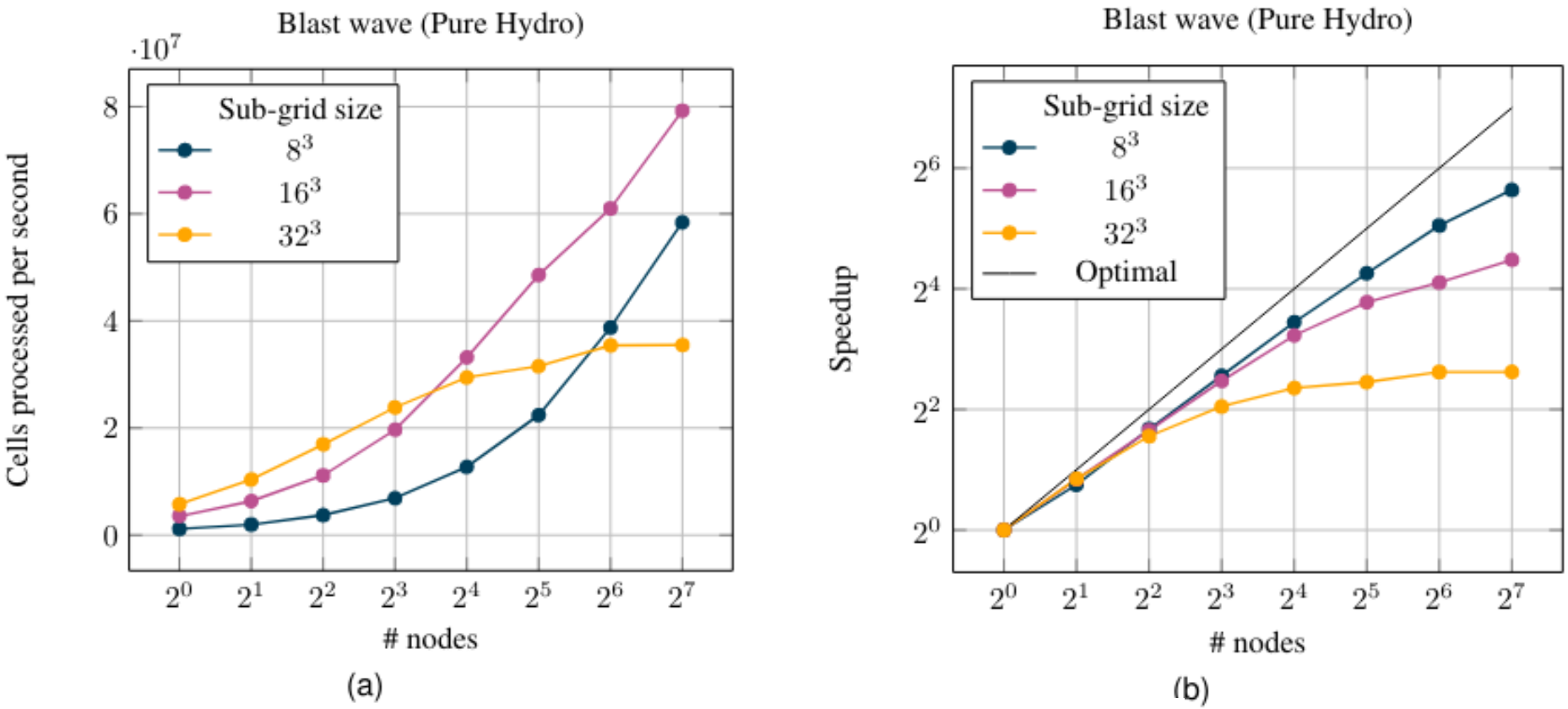


Figure 3: Cells processed per second for the distributed scaling from one Summit node up to 128 Summit nodes. Note that all six NVIDIA® V100 GPUs per node were used.

Reference

- Diehl, Patrick, et al. "Octo-Tiger's New Hydro Module and Performance Using HPX+ CUDA on ORNL's Summit." arXiv:2107.10987 (2021). (Accepted IEEE Cluster 21)

Node level scaling: Hydro + Gravity

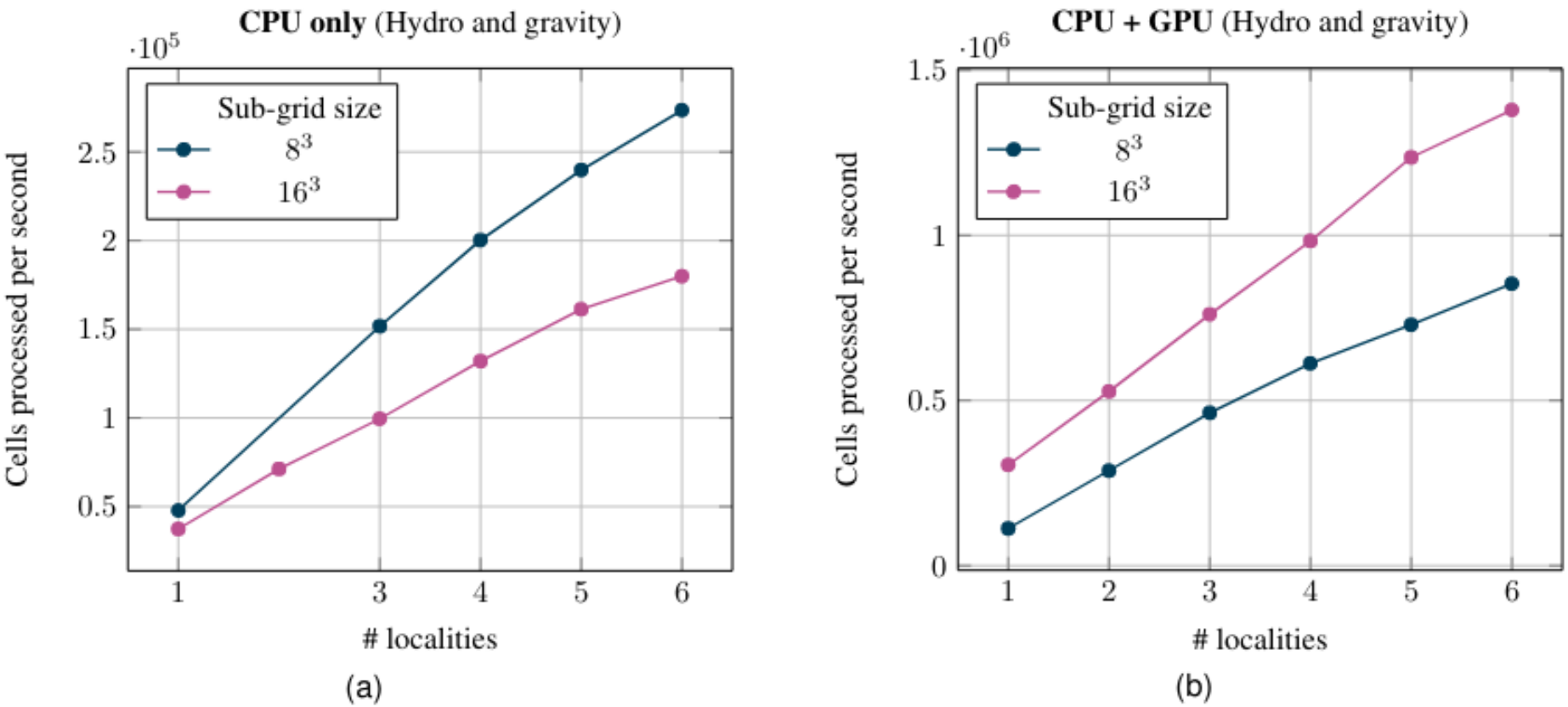


Figure 4: Cells processed per second for the node level scaling. For one up to 6 localities on one Summit node. One locality was assigned to seven CPUs and one NVIDIA® V100 GPU.

Reference

- Diehl, Patrick, et al. "Octo-Tiger's New Hydro Module and Performance Using HPX+ CUDA on ORNL's Summit." arXiv:2107.10987 (2021). (Accepted IEEE Cluster 21)

Distributed scaling: Hydro + Gravity

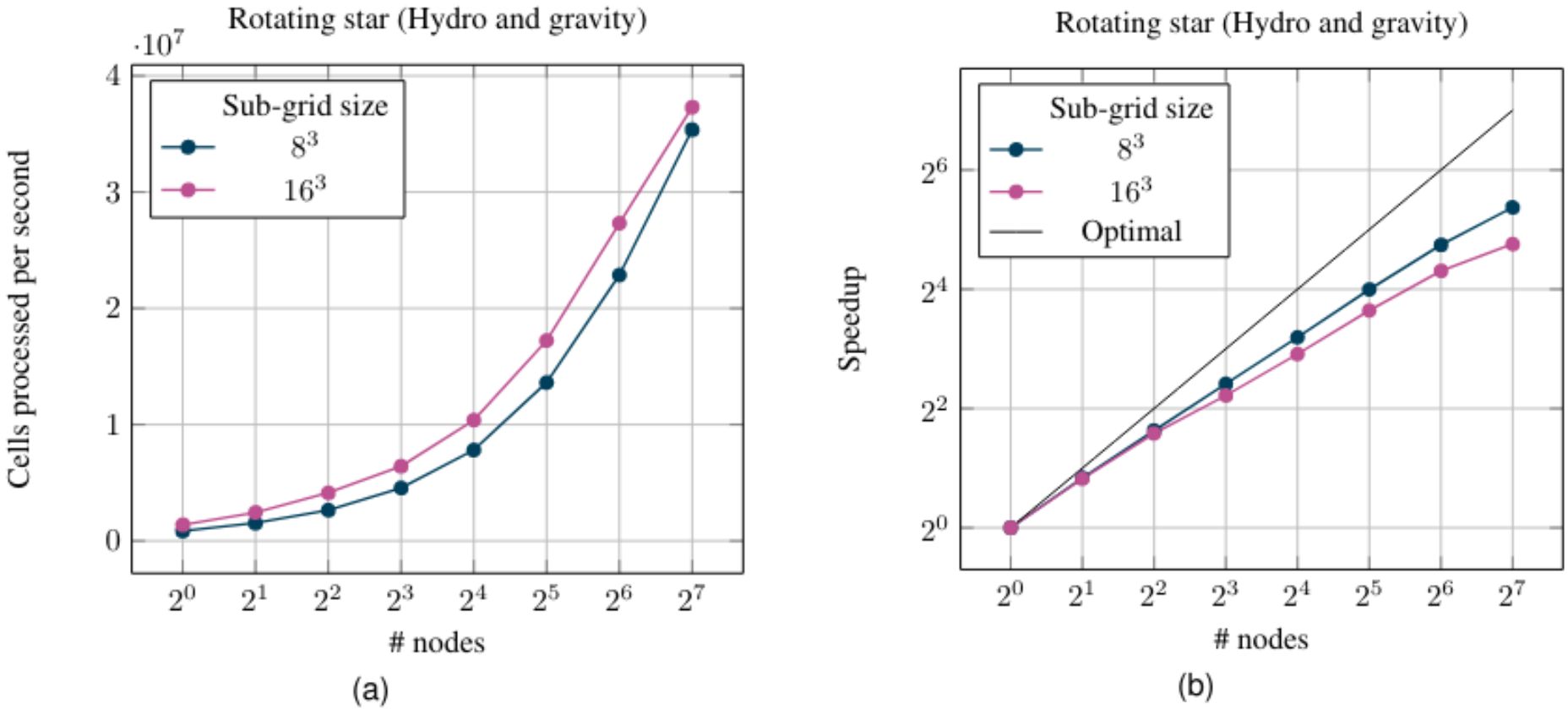


Figure 5: Cells processed per second for the distributed scaling from one Summit node up to 128 Summit nodes. Note that all six NVIDIA ® V100 GPUs per node were used.

Reference

- Diehl, Patrick, et al. "Octo-Tiger's New Hydro Module and Performance Using HPX+ CUDA on ORNL's Summit." arXiv:2107.10987 (2021). (Accepted IEEE Cluster 21)

Kokkos - HPX

Overhead

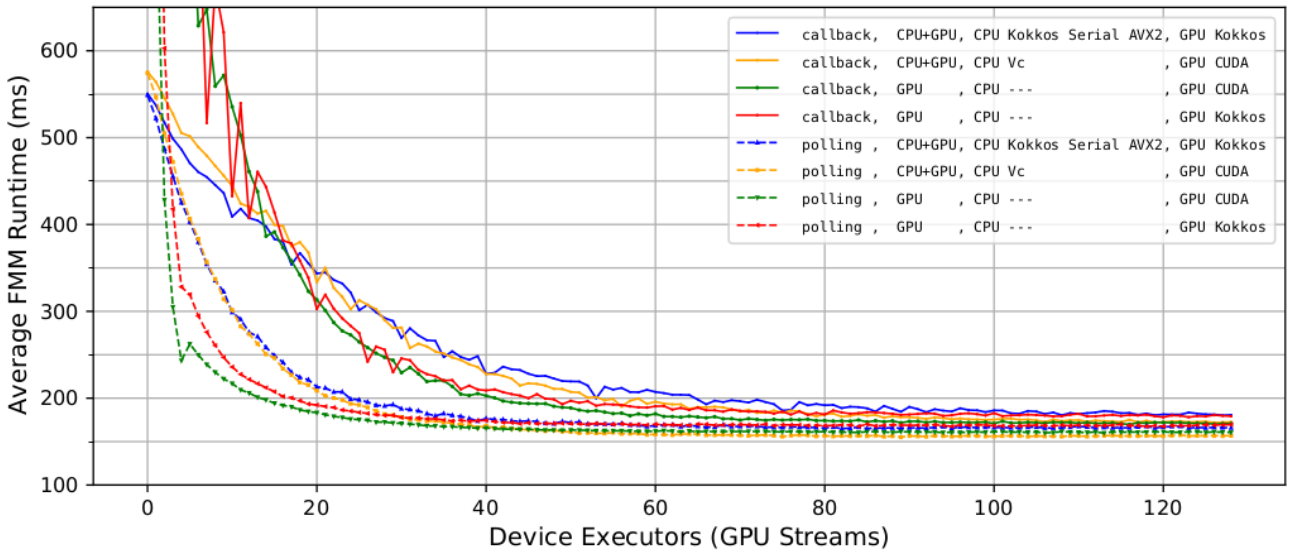


Figure 4. Timings of FMM kernel execution for different configurations using event polling or CUDA callbacks, combined CPU/GPU or GPU only execution, and different CPU and GPU programming models/frameworks in each case.

Reference

- Daiß, Gregor, et al. "Beyond Fork-Join: Integration of Performance Portable Kokkos Kernels with HPX." 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE, 2021.

Distributed scaling

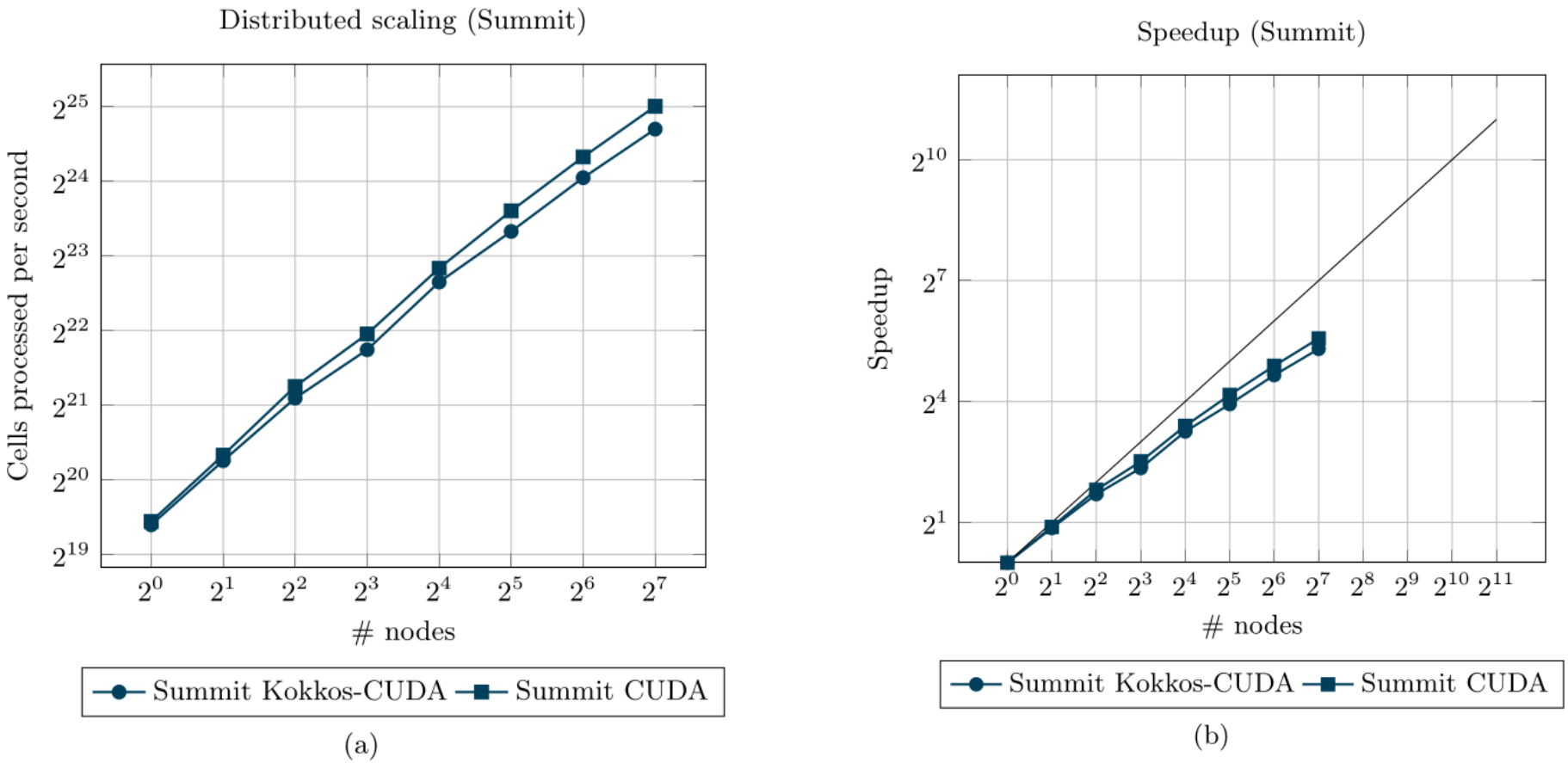


Fig. 1. Cells processed per second (a) and speedup (b). On Piz Daint (blue line) we were able to use 4, 8, 16, 32, 64, 128, 256, 512, 1024, 1400, 1600, 1800, and 2000 nodes. On Summit (violet line) we used 1, 2, 4, 8, 16, 32, 64, and 128 nodes. The speedup was obtained with respect to the smallest amount of nodes the scenario (18 Million cells) fitted on. Note that for the runs with and without APEX a different time on the smallest nodes were used.

Performance profiling

Overhead measurements

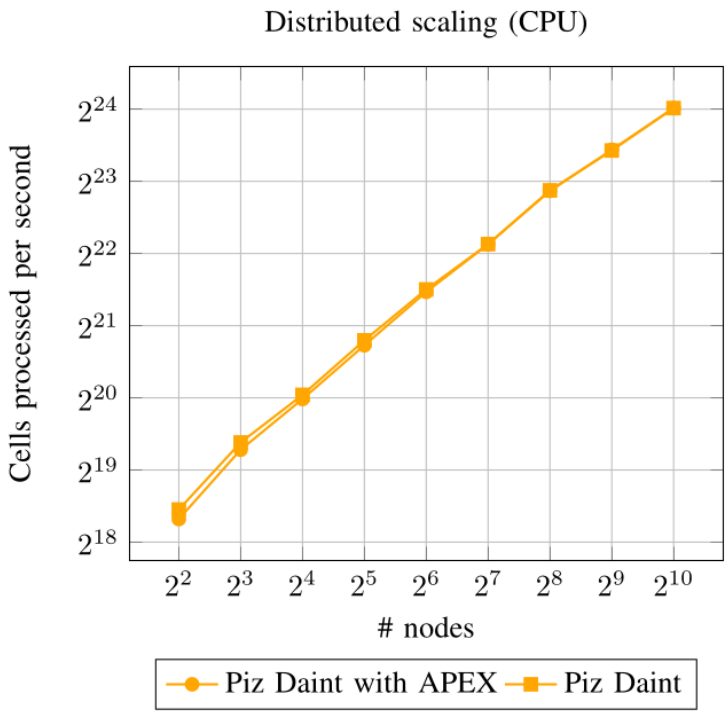


Fig. 2. Cells processed per second on Piz Daint we were able to use 4, 8, 16, 32, 64, 128, 256, 512, 1024, 1400, 1600, 1800, and 2000 nodes. For these runs, we executed the same scenario as in Figure 1a without GPUs. For each amount of nodes, a run without APEX and with APEX pure CPU profiling was done. Since the overhead here is around, 1percent it indicates that the most overhead is introduced by the CUDA™ measurements using CUPTI.

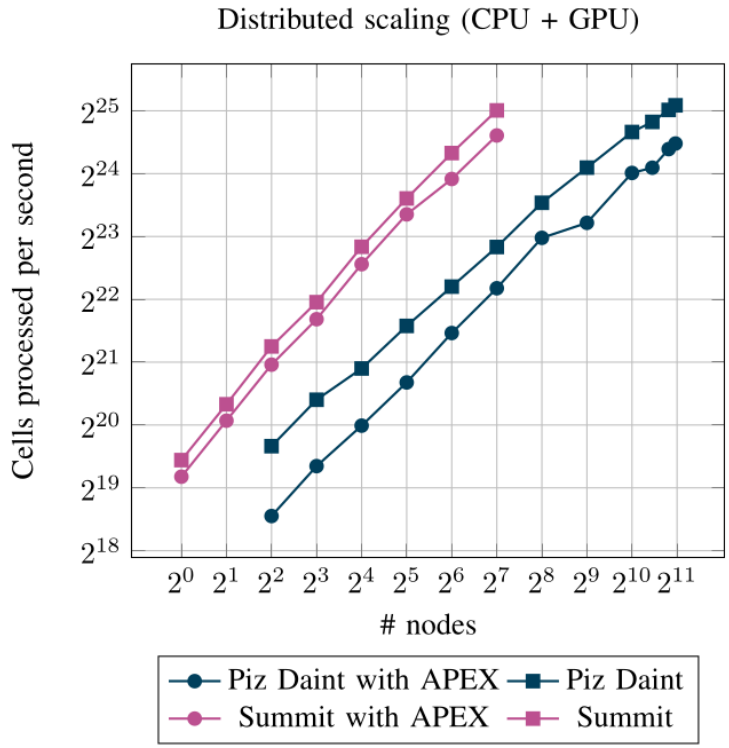
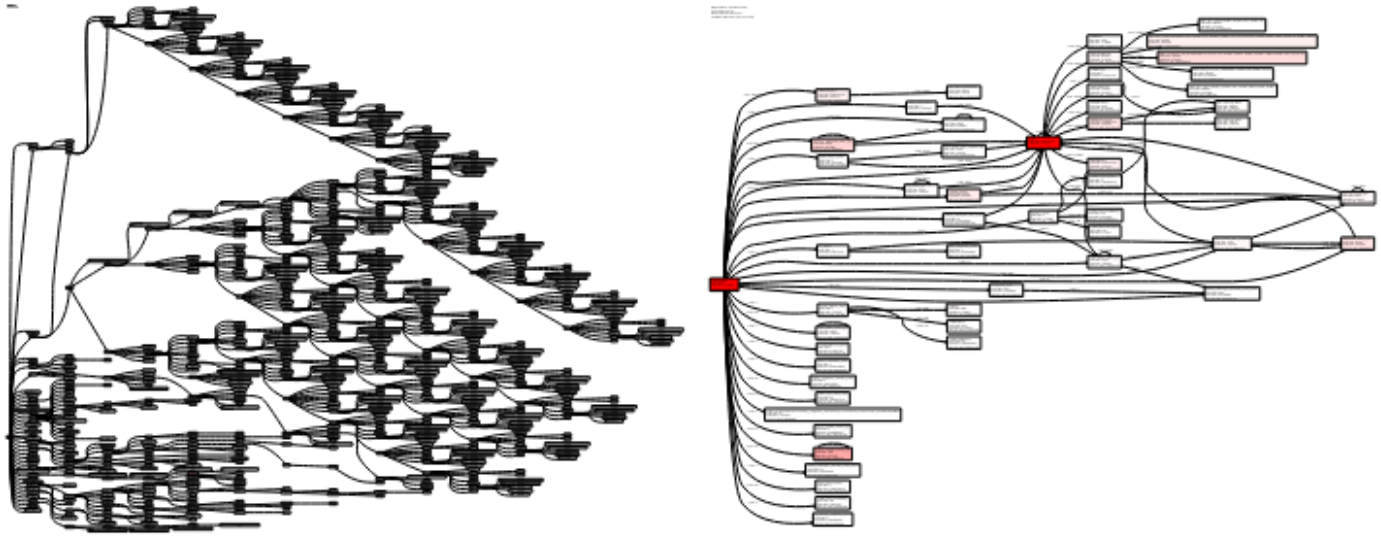


Fig. 1. Cells processed per second (a) and speedup (b). On Piz Daint (blue line) we were able to use 4, 8, 16, 32, 64, 128, 256, 512, 1024, 1400, 1600, 1800, and 2000 nodes. On Summit (violet line) we used 1, 2, 4, 8, 16, 32, 64, and 128 nodes. The speedup was obtained with respect to the smallest amount of nodes the scenario (18 Million cells) fitted on. Note that for the runs with and without APEX a different time on the smallest nodes were used.

Task trees and task graphs



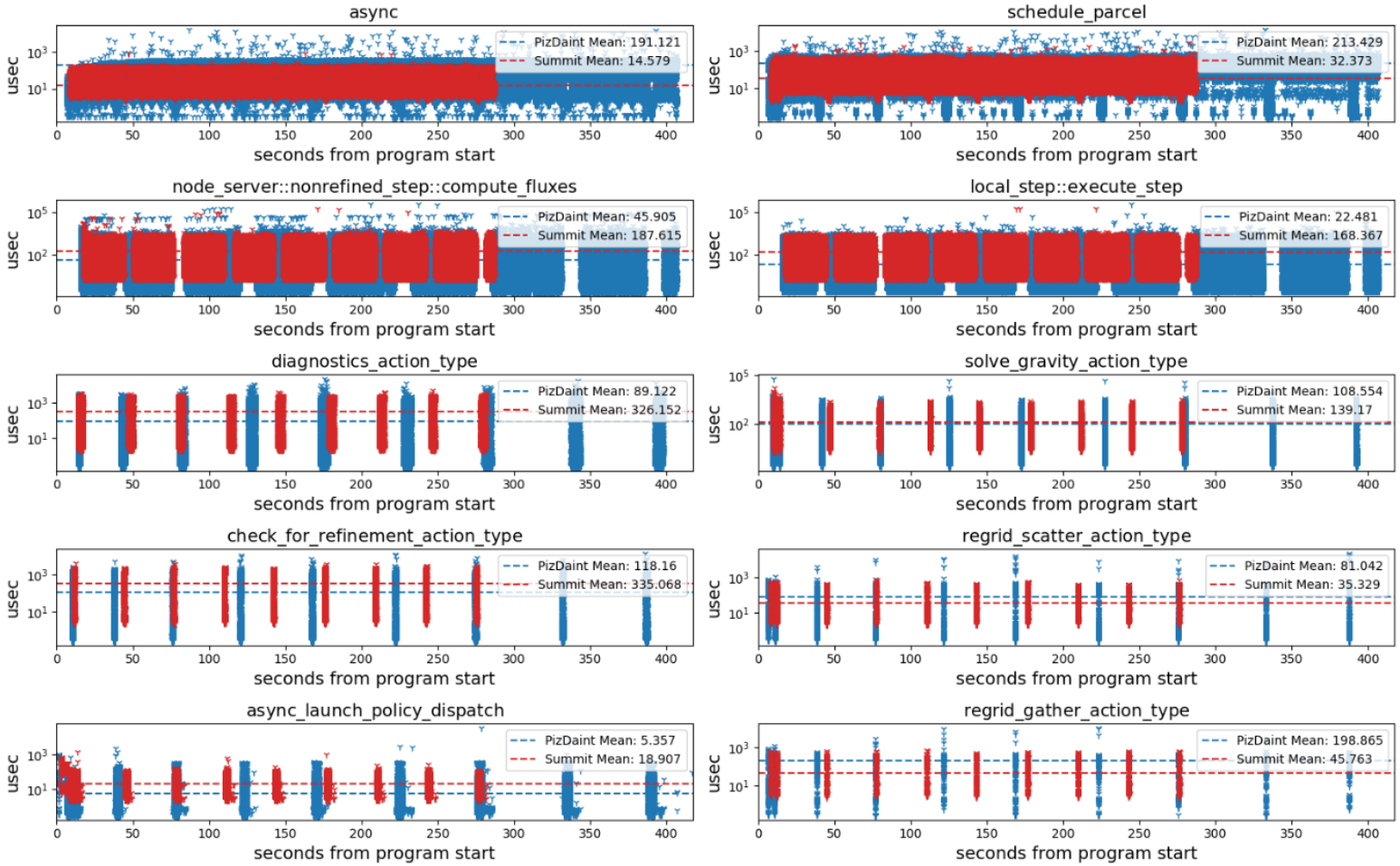
(a) Task tree example. (b) Task graph example.

Figure 1: Task tree and task graph of Octo-Tiger as captured by APEX. Intensity of red color is correlated with the node's contribution to the overall runtime. The recursive structure of the octree is evident in the expanded tree. High resolution images are available here (https://doi.org/10.6084/m9.figshare.14666184.v1).

Reference

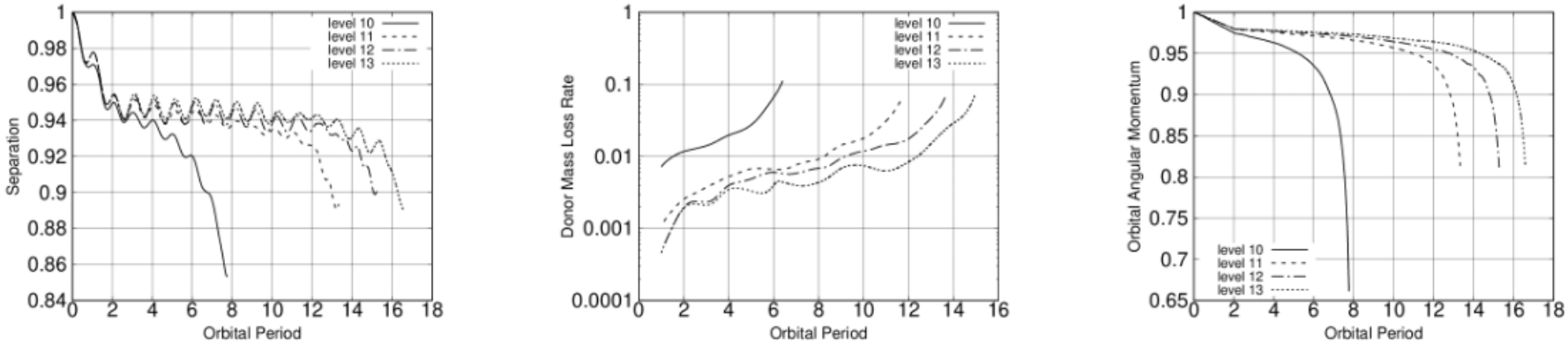
- Diehl, Patrick, et al. "Octo-Tiger's New Hydro Module and Performance Using HPX+ CUDA on ORNL's Summit." arXiv:2107.10987 (2021). (Accepted IEEE Cluster 21)

Sampled profile of tasks on Piz Daint and Summit



Astrophysic validation

Resolution convergence: Double white dwarf merger



(a) (b) (c)

Figure 4: The separation between the stars' centers of mass is depicted in (a), normalized to the initial separation. The donor mass loss rate is shown in (b), normalized to one donor mass per initial orbital period. The orbital angular momentum is shown in (c), normalized to the initial orbital angular momentum. The time coordinate for all three plots is shown in units of the initial orbital period.

Reference

- Diehl, Patrick, et al. "Performance Measurements Within Asynchronous Task-Based Runtime Systems: A Double White Dwarf Merger as an Application." Computing in Science & Engineering 23.3 (2021): 73-81.

Higher reconstruction in the hydro module

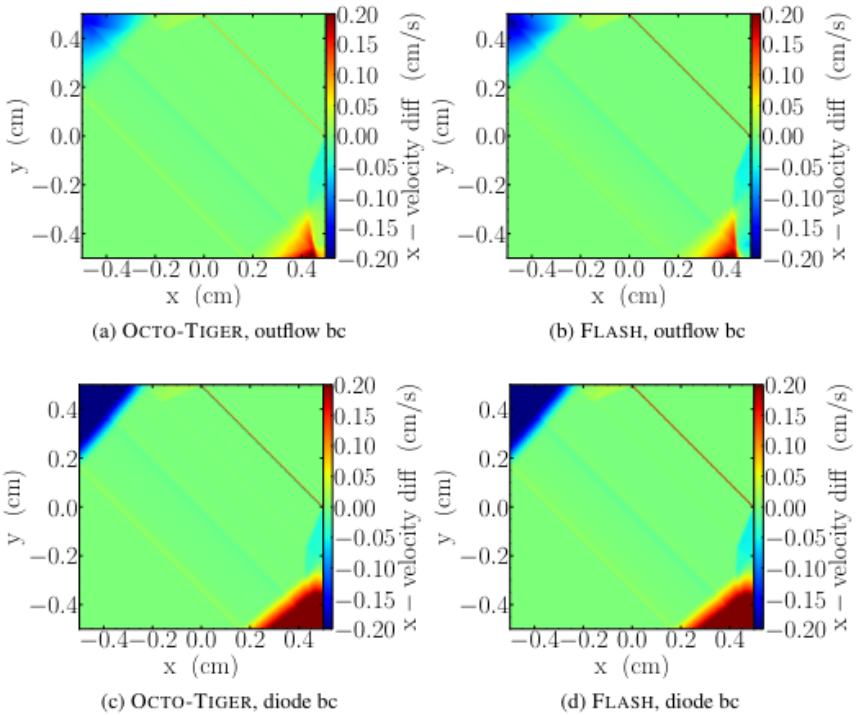
Table 5: The average error in the density field for the rotating star test using the old and new hydro modules. In these units, the central density of the star is 1.

Refinement Level	Opening Criterion	Old	New
6	0.5	2.41×10^{-3}	1.45×10^{-3}
6	0.35	5.22×10^{-4}	3.59×10^{-4}
7	0.5	2.52×10^{-3}	1.51×10^{-3}
7	0.35	4.49×10^{-4}	2.78×10^{-4}

Reference

- Diehl, Patrick, et al. "Octo-Tiger's New Hydro Module and Performance Using HPX+ CUDA on ORNL's Summit." arXiv:2107.10987 (2021). (Accepted IEEE Cluster 21)

Comparison with Flash I



(a) OCTO-TIGER, outflow bc

(b) FLASH, outflow bc

(c) OCTO-TIGER, diode bc

(d) FLASH, diode bc

Figure 2. OCTO-TIGER vs. FLASH for a resolution of 256^3 and an angle of 45 deg for the shock tube. Difference in the x-velocities between simulations and analytic solution at the end of the simulation, time $t = 0.2$. The boundary condition in the top row is outflow without material inflow (diode), while in the lower row it is an outflow condition that allows material to inflow back to the simulation domain (outflow)

Reference

- Marcello, Dominic C., et al. "octo-tiger: a new, 3D hydrodynamic code for stellar mergers that uses hpx parallelization." Monthly Notices of the Royal Astronomical Society 504.4 (2021): 5345-5382.

Comparison with Flash II

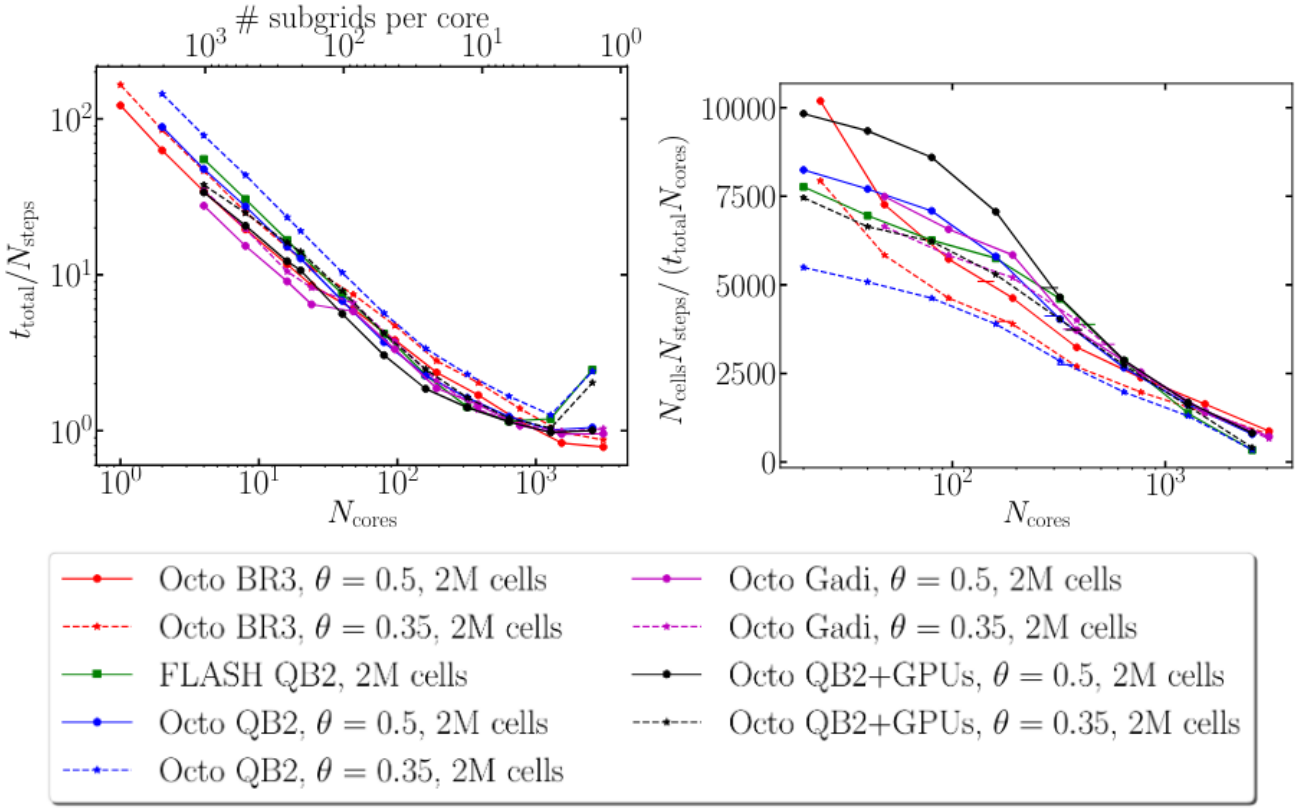


Figure 22. Scaling test carried out on the pulsating polytrope problem of a small size. QB2 refers to QueenBee2, BR3 refers to the BigRed3. The simulations contain $128^3 \approx 2M$ cells or $16^3 = 4096$ subgrids. The short horizontal segments mark the 0.5 efficiency

Reference

- Marcello, Dominic C., et al. "octo-tiger: a new, 3D hydrodynamic code for stellar mergers that uses hpx parallelization." Monthly Notices of the Royal Astronomical Society 504.4 (2021): 5345-5382.

Conlusion and Outlook

Conclusion and Outlook

Conclusion

- Integration of CUDA GPUs within HPX/Kokkos
  $\rightarrow$ AMD still on development
  $\rightarrow$ ISC paper in preparation

Outlook

- Scaling results with the new Kokkos/HPX implementation
- Optimizing and scaling result with the AMD GPUs
- Benchmark the radiation and port to GPU $\rightarrow$ most compute intense kernel

Thanks for your attention! Questions?