Distributed, combined CPU and GPU profiling within HPX using APEX

Patrick Diehl*§, Gregor Daiß†, Kevin Huck‡, Dominic Marcello* , Sagiv Shiber§, Hartmut Kaiser*,
Juhan Frank§, Geoffrey C. Clayton§, and Dirk Pflüger†
*LSU Center for Computation & Technology, Louisiana State University, Baton Rouge, LA, 70803 U.S.A
Email: patrickdiehl@lsu.edu
† IPVS, University of Stuttgart, Stuttgart, 70174 Stuttgart, Germany
‡ *OACISS*, University of Oregon, Eugene, OR, U.S.A.
§ Department of Physics and Astronomy, Louisiana State University, Baton Rouge, LA, 70803 U.S.A.

***Abstract*—Benchmarking and comparing performance of a scientific simulation across hardware platforms is a complex task. When the simulation in question is constructed with an asynchronous, many-task (AMT) runtime offloading work to GPUs, the task becomes even more complex. In this paper, we discuss the use of a uniquely suited performance measurement library, *APEX*, to capture the performance behavior of a simulation built on *HPX*, a highly scalable, distributed AMT runtime. We examine the performance of the astrophysics simulation carried-out by *Octo-Tiger* on two different supercomputing architectures. We analyze the results of scaling and measurement overheads. In addition, we look in-depth at two similarly configured executions on the two systems to study how architectural differences affect performance and identify opportunities for optimization. As one such opportunity, we optimize the communication for the hydro solver and investigated its performance impact.**

***Index Terms*—CUDA™, HPX, Performance Measurements**

I. Introduction

Whether in CPU code, in GPU kernels, or in the inter-node communication – performance bottlenecks in High-Performance Computing (HPC) applications may be hidden in any part of the program. There have been many attempts to ease the development of HPC applications and to indicate and to avoid as many bottlenecks as possible. One example is the asynchronous many-task (AMT) system HPX [1], which aims to solve some of the more common problems. It makes it easier to overlap communication and computation, to employ both CPUs and GPUs, and to avoid overhead for fine-grained parallelism using lightweight threading.

Even with HPX, of course, it is still perfectly possible to introduce performance bottlenecks into one's application. Being able to collect performance measurements to profile an HPX application remains important. For AMT systems such as HPX, it is beneficial to have a profiling tool that understands the task-based nature of the runtime system – for example, call stacks themselves are less useful: A system thread may jump back and forth between various HPX tasks as they are yielded and resumed, and the call stack itself may be dozens of levels of runtime functions that have no particular interest to the application developer.

Collecting these measurements in a profiling run is challenging, as one not only needs a profiling framework that supports both CPU and GPU code as well as the distributed collection of profiling data across many compute nodes. One also needs to keep any overheads introduced by the profiling itself to an absolute minimum. Otherwise, it would not only distort the collected measurements, but also make large, distributed runs infeasible, rendering us unable to detect potential performance bottlenecks that only appear at scale.

HPX is integrated with a performance measurement library, APEX (Automatic Performance for Exascale), which was designed specifically for the HPX runtime and the above requirements. In previous work, APEX was used together with the HPX performance counters to collect performance measurements for Octo-Tiger [2], an astrophysics application which is built upon HPX and contains optimized kernels for both CPUs and GPUs [3], [4]. Octo-Tiger is capable of running the same kernels on the CPUs and GPUs simultaneously (on different data) depending on the load. In that work, profiling data of Octo-Tiger was gathered with APEX in distributed CPU-only runs where the energy usage, the idle rate, and overhead of the HPX AGAS (Active Global Address Space) was analyzed [5]. Furthermore, combined CPU-GPU profiling runs have been performed on Summit, analyzing the performance behavior of Octo-Tiger's new CUDA™ hydro module in different configurations for simple benchmark scenarios [6].

All these previous efforts inspire this new work, collecting performance measurements on both CPU and GPU during a full-scale production-scenario run on Piz Daint (Cray™ XC50 with one 12-core Intel® Xeon® E5-2690 + one NVIDIA® Tesla® P100 per node) [7] and Summit (IBM® AC922 with two 22-core Power9™ + six NVIDIA® Tesla® V100 per node) [8].

The purpose of this work is thus twofold: First, we collect data that we can actually use to improve Octo-Tiger by identifying potential bottlenecks. To do so, we collect measurements running the production scenario for 40 time-steps both on Summit and Piz Daint, using 48 compute nodes on Piz Daint and 8 compute nodes on Summit (resulting in 48 GPUs in either case). With those measurements, we can investigate the specific parts of Octo-Tiger on two distinct architectures in a distributed CPU/GPU run, providing insights into the different runtime behavior of Octo-Tiger regarding GPU-performance, CPU performance, and communication. For example, the communication seems to have a larger overhead on Piz Daint.

Second, we are showcasing the feasibility of APEX for large-scale runs, collecting combined CPU and GPU performance measurements, showing that the overhead introduced by the profiling itself is small enough to handle large production-scale scenarios. To this end, we are running the scenario both with and without APEX profiling enabled for a scaling run on each machine, to see both the overhead on a few compute nodes, and the runtime behavior when scaling to more nodes (with up to 2000 compute nodes on Piz Daint). Furthermore, we repeat these overhead measurements on Piz Daint for a CPU-only run to determine the performance impact of the NVIDIA® CUDA™ Profiling Tools Interface (CUPTI) which is used to collect the GPU performance data.

To highlight the need for low profiling overhead, we can look at the short runtimes for each time-step of Octo-Tiger: During the test runs on Summit, we gather 5 GB of data during all eight runs. For each run 40 time-steps were executed. Each time-step takes about 0.72s on 128 Summit nodes and consists of 6 iterations of the gravity solver, 3 iterations of the hydro solver, and all required communication. On Piz Daint we collected 55 GB of data in total. Here, each time step on 2000 nodes took 0.79s. As time steps are serial in nature, these iterations are our smallest parallel unit. As the time-steps only run for a few hundred milliseconds overall, overheads introduced by the profiling can be very noticeable even if they only take a few milliseconds in total as well.

The remainder of this work is structured as follows: In Section II, we take a brief look at profiling solutions in other AMT frameworks. We then introduce the scientific scenario which we are simulating with Octo-Tiger in Section III. This is the scenario we also used to collect the profiling data by running it for 40 timesteps with APEX enabled. Section IV in turn introduces Octo-Tiger itself, as well as the utilized software stack. In Section V, we show and discuss the collection of the performance measurements for Octo-Tiger. In Section VI, we test communication optimization and analyze the performance improvement. Finally, we conclude the paper in Section VII.

II. Related work

For the related work, we focus on AMTs with distributed capabilities which are: Legion [9], Charm++ [10], Chapel [11], and UPC++ [12]. For a more detailed review, we refer to [13]. Legion [9] provides `Legion Prof` for combined CPU and GPU profiling which is compiled into all builds. Enabling the profiler produces log files which can be viewed using the profiler. Charm++ [10] provides `Charm debug` [14] and the `Projections` framework [15] for performance analysis and visualization. Chapel [11] provides `ChplBlamer` [16] for profiling. UPC++ seems not to have some dedicated tool for profiling, and any profiler supporting C++ is recommended in their documentation.

Like HPX, nearly all of these runtimes provide a specialized tool that has been designed to deal with the particular challenges of AMTs in general, and the needs of the runtime system in particular. APEX is a specialized tool in the case of HPX, and provides similar measurement and analysis abilities of the above tools, including flat profiling, tracing, sampling, taskgraphs/trees, and concurrency graphs. In addition, APEX provides support for several programming models/abstractions with or without HPX, including CUDA™, HIP, OpenMP, OpenACC, Kokkos, POSIX threads, and C++ threads. APEX does not provide analysis tools directly, but rather uses commonly accepted formats and targets both HPC performance analysis tools (ParaProf [17], Vampir [18], Perfetto [19]) and standard data analysis tools (Python, Graphviz [20]).

III. Scientific application

Stellar mergers are mysterious phenomena that pack a broad range of physical processes into a small volume and a fleeting time duration. With the proliferation of deep wide-field, time-domain surveys, we have been catching on camera a vastly increased number of outbursts, many of which have been interpreted as stellar mergers. The best case, so far, of an observed merger is V1309 Sco, a contact binary identified using a recent survey database, the Optical Gravitational Lensing Experiment [21]. Fortunately, not only the merger itself was observed, but archival data from other observing programs enabled the reconstruction of the light curve years before the merger. During the merger itself, the system brightness increased by 4 magnitudes, with a peak luminosity in the red visible light [22]. This complete record of observations has led to term V1309 Sco the "Rosetta Stone" of mergers. Previous attempts to model this merger included semi-analytical calculations [23] and hydrodynamic simulations (e.g., [24]). However, the hydrodynamic simulations fail to adequately resolve the atmosphere, the rapid transition between the optically thick merger fluid and the optically thin, nearly empty space, surroundings of the simulated stellar material. To overcome this barrier, computational scientists intend to use the adaptive mesh-refinement hydrodynamics code Octo-Tiger. Using Octo-Tiger's dynamic mesh refinement, the simulations are able to resolve the atmosphere at a higher resolution than ever before. Simulation of the V1309 merger in high resolution provide greater insight into the nature of the mass flow and the consequential angular momentum losses. In this paper, we analyze the performance of Octo-Tiger to identify potential bottlenecks in the combined CPU and GPU long-term production runs, where the atmosphere is maximally resolved. Analyzing the performance is essential at this stage since this model will serve as the necessary baseline for extending Octo-Tiger to include radiation transport to the V1309 model, as well as to other binary merger models.

IV. Software framework

A. C++ standard library for parallelism and concurrency

HPX is the C++ standard library for parallelism and concurrency [1] and one of the distributed asynchronous many-task runtime systems) AMT. Other notable AMTs with distributed capabilities are: Uintah [25], Chapel [11], Charm++ [10], Legion [9], and PaRSEC [26]. However, according to [13] HPX is one with a higher technology readiness level. HPX's API is fully conforming with the recent C++ standard [27]–[29] which is the major difference to the other mentioned AMTs. For more details about HPX, we refer to [1], [30]–[32]. In this work, HPX has two purposes: *1)* the coordination of the synchronous execution of a multitude of heterogeneous tasks (both on CPUs and GPUs), thus managing local and distributed parallelism while observing all necessary data dependencies and *2)* as the parallelization infrastructure for executing CUDA™-kernels on the GPUs via the asynchronous HPX backend.

B. APEX

APEX [33] is a performance measurement library for distributed, asynchronous multitasking systems. It provides lightweight measurements without perturbing high concurrency through synchronous and asynchronous interfaces. 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. The synchronous APEX instrumentation application programming interface (API) can be used to add instrumentation to a runtime, library or application, and includes support for timers and counters. For measuring kernels on NVIDIA® accelerated platforms, APEX is integrated with the NVIDIA® CUDA™ Profiling Tools Interface [34] that provides CUDA™ host callback and device activity measurements. Similarly, on AMD accelerated platforms, APEX is integrated with the Roctracer library [35] providing HIP host callback and device activity measurements. In addition to timer measurements, the hardware and operating system are monitored through an asynchronous measurement that involves the periodic or on-demand interrogation of the operating system, hardware states, or runtime states (e.g., CPU use, resident set size, memory "high water mark"). The NVIDIA® Management Library interface [36] provides periodic CUDA™ device monitoring to APEX, and the ROCm SMI API provides periodic HIP device monitoring. APEX has been extended to capture additional timers and counters related to CUDA™ device-to-device memory transfers, as well as tracking memory consumption on both device and host when requested with the CUDAMalloc∗ /cudaFree∗ or hipMalloc∗/hipFree∗ API calls [37].

APEX supports tracing in both the OTF2 and Google Trace Events formats, but for comparing results between platforms, profile data can be easier to work with. To complement the profile data which collapses the time axis, APEX also captures task and counter scatter plot data, indicating on the x axis when the task started or the counter was captured, and the y axis contains the duration of the task or the value of the counter. The tasks are sampled using a user-specified fraction (default 1%) whereas the counters are sampled at every value capture. This data collection allows the application developer to capture a time sequence of data without the file system overhead of a full event trace.

C. Octo-Tiger

Octo-Tiger is a 3D adaptive mesh refinement (AMR) hydrodynamics finite volume code with Newtonian gravity designed specifically for the study of interacting stellar binaries [38], [2]. The AMR grid rotates with the initial orbital frequency of the binary, which reduces inaccuracies due to numerical viscosity. The gravitational potential and force are computed using a modified version of the Fast Multipole Method that eliminates the gravitational field as a source of angular momentum conservation violation [39]. This enables Octo-Tiger to conserve energy and linear momentum to machine precision in the rotating frame. The astrophysical fluid is modeled using the inviscid Euler equations, which are solved with a finite volume central scheme [40]. The computational domain is based on a properly nested three-dimensional octree structure. Each node in the structure is an N^3 Cartesian sub-grid (usually, $N = 8$), and may be further refined into eight child nodes, each containing its own N^3 sub-grid with twice the resolution of the parent.

In [2], an improved hydro solver for Octo-Tiger that includes a full three-dimensional reconstruction technique has been introduced. The performance of this improved hydro solver (after porting it to GPUs) on Summit's ORNL has been tested as well as its accuracy [6], showing a good GPU speedup (the exact amount depending on the chosen sub-grid size), and a greater accuracy in maintaining a rotating star in equilibrium than the old hydro solver. It has been fully benchmarked demonstrating superior angular momentum conservation and extreme scalability properties allowing it to compute larger problems in a shorter wall-clock time [2]. A convergence study in a real production run of a white dwarf merger has been also performed [5].

Octo-Tiger's CUDA™ implementation is worth elaborating on in a bit more detail here to understand the later performance results in Section V, in particular the short runtimes of the GPU kernels. Usually, each of the compute kernels in the solver (both hydro solver and gravity solver) operates on a single sub-grid at a time with one CPU core. Multi-core usage is achieved by launching many of those methods on different sub-grids concurrently as HPX tasks.

In a GPU run, an issue arises where a single sub-grid does not provide enough work to utilize the entire GPU. However, the scheduler can simply launch multiple kernels on different sub-grids concurrently. They are launched in different CUDA™ streams, which are drawn from a pre-allocated pool to avoid the overhead of stream creation. Usually, we use 128 streams per GPU. As managing multiple CUDA™ streams in each of the CPU threads might become unwieldy, the code uses an HPX-CUDA™ integration. With it, HPX futures can be obtained from CUDA™ kernel launches, allowing HPX to treat them as any other HPX task. Using this strategy, HPX can simply launch a CUDA™ kernel, define subsequent post-processing tasks to be run once it is done, and then suspend the current HPX task that starts the kernel. The current CPU thread can then work on another task, potentially launching another GPU kernel in a different stream.

This implementation scheme has the implication that HPX runs many GPU kernels and CPU tasks concurrently, allowing the application to interleave GPU kernels, CPU tasks, CPU-GPU memory transfers, and inter-node communication seamlessly. Each of the tasks, both on CPU and GPU, has a rather short runtime as a result of only dealing with a single sub-grid at a time. For example, the individual GPU kernel execution time ranges from 100 microseconds (less for smaller auxiliary kernels) to about 2 milliseconds.

V. Performance measurements

Octo-Tiger's dependencies are listed in Table I in the Artifact Description of the appendix. On both systems, we used Octo-Tiger's git hash (*4c38f3bf*) as the baseline. However, we had to do some minor changes with regard to compilation, which do not affect the astrophysics kernels. Therefore, we used a slightly diverged git hash (*b091fd26*) on Piz Daint and git hash (*fd7faf5e*) on Summit. To quantify the overhead $o(n)$ introduced by APEX in percent, we define the metric

$$o(n) = \frac{comp\_apex(n)}{comp\_time\_no\_apex(n)} \cdot 100 - 100\,, \qquad (1)$$

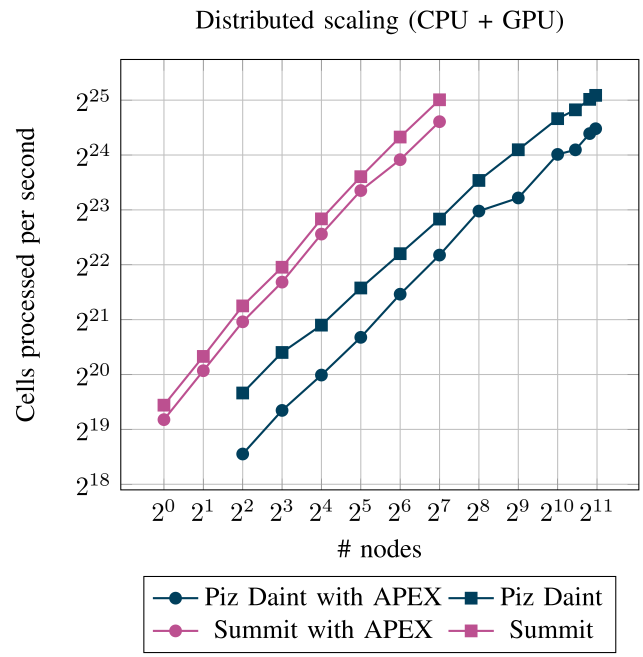
where n is the number of nodes, $comp\_time\_no\_apex(n)$ without APEX, and $comp\_apex(n)$ the computation time with APEX. Note that we only measure the time for the actual computation and not the IO.

A. Distributed scaling

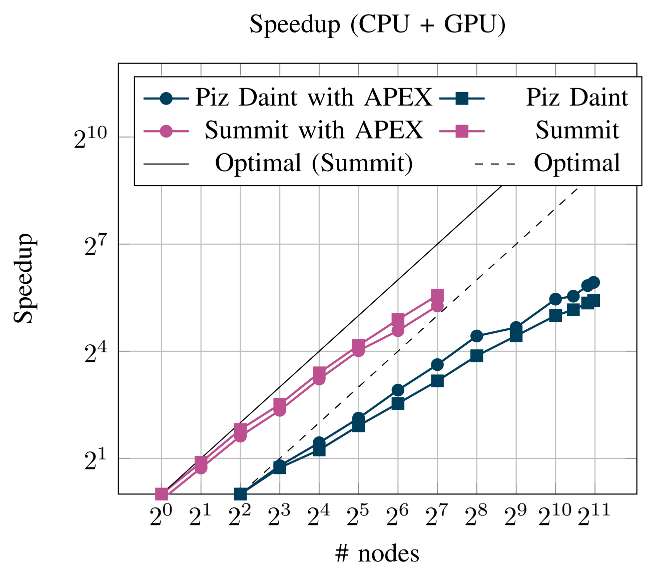
The largest V1309 scenario (18 million cells) fitting on four CSCS's Piz Daint nodes was chosen for this scaling test. Note that the largest scenario for a single node was too small to scale out up to 2000 nodes with 2000 NVIDIA® P100 GPUs and 24 000 CPU cores. However, due to the large amount of memory per node, the same scenario fits on one of ORNL's Summit node. On both machines, we executed the same runs with the APEX + CUDA™ profiling on and off to investigate the overhead introduced by the profiling. Figure 1a shows the cells processed per second for increasing amount of nodes. On Piz Daint the scaling was done using Octo-Tiger's new hydro solver [6] using a three-dimensional reconstruction scheme and scaling results using the old solver for a different scenario are shown here [41]. The blue lines show the scaling on Piz Daint. For both configurations, the scenario scales up to 2000 nodes. However, for 1400 Piz Daint nodes the problem size gets too small, and the scaling starts to flatten out. The overhead $o(n)$ in Equation 1 for Piz Daint is shown in Figure 2a. The overhead is the largest on 4 nodes and seems to decline with increasing node counts.

The violet lines show the scaling on Summit. Here, again, the code scales well up to 128 nodes (using 768 NVIDIA® V100 GPUs overall) and the work is not sufficient for more nodes. The overhead $o(n)$ in Equation (1) on Summit is shown in Figure 2b. The overhead is less prominent on Summit.

We have seen an introduced overhead by the combined profiler on both systems. In a previous study using a different

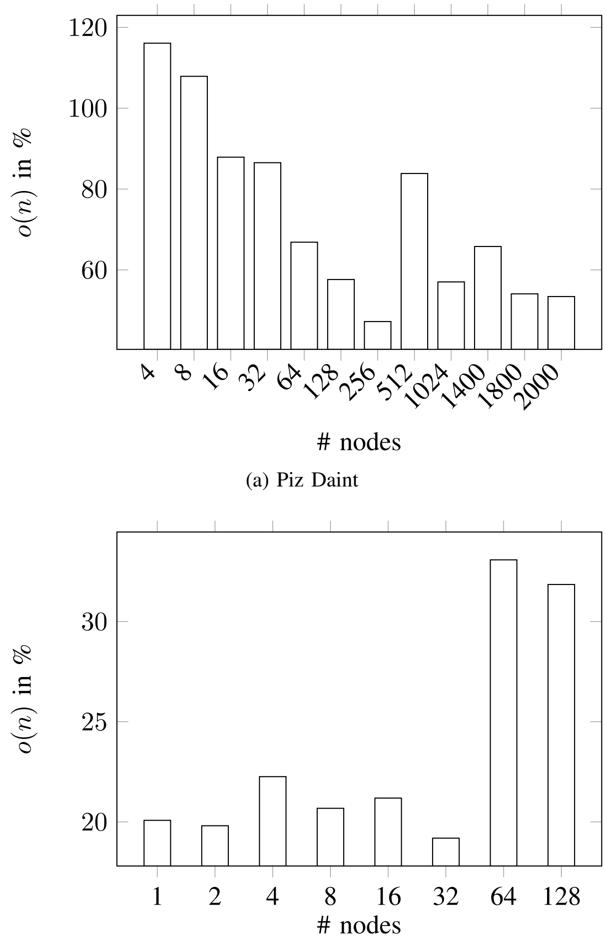


(a)



(b)

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.



(a) Piz Daint

(b) Summit

Fig. 2. Overhead $o(n)$ in Equation (1) for the runs on Piz Daint and Summit, respectively.

problem and the old version of the hydro module, the overhead introduced by pure APEX CPU profiling within HPX was around 1% [5]. To verify if this still holds with the new hydro module and the V1309 scenario, we ran the V1309 scenario on Piz Daint with pure APEX CPU profiling. First, we observed scaling for the CPU kernels up to 2000 Piz Daint nodes. Second, we observed see that the difference is again around 1% as in the previous study. It seems that the overhead is mostly introduced by enabling CUPTI measurement in APEX for both systems. However, it seems that the overhead for smaller node counts on Piz Daint is larger, which needs to be investigated. Regardless, CUPTI provides varying levels of detail/support, and APEX should be refactored to enable the minimum amount of useful support by default, and allow the user to request additional details as needed.

B. Profiling of Octo-Tiger

1) Setup: Because APEX is directly integrated into the HPX runtime, annotating HPX actions (tasks) is simply a matter of providing an annotation for the task when it is instantiated in the code. Not all actions are annotated (anonymous lambdas launched through hpx::async() calls, for example), but major operations in the Octo-Tiger code and in the HPX runtime are annotated. When HPX is configured and built with APEX support, all annotations are provided to APEX, and APEX times the life cycle of each task. HPX is designed to yield and resume tasks when resources are unavailable (futures, system calls, other dependencies), so APEX also provides the ability to yield (and resume) timing a task when it is not executing. For GPU kernels, the kernel name is obtained using the instruction pointer address and debug information in the executable provided by the compiler.

For the following Octo-Tiger profiling results, we use 48 HPX localities (processes). As we use one HPX locality per GPU, this results in using 48 compute nodes on Piz Daint and 8 compute nodes on Summit (as each node contains 6 GPUs here). This ensures that we use the same number of GPUs on both systems. We are still using the same V1309 scenario as before.

Below is a short description of the tasks and kernels of interest in Octo-Tiger. The main CPU tasks include node_server :: non−refined_step :: compute_fluxes: directs the computation of a single Runge-Kutta substep in the hydro and gravity solvers for a given sub-grid. It directs the GPU to compute hydrodynamic fluxes, corrects these fluxes on coarse-fine boundaries, works with other invocations of the same action to compute the global time-step size, computes hydrodynamic sources, then lastly updates the hydrodynamic variables and directs the GPU to update the gravitational variables; local_step :: execute_step: directs the execution of an entire time-step for a given sub-grid. This involves multiple calls to node_server :: nonrefined_step :: compute_fluxes ; diagnostics_actions_type : performs some measurements on the grid, for example, computing total mass on the grid, total angular momentum, and center of masses of each star (for binaries); solve_gravity_action_type : solves for gravity. This is called when an additional gravity solve is needed (other than what node_server :: nonrefined_step :: compute_fluxes calls), such as after grid refinement; check_for_refinement_action_type flags each sub-grid in need of refinement; regrid_gather_action_type gathers information about the sub-grid structure and determines boundaries for the global decomposition; regrid_scatter_action_type uses the information computed by regrid_gather_action_type to redistribute the sub-grids.

The GPU kernels include cuda multipole interactions kernel: Computes the cell to cell interactions for the gravity solver in refined sub-grids (non-rho is without the angular momentum correction); cuda p2p interactions kernels: Computes the cell to cell interactions for the gravity solver in non-refined sub-grids; Special case: cuda p2m interactions kernel: Computes the cell to cell interactions for the gravity solver in non-refined sub-grids with refined neighbor sub-grids (non-rho is without the angular momentum correction); Special case: multipole root: computes the remaining cell to cell interactions in the root sub-grid; reconstruct cuda kernel reconstructs the evolution variables using the PPM method, see [6]; flux cuda kernel the flux method computes the fluxes

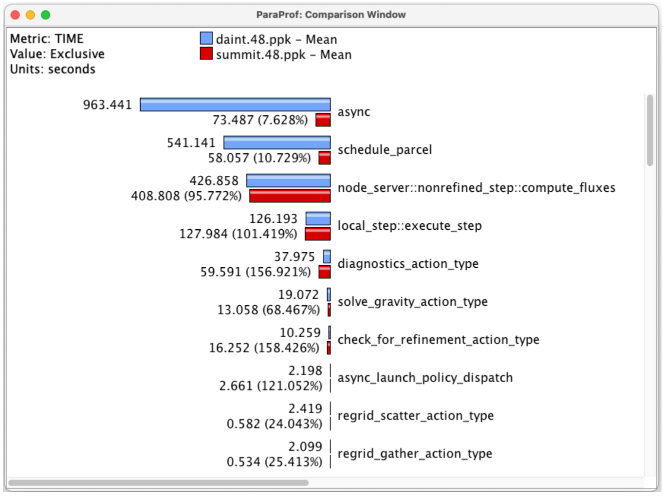


Fig. 3. Comparison of top CPU tasks with TAU ParaProf.

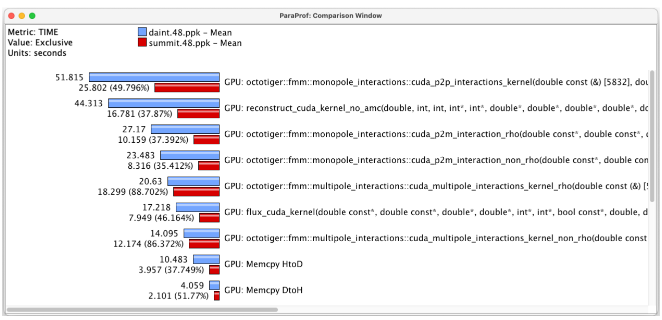


Fig. 4. Comparison of top GPU kernels with TAU ParaProf.

and the Newtonian quadrature to get the final flux.

2) Results / Analysis: In this section, we analyze the collected profiling results and draw conclusions regarding Octo-Tiger's performance on two different architectures. These insights are important for further optimization on the code. Some of them are straightforward and others are quite puzzling. Let us start with the insights on Summit:

1) Figure 3 shows that on Summit, the CPU tasks are not always faster with respect to Piz Daint. However, the overall computational time is lower due to the benefit of the newer GPUs. Looking at the scatterplot data for sampled timers in Figure 5, we see that while the average time *per task* on Summit is slower for some tasks (compute_fluxes, execute_step), the aggregated mean profile is approximately the same.
2) Figure 4 shows that on Summit, the GPU tasks are nearly always faster on NVIDIA® V100 GPUs. This is not surprising, since the GPU kernels are working better on the newer NVIDIA® V100 than on the older NVIDIA® P100. We found two exception, both the cuda_multipole_interactions_kernel_no_rho () and cuda_interactions_kernel__rho () were faster on the NVIDIA® P100. These tasks computes the monopole-multipole gravity interactions in the case that the leave nodes have different refinements. More investigation is needed to determine if it is possible to improve this compute kernel for the NVIDIA® V100.

On Piz Daint we gather the following insights:

Unrelated to the application itself, we found some differences within the HPX run time system. HPX provides the function hpx::async to asynchronously launch functions and lambda functions. Figures 3 and 5 show that this operation's mean was $4.6\times$ more expensive on Piz Daint as on Summit. Here, the HPX main developers need to investigate this behavior. Figure 3 also shows that an another more expensive HPX operation on Piz Daint was schedule_parcel which mean was nearly $5.6\times$ higher as on Summit. A delay in the schedule_parcel potentially happens when some hpx::future is not ready, which schedule_parcel depends on. Our primary focus here is distributed combined CPU and GPU profiling. We anticipate future research concentrating on the interpretation of these findings and major optimization. However, we will show one minor optimization.

VI. Performance Improvements

On Piz Daint the CPU tasks async and schedule_parcle, see Figure 5, took longer as on Summit. We further noticed that this seems to be worse when running scenarios that included the Octo-Tiger hydro solver on Piz Daint when doing more tests. Consequently, we took a look at the boundary communication within the Hydro module, and a way to reduce the overall number of messages. For sub-grids located on the same HPX locality, we now access the memory of these directly (foregoing HPX actions and temporary communication buffers) to fill the ghost-cells. This requires some more overhead within the node_server::collect_hydro_boundaries::set_hydro_boundary method (and other associated methods) itself as we need to make sure that the results of those direct neighbors is up-to-date, which is done with local HPX promises/future pairs. This increases the mean time of said communication methods as they need to handle the promises, but reduces the overall HPX action calls and thus reduces the calls to schedule parcel. We gained a noticeable speedup on Daint (reduced total runtime from 400 to 320 seconds) and will be become helpful for other machines in the future as well. This becomes evident in, Figure 7 compares the run on 48 Piz Daint nodes with the optimization enabled (orange) and the previous run (blue). This does not address the root issue, which seem to be the longer runtimes for async and schedule_parcle on Daint, but it reduces the symptoms considerably and is furthermore a useful optimization for other machines as well.

Unfortunately, the test-bed allocation on Summit ended before we could finish implementing the optimization. Therefore, we could not show results here.

VII. Conclusion

In this work, we have analyzed the overhead of performing combined CPU and GPU performance measurements with

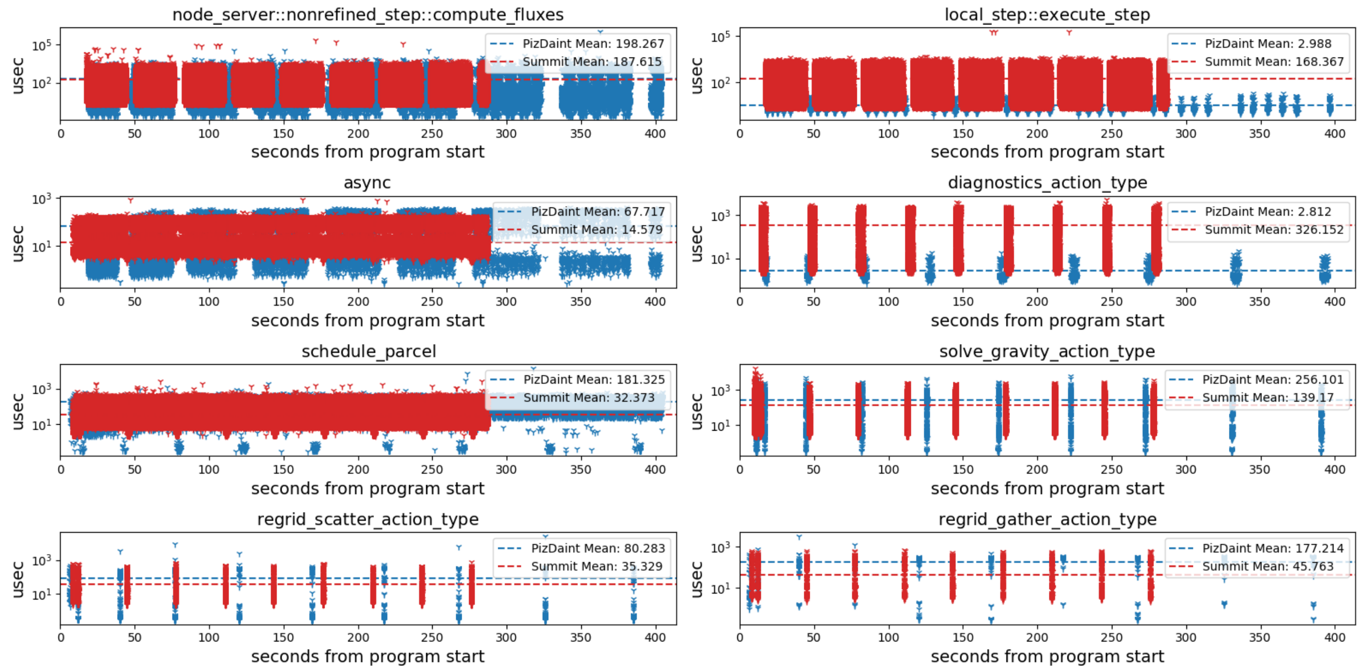


Fig. 5. Comparison of sampled profiles of CPU tasks on 48 localities between Summit (red) and Piz Daint (blue). The overall runtime on Summit is shorter, but interestingly, some CPU tasks took longer (per call) to execute on Summit than on Piz Daint.

APEX in a large-scale HPX application distributed across up to 2000 compute nodes.

We have demonstrated that Octo-Tiger easily scales to that many nodes on Piz Daint with the APEX profiling enabled. Profiling is thus feasible for real-world production-size runs on high-performance systems equipped with GPUs. However, we encountered a noticeable profiling overhead at scale (52.41% on 2000 Piz Daint nodes). This seems to be due to the GPU measurements with CUPTI, as a subsequent CPU-only run exhibited a smaller profiling overhead. Note that the overhead on Piz Daint with very few nodes is about two times higher and requires further investigation. On Summit, there is a noticeable overhead at scale from profiling, too, but significantly less (31.85% on 128 Summit nodes). Overall, regarding the APEX profiling overhead, the distributed profiling with both CPU and GPU measurements works, scales up, and is ready to use; yet more investigation is needed to work on minimizing the overhead of the GPU measurements, if possible.

While we focused on evaluating how suitable APEX is for these large-scale, distributed analyses, there are some interesting results regarding Octo-Tiger itself as well. It is notable that the speedup of the average GPU kernel runtime from a P100 to a newer V100 varies a lot between the kernels. There are many factors that may influence the average kernel runtime difference between the devices: For instance, going to the V100, there is an increase in the number of Streaming Multiprocessors (SM), an increase in L1 cache available per SM, an increase in global memory bandwidth and a slight increase in clock speed. This does not even take into account the fact that we compile for different architectures.

Considering that we use concurrent kernel execution via 128 CUDA™ streams per GPU to achieve device utilization (as outlined in Section IV-C), it is unlikely that the increased numbers of SMs in the V100 has a major impact on the average kernel execution time (as those would rather facilitate more concurrent kernel execution). Thus, it seems more likely that the speedup is due to a combination of the other factors, the exact speedup depending on what is currently limiting the kernel. The larger speedups indicate that the kernels benefit from the larger L1 cache available, however, determining the exact cause is subject of future work, especially as the kernels are currently still undergoing changes. However, the results here give us an idea which kernels need more attention during this process, particularly when targeting older architectures, thus helping us to steer our development focus.

Beyond the GPU results, the profiling uncovered that there are some crucial methods that run significantly slower on Piz Daint than on Summit, e.g. schedule_parcel . We consequently optimized the communication of the hydro solver to alleviate this issue. Using the APEX measurements we could identify the parts of the code which benefited from the optimizations. This shows the need for distributed performance measurements on a production system. Of course, while we focused on the usability of APEX for these kinds of analyses in this work and fixed some of the issues, the uncovered issues still need to be further investigated and addressed. Consequently, we will examine these remaining issue in future work, hopefully further improving the runtime of future simulations with Octo-Tiger.

A radiation module for Octo-Tiger is currently being im-

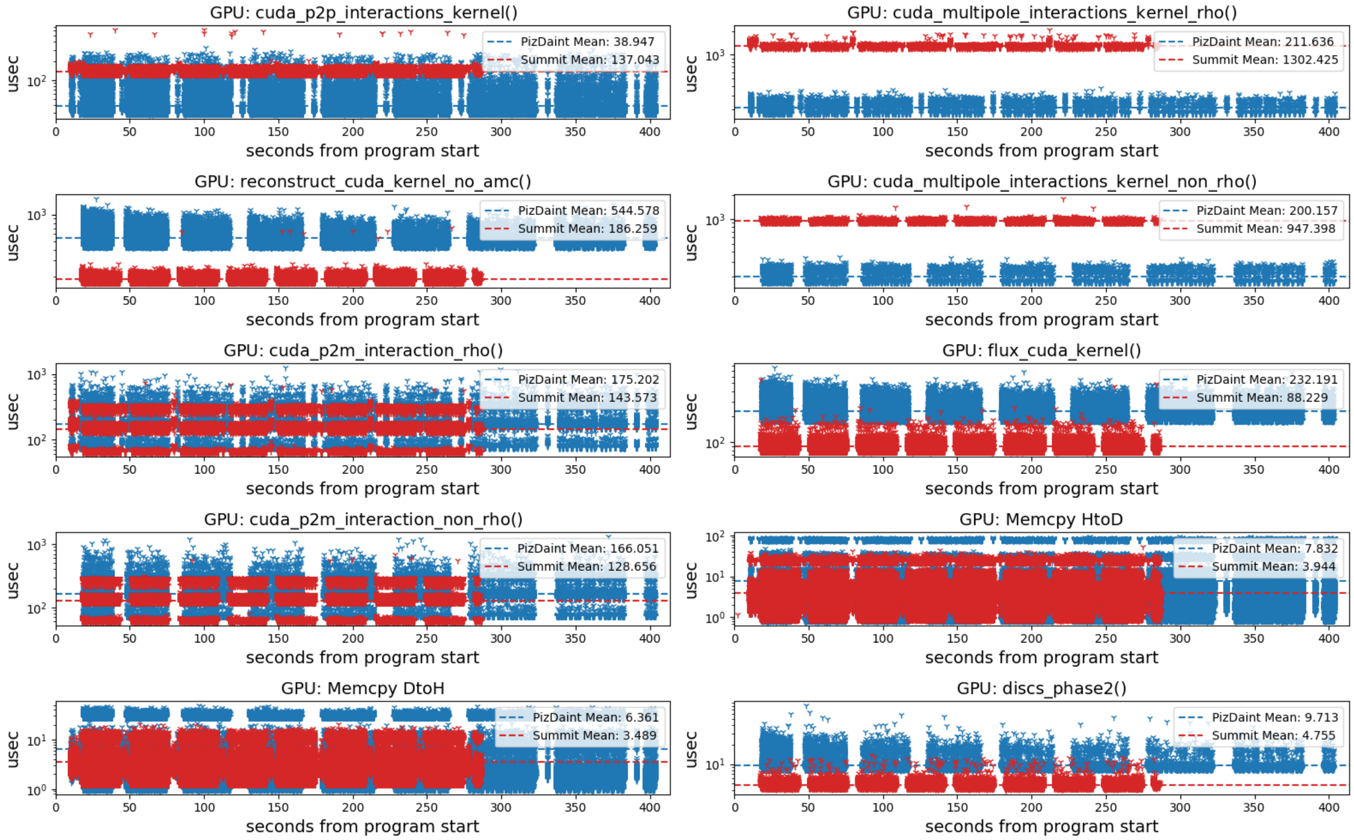


Fig. 6. Comparison of sampled GPU kernels and memory transfers on 48 localities between Summit (red) and Piz Daint (blue). Not surprisingly, in all cases, the performance of the V100s on Summit outperformed that of the P100s on Piz Daint. However, not all kernels saw a significant improvement in performance.

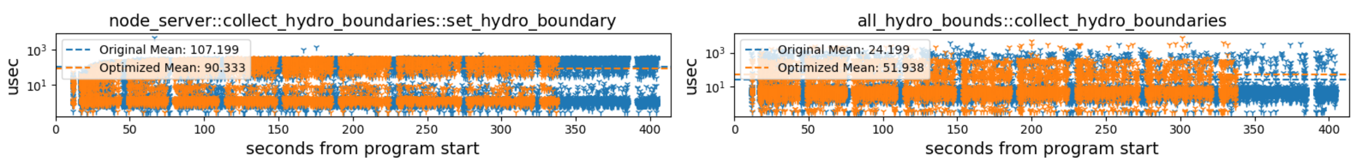


Fig. 7. Comparison of sampled CPU and GPU kernels and 48 localities between the original code (blue) and the optimized code (orange) on Piz Daint.

plemented by its developers, and it is in the testing phase. Our performance analysis of the current modules will be crucial to estimate the performance impact of the new module prior to its inclusion. Including radiation in the simulations of V1309 together with resolving the star atmosphere at a higher resolution than ever before will enable one to self-consistently compute the light curve and directly compare it with the observed one of V1309. If one is able to accurately reproduce the light curve of the "Rosetta Stone of mergers", it will be possible to reliably simulate the outburst light curves of other mergers.

Acknowledgment

This work was supported by a grant from the Swiss National Supercomputing Centre (CSCS) under project ID s1078. This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725. The APEX work was supported by the Scientific Discovery through Advanced Computing (SciDAC) program funded by U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research (ASCR) under contract DE-SC0021299.

VIII. Supplementary Materials

The build scripts are available on GitHub[1] and the input files are available on Zenodo [42].

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

REFERENCES

[1] H. Kaiser *et al.*, "HPX - The C++ Standard Library for Parallelism and Concurrency," *Journal of Open Source Software*, vol. 5, no. 53, p. 2352, 2020.

[2] D. C. Marcello *et al.*, "Octo-Tiger: a new, 3D hydrodynamic code for stellar mergers that uses HPX parallelization," *Monthly Notices of the Royal Astronomical Society*, vol. 504, no. 4, pp. 5345–5382, 2021.

[3] D. Pfander *et al.*, "Accelerating Octo-Tiger: Stellar Mergers on Intel Knights Landing with HPX," in *Proceedings of the International Workshop on OpenCL*, ser. IWOCL '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3204919.3204938

[4] G. Daiß *et al.*, "Beyond fork-join: Integration of performance portable Kokkos kernels with HPX," in *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. IEEE, 2021, pp. 377–386.

[5] P. Diehl *et al.*, "Performance measurements within asynchronous task-based runtime systems: A double white dwarf merger as an application," *Computing in Science & Engineering*, vol. 23, no. 3, pp. 73–81, 2021.

[6] ——, "Octo-Tiger's new hydro module and performance using HPX+CUDA on ORNL's Summit," 2021.

[7] CSCS, "Piz Daint — CSCS," 2021, https://www.cscs.ch/computers/piz-daint/.

[8] Oak Ridge National Laboratory, "Summit - Oak Ridge Leadership Computing Facility," 2021, https://www.olcf.ornl.gov/olcf-resources/compute-systems/summit/.

[9] M. Bauer *et al.*, "Legion: Expressing locality and independence with logical regions," in *SC'12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. IEEE, 2012, pp. 1–11.

[10] L. V. Kale *et al.*, "Charm++ a portable concurrent object oriented system based on c++," in *Proceedings of the eighth annual conference on Object-oriented programming systems, languages, and applications*, 1993, pp. 91–108.

[11] B. L. Chamberlain *et al.*, "Parallel programmability and the chapel language," *The International Journal of High Performance Computing Applications*, vol. 21, no. 3, pp. 291–312, 2007.

[12] J. Bachan *et al.*, "The UPC++ PGAS library for exascale computing," in *Proceedings of the Second Annual PGAS Applications Workshop*, 2017, pp. 1–4.

[13] P. Thoman *et al.*, "A taxonomy of task-based parallel programming technologies for high-performance computing," *The Journal of Supercomputing*, vol. 74, no. 4, pp. 1422–1434, 2018.

[14] R. Jyothi *et al.*, "Debugging support for Charm++," in *PADTAD Workshop for IPDPS 2004*. IEEE Press, 2004, p. 294.

[15] S. Biersdorff *et al.*, "Integrated Performance Views in Charm++: Projections Meets TAU," in *Proceedings of The 38th International Conference on Parallel Processing (ICPP)*, Vienna, Austria, September 2009, pp. 140–147.

[16] H. Zhang *et al.*, "Chplblamer: A data-centric and code-centric combined profiler for multi-locale chapel programs," in *Proceedings of the 2018 International Conference on Supercomputing*, 2018, pp. 252–262.

[17] R. Bell *et al.*, "Paraprof: A portable, extensible, and scalable tool for parallel performance profile analysis," in *European Conference on Parallel Processing*. Springer, 2003, pp. 17–26.

[18] A. Knüpfer *et al.*, "The vampir performance analysis tool-set," in *Tools for high performance computing*. Springer, 2008, pp. 139–155.

[19] Google, "Perfetto – system profiling, app tracing and trace analysis," 2021, https://perfetto.dev.

[20] The Graphviz Authors, "Graphviz," 2021, https://graphviz.org.

[21] A. Udalski *et al.*, "The Optical Gravitational Lensing Experiment," Acta Astron., vol. 42, pp. 253–284, 1992.

[22] R. Tylenda *et al.*, "V1309 Scorpii: merger of a contact binary," *A&A*, vol. 528, p. A114, 2011.

[23] O. Pejcha, "Burying a Binary: Dynamical Mass Loss and a Continuous Optically thick Outflow Explain the Candidate Stellar Merger V1309 Scorpii," ApJ, vol. 788, no. 1, p. 22, Jun. 2014.

[24] J. L. A. Nandez *et al.*, "V1309 Sco—Understanding a Merger," ApJ, vol. 786, no. 1, p. 39, May 2014.

[25] J. D. de St Germain *et al.*, "Uintah: A massively parallel problem solving environment," in *Proceedings the Ninth International Symposium on High-Performance Distributed Computing*. IEEE, 2000, pp. 33–41.

[26] G. Bosilca *et al.*, "Parsec: Exploiting heterogeneity to enhance scalability," *Computing in Science & Engineering*, vol. 15, no. 6, pp. 36–45, 2013.

[27] The C++ Standards Committee, "ISO International Standard ISO/IEC 14882:2014, Programming Language C++," Geneva, Switzerland: International Organization for Standardization (ISO)., Tech. Rep., 2014, http://www.open-std.org/jtc1/sc22/wg21.

[28] ——, "ISO International Standard ISO/IEC 14882:2017, Programming Language C++," Geneva, Switzerland: International Organization for Standardization (ISO)., Tech. Rep., 2017, http://www.open-std.org/jtc1/sc22/wg21.

[29] ——, "ISO International Standard ISO/IEC 14882:2020, Programming Language C++," Geneva, Switzerland: International Organization for Standardization (ISO)., Tech. Rep., 2020.

[30] H. Kaiser *et al.*, "HPX: A Task Based Programming Model in a Global Address Space," in *Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models*, ser. PGAS '14. ACM, 2014, pp. 6:1–6:11.

[31] ——, "Higher-level parallelization for local and distributed asynchronous task-based programming," in *Proceedings of the First International Workshop on Extreme Scale Programming Models and Middleware*, ser. ESPM '15. New York, NY, USA: ACM, 2015, pp. 29–37.

[32] T. Heller *et al.*, "Closing the Performance Gap with Modern C++," in *High Performance Computing*, ser. Lecture Notes in Computer Science, M. Taufer *et al.*, Eds., vol. 9945. Springer International Publishing, 2016, pp. 18–31.

[33] K. A. Huck *et al.*, "An autonomic performance environment for exascale," *Supercomputing frontiers and innovations*, vol. 2, no. 3, pp. 49–66, 2015.

[34] NVIDIA, "CUDA Profiling Tools Interface," 2021, https://docs.nvidia.com/cuda/cupti/index.html.

[35] AMD, "AMD ROCTracer API," 2021, https://rocmdocs.amd.com/en/latest/ROCm_Tools/ROCTracer-API.html.

[36] NVIDIA, "NVIDIA Management Library (NVML)," 2020, https://developer.nvidia.com/nvidia-management-library-nvml.

[37] W. Wei *et al.*, "Memory Reduction using a Ring Abstraction over GPU RDMA for Distributed Quantum Monte Carlo Solver," 2021.

[38] D. C. Marcello *et al.*, "Introducing Octo-tiger/HPX: Simulating Interacting Binaries with Adaptive Mesh Refinement and the Fast Multipole Method," in *Proceedings of the International Conference on Accretion Processes in Cosmic Sources*, 2016.

[39] D. C. Marcello, "A Very Fast and Angular Momentum Conserving Tree Code," *Astronomical Journal*, vol. 154, p. 92, 2017.

[40] A. Kurganov *et al.*, "New High-Resolution Central Schemes for Nonlinear Conservation Laws and Convection-Diffusion Equations," *Journal of Computational Physics*, vol. 160, no. 1, pp. 241–282, 2000.

[41] G. Daiß *et al.*, "From Piz Daint to the stars: Simulation of stellar mergers using high-level abstractions," in *Proceedings of the international conference for high performance computing, networking, storage and analysis*, 2019, pp. 1–37.

[42] Marcello *et al.*, "Restart data for the v1309 scenario," Aug. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5213015

APPENDIX

A. Reproducibility Information: Artifact Description (AD)

- **Computational Artifacts:** yes
- **Summarize the experiments reported in the paper and how they were run:** We build Octo-Tiger with its BuildChain on both machines. We loaded compilers, MPI implementation, and CUDA using module files. All other dependencies were built using these. We used SLURM batch files on Piz Daint to submit the jobs. We used IBM Job Step Manager (JSM) to submit the jobs on Summit. All the resulting APEX data was copied to the LSU cloud for permanent storage. We processed the APEX data on a local work station using Python and TAU ParaProf for visualizaiton.
- **Software Artifact Availability:** All author-created software artifacts are maintained in a public repository under an OSI-approved license.
- **Hardware Artifact Availability:** There are no author-created hardware artifacts.
- **Data Artifact Availability:** All author-created data artifacts are maintained in a public repository under an OSI-approved license.
- **Proprietary Artifacts:** No author-created artifacts are proprietary.
- **URL/DOI List (separate line per URL/DOI):**
  - https://github.com/STEllAR-GROUP/hpx
  - https://www.boost.org/
  - https://gcc.gnu.org/
  - https://clang.llvm.org/
  - https://www.open-mpi.org/projects/hwloc/
  - https://cmake.org/
  - https://github.com/VcDevel/Vc
  - https://wci.llnl.gov/simulation/computer-codes/silo
  - https://www.hdfgroup.org/solutions/hdf5/
  - https://github.com/jemalloc/jemalloc
  - https://developer.nvidia.com/cuda-zone
  - https://www.ibm.com/products/spectrum-mpi
  - https://github.com/STEllAR-GROUP/OctoTigerBuildChain
- **Relevant hardware details:** CSCS's Piz Daint and ORNL's Summit
- **Compilers and versions:** gcc 10 and clang 11
- **Applications and versions:** Octo-Tiger
- **Libraries and versions:**

TABLE I
OCTO-TIGER'S DEPENDENCIES USED IN THIS STUDY. FIRST NUMBERS SHOW THE VERSIONS ON PIZ DAINT AND SECOND ONES ON SUMMIT IF DIFFERENT VERSIONS WERE USED.

clang/gcc	11/10	hdf5	1.8.12
cray-mpich	7.7.16	jemalloc	5.1.0 or 5.2.1
hpx	1.6.0	CUDA™	11.0.2
silo	4.10.2	boost	1.76.0
hwloc	1.11.12	VC	1.4.1
IBM® spectrum	10.4.0.3	CMake	3.19.5/3.18.0

- **Input datasets and versions:** The input files are available on Zenodo (https://doi.org/10.5281/zenodo.5213015)
- **Paper Modifications:** However, we had to use the IBM Spectrum compiler wrappers on Summit and the cray-mpich compiler wrappers on Piz Daint. On Summit the gcc 10.1.0 was used to compile all dependencies and Octo-Tiger itself. On Piz Daint clang 11 which is the default compiler was used to compile all dependencies and Octo-Tiger itself. The cmake version on Summit was 3.18.0 and 3.19.5 on Piz Daint.
  On both systems, we used Octo-Tiger's git hash (4c38f3bf) as the baseline. However, we had to do some minor changes with regard to compilation, which do not affect the astrophysics kernels. Therefore, we used a slightly diverged git hash (b091fd26) on Piz Daint and git hash (fd7faf5e) on Summit.
  We used a check point file from the production run for our scaling tests to measure the overhead introduced by APEX and NVIDIA CUDA Profiling Tools Interface. We loaded this check point file and ran 40 time steps for the scaling runs. Furthermore, we ran each simulation for the following configurations: No performance measurements, only CPU, and CPU + GPU on Piz Daint. On Summit, we did not run the CPU only run.
- **Output from commands that gather execution environment information:**
  - See Listing 1 for Piz Daint and Listing 2 for Summit, respectively.

Listing 1. Environment information on Piz Daint

```
ASSEMBLER_AARCH64=/opt/cray/pe/cce/11.0.0/binutils/cross/x86_64-aarch64/aarch64-linux
    ↪ -gnu/bin/as
PE_TPSL_DEFAULT_GENCOMPILERS_GNU_x86_skylake=8.2
XALT_DIR=/apps/daint/UES/xalt/xalt2/software/xalt/2.8.10
PE_LIBSCI_DEFAULT_VOLATILE_PRGENV=CRAYCLANG GNU INTEL
PE_ATP_PKGCONFIG_VARIABLES=ATP_CFLAGS_@prgenv@_@language@
LD_LIBRARY_PATH=/apps/daint/UES/xalt/xalt2/software/xalt/2.8.10/lib64:/apps/daint/UES
    ↪ /xalt/xalt2/software/xalt/2.8.10/lib:/opt/cray/pe/papi/6.0.0.4/lib64:/opt/cray/
    ↪ job/2.2.4-7.0.2.1_2.77__g36b56f4.ari/lib64
LS_COLORS=no=00:fi=00:di=01;34:ln=00;36:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd
    ↪ =40;33;01:or=41;33;01:ex=00;32:*.cmd=00;32:*.exe=01;32:*.com=01;32:*.bat
    ↪ =01;32:*.btm=01;32:*.dll=01;32:*.tar=00;31:*.tbz=00;31:*.tgz=00;31:*.rpm
    ↪ =00;31:*.deb=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31:*.lzma=00;31:*.zip
    ↪ =00;31:*.zoo=00;31:*.z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.tb2=00;31:*.tz2
    ↪ =00;31:*.tbz2=00;31:*.xz=00;31:*.avi=01;35:*.bmp=01;35:*.dl=01;35:*.fli
    ↪ =01;35:*.gif=01;35:*.gl=01;35:*.jpg=01;35:*.jpeg=01;35:*.mkv=01;35:*.mng
    ↪ =01;35:*.mov=01;35:*.mp4=01;35:*.mpg=01;35:*.pcx=01;35:*.pbm=01;35:*.pgm
    ↪ =01;35:*.png=01;35:*.ppm=01;35:*.svg=01;35:*.tga=01;35:*.tif=01;35:*.webm
    ↪ =01;35:*.webp=01;35:*.wmv=01;35:*.xbm=01;35:*.xcf=01;35:*.xpm=01;35:*.aiff
    ↪ =00;32:*.ape=00;32:*.au=00;32:*.flac=00;32:*.m4a=00;32:*.mid=00;32:*.mp3
    ↪ =00;32:*.mpc=00;32:*.ogg=00;32:*.voc=00;32:*.wav=00;32:*.wma=00;32:*.wv=00;32:
CRAY_IAA_INFO_FILE=/tmp/cray_iaa_info.33573830
PE_TPSL_64_DEFAULT_GENCOMPILERS_GNU_sandybridge=8.2
PE_TPSL_64_DEFAULT_GENCOMPS_CRAYCLANG_x86_64=90
SRUN_DEBUG=3
HOSTTYPE=x86_64
ALPS_LLI_STATUS_OFFSET=1
SLURM_STEP_ID=0
PE_TPSL_64_DEFAULT_GENCOMPILERS_GNU_x86_skylake=8.2
ATP_IGNORE_SIGTERM=1
XTPE_NETWORK_TARGET=aries
CSCS_CUSTOM_ENV=true
SLURM_STEP_GPUS=0
CRAY_UDREG_POST_LINK_OPTS=-L/opt/cray/udreg/2.3.2-7.0.2.1_2.41__g8175d3d.ari/lib64
PE_TPSL_DEFAULT_GENCOMPS_CRAYCLANG_haswell=90
SLURM_NODEID=0
PE_FFTW_DEFAULT_TARGET_share=share
PE_FFTW_DEFAULT_TARGET_ivybridge=ivybridge
SLURM_TASK_PID=12653
PE_TRILINOS_DEFAULT_GENCOMPS_CRAYCLANG_x86_64=90
PKG_CONFIG_PATH_DEFAULT=/opt/cray/pe/papi/6.0.0.4/lib64/pkgconfig
PRGENVMODULES=PrgEnv-cray:PrgEnv-gnu:PrgEnv-intel:PrgEnv-pgi
PE_TPSL_64_DEFAULT_GENCOMPILERS_INTEL_sandybridge=19.0
PE_TRILINOS_DEFAULT_GENCOMPS_GNU_x86_64=82
PE_LIBSCI_OMP_REQUIRES=
SSH_CONNECTION=148.187.1.9 33490 148.187.26.68 22
PE_MPICH_NV_LIBS_nvidia35=-lcudart
CRAY_NUM_COOKIES=2
LESSCLOSE=lessclose.sh %s %s
PE_SMA_DEFAULT_DIR_PGI_DEFAULT64=64
CRAY_LD_LIBRARY_PATH=/opt/cray/pe/mpt/7.7.16/gni/mpich-cray/9.0/lib:/opt/cray/pe/mpt
    ↪ /7.7.16/gni/mpich-crayclang/9.0/lib:/opt/cray/pe/perftools/20.10.0/lib64:/opt/
    ↪ cray/rca/2.2.20-7.0.2.1_2.83__g8e3fb5b.ari/lib64:/opt/cray/alps/6.6.59-7.0.2.1
    ↪ _3.72__g872a8d62.ari/lib64:/opt/cray/xpmem/2.2.20-7.0.2.1_2.65__g87eb960.ari/
    ↪ lib64:/opt/cray/dmapp/7.1.1-7.0.2.1_2.84__g38cf134.ari/lib64:/opt/cray/pe/pmi
```

```
↪ /5.0.17/lib64:/opt/cray/ugni/6.0.14.0-7.0.2.1_3.66__ge78e5b0.ari/lib64:/opt/
↪ cray/udreg/2.3.2-7.0.2.1_2.41__g8175d3d.ari/lib64:/opt/cray/pe/libsci/20.09.1/
↪ CRAYCLANG/9.0/x86_64/lib:/opt/cray/pe/cce/11.0.0/cce-clang/x86_64/lib:/opt/cray
↪ /pe/cce/11.0.0/cce/x86_64/lib
PE_MPICH_DEFAULT_DIR_CRAY_DEFAULT64=64
PE_PAPI_DEFAULT_ACCEL_FAMILY_LIBS_nvidia=,-lcupti,-lcudart,-lcuda
PE_LIBSCI_ACC_DEFAULT_PKGCONFIG_VARIABLES=
↪ PE_LIBSCI_ACC_DEFAULT_NV_SUFFIX_@accelerator@
SLURM_PRIO_PROCESS=0
PE_TRILINOS_DEFAULT_VOLATILE_PKGCONFIG_PATH=/opt/cray/pe/trilinos/12.18.1.1/@PRGENV@/
↪ @PE_TRILINOS_DEFAULT_GENCOMPS@/@PE_TRILINOS_DEFAULT_TARGET@/lib/pkgconfig
XKEYSYMDB=/usr/X11R6/lib/X11/XKeysymDB
PE_ENV=CRAY
LINKER_AARCH64=/opt/cray/pe/cce/11.0.0/binutils/cross/x86_64-aarch64/aarch64-linux-
↪ gnu/bin/ld
PE_TPSL_64_DEFAULT_GENCOMPS_INTEL_x86_skylake=190
CRAY_MPICH2_DIR=/opt/cray/pe/mpt/7.7.16/gni/mpich-crayclang/9.0
PE_LIBSCI_DEFAULT_GENCOMPS_GNU_x86_64=81
SLURM_CPU_BIND_VERBOSE=quiet
PE_PETSC_DEFAULT_GENCOMPS_GNU_x86_skylake=82
PE_PETSC_DEFAULT_GENCOMPILERS_CRAYCLANG_haswell=9.0
CRAYPAT_LD_LIBRARY_PATH=/opt/cray/pe/gcc-libs:/opt/cray/gcc-libs:/opt/cray/pe/
↪ perftools/20.10.0/lib64
SLURM_LOG_ACTIONS=yes
PE_PETSC_DEFAULT_VOLATILE_PRGENV=CRAYCLANG CRAYCLANG64 GNU GNU64 INTEL INTEL64
APPS_CSCS=/apps/cscs/daint
FTN_X86_64=/opt/cray/pe/cce/11.0.0/cce/x86_64
CSCS_PELOCAL_PRGENV=true
PE_PRODUCT_LIST=CRAYPE_HASWELL:CRAY_RCA:CRAY_ALPS:DVS:CRAY_XPMEM:CRAY_DMAPP:CRAY_PMI:
↪ CRAY_UGNI:CRAY_UDREG:CRAY_LIBSCI:CRAYPE:CRAY:PERFTOOLS:CRAYPAT
PE_TPSL_64_DEFAULT_REQUIRED_PRODUCTS=PE_MPICH:PE_LIBSCI
CRAY_CRAYPE_PREFIX=/opt/cray/pe/craype/2.7.3
CRAYPAT_ROOT=/opt/cray/pe/perftools/20.10.0
CRAY_DMAPP_INCLUDE_OPTS=-I/opt/cray/dmapp/7.1.1-7.0.2.1_2.84__g38cf134.ari/include -I
↪ /opt/cray/gni-headers/5.0.12.0-7.0.2.1_2.24__g3b1768f.ari/include
PE_TPSL_DEFAULT_GENCOMPILERS_GNU_x86_64=8.2
_=/usr/bin/env
PE_LIBSCI_MODULE_NAME=cray-libsci/20.09.1
PE_TPSL_64_DEFAULT_GENCOMPS_GNU_x86_skylake=82
LANG=en_US.UTF-8
PE_MPICH_DEFAULT_DIR_CRAYCLANG_DEFAULT64=64
TZ=Europe/Zurich
CRAY_BINUTILS_ROOT_X86_64=/opt/cray/pe/cce/11.0.0/binutils/x86_64/x86_64-pc-linux-gnu
↪ /../
SLURM_SUBMIT_DIR=/scratch/snx3000/USER/collect/Author-Kit
PE_MPICH_CXX_PKGCONFIG_LIBS=mpichcxx
PE_TPSL_DEFAULT_GENCOMPS_CRAYCLANG_x86_skylake=90
PE_TPSL_DEFAULT_GENCOMPILERS_INTEL_x86_skylake=19.0
PE_TPSL_64_DEFAULT_GENCOMPS_GNU_haswell=82
WINDOWMANAGER=xterm
PE_TPSL_64_DEFAULT_GENCOMPILERS_CRAYCLANG_x86_skylake=9.0
LESS=-M -I -R
PE_LIBSCI_DEFAULT_GENCOMPS_CRAYCLANG_x86_64=90
PE_PETSC_DEFAULT_VOLATILE_PKGCONFIG_PATH=/opt/cray/pe/petsc/3.13.3.0/complex/@PRGENV@
↪ /@PE_PETSC_DEFAULT_GENCOMPS@/@PE_PETSC_DEFAULT_TARGET@/lib/pkgconfig
ATP_INSTALL_DIR=/opt/cray/pe/atp/3.8.1
```

```
JAVA_ROOT=/usr/lib64/jvm/java
XALT_BINARYDATA_SIZE=5000
PE_TPSL_DEFAULT_GENCOMPILERS_CRAYCLANG_x86_skylake=9.0
HOSTNAME=daint105
SLURM_CSCS=yes
PE_TPSL_DEFAULT_GENCOMPILERS_CRAYCLANG_sandybridge=9.0
OLDPWD=/scratch/snx3000/USER/collect
PE_TPSL_DEFAULT_GENCOMPS_GNU_haswell=82
APPS=/apps/daint
CSHEDIT=emacs
PE_TPSL_DEFAULT_GENCOMPILERS_GNU_haswell=8.2
SLURM_STEPID=0
SLURM_SRUN_COMM_HOST=148.187.26.68
CRAY_RCA_INCLUDE_OPTS=-I/opt/cray/rca/2.2.20-7.0.2.1_2.83__g8e3fb5b.ari/include -I/
    ↪ opt/cray/krca/2.2.7-7.0.2.1_2.73__ge897ee1.ari/include -I/opt/cray-hss-devel
    ↪ /9.0.0/include
SLURM_DISTRIBUTION=cyclic
PE_TPSL_DEFAULT_VOLATILE_PKGCONFIG_PATH=/opt/cray/pe/tpsl/20.03.2/@PRGENV@/
    ↪ @PE_TPSL_DEFAULT_GENCOMPS@/@PE_TPSL_DEFAULT_TARGET@/lib/pkgconfig
GPG_TTY=/dev/pts/7
LESS_ADVANCED_PREPROCESSOR=no
PE_MPICH_DIR_CRAY_DEFAULT64=64
PE_ATP_MODULE_NAME=atp
PE_MPICH_GENCOMPS_CRAY=90
PE_PETSC_DEFAULT_GENCOMPS_CRAYCLANG_x86_skylake=90
PE_PETSC_DEFAULT_GENCOMPILERS_INTEL_x86_64=19.1
PE_FFTW_DEFAULT_TARGET_x86_64=x86_64
COLORTERM=1
GCC_X86_64=/opt/gcc/8.1.0/snos
ASSEMBLER_X86_64=/opt/cray/pe/cce/11.0.0/binutils/x86_64/x86_64-pc-linux-gnu/bin/as
FPATH=:/opt/cray/pe/modules/3.2.11.4/init/sh_funcs/no_redirect:/opt/cray/pe/modules
    ↪ /3.2.11.4/init/sh_funcs/no_redirect:/opt/cray/pe/modules/3.2.11.4/init/sh_funcs
    ↪ /no_redirect
PE_TPSL_64_DEFAULT_VOLATILE_PKGCONFIG_PATH=/opt/cray/pe/tpsl/20.03.2/@PRGENV@64/
    ↪ @PE_TPSL_64_DEFAULT_GENCOMPS@/@PE_TPSL_64_DEFAULT_TARGET@/lib/pkgconfig
PE_TPSL_64_DEFAULT_GENCOMPS_CRAYCLANG_x86_skylake=90
PE_TPSL_64_DEFAULT_GENCOMPILERS_CRAYCLANG_sandybridge=9.0
CRAY_PERFTOOLS_VERSION=20.10.0
ROCR_VISIBLE_DEVICES=0
PE_PKGCONFIG__PRODUCTS=PE_ATP
PE_PETSC_DEFAULT_GENCOMPS_GNU_sandybridge=82
PE_NETCDF_HDF5PARALLEL_DEFAULT_REQUIRED_PRODUCTS=PE_HDF5_PARALLEL
SQUEUE_SORT=-t,e,S
ATP_CFLAGS=
JAVA_HOME=/usr/lib64/jvm/java
COMPILERRT_PATH_X86_64=/opt/cray/pe/cce/11.0.0/cce-clang/x86_64/lib/clang/11.0.0/lib/
    ↪ linux
PE_LIBSCI_GENCOMPILERS_CRAYCLANG_x86_64=9.0
PE_PETSC_DEFAULT_GENCOMPILERS_INTEL_x86_skylake=19.1
PE_FFTW_DEFAULT_TARGET_x86_skylake=x86_skylake
SQUEUE_FORMAT=%.8i %.8u %.7a %.14j %.3t %9r %19S %.10M %.10L %.5D %.6C
SLURM_PROCID=0
APP2_STATE=20.10.0
PE_LIBSCI_ACC_DEFAULT_GENCOMPS_GNU_x86_64=81
PE_MPICH_DEFAULT_VOLATILE_PKGCONFIG_PATH=/opt/cray/pe/mpt/7.7.16/gni/mpich-
    ↪ @PRGENV@@PE_MPICH_DEFAULT_DIR_DEFAULT64@/@PE_MPICH_DEFAULT_GENCOMPS@/lib/
```

```
    ↪ pkgconfig
LINKER_X86_64=/opt/cray/pe/cce/11.0.0/binutils/x86_64/x86_64-pc-linux-gnu/bin/ld
SLURM_JOB_GID=31496
MACHTYPE=x86_64-suse-linux
PE_FFTW_DEFAULT_TARGET_broadwell=broadwell
SLURM_CPU_BIND=quiet,mask_cpu:0xFFFFFF
PE_TRILINOS_DEFAULT_GENCOMPILERS_CRAYCLANG_x86_64=9.0
PE_PAPI_DEFAULT_ACCEL_LIBS=
SLURMD_NODENAME=nid03508
PE_LIBSCI_ACC_DEFAULT_GENCOMPS_CRAYCLANG_x86_64=90
PE_SMA_DEFAULT_COMPFLAG=
XALT_EXECUTABLE_TRACKING=yes
PE_PKGCONFIG_PRODUCTS=PE_MPICH:PE_LIBSCI
PE_CRAY_DEFAULT_FIXED_PKGCONFIG_PATH=/opt/cray/pe/ga/5.3.0.10/CRAY/8.6/lib/pkgconfig
PE_PETSC_DEFAULT_GENCOMPS_CRAYCLANG_x86_64=90
CRAY_MPICH_BASEDIR=/opt/cray/pe/mpt/7.7.16/gni
PE_PETSC_DEFAULT_GENCOMPS_INTEL_sandybridge=191
PE_TPSL_64_DEFAULT_GENCOMPS_INTEL_sandybridge=190
PE_LIBSCI_GENCOMPILERS_GNU_x86_64=8.1
DVS_VERSION=0.9.0
MINICOM=-c on
SLURM_TASKS_PER_NODE=1
PAT_BUILD_PAPI_LIBDIR=/opt/cray/pe/papi/6.0.0.4/lib64
XALT_GPU_TRACKING=no
PE_MPICH_PKGCONFIG_VARIABLES=PE_MPICH_NV_LIBS_@accelerator@:
    ↪ PE_MPICH_ALTERNATE_LIBS_@multithreaded@:PE_MPICH_ALTERNATE_LIBS_@dpm@
PE_MPICH_PKGCONFIG_LIBS=mpich
PE_PARALLEL_NETCDF_DEFAULT_FIXED_PRGENV=PGI INTEL CRAYCLANG GNU
QT_SYSTEM_DIR=/usr/share/desktop-data
OSTYPE=linux
PE_LIBSCI_ACC_DEFAULT_NV_SUFFIX_nvidia60=nv60
PE_LEVEL=11.0
PE_NETCDF_DEFAULT_REQUIRED_PRODUCTS=PE_HDF5
PE_MPICH_NV_LIBS=
PE_FFTW2_DEFAULT_REQUIRED_PRODUCTS=PE_MPICH
PE_PETSC_DEFAULT_GENCOMPS_GNU_x86_64=82
XDG_SESSION_ID=678276
PE_TPSL_DEFAULT_REQUIRED_PRODUCTS=PE_MPICH:PE_LIBSCI
PE_TPSL_DEFAULT_GENCOMPS_GNU_sandybridge=82
USER=USER
SLURM_NNODES=1
SLURM_LAUNCH_NODE_IPADDR=148.187.26.68
PAGER=less
MODULE_VERSION=3.2.11.4
CRAY_CXX_IPA_LIBS_AARCH64=/opt/cray/pe/cce/11.0.0/cce/aarch64/lib/libcray-c++-rts.a
CRAY_ALPS_POST_LINK_OPTS=-L/opt/cray/alps/6.6.59-7.0.2.1_3.72__g872a8d62.ari/lib64
PE_TPSL_DEFAULT_GENCOMPS_CRAYCLANG_x86_64=90
SHMEM_ABORT_ON_ERROR=1
PE_PKG_CONFIG_PATH=/opt/cray/pe/valgrind4hpc/2.8.1/lib/pkgconfig:/opt/cray/pe/cti
    ↪ /2.8.1/lib/pkgconfig:/opt/cray/pe/atp/3.8.1/lib/pkgconfig
PE_CRAYCLANG_DEFAULT_FIXED_PKGCONFIG_PATH=/opt/cray/pe/parallel-netcdf/1.12.1.0/
    ↪ CRAYCLANG/9.0/lib/pkgconfig:/opt/cray/pe/netcdf-hdf5parallel/4.7.4.0/CRAYCLANG
    ↪ /9.0/lib/pkgconfig:/opt/cray/pe/netcdf/4.7.4.0/CRAYCLANG/9.0/lib/pkgconfig:/opt
    ↪ /cray/pe/hdf5-parallel/1.12.0.0/CRAYCLANG/9.0/lib/pkgconfig:/opt/cray/pe/hdf5
    ↪ /1.12.0.0/CRAYCLANG/9.0/lib/pkgconfig
PE_MPICH_DEFAULT_GENCOMPILERS_CRAY=9.0
```

```
PE_LIBSCI_DEFAULT_REQUIRED_PRODUCTS=PE_MPICH
PE_LIBSCI_ACC_DEFAULT_NV_SUFFIX_nvidia35=nv35
DVS_INCLUDE_OPTS=-I/opt/cray/dvs/2.12_2.2.176-7.0.2.1_12.2__g02f1c7d9/include
TOOLMODULES=apprentice:apprentice2:atp:chapel:cray-lgdb:craypat:craypkg-gen:cray-
    ↪ snplauncher:ddt:gdb:iobuf:papi:perftools:perftools-lite:stat:totalview:xt-
    ↪ craypat:xt-lgdb:xt-papi:xt-totalview
SLURM_STEP_TASKS_PER_NODE=1
CRAY_COOKIES=2223964160,2519859200
CRAY_CPU_TARGET=haswell
PE_TPSL_64_DEFAULT_GENCOMPILERS_GNU_x86_64=8.2
PE_LIBSCI_GENCOMPILERS_INTEL_x86_64=16.0
PE_INTEL_DEFAULT_FIXED_PKGCONFIG_PATH=/opt/cray/pe/parallel-netcdf/1.12.1.0/INTEL
    ↪ /19.1/lib/pkgconfig:/opt/cray/pe/netcdf-hdf5parallel/4.7.4.0/INTEL/19.1/lib/
    ↪ pkgconfig:/opt/cray/pe/netcdf/4.7.4.0/INTEL/19.1/lib/pkgconfig:/opt/cray/pe/mpt
    ↪ /7.7.16/gni/mpich-INTEL/16.0/lib/pkgconfig:/opt/cray/pe/hdf5-parallel/1.12.0.0/
    ↪ INTEL/19.1/lib/pkgconfig:/opt/cray/pe/hdf5/1.12.0.0/INTEL/19.1/lib/pkgconfig:/
    ↪ opt/cray/pe/ga/5.3.0.10/INTEL/18.0/lib/pkgconfig
PE_GA_DEFAULT_GENCOMPS_GNU=82 73
PE_SMA_DEFAULT_PKGCONFIG_VARIABLES=PE_SMA_COMPFLAG_@prgenv@
PE_LIBSCI_VOLATILE_PRGENV=CRAYCLANG GNU INTEL
KSH_AUTOLOAD=1
PE_MPICH_GENCOMPILERS_PGI=20.1
PE_TPSL_64_DEFAULT_GENCOMPS_CRAYCLANG_sandybridge=90
PKGCONFIG_ENABLED=1
PE_PETSC_DEFAULT_GENCOMPILERS_CRAYCLANG_sandybridge=9.0
CRAY_UGNI_POST_LINK_OPTS=-L/opt/cray/ugni/6.0.14.0-7.0.2.1_3.66__ge78e5b0.ari/lib64
PMI_CONTROL_PORT=31582
PE_MPICH_GENCOMPS_GNU=82 81 71
MORE=-sl
PE_PAPI_DEFAULT_ACCEL_LIBS_nvidia35=,-lcupti,-lcudart,-lcuda
CRAY_PERFTOOLS_PREFIX=/opt/cray/pe/perftools/20.10.0
PE_FORTRAN_PKGCONFIG_LIBS=mpichf90
PE_MPICH_DEFAULT_GENCOMPILERS_CRAYCLANG=9.0
PE_TRILINOS_DEFAULT_GENCOMPILERS_INTEL_x86_64=19.1
PE_TPSL_64_DEFAULT_GENCOMPS_CRAYCLANG_haswell=90
CRAY_CXX_IPA_LIBS=/opt/cray/pe/cce/11.0.0/cce/x86_64/lib/libcray-c++-rts.a
PE_MPICH_GENCOMPILERS_CRAY=9.0
CRAY_LIBSCI_BASE_DIR=/opt/cray/pe/libsci/20.09.1
PE_NETCDF_HDF5PARALLEL_DEFAULT_FIXED_PRGENV=GNU CRAYCLANG PGI INTEL
PWD=/scratch/snx3000/USER/collect/Author-Kit
TARGETMODULES=craype-abudhabi:craype-abudhabi-cu:craype-accel-host:craype-accel-
    ↪ nvidia20:craype-accel-nvidia30:craype-accel-nvidia35:craype-barcelona:craype-
    ↪ broadwell:craype-haswell:craype-hugepages128K:craype-hugepages128M:craype-
    ↪ hugepages16M:craype-hugepages256M:craype-hugepages2M:craype-hugepages32M:craype
    ↪ -hu...
```

Listing 2. Environment information on Summit

```
LS_COLORS=rs=0:di=38;5;33:ln=38;5;51:mh=00:pi=40;38;5;11:so=38;5;13:do=38;5;5:bd
    ↪ =48;5;232;38;5;11:cd=48;5;232;38;5;3:or=48;5;232;38;5;9:mi=01;05;37;41:su
    ↪ =48;5;196;38;5;15:sg=48;5;11;38;5;16:ca=48;5;196;38;5;226:tw=48;5;10;38;5;16:ow
    ↪ =48;5;10;38;5;21:st=48;5;21;38;5;15:ex=38;5;40:*.tar=38;5;9:*.tgz=38;5;9:*.arc
    ↪ =38;5;9:*.arj=38;5;9:*.taz=38;5;9:*.lha=38;5;9:*.lz4=38;5;9:*.lzh=38;5;9:*.lzma
    ↪ =38;5;9:*.tlz=38;5;9:*.txz=38;5;9:*.tzo=38;5;9:*.t7z=38;5;9:*.zip=38;5;9:*.z
    ↪ =38;5;9:*.dz=38;5;9:*.gz=38;5;9:*.lrz=38;5;9:*.lz=38;5;9:*.lzo=38;5;9:*.xz
    ↪ =38;5;9:*.zst=38;5;9:*.tzst=38;5;9:*.bz2=38;5;9:*.bz=38;5;9:*.tbz=38;5;9:*.tbz2
    ↪ =38;5;9:*.tz=38;5;9:*.deb=38;5;9:*.rpm=38;5;9:*.jar=38;5;9:*.war=38;5;9:*.ear
```

```
    ↪ =38;5;9:*.sar=38;5;9:*.rar=38;5;9:*.alz=38;5;9:*.ace=38;5;9:*.zoo=38;5;9:*.cpio
    ↪ =38;5;9:*.7z=38;5;9:*.rz=38;5;9:*.cab=38;5;9:*.wim=38;5;9:*.swm=38;5;9:*.dwm
    ↪ =38;5;9:*.esd=38;5;9:*.jpg=38;5;13:*.jpeg=38;5;13:*.mjpg=38;5;13:*.mjpeg
    ↪ =38;5;13:*.gif=38;5;13:*.bmp=38;5;13:*.pbm=38;5;13:*.pgm=38;5;13:*.ppm
    ↪ =38;5;13:*.tga=38;5;13:*.xbm=38;5;13:*.xpm=38;5;13:*.tif=38;5;13:*.tiff
    ↪ =38;5;13:*.png=38;5;13:*.svg=38;5;13:*.svgz=38;5;13:*.mng=38;5;13:*.pcx
    ↪ =38;5;13:*.mov=38;5;13:*.mpg=38;5;13:*.mpeg=38;5;13:*.m2v=38;5;13:*.mkv
    ↪ =38;5;13:*.webm=38;5;13:*.ogm=38;5;13:*.mp4=38;5;13:*.m4v=38;5;13:*.mp4v
    ↪ =38;5;13:*.vob=38;5;13:*.qt=38;5;13:*.nuv=38;5;13:*.wmv=38;5;13:*.asf
    ↪ =38;5;13:*.rm=38;5;13:*.rmvb=38;5;13:*.flc=38;5;13:*.avi=38;5;13:*.fli
    ↪ =38;5;13:*.flv=38;5;13:*.gl=38;5;13:*.dl=38;5;13:*.xcf=38;5;13:*.xwd=38;5;13:*.
    ↪ yuv=38;5;13:*.cgm=38;5;13:*.emf=38;5;13:*.ogv=38;5;13:*.ogx=38;5;13:*.aac
    ↪ =38;5;45:*.au=38;5;45:*.flac=38;5;45:*.m4a=38;5;45:*.mid=38;5;45:*.midi
    ↪ =38;5;45:*.mka=38;5;45:*.mp3=38;5;45:*.mpc=38;5;45:*.ogg=38;5;45:*.ra
    ↪ =38;5;45:*.wav=38;5;45:*.oga=38;5;45:*.opus=38;5;45:*.spx=38;5;45:*.xspf
    ↪ =38;5;45:
LD_LIBRARY_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/darshan-
    ↪ runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/lib:/sw/summit/spack-envs/base/
    ↪ opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi-10.4.0.3-20210112-
    ↪ v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/lib:/sw/summit/xl/16.1.1-10/xlsmp/5.1.1/lib:/
    ↪ sw/summit/xl/16.1.1-10/xlmass/9.1.1/lib:/sw/summit/xl/16.1.1-10/xlC/16.1.1/lib
    ↪ :/sw/summit/xl/16.1.1-10/xlf/16.1.1/lib:/sw/summit/xl/16.1.1-10/lib:/opt/ibm/
    ↪ spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-csm/lib
__LMOD_REF_COUNT_PATH=/sw/summit/xalt/1.2.1/bin:1;/sw/sources/lsf-tools/2.0/summit/
    ↪ bin:2;/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/darshan-
    ↪ runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/bin:1;/sw/sources/hpss/bin:1;/sw
    ↪ /summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/bin:1;/sw/summit/xl
    ↪ /16.1.1-10/xlC/16.1.1/bin:1;/sw/summit/xl/16.1.1-10/xlf/16.1.1/bin:1;/opt/ibm/
    ↪ spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-csm/etc:1;/opt/ibm/
    ↪ spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-csm/bin:1;/opt/ibm/
    ↪ csm/bin:1;/usr/local/bin:1;/usr/bin:1;/usr/local/sbin:1;/usr/sbin:1;/opt/ibm/
    ↪ flightlog/bin:1;/opt/ibm/jsm/bin:1
SSH_CONNECTION=174.64.18.156 41910 128.219.134.71 22
LMOD_FAMILY_MPI_VERSION=10.4.0.3-20210112
_=/usr/bin/env
LMOD_SYSTEM_NAME=summit
HISTCONTROL=ignoredups
MEMBERWORK=/gpfs/alpine/scratch/USER
BINARY_TYPE_HPC=
JSM_JSRUN_NO_WARN_TOPOLOGY=1
OMPI_MCA_io=romio321
LMOD_FAMILY_COMPILER_VERSION=16.1.1-10
HOSTNAME=login1
LMOD_SYSTEM_DEFAULT_MODULES=DefApps
OLDPWD=/ccs/proj/cph102/diehl/PowerTiger
__LMOD_REF_COUNT__LMFILES_=/sw/summit/spack-envs/base/modules/site/Core/xl/16.1.1-10.
    ↪ lua:1;/sw/summit/spack-envs/base/modules/spack/linux-rhel8-ppc64le/xl
    ↪ /16.1.1-10/spectrum-mpi/10.4.0.3-20210112.lua:1;/sw/summit/spack-envs/base/
    ↪ modules/site/Core/lsf-tools/2.0.lua:1;/sw/summit/spack-envs/base/modules/site/
    ↪ Core/hsi/5.0.2.p5.lua:1;/sw/summit/spack-envs/base/modules/spack/linux-rhel8-
    ↪ ppc64le/Core/darshan-runtime/3.3.0-lite.lua:1;/sw/summit/modulefiles/core/xalt
    ↪ /1.2.1.lua:1;/sw/summit/spack-envs/base/modules/site/Core/DefApps.lua:1
__LMOD_SET_FPATH=1
FPATH=/sw/summit/lmod/8.4/init/ksh_funcs
PAMI_ENABLE_STRIPING=0
```

```
OLCF_DARSHAN_RUNTIME_ROOT=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc
    ↪ -8.3.1/darshan-runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka
__LMOD_REF_COUNT_NLSPATH=/sw/summit/xl/16.1.1-10/msg/en_US/%N:2;/sw/summit/xl
    ↪ /16.1.1-10/xlC/16.1.1/msg/en_US/%N:1;/sw/summit/xl/16.1.1-10/xlf/16.1.1/msg/
    ↪ en_US/%N:1
OMPI_DIR=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3
__LMOD_REF_COUNT_LD_LIBRARY_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/
    ↪ gcc-8.3.1/darshan-runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/lib:1;/sw/
    ↪ summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/lib:1;/sw/summit/xl
    ↪ /16.1.1-10/xlsmp/5.1.1/lib:1;/sw/summit/xl/16.1.1-10/xlmass/9.1.1/lib:1;/sw/
    ↪ summit/xl/16.1.1-10/xlC/16.1.1/lib:1;/sw/summit/xl/16.1.1-10/xlf/16.1.1/lib:1;/
    ↪ sw/summit/xl/16.1.1-10/lib:1;/opt/ibm/spectrumcomputing/lsf/10.1.0.11/linux3
    ↪ .10-glibc2.17-ppc64le-csm/lib:1
__LMOD_REF_COUNT_PKG_CONFIG_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/
    ↪ gcc-8.3.1/darshan-runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/lib/pkgconfig
    ↪ :1
__LMOD_Priority_PATH=/sw/sources/lsf-tools/2.0/summit/bin:-9999;/sw/summit/xalt
    ↪ /1.2.1/bin:-9999
LSF_SERVERDIR=/opt/ibm/spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-
    ↪ csm/etc
__LMOD_REF_COUNT_PYTHONPATH=/sw/summit/xalt/1.2.1/site:1;/sw/summit/xalt/1.2.1/
    ↪ libexec:1
LSF_ENVDIR=/opt/ibm/spectrumcomputing/lsf/conf
XDG_SESSION_ID=8024
PAMI_IBV_ADAPTER_AFFINITY=1
USER=USER
OMPI_CXX=/sw/summit/xl/16.1.1-10/xlC/16.1.1/bin/xlc++_r
__LMOD_REF_COUNT_MODULEPATH=/sw/summit/spack-envs/base/modules/spack/linux-rhel8-
    ↪ ppc64le/spectrum-mpi/10.4.0.3-20210112-v7qymni/xl/16.1.1-10:1;/sw/summit/spack-
    ↪ envs/base/modules/spack/linux-rhel8-ppc64le/xl/16.1.1-10:1;/sw/summit/spack-
    ↪ envs/base/modules/spack/linux-rhel8-ppc64le/Core:1;/sw/summit/spack-envs/base/
    ↪ modules/site/Core:1;/sw/summit/modulefiles/core:1
__LMOD_REF_COUNT_LOADEDMODULES=xl/16.1.1-10:1;spectrum-mpi/10.4.0.3-20210112:1;lsf-
    ↪ tools/2.0:1;hsi/5.0.2.p5:1;darshan-runtime/3.3.0-lite:1;xalt/1.2.1:1;DefApps:1
OLCF_FAMILY_MPI_VERSION=10.4.0.3-20210112
PAMI_IBV_QP_SERVICE_LEVEL=8
PWD=/ccs/proj/cph102/diehl/PowerTiger/src/octotiger-kokkos
OLCF_FAMILY_COMPILER_VERSION=16.1.1-10
HOME=/ccs/home/USER
CMAKE_PREFIX_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/
    ↪ darshan-runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka:/sw/summit/spack-envs/
    ↪ base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi-10.4.0.3-20210112-
    ↪ v7qymniwgi6mtxqsjd7p5jxinxzdkhn3
SSH_CLIENT=174.64.18.156 41910 22
LMOD_VERSION=8.4
OPAL_PREFIX=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-
    ↪ mpi-10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3
CPATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/include
OPAL_LIBDIR=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-
    ↪ mpi-10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/lib
BASH_ENV=/sw/summit/lmod/8.4/init/bash
LMOD_MPI_VERSION=10.4.0.3-20210112-v7qymni
NLSPATH=/sw/summit/xl/16.1.1-10/msg/en_US/%N:/sw/summit/xl/16.1.1-10/xlC/16.1.1/msg/
```

```
    ↪ en_US/%N:/sw/summit/xl/16.1.1-10/xlf/16.1.1/msg/en_US/%N
OLCF_FAMILY_MPI=spectrum-mpi
LSF_BINDIR=/opt/ibm/spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-csm/
    ↪ bin
PAMI_IBV_ENABLE_DCT=1
__LMOD_REF_COUNT_LIBRARY_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc
    ↪ -8.3.1/darshan-runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/lib:1;/sw/summit/
    ↪ spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/lib:1
LIBRARY_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/darshan-
    ↪ runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/lib:/sw/summit/spack-envs/base/
    ↪ opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi-10.4.0.3-20210112-
    ↪ v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/lib
__LMOD_REF_COUNT_CMAKE_PREFIX_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le
    ↪ /gcc-8.3.1/darshan-runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka:1;/sw/summit/
    ↪ spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3:1
LMOD_sys=Linux
LOADEDMODULES=xl/16.1.1-10:spectrum-mpi/10.4.0.3-20210112:lsf-tools/2.0:hsi/5.0.2.p5:
    ↪ darshan-runtime/3.3.0-lite:xalt/1.2.1:DefApps
HWLOC_KEEP_NVIDIA_GPU_NUMA_NODES=1
__LMOD_REF_COUNT_MANPATH=/sw/sources/hpss/man:1;/sw/summit/spack-envs/base/opt/linux-
    ↪ rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi-10.4.0.3-20210112-
    ↪ v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/share/man:1;/sw/summit/xl/16.1.1-10/xlC
    ↪ /16.1.1/man/en_US:1;/sw/summit/xl/16.1.1-10/xlf/16.1.1/man/en_US:1;/sw/summit/
    ↪ lmod/8.4/share/man:1;/opt/ibm/spectrumcomputing/lsf/10.1.0.11/man:1
LMOD_ROOT=/sw/summit/lmod
SSH_TTY=/dev/pts/182
MAIL=/var/spool/mail/USER
OLCF_MODULEPATH_ROOT=/sw/summit/modulefiles
__Init_Default_Modules=1
SHELL=/bin/bash
TERM=xterm-256color
_ModuleTable_Sz_=6
OMPI_LD_PRELOAD_PREPEND=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/
    ↪ darshan-runtime-3.2.1-rucchppqoqp7jf2irm233h6q523wpzvf/lib/libdarshan.so
__LMOD_REF_COUNT_CPATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl
    ↪ -16.1.1-10/spectrum-mpi-10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/
    ↪ include:1
OLCF_SPECTRUM_MPI_ROOT=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl
    ↪ -16.1.1-10/spectrum-mpi-10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3
XLSF_UIDDIR=/opt/ibm/spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-csm/
    ↪ lib/uid
LMOD_FAMILY_COMPILER=xl
OMPI_FC=/sw/summit/xl/16.1.1-10/xlf/16.1.1/bin/xlf2008_r
OLCF_LMOD_ROOT=/sw/summit
PROJWORK=/gpfs/alpine/proj-shared
SHLVL=2
PYTHONPATH=/sw/summit/xalt/1.2.1/site:/sw/summit/xalt/1.2.1/libexec
XL_LINKER=/sw/summit/xalt/1.2.1/bin/ld
MANPATH=/sw/sources/hpss/man:/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl
    ↪ -16.1.1-10/spectrum-mpi-10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/
    ↪ share/man:/sw/summit/xl/16.1.1-10/xlC/16.1.1/man/en_US:/sw/summit/xl/16.1.1-10/
    ↪ xlf/16.1.1/man/en_US:/sw/summit/lmod/8.4/share/man:/opt/ibm/spectrumcomputing/
    ↪ lsf/10.1.0.11/man:::/opt/puppetlabs/puppet/share/man
LSF_LIBDIR=/opt/ibm/spectrumcomputing/lsf/10.1.0.11/linux3.10-glibc2.17-ppc64le-csm/
```

```
    ↪ lib
PAMI_CUDA_AWARE_THRESH=320000
MODULEPATH=/sw/summit/spack-envs/base/modules/spack/linux-rhel8-ppc64le/spectrum-mpi
    ↪ /10.4.0.3-20210112-v7qymni/xl/16.1.1-10:/sw/summit/spack-envs/base/modules/
    ↪ spack/linux-rhel8-ppc64le/xl/16.1.1-10:/sw/summit/spack-envs/base/modules/spack
    ↪ /linux-rhel8-ppc64le/Core:/sw/summit/spack-envs/base/modules/site/Core:/sw/
    ↪ summit/modulefiles/core
OLCF_XLF_ROOT=/sw/summit/xl/16.1.1-10/xlf/16.1.1
OMPI_CC=/sw/summit/xl/16.1.1-10/xlC/16.1.1/bin/xlc_r
OLCF_HSI_ROOT=/sw/sources/hpss
LOGNAME=USER
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/15911/bus
XDG_RUNTIME_DIR=/run/user/15911
XALT_OLCF=1
MODULEPATH_ROOT=/sw/summit/modulefiles
OLCF_XLSMP_ROOT=/sw/summit/xl/16.1.1-10/xlsmp/5.1.1
PAMI_IBV_ENABLE_OOO_AR=1
LMOD_MPI_NAME=spectrum-mpi
PATH=/sw/summit/xalt/1.2.1/bin:/sw/sources/lsf-tools/2.0/summit/bin:/sw/summit/spack-
    ↪ envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/darshan-runtime-3.3.0-
    ↪ foy76v6kypdpepgs4nmahubhm2yfwmka/bin:/sw/sources/hpss/bin:/sw/summit/spack-envs
    ↪ /base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi-10.4.0.3-20210112-
    ↪ v7qymniwgi6mtxqsjd7p5jxinxzdkhn3/bin:/sw/summit/xl/16.1.1-10/xlC/16.1.1/bin:/sw
    ↪ /summit/xl/16.1.1-10/xlf/16.1.1/bin:/opt/ibm/spectrumcomputing/lsf/10.1.0.11/
    ↪ linux3.10-glibc2.17-ppc64le-csm/etc:/opt/ibm/spectrumcomputing/lsf/10.1.0.11/
    ↪ linux3.10-glibc2.17-ppc64le-csm/bin:/opt/ibm/csm/bin:/usr/local/bin:/usr/bin:/
    ↪ usr/local/sbin:/usr/sbin:/opt/ibm/flightlog/bin:/opt/ibm/jsm/bin:/opt/
    ↪ puppetlabs/bin:/usr/lpp/mmfs/bin
_LMFILES_=/sw/summit/spack-envs/base/modules/site/Core/xl/16.1.1-10.lua:/sw/summit/
    ↪ spack-envs/base/modules/spack/linux-rhel8-ppc64le/xl/16.1.1-10/spectrum-mpi
    ↪ /10.4.0.3-20210112.lua:/sw/summit/spack-envs/base/modules/site/Core/lsf-tools
    ↪ /2.0.lua:/sw/summit/spack-envs/base/modules/site/Core/hsi/5.0.2.p5.lua:/sw/
    ↪ summit/spack-envs/base/modules/spack/linux-rhel8-ppc64le/Core/darshan-runtime
    ↪ /3.3.0-lite.lua:/sw/summit/modulefiles/core/xalt/1.2.1.lua:/sw/summit/spack-
    ↪ envs/base/modules/site/Core/DefApps.lua
OLCF_FAMILY_COMPILER=xl
MODULESHOME=/sw/summit/lmod/8.4
LMOD_SETTARG_FULL_SUPPORT=no
PKG_CONFIG_PATH=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/gcc-8.3.1/darshan-
    ↪ runtime-3.3.0-foy76v6kypdpepgs4nmahubhm2yfwmka/lib/pkgconfig
XALT_ETC_DIR=/sw/summit/xalt/1.2.1/etc
HISTSIZE=1000
LMOD_PKG=/sw/summit/lmod/8.4
OLCF_XL_ROOT=/sw/summit/xl/16.1.1-10
OLCF_XLMASS_ROOT=/sw/summit/xl/16.1.1-10/xlmass/9.1.1
OLCF_XLC_ROOT=/sw/summit/xl/16.1.1-10/xlC/16.1.1
LMOD_CMD=/sw/summit/lmod/8.4/libexec/lmod
WORLDWORK=/gpfs/alpine/world-shared
MPI_ROOT=/sw/summit/spack-envs/base/opt/linux-rhel8-ppc64le/xl-16.1.1-10/spectrum-mpi
    ↪ -10.4.0.3-20210112-v7qymniwgi6mtxqsjd7p5jxinxzdkhn3
CVS_RSH=ssh
LESSOPEN=||/usr/bin/lesspipe.sh %s
LMOD_DIR=/sw/summit/lmod/8.4/libexec
LMOD_FAMILY_MPI=spectrum-mpi
BASH_FUNC_module%%=() {  eval $($LMOD_CMD bash "$@") && eval $(${LMOD_SETTARG_CMD:-:}
    ↪  -s sh)
```

```
}
BASH_FUNC_ml%%=() {   eval $($LMOD_DIR/ml_cmd "$@")
}
+ lsb_release -a
LSB Version:    :core-4.1-noarch:core-4.1-ppc64le
Distributor ID: RedHatEnterprise
Description:    Red Hat Enterprise Linux release 8.2 (Ootpa)
Release:        8.2
Codename:       Ootpa
+ uname -a
Linux login1 4.18.0-193.46.1.el8_2.ppc64le #1 SMP Thu Feb 18 09:47:51 EST 2021
    ↪ ppc64le ppc64le ppc64le GNU/Linux
+ lscpu
Architecture:          ppc64le
Byte Order:            Little Endian
CPU(s):                128
On-line CPU(s) list:   0-127
Thread(s) per core:    4
Core(s) per socket:    16
Socket(s):             2
NUMA node(s):          6
Model:                 2.1 (pvr 004e 1201)
Model name:            POWER9, altivec supported
CPU max MHz:           3800.0000
CPU min MHz:           2300.0000
L1d cache:             32K
L1i cache:             32K
L2 cache:              512K
L3 cache:              10240K
NUMA node0 CPU(s):     0-63
NUMA node8 CPU(s):     64-127
NUMA node252 CPU(s):
NUMA node253 CPU(s):
NUMA node254 CPU(s):
NUMA node255 CPU(s):
+ cat /proc/meminfo
MemTotal:       601136896 kB
MemFree:        79823104 kB
MemAvailable:   289907648 kB
Buffers:               0 kB
Cached:         208338048 kB
SwapCached:            0 kB
Active:         127597376 kB
Inactive:       92910912 kB
Active(anon):   29790016 kB
Inactive(anon): 12175296 kB
Active(file):   97807360 kB
Inactive(file): 80735616 kB
Unevictable:    16891840 kB
Mlocked:        16891840 kB
SwapTotal:             0 kB
SwapFree:              0 kB
Dirty:                 0 kB
Writeback:             0 kB
AnonPages:      29051904 kB
Mapped:         191502016 kB
```

```
Shmem:              29811776 kB
KReclaimable:       34141888 kB
Slab:               83703936 kB
SReclaimable:       34141888 kB
SUnreclaim:         49562048 kB
KernelStack:           66000 kB
PageTables:            73984 kB
NFS_Unstable:              0 kB
Bounce:                    0 kB
WritebackTmp:              0 kB
CommitLimit:        300568448 kB
Committed_AS:       97673088 kB
VmallocTotal:       549755813888 kB
VmallocUsed:               0 kB
VmallocChunk:              0 kB
Percpu:               778240 kB
HardwareCorrupted:       128 kB
AnonHugePages:       3235840 kB
ShmemHugePages:            0 kB
ShmemPmdMapped:            0 kB
CmaTotal:           26853376 kB
CmaFree:             1799168 kB
HugePages_Total:           0
HugePages_Free:            0
HugePages_Rsvd:            0
HugePages_Surp:            0
Hugepagesize:           2048 kB
Hugetlb:                   0 kB
+ inxi -F -c0
./collect_environment.sh: line 14: inxi: command not found
+ lsblk -a
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda         8:0    1  1.8T  0 disk
sdb         8:16   1  1.8T  0 disk
nvme0n1   259:1    0  1.5T  0 disk
+ lsscsi -s
./collect_environment.sh: line 16: lsscsi: command not found
+ module list
++ /sw/summit/lmod/8.4/libexec/lmod bash list

Currently Loaded Modules:
  1) xl/16.1.1-10     2) spectrum-mpi/10.4.0.3-20210112     3) lsf-tools/2.0     4) hsi
     ↪ /5.0.2.p5     5) darshan-runtime/3.3.0-lite     6) xalt/1.2.1     7) DefApps

+ eval 'MODULEPATH=/sw/summit/spack-envs/base/modules/spack/linux-rhel8-ppc64le/
    ↪ spectrum-mpi/10.4.0.3-20210112-v7qymni/xl/16.1.1-10:/sw/summit/spack-envs/base/
    ↪ modules/spack/linux-rhel8-ppc64le/xl/16.1.1-10:/sw/summit/spack-envs/base/
    ↪ modules/spack/linux-rhel8-ppc64le/Core:/sw/summit/spack-envs/base/modules/site/
    ↪ Core:/sw/summit/modulefiles/core;'
++ MODULEPATH=/sw/summit/spack-envs/base/modules/spack/linux-rhel8-ppc64le/spectrum-
    ↪ mpi/10.4.0.3-20210112-v7qymni/xl/16.1.1-10:/sw/summit/spack-envs/base/modules/
    ↪ spack/linux-rhel8-ppc64le/xl/16.1.1-10:/sw/summit/spack-envs/base/modules/spack
    ↪ /linux-rhel8-ppc64le/Core:/sw/summit/spack-envs/base/modules/site/Core:/sw/
    ↪ summit/modulefiles/core
++ : -s sh
+ eval
```

```
+ nvidia-smi
Wed Sep  8 16:13:52 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.80.02    Driver Version: 450.80.02    CUDA Version: 11.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000035:04:00.0 Off |                    0 |
| N/A   41C    P0    38W / 300W |      0MiB / 16160MiB |      0%   E. Process |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
+ lshw -short -quiet -sanitize
+ cat
./collect_environment.sh: line 19: lshw: command not found
+ lspci
0000:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0000:01:00.0 Non-Volatile memory controller: Samsung Electronics Co Ltd NVMe SSD
    ↪ Controller 172Xa/172Xb (rev 01)
0001:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0001:01:00.0 USB controller: Texas Instruments TUSB73x0 SuperSpeed USB 3.0 xHCI Host
    ↪ Controller (rev 02)
0002:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0002:01:00.0 PCI bridge: ASPEED Technology, Inc. AST1150 PCI-to-PCI Bridge (rev 04)
0002:02:00.0 VGA compatible controller: ASPEED Technology, Inc. ASPEED Graphics
    ↪ Family (rev 41)
0003:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0003:01:00.0 Infiniband controller: Mellanox Technologies MT28800 Family [ConnectX-5
    ↪ Ex]
0003:01:00.1 Infiniband controller: Mellanox Technologies MT28800 Family [ConnectX-5
    ↪ Ex]
0004:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0004:01:00.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0004:01:00.1 System peripheral: PLX Technology, Inc. Device 87d0 (rev ca)
0004:01:00.2 System peripheral: PLX Technology, Inc. Device 87d0 (rev ca)
0004:01:00.3 System peripheral: PLX Technology, Inc. Device 87d0 (rev ca)
0004:01:00.4 System peripheral: PLX Technology, Inc. Device 87d0 (rev ca)
0004:02:02.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0004:02:0a.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0004:02:0b.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0004:02:0c.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0004:03:00.0 SATA controller: Marvell Technology Group Ltd. 88SE9235 PCIe 2.0 x2 4-
    ↪ port SATA 6 Gb/s Controller (rev 11)
0004:04:00.0 3D controller: NVIDIA Corporation GV100GL [Tesla V100 SXM2 16GB] (rev a1
    ↪ )
0004:05:00.0 3D controller: NVIDIA Corporation GV100GL [Tesla V100 SXM2 16GB] (rev a1
    ↪ )
```

```
0005:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0005:01:00.0 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme BCM5719
    ↪ Gigabit Ethernet PCIe (rev 01)
0005:01:00.1 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme BCM5719
    ↪ Gigabit Ethernet PCIe (rev 01)
0006:00:00.0 Bridge: IBM Device 04ea (rev 01)
0006:00:00.1 Bridge: IBM Device 04ea (rev 01)
0006:00:00.2 Bridge: IBM Device 04ea (rev 01)
0006:00:01.0 Bridge: IBM Device 04ea (rev 01)
0006:00:01.1 Bridge: IBM Device 04ea (rev 01)
0006:00:01.2 Bridge: IBM Device 04ea (rev 01)
0007:00:00.0 Bridge: IBM Device 04ea (rev 01)
0007:00:00.1 Bridge: IBM Device 04ea (rev 01)
0007:00:00.2 Bridge: IBM Device 04ea (rev 01)
0007:00:01.0 Bridge: IBM Device 04ea (rev 01)
0007:00:01.1 Bridge: IBM Device 04ea (rev 01)
0007:00:01.2 Bridge: IBM Device 04ea (rev 01)
0030:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0030:01:00.0 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme II
    ↪ BCM57800 1/10 Gigabit Ethernet (rev 10)
0030:01:00.1 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme II
    ↪ BCM57800 1/10 Gigabit Ethernet (rev 10)
0030:01:00.2 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme II
    ↪ BCM57800 1/10 Gigabit Ethernet (rev 10)
0030:01:00.3 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme II
    ↪ BCM57800 1/10 Gigabit Ethernet (rev 10)
0033:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0033:01:00.0 Infiniband controller: Mellanox Technologies MT28800 Family [ConnectX-5
    ↪ Ex]
0033:01:00.1 Infiniband controller: Mellanox Technologies MT28800 Family [ConnectX-5
    ↪ Ex]
0034:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0035:00:00.0 PCI bridge: IBM POWER9 Host Bridge (PHB4)
0035:01:00.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0035:02:04.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0035:02:05.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0035:02:0d.0 PCI bridge: PLX Technology, Inc. Device 8725 (rev ca)
0035:03:00.0 3D controller: NVIDIA Corporation GV100GL [Tesla V100 SXM2 16GB] (rev a1
    ↪ )
0035:04:00.0 3D controller: NVIDIA Corporation GV100GL [Tesla V100 SXM2 16GB] (rev a1
    ↪ )
```