

Improving Energy Efficiency and Performance of Weather and Climate Simulations by Leveraging the Heterogeneity of Modern Systems

Julius Plehn
Deutsches Klimarechenzentrum
GmbH
Hamburg, Germany
plehn@dkrz.de

Christian von Elm
CIDS, Information Services and High
Performance Computing (ZIH)
TU Dresden
Dresden, Germany
christian.von_elm@tu-dresden.de

Pay Giesselmann
Deutsches Klimarechenzentrum
GmbH
Hamburg, Germany
giesselmann@dkrz.de

Carsten Clauss
ParTec AG
Munich, Germany
clauss@par-tec.com

Hendryk Bockelmann
Deutsches Klimarechenzentrum
GmbH
Hamburg, Germany
bockelmann@dkrz.de

Robert Schöne
CIDS, Information Services and High
Performance Computing (ZIH)
TU Dresden
Dresden, Germany
robert.schoene@tu-dresden.de

Jan Frederik Engels
Deutsches Klimarechenzentrum
GmbH
Hamburg, Germany
engels@dkrz.de

Abstract

The increasing need for higher resolution and greater accuracy in weather forecasts and climate simulations continues to drive software and hardware developments in high-performance computing (HPC) systems. To achieve increasingly faster simulations, the adoption of hardware accelerators has proven highly effective in recent years. However, these HPC codes exhibit, in parts, divergent memory access and computation patterns. Hence, their performance benefits strongly depend on how well the software's computational characteristics align with the underlying hardware architecture. While an accelerator may be well-suited for some code regions, other architectures may achieve better performance and energy efficiency elsewhere. We therefore propose a highly heterogeneous setup for a performant and energy-efficient execution of complex HPC codes such as those used in the weather and climate domain, incorporating a variety of processor and accelerator architectures.

Using the climate and weather model ICON, we explore the potential of mapping of components onto compute architectures. We describe the design of a highly heterogeneous test cluster and discuss its implications for middleware and software management. Furthermore, we present our comprehensive energy measurement infrastructure which allows for comparisons with water-cooled

production systems. Leveraging this, we demonstrate that a heterogeneous cluster can achieve up to 40% higher energy efficiency compared to a traditional homogeneous configuration.

CCS Concepts

• **Computer systems organization** → **Parallel architectures**;
• **Applied computing** → **Earth and atmospheric sciences**; •
Hardware → **Power estimation and optimization**; • **General and reference** → **Performance**; **Measurement**.

Keywords

High-performance computing; Heterogeneous computing; Energy efficiency; Climate and weather modeling; Performance optimization; Energy measurement

ACM Reference Format:

Julius Plehn, Christian von Elm, Pay Giesselmann, Carsten Clauss, Hendryk Bockelmann, Robert Schöne, and Jan Frederik Engels. 2026. Improving Energy Efficiency and Performance of Weather and Climate Simulations by Leveraging the Heterogeneity of Modern Systems. In *Proceedings of the 17th ACM/SPEC International Conference on Performance Engineering (ICPE '26)*, May 04–08, 2026, Florence, Italy. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3777884.3796996>



This work is licensed under a Creative Commons Attribution 4.0 International License. *ICPE '26, Florence, Italy*

© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2325-4/2026/05
<https://doi.org/10.1145/3777884.3796996>

1 Introduction

High Performance Computing (HPC) provides hardware resources, which enable software developers to run simulations that exceed the capacities of standalone computers. Given the expense of HPC systems, they are usually used for simulations with economic or societal benefits, such as thermo-fluid systems or weather prediction. The ICOSahedral Nonhydrostatic (ICON) model is an example for one of the latter. The model is employed operationally for weather forecasting at the Deutscher Wetterdienst (DWD) and other meteorological services as well as for high-resolution production runs on large high-performance computers, such as LUMI and JUPITER. Both applications require substantial energy, with JUPITER capable of consuming up to 20 MW. Given the cost of energy and hardware, an efficient usage of system resources is beneficial in terms of total cost-of-ownership, but also helps reducing the environmental footprint of the computing center. However, using hardware efficiently is not an easy task since the algorithmic design defines the patterns for data access and computation – and these patterns are not necessarily reflected in the used hardware. While the increasing heterogeneity of hardware could solve this issue and specialized accelerators provide benefits for specific application classes, they raise the issue of finding the best processor architecture for an application. The era of heterogeneity in HPC is mostly dominated by accelerators, which contribute more than 80 % to the most recent Top500’s performance share¹ as shown in Figure 1.

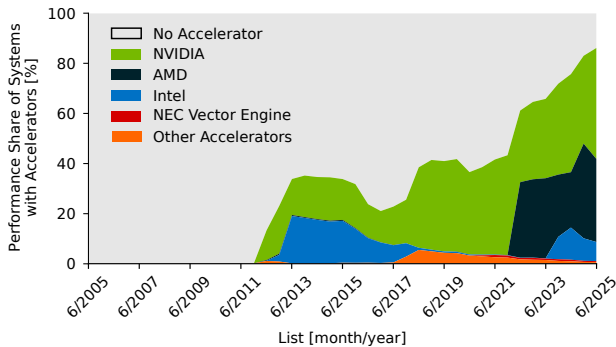


Figure 1: Performance share of systems with accelerators in Top500 lists over 20 years.

However, host processors that do not implement an x86-like Instruction Set Architecture (ISA) are also increasing their share. With this wide selection of architectures, systems can nowadays be matched better to the software that they need to run.

In this paper, we analyze the efficiency of the ICON code on a variety of hardware and propose a heterogeneous system that can run this code efficiently as a decision aid for computing center operators. The paper is structured as follows: In Section 2 we provide background information on ICON and how heterogeneous computing applies to it. This is followed by Section 3, where we present selected related work in the field of heterogeneous high-performance computing. In Section 4, we describe a cluster with

five different contemporary processor architectures and two different accelerators that we analyze for their performance and energy efficiency. Following that, we present an analysis of how efficient ICON can be executed on the selected architectures in Section 5. We conclude our paper with a summary of our findings and an outlook on future work in Section 6.

2 Background

The following section introduces the ICON modelling framework and discusses its execution on heterogeneous hardware. We first give a brief overview of ICON’s modular structure and the interaction between its Earth system components. Afterwards, we provide background on the challenges of utilizing heterogeneous hardware and coordinating interactions between different components.

2.1 ICON

ICON is a state-of-the-art numerical framework developed to simulate weather and climate processes across a wide range of spatial and temporal scales. It integrates multiple Earth system components, including the atmosphere, the ocean, and ocean biogeochemistry, enabling simulations of the full Earth system. Figure 2 illustrates available components, their interplay and the corresponding fluxes. It covers more capabilities of ICON than those evaluated in the experiment presented in this paper. Ocean and atmosphere exchange water, momentum and carbon at the sea surface via the coupler YAC [15]. In addition, the land component of the atmosphere exchanges water with the ocean via the coupler and momentum, water and in other setups carbon with the atmosphere directly. The ocean biogeochemistry sub-model HAMOCC exchanges various quantities with the ocean directly and in some setups carbon with the atmosphere via the coupler. The tightly integrated land component is not computationally demanding and therefore not further differentiated in this paper, in contrast to the radiation component of the atmosphere, which we regard explicitly.

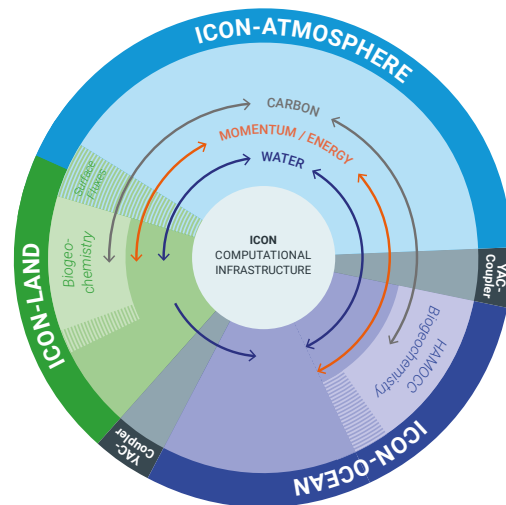


Figure 2: ICON components and their interaction, reproduced from [16], licensed under CC BY 4.0.

¹<https://www.top500.org/lists/green500/2025/06/>

2.2 Running Climate and Weather Prediction Codes on Heterogeneous Hardware

Achieving high performance and energy efficiency in HPC applications across various hardware architectures typically requires dedicated porting efforts. ICON, in its various scientific configurations, can run on a large number of architectures. However, not every component is ported to each architecture. The most versatile component is the atmosphere, which is ported to various kinds of CPUs, including the ability to leverage OpenMP, to (NVIDIA-)GPUs using OpenACC and to NEC SX-Aurora TSUBASA, a vector architecture. All relevant subcomponents of the atmosphere, especially the computation of radiative fluxes, are ported to these architectures as well. The ocean component and its subcomponent, however, are, at the time of writing, only fully ported to CPUs. One of the fundamental computational characteristic of all of ICON's components is that they are memory bandwidth bound on most architectures.

Like in many other climate codes, the atmosphere and the ocean run concurrently and exchange fluxes at the sea surface via a coupling library (in ICON's case YAC), which eventually uses MPI. Therefore, these components can be mapped to distinct computing architectures if a common MPI installation exists. A challenge beyond the scope of this paper is the correct resource allocation.

Mapping additional components to separate architectures is more complicated. First of all, we need two simulation processes that can run concurrently – leaving an architecture idle during waiting phases is almost guaranteed to decrease energy efficiency, due to the fact that the systems power consumption and wait time will still contribute to the overall energy. Second, the amount of data that needs to be exchanged between the two components should be rather small, since exchanging large amounts of data takes additional time and a minimal runtime is beneficial to lower energy consumption. In ICON, there are at least two components which fulfill both requirements: radiation and ocean biogeochemistry. Currently, only the latter one can be run asynchronously using the YAXT library. For the former, there was an asynchronous implementation for a now outdated radiation scheme, which is no longer present in the code.

The high-level interaction between ICON's components running on three different architectures is depicted in Figure 3. As the simulation progresses, components exchange data via different coupling mechanisms and at different intervals. This also highlights the challenge of balancing the computational cost of each component with respect to the architecture. In this idealized figure, the components are perfectly balanced such that there is virtually no idle time between communication and computation. The challenge of resource balancing is amplified as the number of participating components increases and is a common issue in climate computing, as highlighted, for example, in [1]. In a similar fashion, we balance the resources for our experiment, which is described in more detail in Section 5.

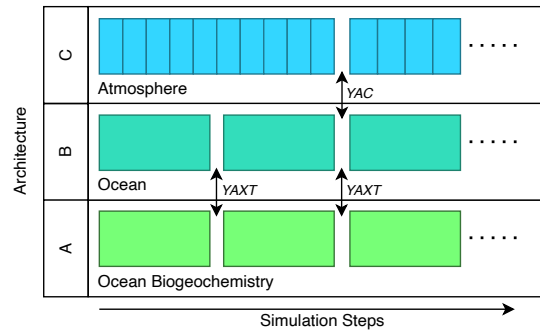


Figure 3: Coupling between the ICON atmosphere, ocean and ocean biogeochemistry components

3 Related Work

In the following, we provide an overview of previous efforts on leveraging heterogeneous HPC clusters and how they integrate into modern sites in Section 3.1. Finally, in Section 3.2, we discuss approaches and challenges for reliable tooling to improve performance and energy efficiency.

3.1 Heterogeneous High Performance Computing

Heterogeneous high-performance computing is not a new concept but has existed in various manifestations at least since the 1990s, when massively parallel systems on the one hand and Beowulf-type cluster systems on the other (initially still with homogeneous units) started a shift away from the previous era of monolithic supercomputers. With the resulting increase in using standard hardware for HPC and the independence of the parallel processors, the introduction of heterogeneity was an obvious and simple step for cost reduction and better scaling of specific applications. Further development steps in heterogeneous HPC were the coupling of different dedicated systems within supercomputing centers (*meta-computing*) [33] as well as across data center boundaries (*wide-area computing*) [7]. Similarly, in the field of cluster computing, heterogeneity developed from a local network of inhomogeneous computing resources (*heterogeneous clusters*), through the local combination of different cluster installations (*clusters-of-clusters*), to the transparent access of different cluster resources at different computing sites (*Grid computing*) [9, 10]. However, in recent years, heterogeneity in the presence of co-processors and accelerators such as GPGPUs has also become increasingly widespread at the node level, so that even homogeneous cluster systems of today usually feature this form of node-local heterogeneity. Previous efforts have also explored employing heterogeneity by using multiple HPC sites simultaneously, as detailed in [24]. In that work, the authors motivate their approach by the increased resources available when combining HPC systems, while also highlighting the performance implications of using specialized sites, for example due to the availability of GPUs. However, aspects of energy efficiency and further evaluations of the architectures used are not covered. In this paper, we propose the use of a heterogeneous cluster with purpose-chosen hardware, which we detail in the following chapters.

3.2 Performance and Energy Efficiency Analysis in Heterogeneous Platforms

There is a variety of interfaces and tools to monitor the performance and occupancy of heterogeneous compute resources. These interfaces are highly hardware dependent and implemented in the respective drivers. Drivers typically provide monitoring data in a proprietary way either through the device or through standardized interfaces, such as the Linux perf infrastructure [11, Chapter 13] or sysfs files. The kind of available monitoring data also depends on the hardware architecture and is not standardized. To support users in their monitoring efforts, hardware vendors also provide profilers for their hardware. This includes NVIDIA (Nsight Compute [6]), AMD (rocprof [3]), Intel (VTune [23]), and NEC (veprof [8]), which are vital tools to analyze and optimize performance. However, in the context of widely heterogeneous HPC, multiple additional demands have to be met.

First, information should be provided in a global context: Since parallelization paradigms such as the Message Passing Interface (MPI) run applications that can span a large number (e.g. thousands of) nodes, the gathered information needs to be merged and made available within the logical context of the application. Second, programs can run on different architectures simultaneously, e.g., a parallel program might use AMD *and* NVIDIA GPUs for acceleration, which is not supported by most vendor tools. Multiple HPC tools have been developed to fit these demands, such as Score-P [22], Caliper [5], HPCToolkit [2], and TAU [32]. We chose Score-P for our additions described in Section 4.3 to be able to measure heterogeneous ICON runs. Score-P furthermore supports a diverse set of architectures as well as measurement modes for recording traces, profiles and metrics [28].

While the optimization for runtime is generally considered to reduce the energy footprint of an application, it can also impair the energy efficiency, e.g., if the new, faster architecture has a much higher power demand. We therefore focus on *energy to solution* as a comparable metric across different architectures. For this, it is necessary to be able to (a) measure the power of the computing systems (b) measure it correctly and (c) make the power measurements between different systems comparable. While the first criterion can be achieved easily, given vendor-based power sensors, like Intel's RAPL [13] (which was adopted by AMD processors as well [30]), NVIDIA's NVML [36], and AMD's ROCM [4] interfaces, the second criterion is already problematic. Due to the limited (and sometimes not completely defined) scope of the sensors and timing accuracy, they are not necessarily suitable as a reliable source of information. Even worse, on some processors, these interfaces are likely based on models that introduce additional incoherency [13, 29, 30]. Alternatively, power measurements can be implemented by the system vendor [12, 19, 34] using additional onboard components. While this is preferable to using a model, the remaining properties (definition of sensor placement, timing information, accuracy information of used sensors ...) need to be accurate as well, but are often not well documented. A third option involves manual instrumentation, where power measurement devices are attached to instrumentation points within a computer or on its power inlet. Lastly, projecting the power consumption of a prototypical system to an assumed production cluster adds new limitations: While the

measurement of a node's power consumption is certainly a valid approach, power of components that will not be present in the final system needs to be subtracted. This may include hard drives or fans in the prototypical server, which will not be used in an integrated production cluster if the servers there are diskless and watercooled. To take all of this into account when comparing the efficiency of the architectures in our test clusters, we describe our power measurement and projection method in Section 4.4.

4 Design of a Widely Heterogeneous Test Cluster

When it comes to the advantages of heterogeneous cluster systems, the first thing to mention is the possibility of optimized resource utilization, since different hardware can selectively be used by suitable jobs, tasks and/or application parts. If such targeted task distribution can also help to reduce energy consumption remains to be proven. Furthermore, scalability and expandability are also key considerations, as new hardware types can be integrated into the system without breaking the assumption of a homogeneous hardware landscape.

Our goal when setting up the heterogeneous test cluster is to find a selection of different hardware platforms that are suitable to execute ICON and are also promising regarding their energy efficiency. In this section, we will therefore first discuss the hardware selection in detail in Section 4.1, before returning to middleware challenges and required adaptations in Section 4.2. Afterwards, we focus on performance measurements and, in particular, present the extensions made to the software tools for the analysis of the performance counters on the different hardware platforms in Section 4.3. This is followed by Section 4.4, where we describe energy measurement at the node/hardware level and discuss the measuring devices and metrics used for this purpose.

4.1 Hardware Overview

Our purpose-built cluster contains a representative selection of state-of-the-art hardware, widely found in recent HPC deployments. Beyond that, some legacy but very unique architectures are represented. Table 1 summarizes the equipped CPU and memory type, accelerators, if present, the operating system, and the compilers used. The energy efficiency of ICON running on a to be described heterogeneous architecture combination is compared to a homogeneous reference. The reference system is equipped with AMD Milan CPUs, which are part of the supercomputer Levante at Deutsches Klimarechenzentrum (DKRZ).

As shown in the table, a broad selection of different memory technologies is represented. In addition to regular DRAM, many recent systems include HBM memory or even rely on it exclusively. HBM memory offers higher bandwidth than regular DRAM, indicating that especially memory-bound code paths would profit from such memory. Over the years, the HBM standard has evolved, resulting in increased memory bandwidth, which is reflected in the various manufacturing periods represented in our cluster. Another development in the market is the introduction of MRDIMM. By using multiplexing, the bandwidth per module increases. As MRDIMM requires a processor that supports such memory, this feature goes alongside an updated Intel processor from the latest

Architecture	Host Processor	Host-accessible Memory	Accelerator	Operating System	Compiler
AMD Milan	2x AMD EPYC 7763	16x16 GiB DDR4, 3200 MHz	-	RHEL 8.10	Intel Classic 2021.10
AMD Genoa	2x AMD EPYC 9654	24x32 GiB DDR5, 4800 MHz	-	AlmaLinux 9.2	Intel Classic 2021.10
AMD Genoa-X	2x AMD EPYC 9684-X	24x32 GiB DDR5, 4800 MHz	-	AlmaLinux 9.2	Intel Classic 2021.10
Intel Sapphire Rapids	2x Intel Xeon Max 9468	16x16 GiB DDR5, 4800 MHz 8x16 GiB HBM2 3200 MHz	-	AlmaLinux 9.2	Intel Classic 2021.10
Intel Granite Rapids	2x Intel Xeon 6980P	24x32 GiB MRDIMM, 8800 MHz	-	AlmaLinux 9.5	Intel Classic 2021.10
NVIDIA Grace Grace	NVIDIA Grace CPU Superchip	240 GiB LPDDR5X	-	AlmaLinux 9.3	NVHPC 25.5
NVIDIA Grace Hopper	NVIDIA Grace (GH200)	480 GiB LPDDR5X, 96 GiB HBM3e	NVIDIA Hopper (GH200)	AlmaLinux 9.6	NVHPC 25.5
NEC SX-Aurora TSUBASA	Intel Xeon Gold 6126	6x16 GiB DDR4, 2666 MHz	NEC SX-Aurora Tsubasa VE10B 48 GiB HBM2	Rocky Linux 8.6	NEC 5.1.0

Table 1: Architectures evaluated for their energy efficiency in heterogeneous ICON experiments. Information about memory was gained via `hwinfo` and datasheets from NVIDIA and NEC.

product line. Whereas HBM memory, as of now, is no longer being pursued for future Intel processor product lines, the adoption of MRDIMM marks a shift in their product offerings. In addition to the listed CPU architectures, our cluster also includes integrated accelerators. It features NVIDIA Grace Hopper, which is widely used in many recent HPC deployments. Furthermore, a particularly unique component is a node equipped with NEC SX-Aurora vector processors. This type of accelerator is specifically designed to enhance computing tasks that rely on vectorization and is used for operational forecasting by the DWD using ICON.

4.2 Middleware Adaptations

As previously discussed, heterogeneity puts special demands on the middleware, especially if it should be able to span an MPI session of parallel processes across different node types and when exchanging messages between them via a high-performance network. However, since its origins in the 1990s, the design of the MPI standard [25, 26] has always taken also the possibility of heterogeneity into account, for example, the possibility to exchange messages in the form of explicitly declared data types, so that the respective MPI library can perform transparently any necessary data representation conversions (for example, between little and big endian). Other library-related aspects include support for different instruction set architectures, as well as support for various network technologies, memory models, and/or potential accelerators.

Fortunately, all host architectures selected for the test cluster have more or less standard instruction sets (x86-64 or ARM, and thus all operating in little endian), so no major problems were to be expected in terms of support by the MPI library here. If that had been the case, or if such a scenario arises in the future, an MPI stack with heterogeneous capabilities would be required. This is where ParaStation MPI comes into play, as it natively provides this functionality.

ParaStation MPI library is inherently well suited as an MPI provider for the deployment in heterogeneous environments. For instance, ParaStation MPI also supports the simultaneous use of different network technologies and the respective routing of messages through such heterogeneous networks. However, this feature did not play a role for this effort, as the heterogeneous nodes in the test cluster are all part of the same InfiniBand network. Nevertheless, it should be mentioned that if a hardware platform did not support InfiniBand and/or the required drivers, it would have been possible

to fall back to Ethernet and TCP for this specific connection. In this way, such a platform can still be integrated transparently into the primarily InfiniBand-based MPI sessions across the test cluster.

4.3 Performance Measurements

As introduced in Section 3.2, specially adapted performance analysis tools are needed for an efficient profiling of heterogeneous applications. Supporting performance analysis in heterogeneous systems comes with two key challenges: First, supporting the analysis of the individual architectures of the heterogeneous cluster in isolation. Secondly, combining these independent analysis data streams from a distributed, heterogeneous application run into a coherent global view. A tool matching the first criterion could provide a comprehensive analysis framework for portable analysis workflows, where an analysis workflow for architecture A could be easily applied to architecture B. On the basis of architecture support from the first criterion, a tool that is able to satisfy the second criterion could be built that supports the holistic analysis of heterogeneous HPC applications. For our use-case, the architectures in Section 4.1 are relevant to the first criterion. To the best of our knowledge, no tool exists that supports the analysis of all the listed architectures. Also, no clear workflow exists for any of the listed tools that enables out-of-the-box analysis of heterogeneous runs as a whole, as required by the second criterion.

Given that no pre-existing tool can satisfy the criteria, we chose to extend already existing tools to better match them. On the basis of these tools, we then created a heterogeneous performance analysis workflow for our use-case. The tools we chose to extend are Score-P [22] and lo2s [18]. Score-P is a unified performance analysis architecture for analyzing multi-node applications. It is well suited for the analysis of large runs of distributed scientific applications, such as the ICON model that is of interest here. Use of Score-P in a program-under-test is primarily enabled by recompiling the program-under-test using the `scorep- $\$$ COMPILER` compiler wrapper for the given target language. This compiler wrapper instruments the program-under-test according to the options supplied to `scorep- $\$$ COMPILER`. In this instrumentation, Score-P can make use of interfaces such as compiler, MPI (through PMPI) and OpenMP (through OMPT) instrumentation.

Regarding the first criterion, Score-P already provides support for all CPU and GPU architectures listed in Section 4.1. The only

architecture not currently covered by Score-P was the NEC SX-Aurora. We have detailed our adaptations to Score-P in [35]. The NEC SX-Aurora, despite being a PCIe attached accelerator card, is in its programming model used more like a standalone computing system. We implemented support for the NEC SX-Aurora as a host architecture by using its interfaces for compiler, MPI, and OpenMP instrumentation. Our porting effort was helped by the "cross compiling" mode of Score-P. In this mode, frontend tools such as `scorep` are compiled for the (x86) host architecture. The only part of Score-P that is actually compiled for the NEC SX-Aurora is the one that interfaces directly with the architecture's performance analysis facilities, such as compiler instrumentation. This approach greatly reduced the footprint of the code that needed porting. We are currently in the process of upstreaming this NEC SX-Aurora support to Score-P. With the first criterion covered, we moved to make the support for the different architectures interoperable in the same heterogeneous run. In the case of Score-P, this is challenging, not only due to the heterogeneity of the different architectures involved, but as it uses compile time instrumentation – with different compilers needed for different platforms. For ICON, the most efficient compiler for any architecture is most often the one supplied by its vendor – for accelerator support, usage of the vendor compiler is even required. In principle, this should not be a major issue, as the instrumentation building blocks of Score-P are largely built on platform-independent interfaces, such as `PMPI`, `OMPT`, or `cyg_profile_func`. In practice, however, small differences in each instrumented application can lead to major issues, especially as Score-P, and similar analysis tools, were created for a world of homogeneous supercomputers. Such a homogeneous-centric implementation will not always take the required care to ensure that the code paths for different architectures are compatible. For example, on x86 systems, the hardware `tsc` clock source is used, while on ARM systems a `gettimeofday` based mechanism is used by default. If a Score-P run combines x86 and ARM nodes, this by default results in a dead-lock. This specific dead-lock is caused by the fact that, the x86 `tsc` based time-stamp mechanism synchronizes its clock across all nodes through the use of collective MPI communications, while the `gettimeofday` based implementation of the ARM side does not do such a synchronization. This leaves MPI operations on the x86 side with no communication partner on the ARM side, resulting in the dead-lock. The solution, as shown in the Score-P documentation, is to explicitly set the clock-source for all processes to `gettimeofday`. Beyond this specific issue, we took care to explicitly set any of the Score-P parameters to the same value, to avoid further interface mismatches.

Besides Score-P, we extended `lo2s` [18] to be usable as a performance analysis tool in our heterogeneous setup. `lo2s` is a system analysis tool, providing information about cross-cutting interactions between software, operating system and hardware, while Score-P has a strong software-centric view on applications. As this, `lo2s` can provide further insights into ICONs execution, as it shows how ICON interacts with the operating system and hardware. For its purpose, `lo2s` makes use of operating system and hardware interfaces, such as hardware sampling and tracepoints. Unlike the software instrumentation sources on which Score-P depends, this approach does not require changes to the program- or system-under-test.

To satisfy the first criterion for heterogeneous performance analysis tools, we implemented support for recording both NVIDIA and AMD GPU activity. With regard to the remaining architectures listed in Section 4.1, `lo2s` already supports recording per-CPU timelines [18] for all listed CPU architectures. Our addition of NEC SX-Aurora support, detailed in [35], is primarily based on a vendor-provided library that enables recording NEC SX-Aurora program counters and performance monitor counters from the host system side. Furthermore, to provide more useful insights for HPC applications, we created interfaces to OpenMPs OpenMP Tool (OMPT) interface. The implementation challenge in this is that the interfaces provided for the analysis of NVIDIA and AMD GPUs as well as the interface provided for OpenMP analysis are based on callbacks. In this architecture, whenever an event-of-interest happens in the application, as for example the start of a GPU kernel, the application calls a callback that we previously registered with it. As `lo2s` runs as a separate process, there is no way for the program-under-test to call a function inside `lo2s`. The solution we developed for this problem is composed of two parts: First, we created a stub library, which is loaded into the program-under-test via mechanisms such as the `OMPT_OMP_TOOL_LIBRARIES` environment variable. Inside the program-under-test, this stub library registers itself for the events in the application that we want to monitor. Secondly, it sends `lo2s` the file descriptor to a piece of shared memory created with `memfd_create` via an Unix domain socket. When an event occurs, the stub library then writes the event to this shared memory, from which `lo2s` can then retrieve and record it. This not only allows us to specifically extend `lo2s` perspective towards accelerators, but also bridges the gap between `lo2s`' purely program-under-test-external implementation and program-under-test-internal measurement interfaces. As `lo2s` is by design a single-node analysis tool, the second criterion of being able to combine data streams from a heterogeneous multi-node execution does not apply to it. However, using the `Vampir` tool [21], multiple traces can be opened in a comparison view to analyze concurrent execution of multiple nodes. All `lo2s` development detailed here has been upstreamed² and is available in the 1.8.0 release.

With these performance analysis tools in place, it is possible for us to analyze runs of ICON with Score-P that combine, for example, usage of the CPU-only Intel Sapphire Rapids Node (using the Intel Classic Compiler) and the CPU+GPU Grace Hopper Superchip Node (using the NVHPC compiler) as listed in Section 4.1. Using the HPC/Score-P topdown methodology developed and successfully demonstrated on homogeneous ICON runs in [27], this extends the analysis workflow developed therein to heterogeneous applications. Furthermore, Score-P makes it easy to detect imbalances in the execution of heterogeneous ICON runs, as such imbalances will show up clearly in the resulting trace: In an ICON/Score-P run that executes ICON heterogeneously with the same core count spread over an Intel Sapphire Rapids and an Grace Hopper node, the resulting trace easily shows that the Sapphire Rapids node is heavily underutilized as its ICON processes spend the majority of their time waiting on MPI communication from the Grace Hopper side. We have used `lo2s`, on the other hand, for analysing application operating-system interactions of ICON runs. For example, `lo2s`

²<https://github.com/tud-zih-energy/lo2s>

can record file I/O operations on multiple levels of the system I/O stack. With this, we can tell whether the time cost of some file operation is due to time spent in ICON or library code, the operating system or in the underlying hardware, providing further pointers for optimization of ICONs file operations.

4.4 Energy Measurement

Accurate per-node energy measurements are essential for assessing overall efficiency. Choosing between on-board (e.g. RAPL) and out-of-band measurements is challenging: on-board data varies by vendor and architecture and may omit or include parts like memory, fans or interconnect. We therefore use energy counters of the PDUs (Raritan PX2-5530) at node level, providing a 1% accuracy that was additionally validated using LMG450 high precision powermeters. For the challenging DC power measurements of PWM controlled motors, we employ a SIGLENT SDS3054X 12bit 500MHz oscilloscope in combination with DP700 voltage and CP503B current probes. Due to uncertainty propagation, the overall power measurement accuracy of this setup is in the range of $\sim 2.3\%$.

However, these out-of-band methods also include fan power, which limits their transferability to water-cooled production systems. As the test cluster is expected to expand, subtracting air-cooling power from node measurements is expected to provide a more scalable alternative to evaluating board-level energy or power metrics per platform. In absence of power sensors for all fans within a chassis, we exploit the fact that fan speed is tightly monitored by the BMC and commonly available via IPMI and create a projection of fan speed to fan power consumption, which we will use in our subsequent experiments.

The projection method is based on measurements of fans at certain speeds and their measured power consumption. However, a direct instrumentation was not always possible, since the fan speed cannot always be manually set and the cabling from mainboard to the fan cannot always be temporarily replaced to connect measurement devices. We therefore use three different approaches to collect our fan power measurements:

- If fan speed can be set and fans can be instrumented, we employ custom cabling to hook into the connections between mainboard and fans and derive fan power from direct voltage and current measurements.
- If fan speed can be set, but proprietary cables hamper a non-destructive connection, we model fan power from overall node power and sweep through various speed permutations of all fans in the chassis. The fan power function for n fans (1) is then fit to all measurements using non-linear least squares. For this approach it is crucial to keep the remaining node idle in order to maintain a constant idle offset power.
- If the fan speed can not be set manually, we move fans into an external setup with an external power supply and generate the PWM control signal on a microcontroller.

Being closest to normal operation, the first option is considered the most accurate. The second is the least invasive but less accurate, while the last one is electrically accurate but misses the potential impact of a different airflow inside the chassis. For most platforms in the project, we employed the direct measurement method. In

case of the Granite Rapids architecture, we relied on the modeling approach, and for Genoa-X, we needed a bench setup.

We note, that the actual fan power functions are highly hardware and site specific, and report the results for the AMD Genoa system exemplarily in Figure 4.

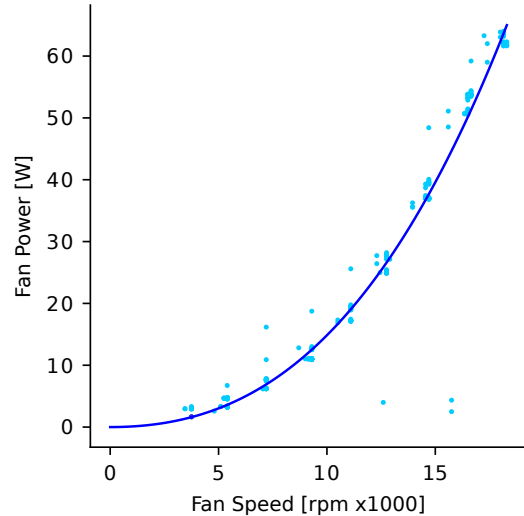


Figure 4: AMD Genoa fan power model. Computed fan power metric derived from direct measurements during fan speed sweep.

The system was evaluated using direct voltage and current measurements resulting in parameters $a = 0.01$, $b = 0.09$ for a fan power according to Equation (1).

$$P_{node} = P_{fans} + P_{idle} = \sum_{f=1}^n (ax_f^3 + bx_f^2) + P_{idle} \quad (1)$$

Lastly, all measured and computed metrics are aggregated using the MetricQ [17] framework, allowing for straightforward access from test scripts. In order to minimize aliasing effects, node energy is recorded after masking init and warmup phases and running experiments for a reasonably long time. Fan power from fan speed is sampled every 0.5s, averaged and subtracted from node energy. Since each benchmark is designed to optimally utilize a node, the energy measurements need to be further normalized to represent efficiency relative to a comparable amount of computational work. With this measurement infrastructure in place, the evaluation of energy efficiency has been largely automated, allowing for reproducible and continuous evaluation of hardware.

5 Performance and Efficiency Analysis of ICON

In the following, the performance and energy efficiency of ICON are analyzed. To this end, Section 5.1 provides details about the investigated experiment and its relevance to production runs. Subsequently, Section 5.2 outlines the metrics used in the evaluation. After introducing the experiment and describing how its performance is assessed on heterogeneous architectures, we proceed with the actual evaluation, which is a threefold process. We distinguish

between extracted relevant code paths, referred to as kernels hereafter, and independently runnable parts of ICON, which we refer to as components. In Section 5.3, the extracted kernels that make up ICON experiments are introduced. Furthermore, their individual energy efficiency in our test cluster is evaluated. Afterwards, in Section 5.4, components that make up ICON experiments are highlighted and evaluated for their most energy-efficient architecture. Building on these insights, in Section 5.5 a heterogeneous ICON experiment is assembled and evaluated.

5.1 Experiment design

The experiment design leverages some properties of climate simulations, which we want to introduce in this section. First of all, we neglect initialization time and energy in our measurements, since for climate simulations this is small compared to the runtime doing actual simulation. The smallest unit representative for all computational processes in the atmosphere is a radiation time step which takes few ten seconds to minutes in production simulations, depending on resources and resolution. During this, all other processes are called a number of times. In theory, simulating only one radiation time step would therefore be sufficient, but to obtain better statistics on performance and energy consumption, we usually run for at least one minute, which does not include the initialization and warmup iterations. Additionally, each measurement is repeated three times which allows for averaging the results.

In this paper, the atmosphere is configured with a 40 km resolution and the ocean with a 20 km resolution. The experiment itself is a down-scaled version of the Cycle 4 experiment in [31] employing a 10km atmosphere and a 5km ocean. While this downscaling keeps the computational effort per node constant compared to the original run, physical processes are no longer represented well which is acceptable for technical experiments. We argue that the changes in communication due downscaling do not have a significant impact since weak and strong scaling of ICON is shown to be good in [14, 20]. Consequently, our setup is a good representation of a high resolution production setup.

5.2 Key metrics

For this paper we focus on two key metrics: time per step and energy per step. In the Earth system simulation community throughput (sometimes also called time compression τ) is measured in simulated years (or days) per day of wall clock time. This metric is advantageous compared to pure wall clock time, since it is independent of the simulated timespan, which may vary from a few days when studying microphysics at high resolution to centuries when studying global climate evolution. Throughput is also the metric domain scientists are most interested in, since they want to make the best use of their computational resources and are interested in obtaining results fast. In this paper we are using wall-clock time per simulated time step or time per step which is a slightly different metric. The two differences are that time per step is inversely proportional to throughput and that it neglects the size of each time step Δt which is constant throughout our experiments.

The second key metric is energy per step, which captures energy efficiency in a manner analogous to time per step. This metric optimizes energy consumption, but in many cases it also optimizes

runtime since consumption for many architectures is not highly load dependant. The latter argument breaks down when comparing architectures, as a particular architecture may simultaneously exhibit both low consumption and low throughput, resulting in a good energy efficiency and unacceptable turnover. For this reason, we always report both: throughput and energy consumption per simulated time, with a primary focus on energy.

5.3 Kernels

ICON is capable of simulating a wide range of weather and climate applications. To assess the applicability of heterogeneous computing architectures, we extracted a representative selection of compute-intensive code paths. In this context, kernels do not resemble those found, for example, in GPUs, but instead represent a much coarser level of granularity, often referred to as dwarfs. This allows for an independent evaluation of relevant code paths that occur in widely used ICON experiments. Kernels enable us to observe relevant code paths in isolation of other interacting aspects of an experiment, which helps with tool development and reduces complexity for benchmarking.

We show the investigated kernels in Figure 5 and discuss their significance within ICON in the following. The kernel denoted as non-hydro atmosphere represents the non-hydrostatic part of the atmospheric model. As can be seen, this part contributes significantly to the runtime of a typical ICON experiment. Another part of the atmosphere is computation of radiative fluxes, which represents the simulation of radiation in numerical weather prediction applications. This is referred to as radiation in the chart. While radiation *can* consume a large portion of computational resources, this depends on the configuration of the experiment and varies. In the baseline experiment used as a reference for kernel extraction, radiation effects are computed only at every fifth timestep. Another set of kernels represents parts of the ocean. Shown are the vertical velocity advection (vert. ocean adv.) and the horizontal velocity advection (horiz. ocean adv.) within the ocean. These, along with other physical processes, represent the computational portion of the ocean model. The remaining 8% represent runtime contributing code, that is not covered by the investigated kernels.

The performance and energy efficiency depends on several factors that need to be considered during benchmarking. As with

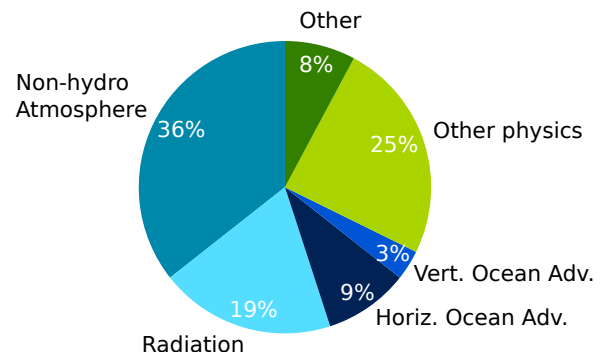


Figure 5: ICON kernel runtime. Captured independent kernels contribution to overall ICON runtime.

most HPC applications, the number of participating MPI ranks and OpenMP threads affects performance. Furthermore, ICON-specific tuning parameters, such as the loop chunk length has a significant impact on performance. CPU architectures can benefit from this parameter by exploiting vectorization, while GPUs depend on it to utilize a sufficient number of threads. In ICON, this is referred to as `nproma`, and its value is architecture-dependent. On CPU architectures, this value ranges from 8 to 32 and on NEC SX-Aurora, 752 has proven to be a good value used in production. On GPUs, however, `nproma` should be set to the number of cells handled by the corresponding MPI rank. Overall, it should be noted that a wide range of possible tuning parameters exists. By employing automated and parameterized benchmarking, we are able to account for this variability, and invested considerable effort to ensure that each architecture is evaluated fairly. In terms of optimizing the architectures, we followed the recommendations of the vendors.

Figure 6 shows the energy consumption of the investigated architectures for running the selected kernels. Considering the tuning options described above, the most energy-efficient configurations for each kernel on each architecture are visualized. The amount of performance engineering invested in each component through previous porting efforts varies across ICON. However, despite differing levels of optimization, there are fundamental characteristics of the simulations that make them more or less suitable for certain architectures. At the time of writing, only the kernels originating from the atmosphere are sufficiently mature in terms of GPU performance portability to be included in this comparison. For the atmospheric non-hydro kernel, NEC SX-Aurora demonstrates the highest energy efficiency, closely followed by Grace Hopper. For this specific kernel, there is a significant gap compared to the CPU architectures. The closest contenders are Intel Sapphire Rapids with HBM memory and AMD Genoa-X, which features large L3 caches. This indicates a memory-bound kernel, which was also confirmed by roofline modeling. In contrast to the excellent performance of the NEC SX-Aurora architecture for the non-hydro atmosphere kernel, its performance in computing radiation in the atmosphere is the lowest among the evaluated architectures. This highlights, that no single architecture achieves the highest energy efficiency across all investigated kernels, reinforcing the importance of relying on a heterogeneous architecture. For the vertical and horizontal advection in the ocean, as well as diffusion, the Grace and Genoa-X CPU architectures perform best. The evaluation of these kernels in isolation lacks interaction with other code paths. Nevertheless, they give guidelines which architectures are promising when designing an energy efficient heterogeneous setup as we do in the next section.

5.4 Energy Efficiency of ICON Components

Kernels, as described above, provide a much finer granularity than what is practically achievable with the reference application. This is due to data dependencies impeding lightweight communication between architectures that might be most energy-efficient for a specific computation task. However, ICON implements several of its capabilities in the form of components, which naturally correspond to distinct physical processes, which also reduces data movement between them. In the following, the atmosphere, ocean, and ocean

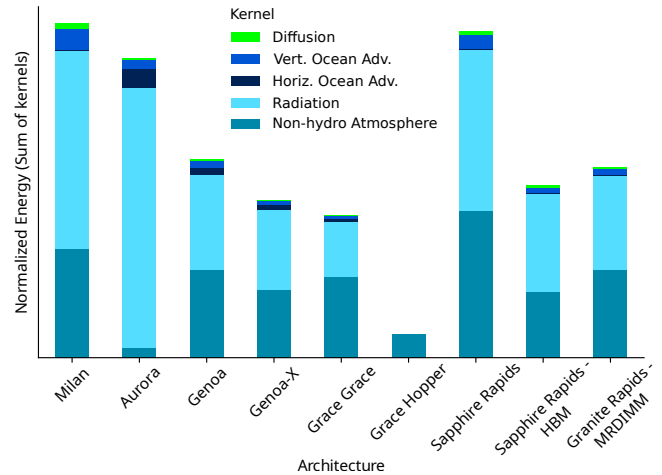


Figure 6: ICON kernel energy. Efficiency of kernel execution on different architectures, normalized to one simulation step.

biogeochemistry components are each evaluated for their most energy-efficient compute architecture. More specifically, the atmosphere is modeled using the AES physics package and the ocean biogeochemistry is provided by the Hamburg Model about Ocean Carbon Cycle (HAMOCC). The atmosphere and ocean components can be evaluated completely independently of one another, as they do not rely on other components. In production, they communicate through a coupling mechanism that exchanges relevant variables at defined intervals. Coupling also has the added benefit that it allows to combine components that compute on different grid resolutions, which thereby provides means of influencing the computational workload. In contrast to these components, the HAMOCC component runs concurrently with the ocean component, exchanging variables required for its execution. To evaluate HAMOCC independently, as done for the other components, an ocean component was executed on a separate node to provide the necessary data. Since the ocean component requires fewer resources than HAMOCC, this setup allows the HAMOCC component to be evaluated without side effects.

The energy and runtime per step, as defined in Section 5.2, of each investigated architecture for each component are shown in Figure 7. For each component, the measurements have been normalized relative to the reference architecture, which serves as the basis for the homogeneous runs. A value outside the dashed line indicates an architecture that performs worse compared to the AMD Milan reference architecture, while lower values represent more energy-efficient or performant architectures. For the atmosphere, NVIDIA Grace Hopper outperforms all other architectures in terms of energy efficiency by a large margin. It is followed by AMD's Genoa-X, which demonstrates higher efficiency than AMD Genoa, presumably due to its larger L3 cache. Across the AMD lineup, the investigated ICON components consistently profit from that larger L3 cache. However, this influence is most noticeable in the atmospheric component. Across all components, Intel Sapphire Rapids with DRAM performs worse than when relying on HBM memory. Without HBM, its energy efficiency is even lower than

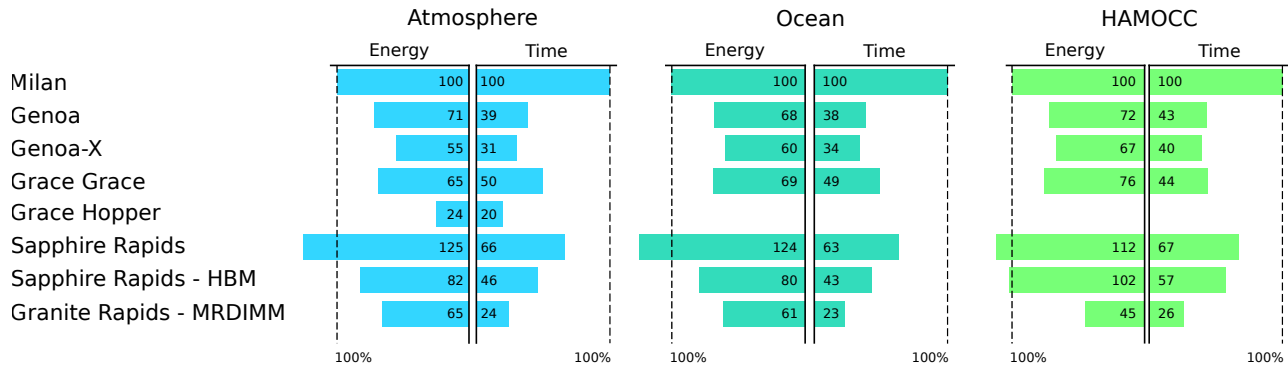


Figure 7: ICON component runtime and energy efficiency. Shown are the investigated ICON components, evaluated for their energy and runtime per step. Each result is normalized to the AMD Milan reference architecture, with smaller values indicating better performance.

that of the older AMD Milan architecture. The impact of HBM on the performance of Intel Sapphire Rapids is particularly evident for the atmosphere and ocean components, but less pronounced for the HAMOCC component. NVIDIA Grace Grace performs very similar to AMD Genoa, which once again shows that ARM architectures are competitive. Intel Granite Rapids, equipped with MRDIMM, performs similarly to AMD Genoa for the ocean component but outperforms all other architectures for the HAMOCC component. NEC SX-Aurora is not part of this component evaluation. This is due to challenges in incorporating this architecture into the chosen experiment configuration because of its memory size. As we describe in Section 5.3, NVIDIA Grace Hopper is currently only competitive for the atmosphere due to varying performance portability, and has therefore not been considered for the other components. Throughout the measurements, we observe a general trend of performant architectures also being most energy-efficient.

This evaluation shows that achievable energy efficiency varies across models on a given hardware architecture. Furthermore, selecting a single architecture is not a viable option when aiming to maximize the energy efficiency in a coupled ICON experiment. Comparing this component evaluation with the kernel evaluation in Section 5.3 shows that less complex kernels can provide valuable guidance when selecting promising architectures. Finally, by employing a heterogeneous architecture and leveraging ICON’s coupling functionality, the overall energy consumption can be reduced. In this case, the evaluation suggests a heterogeneous configuration in which the atmosphere runs on NVIDIA Grace Hopper, while the ocean and HAMOCC components run on Intel Granite Rapids with MRDIMM.

5.5 Performance and Efficiency of a Heterogeneous Execution

In the previous section, the ICON components that make up the experiment described in Section 5.1 were evaluated in isolation. The results allow us to infer how these components can be distributed in a heterogeneous deployment. In this section, we evaluate our representative ICON experiment running on the homogeneous reference system and compare it with a heterogeneous deployment based on the insights from the preceding evaluation. When making such a

comparison, it is important to balance the used compute resources. We aim to use as few nodes of the homogeneous reference system as possible while maintaining the same experimental throughput as the heterogeneous system. This ensures that no artificial advantage is introduced, for example, by unfairly increasing the communication overhead on the homogeneous system. Balancing compute resources for ICON experiments involves analyzing the integrated ICON timers in combination with adjusting the number of MPI ranks assigned to a specific component between balancing runs. When balancing resources across three components, it is helpful to begin with only two components initially. Running only the atmosphere together with the ocean significantly reduces the number of moving parts. Once the ICON timers indicate that these two components are not wasting time by waiting on one another, the third component can be re-enabled. Using this balancing approach we determined that five homogeneous reference nodes are needed to achieve the same throughput as the heterogeneous system. As suggested in the previous section, we consider a heterogeneous cluster combining NVIDIA Grace Hopper and Intel Granite Rapids with MRDIMM the most energy-efficient configuration for the described ICON setup. During execution, the heterogeneous cluster consumes ≈ 2.05 MJ of energy. In comparison, the homogeneous setup on the reference system consumes ≈ 3.25 MJ of energy. This equates to energy savings of about 40% with comparable throughput.

6 Conclusion and outlook

In this paper, we have shown that significant energy savings can be achieved by exploiting heterogeneity. This is accomplished by leveraging the strengths of multiple compute architectures simultaneously, with each architecture computing only the model component for which it is most energy-efficient. To evaluate architectures for energy efficiency, easy-to-use and reliable measurement tools are required. This is especially true for heterogeneous applications, as Section 5.5 showed novel performance engineering challenges arising from the use of heterogeneous systems. Contributions in this area include extending architectural support in existing performance analysis tools and improving support for heterogeneous clusters. One special concern in improving tools for heterogeneous

systems is usability: As shown in Section 4.3, analysing heterogeneous systems can give rise to a range of both overt and subtle challenges that complicate the collection of performance data. The usability of performance analysis tools for heterogeneous systems therefore remains an ongoing concern, as does keeping pace with the continued development of new architectures and their associated performance analysis interfaces. Accordingly, future work includes efforts to improve the usability of tools for measuring performance and energy consumption, with the aim of making simultaneous measurements across heterogeneous systems with multiple host and accelerator architectures as straightforward as those for homogeneous systems.

ICON's native support for disaggregating model components through coupling mechanisms enables heterogeneous deployments in various ways. Beyond the components investigated in this paper, additional components would further expand the potential for employing heterogeneous architectures. Together with continually improving architectures and the introduction of new types of hardware, fully exploiting heterogeneity remains an ongoing process. This requires continuous efforts to extend tool support and improve ICON with respect to performance portability, with several initiatives already underway.

The energy savings described in this paper come with the trade-off of extensive initial measurements. While this overhead is justified by the potential benefits in production environments, it can be reduced by narrowing the range of parameters that need to be explored. For this, previous generations can serve as a baseline, but detailed measurement remain necessary. The same holds for our approach of measuring kernels on their own. These can serve as a guideline which combinations of architectures are most interesting, but still a number of measurements are necessary. Consequently, developing the necessary tooling and methodologies to enable thorough and efficient evaluation of new architectures is fundamental.

Acknowledgments

This work was funded by the German Federal Ministry of Research, Technology and Space via the EEChIPs project under grant no. 16ME0599K and 16ME0602. It is also financed on the basis of the budget passed by the Saxon State Parliament in Germany. We would like to thank Stéphan Jauré from Eviden for providing the ICON kernels used in this work.

References

- [1] Mario Acosta, S. Palomas, and Etienne Tourigny. 2023. Balancing EC-Earth3 Improving the Performance of EC-Earth CMIP6 Configurations by Minimizing the Coupling Cost. *Earth and Space Science* 10, 8 (2023), e2023EA002912. doi:10.1029/2023EA002912 arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2023EA002912 e2023EA002912 2023EA002912.
- [2] Laksono Adhianto, S Banerjee, M. Fagan, M Krentel, Gabriel Marin, John Mellor-Crummey, and Nathan Tallent. 2010. HPCTOOLKIT: tools for performance analysis of optimized parallel programs. *Concurrency and Computation: Practice and Experience* 22, 6 (2010), 685–701. doi:10.1002/cpe.1553 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.1553
- [3] Inc. Advanced Micro Devices. 2025. AMD Instinct MI300 X Workload Optimization – ROCProfiler. https://rocm.docs.amd.com/en/latest/how-to/rocm-for-ai/inference-optimization/workload.html#mi300x-rocprof. Accessed: 2025-11-10.
- [4] Inc. Advanced Micro Devices. 2025. rocm_smi_lib: ROCm System Management Interface Library (GitHub Repository). https://github.com/RadeonOpenCompute/rocm_smi_lib. Accessed: 2025-11-10.
- [5] David Boehme, Todd Gamblin, David Beckingsale, Peer-Timo Bremer, Alfredo Gimenez, Matthew LeGendre, Olga Pearce, and Martin Schulz. 2016. Caliper: Performance Introspection for HPC Software Stacks. In *SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 550–560. doi:10.1109/SC.2016.46
- [6] NVIDIA Corporation. 2022. NVIDIA Nsight™ Compute v2022.3: An Interactive Kernel Profiler for CUDA® Applications. https://developer.nvidia.com/nsight-compute-2022_3. Accessed: 2025-11-10.
- [7] Thomas A. DeFanti, Ian Foster, Michael E. Papka, Rick Stevens, and Tom Kuhfuss. 1996. Overview of the I-WAY: Wide-Area Visual Supercomputing. *International Journal of High Performance Computing Applications* 10, 2–3 (1996), 123–131. doi:10.1177/109434209601000201
- [8] Erich Focht. 2024. Using veprof – external profiler for VE programs on NEC SX-Aurora TSUBASA. Blog post at SX-Aurora Developer Blog. Available at https://sx-aurora.github.io/posts/Testing-VEOS-DMA-profi/ (accessed 2025-11-10).
- [9] Ian Foster and Carl Kesselman (Eds.). 1999. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, San Francisco, CA.
- [10] Ian Foster and Carl Kesselman (Eds.). 2004. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, Amsterdam; Boston.
- [11] Brendan Gregg. 2020. *Systems Performance: Second Edition*. Pearson (Addison-Wesley), Boston, MA.
- [12] Daniel Hackenberg, Thomas Ilsche, Joseph Schuchart, Robert Schöne, Wolfgang E. Nagel, Marc Simon, and Yiannis Georgiou. 2014. HDEEM: High Definition Energy Efficiency Monitoring. In *2014 Energy Efficient Supercomputing Workshop*. 1–10. doi:10.1109/E2SC.2014.13
- [13] Daniel Hackenberg, Thomas Ilsche, Robert Schöne, Daniel Molka, Maik Schmidt, and Wolfgang E. Nagel. 2013. Power measurement techniques on standard compute nodes: A quantitative comparison. In *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. 194–204. doi:10.1109/ISPASS.2013.6557170
- [14] Ioan Hadade, Daniel Klocke, Jussi Enkovaara, Tuomas Lunttila, Thomas Rackow, Jan Frederik Engels, Claudia Frauen, René Redler, Jenni Kontkanen, Thomas Jung, Dmitry Sein, Irina Sandu, Balthasar Reuter, Nils Wedi, Sebastian Milinski, Francisco Doblas-Reyes, Miguel Castrillo, Mario Acosta, Sergi Girona, and Pekka Manninen. 2025. Destination Earth: The Climate Change Adaptation Digital Twin. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '25)*. Association for Computing Machinery, New York, NY, USA, 99–110. doi:10.1145/3712285.3771790
- [15] Moritz Hanke, René Redler, Teresa Holfeld, and Maxim Yastremsky. 2016. YAC 1.2.0: new aspects for coupling software in Earth system modelling. *Geoscientific Model Development* 9, 8 (2016), 2755–2769. doi:10.5194/gmd-9-2755-2016
- [16] Cathy Hohenegger, Peter Korn, Leonidas Linardakis, René Redler, Reiner Schnur, Panagiotis Adamidis, Jiawei Bao, Swantje Bastin, Milad Behraves, Martin Berge-mann, Joachim Biercamp, Hendryk Bockelmann, Renate Brokopf, Nils Brüggemann, Lucas Casaroli, Fatemeh Chegini, George Datsaris, Monika Esch, Geet George, and Bjorn Stevens. 2023. ICON-Sapphire: simulating the components of the Earth system and their interactions at kilometer and subkilometer scales. *Geoscientific Model Development* 16 (01 2023), 779–811. doi:10.5194/gmd-16-779-2023
- [17] Thomas Ilsche, Daniel Hackenberg, Robert Schöne, Mario Bielert, Franz Höpfner, and Wolfgang E. Nagel. 2019. MetricQ: A Scalable Infrastructure for Processing High-Resolution Time Series Data. In *2019 IEEE/ACM Industry/University Joint International Workshop on Data-center Automation, Analytics, and Control (DAAC)*. 7–12. doi:10.1109/DAAC49578.2019.00007
- [18] Thomas Ilsche, Robert Schöne, Mario Bielert, Andreas Gocht, and Daniel Hackenberg. 2017. lo2s – Multi-core System and Application Performance Analysis for Linux. In *2017 IEEE International Conference on Cluster Computing (CLUSTER)*. 801–804. doi:10.1109/CLUSTER.2017.116
- [19] Matthew Kappel, Jens Doleschal, Thomas Ilsche, Mario Bielert, Alistair Hart, and Harvey Richardson. 2014. User-level Power Monitoring and Application Performance on Cray XC30 Supercomputers. In *Proceedings of the Cray User Group (CUG) Conference 2014, Lugano, Switzerland*. Paper 136, available at https://cug.org/proceedings/cug2014_proceedings/includes/files/pap136.pdf.
- [20] Daniel Klocke, Claudia Frauen, Jan Frederik Engels, Dmitry Alexeev, René Redler, Reiner Schnur, Helmuth Haak, Luis Kornblueh, Nils Brüggemann, Fatemeh Chegini, Manolo Römmer, Lars Hoffmann, Sabine Griessbach, Mathis Bode, Jonathan Coles, Miguel Gila, William Sawyer, Alexandru Calotou, Yakup Budanaz, Pratyai Mazumder, Marcin Copik, Benjamin Weber, Andreas Herten, Hendryk Bockelmann, Torsten Hoeffler, Cathy Hohenegger, and Bjorn Stevens. 2025. Computing the Full Earth System at 1km Resolution. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '25)*. Association for Computing Machinery, New York, NY, USA, 125–136. doi:10.1145/3712285.3771789
- [21] Andreas Knüpfer, Holger Brunst, Jens Doleschal, Matthias Jurenz, Matthias Lieber, Holger Mickler, Matthias S Müller, and Wolfgang E Nagel. 2008. The vampire performance analysis tool-set. In *Tools for High Performance Computing: Proceedings of the 2nd International Workshop on Parallel Tools for High Performance Computing, July 2008, HLRS, Stuttgart*. https://doi.org/10.1007/978-3-540-68564-7_9

- [22] Andreas Knüpfer, Christian Rössel, Dieter an Mey, Scott Biersdorff, Kai Diethelm, Dominic Eschweiler, Markus Geimer, Michael Gerndt, Daniel Lorenz, Allen Malony, Wolfgang E. Nagel, Yury Oleyunik, Peter Philippen, Pavel Saviankou, Dirk Schmidl, Sameer Shende, Ronny Tschüter, Michael Wagner, Bert Wesarg, and Felix Wolf. 2012. Score-P: A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir. In *Tools for High Performance Computing 2011*. 79–91. doi:10.1007/978-3-642-31476-6_7
- [23] Alexandr Kurylev and Vladimir Tsymlal. 2021. Profiling Heterogeneous Computing Performance with Intel® VTune™ Profiler. In *Proceedings of the 2021 ACM/IEEE International Conference on Performance Engineering (ICPE) Companion*. doi:10.1145/3456669.3456678
- [24] Jason Maassen, Ben van Werkhoven, Maarten van Meersbergen, Henri E. Bal, Michael Kliphuis, Sandra E. Brunnabend, Henk A. Dijkstra, Gerben van Malenstein, Migiel de Vos, Sylvia Kuijpers, Sander Boele, Jules Wolfrat, Nick Hill, David Wallom, Christian Grimm, Dieter Kranzlmüller, Dinesh Ganpathi, Shantenu Jha, Yaakoub El Khamra, Frank O. Bryan, Benjamin Kirtman, and Frank J. Seinstra. 2017. On the complexities of utilizing large-scale lightpath-connected distributed cyberinfrastructure. *Concurrency and Computation: Practice and Experience* 29, 2 (2017), e3853. doi:10.1002/cpe.3853 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.3853 cpe.3853.
- [25] Message Passing Interface. 1994. *MPI: A Message-Passing Interface Standard, Version 1.0*. Technical Report. Message Passing Interface Forum. Initial MPI Standard Document.
- [26] Message Passing Interface Forum. 2025. *MPI: A Message-Passing Interface Standard, Version 5.0*. <https://www.mpi-forum.org/docs/mpi-5.0/mpi50-report.pdf>
- [27] Maximilian Sander, Hannes Tröppen, Christian von Elm, and Robert Schöne. 2025. Towards Large-Scale Top-Down Microarchitecture Analysis Using the Score-P Framework. In *Euro-Par 2024: Parallel Processing Workshops*, Silvina Caino-Lores, Demetris Zeinalipour, Thaleia Dimitra Doudali, David E. Singh, Gracia Ester Martín Garzón, Leonel Sousa, Diego Andrade, Tommaso Cucinotta, Donato D'Ambrosio, Patrick Diehl, Manuel F. Dolz, Admela Jukan, Raffaele Montella, Matteo Nardelli, Marta Garcia-Gasulla, and Sarah Neuwirth (Eds.). Springer Nature Switzerland, Cham, 189–200.
- [28] Robert Schöne, Ronny Tschüter, Thomas Ilsche, Joseph Schuchart, Daniel Hackenberg, and Wolfgang E. Nagel. 2017. Extending the Functionality of Score-P Through Plugins: Interfaces and Use Cases. In *Tools for High Performance Computing 2016*. Springer International Publishing, Cham, 59–82. doi:10.1007/978-3-319-56702-0_4
- [29] Robert Schöne, Markus Velten, Daniel Hackenberg, and Thomas Ilsche. 2024. Energy Efficiency Features of the Intel Alder Lake Architecture. In *Proceedings of the 15th ACM/SPEC International Conference on Performance Engineering* (London, United Kingdom) (*ICPE '24*). Association for Computing Machinery, New York, NY, USA, 95–106. doi:10.1145/3629526.3645040
- [30] Robert Schöne, Thomas Ilsche, Mario Bielert, Markus Velten, Markus Schmidl, and Daniel Hackenberg. 2021. Energy Efficiency Aspects of the AMD Zen 2 Architecture. In *2021 IEEE International Conference on Cluster Computing (CLUSTER)*. 562–571. doi:10.1109/Cluster48925.2021.00087
- [31] Hans Segura, Xabier Pedruzo-Bagazgoitia, Philipp Weiss, Sebastian K. Müller, Thomas Rackow, Junhong Lee, Edgar Dolores-Tesillos, Imme Benedict, Matthias Aengenheyster, Razvan Aguridan, Gabriele Arduini, Alexander J. Baker, Jiawei Bao, Swantje Bastin, Eulàlia Baulenas, Tobias Becker, Sebastian Beyer, Hendryk Bockelmann, Nils Brüggemann, Lukas Brunner, Suvarchal K. Cheedela, Sushant Das, Jasper Denissen, Ian Dragaud, Piotr Dziekan, Madeleine Ekblom, Jan Fredrik Engels, Monika Esch, Richard Forbes, Claudia Frauen, Lilli Freischem, Diego Garcia-Maroto, Philipp Geier, Paul Gierz, Álvaro González-Cervera, Katherine Grayson, Matthew Griffith, Oliver Gutjahr, Helmuth Haak, Ioan Hadade, Kerstin Haslehner, Shabeh ul Hasson, Jan Hegewald, Lukas Kluff, Aleksei Koldunov, Nikolay Koldunov, Tobias Kölling, Shunya Koseki, Sergey Kosukhin, Josh Kousal, Peter Kuma, Arjun U. Kumar, Rumeng Li, Nicolas Maury, Maximilian Meindl, Sebastian Milinski, Kristian Mogensen, Bimochan Niraula, Jakob Nowak, Divya Sri Praturi, Ulrike Proske, Dian Putrasahan, René Redler, David Santuy, Domokos Sármany, Reiner Schnur, Patrick Scholz, Dmitry Sidorenko, Dorian Spät, Birgit Sützl, Daisuke Takasuka, Adrian Tompkins, Alejandro Uribe, Mirco Valentini, Menno Veerman, Aiko Voigt, Sarah Warnau, Fabian Wachsmann, Marta Waclawczyk, Nils Wedi, Karl-Hermann Wieners, Jonathan Wille, Marius Winkler, Yuting Wu, Florian Ziemien, Janos Zimmermann, Frida A.-M. Bender, Dragana Bojovic, Sandrine Bony, Simona Bordoni, Patrice Brehmer, Marcus Dengler, Emanuel Dutra, Saliou Faye, Erich Fischer, Chiel van Heerwaarden, Cathy Hohenegger, Heikki Järvinen, Markus Jochum, Thomas Jung, Johann H. Jungclaus, Noel S. Keenlyside, Daniel Klocke, Heike Konow, Martina Klose, Szymon Malinowski, Olivia Martius, Thorsten Mauritsen, Juan Pedro Mellado, Theresa Mieslinger, Elsa Mohino, Hanna Pawłowska, Karsten Peters-von Gehlen, Abdoulaye Sarré, Pajam Sobhani, Philip Stier, Lauri Tuppi, Pier Luigi Vidale, Irina Sandu, and Bjorn Stevens. 2025. nextGEMS: entering the era of kilometer-scale Earth system modeling. *Geoscientific Model Development* 18, 20 (2025), 7735–7761. doi:10.5194/gmd-18-7735-2025
- [32] Sameer Shende, Allen D. Malony, Wyatt Spear, and Karen Schuchardt. 2012. Characterizing I/O Performance Using the TAU Performance System. In *Applications, Tools and Techniques on the Road to Exascale Computing (Advances in Parallel Computing)*.
- [33] Larry Smarr and Charles E. Catlett. 1992. Metacomputing. *Commun. ACM* 35, 6 (1992), 44–52. doi:10.1145/129888.129892
- [34] Hannes Tröppen, Mario Bielert, and Thomas Ilsche. 2023. Evaluating the Energy Measurements of the IBM POWER9 On-Chip Controller. In *Proceedings of the 2023 ACM/SPEC International Conference on Performance Engineering* (Coimbra, Portugal) (*ICPE '23*). Association for Computing Machinery, New York, NY, USA, 67–76. doi:10.1145/3578244.3583729
- [35] Christian von Elm and Robert Schöne. 2025. Performance Tools for the NEC SX-Aurora Tsubasa. In *Companion of the 16th ACM/SPEC International Conference on Performance Engineering* (Toronto ON, Canada) (*ICPE '25*). Association for Computing Machinery, New York, NY, USA, 215–222. doi:10.1145/3680256.3721323
- [36] Zeyu Yang, Karel Adamek, and Wesley Armour. 2024. Accurate and Convenient Energy Measurements for GPUs: A Detailed Study of NVIDIA GPU's Built-In Power Sensor. In *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. 1–17. doi:10.1109/SC41406.2024.00028