

Workshop Report:

2024 International Symposium of Quantitative Codesign of Supercomputers

Version: 1.0



Acknowledgements

We would like to acknowledge the efforts of the following people who not only made our symposium possible, but worked to make our symposium outstanding: SC'24 workshop committee chair Janine Bennett, vice chair Bruno Raffin; the anonymous workshop proposal reviewers who provided helpful guidance; our web designer Douglas Edwardson; and the audio-visual team that provided support for our symposium.



Thank You!

— *Terry Jones*, chair Quantitative Codesign of Supercomputers

TABLE OF CONTENTS

1.	BACKGROUND ON QUANTITATIVE CODESIGN.....	3
2.	PURPOSE OF THE WORKSHOP	4
3.	WORKSHOP STRUCTURE.....	6
3.2	AGENDA	6
3.3	A HYBRID FORMAT: ACCOMMODATING IN-PERSON AND REMOTE PARTICIPATION.....	6
3.4	VENUE FEEDBACK	7
4.	WORKSHOP OUTCOMES.....	9
4.1	HPC CONTRIBUTIONS.....	9
4.2	WORKSHOP DISCUSSION AND FINDINGS	9
5.	POST-WORKSHOP RECOMMENDATIONS AND “NEXT STEP” STRATEGIES.....	15
5.1	CONTINUE THE WEBSITE (LINK).....	15
5.2	DISSEMINATE THE WORKSHOP REPORT	15
5.3	TRACK POTENTIAL MISSION FURTHERING OPPORTUNITIES	15
5.4	ADVANCE THE QUANTITATIVE CODESIGN AGENDA WITH A 2024 SYMPOSIUM	15
	APPENDIX 1 – RELATED ACTIVITIES.....	16
	APPENDIX 2 – SYMPOSIUM BIOGRAPHIES	17
	APPENDIX 3 – ORGANIZING COMMITTEE AND PROGRAM COMMITTEE	20
	APPENDIX 4 – ATTENDEES & WORKSHOP PHOTOGRAPHS	21
	APPENDIX 5 – FLASH TALKS.....	22
	ROBERT KEßLER, UNIVERSITY OF COLOGNE	22
	J. ZACH McMICHAEL, UNIVERSITY OF TENNESSEE	25
	MIWAKO TSUJI, RIKEN	29

2023 International Symposium on the Quantitative Design of Supercomputers
Held in conjunction with Supercomputing '24
Atlanta, GA – November 17, 2023

1. Background on Quantitative Codesign

The Quantitative Codesign of Supercomputers symposium is an annual workshop series that aims to significantly improve the effectiveness of high-performance computing through bringing about increased understanding of current limitations and improved development processes. This symposium considers combining two methodologies—collaborative codesign and data-driven analysis—to realize the full potential of supercomputing. For full potential of supercomputing, we consider everything pertaining to output production including, but not limited to, the performance of applications, system software, workflows, health of hardware. Our centers store vast sums of information, yet using this data is a demanding task. To a large extent the difficulty in obtaining quantitative insight has to do with discovering, accessing, and analyzing the right data. Codesign also presents formidable challenges, e.g. on how to use the data collected on current systems to facilitate the (potentially very different) design of next-generation supercomputers and successfully support our upcoming environments. Quantitative codesign offers a collaborative evidence-based approach to address our existing needs and our upcoming ambitions. This symposium was created to bring together leaders in the field to review current efforts across centers and discuss areas that show potential.

Over the past decade, there has been a growing awareness of the multi-faceted benefits we can derive from data-driven strategies like Quantitative Codesign. This increasing awareness, along with improvements in Machine Learning (ML) technologies, have driven vendors, operations staff, and application developers to espouse integrating an ever-increasing level of instrumentation into their products. The time is ripe for turning this vast trove of available information and the incredible advances in analysis technologies it represents into appropriate knowledge and understanding. Doing so would create a feedback loop that could assist vendors and software developers in their designs. The recent National Strategic Computing Initiative Update Report has recommended that we promote timely access for developers of technologies, architectures, and systems to carry out the research needed to create the future computing software ecosystem, and Quantitative Codesign provides a solution to the ‘access problem’ of these extremely rare machines. If the future envisioned by the CSESSP report is to be realized, our software base will require significant investment in both modified and new code — an activity enormously assisted by Quantitative Codesign. There is no disagreement that more knowledge is good though there is still lack of concurrence across HPC stakeholders as to the cost/benefit tradeoff for varying fidelities of information collection and long term storage. The benefits of Quantitative Codesign will come through integrating design processes with more detailed knowledge of the interactions of the various components within the HPC ecosystem.

Quantitative Codesign is also essential for addressing challenges brought about by the recent trend of increasing heterogeneity and varied accelerators in HPC architectures. For example, many HPC machines now incorporate alternative types of memory alongside conventional DDR SDRAM. Technologies such as "on-package" or "die-stacked" DRAM as well as non-volatile RAMs can provide distinct advantages compared to conventional DRAM, including higher performance as well as cheaper and more energy

efficient storage per byte. Each of these technologies also comes with its own limitations, such as smaller capacity or less bandwidth for reads and writes. Further complications arise because some of these new technologies can interface directly with processor caches, while others can only be accessed through peripheral devices, such as GPUs or other accelerators.

Quantitative Codesign could mitigate many of the current problems with allocating and managing such heterogeneous resources effectively. Detailed knowledge of application demands will enable architects to make better decisions about how to select and organize computing and memory hardware. This approach can also help system software, including operating systems, compilers, and runtime software, distribute the available hardware resources among applications more effectively. Codesigned system software could utilize knowledge from new data sources for better energy efficiency and workflow management. Integrating high-level profiling and analysis with low-level resource management routines will enable these systems to implement new policies that respond flexibly to changes in application demands and could potentially expose important new efficiencies on platforms with heterogeneous hardware.

2. Purpose of the Workshop

The purpose of the workshop was to build the necessary community support to build up and foster concrete implementations of quantitative codesign. As architectural options expand in type and complexity, the need for a quantitative basis to drive architectural directions becomes increasingly urgent. We do not have the primary mission to raise awareness of an individual's research; rather we wish to bring more wide-ranging interactions highlighting vision and positions and stimulating discussions.

Any shortfall in our detailed understanding of operations and performance impacts the whole spectrum of stakeholders. Whether providing hardware architectures, system software, application programming environments, or production run-time environments, having the appropriate knowledge to optimize the interaction and configuration of all of these critical components as well as the evolution of the HPC ecosystem is critical to continued growth. The rapidly changing HPC landscape demands a codesign that effectively uses the data collected on previous and current systems to facilitate the design of next-generation supercomputers and successfully support our upcoming environments. Specifically, we would like to bring increased clarity for our challenges and opportunities.

- **Challenges:** We have important issues to resolve, but we are not starting from scratch. HPC computing centers already collect a wealth of information on the health, usage, and efficiency of our machines, workflows and programming environments. While collection and analysis of this information has evolved and improved over the years, there are still severe gaps that have left us unable to provide the knowledge that is needed by hardware and software vendors, system operations staff, application developers, and user groups to create and operate highly efficient and secure large scale HPC systems. Would-be users of this information face difficulties in obtain insight from the collected data a timely manner, and efforts to provide both data and analysis means are currently fragmented across centers both at national and international levels. The infrastructure to collect, store, share and analyze the volumes of available information is a core capability—yet, many barriers remain due in large part to the many stakeholders and insufficient coordination, but also due to data privacy and security issues. With many new potential information sources in future systems, we must quickly identify and address critical requirements and gaps across the various stakeholders. Doing so will enable us to create collective and collaborative solutions that address both existing challenges and emerging needs and effectively support our upcoming HPC environments. The nature of this challenge suggests that it is an excellent opportunity for a codesign approach. Codesign is defined as the process of jointly designing interoperating components of a computer system—in particular: applications, algorithms, programming models, system software, as well as the hardware on which they run, and the facilities hosting them. Designing solutions based on intelligence derived from the data collection and analysis processes described above are henceforth referred to as Quantitative Codesign of Supercomputers.

Making progress at the highest end of HPC without access to the needed data can be compared to being asked to fly an airplane at night without sufficient instrumentation. Vendors are provided with example applications to target, but often lack a true understanding of where inefficiencies manifest on full scale workloads. Furthermore, computer architecture simulators face an inevitable challenge in trying to incorporate all the critical performance-killing attributes of current generation technologies and their integration: a simulation that includes all details of the architecture, from the chip micro-architecture up to infrastructure, would take forever to run. For this reason, simulations must make tradeoffs between the accuracy of their representation and the required modelling time. Hence the vendors miss opportunities for improvement. Moreover, users often only have feedback on operating efficiency at the granularity of total application execution time. Low-level interactions frequently cause substantial performance degradations that users are unable to explain. Likewise, operations staff often lack knowledge of application resource utilization and cannot diagnose the longer run times experienced by the users. In addition, operations staff cannot ensure secure operations without an understanding of normal (expected) behavior and anomalies that deviate from that. Since root causes go undiagnosed on current systems, next generation systems will also fail to address the very same problems.

- **Opportunities:** First and foremost, we wish to discuss the merits of a coordinated effort to bring together the helpful data from each stakeholder in the codesign space into a framework where data discovery and access is straightforward regardless of data source while respecting data privacy and security concerns. The envisioned Quantitative Codesign environment would pull together data traditionally held by disjointed communities (e.g., sysadmins, application teams, vendors, and so on) into a framework where the needed data is easily accessible. This framework would provide flexible but secure mechanisms for data providers who wish to share their data with others including application teams, vendors, facilities, operations, and system software researchers. In many cases, we seek to bring together data that is currently being produced although not generally known or utilized for a variety of reasons; in a few instances, we seek to extend and provide new data collection capabilities.

For example, one area that is ripe for integration with Quantitative Codesign processes is the intersection of application development and run-time environments. In the past few years Continuous Integration (CI) has been widely adopted by development teams to continuously test development efforts. As part of these CI efforts, developers test across a variety of platforms on a daily basis and typically provide a pass/fail result for each. Introducing targeted run-time data collection (e.g., memory, application & hardware counters, MPI, OpenMP, GPGPU, I/O, energy consumption) and quantitative analysis into this process would enable feedback to users and identify issues within applications, compiler capabilities, runtimes, and differences across platform architectures that ultimately would drive improvements across the spectrum of stakeholders.

Integrating Quantitative Codesign capabilities with existing design processes will enable more effective solutions across the computing stack. Information derived from monitoring and analysis would provide valuable insight for users, application developers, system architects, and facility designers as to how, and why, applications make use of the underlying system resources. Furthermore, by identifying the appropriate stakeholders and introducing them to information originating from diverse collection regimes, this symposium seeks to facilitate the discovery and sharing of potentially useful intelligence among larger teams and communities. In doing so, this approach also has the potential to spark further discussions and research on how to collect, employ and share this information more effectively. Thus, there is significant opportunity for discoveries that will not only increase application performance, but also benefit the broader HPC and scientific communities.

3. Workshop Structure

The Quantitative Codesign of Supercomputers symposium took place during the opening day of the 2024 Supercomputing conference. The workshop was structured for hybrid-attendance with both in-person and virtual attendees and speakers. Further, the workshop was framed in the Symposium format to achieve the kind of deep interactions that lead to change within HPC. Our preference for audience interaction was in response to the state of the field (which we see as in its infancy).

3.1 Workshop Theme

The theme for SQCS'24 was opportunities and challenges in ADVANCED MEMORY. There are new research topics in heterogeneous computing, energy efficient computing performance, AI architectures, and edge computing that are driving innovations in advanced memory technology. Generative AI, Foundation Models, and HPC are important drivers for performance improvements in high bandwidth memory. Growing industry support and adoption of Compute Express Link (CXL) is driving interesting codesign explorations with various application drivers for CXL capabilities including: multi-tiered memory hierarchy, memory disaggregation large memory pools with global fabric attached memory, support for heterogeneous computing with shared memory pools, and revisited concepts for compute near memory designs. In shared memory, application codesign tradeoffs are raised for hardware vs software coherency and consistency management. New codesign opportunities also arise to understand memory requirements for Federated Learning at low power edge devices.

3.2 Agenda

Given our desire to bring more wide-ranging interactions highlighting vision and positions and stimulating discussions, we developed a schedule designed to facilitate these interactions (see Table 1 below). In particular:

- The keynote speaker was chosen based on his long history in HPC with work that spans all areas of codesign including novel architectures, system and application software, tool development, performance diagnostics and more, in both lab and academic environments.
- Three distinguished speakers were chosen who, as an aggregate, provided codesign perspectives on the use of carefully crafted kernels to inform telemetry and monitoring, how quantitative co-design was recently used to design memory architectures for CEA, and ORNL's experience with quantitative co-design.
- A collection of position papers from an international collection of experts with diverse backgrounds in codesign, HPC system software and middleware research, center wide monitoring and operational aspects, bringing HPC products to market, and application / libraries.
- A moderated discussion of audience, speakers, and panelists was included to enable both technical discussions and community-building.

3.3 A Hybrid Format: Accommodating In-Person and Remote Participation

As with our previous Symposium, the lingering effects of COVID and an increased confidence in the effectiveness of virtual participation had an impact on the format and character of the workshop. This was the third time for the SC series of conferences to ever have a hybrid format: SC24 supported both in person attendees at the World Conference Center in Atlanta, Georgia and remote attendees through the revamped SC24 online platform, Zoom and Sli.do. The role of the session chair and organizer remained largely the same as in previous years with some adjustments and increased responsibilities to account for remote participation by speakers and attendees. The Quantitative Codesign of Supercomputers symposium was presented via *Live stream sessions*. Under this format, content was recorded by AV technicians at the convention center and sent to remote participants in real time via Vimeo. Remote presenters connected via zoom (see Figure 1). For all remote symposium presenters, we arranged for an internet assessment on

the day of the symposium prior to the symposium start. This was used to ensure no fallback measures were needed. All remote speakers were able to participate as planned.

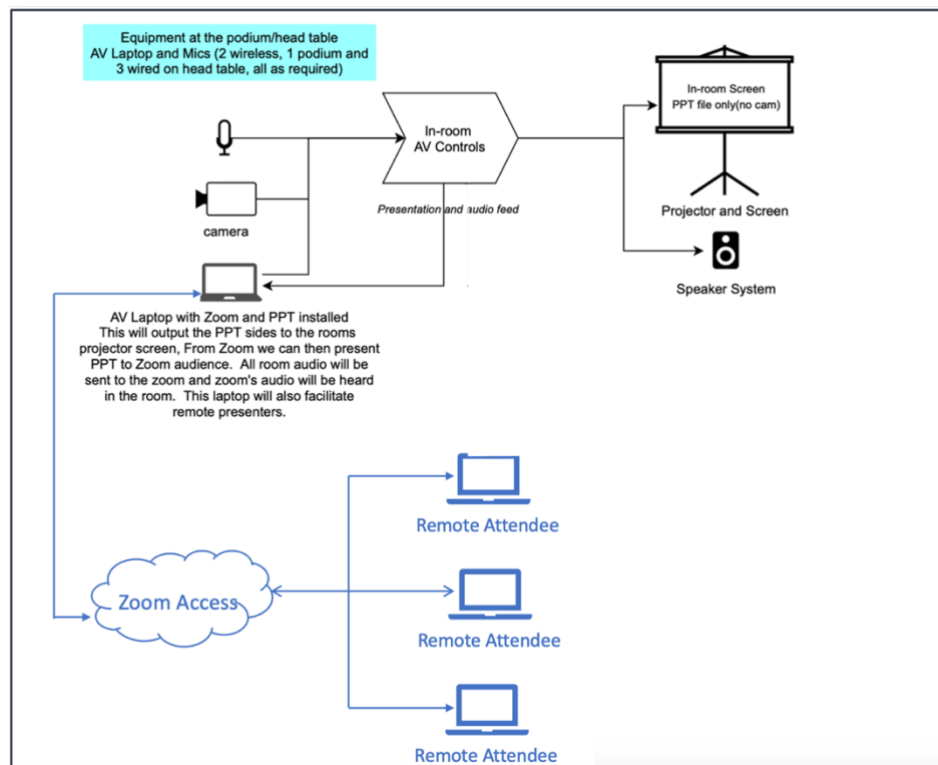


Figure 1 – Logistical Setup of Live Stream Format Used By Symposium.

3.4 Venue Feedback

The Georgia World Conference Center and Atlanta receives mostly positive marks from SQCS'24. Positive aspects include:

- The SC'24 hotel-to-venue buses were a positive. They were convenient and reliable.
- Perhaps the most notable thing about the convention center was its enormous size -- SC24 only used half of the convention space. The rooms, AV, temperature control, acoustics, and general aesthetics were fine.
- The room size and location within the conference center for SQCS'24 were fine.

The following offer areas of potential improvement:

- Signage: Several of us went to the convention center the day before the symposium to get our credentials and check out the space, and there was no signage for SC'24 on the half of the convention center we went to. In addition, there was insufficient room number signage for the collection of rooms at the end of the corridor where SQCS'24 was held.
- AV Support: The AV support needs to show up at least 30 minutes before the symposium starts.
- Room Setup: Our workshop had a panel. It would have been nice to have a place with chairs for panelists to sit in front.

Table 1 – Symposium Agenda

All Times US CT	Speaker/Panelist		Abstract
9:00 to 9:10		Terry Jones	Opening Remarks from Workshop Chair Welcome and workshop logistics
9:10 to 9:40		Dorian Arnold	From Data-Driven Designs to Data Driven Co-Design. Much attention is being placed on the use of artificial intelligence and machine learning for application development, system design, and performance analysis. Using our ongoing work on data-driven HPC, including AI and ML-based approaches, for HPC application trace and performance data synthesis, HPC software synthesis and code optimizations, and HPC system-monitoring, this talk inspires the promise of principled and automated data-driven approaches in the HPC system design lifecycle.
9:40 to 10:00		Jeff Hammond	The Parallel Research Kernels and Their Use in Co-design The Parallel Research Kernels (PRK) were created to be the simple yet still interesting implementations of fundamental algorithms in high-performance computing, which could be used to evaluate and improve hardware and software systems. In this talk, I will describe the design methodology of the PRK and their use in multiple contexts. First, we consider the viability of alternative distributed programming models as compared to multiple flavors of MPI, especially the sensitivity to message granularity. Second, we demonstrate the use of the PRK to evaluate programming languages. Finally, we use the PRK to measure the behavior of accelerators and heterogeneous memory systems.
10:00 to 10:30	[break]		[break location]
10:30 to 10:50		Lilia Zaourar	A Co-Design Approach to NUMA Architectures in HPC: Quantitative Evaluation and Design Exploration Understanding the performance potential and data placement challenges in Non-Uniform Memory Access (NUMA) architectures is crucial for optimizing High-Performance Computing (HPC) systems. We will present a quantitative approach, using simulations and models, that provides essential insights into how system architecture impacts microbenchmarks and real-world applications. We model a NUMA architecture with ARMv8 Neoverse V1 processors, leveraging the gem5 and VPSim simulation platforms.
10:50 to 11:10		Jack Lange	Using Telemetry to Derive System Architecture Requirements: Experiences at the Oak Ridge Leadership Computing Facility Increasing system complexity and component costs mean that designing supercomputers and other HPC systems requires significant architectural compromises to be made. System architects are being forced to make ever more significant tradeoffs. To guide these decisions, it is first necessary to understand what the resource requirements of the workloads are. At ORNL we have been investigating the feasibility of using telemetry collected from existing systems to better understand how those systems are being used by users and their applications. I will give an overview of this effort and the challenges we have faced.
11:10 to 11:35	Work in Progress Flash Talks	Miwako Tsuji (Riken)	Feasibility study of compiler model toward the co-design of the next Fugaku
		Robert Keßler (Univ of Cologne)	Challenges to Overcome the Disparity Between Resource Allocation and Utilization of HPC Clusters on Reconfigurable Architectures
		J. Zach McMichael (Univ of Tennessee)	VMem: A User-level Runtime for Enabling Fine-Grained Control of Physical Memory
11:35 to 12:25	Moderated Discussion		Your opportunity for audience & panelists (our 4 invited speakers) to dig deeper
12:25 to 12:30		Terry Jones	Closing Remarks.

4. Workshop Outcomes

4.1 HPC Contributions

The following positive results have been achieved with from the workshop:

- A large group of high performance computing professionals came together to pursue community building
- Monitoring journals (outcome and strategy) were discussed and templates provided to guide the process of data collection and the use of these data
- Videos of the invited talks and panels were recorded by SC's Live Stream AV team
- Discussion on Vision and Possibilities of Quantitative Codesign of Supercomputers were discussed, and ideas for future work were identified
- This workshop report was written to document the results

In addition, monitoring journals (outcome and strategy) were discussed and templates provided to guide the process of data collection and the use of this data.

4.2 Workshop Discussion and Findings

The workshop panel and ensuing audience interaction produced a very interesting discussion. Each distinguished panelist was given the following instructions:

Please address the following two charge questions:

- Engineers are taught the KISS principle (Keep It Simple Stupid) for good reason, but computer architectures have been trending toward increased node-level complexity and heterogeneity for more than a decade. How can SQCS have a role in reducing the growing computer architecture complexity trend?
- Given that HPC is no longer the driver for the largest computing systems, how can the HPC community drive innovation in parallel computing architectures or participate in co-design with non-HPC users (i.e. AI)?

We are planning for the Panel Session to last 50 minutes. Please try to cover the two questions in about 6 minutes total time so that we will have around 25 minutes for audience discussion. Finally, to spark a lively discussion, we would encourage you to take the most provocative stance that you can truly support.

- Charge Question 1 Discussion

Engineers are taught the KISS principle (Keep It Simple Stupid) for good reason, but computer architectures have been trending toward increased node-level complexity and heterogeneity for more than a decade. How can SQCS have a role in reducing the growing computer architecture complexity trend?

- Jeff Hammond: Simplicity is bad. It makes the computer scientists feel good, and it makes the applications run slower. BlueGene was simple, and it was slower than Xeon based Cray

machines for almost everything. Simplicity in the architecture is turning off all the things that make bad code go fast. So I argue against simplicity. At the same time, we have to decide *what kind of simplicity*: Grace hopper is crazy complicated and yet super simple -- its simple in that the amount of memory levels are small and pipes between them are big. To a computer architect, Grace Hopper is hard; to a programmer trying to get good performance, it's easy. So simple or not simple depends on who you are asking. I say let the hardware be as complicated as it needs to be, to make the application scientist feel like it's easy to get good performance.

- Jack Lange: I think we will need to move to simple architectures just because increasingly complex architectures are not going to be *affordable*. I view complexity as general purpose -- the complex architecture can do lots of different things. We're reaching the point where we are going to have to choose what do we want to do well, and what can we sacrifice to do not so well. This will lead to more streamlined architectures that are more purpose-built for certain applications. Every complexity that we've incorporated has come with an abstraction that makes it understandable to people. As long as abstractions keep up, we can have increasing complexity but we will need abstractions to come with the complexity.
- Dorian Arnold: The hardware modules need to be sophisticated in order to provide the type of performance capabilities that we need -- the *boundaries* are where we simplify things. The HPCers that imagine that they need to eek out every last bit of performance are the ones that go deep into the bowles of hardware modules to understand the sophistication and get as much performance as they can from that sophistication -- but for most applications that's not necessary and you can succeed with 'good enough' where you utilize simple boundaries that might take away some of the possible performance gains. So the philosophy of complicated modules and simple boundaries is what you will need in the co-design space.
- Lilia Zaourar: I agree with the previous opinions, and for me the primary question is *the cost*. To have a complex hardware that will 'feel' simplicity, but to have this simplicity you must invest great cost on abstraction, on development, and so on. You can always have what you want if you okay with paying the cost for it.
 - Question: do you mean the cost of development, or the overhead with delivering the hardware?
 - Lilia: Both.
- Nick Wright: In a way, the charge question is ill-formed. If you're a race car driver, you have a very deep understanding of the car and you drive it very fast. I, on the other hand, don't care how my car works and it goes plenty fast for my needs. What role can co-design have here? Well, it can tell you the delta between the race car drivers performance and the casual drivers performance such that you can make determinations on whether it's worth it to you to extra feature (added complexity) in or not, and whether effort needs to be put into the abstraction so that it can be used by many other people.
- Jim Ang: If we have a simpler architecture or simpler data utilization model, we may be able to better control over energy consumption decisions. Complexity may lead to higher energy consumption. Many energy concerns are mapped to the data utilization patterns. Ultimately, one of the factors of growing importance is the amount of energy efficiency we can obtain in our future systems -- and for this goal simplicity may help. If systems are too complex, it will be difficult to get a handle and control energy efficiency.

- Estela Suarez: I thought the systems were becoming more complex, that the heterogeneous systems give us more performance for the power consumed. Isn't it the combination of trying to have a general purpose system combined with a specialized performance system that adds lots of complexity?
- Jeff Hammond: We still don't have a formal definition of "complexity". Xeon is one of the most complex pieces of technology ever built -- and yet on another level, it's incredibly simple. Is Xeon more "simple" than an NVIDIA GPU? I don't know how to answer that. What I can tell you is the amount of transistors devoted to load, store, and math is way higher in the GPU -- NVIDIA has a whole bunch of buffers that are designed to make terrible code not run terrible. In contrast, on GPU we're able to say *we're not running java script, we're running CUDA or something similar*. GPU code used to be very simple -- you wrote vector code similar to what ran on a Cray back in the day, and it just ran beautifully with no caches. Things have gotten much more complicated because of demands on bandwidth and eventually you have to have a hierarchy and a hierarchy is complexity. Different communities, hardware designers, OS designers, application developers, have totally different views on what complexity they like.
- Terry Jones: My thoughts on complexity ran to *performance portability*. It's no longer the case that you can count on the C compiler to realize great performance when you move your application from one complex leadership class machine to another.
- Estela Suarez: Complexity is related to how much of the abstraction you have to be aware of. How much do you have to know about the system and code for the particular system to get performance out of the system. How transportable is the performance.
- Jack Lange: I'd say another component of this is second order effects. If you have a component in the system that has dependencies on other components in the system, that's where we see a lot of our issues arising. How do we make these things talk to each other? What are the interfaces and how inter-dependent are they? So if you have the ability to partition the components, and clean interfaces that are able to operate interdependently, you can contain the complexity inside those silos and that's a much more attractive solution than having the complexity all-encompassing.
- [unidentified]: On the complexity question, it really depends on who or what is programming the system. If it's humans, yes absolutely complexity should be bounded. But what will be programming these machines in the future? It may not be us. At the moment, we've designed the machines for us to program, but that may not be the case in the future. Perhaps machines will design themselves and program themselves in the future.
- Dorian Arnold: Another aspect of this is related to *correctness* which has a second-order effect on performance. You can modularize the complexity to define simple boundaries that allow you to use the modules easily, but it becomes complex to understand the performance effects of these now complex modules put together correctly from a performance standpoint. This leads to the quantitative co-design approach where you take a data-driven approach and using analysis to help understand those performance complexities in ways that are very difficult for us to unravel otherwise.
- [unidentified (perhaps John McCalpin? ...the camera doesn't pan to the microphone)]: An architecture should provide functionality that lets the hardware plus the user control the things that are most expensive. We have 64-bit floating point hardware because it would be

insane to try to build a history-based predictor to recognize the sequence of integer instructions that make a 64-bit floating point operation and convert it on the fly, and yet the most expensive thing in current systems is *memory references* and we have zero control in the architecture over how that happens. Even if you split the software up into a compute part and a memory part, the memory part has no hardware controls, the hardware has no features that allow you to control caching -- it has very minimal features that don't work very well. As to Jeff's comment on cache blocking, one of the reasons that it's so hard to get a compiler to cache block well is that they don't well with aliasing and set associativity, they can work a lot better if you have scratchpad memories. So what I'm arguing for is the architectures that we use don't include the things that are most important. The architects don't mention communication, they don't mention synchronization, and those are the things that cost all of the time and all of the energy. A huge fraction of the energy on an Intel type processor is the caching and speculation and coherence operations that allow the thing to be apparently easy to use. If we want things to be significantly more energy efficient, we need to re-architect them to expose a much higher semantic level for data motion, and then figure out the 'little problem of software' to make them work.

- Nick Wright: Here's two points. First, you can switch a cache for a scratchpad, you get bad performance. How many people want to write code for a system that uses a scratchpad? Second point: the problem is that the compute and the memory are coupled: as soon as you have indirect addressing, you don't actually know what piece of memory you want to move until you compute which piece of memory you want to move. So a scratchpad doesn't solve that problem, and if you're writing a real code you have indirect addressing so the cache is the least bad solution that people have found. I'm sure that if you were allowed to run the code once, and instrument it completely, you could then rewrite it to run faster using a scratchpad, but that's a pointless exercise.
- [Unidentified]: That's absolutely true. Scratchpads have been focused and extremely successful in *niches* like digital streaming applications. Most of those systems allow the hardware to be configured to some fraction of scratchpad and some fraction of cache. One of the reasons that people have been reluctant to look at scratchpads for more general use has been the limitations on when they are beneficial. However, the idea that I'm proposing is a computing substrate that is a factor 20x cheaper, but you have to overcome some of the challenges of scratchpads. At one time, people were content with vector computing. The ensuing x86 style of computing has lasted for many years now and there hasn't been movement to make nodes significantly cheaper. In fact, nodes are now more expensive than they were. If you want machines to be very inexpensive and energy efficient, the architecture has to be significantly different than what we have now.
- Estela Suarez: To me, the question seems to be one of finding a sweet spot: how much complexity do we want to expose and transfer to the user so that they can control, and how much can we abstract away.
- Jeff Kuehn: I have a question for the audience: How many of you are using an LLM to write code now. Is there a reason that other people have not started down this path already? You can run an LLM on your laptop.
 - Jeff Hammond: This is related to something Dorian commented to in his talk earlier today. These models capture the collected available intelligence. If I ask ChatGPT to

tell me about the strict aliasing rule, it will do fine. But if I ask it to do something very specific without the supporting information, it will make up answers. I asked ChatGPT to write Ada and time a software module with start-time, stop-time, and dividing the duration by the iteration count, it gave me 12 consecutive garbage responses that were non-compliant code with made up Ada functions that it claimed were standard (but were actually completely bogus). It hallucinated everything! If there's not enough human intelligence on the internet to be trained on, the models don't know. ChatGPT is no better than the info already on StackOverflow, it's just faster for people to use.

- Dorian Arnold: The one part that I'll add to that is that it does unify the collective wisdom. That's the power that it really has. And with enough computational power, it brings together the available collective wisdom very quickly.
- Jeff Hammond: It [ChatGPT] will never solve problems that people haven't already solved.
- Dorian Arnold: Except in situations where the solution is the composition of multiple individual solutions that have not previously been brought together.
- Jeff Kuehn: Or solutions that are near existing solutions. Just as an anecdote, I produced 25,000 lines of working code in an evening while watching a movie with my wife. There's an immense capability here for productivity and addressing complexities. LLMs certainly have their limitations and trying to get them to write an obscure or niche programming language is an example. Getting it to produce C code or Python code is a different matter. There's a corpus available for that.
- Estela Suarez: How efficient was your code? Because I tried ChatGPT, and the code efficiency was very poor.
- Jeff Kuehn: You have to guide it like a dumb graduate student. If you tell it "cache block this loop," it can do that in an instant rather than 30 minutes. You do have to accept the limitations and recognize that you have to guide it. You have to be able to recognize when its walking down a rat hole and steer it back out. While LLMs are not going to replace good programmers, that doesn't mean don't use LLMs.
- Dorian Arnold: I'll reframe that in a slightly different way. Consider it a productivity framework. It's not producing your programs, but rather its helping you produce your programs in a very productive way. Just like visual programming, or compilers before the days of compilers when everyone wrote assembler, or before there was assembler and you had to use machine code. It's a productivity framework that can boost your productivity significantly.
- Jack Lange: AI works well at a certain abstraction level. I think its a higher abstraction level than I've worked in, but you have to accept that you're going to be working at a higher abstraction level and not looking too deeply into the code.
- [Unidentified]: I'm a grad student that has experimented with LLMs. It doesn't work for problem solving -- I need to do that. And if I'm doing that, I might as well write it myself. If it's time to write the core structure, LLMs have never worked for me. It's not there for solving novel problems. Hopefully that changes soon.

- Jeff Kuehn: I'll provide one more example that I can't get into the specifics because its in a forthcoming paper. ChatGPT was able to bring together obscure theorems to redesign an algorithm that has been in use for 30 years. It was able to suggest a mathematical change in the structure of the problem that resulted in a 10x improvement in the algorithm. I don't want to suggest that ChatGPT or any LLM is going to do something creative. But it will fuel the creativity of the experts who use it.
- Charge Question 2 Discussion

Given that HPC is no longer the driver for the largest computing systems, how can the HPC community drive innovation in parallel computing architectures or participate in co-design with non-HPC users (i.e. AI)?

- Jeff Hammond: NVIDIA is still doing HPC -- we have a couple of 100 people at the SC24 conference -- and we're still pretty successful at it. People don't like the prices, but other than that, nobody is complaining too much about what we build. It turns out that there are a lot of things about HPC and AI that are similar. The first one is that you move a lot of data. The second is you do a lot of math. The key difference between us HPC people and those AI people is numerical precision. As I was thinking about this charge question and symposium, it seemed to me that there are 2 fundamental co-design things to do with the interface of HPC and AI: the first one is figuring out how to do physics with different floating point types; the second one is AI capable of physics some of the time. Damian Rouson and people from NOAA are looking into using AI to do cloud micro-physics. Cloud physics is currently a collection of fudge factors with a bunch of empirical findings and a bunch of grid fittings. I think people just need to embrace the areas of overlap, and once in a while figure out where they don't overlap if that makes things cheaper on the HPC side.
 - Estela Suarez: If I were to paraphrase what you just said, there's codesign space between HPC and AI as long as this means transforming the applications that used to be called *HPC applications* into new forms that look like *AI applications*. But the hardware will be designed for AI.
 - Jeff Hammond: This is a good thing. If you go out and you want to create an HPC only company, and get money to buy first class architects, good luck! There are people at this conference that used to work for those companies and they will tell you that it doesn't work. Economics matters. If you want to get the best hardware designers in the world to give you the best memory bandwidth and the best floating point units, and the best compute, you want those people to be getting their paychecks from a company that makes money off of AI. So HPC should take it because Facebook and Amazon have already paid for it.
- Jeff Kuehn: I'm from AMD. What I would say is that if you want those systems that are going to do HPC well for you, it comes down to the memory performance, it comes down to the floating point performance, it comes down to power and price. Those are the four keys here. Putting too much memory on an SOC is going to blow out the price. Putting too little memory on the SOC is going to significantly damage the performance -- the performance and energy costs of pushing data on and off the SOC is going to kill you. So you need to dial in very carefully, very exactly with quantitative data how much memory do you need on that SOC. And if you can work with the footprint of memory on an SOC that is being driven by AI, all the better. On the floating point side -- AI folks are exploring fp6, fp8, fp16, bf16, a variety of different formats for very low precision computation. There's some applied math work that

needs to be recovered out of the 1950s, 60s, 70s to understand what are the necessities driving 64-bit math. Do you really need it? Then you can start asking yourself what features do I need in 64-bit versus 32-bit. Do I need tensor cores in 64-bit or do I only need them in 32-bit and lower precisions.

- [unidentified] What information would a chip vendor or a system vendor want.
 - Jeff Hammond: Jeff Kuehn actually answered this, and I agree with him. The two most expensive things are memory (how much is there and what is the speed), and how many floating point units do you include. We want to avoid top500-based procurements.
 - Jeff Lange: Just as an OLCF person, we don't ask for linpack numbers. That's not part of our benchmark suite either. So we're very aware that scientific computing is not just DGEMM. The other think that I would push back on, is that we do max out HBM capacity on many of our workloads. HBM is not just a vanity thing for us, we do actually need it.
- Recommended Reading:
 - DOE-SC Basic Research Needs for Microelectronics: Oct 23-25, 2018. https://science.osti.gov/-/media/bes/pdf/reports/2019/BRN_Microelectronics_rpt.pdf
 - PCAST Report: Revitalizing the U.S. Semiconductor Ecosystem https://www.whitehouse.gov/wp-content/uploads/2022/09/PCAST_Semiconductors-Report_Sep2022.pdf
 - Government Role: Crossing the Valley of Death. <https://www.nist.gov/chips/vision-and-strategy-national-semiconductor-technology-center>:

5. Post-Workshop Recommendations and “Next step” Strategies

The workshop finds that the present state of quantitative co-design is still nascent with plenty of divergent paths to choose from. There are a number of recommended “next steps” that should be followed to increase the usability of quantitative codesign of supercomputers.

5.1 Continue the website ([Link](#))

Provide ongoing support to Quantitative Codesign of Supercomputers website. This web presence becomes an anchor for announcements and a source to discover resources and pertinent email addresses.

5.2 Disseminate the Workshop Report

Providing this post-workshop report of the event will an important resource for the symposium’s community building objective. The contact data of the participants interested on receiving the report have been collected and will be used to spread the report in the community

5.3 Track Potential Mission furthering opportunities

This follow-up activity is to ensure that a wide segment of high performance computing is monitored for events, interactions and publications for opportunities to advance high performance computing through quantitative codesign concepts.

5.4 Advance the Quantitative Codesign agenda with a 2024 Symposium

Finally, we are encouraged to repeat the workshop in 2024. This fourth workshop should consider ...

Appendix 1 – Related Activities

Among the related activities that we wish to augment are the following:

- The Center and Application Monitoring Session held during the ECP Annual Meeting.
- The [International Workshop on Monitoring and Operational Data Analytics \(MODA\)](#) held with the annual ISC High Performance conference.
- The [Workshop on Monitoring and Analysis for High Performance Computing Systems Plus Applications](#) (HPCMASPA) held with the annual IEEE Cluster conference.
- The Workshop on Performance Monitoring and Analysis of Cluster Systems (PMACS) held with the annual Euro-Par conference.

Each of these related activities share an interest in the wealth of information exposed by these systems about how the system resources are being utilized. Our Symposium is unique in its emphasis on applying data to improve the codesign process. The Quantitative Codesign Symposium also has a distinguishing format and venue.

Appendix 2 – Symposium Biographies

Dorian Arnold – Leadoff Speaker and Panelist

Dorian Arnold is a tenured, associate professor of Computer Science at Emory University with over two decades of experience in large scale distributed systems, fault-tolerance, and software tools for high-performance computing (HPC) environments. He has 60+ peer-reviewed publications with 2300+ citations. His research projects have won two Top 100 R&D awards. In 2017, he was named an ACM Distinguished Speaker. Arnold earned Ph.D. and M.S. degrees in Computer Science from the Universities of Wisconsin and Tennessee, respectively. He earned a B.S. in Math and Computer Science from Regis University (Denver, CO) and his A.S. in Physics, Chemistry and Math from St. John's College (Belize)..

Jeff Hammond – Invited Speaker and Panelist

Jeff Hammond is a Principal Architect at NVIDIA, where he focuses on parallel programming models for GPUs and ARM CPUs. His life goal is to make programming supercomputers easier and more effective for scientists. At NVIDIA, Jeff works on HPC software for GPUs and ARM CPUs. His research interests include parallel programming models and system architecture. Previously, Jeff worked at Intel and the Argonne Leadership Computing Facility where he worked on a range of projects, including MPI-3, oneAPI, Blue Gene and Xeon Phi. Jeff received his PhD in Chemistry from the University of Chicago for work on NWChem.

Lilia Zaourar – Invited Speaker and Panelist

Dr. Lilia Zaourar is a CEA expert in co-design techniques for Computing Architectures at CEA LIST. She received an MS and PhD in Operational Research and Computer Science from the University Joseph Fourier, Grenoble, in 2007 and 2010, respectively. She developed various optimization algorithms for the design and test of integrated circuits. Then, she was a temporary teaching and research assistant at the SoC department in Computer Science PARIS 6 Laboratory, Sorbonne University, from 2010 to 2012. She was involved in developing optimization strategies for the resource-sharing problem to test embedded memories. She joined the CEA LIST in 2012 and has participated in various national, European, and industrial research projects on real-time mixed-criticality systems, optimization strategies of runtime software for heterogeneous HPC and microservers, and FPGA emulation. She led Modelling and Simulation activities within the first phase of the European Processor Initiative (EPI) project. She is currently involved in the second phase of EPI on co-design and exploration. Her research interests cover combinatorial optimization and operational research techniques with a special focus on optimization problems for electronic design automation and high-performance embedded systems, as well as testing and security. She is the project leader of the working group "Optimized Integrated Circuit" funded by the French institution CNRS. She has been a SAMOS, SC, PMBS, and CoDit technical programs member. She has served as General Chair for Hipeac/Rapido 2023, 2024, and General Chair of the 50th Euromicro DSD/SEAA 2024 conference.

Jack Lange – Invited Speaker and Panelist

ORNL, Jack was an Associate Professor of Computer Science in the School of Computing and Information at the University of Pittsburgh. Prior to joining the faculty at Pitt, Jack received his Ph.D. from Northwestern University. During his career in academia, Jack's research has focused on systems software for high performance computing environments, including high performance virtual machine managers (VMMs) and hypervisors, novel operating systems (OS) architectures, shared memory frameworks, and overlay networks. This work has served as the foundation for several Department of Energy research projects including the Hobbes Exascale operating system and runtime (OS/R.).

Terry Jones – Chair

Terry Jones is a Senior Research Staff member at Oak Ridge National Laboratory (ORNL) where he has worked since 2008 in the Computer Science and Mathematics Division (CSMD) as a Computer Scientist. Prior to that, he held a Computer Scientist position at Lawrence Livermore National Laboratory (LLNL). Terry earned a Master of Computer Science degree from Stanford University. Terry's research interests include system software for high performance computing, runtime systems and middleware, parallel and distributed architectures; performance monitoring; memory and storage systems; distributed clock synchronization, and resilience for complex distributed systems.

Estela Suarez – Co-organizer and Moderator

Dr. Estela Suarez is research group leader at the Jülich Supercomputing Centre from Forschungszentrum Jülich, which she joined in 2010. Since 2022 she is also Professor for High Performance Computing at the University of Bonn. Her research focuses on HPC system architectures and codesign. As leader of the EU-funded DEEP project series she has driven the development of the Modular Supercomputing Architecture, including hardware, software and application implementation and validation. Additionally, since 2018 she leads the codesign efforts within the European Processor Initiative. She holds a PhD in Physics from the University of Geneva (Switzerland) and a Master degree in Astrophysics from the University Complutense of Madrid (Spain).

Jim Ang – Co-Organizer

James is the Chief Scientist for Computing in the Physical and Computational Sciences Directorate at Pacific Northwest National Laboratory, where he serves as the lab lead for the DOE Office of Science (DOE/SC), Advanced Scientific Computing Research (ASCR) Program. PNNL's ASCR portfolio includes over 20 R&D projects in applied mathematics, computer science, advanced architectures, and computational modeling and simulation. His computing leadership role also intersects with foundational technology challenges associated with microelectronics and semiconductors. James helped organize the panel on co-design for beyond exascale at the DOE/SC workshop on Basic Research Needs for Microelectronics; served on the executive committee for the Semiconductor Research Corporation Decadal Plan; and was appointed by the U.S. Commerce Secretary to serve on the NIST Industrial Advisory Committee to provide input on R&D gaps for the CHIPS and Science Act. James has a BA in Physics from Grinnell College, a BS in Mechanical Engineering from the University of Illinois at Urbana-Champaign, and MS and PhD degrees in Mechanical Engineering from the University of California at Berkeley.

Jim Brandt – Co-Organizer

James (Jim) Brandt is a Distinguished Research Staff Member (Computer Scientist) at Sandia National Laboratories. Jim's research interest for the past two decades has been in holistic data-driven analysis of HPC eco-system resource utilization and state. He leads the development effort for Sandia's Lightweight Distributed Metric Service (LDMS) which has been in production use for a decade and installed on largescale systems across the DOE and NSF. Jim also leads SNL's AppSysFusion project, which enables run time combined application+system monitoring, through the interoperability of LDMS with other tools including Kokkos, Darshan, and Caliper. Jim leads work in the area of application of AI/ML to modeling and optimization of application resource utilization and anomaly detection. Jim has a M.S. degree in Computer Engineering from Santa Clara University and a B.S in Physics from California State University Hayward.

Mike Jantz – Co-Organizer

Mike Jantz is an Associate Professor of Computer Science at the University of Tennessee, Knoxville. At UT, Mike leads the CORSys research group, which aims to design and build innovative system tools and techniques to achieve faster, safer, and more efficient execution on modern and emerging architectures. His group has conducted and published research on a variety of topics related to computing performance and efficiency, program profiling and analysis, runtime data management, and dynamic compilation. His

work is supported by a number of government and industrial institutions, including the National Science Foundation (NSF), the U.S. Department of Energy, and Intel Corporation. In 2020, he received the NSF CAREER award for his proposal on application guided data management for complex memory systems.

Ann Gentile – Co-Organizer

Ann is a Manager in Sandia's High Performance Computing (HPC) Development Department which develops innovative management methodologies to improve the utilization of leading and next-generation computing systems across the world. Ann's research interests are in HPC Monitoring and Analysis and Dynamic, Resource-Aware Computing based on system and application monitoring data.

Appendix 3 – Organizing Committee and Program Committee

Workshop Organizing Committee

- Terry Jones - Oak Ridge National Laboratory, USA
- Estela Suarez- Jülich Supercomputing Centre & University of Bonn, Germany
- Ann Gentile - Sandia National Laboratories, USA
- Michael Jantz - the University of Tennessee, USA

Workshop Program Committee

- Jim Brandt - Sandia National Laboratories, USA
- Florina Ciorba - University of Basel, Switzerland
- Hal Finkel - US DOE office of Advanced Scientific Computing Research, USA
- Lin Gan - National Supercomputing Center, Wuxi, China
- Maya Gokhale - Lawrence Livermore National Laboratory, USA
- Thomas Gruber - Friedrich-Alexander-University Erlangen-Nuernberg, Germany
- Oscar Hernandez - nVidia, USA
- Jesus Labarta - Barcelona Supercomputing Center, Barcelona, Spain
- Hatem Ltaief, King Abdullah University of Science and Technology (KAUST), Saudi Arabia
- Yutong Lu - Director of National Supercomputing Center in Guangzhou, China
- Esteban Meneses - Costa Rica National High Technology Center, Costa Rica
- Bernd Mohr - Jülich Supercomputing Centre, Germany
- David Montoya - Trenza, USA
- Dirk Pleiter - KTH Royal Institute of Technology, Sweden
- Mitsuhsa Sato - Riken, Japan
- Martin Schulz - Technical University of Munich, Germany

Appendix 4 – Attendees & Workshop Photographs

We noted approximately 60 in-person participants with a few participants coming and going during the morning; we were unable to collect information on remote participants. We collected names and email addresses for our attendees.

Last year, our SC'23 attendance was 64, and the year before that we had 61 in-person participants.



These pictures show the SQCS'24 room format. The room was of an irregular shape, but provided plenty of room.

Appendix 5 – Flash Talks

Robert Keßler, University of Cologne

INTERNATIONAL SYMPOSIUM OF QUANTITATIVE CODESIGN OF SUPERCOMPUTERS, NOVEMBER 2024

1

Challenges to Overcome the Disparity Between Resource Allocation and Utilization of HPC Clusters on Reconfigurable Architectures

Robert Keßler, Simon Volpert, Viktor Achter, Lutz Schubert, and Stefan Wesner

I. MOTIVATION

While High Performance Computing (HPC) systems used to be quite homogeneous until a few years ago and usually only consisted of CPU nodes for a long time, today's clusters are usually composed at least of both CPU-only and GPU-booster nodes. Future systems like Europe's first exascale computer JUPITER at Forschungszentrum Jülich (FZJ) plan to incorporate for later extensions even a broader variety of compute resources such as Quantum units or accelerators like Field Programmable Gate Arrays (FPGAs) [13]. However, the increasing heterogeneity not only affects the computing resources but also the different available memory types such as Non-Volatile Memory (NVM) and High-Bandwidth Memory (HBM) or even the hierarchical memory systems. I/O resources have not been part of this consideration so far, but should not be neglected, as gateways within the cluster or the storage nodes can also differ.

The reasons for this steadily increase in heterogeneity are manifold, first of all (i) particular HPC applications have a reduced time to result or energy footprint on dedicated hardware and usually no longer only involve monolithic computations, but consist of workflows with different requirements especially with regard to the convergence of HPC and Artificial Intelligence (AI); furthermore (ii) extreme scaled problems in many cases can benefit strongly from highly parallel hardware architectures such as accelerators and GPUs; and ultimately (iii) the operators are subject to ecological and economical constraints. The very static resource allocation model of contemporary HPC systems is not suitable for such a high degree of heterogeneity, since it fails to cope with the changing resource demands of the dynamic workflows and therefore reaches its efficiency limits. In this respect, more adaptable and dynamic systems promise better performance and utilization of this resource diversity, but introduce new challenges at potentially all HPC system layers including the hardware, system software, cluster management, programming models, application environments, as well as the applications themselves.

A key concern in our previous work [10] and of our further ongoing evaluations is the disparity between allocated resources and their actual utilization in such environments. A potential under-utilization of an HPC system can essentially result from two factors (a) inefficient scheduling due to static resource allocation (cf. Figure 1a); and (b) the actual physical utilization of the resources allocated to a specific job (cf. Figure 1b). This inefficiency further amplifies the demand for reconfigurable architectures. In line with our ongoing work, this paper examines the challenges of bridging this gap. Therefore, we explore ongoing research and possible solutions.

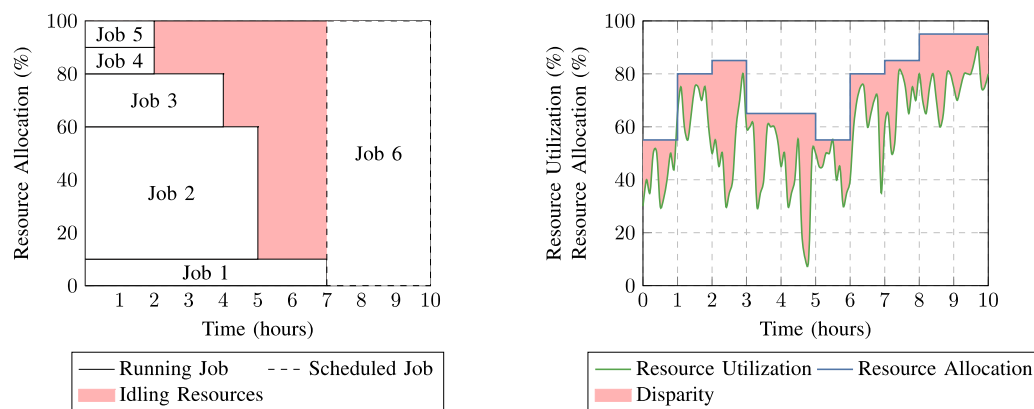


Fig. 1: (a) Allocation under-utilization and (b) Resource under-utilization

Flash Talks: Robert Keßler

INTERNATIONAL SYMPOSIUM OF QUANTITATIVE CODESIGN OF SUPERCOMPUTERS, NOVEMBER 2024

2

II. RELATED WORK

As mentioned in the previous section, the transition from a static to a dynamic HPC operating model requires adjustments along the entire system stack. In the following, we provide a brief overview of what we consider to be among the most relevant work across these research areas in recent years. Furthermore have Taraff et al. recently published an article that thoroughly presents experiences, challenges and opportunities that arise in the context of malleability for HPC systems [14].

a) Application Malleability & Programming Models: Both applications and the programming models or frameworks used in them for parallelization must in principle be capable of coping with changing resources [8], [11], [12].

b) Monitoring and Operational Data Analytics (MODA): The dynamic adaptation of resources represents a mutual relationship between applications and the hardware/system on which they run. On the one hand, an adjustment may be made due to external conditions such as power caps or the failure of components or, on the other hand, it may be intrinsically driven by the application itself because its demands have changed. In both cases, a comprehensive monitoring of the relevant application metrics and the status of the physical hardware is required in order to provide appropriate feedback and to carry out any necessary changes to address the dynamic circumstances [3], [4].

c) Workload-Aware Scheduling: In order to efficiently utilize the increasingly heterogeneous infrastructure, but also to comply with sustainability and power constraints, the applications must be profiled to schedule jobs based on their resource needs and characteristics [1], [5], [9].

d) Dynamic Resource Provisioning: Flexible and dynamic resource provisioning is twofold, on the one hand it focuses on developing algorithms that can adjust resource allocation based on real-time workload demands, otherwise the Resource and Job Management System (RJMS) must be able to handle the effects of added or failed resources at runtime [2], [7].

III. CHALLENGES

We want to emphasize once again that the transition to dynamic and reconfigurable supercomputers fundamentally requires changes and adaptations at all levels of the system stack. With regard to our research, we will initially concentrate our efforts on the challenges described in more detail below.

a) Reconfiguring Workloads: Traditional RJMS like Slurm, while effective for homogeneous workloads, struggle to cope with the dynamic nature and diverse demands of modern scientific applications and elastic workflows [6]. Therefore, a coherent interface from the application to the RJMS is required, which can be used to reconfigure both the type and quantity of resources required or vice versa grantable at runtime. Modifying the resources assigned to an active job and its processes, making these aware of the new resources and enabling their seamless utilization is a complex effort.

b) Utilizing Reconfigurable Architectures: The ability to adapt hardware configurations on-the-fly, while beneficial, introduces significant complexity in resource allocation, job scheduling, and data movement strategies. Apart from the familiar parameters of static operating models additional factors impact a scenario for dynamic resource reconfiguration including workload priority, malleability criteria and the reconfiguration overhead. The larger number of boundary conditions that must be taken into account in such an operating model results in a significantly more complex optimization problem for the scheduler. The scheduler must be able to estimate whether a reconfiguration is really efficient.

c) Limitations of Current Resource Management Techniques: Ultimately an RJMS should be capable of handling changes to the resources in the system at runtime. For example in the event that resources such as switches or entire nodes fail, it is very difficult to adapt to such circumstances in a static operating model. Awareness of the possibly dynamic changes in resources and their resulting reconfiguration as well as reallocation could counteract this.

d) Workload Isolation: Besides the dynamic allocation of resources, it will also be necessary to co-locate jobs to a greater extent on shared resources to close the allocation-utilization disparity gap. For this purpose, it must be ensured that jobs that share resources do not impair each other's performance too much due to the neighbor-noisy effect. On the other hand, it must also be ensured in the course of a reallocation that no performance degradation of other jobs occurs, which for example might be caused by the massive shifting of data and the resulting network overload.

IV. OPPORTUNITIES

Several approaches contribute to possibly bridging the disparity between resource allocation and utilization in HPC clusters on reconfigurable architectures, the ones we will focus our research in the first place are described in the following.

A. Co-location via Virtualization

While virtualization offers a degree of resource isolation and flexibility, it often introduces performance overhead, particularly in HPC workloads that require high-bandwidth and low-latency communication.

Challenge: Finding the right balance between virtualization benefits, sufficient isolation and performance penalties.

Approach: Exploring lightweight virtualization techniques, leveraging hardware-assisted virtualization features, and investigating containerization or sandboxing solutions.

Flash Talks: Robert Keßler

INTERNATIONAL SYMPOSIUM OF QUANTITATIVE CODESIGN OF SUPERCOMPUTERS, NOVEMBER 2024

3

B. Elastic Workloads

Scientific applications increasingly exhibit dynamic resource needs, scaling up or down based on computational demands or even for more suitable hardware to execute a certain task. This elasticity poses a significant challenge for traditional resource allocation schemes designed for static resource requirements.

Challenge: Predicting and adapting to fluctuating and diverse resource demands in real time.

Approach: Developing dynamic scheduling algorithms and implementing runtime resource scaling mechanisms that provide a interface between RJMS and jobs to reconfigure the job parameters.

C. Backfilling with Foreign Workloads

Backfilling as in using idle resources to run lower-priority jobs can improve utilization. However, effectively integrating diverse workloads like big data analytics, serverless functions, and AI applications alongside traditional HPC jobs requires careful consideration of their unique characteristics and resource profiles.

Challenge:: Ensuring fairness and minimizing interference between different types of workload while maximizing resource utilization. This relates to the virtualization challenge mentioned above but also requires much more sophisticated and enhanced scheduling algorithms as well as an awareness of the HPC system's state.

Approach:: Develop workload-sensitive scheduling policies, explore priority-based resource allocation strategies, and investigate techniques for resource isolation and quality of service guarantees including application and system monitoring pipelines to provide an adequate feedback loop.

V. SUMMARY

Bridging the gap between resource allocation and utilization in HPC clusters on reconfigurable architectures is crucial for maximizing scientific output, achieving cost-effectiveness and becoming more sustainable. While challenges persist, ongoing research in dynamic resource allocation, workload-aware scheduling, and reconfigurable computing middleware offers promising avenues for improvement. Future work will likely focus on developing more intelligent and adaptive resource management systems that can effectively handle the increasing complexity and heterogeneity of workloads in future HPC environments.

REFERENCES

- [1] Achilleas P. Achilleos, Kyriakos Kritikos, Alessandro Rossini, Georgia M. Kapitsaki, Jörg Domaschka, Michal Orzechowski, Daniel Seybold, Frank Griesinger, Nikolay Nikolov, Daniel Romero, and George A. Papadopoulos. The cloud application modelling and execution language. 8(1):20.
- [2] Rajat Bhattarai, Howard Pritchard, and Sheikh Ghafoor. Dynamic Resource Management for Elastic Scientific Workflows using PMix. In *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 686–695.
- [3] Francieli Boito, Jim Brandt, Valeria Cardellini, Philip Carns, Florina M. Ciorba, Hilary Egan, Ahmed Eleliemy, Ann Gentile, Thomas Gruber, Jeff Hanson, Utz-Uwe Haus, Kevin Huck, Thomas Ilsche, Thomas Jakobsche, Terry Jones, Sven Karlsson, Abdullah Mueen, Michael Ott, Tapasya Patki, Ivy Peng, Krishnan Raghavan, Stephen Simms, Kathleen Shoga, Michael Showerman, Devesh Tiwari, Torsten Wilde, and Keiji Yamamoto. Autonomy Loops for Monitoring, Operational Data Analytics, Feedback, and Response in HPC Operations. In *2023 IEEE International Conference on Cluster Computing Workshops (CLUSTER Workshops)*, pages 37–43.
- [4] J Brandt, A Gentile, C J Morrone, K S Shoga, T Tucker, and J Hanson. Evaluating and Influencing Extreme-Scale Monitoring Implementations.
- [5] Danilo Carastan-Santos, Georges Da Costa, Millian Poquet, Patricia Stolf, and Denis Trystram. Light-Weight Prediction for Improving Energy Consumption in HPC Platforms. In Jesus Carretero, Sameer Shende, Javier Garcia-Blas, Ivona Brandic, Katzalin Olcoz, and Martin Schreiber, editors, *Euro-Par 2024: Parallel Processing*, pages 152–165. Springer Nature Switzerland.
- [6] Mohak Chadha, Jophin John, and Michael Gerndt. Extending SLURM for Dynamic Resource-Aware Adaptive Batch Scheduling. In *2020 IEEE 27th International Conference on High Performance Computing, Data, and Analytics (HiPC)*, pages 223–232.
- [7] Marco D’Amico, Marta Garcia-Gasulla, Víctor López, Ana Jokanovic, Raúl Sirvent, and Julita Corbalan. DROM: Enabling Efficient and Effortless Malleability for Resource Managers. In *Workshop Proceedings of the 47th International Conference on Parallel Processing, ICPP Workshops ’18*, pages 1–10. Association for Computing Machinery.
- [8] Jorge Ejarque, Rosa M. Badia, Loïc Albertin, Giovanni Aloisio, Enrico Baglione, Yolanda Becerra, Stefan Boschert, Julian R. Berlin, Alessandro D’Anca, Donatello Elia, François Exertier, Sandro Fiore, José Flich, Arnau Folch, Steven J. Gibbons, Nikolay Koldunov, Francesc Lordan, Stefano Lorito, Finn Løvholt, Jorge Macías, Fabrizio Marozzo, Alberto Michelini, and Marisol Monterrubio-Velasco. Enabling dynamic and intelligent workflows for HPC, data analytics, and AI convergence. 134:414–429.
- [9] Brandon Kammerdiener, J. Zach McMichael, Michael R. Jantz, Kshitij A. Doshi, and Terry Jones. Flexible and Effective Object Tiering for Heterogeneous Memory Systems. In *Proceedings of the 2023 ACM SIGPLAN International Symposium on Memory Management*, pages 163–175. ACM.
- [10] Robert Keßler, Simon Volpert, and Stefan Wesner. Towards improving resource allocation for multi-tenant hpc systems: An exploratory hpc cluster utilization case study, September 2024. Presented at the Sustainable HPC SOP Workshop, IEEE Cluster 2024.
- [11] Iker Martín-Álvarez, José I. Aliaga, Maribel Castillo, and Sergio Iserte. Configurable synthetic application for studying malleability in HPC. In *2023 31st EuroMicro International Conference on Parallel, Distributed and Network-Based Processing (PDP)*, pages 128–135.
- [12] Iker Martín-Álvarez, José I. Aliaga, Maribel Castillo, and Sergio Iserte. Proteo: A framework for the generation and evaluation of malleable MPI applications. 80(15):23083–23119.
- [13] Estela Suarez, Norbert Eicker, and Thomas Lippert. Modular Supercomputing Architecture: From Idea to Production. In *Contemporary High Performance Computing*. CRC Press.
- [14] Ahmad Tarraf, Martin Schreiber, Alberto Cascajo, Jean-Baptiste Besnard, Marc-André Vef, Dominik Huber, Sonja Happ, André Brinkmann, David E. Singh, Hans-Christian Hoppe, Alberto Miranda, Antonio J. Peña, Rui Machado, Marta Garcia Gasulla, Martin Schulz, Paul Carpenter, Simon Pickartz, Tiberiu Rotaru, Sergio Iserte, Victor Lopez, Jorge Ejarque, Heena Sirwani, and Felix Wolf. Malleability in Modern HPC Systems: Current Experiences, Challenges, and Future Opportunities. pages 1–14.

J. Zach McMichael, University of Tennessee

Work-in-Progress: VMem: A User-level Runtime for Enabling Fine-Grained Application Control of Physical Memory

J. Zach McMichael and Michael R. Jantz
University of Tennessee, Knoxville
[jmcMicha,mrjantz]@vols.utk.edu

1 Introduction

As scientific computing increasingly relies on data-driven analyses and AI, demands for higher-throughput, lower-latency processing of larger and larger sets of data in memory are being driven to new heights. At the same time, the need for high-density sharing has led to the widespread adoption of supercomputer configurations with large amounts of memory attached to each node and connected through efficient networking resources. New media technologies, such as high bandwidth memories, low-power and non-volatile RAMs, and many others, new accelerator options, including processing-in-memory (PIM), and new memory interconnect options, including the Compute Express Link (CXL), are bringing rich opportunities for addressing the diverse needs of applications under various cost, performance, and power constraints. In response to these trends, most supercomputing systems now include a heterogeneous mix of memory devices and organizations, which can enable the combined benefits of their unique capabilities and support the diverse set of workloads that are present in modern supercomputing centers.

However, new data management strategies are needed to capitalize on the different strengths of each type of memory. Specifically, the system must be able to efficiently match application data to the type of memory that best suits its purpose for the optimal amount of time. Applications, as the primary generators of memory usage, are well-suited to guide such tasks. However, conventional data management approaches are limited in how they use application-level information due to some longstanding divisions that have traditionally been present during memory management. While applications control the structure and usage of program data, physical memory management tasks, including the allocation of memory resources, management of application page tables, and translations of virtual to physical addresses, are under the purview of the operating system and hardware, and thus, proceed with little or no knowledge of application behavior. This semantic gap limits optimization opportunities and can lead to inefficiencies in how application data is mapped to memory with different performance and capabilities.

To address these shortcomings, this work proposes a new memory management framework, called VMem, that provides applications with fine grain controls to conduct their own data tiering and physical memory management. VMem employs standard system facilities to acquire different types of physical memory resources and empowers applications to map, use, and manipulate these resources for their own program data. Its Linux-based implementation does not require any kernel modifications or non-standard hardware to provide this core functionality. Moreover, applications can use many of the VMem features and optimizations automatically, and without updating or recompiling program source code, by linking a custom allocator that invokes VMem to manage the application heap.

This work presents a high-level description of the design of our VMem framework, including its two primary components: the VMem Server and VMem Runtime. To demonstrate the potential of this approach, it then presents an example that shows how applications could use VMem to accelerate data movement between different types of memory in a heterogeneous architecture. As VMem remains a work-in-progress, this paper closes with a discussion of opportunities for improving memory utilization and management with VMem and our planned future research.

2 Design Overview of VMem

Figure 1 depicts the main components of VMem and their interactions. It primarily consists of two new pieces of software:

1. The VMem Server acquires free memory resources from the different types of memory that are present on the platform and organizes these pages into shared memory pools. These pools are then used to serve the memory needs of connected VMem processes.
2. The VMem Runtime is a lightweight runtime that connects an application process to the VMem server, thereby enabling it to participate in shared memory management with VMem. It implements

J. Zach McMichael

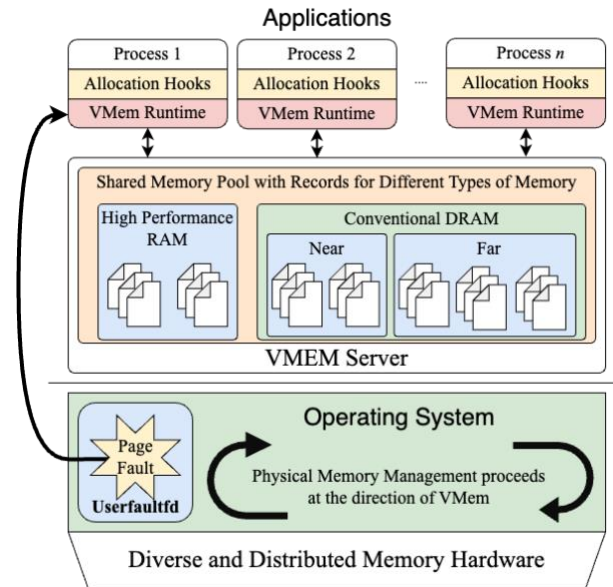


Figure 1: Design overview of the VMem framework.

an API that allows applications to register virtual address ranges with VMem and control allocation and recycling of the memory resources exposed by the VMem server within these ranges.

The VMem framework, implemented in Linux, does not require any kernel modifications on non-standard configuration options to enable fine-grained memory management features for participating applications. Rather, it leverages standard Linux system calls, including `memfd_create`, `mmap`, and `userfaultfd`, to implement its core functionality. Let us next describe how these facilities are used to implement VMem.

2.1 The VMem Server

The VMem Server provides a holistic view of the available memory resources and exposes these resources to VMem Runtime processes through a set of shared memory pools. To initialize the shared memory pools, the VMem server creates an anonymous file corresponding to each physical memory device tier using the `memfd_create` system call. It then maps each anonymous file into its own address space as shared memory and populates the shared mappings with physical memory corresponding to the appropriate tier of memory. For this operation, VMem employs `mbind` to ensure that the shared ranges are mapped to different types of memory. Hence, it is able to differentiate memory device tiers that are distinguished in the underlying platform as separate NUMA nodes. Next, the shared ranges are subdivided into pages, called *vpages*, which are then inserted into free lists corresponding to each tier of physical memory. The *vpages* and their associated lists are created and mapped into the VMem Server as shared memory, so that they can be accessed and manipulated directly by processes that use the VMem Runtime. Note also that, in addition to metadata for their associated data structures, each *vpage* maintains information about the anonymous file and offset within the file from which it was originally mapped.

2.2 The VMem Runtime

The VMem Runtime is implemented as a shared library which is intended to be dynamically linked into a running process before the application invokes its main execution routine. During initialization, the

J. Zach McMichael

VMem Runtime will connect to the (previously initialized) VMem Server running on the same platform and collect information regarding its shared memory mappings, including the anonymous files corresponding to each type of memory and their associated vpages. To use these shared memory resources in the application process, the VMem Runtime leverages the `userfaultfd` facility in Linux. `userfaultfd` enables user-level processes to intercept and handle page faults corresponding to specific ranges of virtual memory by issuing a callback from the kernel to the user process whenever a fault occurs within the specified range. Thus, the VMem Runtime includes an API that allows applications to register anonymous memory regions with `userfaultfd`. When the runtime receives a page fault callback from `userfaultfd`, it will then query the shared vpage structures to find an appropriate vpage to satisfy the fault. Specifically, it uses `mmap` to map the device file and offset associated with the selected vpage into the virtual addresses corresponding to the fault, thereby satisfying the fault with memory shared by the VMem Server.

2.3 Design Features

With this design, the VMem framework enables significant new flexibility and control over how applications manage their own physical memory resources. Applications can design and implement custom vpage allocation and recycling routines, create alternative data tiering strategies, and experiment with new memory management optimizations, without requiring any further updates to the application or operating system. One important feature of flexibility is that the vpage size does not need to be equal to the host platform page size, but rather, can be optionally set as any multiple of the host page size. In this way, applications can control overheads associated with `userfaultfd` by configuring VMem to use larger vpage sizes. For our initial studies, we have only implemented a simple free list allocator for vpage faults and have not experimented with alternative tiering strategies. To demonstrate its potential, we have used this framework to create and evaluate an optimization for accelerating data migration in systems with diverse and/or distributed memory modules, as described next.

3 Reducing Data Migration Costs with VMem

Data migration costs are a significant challenge for complex memory platforms. Before moving any program data from one type of memory to another, the system must suspend any application threads that may access the data to prevent inconsistencies due to data races. Additionally, page tables must be updated and MMU caches (i.e., TLBs) flushed before the suspended activities can resume. On modern platforms, these costs are substantial and can limit the performance of applications that must adapt to the underlying hardware. Some previous works (e.g., [1]) have proposed to reduce these costs by separating data copies from page table update operations during memory migration. In these schemes, the system optimistically write-protects and copies the data to the target device asynchronously and then discards the copied data if a write is detected. While this approach can be effective for some workloads, modern Linux platforms do not implement it, supposedly because this feature would increase the complexity of performance critical memory management code.

For this work, we extended the VMem Runtime with data migration facilities that copy program data prior to and separately from updating the application page tables. To demonstrate the potential of this approach, we deployed it with a simple test program on our Intel-based platform. This platform contains a single Intel Xeon Gold 6246R CPU (codename: Cascade Lake) with two types of memory: a faster tier of conventional DDR4 DRAM and a slower tier of non-volatile Optane DC RAM. The test program creates and initializes an anonymous memory region of size 1 GB on each type of memory and then migrates the data from each type of memory to the other. We compared our custom VMem migration routines to the default Linux memory management and migration stack (i.e., with the system allocator and `mbind`) as well as a VMem routine that performs both data copy and page table updates together. Additionally, we tested our approach with both 4 KB and 2 MB page sizes, both on the host platform and within VMem (i.e., the vpage size). Our tests show that VMem effectively separates data copy and page table operations. If an application continues to access virtual addresses that have been copied, but not remapped, these accesses will be resolved to physical memory corresponding to the original copy. Continued access to these addresses will resolve to the copied only after the application page tables have also been remapped.

Table 1 presents the average throughput (in GB/s) of the data migration operations for each configuration. The results allow several interesting observations. First, data migration throughput improves

J. Zach McMichael

	Linux Default		VMem Default			VMem Separate		
Page Sizes	4 KB	2 MB	4 KB (H) 4 KB (V)	4 KB (H) 2 MB (V)	2 MB (H) 2 MB (V)	4 KB (H) 4 KB (V)	4 KB (H) 2 MB (V)	2 MB (H) 2 MB (V)
Migration TP (GB/s)	1.47	2.96	0.47	1.43	3.3	1.63 copy 0.65 remap	1.7 copy 8.6 remap	3.3 copy 474.2 remap

Table 1: Average throughput of migration operations with VMem (higher is better). The Linux Default and VMem Default configurations perform both data copy and page table updates together and therefore show only the throughput of the complete migration. The VMem Separate configuration performs data copy and page table updates in separate routines and presents the throughput of each routine.

with larger page sizes, but even with 2 MB page sizes, throughput is less than 3.3 GB/s in every configuration. In cases where the VMem framework uses both 4 KB host (Linux) pages and 4 KB vpages, performance is somewhat worse than the default Linux facilities. This slowdown is primarily due to additional transfers between the host OS and VMem runtime to conduct operations for each vpage. Indeed, the initial page faults themselves are also much slower in these configurations, incurring more than 20x slowdown for each 4 KB page fault. However, most of this overhead can be eliminated by batching operations with larger vpage sizes, as can be seen in configurations that use 4 KB host pages and 2 MB vpages. Lastly, observe that the VMem Separate configurations show significant potential to increase data migration throughput in some scenarios. While copy throughput is mostly similar to configurations that perform data copy and page table remapping together, this operation can now be performed asynchronously, in many cases. For configurations with larger vpage sizes, operations that remap and free data in a particular memory device are now 5.8x to over 160x faster than the standard data migration operations. Thus, VMem has potential to unlock significant performance and efficiency improvements on complex memory architectures by enabling applications to perform time-consuming data copy operations asynchronously, rather than in the critical path of program execution.

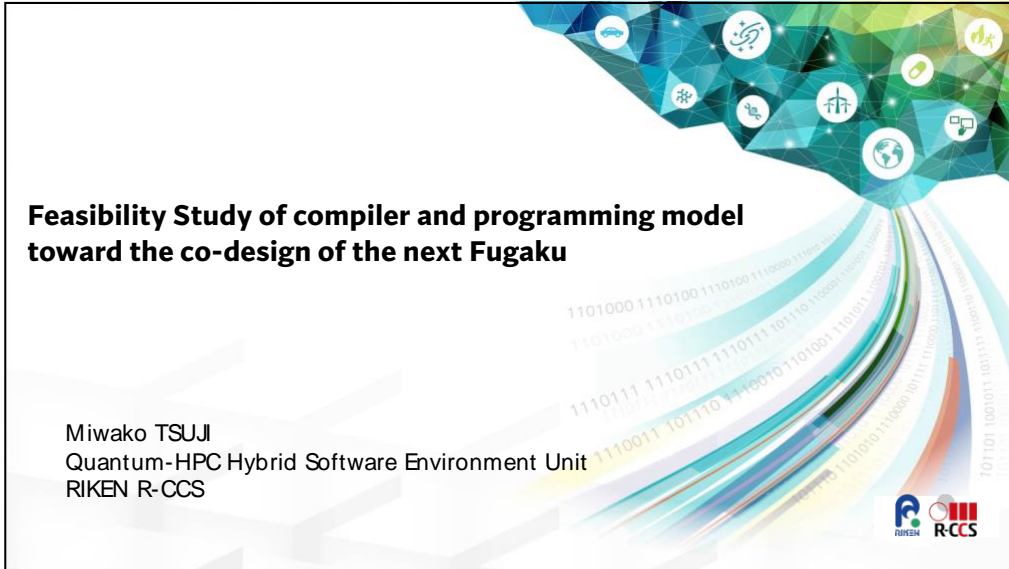
4 Conclusions

This work presents VMem: a novel server and runtime framework for enabling application control of physical memory resources. VMem allows applications to exert fine-grain control over low-level memory management tasks, including translation of virtual addresses, physical memory allocation and recycling, and memory-to-memory migration. Its unique design provides these features without require any kernel modifications, custom hardware, or non-standard system configurations. To demonstrate the potential benefits of this approach, this work uses VMem to implement an alternative data migration scheme that separates data copy from page table updates. The evaluation shows that this scheme could substantially reduce pause times for applications that migrate data frequently on complex memory architectures. While VMem is still a work-in-progress and is under active development, it has significant potential to unlock new optimizations and efficiencies by reducing the semantic gap between the system and application software during memory management. VMem source code is available upon request.

References

- [1] DASHTI, M., FEDOROVA, A., FUNSTON, J., GAUD, F., LACHAIZE, R., LEPEERS, B., QUEMA, V., AND ROTH, M. Traffic management: a holistic approach to memory placement on numa systems. In *ACM SIGPLAN Notices* (2013), vol. 48, ACM, pp. 381–394.

Miwako Tsuji, Riken



Feasibility Study of compiler and programming model toward the co-design of the next Fugaku

Miwako TSUJI
Quantum-HPC Hybrid Software Environment Unit
RIKEN R-CCS

1

Feasibility Study for the Next Generation Supercomputer

Feasibility Studies on Next-Generation Supercomputing Infrastructures has been conducted from August 2022, commissioned by MEXT (the Ministry of Education, Culture, Sports, Science and Technology).

Anticipating the work to develop the Fugaku-**next** supercomputer, the projects will clarify the needs of computational sources for

- various fields of science and technology
- Society 5.0
- Advanced Digital Twin
- Data-driven science, BigData at scale
- FLOPS to Byte

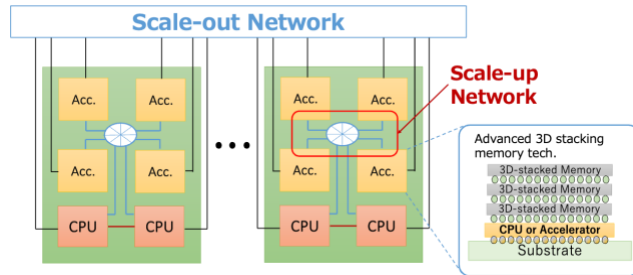
- **Architecture**
- **Research on System Software and Library**
- **Research on Applications**

2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031
NGACI roadmap discussion		FS CFP	Feasibility Study		Preliminary Design	Detailed Design		Production, Installation			

2

Miwako Tsuji

Architectural Direction toward Next-Generation Computational Infrastructure



- High bandwidth and heterogeneous node architecture design
- Significant increase in relative memory bandwidth using 3D stacked memory technology
- System network suitable for both strong scaling and weak scaling
- Massively parallel system with tens of thousands of accelerator sockets

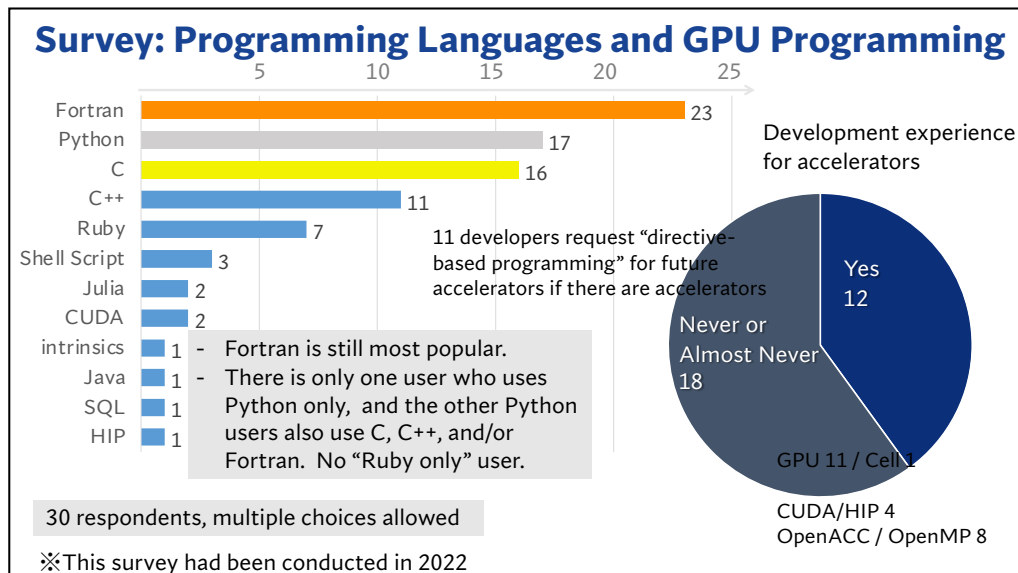
3

Compiler / Programming Model Sub Working Group

- Investigates following items (w/ collaborations with the architecture group)
 - Needs and trends of application developments in terms of programming languages
 - Existing code generation techniques including open-source compilers
 - Programming frameworks enabling high portability including performance portability across various architectures
 - Programing tools such as tracers, profilers and debuggers
- Clarifies programming environments for the next-generation supercomputers

4

Miwako Tsuji



5

Ongoing and Future works: From CPU to CPU+Accelerator

- From Fugaku, a massively parallel CPU-only cluster, to Fugaku-next, CPU + Accelerator
- How to support applications' porting from CPU to “CPU+Acc. from the viewpoint of the programming model and compilers
 - performance portable programming, such as Kokkos, Raja, ..
 - programming environment matrix for different systems
 - NVIDIA, AMD, Intel GPUs
 - OpenMP, OpenACC, DPC++, CUDA/HIP, stdpar
 - case studies
 - OFP (KNL) to OFP2 (GH) in JCAHPC
 - Summit (NVIDIA) to Frontier (AMD) in ORNL
 - etc...
 - role of GPU vendor, Integrator, and OSS communities for existing systems
 - Fortran/FORTRAN

6