HPC.NRW

HPC.NRW - Technology Scouting

Report December 2025

Contents

1 Introduction

This report aims to introduce researchers and users to new technology trends and ideas in the broad field of high-performance computing, covering numerous topics from new hardware to interesting software, searching for potential directions of research and evaluation. This collection of topics was compiled by participants of the **HPC.NRW** project from different HPC centres. Each section gives a clear rationale, explaining why the project is noteworthy or interesting. It also highlights the motivation and technical details, as well as the potential pitfalls and drawbacks of the given concept.
In this edition of the report we give a brief summary of the high lights from the ISC High Performance 2025 conference in Hamburg and will introduce the readers to

- *NVIDIA BlueField DPUs* – NVIDIA's novel solution for improving latency and bandwidth for networking, storage and provisioning as well as improving security.
- *uv* – new tool for Python package and project management written in rust
- *micro* – simple and intuitive commandline text editor written in *Go*
- *Photonic Accelerators* – new device to accelerate matrix-vector multiplications through superposition of light beams in a grid, predominantly used for faster training of AI models. The calculations are performed with the speed of light but current devices lack in the precision of the calculations.
- *Tightly coupled heterogeneous systems* – as research and development continue to push the need for higher memory bandwidth and lower latency, the concept of tightly coupled heterogeneous systems has been introduced. This section shows the trends in developments of new hardware for heterogeneous systems based on example of NVIDIA's GH200 architecture.

2 Frontiers in ISC2025 High-Performance Computing: Chips, Accelerators, and Quantum Paradigms

2.1 Background

The ISC High Performance 2025 conference in Hamburg showcased major architectural and technological advances across the modern HPC stack, including support for large-scale AI workloads, from advanced semiconductors to emerging specialized compute paradigms such as quantum and neuromorphic architectures. Central to this year's discussions were the next-generation Intel and AMD CPUs, NVIDIA's evolving GPU roadmap spanning the Blackwell and Rubin architectures, and advances in high-speed interconnects such as InfiniBand XDR. Together, these developments indicate a shift toward heterogeneous and energy-efficient computing platforms optimized for both simulation-driven HPC applications and AI workloads. Beyond conventional architectures, the conference highlighted increasing interest in hybrid quantum–classical integration, where quantum processors execute specific algorithmic subroutines while classical HPC systems manage orchestration and bulk computation, and neuromorphic co-processing for the efficient execution of specialized workloads. For HPC.NRW, these trends emphasize the need for a modular and scalable infrastructure strategy that balances performance and sustainability, while fostering the expertise required to leverage emerging computing paradigms for scientific discovery and data-driven innovation.

2.2 Next-Generation CPUs (Intel Xeon 6 and AMD EPYC)

The CPU landscape for HPC and AI workloads is marked by major advancements from Intel and AMD, each pursuing distinct optimization strategies.

2.2.1 Intel Xeon 6 Processors

Launched in early 2025, Intel's Xeon 6 processors represent a significant evolution in server CPU design. The architecture combines performance cores (P-cores), optimized for high single-thread throughput, with efficient cores (E-cores), designed for high parallelism and energy-efficient execution. For HPC-adjacent and latency-sensitive workloads, this heterogeneous core design can improve task scheduling flexibility and overall performance-per-watt [5]. While many of Intel's public benchmarks highlight advantages in telecommunications and RAN processing, the underlying architectural improvements, including integrated acceleration units, improved power efficiency, and core-mix flexibility, are relevant for HPC environments that run mixed simulation, inference, and service-oriented workloads.

2.2.2 AMD EPYC Processors

AMD's EPYC series continues to set benchmarks in compute density, memory bandwidth, and I/O throughput. The latest generation offers up to 128 cores and 256 threads per socket, paired with 12-channel DDR5 memory and PCIe 5.0/CXL 1.1 connectivity [9]. These characteristics directly benefit HPC workloads by enabling high FP64/FP32 throughput and efficient scaling of memory-intensive simulations. Additionally, AMD's Infinity Guard introduces hardware-level security features such as Secure Encrypted Virtualization (SEV), which can be valuable in HPC contexts involving sensitive data, such as bioinformatics or clinical modeling. AMD's development roadmap further emphasizes chiplet-based scalability and architecture-level support for ML acceleration, targeting improved performance-per-watt and system-level efficiency [9].

2.3 Advanced GPUs and AI Accelerators (NVIDIA Blackwell, Rubin, and beyond)

Graphics processing units (GPUs) and specialized accelerators for machine-learning workloads are key performance drivers in modern HPC environments [4]. In this context, the accelerators refer to hardware designed to accelerate machine-learning operations, including GPUs, NPUs, tensor processors, and other inference-focused architectures. NVIDIA, a dominant force in this sector, unveiled a multi-year roadmap during GTC 2025 that extends well into 2028.

2.3.1 Blackwell Ultra NVL72 Platform

Slated for release in late 2025, the Blackwell Ultra NVL72 features 72 Blackwell Ultra GPUs, each equipped with 288 GB of HBM3e memory. Integrated with 36 Arm Neoverse-based Grace CPUs, the configuration provides 20 TB of HBM and 40 TB of total fast memory per rack, offering a 1.5× performance uplift compared to the GB200 NVL72 (Grace Blackwell) platform [5].

2.3.2 Rubin and Feynman Architectures

The Rubin platform (expected 2H 2026) will leverage HBM4 and the new Vera CPUs, targeting a 3.3× improvement over Blackwell Ultra NVL72. Rubin Ultra (2H 2027) aims for an order-of-magnitude leap—up to 14× the performance of Blackwell Ultra by adopting 1 TB of HBM4e per GPU [2]. The Feynman architecture (anticipated 2028) is projected to exploit next-generation HBM for further bandwidth and energy gains.

2.4 Emerging Compute Paradigms (Neuromorphic and Quantum Hardware)

Beyond classical architectures, neuromorphic and quantum computing are emerging as specialized paradigms addressing domain-specific challenges.

2.4.1 Neuromorphic Computing

A poster presented at ISC 2025 titled "Neuromorphic Computing: Brain-Inspired Concepts, Platforms, and Their Role in HPC" proposed coupling neuromorphic modules with classical supercomputers for the energy-efficient processing of event-driven workloads [1]. These architectures excel at adaptive inference for low-power edge scenarios such as robotics, IoT, and sensory fusion but remain primarily experimental.

2.4.2 Quantum Computing

Quantum computing continues to mature as a complementary accelerator to HPC. Sessions at ISC 2025 emphasized hybrid integration over standalone quantum systems [2]. On-premises quantum deployments are gaining traction, with several HPC sites announcing plans to explore integration [7]. Fujitsu revealed a 256-qubit superconducting quantum computer, available since April 2025, and ongoing research toward a 10,000+ qubit system by 2030 [3].

Building on this trend, NVIDIA unveiled **NVQLink**, an open system architecture designed to tightly couple quantum processors (QPUs) with GPU-based supercomputers for high-throughput, low-latency hybrid execution [6]. NVQLink supports collaboration with 17 quantum processor manufacturers and five controller vendors, as well as research partnerships involving nine U.S. national laboratories, enabling unified orchestration of quantum and classical workflows via NVIDIA's CUDA-Q platform [6, 8]. This reinforces the roadmap toward large-scale hybrid systems where GPUs orchestrate calibration, error correction, and classical–quantum interaction loops.

Despite hardware advances, the software layer that connects QPUs to existing HPC schedulers and middleware remains a bottleneck [2]. This challenge arises because traditional HPC schedulers are designed for long-running, deterministic workloads, while QPU tasks consist of extremely short, probabilistic operations that require continuous calibration and error correction. As a result, orchestration between HPC nodes and quantum processors requires dedicated control interfaces and precise timing coordination that current middleware layers were not built to handle. In the near term, the focus remains on loosely coupled hybrid integration, where classical HPC systems invoke QPUs for specific subroutines rather than tightly coupled quantum-native workloads.

References

[1] ISC High Performance 2025. *Neuromorphic Computing: Brain-Inspired Concepts, Platforms, and Their Role in HPC*. Conference session, accessed: 2025-10-16. 2025. URL: https://isc.app.swapcard.com/event/isc-high-performance-2025/planning/UGxhbm5pbmdfMjU4Mjc3MQ%3D%3D.

[2] ISC High Performance 2025. *Quantum Computing at ISC 2025*. Conference report, accessed: 2025-10-16. 2025. URL: https://isc-hpc.com/quantum-computing-isc2025/.

[3] Fujitsu. *Fujitsu Starts Official Development of Plus-10,000 Qubit Superconducting Quantum Computer Targeting Completion in 2030*. Press release, accessed: 2025-10-16. 2025. URL: https://global.fujitsu/en-global/newsroom/gl/2025/08/01-01.

[4] IDTechEx. *Hardware for HPC, Data Centers, and AI 2025–2035: Technologies, Markets, Forecasts*. Accessed: 2025-10-16. 2025. URL: https://www.idtechex.com/en/research-report/hardware-for-hpc-and-ai-2025/1058.

[5] Intel. *MWC 2025: Intel Xeon 6 Proves Foundational to Network Infrastructure*. Accessed: 2025-10-16. 2025. URL: https://newsroom.intel.com/5g-wireless/mwc-2025-intel-xeon-6-proves-foundational-network-infrastructure.

[6] NVIDIA. *NVIDIA NVQLink — Connecting Quantum and GPU Computing for 17 Quantum Builders and Nine Scientific Labs*. Press release, October 28, 2025, accessed: 2025-10-29. 2025. URL: https://nvidia.com/en-us/solutions/quantum-computing/nvqlink/.

[7] Hyperion Research. *Special Analysis: Top 10 Predictions for the Global HPC-AI Community in 2025*. White paper, accessed: 2025-10-16. 2025. URL: https://hyperionresearch.com/wp-content/uploads/2025/02/Hyperion-Research-Special-Analysis-2025-Top-10-Predictions-February-2025.pdf.

[8] M. Swayne. *NVIDIA Launches NVQLink to Bridge Quantum and Classical Supercomputing*. The Quantum Insider, October 28, 2025, accessed: 2025-10-29. 2025. URL: https://thequantuminsider.com/2025/10/28/nvidia-launches-nvqlink-to-bridge-quantum-and-classical-supercomputing/.

[9] Wix.com. *Revolutionize Your Data Center in 2025 with AMD EPYC's Server Power*. Accessed: 2025-10-16. 2025. URL: https://cloudinformation.wixsite.com/processorinfo/post/revolutionize-your-data-center-in-2025-with-amd-epyc-s-unmatched-server-power.

3 *NVIDIA BlueField DPU*

3.1 Background

The BlueField (BF) Data Processing Unit (DPU) is NVIDIA's approach to the ongoing need for latency and bandwidth improvement, largely driven by the rapidly growing demands of AI-computing. In efforts to compete with Intel, NVIDIA started boosting its data-center and AI business in 2019 through incorporating the BF intellectual property when acquiring Mellanox Technologies [1].

The term DPU is often used interchangeably with Smart Network Interface Cards (SmartNICs). Essentially representing a *datacenter infrastructure* on a chip, DPUs combine a high-speed NIC with software-programmable cores. Given this compact design, running the control-plane logic requires the cores to be power-efficienct and scalable; therefore, DPUs rely on the Arm architecture. Unlike x86 processors (Intel/AMD), Arm cores use a RISC (Reduced Instruction Set Computing) model architecture, given by fewer yet highly optimized instructions.

The main purpose of a DPU is to offload and accelerate infrastructure tasks from the CPU, so CPU cycles are saved for actual computational workloads. Main infrastructure tasks involve (1) networking: moving data between servers, (2) security: firewalls, encryption, access control; (3) storage: managing data flow between storage and compute; (4) virtualization: running mini operating systems or hypervisors.
DPUs are increasingly seen as an evolving third pillar of computing, alongside CPUs and GPUs. The result is a currently dynamic DPU market. Other notable vendors are Microsoft (Azure Boost DPU), Intel (Infrastructure Processing Unit), and Marvell (Octeon).

3.2 Core capabilities

The BF DPU is marketed as offering to isolate a broad range of software-defined services. Programmability is realized through the NVIDIA DOCA (Datacenter-infrastructure On-a-Chip Architecture) SDK. DOCA provides APIs and libraries for building custom management services and applications directly on the DPU.

While these services are manifold, a focus on some HPC-relevant applications is given here.

- **Networking:** In HPC clusters, Remote Direct Memory Access (RDMA) is critical for low-latency, high-throughput communication, especially for MPI workloads. DPUs can offload RDMA operations, including packet parsing and flow control.
- **Security:** In the presence of sensitive data, DPUs can perform encryption and authentication tasks in order to secure data-in-transit between nodes without CPU involvement.
- **Storage:** When HPC applications require direct access to large datasets, NVIDIA's GDS (GPUDirect Storage) aims at reducing latency and CPU overhead. This technology enables direct data transfers between GPUs and storage (e.g., via NVMe over Fabric). Applicability, however, may depend on a host system's PCIe layout.
- **Provisioning:** Bare-metal provisioning is the OS deployment onto physical servers without virtualization layers and is an overhead-reducing alternative that can be carried out by DPUs in an out-of-band manner (without CPU involvement).

3.3 BlueField-2 versus BlueField-3 capabilities

For most metrics, the capabilities of the BF-2 [2] are effectively doubled by the BF-3 [3]. However, benchmark tests showed that this does not always translate to a 100% improvement [4]. The following table summarizes some important metrics.

3.4 Summary

In (more *traditional*) academic HPC environments, where clusters run a stable OS image, workloads are batch-scheduled, and performance is tightly tuned, the benefits of a DPU may be more narrow than for cloud providers. However, it is worthwhile researching the potential benefits given by networking and storage offload, security isolation, and (CPU-less) out-of-band management.

Metric	BlueField-2	BlueField-3
Processor	ARM Cortex A-72 (2.0 GHz)	ARM Cortex A-78 (3.0 GHz)
Core count	8	16
L1 Cache (per-core)	32KB D-cache, 48KB I-Cache	64KB D-Cache, 64KB I-Cache
L2 Cache (per-core)	1MB	512KB
L3 Cache (shared)	6 MB	16 MB
On-Chip RAM	16GB	32GB
Memory Controller Count	1	2
DRAM clock speed	1600 MT/s (DDR4)	5600 MT/s (DDR5)
Max Interconnect Speed (up to)	200 Gb/s	400 Gb/s
PCIe Interface	PCIe Gen4	PCIe Gen5
Release Date	2020/2021	2022/2023

Table 1: BlueField-2 vs BlueField-3: Key metrics in comparison

References

[1] Tyler Clifford. *Nvidia completes homerun deal after closing $7 billion acquisition of Mellanox*. https://www.cnbc.com/2020/04/27/nvidia-ceo-calls-mellanox-acquisition-a-homerun-deal.html. Accessed: 2025-08-20. 2020.

[2] NVIDIA Corporation. *NVIDIA BlueField-2 Ethernet DPU User Guide*. https://docs.nvidia.com/networking/display/bluefield2dpuenug/introduction/. Accessed: 2025-08-20. 2025.

[3] NVIDIA Corporation. *NVIDIA BlueField-3 DPU Controller User Manual*. https://docs.nvidia.com/networking/display/bf3dpu/introduction/. Accessed: 2025-08-20. 2025.

[4] Benjamin Michalowicz et al. "Battle of the BlueFields: An In-Depth Comparison of the BlueField-2 and BlueField-3 SmartNICs". In: *2023 IEEE Symposium on High-Performance Interconnects (HOTI)*. 2023, pp. 41–48. DOI: 10.1109/HOTI59126.2023.00020.

4 Tightly coupled heterogeneous systems

4.1 Introduction

Constant scientific research and development require an increasing need for computational capacity, especially in the field of high-performance computing (HPC). Today, heterogeneous platforms have become dominant in large-scale computing. Graphical processing units (GPUs) in particular have grown in popularity as accelerators, introducing the concept of general-purpose computing on GPUs (GPGPU) and turning them conceptually into more general-purpose parallel processing units. Scientific computing, artificial intelligence and large-scale data processing, as fields of research and as applied technologies, has become the driving force behind the development of new, dedicated hardware, which requires high performance,

Figure 1: NVIDIA GH200 Grace Hopper Superchip, taken from [2].

memory bandwidth, and reduced latency. To address data-transfer bottlenecks, new solutions have been developed to improve both intra- and inter-node communication, evolving the CPU-GPU coupling, from loosely coupled to tightly coupled systems [6]. Tightly coupled heterogeneous systems aim at improving data-movement bottlenecks and overall system efficiency. NVIDIA's GH200 Grace Hopper Superchip [3] can be treated as one example of such a design. In following section, we will discuss the important aspects of its architecture to illustrate the current trend of tighter integration of CPU and GPU hardware in the HPC context.

4.2 NVIDIA GH200

The NVIDIA GH200 Grace Hopper Superchip (see Fig. 1) enhances CPU-GPU interaction with a high-bandwidth and memory-coherent interconnect that reduces communication overhead. This capability is enabled by the NVLink-C2C interface, which provides up to 900 GB/s of bidirectional bandwidth between the Grace CPU and the Hopper GPU.

Memory technologies play an important role in the performance of such systems, not only through bandwidth, but also through capacity and lower latency. The Grace CPU's LPDDR5X memory offers up to 500 GB/s of bandwidth with up to 480 GB of capacity, making it well suited for large, memory-intensive CPU workloads. In contrast, the Hopper GPU's HBM3 memory provides up to 3 TB/s of bandwidth and 96 GB of capacity, supplying the throughput required by compute-intensive GPU kernels.

One of the key features of the GH200 architecture is Address Translation Services (ATS), which enables the CPU and GPU to share a unified virtual address space and a single system page table. This enables threads on either processor to access any system allocated memory directly regardless of whether it resides in the CPU's LPDDR5X or the GPU's HBM3. From the operating system's perspective, the coherent NVLink-C2C interconnect exposes these memory pools as distinct non-uniform memory access (NUMA) domains. As a result, data can be accessed without explicit host-device copies, reducing memory-operation latency and enabling flexible placement of data structures across the heterogeneous memory hierarchy.

The characteristics of this architecture are also very interesting in multi-chip and multi-node configurations. For example, a Quad GH200 node, like those used in the exascale JUPITER system at Forschungszentrum Jülich, interconnects four GH200 Superchips using NVLink technology. In this configuration, the bandwidth for GPU-to-GPU communication is up to 150 GB/s in each direction for each pair of GPUs, while the bandwidth for CPU-to-CPU communication is up to 100 GB/s. Each node has four InfiniBand NDR200 (ConnectX7) network adapters, which allow for a bidirectional bandwidth of 128 GB/s for inter-node communication. Thanks to the memory-coherent interconnect and unified memory model, GPU threads can access all memory resources – LPDDR5X on the Grace CPUs and HBM3 on the Hopper GPUs. This in turn enables unified way to manage data placement but also requires careful consideration of programming models used and data pathways in order to best utilize available computational resources.

4.3 Programming models and data paths

As processing units have evolved to meet the increasing requirements of HPC workloads, memory hierarchies have become more complex. This, in turn, requires programmers to pay closer attention and develop a deeper understanding of underlying architecture in order to effectively utilize available computational resources. This is particularly important in heterogeneous systems, where additional care must be taken to manage memory correctly. Traditional, loosely coupled systems have separate memory pools for CPUs and GPUs, and developers must explicitly handle data transfers between the host and the device. In contrast, for the tightly coupled systems, the trend is to enable unified and coherent memory access, reducing the need for manual data movement but introducing new considerations regarding data locality. Selecting appropriate data paths become essential, especially in systems like those described in the previous section, to leverage the highest bandwidth and lowest latency of the interconnects [5]. Significant effort has therefore been directed toward optimizing data movement through improved interconnects (e.g. NVLink-C2C, PCIe 5.0), unified memory mechanisms (e.g. CUDA Unified Memory, HMM), memory allocation APIs and modern programming models that expose these capabilities effectively (e.g. OpenMP target offloading, Kokkos). Tightly coupled heterogeneous systems introduce several important shifts in how software accesses memory and manages data movement. Compared to loosely coupled architectures, they simplified the programming model while introducing new opportunities for achieving higher performance.

4.4 Summary

The new architectures like NVIDIA's GH200 is a definite step in direction of creating tightly coupled heterogeneous systems. This seems to be a continued trend with the demand of systems with a large memory pool for applications like climate and weather modeling, computational fluid dynamics, molecular dynamics, machine learning and generative AI. Aforementioned technologies are pushed further by NVIDIA's new architecture found in GB300 systems where now two Grace CPUs are coupled with one Blackwell GPU using C2C interconnect [4]. However, with a new trend like this, it is equally important to develop the necessary under-

standing of these new architectures and for developers to adapt their codes and algorithms – particularly their memory-management strategies and data-movement patterns – to fully exploit the computational capabilities of such systems. In order to illustrate the shift in trends regarding CPU-GPU coupling, the focus was placed on NVIDIA's hardware, but other manufactures also offer products aiming at creating tightly coupled systems. AMD, for instance, offers the INSTINCT MI300A APU, which combines Zen4 CPU cores with CDNA3 GPU units, integrated into single chip with 128 GB HBM3 memory [1]. In this design both CPU and GPU use the same physical memory pool, in contrast to GH200 architecture, where the memory is unified only virtually. Concepts of tightly coupled systems are also apparent not only in HPC environments, but also desktop and mobile computing platforms such as Apple's M-series chips or the Qualcomm's Snapdragon X-series.

References

[1] AMD Corporation. *AMD INSTINCT MI300A APU data sheet*. `https://www.amd.com/content/dam/amd/en/documents/instinct-tech-docs/data-sheets/amd-instinct-mi300a-data-sheet.pdf`. Accessed: 2025-11-18. 2025.

[2] NVIDIA Corporation. *NVIDIA Grace CPU Superchip Architecture In Depth – Technical Blog article*. `https://resources.nvidia.com/en-us-grace-cpu/grace-cpu-1`. Accessed: 2025-12-01. 2025.

[3] NVIDIA Corporation. *NVIDIA Grace Hopper Superchip Architecture Whitepaper*. `https://resources.nvidia.com/en-us-grace-cpu/nvidia-grace-hopper`. Accessed: 2025-11-18. 2025.

[4] NVIDIA Corporation. *NVIDIA Grace Hopper Superchip Architecture Whitepaper*. `https://resources.nvidia.com/en-us-blackwell-architecture`. Accessed: 2025-11-18. 2025.

[5] Luigi Fusco et al. "Understanding data movement in tightly coupled heterogeneous systems: A case study with the Grace Hopper superchip". In: *arXiv preprint arXiv:2408.11556* (2024).

[6] Prabhu Vellaisamy et al. "Characterizing and Optimizing LLM Inference Workloads on CPU-GPU Coupled Architectures". In: *arXiv preprint arXiv:2504.11750* (2025).

5 Photonic Accelerators

5.1 Rationale

A Photonic Accelerator is a specialized hardware accelerator designed specifically for matrix-vector multiplications (MVMs), which are at the heart of artificial intelligence workloads and machine learning. They utilize natural properties of light such as diffraction and interferometry for the computations and are thus also called natural processing units. The accelerator can offer ultrafast computations (at the speed of light) at a very high degree of parallelism. It also is much more energy efficient as similar electronic technologies. Bisides machine learning

Figure 2: Picture of the Q.ANT photonic acellerator device, taken from [2]

applications, preliminary work has shown that photonic acellerators can speed-up lattice QCD calculations [1]. A first commercial solution is show in Figure 2.

5.2 Technology Description

The input vector is converted into light signals, where each element becomes represented by a different amplitude or phase of the light. The matrix is represented by an optical device such as an optical interferometer meshes. These structures perform weighting and interference, effectively implementing each matrix element. As light passes through the photonic mesh or device, it interferes and is modulated in a way that encodes the result of the multiplication. Because light propagates in parallel, the whole MVM happens at once – in a single shot – at light speed. At the output, photodetectors (like photodiodes) convert the resulting light signals (intensity or phase) back into electrical signals. These represent the result of the multiplication. Its advantages lie in speed, parallelism and energy efficiency since operations happen at the speed of light – nanoseconds or less, multiple operations can occur simultaneously using different wavelengths (Wavelength-Division Multiplexing, or WDM) or spatial paths and no transistor switching is needed, which means less heat and power consumption.

5.3 Drawbacks

Since the computations are analog, its precision is low, current devices implement 16-bit floating point precision [2]. The technology is still experimental, there are not many commercial accelerators on the market. The first commercial photonic processor is offered by [2]

References

[1] Felipe Attanasio et al. "Speeding up fermionic lattice calculations with photonic accelerated inverters". In: *Computer Physics Communications* 317 (2025), p. 109825.

ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2025.109825. URL: https://www.sciencedirect.com/science/article/pii/S0010465525003273.

[2] "Q.ANT NPS – The Photonic AI Accelerator". In: (2025). URL: https://qant.com/wp-content/uploads/2025/06/2506-QANT-Photonic-AI-Accelerator.pdf.

Software

6 *uv* - A Swiss army knife for Python software

6.1 Rationale

The Python programming language is enjoying great popularity in Scientific Computing due to its ease of use and its vast ecosystem of available software packages. However, due to the explosive growth of the number of available Python packages the complex dependency trees are becoming an increasing concern for maintainability and reproducibility of Python software. This problem is exacerbated by the fact that unlike some other programming languages (Go[1], Rust[12]), many Python distributions are only shipping rudimentary tools for package management. Modern distributions usually include *pip* and *venv*, which enables basic functionalities of package installation and the creation of "virtual environments" for isolating Python package installations. However, these tools lack many higher-level functions like project management or automatic management of virtual environments. This has lead to the development of various external tools to handle packaging and project management like *pipx*[10], *virtualenv*[2], *pyenv*[13], *hatch*[8], *flit*[7], *pdm*[9], *poetry*[3] or *Conda*[5]. The features offered by these tools vary wildly as does their adherence to Python packaging standards. Another common issue of these tools is their rather slow performance that stems from the fact that they are usually written in Python, which as an interpreted language is often much less efficient than (JIT-)compiled languages. `uv`[6] is a novel package and project management tool for Python that aims to solve these problems by consolidating most functionalities of the aforementioned tools into one while offering typically much greater performance.

6.2 Technical description

Unlike many other Python tools *uv* is written in the Rust programming language, a novel systems programming language with a focus on reliable, performant software. This choice makes *uv* much faster compared to other tooling. It is designed to deliver all functionalities for a typical Python development workflow in one comprehensive tool. Like *pipx* it can download and install Python based tools in automatically managed virtual environments. Similar to *virtualenv* and the *venv* module built into modern Python versions, *uv* can create virtual environments and like *pyenv* and *Conda* it can download and manage different versions of the Python interpreter. *uv* can also publish packages to the Python Package Index making specialized tools like *flit* or *twine* largely redundant. When it comes to project management features (e.g., creating projects, adding or removing dependencies, building and running a project), the features offered by *uv* are very similar to *poetry* and *pdm*. Looking for comparisons beyond the Python ecosystem, *uv* seems to be inspired by *Cargo*[11], the default package manager of the Rust programming language that handles dependency management, builds, testing, and publishing. Much like *Cargo*, *uv* seeks to be a comprehensive and easy-to-use

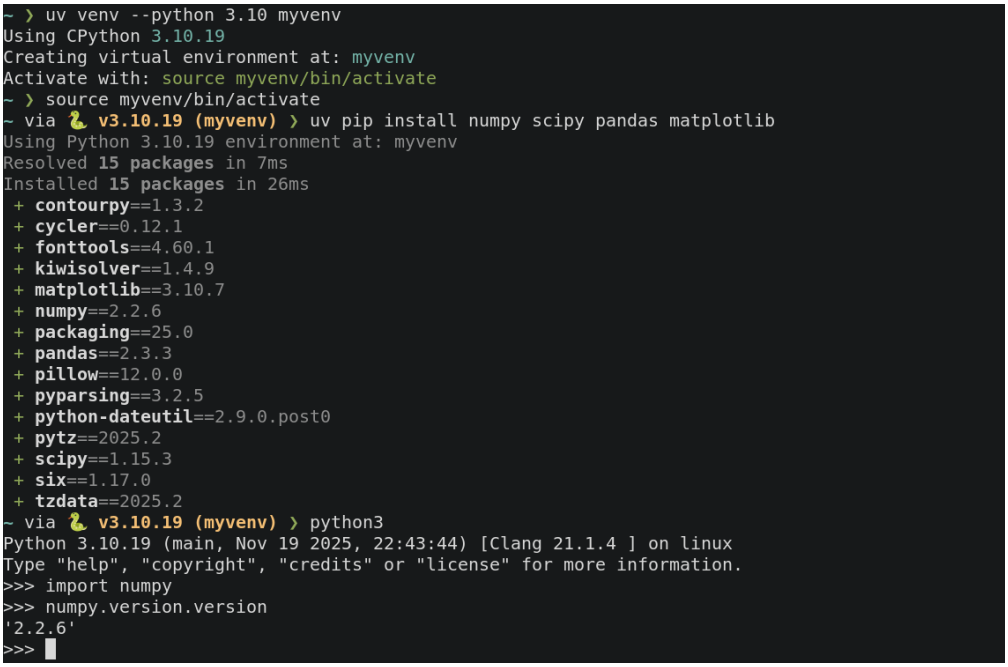

Figure 3: Screenshot of a typical *uv* workflow to create a virtual environment using Python 3.10 and install some common scientific libraries. In this example, the package cache was warm and as such no downloads of packages were necessary. As can be seen by the timings reported by *uv*, package installation only takes a small fraction of a second in this case.

tool to handle all common aspects of Python development.

Since *uv* is not written in Python, it is independent of any existing Python installation, thus avoiding the bootstrapping problem. As is common in Rust, *uv* statically links all of its own dependencies and is thus distributed as a single file. By using the *musl* C standard library[4], which unlike the *glibc* C standard library also supports static linking, it is possible to create a portable executable without any dependencies which trivializes the installation process.

uv is developed by an open-source community under the stewardship of Astral Software Inc. and licensed under the Apache 2.0 and the MIT license.

6.3 Drawbacks

The main drawback of uv is the fact that the project is still comparatively young with the first release in February 2024. It has since seen a rapid development which also implies a very fast release cycle with often multiple releases per week. The authors of *uv* still do not consider the project stable enough to release the software as *1.0* thus leaving open the opportunity for breaking backwards compatibility when necessitated by design.

References

[1] Russ Cox et al. "The Go programming language and environment". In: *Commun. ACM* 65.5 (Apr. 2022), pp. 70–78. DOI: 10.1145/3488716.

[2] The virtualenv developers. *virtualenv: Virtual Python Environment builder*. https://github.com/pypa/virtualenv. Accessed: 2025-11-24. 2025.

[3] Sébastien Eustace et al. *Poetry - Python packaging and dependency management made easy*. https://python-poetry.org/. Accessed: 2025-11-24. 2025.

[4] Rich Felker et al. *musl libc*. https://musl.libc.org/. Accessed: 2025-11-24. 2025.

[5] Anaconda Inc. et al. *Conda: A system-level, binary package and environment manager running on all major operating systems and platforms*. https://github.com/conda/conda. Accessed: 2025-11-24. 2025.

[6] Astral Software Inc. et al. *uv - An extremely fast Python package and project manager, written in Rust*. https://github.com/astral-sh/uv. Accessed: 2025-11-24. 2025.

[7] Thomas Kluyver et al. *flit: Simplified packaging of Python modules*. https://github.com/pypa/flit. Accessed: 2025-11-24. 2025.

[8] Ofek Lev et al. *hatch: Modern, extensible Python project management*. https://github.com/pypa/hatch. Accessed: 2025-11-24. 2025.

[9] Frost Ming et al. *pdm: A modern Python package and dependency manager supporting the latest PEP standards*. https://github.com/pdm-project/pdm. Accessed: 2025-11-24. 2025.

[10] Chad Smith et al. *pipx: Install and Run Python Applications in Isolated Environments*. https://github.com/pypa/pipx. Accessed: 2025-11-24. 2025.

[11] The Rust Team. *cargo: The Rust package manager*. https://github.com/rust-lang/cargo. Accessed: 2025-11-24. 2025.

[12] The Rust Team. *Rust Programming Language*. https://rust-lang.org/. Accessed: 2025-11-24. 2025.

[13] Yuu Yamashita, Sam Stephenson, et al. *pyenv: Simple Python version management*. https://github.com/pyenv/pyenv. Accessed: 2025-11-24. 2025.

7 *micro*

7.1 Rationale

A common challenge for new HPC users is the use of terminal text editors such as *emacs*, *nano*, or *vim*. These editors often defy intuitions users might have from using graphical text editors. For example, they use very different key combinations for common operations like saving, searching, or copying and pasting. Another example are the different editor modes of *emacs* and *vim* that inexperienced users often struggle with. Although there are good reasons for the design choices made by traditional commandline text editors, providing an editor that

more closely emulates the behavior of graphical text editors would significantly lower the bar of entry for new HPC users.

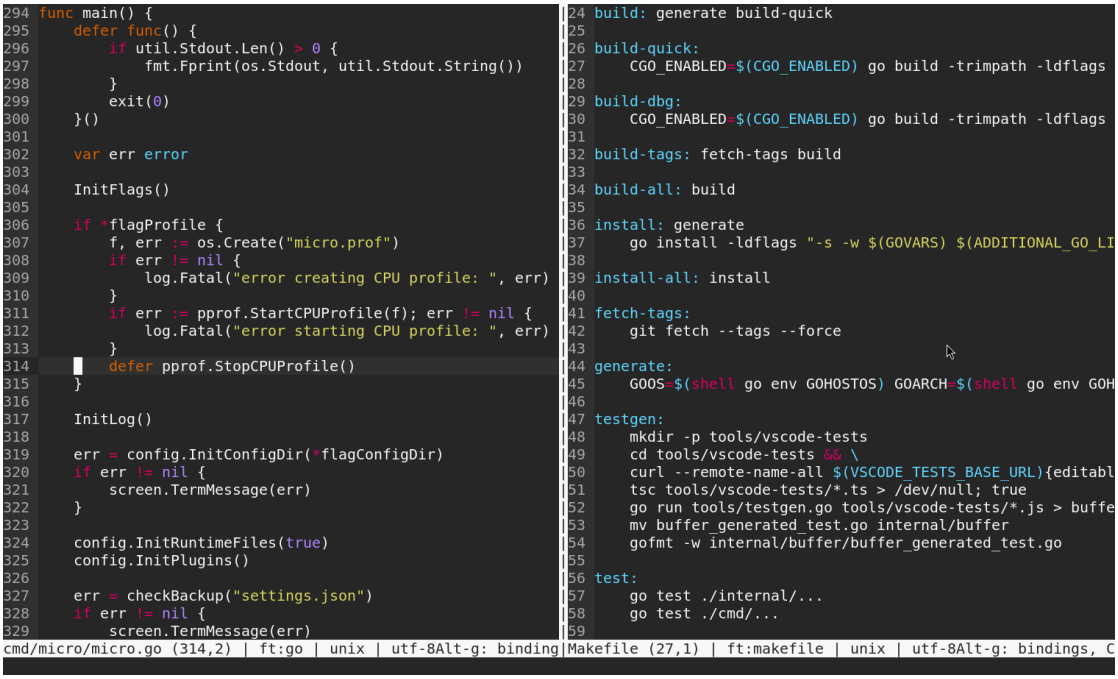

Figure 4: Screenshot from a *micro* session with two open files in a split layout. Micro supports syntax highlighting for many common programming, configuration and markup languages (shown here: Go code and a Makefile). Unlike many other commandline applications, micro also has mouse support, e.g. for selection and scrolling.

7.2 Technology description

micro [3] is a modern commandline text editor written in the *Go* programming language [1] that prides itself on being simple and intuitive. As is common for *Go* applications, it is distributed as a single statically-linked binary, which trivializes installation. It uses common keybindings for operations like saving, searching, copying, pasting and cutting, which makes using it feel like a "normal" text editor to users without prior experience with commandline text editors. Besides the basic editor functionality, *micro* also has advanced features like syntax highlighting with error reporting, multi-line cursors, simple autocompletion, mouse support, and integration with the system clipboard. In addition, it can perform terminal multiplexing operations, e.g., to run a shell next to the editor. Keybindings can be freely configured and additional functionality can be added via a plugin system using the *Lua* programming language [2].

micro is developed by an open-source community under the MIT license since 2016.

7.3 Drawbacks

Due to its relativly young age compared to the likes of *emacs* or *vim*, the plugin ecosystem of *micro* is much less extensive. Also, depending on the desktop environment, the clipboard settings might require manual intervention for the integration with the system clipboard to work.

References

[1] Russ Cox et al. "The Go programming language and environment". In: *Commun. ACM* 65.5 (Apr. 2022), pp. 70–78. DOI: 10.1145/3488716.

[2] Roberto Ierusalimschy, Luiz Henrique de Figueiredo, and Waldemar Celes Filho. "Lua - An Extensible Extension Language". In: *Software: Practice and Experience* 26.6 (1996), pp. 635–652. DOI: https://doi.org/10.1002/(SICI)1097-024X(199606)26:6<635::AID-SPE26>3.0.CO;2-P.

[3] Zachary Yedidia et al. *micro - a modern and intuitive terminal-based text editor*. https://micro-editor.github.io/. Accessed: 2025-11-24. 2025.