OS
Osservatorio sulle tendenze
e le applicazioni del Supercalcolo

ICSC
Centro Nazionale di Ricerca in HPC,
Big Data and Quantum Computing

The Big Ideas for HPC Education

From Existing Needs in High-Performance Computing Training to Recommendations for Instructional Design

Authors:
Eleonora Barelli
Michael Lodi

iFAB
INTERNATIONAL FOUNDATION
BIG DATA AND ARTIFICIAL INTELLIGENCE
FOR HUMAN DEVELOPMENT

Executive Summary

This white paper presents a comprehensive investigation into the current state and future directions of High-Performance Computing (HPC) education, addressing the critical gap between industry demand for HPC-skilled professionals and the inadequate educational offerings. Through expert consultation, literature review, overviews on the existing programs at Italian and European level, and survey methodology, this study delivers three key contributions to advance HPC education globally.

The research reveals persistent and widespread skill gaps across all stakeholder groups in the HPC ecosystem. Foundational deficiencies include the absence of core parallel programming concepts such as concurrency, parallel programming patterns, performance optimization skills, and understanding of HPC system architectures in most undergraduate curricula. Technical gaps encompass insufficient training in software optimization tools, data-intensive computing workflows, and emerging technologies like heterogeneous computing and containerization. The analysis identifies distinct training needs where Computer Science students lack integration of HPC fundamentals in core curricula, systematic optimization training, and exposure to real-world data workflows, while domain scientists and researchers suffer from low computational literacy, overreliance on domain-specific tools without understanding underlying principles, and late exposure to computational problem solving. Academic institutions face curricular fragmentation, limited educator expertise, underemphasis on reproducibility practices, and inadequate access to HPC infrastructure for hands-on learning. Industry professionals encounter barriers including absence of HPC foundations in formal education, misaligned training formats, lack of standardized certification, and insufficient knowledge transfer mechanisms within organizations. Cross-cutting gaps affect all profiles through underrepresentation of emerging technologies like AI-HPC integration and quantum computing, inadequate data management and governance training, limited focus on sustainability and societal impact, poor interdisciplinary communication skills, and pedagogical methods that fail to engage diverse learners effectively.

Drawing from educational frameworks in science and computer science, this study proposes ten fundamental "Big Ideas" of HPC that should guide curriculum design and instructional approaches. These include understanding that computation has physical limits and we can push them, recognizing that parallelism is a key to speed and scale, understanding that performance comes from matching problem, algorithm,

and architecture, knowing that decomposition is fundamental, realizing that communication can be more expensive than computation, recognizing that HPC requires collaboration between software and hardware, understanding that scientific discovery and engineering innovation depend on HPC, knowing that not all problems scale and some never will, appreciating that reproducibility and precision matter at scale, and understanding that HPC skills are transferable and evolving. These Big Ideas provide a stable conceptual framework that transcends rapidly changing technologies, enabling educators to design coherent curricula and helping learners understand HPC as an integrated discipline rather than a collection of disconnected tools.

In the final section, the study presents comprehensive recommendations for transforming HPC education and training programs. Universities should embed HPC fundamentals across relevant degree programs rather than treating it as a specialized elective, ensuring early exposure to parallel thinking and computational literacy while adopting the "Big Ideas" approach to create structured, coherent curricula that emphasize enduring principles over transient technologies. Building capacity through HPC education communities of practice, faculty training workshops, and shared teaching resources addresses the critical shortage of qualified instructors, while ensuring access to hands-on HPC environments through cloud platforms, educational clusters, or partnerships with HPC centers provides essential experiential learning for understanding scalability, performance optimization, and system management. The recommendations emphasize tailoring training to different learner profiles while encouraging cross-disciplinary collaboration, recognizing that computer scientists need domain exposure while domain scientists need computational foundations. Strengthening connections through internships, joint projects, and industry expert involvement ensures curricula remain relevant to evolving workplace needs and emerging technologies. The approach integrates data management, AI governance, sustainability considerations, and emerging technology trends such as quantum computing and digital twins to prepare professionals for the converging computational landscape, while supporting diversity initiatives, providing multiple entry points for learners with varying backgrounds, and ensuring equitable access to training resources and opportunities.

This research establishes a foundation for systematic transformation of HPC education, moving from ad-hoc, fragmented approaches toward coherent, principle-based training that can scale to meet growing workforce demands while adapting to technological evolution. The framework provides educators, policymakers, and industry

leaders with actionable guidance for developing effective HPC education programs that bridge the current skills gap and prepare professionals for the computational challenges of the future.

Authors

Eleonora Barelli, Head of Education and Foresight, IFAB Foundation

Michael Lodi, Junior Assistant Professor at the Department of Computer Science and Engineering, University of Bologna

Funding Statement

Eleonora Barelli's work has been supported by Spoke 0, "Supercomputing Cloud Infrastructure", of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC), funded under MUR Missione 4 Componente 2 Investimento 1.4: Strengthening research infrastructures and creating "national R&D champions" (M4C2-19) – Next Generation EU (NGEU).

Michael Lodi's work has been supported by Spoke 1, "FutureHPC & BigData", of the Italian Research Center on High-Performance Computing, Big Data and Quantum Computing (ICSC), funded under MUR Missione 4 Componente 2 Investimento 1.4: Strengthening research infrastructures and creating "national R&D champions" (M4C2-19) – Next Generation EU (NGEU).

Barelli, E., & Lodi, M. (2025). The Big Ideas for HPC Education: From Existing Needs in High-Performance Computing Training to Recommendations for Instructional Design. Doi: 10.5281/zenodo.17097602

Acknowledgements

We would like to express our sincere gratitude to all the participants who contributed to this research.

Thanks to all the experts we interviewed for more than one year to explore the foundation of HPC in the light of educational reconstruction. Thanks in particular to Marco Aldinucci (University of Torino), Moreno Marzolla (University of Bologna), and Eric Pascolo (Cineca) who shared with us their insights with extraordinary intellectual generosity.

Thanks to all questionnaire respondents, both anonymous and non-anonymous. Thanks in particular to: Ludovic Capelli (CSIRO), Mary Chessey (Barcelona Supercomputing Center), Maksym Deliyergiyev (High-Performance Computing Center of Stuttgart), Ivan Gentile (IFAB Foundation), Miriam Leeser (Northeastern University).

Special thanks are extended to the participants of PIPSQUEAK, both authors and audience members, as well as to the teams of CASTIEL and EUROCC for their invaluable support and collaboration. Thanks in particular to Aline Melinette (Barcelona Supercomputing Center) who promoted our research to the networks' members.

We acknowledge the support and guidance of the ICSC Observatory Advisory Board that allowed our research questions and methods to be constantly aligned with the research and innovation directions of the ICSC National Center on Supercomputing. Specifically, we thank Marco Aldinucci (Università di Torino), Renzo Giovanni Avesani (UnipolSai), Stefano Baroni (Scuola Internazionale Superiore di Studi Avanzati), Sanzio Bassini (Cineca), Andrea Cavalli (Università di Bologna & Italian Institute of Technology), Carlo Cavazzoni (Leonardo), Paolo Cremonesi (Politecnico di Milano), Alessia D'Orazio (Fondazione ICSC),Giovanni Foresti (Intesa Sanpaolo), Serena Fumagalli (Intesa Sanpaolo), Silvio Gualdi (Centro Euro-Mediterraneo sui Cambiamenti Climatici), Giovanni Ginocchini (Fondazione Innovazione Urbana), Michael Lodi (Università di Bologna), Riccardo Malpica Grassi (Sapienza Università di Roma), Vito Morreale (Engineering), Marco Pistore (Fondazione Bruno Kessler), Antonino Rotolo (Università di Bologna), Davide Salomoni (Fondazione ICSC), Fabio Sciarrino (Sapienza Università di Roma), Luigi Terracciano (Humanitas), Matteo Zanaroli (Fondazione ICSC).

The review in the Framework section is heavily based on the work of M. Lodi with Marco Sbaraglia and Simone Martini, published in Sbaraglia's PhD thesis (Sbaraglia, 2023, chapter 4).

Table of Contents

Executive Summary 2
Acknowledgements 5
The state of the art of HPC education 8
Core competencies and curricular guidelines 8
HPC Education in K–12 Schools 11
HPC Education in Higher Education 13
HPC Education and Training for Professionals 16
Existing initiatives in Italy and Europe on HPC education 20
The ICSC Observatory initiatives on HPC education: Pipsqueak 25
Framework: the Big Ideas 28
Big Ideas in Science Education 28
The importance of Big Ideas 29
How to distil Big Ideas 31
Big Ideas of CS 32
The Research 34
Research Questions 34
Methodology 34
Literature Review 34
Expert Consultation 34
Iterative Drafting of Educational Needs and Big Ideas 35
Questionnaire-Based Survey 35
Questionnaire Design 35
Questionnaire implementation 37
Survey dissemination 37
Results 41
Emerging Needs and Training Gaps in HPC Education 41

Training Gaps for Computer Science (CS) Students and Graduates 43
Training Gaps for Domain Students and Researchers (Non-CS) 44
Training Gaps in Academia (Educators, Students, Researchers) 45
Training Gaps for Industry Professionals and Practitioners 46
Cross-Cutting Gaps (Affecting All Profiles) 48
The Big Ideas of HPC 51
From Results to Recommendations for Instructional Design 62
Conclusion 68
References 69
Annex – Questionnaires' analysis and results 73

The state of the art of HPC education

High-Performance Computing (HPC)[1] refers to systems that are orders of magnitude more powerful than standard systems such as desktops or mobile devices (Raj et al., 2020). These platforms can solve problems in minutes or days that would take months or years on conventional computers, enabling breakthroughs in climate modelling, genomics, physics, engineering, and artificial intelligence. While HPC can in principle include paradigms such as quantum computing, in this report we focus mainly on HPC realized primarily through parallel and distributed computing (PDC), which underpins nearly all current large-scale systems (Raj et al., 2020).

Core competencies and curricular guidelines

A crucial step in shaping HPC education is defining the core concepts and competencies that students and professionals should acquire. Two major curriculum initiatives serve as global benchmarks.

The NSF/IEEE-TCPP Curriculum Initiative

The NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing is a long-standing global effort to embed parallel and distributed thinking into undergraduate computer science and engineering education. Its first version (2011–2012) laid out a systematic framework for introducing concurrency and parallelism across the undergraduate curriculum (Prasad et al., 2011).

In 2020, the initiative released a major revision (Prasad et al., 2020), reflecting the rapid evolution of computing hardware and applications. The updated guidelines emphasize that all computing graduates should acquire a baseline competence in PDC concepts, as these permeate both explicit system design and everyday applications that rely on distributed services. The revised curriculum organizes content into four complementary layers:

- Pervasive concepts cutting across the curriculum, including concurrency, asynchrony, locality, and performance. These are introduced early—often

[1] Although they are sometimes used as synonyms, we will tend to talk about HPC more generally, of which supercomputing is a subset: https://www.ichec.ie/news/press-corner/about-high-performance-computing-hpc .

even in CS1 or CS2—and reinforced throughout systems, architecture, and algorithms courses.

- Core areas of Architecture, Programming, and Algorithms, updated to reflect modern platforms. For example, architectural coverage now extends beyond memory hierarchies to include *underlying mechanisms* such as caching, atomicity, and coherence; programming guidelines include not only shared- and distributed-memory models but also hybrid approaches and accelerator programming; algorithms topics emphasize scalability, data-intensive computation, and trade-offs between computation and communication.
- Cross-cutting updates that bring in emerging priorities such as Big Data, Energy efficiency, and Distributed computing. For instance, data movement and locality are framed as central bottlenecks in both algorithm and system design; energy is treated as a first-order performance metric at both the device and system level; and distributed computing concepts are introduced earlier, reflecting the ubiquity of cloud and collaborative applications.
- Emerging topics, reserved primarily for advanced or elective courses, which highlight evolving areas such as GPU/TPU architectures, cloud and edge computing, and large-scale data analytics.

A distinctive feature of the TCPP initiative is its integration-first philosophy: instead of relegating parallelism to a standalone elective, it advocates *sprinkling PDC topics throughout the curriculum*. This includes teaching concurrency in introductory programming, discussing cache coherence in systems courses, and exploring parallelism in algorithms alongside their sequential counterparts.

The ACM 2023 Recommendations on PDC

The ACM Computing Curricula 2023 (CC2023) report built on the TCPP initiative, making Parallel and Distributed Computing a cross-cutting knowledge area across computing programs (ACM, 2023). Its framework identifies competencies in several domains:

- Fundamentals: concurrency, decomposition of tasks, parallel execution, and scalability.
- Programming models: threads, OpenMP, MPI, GPU programming, and distributed frameworks such as MapReduce or Spark.

- Systems and architecture: multicore/manycore processors, interconnection networks, cluster and cloud systems.
- Design and analysis: parallel algorithms, performance models, and trade-offs between computation and communication.
- Software engineering: debugging, profiling, performance analysis, and reproducibility in HPC.
- Applications and ethics: HPC for AI, data science, and scientific discovery, with attention to sustainability and responsible computing.

Compared to the TCPP guidelines, CC2023 places stronger emphasis on heterogeneous and accelerator-based computing, as well as links to emerging areas like AI and data analytics. It also explicitly encourages competency-based, modular curricula and micro-credentials.

HPC Education in K–12 Schools

Integrating HPC concepts at the K–12 level is in its infancy. Traditionally, supercomputing and parallel programming were considered too advanced for primary/secondary curricula. However, with the ubiquity of multi-core devices and the rising importance of CS in our society, **educators are experimenting with ways to introduce foundational HPC and parallel computing ideas to younger students**. A key question has been whether secondary school students can grasp parallel programming fundamentals and find them engaging. Encouragingly, initial research indicates that they can: a study in Croatian high schools, for example, found that students were able to understand concepts like race conditions and evaluate parallel programs, rating the content as equally interesting and useful (if slightly more challenging) compared to other informatics topics (Brođanac et al., 2022). This suggests that, **when taught appropriately, even complex HPC concepts can be demystified for teens**.

K–12 HPC education often takes the form of extracurricular workshops, project-based learning, or integrated STEM activities rather than formal courses. One successful approach is "learning by doing", giving students hands-on experience with simplified HPC setups. For instance, the Universidad de Buenos Aires reported running science workshops where secondary students built and experimented with a mini HPC cluster, fostering interest through active problem-solving and collaboration (Slezak et al., 2009). In the United States, some high school programs and summer camps have leveraged small-scale clusters to teach parallel computing basics. The availability of affordable, portable cluster hardware has been a game-changer. Notably, educators have deployed Raspberry Pi–based mini-clusters as accessible "supercomputers" for the classroom. The University of Nebraska's Holland Computing Center developed "Legion," a Raspberry Pi cluster, specifically for K–12 outreach; this portable mini-supercomputer enabled interactive demos and training sessions that introduced students to HPC principles in a tangible way (Bohn & Brown, 2020). Likewise, the longstanding LittleFe project provided a 6-node portable cluster (weighing approximately 22 kg and assembled for just a few thousand dollars) as an "educational appliance" for teaching parallel computing anywhere (Ludin et al., 2013). More recently, researchers in Austria designed the Cluster Coffer, a 16-node ARM-based mini-cluster, to lower the entry barrier for HPC education; it is inexpensive, easy to transport, and has been used in both classroom settings and public outreach events (Gschwandtner et al., 2021). These small-scale clusters allow students to experiment with running jobs

in parallel, see real-time speedups, and even participate in simplified versions of supercomputing challenges.

Another avenue for early HPC education is **incorporating parallel thinking into existing computing or science curricula without requiring actual supercomputer access**. PDC "unplugged" (Matthews, 2020) activities – which teach parallel concepts through analogies or games. For example, teachers might illustrate parallel processing by having groups of students simulate processors working on subtasks of a larger problem. Such conceptual exercises can plant the seeds of parallel thinking even when hardware is not available.

Block-based parallel programming tools have been explored to convey parallel programming abstractions, including producer-consumer, master-worker, and MapReduce (Feng et al., 2015).

Contests and events also motivate students: the annual Student Cluster Competition at major HPC conferences now occasionally features teams of high schoolers competing alongside university teams to build and optimize a small cluster system in a 48-hour challenge[2]. Such experiences not only build technical skills but also excitement and awareness about HPC careers.

Despite these promising efforts, **scaling HPC education in primary and secondary schools faces challenges**. One issue is the already crowded K–12 curriculum – finding room for HPC topics (which are not part of standard testing regimes) often relies on passionate teachers or external programs. Teacher preparation is another hurdle: most K–12 instructors have little or no background in parallel computing. Initiatives to train the teachers themselves (through professional development workshops) are therefore critical (Saule et al., 2021). There is also the matter of appropriate curriculum design: identifying which HPC concepts are age-appropriate and how to contextualize them. The Croatian study's success was partly due to careful scaffolding – connecting parallel programming to familiar concepts (network programming and problems students had seen before) and limiting the scope to fundamental principles within a short course unit. This kind of pedagogical strategy needs to be generalized and shared. Efforts like the NSF/IEEE-TCPP working group's curriculum also recognize the importance of early exposure: while their primary focus is undergraduate, the spirit

[2] https://tacc.utexas.edu/research/tacc-research/student-cluster/

of their recommendations – that parallelism is now a core competency – supports introducing key ideas even before college.

HPC Education in Higher Education

Universities have been the traditional home of HPC education, whether through specialized courses in computer science programs or training within domain science curricula. Over the past decade, there has been a concerted push to integrate parallel and distributed computing topics into the core undergraduate curriculum, reflecting a paradigm shift in computing hardware (the end of frequency scaling and the rise of multi-core and GPU-based systems). Thanks to the curricular initiatives we mentioned earlier, more and more universities globally now introduce basics of parallelism (e.g. threads, multicore programming, GPU computing or message-passing paradigms) in either introductory courses or as dedicated electives before students graduate.

Implementing HPC/PDC at the undergraduate level takes various forms. Some programs embed parallel topics into existing courses (for example, modifying a Data Structures or Operating Systems course to include multi-threading assignments), while others create new courses focused on parallel algorithms, distributed systems, or high-performance computing. A number of interdisciplinary programs have also emerged: for instance, a project at a liberal arts college described a three-semester sequence where students from diverse majors collaboratively learned parallel programming applied to different domains (Morris & Frinkle, 2014). The literature suggests that active learning and project-based approaches are especially effective in HPC education. Students benefit from "learning by doing" – working on projects that require writing and optimizing code on actual parallel hardware. A systematic review (Mwasaga & Joy, 2020b) of 211 HPC education papers (1988–2018) found that the majority of educational interventions leveraged Beowulf clusters and similar multi-node systems as pedagogical tools, underscoring the importance of hands-on access to parallel computing resources. Many universities maintain a campus HPC cluster or leverage national supercomputing allocations for classroom use, giving students the invaluable experience of running and scaling their programs on real HPC environments. Where dedicated hardware is not available, instructors have turned to cloud computing or simulators to approximate HPC conditions (for example, using cloud VMs to create a virtual cluster for a course). The use of portable mini-clusters (like LittleFe or Raspberry Pi clusters mentioned earlier) is also common in undergraduate labs or capstone projects, bringing the feel of a supercomputer into the classroom. One recent

approach, in line with "maker" trends, is to have students build a small cluster as part of a course, which not only teaches HPC architecture concepts but also instils practical systems skills (Vargas-Pérez, 2022).

To support instructors, the HPC education community has been actively sharing curricular materials and strategies. Workshops such as EduPar (Education in Parallel and Distributed Computing, co-located with IEEE IPDPS) and EduHPC (held with ACM/IEEE Supercomputing conferences) serve as forums for educators to present experience reports, new tools, and pedagogical research. Model assignments and pedagogic patterns – for instance, the "Peachy Parallel Assignments" which are peer-reviewed, adoptable parallel programming exercises – have been curated to ease adoption by faculty (Bücker et al., 2023). These community efforts address a key challenge: many computer science instructors themselves did not receive formal training in HPC or parallel programming, and thus may feel ill-prepared to teach it. By providing ready-to-use modules and instructor training workshops, initiatives like the NSF/TCPP early adopter program and the Center for Parallel Computing Education (CDER) have lowered barriers. Still, scaling up instructor expertise remains an issue; a position paper by Saule et al. argues that without a broader community "strike force" to develop and disseminate courseware, PDC adoption in curricula will continue slower than needed (Saule et al., 2021).

Beyond core undergraduate education, HPC forms a significant component of graduate and professional degrees. Many universities offer specialized master's programs or concentrations in HPC, Computational Science, or Parallel Software Engineering. These programs typically dive deeper into supercomputing architecture, parallel algorithms, performance optimization, and HPC applications in science and engineering. Today, dozens of universities worldwide have formal HPC tracks. In Europe, a particularly notable initiative is the EUMaster4HPC, that we will describe in the next section. The success of such programs points to a model for cultivating HPC expertise through immersive, multi-institutional education – something that could be replicated in other regions or globally.

Universities are adapting by updating multiple courses – not only those titled "HPC" – to reflect parallelism. For instance, a databases course might touch on distributed data processing, or an algorithms course might analyze parallel algorithm complexity. The overarching goal is to ensure graduates have the conceptual background to thrive in a world where computing is inherently parallel and distributed. HPC-specific

courses then build on that foundation for students who pursue computing-intensive careers.

University-level HPC education faces its own set of challenges. One perennial issue is keeping curricula up-to-date with rapid technological changes. HPC hardware and software ecosystems evolve quickly (consider the rise of general-purpose GPU computing in the last decade, or the current emergence of accelerator-rich and heterogeneous architectures). Educators feel the need to constantly refresh course content – adding topics like CUDA or SYCL programming for GPUs, using modern parallel libraries, or addressing HPC for data analytics and AI workloads – which can strain faculty time and expertise. The need for accessible hardware is another challenge: not every institution has a local supercomputer for student use, and even when they do, scaling assignments to multiple dozens of students can be tricky. Cloud-based solutions and shared educational allocations on national systems are mitigating this, granting students accounts on HPC systems for coursework. On the positive side, there is increasing support for open educational resources. Materials like the HPC Carpentry lessons (described in the next section) and repositories of parallel programming assignments help reduce development effort for instructors. Another trend is the use of interdisciplinary projects to teach HPC – students from computer science partnered with students from physics or bioinformatics to optimize a real simulation code, for example. This not only mirrors real-world HPC team scenarios but also attracts students from application domains to learn HPC skills relevant to their fields. The expansion of data science programs has also had an effect: some HPC education is now delivered under the banner of "big data" or "AI infrastructure" courses, reflecting an intersection of HPC, high-performance data analytics (HPDA), and machine learning.

HPC Education and Training for Professionals

HPC education does not end with formal schooling – in fact, given the pace of innovation, continuous professional development is essential for researchers, engineers, and IT professionals to effectively utilize advanced computing resources. This category of HPC education spans on-the-job training, short courses, certificate programs, and workshops aimed at current practitioners (from academia, industry, or government) who seek to upskill or reskill in high-performance computing techniques. There is also a critical workforce development aspect: **governments and industries investing in HPC infrastructure want to ensure a pipeline of skilled personnel who can leverage those systems for scientific and economic benefit**.

Many national HPC centers and laboratories host regular training events. For example, the U.S. Department of Energy's labs organize tutorials and summer schools open to the community – a flagship offering is the Argonne Training Program on Extreme-Scale Computing (ATPESC), an intensive two-week residential program that each year brings around 70 early-career researchers together to learn cutting-edge HPC tools and methods (covering everything from MPI/OpenMP programming to performance tuning on next-generation architectures)[3]. Attendees are typically required to already have some parallel programming experience, and the curriculum is designed to take them to the next level of expertise. Programs like ATPESC exemplify how professional training can fill the gaps between what formal education provides and the skills needed to exploit leadership-class machines. Similarly, in Europe, the Partnership for Advanced Computing in Europe (PRACE) has historically run seasonal schools and workshops across its member supercomputing centers, free of charge for participants, to disseminate HPC best practices. These range from introductory courses (e.g. "HPC for beginners" covering how to access and run jobs on a cluster) to specialized workshops on topics like GPU programming, code optimization, or HPC for specific scientific domains. During the 2020–22 period **Europe's network of national HPC competence centers collectively organized on the order of a hundred training events** (despite the pandemic), covering key skills for both new and advanced HPC users. This demonstrates a significant scaling-up of training opportunities targeting researchers and professionals.

[3] https://extremecomputingtraining.anl.gov/

A notable trend in professional HPC training is the adoption of interactive, learner-centric approaches akin to those used in software carpentry workshops. One prominent example is HPC Carpentry (Reid et al., 2025), a community-driven training program that adapts the principles of The Carpentries[4] (an organization known for its effective workshops in basic programming and data science) to the HPC context. HPC Carpentry workshops typically span 1–2 days and are designed for novices who have never used a supercomputer. Instructors and learners work together in live coding sessions, typing commands into the terminal to practice tasks like navigating a Linux environment, managing job schedulers, and running simple parallel jobs. The pedagogy emphasizes building "muscle memory" and demystifying errors – instructors intentionally demonstrate common mistakes and how to fix them, which helps reduce participants' intimidation with HPC systems. The content is modular, open-source, and continuously refined by a global community, ensuring that workshops stay up-to-date and effective. By providing a standardized, scalable curriculum, HPC Carpentry aims to "democratize HPC access" – enabling researchers from non-traditional HPC fields or smaller institutions to acquire the skills to utilize modern clusters (Reid et al., 2024). This is crucial as HPC becomes relevant not just to computational scientists, but also to disciplines like biology, medicine, finance, and the arts, where practitioners may not have had formal HPC training.

The rise of online education has significantly impacted HPC training. Many HPC centers now offer webinars, virtual lecture series, and recorded tutorials that professionals can access on-demand. For instance, during 2020–2021, numerous in-person HPC training events pivoted to virtual formats, which in some cases increased their reach (geographic and attendance). HPC MOOCs (Massive Open Online Courses) have also been developed. One early example was the "HPC Parallel Programming" MOOC by the University of Edinburgh (through PRACE and FutureLearn), which introduced thousands of learners to the basics of supercomputing and parallel computing. Another, "HPC for Computational Scientists" on edX, provided domain researchers with HPC skills. While MOOCs face the typical challenge of retention, they serve as valuable introductory funnels and references. Additionally, university extension programs and technical institutes are starting to offer certificate programs in HPC or parallel programming, often delivered online to working professionals. These typically involve a series of courses culminating in a certificate that signals HPC proficiency. An Italian example is the International Professional programme in "HPC and quantum computing", a 60

[4] https://carpentries.org/

ECTS program from the University of Bologna, joint effort between Departments of Physics and Computer Science and Engineering.

A recent development in the HPC professional landscape is the movement toward formalizing HPC competencies and certification. The HPC Certification Forum, an international community of HPC trainers and experts, has been working to define a comprehensive skill tree for HPC practitioners[5]. Their goal is to categorize and standardize the knowledge areas (from shell usage and job scheduling to performance engineering and math libraries) into fine-grained competencies, and to develop examinations leading to a globally acknowledged HPC certification (Kunkel et al., 2020). By 2023, this effort had progressed with beta certification exams being tested, aiming to provide employers and organizations a way to assess HPC skill levels and to guide learners on what proficiencies they should acquire. If widely adopted, such a certification could become akin to how Cisco certifications function in networking – creating a common understanding of what it means to be an "HPC certified" professional. This competency-based approach is also useful for designing curricula and training roadmaps: for example, EuroHPC's new EVITA (European Virtual Training Academy) project explicitly seeks to establish a Europe-wide framework of HPC competences and qualifications[6]. EVITA plans to deliver reference course materials and modular learning units mapped to specific skills, thereby supporting consistent training standards for students, professionals, and educators across Europe. Such frameworks ensure that a professional taking an HPC course in one country learns content that aligns with what is taught elsewhere, facilitating skill recognition and mobility.

HPC has penetrated industry sectors (manufacturing, pharma, finance, etc.), meaning companies increasingly seek to train their workforce in HPC and advanced analytics. HPC education for industry often emphasizes practical application and ROI: for instance, EuroCC Italy in 2025 offered a course titled " High-Performance Computing for Digital Transformation" specifically for corporate decision-makers and R&D staff[7]. The focus was on understanding how HPC can accelerate business innovation, with real case studies on integrating HPC, AI, and simulation into industrial processes. By tailoring content to non-academic audiences (less theory, more strategy and applied examples), such courses help demystify HPC for executives and managers,

[5] https://www.hpc-certification.org/
[6] https://www.eurohpc-ju.europa.eu/research-innovation/our-projects/evita_en
[7] https://euroccitaly.it/en/news/high-performance-computing-for-digital-transformation-a-new-training-course-by-eurocc-italy/

fostering broader adoption in small and medium enterprises. Another form of on-the-job HPC training is through vendor-led courses; manufacturers of supercomputers or software (like NVIDIA for GPU computing or MPI library developers) often provide training to their users. Additionally, mentorship programs and "twinning" between experienced HPC centers and organizations new to HPC have proven effective. The CASTIEL project in Europe, for example, facilitated mentoring and twinning between countries to share HPC expertise, effectively pairing less experienced teams with veteran centers for knowledge transfer[8]. This not only trains individuals but also builds institutional capacity.

One persistent challenge is keeping training content current with the state-of-the-art. HPC professionals need to learn new programming models, tools, and paradigms as they emerge (for example, the recent rise of containerization and cloud-HPC convergence, or the need to learn accelerators like AI chips). Training programs must regularly update their syllabi, which requires close collaboration between training providers and cutting-edge HPC projects. Another challenge is assessing impact – unlike formal education, professional training outcomes can be harder to measure. Organizations often rely on feedback surveys or track whether trainees go on to effectively use HPC resources in their work (e.g. increased software performance or successful project completions) as indicators of training success. The diversity of the audience is also a factor: professional HPC learners range from domain scientists with little CS background to IT specialists needing optimization skills. Designing training that can cater to varying prerequisites is non-trivial. Modular, competency-based content as proposed by EVITA and the HPC Certification Forum can help by allowing tailored learning pathways.

Conferences like SC and ISC now regularly feature "Best Practices in HPC Training" workshops and Birds-of-a-Feather sessions where instructors discuss what works (and what doesn't) in adult HPC education[9]. The consensus is that interactive, hands-on learning – whether through Carpentry-style workshops, hackathons, or guided problem-solving sessions – is far more effective than traditional lecture-only formats for skill uptake.

[8] https://www.hlrs.de/news/detail/eurocc-national-competence-centers-hold-first-all-hands-conference
[9] https://sighpceducation.acm.org/events/bphte25cfp/

Existing initiatives in Italy and Europe on HPC education

In recent years, Europe has launched a comprehensive, multi-tiered effort to strengthen HPC education and training as part of its broader supercomputing strategy. Recognizing that world-class HPC infrastructure must be matched with a skilled workforce to harness it, the European Union and national governments have invested in programs that span academia, industry training, and network-building among HPC centers. Here we highlight several key European initiatives – including those involving Italy – that illustrate a coordinated approach to HPC skill development. These include the EuroHPC Joint Undertaking's education projects, the network of EuroCC National Competence Centres and its coordination arm CASTIEL, the pioneering pan-European Master's program (EUMaster4HPC), and new projects aimed at unifying training materials across the continent. Together, these efforts make Europe a case study in how policy-driven support can elevate HPC education at scale.

Europe's HPC ecosystem has long invested in training and workforce development. The Partnership for Advanced Computing in Europe (PRACE) operated a network of 14 PRACE Training Centres, offering a state-of-the-art curriculum of short courses and workshops in HPC and computational science across the EU[10]. These include modules on MPI/OpenMP programming, GPU programming, code optimization, and domain-specific HPC techniques, delivered to thousands of researchers and students.

Launched in 2020 under the EU's EuroHPC Joint Undertaking, EuroCC established HPC Competence Centres in 33 countries (EU members and partners) to serve as national hubs for HPC expertise[11]. Each National Competence Centre (NCC) was tasked with assessing local HPC skills and resources, identifying gaps, and acting as a focal point for training, industry outreach, and support within its country. In parallel, the CASTIEL project (Coordination and Support Action) was funded to promote interaction and exchange among these NCCs at the European level[12]. CASTIEL emphasized areas like training, industrial engagement, and best-practice sharing – essentially ensuring that no country's centre operates in isolation. Through CASTIEL's coordination, the NCC network has implemented common standards and collaborative activities,

[10] https://prace-ri.eu/prace-association/
[11] https://www.eurohpc-ju.europa.eu/research-innovation/our-projects/eurocc_en
[12] https://www.eurohpc-ju.europa.eu/research-innovation/our-projects/castiel_en

raising the overall quality of HPC education and services across Europe[13]. By the end of the first phase (2020–22), the impact was significant: as already mentioned, despite challenges like the pandemic, the NCCs collectively organized at least 95 training courses and 60 workshops on priority HPC topics, reaching diverse communities of users. CASTIEL also built a Europe-wide HPC competence map and the EuroCC Access web portal (now HPC portal), where anyone (from academia, industry, or public sector) can find information on HPC training offers and expertise available in each country[14].

Italy has been an active participant in EuroCC and CASTIEL. The Italian National Competence Centre (EuroCC Italy) is coordinated by CINECA (Italy's largest supercomputing consortium) along with IFAB Foundation, BI-REX, Dompè, and Leonardo Company. It provides training courses free-of-charge in HPC, High-Performance Data Analytics (HPDA), and AI, targeting both academic researchers and industry users[15]. A recent example, as mentioned, is the course on HPC for Digital Transformation aimed at companies, reflecting Italy's strategy to extend HPC know-how into its industrial base. Moreover, CINECA was directly involved in CASTIEL's consortium, meaning Italian experts have contributed to the Europe-wide exchange of training methodologies. Italian institutions have also engaged in twinning projects under CASTIEL to mentor or learn from other countries in specific competency areas, fostering a two-way knowledge transfer.

Building on the initial success, EuroCC 2 and CASTIEL 2 started in 2023 as a second phase with fresh funding from the EU's Digital Europe Programme[16]. These projects continue the mission for another three years, expanding training offerings and deepening cooperation. CASTIEL 2 places particular focus on advanced training collaborations (such as jointly developed courses), certifications, and integrating emerging topics like quantum and edge computing into the competence network (though quantum education remains distinct, HPC and quantum are sometimes addressed side-by-side under EuroHPC). With a budget of around €3 million, CASTIEL 2 is set to further harmonize HPC training across Europe. It also continues to encourage

[13](https://www.hpcwire.com/off-the-wire/eurocc-and-castiel-begin-2nd-funding-phase
[14] https://hpc-portal.eu/
[15] https://euroccitaly.it/en/
[16]https://www.eurohpc-ju.europa.eu/eurocc-2-and-castiel-2-promoting-hpc-boost-digital-skills-jobs-and-industrial-competitiveness-europe-2023-02-03_en

"twinning" and mentorships – by mid-2023, task forces had formed under CASTIEL 2 to share expertise on specific domains and coordinate new content development.

Europe's effort to bolster HPC skills has also targeted formal education. The flagship academic initiative was EUMaster4HPC, the first pan-European Master's program in HPC launched in late 2022. Coordinated by the University of Luxembourg and funded by EuroHPC JU, EUMaster4HPC convened a consortium of leading universities (including Politecnico di Milano, Sorbonne Université, Universitat Politècnica de Catalunya, and others), research institutions, and industry partners[17]. A primary goal is to design a common reference curriculum for HPC that can be adopted across Europe (Bouvry et al., 2025). The program was designed with a "mobility and excellence" philosophy: students would study in two countries (earning degrees from two institutions) and follow a jointly developed curriculum that reflects the needs of future HPC systems and applications[18]. Over two years (120 ECTS credits), the curriculum covered advanced HPC architecture, parallel programming models, algorithms, mathematics for HPC, and use-case-driven electives (like HPC for life sciences or climate modeling). Crucially, EUMaster4HPC had strong practical components – each cohort attended European HPC summer schools, participated in international competitions (e.g., the ISC/SC student cluster competitions or hackathons), and undertook a thesis internship at a supercomputing center or industrial R&D lab. This blend of academic rigor and real-world experience aimed to produce "T-shaped" professionals: deep HPC technical expertise combined with broader soft skills and domain exposure. The program also provided generous scholarships and fee waivers, supported by EU funding, to attract talent from all over Europe and beyond. Italy's Politecnico di Milano was a key academic node in this program, offering an HPC Engineering track as part of EUMaster4HPC and hosting students for their first year before they moved to a partner university in year two. Polimi established double-degree agreements for this purpose and contributed its strong engineering perspective to the curriculum. The project is concluding in 2025, but is aimed to integrating the pan-European HPC curriculum into existing university offerings, continuing the cross-border collaboration under a different funding scheme or as a network of master's programs.

The success of EUMaster4HPC has already inspired offshoots. For instance, the idea of European micro-credentials in HPC is being floated, where students enrolled in any

[17] https://eumaster4hpc.uni.lu/
[18] https://eumaster4hpc.eu/

university could take a certified HPC module (perhaps online) that is recognized across institutions – a concept aligned with the EU's push for flexible learning pathways. Moreover, elements of the EUMaster4HPC curriculum are feeding into the aforementioned EVITA project (EuroHPC Virtual Academy). With EVITA[19], Europe is essentially creating a repository of high-quality HPC teaching modules, which can be assembled into courses or full programs by any university or training provider. This includes defining qualification standards so that, for example, a learner earning an "HPC Fundamentals Certificate" in one country has demonstrably the same skills as one in another country. EVITA also explicitly targets multiple audiences – from undergraduates and graduate students to industry professionals and even trainers themselves. Such alignment ensures that professional training (through NCCs) and academic education (through universities) are not siloed but complementary and mutually reinforcing.

In Italy, the National Interuniversity Consortium for Informatics (CINI) has established an HPC key technologies lab. Among its many activities, it organizes an annual HPC Summer School that brings together students, young researchers, and professionals. For example, the 3rd CINI International Summer School on High-Performance Computing (June 2025, Naples) offered an intensive program covering the entire hardware and software stack of HPC, from innovative architectures to the latest parallel programming models and tools for heterogeneous platforms[20]. Such schools not only teach state-of-the-art HPC technologies but also promote mentoring, networking, and the formation of an HPC community among academia and industry.

Italy, as part of the EuroHPC ecosystem, has seen a boost in HPC educational activities. Besides Politecnico di Milano's involvement in EUMaster4HPC, other Italian universities have strong HPC groups (for example, the International School for Advanced Studies in Trieste offers HPC and data science specialization. Italy's national HPC facility, CINECA, regularly conducts training workshops on using their Tier-0/Tier-1 systems (like the Leonardo supercomputer) – covering topics from introductory Linux and programming, to advanced code optimization and AI integration. Through EuroCC Italy, these training sessions have been made more accessible to industry and smaller research institutes that might not have participated before. Furthermore, Italian HPC experts are contributing to European training content development (for instance, CINECA staff co-authored lessons in the HPC Carpentry project and contribute to

[19] https://www.evitahpc.eu/
[20] https://www.hpcsummerschool.it/

PRACE training material development). The Italian government's investment in HPC (as evidenced by hosting a pre-exascale computer Leonardo and upcoming quantum computers) is coupled with investments in human capital, such as funding HPC and AI doctoral programs.

Starting in 2025, the HPC Training ecosystem has been gathered around the European AI Factories. AI Factories are dynamic ecosystems that foster innovation, collaboration, and development in the field of artificial intelligence. They bring together computing power, data, and talent to create cutting-edge AI models and applications. They foster collaboration across Europe, linking HPC centres, universities, small and medium sized enterprises (SMEs), industry, and financial actors. AI Factories serve as hubs driving advancements in AI applications across various sectors such as health, manufacturing, climate, finance, space, and more.

Italy is hosting one of the major AI Factories, under the name IT4LIA[21]. In this strategic initiative, designed to transform the national and European AI landscape through cutting-edge HPC infrastructure and a comprehensive suite of services, training is an essential part. Training services offer tailored learning pathways designed to equip users with the skills needed to effectively develop and deploy AI solutions on HPC infrastructure. The training offer includes both foundational and advanced modules, and can be customized to match the specific needs of each use case, covering topics such as data management, AI model development, regulatory compliance, and domain-specific applications.

Responding to the recognized talent shortage (indeed, ~70% of EU companies cite lack of digital and HPC skills as a barrier to investment[22], Europe's multiple fronts HPC education initiatives from coordinating national centers (EuroCC) and fostering exchange (CASTIEL), to innovating academic curricula (EUMaster4HPC) and building common training frameworks (EVITA, competency maps), to AI Factories, represent one of the most structured approaches globally. Challenges remain (e.g., sustaining funding for educational programs, ensuring programs like EUMaster4HPC become permanent, and continuously updating content), but the European case provides valuable lessons. Other regions, such as the US and Asia, are observing this model; for instance, analogous efforts to coordinate HPC training across institutions are being discussed in the US under the banner of NSF's Advanced Cyberinfrastructure programs.

[21] https://it4lia-aifactory.eu/
[22] https://ea.consulting/5-reasons-europe-urgently-needs-cloud-hpc-education/

The ICSC Observatory initiatives on HPC education: Pip-squeak

During the PDP 2025 Conference[23] - the 33rd Euromicro International Conference on Parellel, Distributed, and Network-Based Processing - the Authors of this report organized and chaired a special session on HPC Education[24]. Organized within the activities of ICSC Observatory on trends and applications of Supercomputing, the PIP-SQUEAK (Papers In Parallel/distributed computing, Supercomputing, and QUantum computing Education for Advancing students, professionals and citizens Knowledge) special session[25] invited contributions that explore educational strategies, professional training programs, and outreach initiatives aimed at broadening access to HPC and quantum computing knowledge.

This special session was organized to provide educators, researchers, and practitioners with a platform to discuss the challenges and innovations in teaching HPC and quantum computing. The call was particularly open to contributions that focused on interdisciplinary approaches, integrating HPC and quantum computing into existing curricula, and developing resources for lifelong learning. Additionally, the session explored how professional training for scientists and professionals without specific technical expertise can be enhanced to include foundational knowledge of HPC and quantum computing, enabling them to apply these technologies effectively in their fields.

Topics of interest were:

- Curriculum development and pedagogical strategies for HPC and quantum computing education;
- Professional training programs aimed at non-specialists, including scientists, engineers, and industry professionals;
- Integration of HPC and quantum computing concepts into K-12 and higher education;
- Educational tools, resources, and infrastructures for teaching HPC, quantum computing, and related fields;

[23] https://pdp2025.org/
[24] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10974788
[25] https://www.pipsqueak.it/

- Strategies for addressing diversity and inclusion in HPC and quantum computing education;
- Outreach initiatives to increase public awareness and understanding of HPC and quantum computing;
- Case studies and best practices in teaching and training in HPC and quantum computing.

The special session received 11 submissions and 4 papers were accepted to be part of the conference, showcasing a variety of innovative approaches to teaching high-performance and parallel computing.

Moreno Marzolla (2025) introduced the concept of *Parallel Etudes*, a collection of small, self-contained programming assignments designed as accessible teaching tools for undergraduate students. These mini-applications are drawn from domains such as 3D rendering, image processing, and simple simulations, allowing students without advanced mathematical backgrounds to engage with shared-memory, distributed-memory, and GPU programming. This paper received the "PIPSQUEAK Special Session Best Paper Award".

Giorgio Delzanno, Giovanna Guerrini, and Daniele Traversaro (2025) presented an interdisciplinary approach that employs *Sonic Pi*, a music programming language, to introduce concurrency concepts. Their teaching experiments explore student misconceptions about concurrency through code comprehension and composition tasks, highlighting parallels with difficulties encountered in more traditional programming languages.

Giuliano Laccetti, Marco Lapegna, and Ilaria Merciai (2025) described the design and implementation of a *Massive Open Online Course (MOOC)* on parallel and high-performance computing, developed for the Federica Web Learning platform at the University of Naples Federico II. This initiative aims to make cutting-edge HPC content more broadly accessible to diverse audiences through flexible online learning.

Finally, Marco Danelutto (2025) discussed how to design industry-aligned HPC master's programs—leveraging Pisa's shift from "Computer Science and Networking" to "High Performance Computing and Infrastructures" and lessons from EU-Master4HPC—to fuse computing with advanced networking, expand hands-on labs, internships, and co-supervised theses, and train profiles spanning system architecture, software tools, and operations.

The papers presented in this session highlighted a common effort to broaden access to HPC education across different levels and contexts. They stress the importance of designing teaching materials and methodologies that are both engaging and inclusive, whether through simplified mini-applications, interdisciplinary tools like music-based programming, large-scale online courses, or advanced master's programs. Overall, the special session provided a valuable overview of the Italian initiatives in university-level HPC education, offering insights into how different institutions are working to strengthen the training of future generations in parallel and high-performance computing.

Framework: the Big Ideas

Big Ideas in Science Education

According to Harlen and colleagues (2015), Big Ideas are

> *"the key ideas that students should encounter in their science education to enable them to understand, enjoy and marvel at the natural world".*

There are at least two major frameworks for Big Ideas in science education.

The one by Chalmers and colleagues (2017) is situated within the context of STEM education and aims to address the interdisciplinary challenges that STEM integration presents. In response to the struggle to produce in-depth learning (found in the literature), they propose using a framework to scaffold teachers' development of integrated STEM curriculum units. The framework is based on three types of Big Ideas:

> *"within-discipline Big Ideas that have application in other STEM disciplines, cross-discipline Big Ideas, and encompassing Big Ideas" (Chalmers et al., 2017).*

The other framework arises in science education in response to the widespread perception among students that science curricula, often overcrowded and fragmented, are little more than a series of disconnected facts with minimal relevance to their lives. According to Harlen and colleagues (2010), a group of experts in science education,

> *"[p]art of the solution to this problem was to conceive the goals of science education, not in terms of the knowledge of a body of facts and theories, but as a progression towards understanding key ideas – 'Big Ideas' – of relevance to students' lives".*

The importance of Big Ideas

Harlen and colleagues (2015, p. 1) explain that

> *[t]o prosper in this modern age of innovation requires the capacity to grasp the essentials of diverse problems, to recognize meaningful patterns, to retrieve and apply relevant knowledge. [. . .] Science education [. . .] needs to take account of changes in the work place that require ability to link science with engineering, technology and mathematics (STEM), the urgent need for attention to major global issues such as the adverse impacts of climate change, the positive and negative influences of student assessment and the growing contribution of neurosciences to the understanding of learning. All of these add to the reasons for the development of Big Ideas to provide a framework for decisions about science education.*

The authors highlight several compelling reasons why establishing a framework for developing the Big Ideas of science is particularly timely. First, such a framework can help counter students' tendency to view science as a disconnected collection of facts and theories with little personal relevance. It also offers a structure for designing school activities that encourage students to make sense of phenomena they find meaningful. Additionally, it serves as a practical tool for navigating the vast range of possible curricular content by guiding the selection of what to include. The framework supports the organization of science curricula around the gradual development of overarching concepts, promoting coherent learning progressions. It aligns well with the growing emphasis on inquiry-based teaching approaches in science education.

Furthermore, it encourages recognition of the ways STEM subjects relate to diverse real-world contexts, enhancing their perceived value. Ultimately, it facilitates the construction of interconnected systems of knowledge, allowing students to gain a deeper understanding and more nuanced interpretation of their experiences. This approach is also supported by recent findings in neuroscience, which indicate that ideas learned in a connected way are more easily transferable to new situations.

Even more relevant for this report, the Authors recognize the importance of Big Ideas for teacher training. Teachers must consider how individual lessons' objectives fit into a larger framework of broader and stronger ideas that can help students make sense of a wide range of related phenomena and events. This general development direction

is necessary for selecting content and constructing learning pathways. It also clarifies what teachers need to observe and look for in students' actions, informs their decisions about feedback, and helps them adjust formative assessments. This is particularly challenging for teachers who teach science subjects without having the opportunity to study them in depth. Often, the scientific education teachers received did not allow them to develop Big Ideas. Instead, teachers should also have the opportunity to help students understand the Big Ideas.

The core concepts, carefully derived from scientific knowledge for their broad relevance and explanatory power, enable learners to grasp the fundamental aspects of phenomena without needing to know and understand exhaustive details. Engaging with such ideas can foster a sense of coherence in understanding the world, identifying patterns across different contexts, and recognizing how various phenomena are interconnected. This process also clarifies the aims of scientific inquiry and its significance in everyday life, potentially increasing learners' motivation both within and beyond the classroom.

Moreover, a deeper understanding of science supports informed decision-making, especially in areas related to health, the environment, and career paths, thus reinforcing the individual's role as an active and responsible citizen. These personal decisions, however, extend into the collective domain, as they cumulatively influence environmental and societal well-being over time. Recognizing both the positive and negative consequences of applying scientific knowledge is key to appreciating the role of science, cultivating a sense of responsibility and agency in confronting global challenges, and promoting social equity.

How to distil Big Ideas

Scholars have listed the characteristics of Big Ideas. They should:

- "have explanatory power in relation to a large number of objects, events and phenomena that are encountered by students in their lives during and after their school years" (Harlen et al., 2015, p. 14)
- "provide a basis for understanding issues, such as the use of energy, involved in making decisions that affect learners' own and others' health and wellbeing and the environment" (Harlen et al., 2015, p. 14)
- "lead to enjoyment and satisfaction in being able to answer or find answers to the kinds of questions that people ask about themselves and the natural world" (Harlen et al., 2015, p. 14)
- "have cultural significance – for instance in affecting views of the human condition – reflecting achievements in the history of science, inspiration from the study of nature and the impacts of human activity on the environment" (Harlen et al., 2015, p. 14)
- be stable over time and general enough (broad applicability) to observe its manifestations in multiple phenomena of reality (Harlen et al., 2015; Schwill, 1994; Bell et al., 2018)
- make sense - at different levels of detail/abstraction - to students at various levels and throughout their careers (Schwill, 1994; Bell et al., 2018)

To distill the "Big Ideas of CS for K-12 education", of which we will tell more in the following subsection, Bell and colleagues (2018) used a collaborative process that involved CS education experts, curriculum designers, and computer scientists worldwide.

They developed the first list of candidate Big Ideas by having in-depth discussions with several (university and K-12) CS education researchers. Other early versions were created by showing other fellow researchers the Big Ideas of science and asking them what an analogous list for informatics should include. The suggestions received and their initial list were reviewed for repetitions and synthesized into a single list. The list was shared with teachers reasonably new to the subject to ensure that those Big Ideas were clear and meaningful to educators and curriculum designers. The Big Ideas they propose result from several iterations of the process. The list should not be considered definitive and is open to feedback and input from the entire research and education community.

Big Ideas of CS

The tendency found among teachers to focus on the details while losing sight of the bigger picture when teaching CS prompted Bell and colleagues (2018) to formulate the "Big Ideas of CS for K-12 education".

The technological side of CS is running, giving the impression that new content must always be chased. In this scenario, the tendency to focus on marginal and technological aspects is exacerbated by the lack of specific preparation of a large proportion of teachers asked to teach informatics without having studied it.

Explicitly referring to Harlem's framework, Bell and colleagues underline that

> Computer science is full of paradoxes and surprises that provide opportunities for students to understand, enjoy and marvel at the digital world, and we are particularly interested in ensuring these are captured so that the ideas are in line with the intention of the "Big Ideas of science education". (Bell et al., 2018, sect. 1)

The Big Ideas of CS are not intended as general principles, disciplinary areas, or even curriculum topics but as powerful and general ideas that capture the essence of CS. The Big Ideas are what young students should retain, regardless of whether they will specialize in CS or apply its core knowledge to exercise their right to citizenship; the focus is on concepts that are fundamental in CS but not immediately apparent to the layperson.

The Big Ideas of CS are formulated so that teachers can use them with students to connect the various topics within the broader context of a long-standing discipline. To do so, they do not focus on specific technologies, allowing them to remain the same despite technological advances. The following is an example that Bell himself provides regarding the fundamental concept of programming versus its technological aspects, specifically programming languages.

Some might argue that students should use a language used in industry, such as

> Java or C#, but if we focus on the Big Ideas in programming, writing programs is about getting a computational device to follow the steps in an algorithm, and we know that any algorithm can be programmed using just six key elements: input, output, storage, sequenced instructions, iteration (loops) and conditionals (if and case

> statements). These six elements make a computing system *Turing Complete* [. . .] fully capable of computing anything that a conventional computer could compute [. . .]. If students learn about all six of these elements and how to combine them to achieve a desired result, then they have been exposed to the full power of computation [. . .]. (Bell and Duncan, 2018, sect. 10.7)

Knowing and understanding the significant landmarks of CS can help to see its bigger picture. In particular, the Big Ideas aim to inform curriculum design and, more importantly, to enable (particularly non-specialist) teachers to understand CS in its fundamentals.

Indeed, without a general perspective, teachers may perceive CS as an imposition on their limited classroom time; non-specialist teachers may feel overwhelmed by a series of extraneous and complicated topics.

Additionally, teachers can use Big Ideas to distinguish core knowledge from skills.

> *"For example, programming is just one of our ten Big Ideas, but in many curricula considerable time is spent on this, partly because it is a skill that can require some time to acquire. Conversely, there will be other topics around learning to use or configure computers that appear in curricula but not the Big Ideas because students need to learn to do these things, even though they aren't necessarily a fundamental concept from computer science" (Bell et al., 2018, sect. 1).*

Skills, like programming, are acquired over time, while core knowledge enables students to gain a broad understanding of CS that is useful to support learning the proposed topics.

As for curriculum design, the Big Ideas are intended to help curriculum designers focus on core knowledge and topics, enabling teachers to recognize the long-term value of CS in these curricula. Referring to stable, non-technology-dependent Big Ideas helps build curricula composed of relevant topics that will not soon become obsolete, rather than curricula that must be updated every time a new technology is developed. Also, a big picture helps ensure that curricula are not built around not-so-relevant topics chosen simply because their teaching resources are available in textbooks or online.

The Research

Research Questions

This study is guided by two overarching questions.

1. What conceptual needs are perceived at the academic and industrial level when addressing High-Performance Computing (HPC)?
2. What are the "Big Ideas" of HPC — the fundamental principles that should inform educational design and training?

Methodology

To address these questions, we adopted a mixed qualitative methodology inspired by previous work on Big Ideas frameworks in science and computer science education (Bell et al., 2018; Harlen et al., 2015). The research was designed as an iterative, community-engaged process combining literature reviews, expert consultation, and survey-based data collection.

Literature Review

We began with an extensive review of existing literature on HPC education, focusing on both curricular needs in academia and skill requirements in industry. In parallel, we reviewed the body of work on "Big Ideas" in science and computer science education to establish a conceptual framework adaptable to HPC.

Expert Consultation

Individual interviews and focus groups with educators, researchers, and business professionals were held throughout the process, ensuring iterative feedback on emerging candidate needs and ideas. Early findings were shared and debated during the *PIPSQUEAK* special session at the PDP 2025 conference, fostering interdisciplinary exchange and validation.

Iterative Drafting of Educational Needs and Big Ideas

Preliminary educational needs and candidate concepts to formulate Big Ideas were drafted through aforementioned expert consultations and the analysis of textbooks, course syllabi, and curricula in HPC at undergraduate and graduate levels. These drafts were refined through brainstorming sessions with experts and continuous discussion within the ICSC Observatory board.

The process was indeed intentionally iterative and open-ended. The list of educational needs and HPC Big Ideas presented in this report should not be seen as definitive, but rather as a first step towards a shared framework, subject to further validation by educators, policymakers, and the broader HPC community. Future iterations will expand the survey base and incorporate structured feedback loops to refine and consolidate the framework.

Questionnaire-Based Survey

After conducting a few initial exploratory interviews with experts to investigate our research question, we decided to opt for a questionnaire-based survey methodology. This approach was selected for several reasons. First, the questionnaire enabled us to systematically collect insights from a diverse group of HPC professionals, including researchers, educators, managers, data scientists, and technical specialists. Moreover, the structured nature of the questionnaire ensured consistency in how questions were presented to all participants, reducing variability that might otherwise arise from differing interview styles or phrasing. The digital format of the questionnaire also allowed us to reach a geographically dispersed community of HPC experts, thereby maximizing the diversity of responses and ensuring representation from different sectors and levels of experience within the field.

The analysis of the collected responses is available in Annex 1 – Questionnaires' analysis and results.

Questionnaire Design

The questionnaire design was grounded in educational theory regarding "Big Ideas" as proposed in computer science education literature and presented in the sections above. The whole questionnaire is still available for filling-in at https://forms.office.com/e/sej2Zpy8ma.

The questionnaire began with an extensive explanation of what constitutes a "Big Idea," including positive and negative examples from computer science. This framing was essential to ensure participants understood the level of abstraction and generality required. In particular, we stated that Big Ideas should be "key, powerful, fundamental, but non-obvious messages or concepts for a layperson" that serve multiple pedagogical functions: i) Guide instructors in designing curricula without losing conceptual direction; ii) Enable learners to perceive the discipline as coherent rather than fragmented; iii) Provide enduring knowledge that transcends rapidly changing technological details; iv) Foster imagination about applications even without technical expertise.

The questionnaire was structured into four main sections. Questions moved from concrete (demographic information) to abstract (Big Ideas identification) to applied (educational implications), allowing participants to build context before addressing the core question. Specifically, the four sections were:

1. Demographic and professional background (Questions 1-7). These questions established participant credibility and context, including: professional role in HPC; years of experience; teaching experience; application sectors; frequently used technologies and tools. This information was crucial for understanding the perspective from which participants would identify Big Ideas and for potential subgroup analysis.

2. Core Big Ideas identification (Questions 8-9). The central questions directly addressed the research question. Question 8 asked for a single, well-formulated Big Idea. Question 9 solicited additional Big Ideas. These open-ended questions were deliberately positioned after the demographic questions to ensure participants had considered their expertise and context before responding. By including both primary (Question 8) and supplementary (Question 9) Big Ideas questions, we acknowledged that experts might identify multiple fundamental concepts and provided flexibility in response depth.

3. Educational implications (Questions 10-11). These questions explored how identified Big Ideas should influence HPC education in academic versus industrial contexts, providing practical validation of the concepts' importance.

4. Future directions (Questions 12-14)- The final questions addressed the broader educational implications, including pre-university education and opportunities for follow-up interviews.

Questionnaire implementation

The questionnaire was implemented using Microsoft Forms, providing a user-friendly interface while ensuring data security and GDPR compliance. The questionnaire included comprehensive informed consent procedures with separate agreements for data processing for research purposes and participation in the study. Both consent forms referenced detailed privacy policies, ensuring ethical compliance with research standards.

Survey dissemination

The survey dissemination strategy was designed to systematically reach expert practitioners and researchers within the High Performance Computing community across multiple professional networks and venues. The primary objective was to collect informed perspectives from individuals with substantial expertise in HPC research, education, and industry applications rather than seeking broad general participation.

The questionnaire was strategically introduced through several key channels that provided direct access to HPC experts. The survey was presented at all advisory board meetings of the ICSC (Italian Research Center on High Performance Computing, Big Data and Quantum Computing) observatory[26], ensuring participation from senior researchers and institutional leaders who provide strategic guidance for national HPC initiatives. These advisory board presentations allowed for direct engagement with participants who could provide contextual feedback and ensure the questionnaire's relevance to current HPC research and educational priorities.

The research conferences were one of the most crucial contexts in which the survey was disseminated. The first was the SIF (Italian Physics Society) conference held in Bologna in 2024[27]. During the Symposium on Supercomputing "Emerging data and computing technologies for frontier physics", the authors of the white paper participated with a talk on the Big Ideas for HPC education, presenting the survey and asking the symposium's participants to fill it in. During the PDP (Parallel, Distributed and network-based Processing) conference held in Turin in 2025[28], the survey was actively

[26] https://osservatorio.supercomputing-icsc.it/chi-siamo/
[27] https://www.sif.it/attivita/congresso/110
[28] https://pdp2025.org/

disseminated through the PIPSQUEAK dedicated special session organized specifically for this research initiative.

The questionnaire was also presented during working group sessions of the EuroCC Training Division, reaching professionals specifically focused on HPC education and training initiatives across European national competence centers. These working groups included educators, training coordinators, and curriculum developers who possessed both technical HPC knowledge and practical experience in educational program design and implementation.

Additionally, the survey was introduced through CASTIEL working groups[29], providing access to experts involved in computational science and engineering applications of HPC. This channel ensured representation from professionals who bridge the gap between HPC technology development and domain-specific scientific applications.

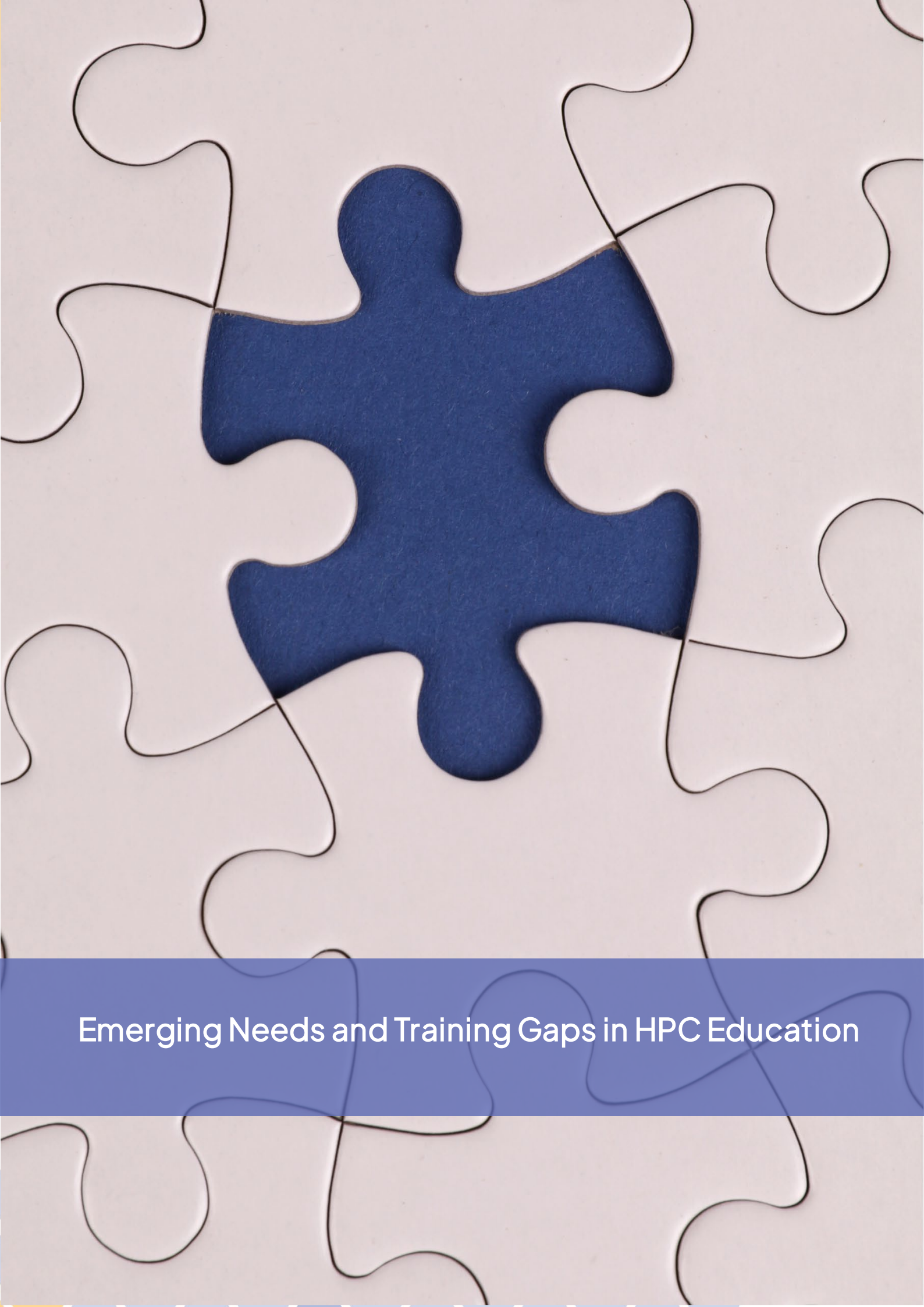
The CINI (National Interuniversity Consortium for Informatics) newsletter served as a broader dissemination channel, reaching the Italian academic computing community and providing opportunities for participation from researchers and educators who might not have been reached through the more specialized venues described above. Also the mailing list of the ACM SIGHPC Education Chapter was used to disseminate the survey, ensuring a broad range of international scholars and professionals interested in HPC Education.

This multi-channel approach ensured that the survey reached experts across different aspects of the HPC ecosystem, including research, education, industry applications, and policy development, while maintaining focus on individuals with substantial professional expertise rather than seeking broad public participation.

The dissemination efforts resulted in 13 completed responses from HPC experts. While this number may appear modest relative to the extensive dissemination activities undertaken, it reflects the inherently specialized nature of the HPC professional community and the demanding nature of expert participation in research studies. The HPC field represents a relatively small, highly specialized professional domain where practitioners with the depth of expertise required for this study constitute a limited population. Furthermore, the qualitative nature of this research prioritizes the depth and quality of expert insights over statistical representativeness. The responses obtained demonstrate high confidence levels, as participants possessed substantial

[29] https://www.eurohpc-ju.europa.eu/research-innovation/our-projects/castiel_en

professional experience and provided thoughtful, detailed contributions that reflect deep understanding of HPC principles and educational challenges. This study should be understood as a qualitative investigation employing grounded theory methodology (Glaser & Strauss, 1967) to capture and analyze expert perspectives, aimed at generating theoretical insights about HPC education rather than producing statistically generalizable findings across broader populations.



Emerging Needs and Training Gaps in HPC Education

Results

Emerging Needs and Training Gaps in HPC Education

High-Performance Computing is central to tackling major scientific and societal challenges, but current educational offerings reveal persistent skill and conceptual gaps.

Foundational HPC skills missing in curricula

A central gap in HPC education is the absence of core skills in many undergraduate programs. Parallel programming with MPI or OpenMP, concurrency, performance optimization, and the principles of HPC system architecture are often treated as advanced or elective topics rather than fundamentals. As a result, computer science graduates frequently complete their studies without exposure to scalable parallel algorithms or the practical realities of working with multi-core and distributed systems. In parallel, students from science and engineering disciplines may rely on HPC-driven applications without developing the computational literacy to understand scalability, memory hierarchies, or communication costs. This leaves both groups ill-prepared to design, implement, or critically evaluate HPC solutions.

Software optimization gaps

Another significant weakness lies in software optimization training. Students and professionals are typically proficient in high-level tools such as MATLAB, R, or Python, which remain the lingua franca of research and industry but are rarely optimized for HPC environments. By contrast, fewer are comfortable with performance-critical languages like C and C++, or with low-level GPU programming through CUDA or HIP. Even when these languages are taught, systematic exposure to profiling, debugging, and performance tuning tools is uncommon. Without these skills, learners may be able to run applications on a supercomputer but struggle to identify bottlenecks, adapt code for heterogeneous architectures, or exploit the full capacity of the hardware. This gap perpetuates a culture of inefficient HPC usage where significant compute resources are consumed without proportional scientific or industrial return.

Data-intensive computing deficits

Modern HPC is increasingly dominated by data-intensive workflows, yet education often lags behind in teaching how to manage and process large-scale data. Few curricula cover parallel I/O, data locality strategies, or storage hierarchies in depth. Similarly, workflow orchestration, reproducibility, and verification practices are rarely

introduced systematically, despite being central to credible computational science. Students may learn how to execute simulations but not how to handle terabytes of resulting data, document their processes, or ensure reproducibility across systems and teams. This training gap undermines the reliability of HPC-enabled research and limits the ability of graduates to participate in emerging data-driven domains where the integration of computing and data governance is paramount.

Conceptual misunderstanding of HPC environments

A recurring issue for new HPC users, particularly those from non-computing backgrounds, is a fundamental misunderstanding of what HPC systems are and how they operate. Many perceive a cluster as simply a "bigger laptop," approaching it with expectations shaped by personal computing environments. Without training in command-line interfaces, resource schedulers such as Slurm or PBS, distributed memory concepts, or the modular software environments typical of HPC, users often experience frustration and inefficiency. This conceptual gap creates barriers to effective adoption, as learners may be able to execute simple tasks but lack the orientation to use HPC systems productively, responsibly, and at scale.

Integration of AI and data analytics

The convergence of HPC with artificial intelligence and big data analytics (often grouped under High-Performance Data Analytics, or HPDA) is reshaping the skill set required of practitioners, yet training programs have only partially caught up. Few curricula explicitly address distributed model training, the use of large-scale GPU resources for AI workloads, or the design of hybrid HPC/AI workflows where simulation, data analysis, and machine learning interact seamlessly. This leaves graduates with fragmented expertise—computer scientists who understand simulation but not AI at scale, or data scientists proficient in machine learning but unfamiliar with HPC infrastructures. Bridging this gap is critical as HPC increasingly underpins AI research and industrial adoption, demanding professionals fluent in both paradigms.

Emerging technologies underrepresented

Finally, educational programs often neglect the rapidly evolving ecosystem of emerging HPC technologies. Training on heterogeneous computing—programming for GPUs, FPGAs, and other accelerators—remains inconsistent, despite their ubiquity in modern HPC systems. Likewise, cloud-based HPC and containerization technologies (such as Docker, Singularity, or Podman), which are now central to reproducibility and portability, are rarely embedded in standard curricula. Even more forward-looking areas, like the integration of quantum accelerators with classical HPC, are largely absent.

This lack of exposure means graduates may be competent in traditional HPC paradigms but unprepared for the heterogeneous, hybrid, and cross-technology infrastructures that define the next generation of high-performance computing.

Training Gaps for Computer Science (CS) Students and Graduates

Lack of core HPC integration: A first gap for computer science students and graduates lies in the limited integration of HPC fundamentals into the core curriculum. Parallel and distributed computing concepts such as MPI, OpenMP, and GPU programming are frequently offered only as advanced electives or confined to specialized tracks. This leaves many students completing a computer science degree without a systematic understanding of parallelism, concurrency, or scalability. As a result, they graduate with strong sequential programming skills but lack the ability to design algorithms and software that can efficiently exploit modern multicore and distributed architectures.

Insufficient optimization skills: Another critical weakness concerns optimization skills. Even when students learn to write parallel code, they are rarely trained to analyze performance, profile applications, or systematically tune software to exploit large-scale systems. The ability to scale a program from a desktop to hundreds or thousands of cores requires not just coding but also performance engineering practices—using tools to identify bottlenecks, understanding memory hierarchies, and applying algorithmic optimizations. The absence of such training means graduates often know the syntax of parallel programming but not the methodology for achieving efficiency on HPC infrastructures.

Weak exposure to data workflows: A further gap relates to the handling of data workflows. HPC today is not limited to computation but increasingly involves complex data pipelines, from parallel I/O to workflow orchestration and reproducibility practices. Yet these aspects are seldom emphasized in computer science education. Students may be introduced to algorithms and systems but receive little guidance on how to manage large volumes of simulation data, design portable workflows, or ensure reproducibility across heterogeneous environments. This omission is problematic, since robust data practices are now as central to scientific computing as code correctness.

Narrow focus: Finally, computer science training often maintains a narrow focus on theoretical knowledge at the expense of practical exposure. Courses tend to emphasize algorithmic complexity, architecture concepts, or formal models without

sufficiently embedding opportunities to interact with real HPC environments or applied scientific problems. As a consequence, graduates may be theoretically well-prepared but inexperienced in deploying and testing applications on actual HPC platforms. This creates a disconnect between academic preparation and the demands of research centers or industry, where proficiency in navigating HPC environments and applying HPC to real-world challenges is indispensable.

Training Gaps for Domain Students and Researchers (Non-CS)

Low computational literacy: A central training gap for domain students and researchers is low computational literacy. Many learners in disciplines such as physics, biology, or engineering lack foundational skills in programming, Unix environments, or the use of job schedulers and parallel computing models. When these students encounter HPC resources, they often approach them with the mindset of desktop computing, without understanding the principles of concurrency, distributed memory, or resource allocation. This creates significant barriers to entry and prevents them from using HPC systems effectively, even when such resources are essential to their research.

Overreliance on domain tools: Another weakness stems from an overreliance on domain-specific software packages. Researchers frequently learn to operate specialized tools such as GROMACS in chemistry or Abaqus in engineering, but they do so as end users rather than developers. While these packages allow them to run simulations on HPC systems, they do not equip them with the skills to adapt, extend, or optimize software for new problems or larger scales. This dependence limits their autonomy and makes them reliant on a small pool of computational experts, slowing innovation and reducing flexibility in applying HPC to emerging research challenges.

Bridging gap: A persistent bridging gap also affects this group. Many STEM curricula assume that computing expertise lies outside their disciplinary scope, leaving students without basic training in programming, data structures, or scalability concepts. The absence of introductory computing courses tailored for non-CS audiences means that domain students must acquire essential skills informally, often during doctoral studies or when they first encounter HPC in practice. This late start makes the learning curve steeper and reduces the potential for interdisciplinary innovation that could emerge if CS fundamentals thinking were embedded earlier in their education.

Contextual training missing: Domain learners also face a lack of contextualized training opportunities. Even when HPC concepts are introduced, they are often presented in abstract or technical terms rather than in relation to the student's field of study. For example, engineers benefit most from seeing HPC applied to digital twin simulations of industrial processes, while biologists engage more effectively when training includes genomics pipelines or molecular dynamics. Without such contextual relevance, HPC education remains disconnected from the problems domain students care about, reducing motivation and hindering knowledge transfer.

Awareness gap: Finally, there is a pronounced awareness gap regarding the convergence of HPC with AI, big data, and emerging technologies such as quantum computing. Many domain researchers are unfamiliar with how large-scale data analytics or machine learning workflows can be accelerated and scaled on HPC systems. Similarly, they lack exposure to forward-looking paradigms like quantum-HPC integration, which are likely to influence their fields in the coming decades. Without structured training in these areas, domain students and researchers risk being excluded from cutting-edge developments, perpetuating a divide between disciplinary expertise and computational innovation.

Training Gaps in Academia (Educators, Students, Researchers)

Curricular fragmentation: A persistent gap in academia is the fragmentation of curricula. HPC topics are not systematically included across disciplines and remain inconsistently integrated between institutions. While some universities offer dedicated modules in parallel computing, many others treat HPC as a niche subject, relegated to electives or specialized graduate programs. This lack of consistency means that students' exposure to HPC depends heavily on where they study, leading to unequal preparation across the academic landscape.

Educator preparation gap: Another major challenge is the limited preparation of educators. Only a small proportion of faculty possess deep expertise in parallel and distributed computing, and even fewer have formal training in pedagogy specific to HPC. Many courses are taught by researchers or technical staff who are experts in their field but unfamiliar with effective teaching strategies for such complex and abstract material. This shortage of qualified and well-prepared instructors reduces the quality and

scalability of HPC education, leaving institutions dependent on a small pool of specialists.

Reproducibility underemphasized: Reproducibility and open science practices are also underemphasized in academic HPC training. Although reproducibility is central to scientific credibility, students are rarely taught to document workflows, manage computational environments, or verify results across heterogeneous platforms. Without explicit instruction in these practices, future researchers risk perpetuating non-reproducible work, undermining the scientific value of HPC-driven research. The use of proprietary software exacerbates the problem. Embedding reproducibility into curricula is essential to align HPC education with broader movements toward transparency and open science.

Late exposure: HPC concepts also tend to be introduced too late in the academic trajectory. Concurrency, scalability, and parallel thinking are often reserved for Master's or doctoral studies, by which time students may have already developed habits of sequential programming that are difficult to unlearn. Introducing these concepts earlier in undergraduate education would normalize parallel thinking and ensure that students approach computing with scalability in mind from the outset. The current late exposure represents a missed opportunity to cultivate computational literacy as a foundational skill.

Infrastructure access limits: Finally, many students suffer from limited access to HPC infrastructure during their studies. While theoretical instruction is available, hands-on experience with real clusters or supercomputers is frequently absent. In some institutions, students may only experiment with small-scale simulations on personal laptops, without learning to interact with schedulers, queues, or large distributed systems. This lack of practical exposure creates a disconnect between academic training and the environments graduates will encounter in research centers or industry, where familiarity with real HPC systems is a baseline requirement.

Training Gaps for Industry Professionals and Practitioners

Missed foundation: A fundamental gap for industry professionals is the absence of HPC foundations in their formal education. Many engineers, analysts, and technical staff never encountered HPC concepts during university studies, which means they arrive in the workplace with strong domain expertise but no understanding of parallel programming, distributed computing, or performance scaling. This creates a steep

entry barrier: professionals must learn the basics of HPC while simultaneously applying it to real-world problems, a process that slows adoption and reduces efficiency in industrial contexts.

Open standards adoption: Another pressing issue is the limited adoption of open standards. Companies often rely on proprietary tools and vendor-specific solutions that solve immediate needs but restrict long-term flexibility. Training on open-source libraries and interoperability standards such as MPI, OpenMP, OpenACC, and containerization practices is essential to build software and workflows that are portable across platforms and avoid costly vendor lock-in. Without such training, organizations risk siloing their HPC capabilities and losing competitiveness as technologies evolve.

Training formats misalignment: The alignment of training opportunities with actual professional needs is also problematic. Many available workshops are either too basic, repeating concepts that participants already know, or too advanced, requiring prior knowledge that many lack. Intermediate-level modules that focus on bridging gaps are scarce. Professionals also require short, flexible learning formats—micro-courses, modular workshops, or online training—that can be integrated into demanding work schedules. The current mismatch reduces the accessibility and impact of training for industry staff.

Continuous upskilling need: Continuous upskilling represents another critical challenge. The HPC landscape evolves rapidly, with new GPU architectures, specialized accelerators, and hybrid workflows combining HPC, AI, and cloud technologies. Professionals in industry must continually retrain to keep pace with these changes, but structured opportunities for ongoing education are limited. Without systematic pathways for upskilling, companies struggle to maintain teams capable of exploiting state-of-the-art HPC resources.

Certification gap: Certification is another area of unmet need. Both professionals and employers value recognized credentials to benchmark HPC skills and to validate that training outcomes translate into workplace competence. Yet standardized certification frameworks are still emerging, and many current training efforts provide no formal recognition. The absence of widely accepted certifications reduces the incentive for individuals to pursue training and limits employers' ability to assess workforce readiness.

Knowledge transfer weakness: Knowledge transfer within companies is also weak. Expertise in HPC is often concentrated in a handful of specialists who carry much of the

organizational know-how. This creates bottlenecks and risks: if those experts leave or change roles, critical competencies are lost. Broader team-level training, mentoring, and structured documentation practices are rarely in place, meaning that HPC knowledge does not disseminate effectively across engineering or R&D teams.

Access challenges: Finally, access to infrastructure poses a significant barrier, especially for small and medium-sized enterprises (SMEs). Unlike large corporations or research centers, SMEs typically lack dedicated HPC resources. While national and European HPC centers, as well as cloud providers, offer opportunities to access HPC power, many companies lack the training needed to navigate these systems, prepare applications, or manage data securely at scale. This lack of capacity building means that SMEs, despite their potential to benefit from HPC, remain underrepresented in its adoption.

Cross-Cutting Gaps (Affecting All Profiles)

Emerging technologies underrepresented: One of the most prominent cross-cutting gaps in HPC education is the limited attention given to emerging technologies. Training programs rarely incorporate content on digital twins, large-scale AI–HPC integration, or the principles of quantum computing. As these fields increasingly shape the future of computational science and industrial innovation, the absence of structured training leaves students and professionals underprepared to work with the hybrid, heterogeneous systems that are already becoming standard. Without exposure to these trends, learners risk graduating with skills that are quickly outdated.

Data management and governance: Data management and governance represent another major deficit. While computation is at the heart of HPC, the ability to manage data efficiently at scale is equally critical. Yet few curricula provide systematic coverage of parallel I/O, storage hierarchies, or strategies for data locality. Even less emphasis is placed on ensuring reproducibility, documenting workflows, or integrating responsible AI and data practices into HPC pipelines. The lack of such training not only undermines scientific rigor but also prevents professionals from meeting industry standards for compliance, auditability, and trustworthy data use.

Sustainability and societal impact: A further gap concerns sustainability and societal impact. HPC is energy-intensive, but training programs seldom address the environmental costs of large-scale computation or strategies for mitigating them. Similarly, little space is given to the broader ethical and societal implications of HPC, such as its

role in climate modelling, healthcare, or national security. Embedding sustainability and ethics into HPC education is essential to prepare professionals who can balance performance goals with social responsibility.

Interdisciplinary communication: Another issue shared across all profiles is the lack of training in interdisciplinary communication. HPC projects nearly always require collaboration between domain scientists, computer scientists, and engineers, yet education continues to train these groups in isolation. Few courses provide opportunities to develop teamwork skills, cross-disciplinary dialogue, or joint problem-solving. This perpetuates silos that slow down innovation and limit the effectiveness of HPC in addressing complex, real-world challenges.

Pedagogical methods: Finally, pedagogical methods themselves contribute to the gap. Although students and professionals consistently express a preference for hands-on, project-based learning, many HPC courses remain lecture-heavy, overly theoretical, and inaccessible to beginners. Entry-level learners in particular often find HPC daunting when it is introduced through abstract models rather than practical exercises. More engaging and inclusive pedagogical approaches—such as interactive labs, collaborative projects, and scaffolded exercises—are urgently needed to lower the barrier of entry and ensure long-term retention of HPC skills.

The Big Ideas of High-Performance Computing

The Big Ideas of HPC

1 Computation Has Physical Limits — and HPC Helps Push Them

Computers are physical devices. Their speed and capability depend on **real-world constraints**: energy, space, heat, and time. HPC exists because we want to solve problems that exceed the capabilities of ordinary systems. From weather prediction to protein folding, HPC helps us push the boundaries of what computation can achieve.

Digging deeper:

- Physical limits (e.g., heat, power, transistor density) ended the era of automatic performance gains via faster clock speeds.
- HPC scales out instead: using many processing units working together.
- This allows us to perform simulations, analyses, and experiments that are infeasible on conventional computers.
- These constraints remind us that computing always comes with trade-offs: energy, money, and time.
- New technologies, such as quantum computing, aim to overcome current physical limits; however, HPC remains the dominant paradigm for solving the most significant and complex problems today.
- Energy efficiency has become a first-class design goal in HPC systems. The push toward exascale computing has made metrics like *FLOPS per watt* as important as raw speed. This focus on energy aligns with the broader goals of green computing, reflecting a societal need to reduce the environmental impact of large-scale computational infrastructure.

2 Parallelism Unlocks Speed and Scale

The key to modern computational power is **parallelism:** doing many things at once. Instead of solving a problem step by step, we divide it into parts that can run simultaneously. Parallelism is not just a technical trick — it's a fundamental principle that lets us think bigger, solve faster, and go further.

Digging deeper:

- Parallelism appears at all levels: within processors (SIMD), across cores (MIMD), and across systems (distributed memory).
- To use parallelism, a problem must be decomposed into independent or semi-independent tasks.
- **Data parallelism:** applying the same operation to different parts of a dataset.
- **Task parallelism:** executing different tasks concurrently.
- Parallel programs require coordination: communication, synchronization, and load balancing.
- **Parallel design patterns** are common strategies for organizing parallel computations. They provide reusable structures for solving classes of problems efficiently. Examples of patterns include *master/worker* (central coordination with distributed work), *pipeline* (stages of processing), *SPMD* (all processors run the same program on different data), and *embarrassingly parallel* (tasks run independently with no need for communication).

3 Performance Comes From Matching Algorithm and Architecture

There is no universally "best" algorithm. Performance depends on how well an algorithm fits the underlying hardware. In HPC, we tune software, data structures, and parallel strategies to match specific architectures.

Digging deeper:

- Memory hierarchies and processor topology influence algorithm efficiency.
- Communication is often more expensive than computation — minimizing data movement matters.
- Parallel algorithms often differ fundamentally from serial ones; effective parallelism may require rethinking the algorithm.

4 Decomposition Is Fundamental

Scalable computing requires **decomposition:** dividing a problem into parts that can be solved concurrently. How a problem is decomposed often determines whether it can be solved in parallel at all.

Digging deeper:

- Data decomposition assigns different parts of a dataset to different processing elements.
- Task decomposition assigns different computations to different elements.
- Decomposition must balance the workload and minimize inter-task communication.
- Effective decomposition is not always obvious; it may require deep domain expertise, problem-specific knowledge, or the use of automated tools to identify parallel structure.

5 Communication Can Be More Expensive Than Computation

In parallel systems, moving data is often the main bottleneck. As systems grow, the cost of communication (in time and energy) can dominate the cost of computation.

Digging deeper:

- Latency is the delay to start communication; bandwidth is the amount of data transferred per unit of time.
- Communication introduces delays, synchronization overheads, and potential contention.
- Minimizing communication is a primary design goal in parallel algorithms.
- Communication cost is not just a bottleneck, but a major factor in energy consumption
- Strategies include ghost cells (duplicated edge data to reduce communication), blocking (organizing data into chunks that fit cache or reduce transfer), overlapping computation and communication (working while data transfers in the background), and communication-avoiding algorithms (designs that minimize the need to exchange data between processing units).

6 HPC Requires Coordination Between Hardware and Software

HPC is a co-design challenge: software and hardware must work together. Knowing how hardware works enables better software; good software unlocks the potential of hardware.

Digging deeper:

- The choice of data layout, memory model, and parallel structure often determines whether a program can effectively exploit modern hardware.
- HPC architectures are increasingly heterogeneous, combining CPUs, GPUs, and specialized accelerators — requiring programmers to understand different execution and memory models.
- Co-design (the joint development of hardware and software with mutual awareness) is becoming a standard approach in HPC system design, ensuring that systems and applications evolve together for optimal performance and efficiency.
- SIMD/vector units enable high-throughput operations on data arrays.
- Coordination encompasses performance modelling, scalability testing, and understanding architectural features such as NUMA or interconnect topology.
- Performance portability — writing code that runs efficiently on many platforms — remains a grand challenge.

7 Scientific Discovery and Innovation Depend on HPC

HPC enables science at scale. From simulating galaxies to designing new materials, it changes what we can model, explore, and discover.

Digging deeper:

- Fields like climate science, genomics, physics, and medicine rely on HPC simulations.
- HPC is also used in industry, where it optimizes supply chains, models financial risk, and trains AI models.
- Many breakthroughs are driven by simulations that would be impossible without parallel computing.
- HPC doesn't just make old things faster; it enables *new* kinds of questions.
- We are in the age of data-driven science. HPC enables the storage, processing, and analysis of petabytes of scientific data — enabling not only simulations, but also data mining, statistical inference, and machine learning.
- HPC is central to large-scale AI training (e.g., LLMs)
- The rise of AI and HPC convergence allows scientists to combine simulations with learning systems that adapt, accelerate, or even replace traditional models

8 Not All Problems Scale — and Some Never Will

Some computations resist parallelization. Others scale poorly due to coordination or hardware limits. Understanding scalability is central to HPC.

Digging deeper:

- Amdahl's Law: even a small serial fraction can limit speedup.
- Gustafson's Law: increasing problem size can maintain efficiency.
- Strong scaling: solving a fixed-size problem faster by using more processors.
- Weak scaling: solving larger problems in the same time by increasing the number of processors proportionally.
- Scalability is limited by serial sections, communication overhead, and algorithm structure.

9 Reproducibility and Precision Matter at Scale

At the HPC scale, minor numerical errors can become significant problems. Scientific computing must be accurate, stable, and reproducible — even in parallel.

Digging deeper:

- Floating-point operations are not associative; different orders of operations can yield different results.
- Parallel reductions (operations that combine values across multiple processors, like summing up an array in parallel) may be non-deterministic unless explicitly controlled.
- Scientific reproducibility also depends on tracking versions, inputs, and computation environments.
- Open-source tools, version-controlled codebases, and well-documented pipelines are essential for ensuring that results can be replicated and verified by others.

10 HPC Thinking Is a Transferable Skill

The skills developed in HPC — parallel reasoning, performance modeling, decomposition — are increasingly relevant across domains, from AI to cloud computing to embedded systems.

Digging deeper:

- The Fourth Paradigm of science (after theory, experiment, and simulation) emphasizes data-intensive discovery. HPC skills — managing data pipelines, optimizing performance, scaling analytics — are foundational to this approach.
- Understanding HPC principles prepares students not just for supercomputing, but for any domain where performance, scale, and complexity matter — including AI, robotics, edge computing, and quantum-classical hybrid systems.
- The same concepts apply in AI model training, real-time analytics, and mobile computing.
- Thinking in terms of coordination, scalability, and performance is functional even outside HPC.
- Education in HPC cultivates a mindset that involves breaking problems down, understanding trade-offs, and reasoning about systems at scale.
- Sustainable computing is an emerging field of expertise. HPC thinking fosters awareness of resource efficiency, which is crucial not only for performance but also for designing greener and more responsible technologies.

The list should by no means be considered definitive. We are warmly open to feedbacks. Be part of our research.

Recommendations for Instructional Design

From Results to Recommendations for Instructional Design

The analysis of current needs and challenges in HPC education can be translated into concrete recommendations for designing more effective instructional programs. In light of our findings – including the "Big Ideas" identified by experts and the gaps in skills for both academia and industry – we propose several strategies to improve HPC education and training.

Integrate HPC fundamentals across curricula

Universities should embed core parallel, distributed, and high-performance computing concepts into relevant degree programs (not only specialized tracks). This means introducing students to parallel programming, algorithmic scalability, and performance considerations as part of computer science or data science courses, rather than treating HPC as an isolated niche. Given that most current computing programs do not cover HPC topics at all (Chen, Ghafoor, & Impagliazzo, 2022), curricular reform is needed to ensure graduates acquire at least basic HPC knowledge and skills. Early exposure to HPC core concepts will better prepare students for the computational demands of modern science and engineering.

Adopt a " Big Ideas" framework for curriculum design

To bring structure and coherence to HPC education, instructors should identify a set of overarching Big Ideas (fundamental principles) and organize coursework around them. Our research indicates that HPC teaching has often been ad-hoc and inconsistent, an issue that a big-ideas approach can mitigate. By centering the curriculum on enduring concepts – for example, "parallelism yields performance (and its limits)," "scalability," "locality and data movement costs," etc. – educators can ensure that students grasp the high-level principles that will remain relevant even as specific hardware and software technologies evolve. This approach also aids in creating shared terminology and definitions in a field where multiple notations and viewpoints currently coexist (Raj et al., 2020). In short, a principles-first (rather than tool-centric) curriculum makes HPC learning more accessible and lasting, which is especially important for newcomers.

Build capacity and community for HPC educators

Expanding HPC education will require investing in those who teach it. Institutions and policymakers should encourage the formation of an HPC education community of practice, where experienced HPC researchers and practitioners contribute to teaching and curriculum development (Raj et al., 2020). Bringing active HPC experts into the classroom can help keep course content up-to-date with fast-evolving technology. At the same time, current faculty need support to gain HPC pedagogical skills. Providing training workshops for instructors, along with openly shared teaching materials (lecture notes, tutorials, assignments), is vital so that individual teachers don't have to reinvent the wheel. Only a small number of faculty members today have the expertise to teach HPC in depth (Raj et al., 2020), so leveraging collaborations (for instance, co-teaching across universities or virtual courses) can reduce the burden on any single instructor. By nurturing a professional community for HPC educators, we can accelerate knowledge transfer and disseminate best practices in teaching parallel computing.

Integrate data management and AI governance into HPC curricula

As data becomes the central driver of scientific discovery and innovation, HPC education must equip learners not only with computational skills but also with the ability to responsibly handle and govern data-intensive processes. Training should cover the technical foundations of large-scale data management (parallel I/O, storage hierarchies, data locality, reproducibility of workflows), as well as principles of responsible AI, including transparency, bias mitigation, and auditability of models trained at scale on HPC systems. Students should also be introduced to the regulatory and societal dimensions of data, such as privacy, data sovereignty, and compliance with international standards. Pedagogically, these topics can be embedded into interdisciplinary projects where computing, data science, and domain-specific knowledge converge—for example, case studies in climate modelling, healthcare, or finance that highlight both technical performance and responsible data use. By integrating these perspectives, HPC education will prepare professionals who are not only technically proficient but also capable of leading data-driven innovation in ways that are ethical, sustainable, and aligned with societal needs.

Ensure access to practical HPC environments for learners

Effective HPC instruction must be experiential. Students need opportunities to run code on multi-core and distributed systems, experiment with real parallel workloads, and observe performance behaviour at scale. We recommend that programs provide accessible HPC platforms or simulators as part of coursework. This could involve using cloud-based clusters, on-premise training supercomputers, or even small "mini-supercomputer" labs. The key is to let students move beyond toy examples – they should grapple with issues of synchronization, communication latency, load balancing, and resource management in a realistic setting. General-purpose computing labs are not sufficient for this purpose (Raj et al., 2020). Universities and consortia might collaborate with national HPC centers to grant students accounts on larger systems, or use containerized HPC environments for teaching. Additionally, assignments should include scaling studies (e.g. measure how runtime changes from 2 to 32 processors) and use of performance analysis tools, so that graduates are comfortable with the tools and techniques of HPC tuning. By encountering the practical constraints (queue systems, memory limits, etc.), students also learn invaluable skills for real-world HPC usage. Allocating resources for educational HPC access – possibly via initiatives like Europe's HPC Training Platform and AI Factories – will significantly enhance instructional outcomes.

Tailor HPC training to different learner profiles with interdisciplinary approaches

HPC education should not adopt a one-size-fits-all mentality, because the needs of computer scientists versus domain scientists (and industry engineers versus academics) can differ. For students in scientific disciplines without a CS background, curricula should emphasize the essential computing concepts needed to use HPC effectively – for example, understanding parallel processing basics (processes, threads, synchronization), parallel I/O and data management, and the notion of scalability and speedup limits (Raj et al., 2020). These learners benefit from a focus on applying HPC to their domain problems (e.g. running simulations, data analysis), with computing material delivered in an accessible way (perhaps via high-level frameworks or interactive

environments) once the concepts are established. In contrast, for computer science and engineering students, programs can delve deeper into HPC internals: covering a wider array of parallel algorithms, the architecture of supercomputers and accelerators, classic challenges like race conditions and deadlock, and advanced topics like parallel design patterns (Raj et al., 2020). This two-track emphasis ensures that each audience gets the appropriate foundation – domain specialists gain computational thinking, while computing specialists gain system-level expertise – yet also finds common ground in core HPC principles. Moreover, HPC training initiatives should encourage interdisciplinary teamwork (e.g. mixed teams of CS and science students solving a problem), reflecting the collaborative nature of modern computational science. Such tailoring and integration will produce graduates who can communicate across disciplines and collectively push forward HPC adoption in both research and industry.

Provide early exposure to HPC concepts

Although still at an early stage, K–12 initiatives in HPC and parallel computing demonstrate that young students can meaningfully engage with fundamental ideas such as parallelism when taught through appropriate methods. Hands-on activities (e.g., building small clusters, using visual analogies) and competitions can spark curiosity and demystify HPC at an early age. The main challenge lies in scalability—moving from isolated pilot workshops to broader curricular integration—which will require targeted teacher training and the establishment of curricular standards for CS in K-12 that explicitly include parallelism.

Integrate reflections on energy, sustainability, and societal impact

HPC education should not focus solely on technical efficiency but also encourage learners to critically reflect on the environmental footprint and broader societal implications of supercomputing. Training should address topics such as the energy consumption of large-scale systems, strategies for sustainable computing, and the role of HPC in tackling global challenges (e.g., climate modelling, healthcare, disaster preparedness). Embedding these reflections into curricula will foster a generation of professionals who not only master the technical aspects of HPC but are also equipped to make responsible decisions about its use in society.

Integrate emerging trends into HPC curricula

As the HPC landscape rapidly evolves, instructional design should anticipate and incorporate connections with emerging paradigms such as artificial intelligence, quantum computing, and digital twins. These technologies are increasingly converging with HPC, reshaping both research and industry applications. By embedding modules or case studies on HPC–AI integration, quantum acceleration, and the use of digital twins for simulation and optimization, curricula can remain future-oriented and adaptable. This ensures that learners develop both strong HPC fundamentals and the agility to engage with the cutting-edge technologies that will define the next generation of high-performance computing.

Strengthen links between academia, industry, and HPC research centers

Instructional design should leverage partnerships outside the traditional classroom to keep content relevant and to facilitate knowledge transfer. We recommend incorporating industry and research experiences into HPC education wherever possible. This can range from internships and project partnerships (where students work on real HPC applications in companies or national labs) to inviting industry experts for seminars or co-developing course modules on state-of-the-practice technologies. The new EuroHPC master's curriculum has made industrial internships a mandatory component, underscoring the value of hands-on experience with production HPC environments (Bouvry et al., 2025). Such experiences expose students to cutting-edge HPC use cases (from automotive to pharma to finance) and modern tools/workflows that may not yet appear in textbooks. They also help bridge the cultural gap between academic theory and practical skill – for example, students learn about software optimization under real constraints, or about teamwork in large HPC projects. Academia-industry cooperation can further ensure that topics like open standards and reproducibility are emphasized, since industry seeks graduates who can write portable, reliable HPC code. By aligning educational outcomes with the needs of employers and the direction of research, these partnerships make HPC programs more dynamic and impactful. In summary, a tight coupling of classroom learning with the broader HPC ecosystem – HPC centers, companies, and international projects – will yield a generation of professionals well-equipped to advance the state of the art.

Recognize and certify skills

We recommend that stakeholders closely watch and contribute to efforts like the HPC Certification Forum. Having widely recognized HPC certifications or badges can help in multiple ways: it provides motivation for learners (a clear goal to aim for), helps employers identify qualified talent, and sets a concrete target for educational programs to align with. Academia-industry collaboration in defining these skill standards is crucial to ensure they meet real-world needs. In the interim, universities might consider offering their own certificates in HPC (for instance, a certificate program requiring a set of HPC-related courses and a project) to formally acknowledge students' specialization, which can be noted on a transcript or CV. The micro-credential approach, experimented in EU Master for HPC (Bouvry et al., 2025), is a promising route to explore.

Support diversity and inclusion in HPC education

Efforts to strengthen HPC education must also address the persistent underrepresentation of women and minority groups in computational science and engineering. Diversity is not only an equity issue but also a driver of innovation, as heterogeneous teams have been shown to produce more creative solutions and broader perspectives on complex challenges. International initiatives provide useful models: programs such as *Women in HPC* (established at SC15 and now active globally) have created mentorship networks, training workshops, and visibility campaigns to support women entering and advancing in high-performance computing. Similarly, outreach projects in the U.S. and Europe have sought to engage underrepresented groups through summer schools, scholarships, and targeted recruitment. To build on these experiences, curricula should incorporate inclusive teaching practices (e.g., using diverse role models, designing entry points accessible to learners with varying levels of prior experience) and provide structured mentoring opportunities for students from underrepresented backgrounds. Institutional policies can further support diversity by sponsoring participation in international HPC conferences, ensuring equitable access to training resources, and establishing partnerships with initiatives that promote gender balance and broader representation in computing. Embedding diversity as a guiding principle in HPC education will not only expand the talent pipeline but also ensure that the next generation of HPC professionals reflects the full spectrum of society.

Conclusion

Implementing these recommendations can help modernize HPC education and ensure that instructional design is informed by current needs. By focusing on core principles, hands-on practice, educator support, inclusivity of different backgrounds, and external collaboration, HPC training programs can produce a larger and more competent workforce. Over time, such efforts will close the gap between the rapid evolution of HPC and the slower pace of curriculum change, ultimately enabling more researchers, engineers, and decision-makers to leverage HPC for innovation.

References

Adams, J. C. (2021). Evolving PDC curriculum and tools: A study in responding to technological change. *Journal of Parallel and Distributed Computing, 157,* 201–219.

Association for Computing Machinery (ACM). (2023). *Computing curricula 2023 (CC2023): Paradigms for global computing education.* ACM. https://doi.org/10.1145/3664191

Bell, T., & Duncan, C. (2018). Teaching computing in primary schools. In S. Sentance, E. Barendsen, & C. Schulte (Eds.), *Computer science education: Perspectives on teaching and learning in school* (Chapter 10, pp. 131–150). Bloomsbury Academic. https://doi.org/10.5040/9781350057142.ch-010

Bohn, C., & Brown, C. (2020). Legion: A K-12 HPC outreach and education cluster. *Practice and Experience in Advanced Research Computing,* 448–451. https://doi.org/10.1145/3311790.3400845

Bouvry, P., Brorsson, M., Canal, R., Eftekhari, A., Höfinger, S., Smets, D., … & Silvano, C. (2025). The European master for HPC curriculum. *Journal of Parallel and Distributed Computing, 201,* 105081. https://doi.org/10.1016/j.jpdc.2025.105081

Brođanac, P., Novak, J., & Boljat, I. (2022). Has the time come to teach parallel programming to secondary school students? *Heliyon, 8*(1), e08662. https://doi.org/10.1016/j.heliyon.2021.e08662

Bücker, H. M., Corrado, J., Fedorin, D., García-Álvarez, D., Gonzalez-Escribano, A., Li, J., Pantoja, M., Pautsch, E., Plesske, M., Ponce, M., Rizzi, S., Saule, E., Schoder, J., Thiruvathukal, G., Van Zon, R., Weber, W., & Bunde, D. P. (2023). Peachy parallel assignments (EduHPC 2023). In *Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis* (pp. 366–373). https://doi.org/10.1145/3624062.3625541

Chalmers, C., Carter, M., Cooper, T., & Nason, R. (2017). Implementing "Big Ideas" to advance the teaching and learning of science, technology, engineering, and mathematics (STEM). *International Journal of Science and Mathematics Education, 15,* 25–43.

Chen, J., Ghafoor, S., & Impagliazzo, J. (2022). Producing competent HPC graduates. *Communications of the ACM, 65*(12), 56–58.

Feng, A., Tilevich, E., & Feng, W. (2015). Block-based programming abstractions for explicit parallel computing. In *2015 IEEE Blocks and Beyond Workshop (Blocks and Beyond)* (pp. 71–75). IEEE. https://doi.org/10.1109/BLOCKS.2015.7369006

Fernandez Slezak, D., Turjanski, P. G., Montaldo, D., & Mocskos, E. E. (2010). Hands-on experience in HPC with secondary school students. *IEEE Transactions on Education, 53*(1), 128–135. https://doi.org/10.1109/TE.2009.2025491

Glaser, B. G., & Strauss, A. L. (1967). The discovery of grounded theory: Strategies for qualitative research. Routledge.

Gschwandtner, P., Hirsch, A., Thoman, P., Zangerl, P., Jordan, H., & Fahringer, T. (2021). The cluster coffer: Teaching HPC on the road. *Journal of Parallel and Distributed Computing, 155,* 50–62. https://doi.org/10.1016/j.jpdc.2021.04.013

Harlen, W., Bell, D., Devés, R., Dyasi, H., Millar, R., Reiss, M., Rowell, P., & Welford, G. (2010). *Principles and Big Ideas of science education.* Association for Science Education.

Harlen, W., Bell, D., Devés, R., Dyasi, H., Millar, R., Reiss, M., Rowell, P., & Welford, G. (2015). *Working with Big Ideas of science education.* Science Education Programme (SEP) of IAP.

Kunkel, J., Filinger, W., Meesters, C., & Gerbes, A. (2020). The HPC certification forum: Toward a globally acknowledged HPC certification. *Computing in Science & Engineering, 22*(4), 110–114. https://doi.org/10.1109/MCSE.2020.2996073

Ludin, M., Weeden, A., Houchins, J., Thompson, S., Peck, C., Babic, I., Muterspaw, K., & Sergienko, E. (2013). LittleFe: The high performance computing education appliance. In *2013 IEEE International Conference on Cluster Computing (CLUSTER)* (pp. 1–1). IEEE. https://doi.org/10.1109/CLUSTER.2013.6702649

Matthews, S. J. (2020). PDCunplugged: A free repository of unplugged parallel distributed computing activities. In *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (pp. 284–291). IEEE. https://doi.org/10.1109/IPDPSW50202.2020.00060

Morris, M., & Frinkle, K. (2014). A three-semester, interdisciplinary approach to parallel programming in a liberal arts university setting. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment* (pp. 1–7). https://doi.org/10.1145/2616498.2616567

Mwasaga, N. M., & Joy, M. (2020). Using high-performance computing artifacts as a learning intervention: A systematic literature review. In *Proceedings of the 2nd International Conference on Intelligent and Innovative Computing Applications* (pp. 1–10). https://doi.org/10.1145/3415088.3415130

Prasad, S. K., Coyle, J., Farley, J., Robey, R., Rosenberg, A. L., Saule, E., & Weems, C. (2011). *NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing: Core topics for undergraduates*. *In Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (pp. 617–618). ACM. https://doi.org/10.1145/1953163.1953336

Prasad, S. K., Estrada, T., Ghafoor, S., Gupta, A., Kant, K., Stunkel, C., Sussman, A., Vaidyanathan, R., Weems, C., Agrawal, K., Barnas, M., Brown, D. W., Bryant, R., Bunde, D. P., Busch, C., Deb, D., Freudenthal, E., Jaja, J., Parashar, M., Phillips, C., Robey, B., Rosenberg, A., Saule, E., & Shen, C. (2020). *NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing – Core topics for undergraduates, version II-beta.* Center for Parallel and Distributed Computing Curriculum Development and Educational Resources *(CDER).* http://tcpp.cs.gsu.edu/curriculum/

Raj, R. K., Romanowski, C. J., Impagliazzo, J., Aly, S. G., Becker, B. A., Chen, J., … & Thota, N. (2020). *High performance computing education: Current challenges and future directions*. In 2020 ITiCSE Working Group Reports (pp. 51–74). Association for Computing Machinery.

Reid, A., Cais, A., Keller, T., Purwanto, W., & Rasel, A. A. (2024). HPC carpentry: A scalable, peer-reviewed training program to democratize HPC access. *The Journal of Computational Science Education, 15*(1), 32–34. https://doi.org/10.22369/issn.2153-4136/15/1/6

Reid, A., Keller, T., O'Cais, A., Rasel, A. A., Purwanto, W., Herriman, J., Muite, B., & Hermanns, M.-A. (2025). HPC carpentry: Recent progress and incubation toward an official carpentries lesson program. *The Journal of Computational Science Education, 16*(1), 31–34. https://doi.org/10.22369/issn.2153-4136/16/1/7

Saule, E., Subramanian, K., & Payton, J. (2021). We need community effort to achieve PDC adoption! In *2021 IEEE 28th International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW)* (pp. 43–49). IEEE. https://doi.org/10.1109/HiPCW54834.2021.00013

Sbaraglia, M. (2023). *Teaching informatics to novices: Big Ideas and the necessity of optimal guidance* [Doctoral dissertation, Alma Mater Studiorum Università di Bologna] Doctorate in Computer Science and Engineering, cycle 35. https://doi.org/10.48676/unibo/amsdottorato/10914

Schwill, A. (1994). Fundamental ideas of computer science. *Bulletin of the European Association for Theoretical Computer Science 53,* 274.

The Council of the European Union, Council Recommendation on a European approach to micro-credentials for lifelong learning and employability, Off. J. Eur. Union C243 (2022) 10–25, https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32022H0627(02)

Valentine, D. (2014). HPC/PDC immunization in the introductory computer science sequence. In *2014 Workshop on Education for High Performance Computing* (pp. 9–14). IEEE. https://doi.org/10.1109/EduHPC.2014.11

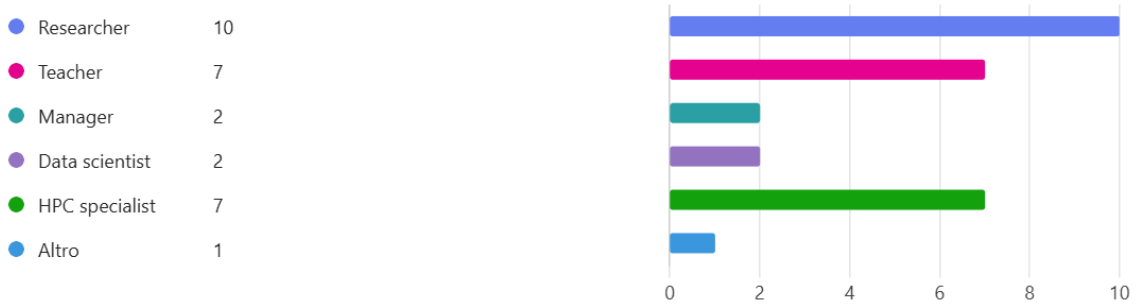
Vargas-Pérez, S. (2022). Designing an independent study to create HPC learning experiences for undergraduates. In *2022 IEEE 29th International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW)* (pp. 6–11). IEEE. https://doi.org/10.1109/HiPCW57629.2022.00006

Annex – Questionnaires' analysis and results

Demographic and professional profiles of the respondents

Roles in the field of HPC

3. What is your role in the field of HPC?



The responses to the question "What is your role in the field of HPC?" highlight the diversity of professional roles that contribute to the High Performance Computing ecosystem. A majority of respondents identify as *Researchers* (appearing in 9 responses), suggesting a strong focus on the theoretical and experimental aspects of HPC, including algorithm development, performance optimization, and computational modelling. This is followed closely by those identifying as *HPC Specialists* (appearing in 9 responses), which highlights the importance of hands-on technical expertise in managing and maintaining HPC systems, as well as optimizing their use for various applications.

The presence of roles such as *Teacher* (appearing in 6 responses) and *Manager* (appearing in 2 responses) suggests that educational and managerial aspects of HPC are also well-represented, which is critical for the dissemination of HPC knowledge and the strategic planning and coordination of HPC initiatives. Notably, the combination of *Researcher* and *Teacher* (5 responses) underscores a blend of academic and pedagogical responsibilities, indicating that many respondents play a dual role in both conducting research and educating the next generation of HPC professionals.

Additional roles such as *Data Scientist* (appearing in 2 responses) and *Education and Training Officer* (appearing in 1 response) reflect the growing intersection between HPC and other fields like data science, where HPC is crucial for processing large datasets, and the dedicated focus on training and skill development within the HPC community.

Years of experience in HPC

4. How many years of experience do you have in the field of HPC?

The responses to the question "How many years of experience do you have in the field of HPC?" show a diverse range of experience levels within the survey participants, providing valuable insight into the expertise of the HPC community represented.

A significant proportion of respondents (7 out of 13) report having 10+ years of experience in the field, highlighting a strong presence of highly experienced professionals. This is a mature group of individuals with deep expertise in HPC, likely possessing comprehensive knowledge about both its historical evolution and current state. Their extensive experience could be invaluable in identifying "Big Ideas" in the field, especially in understanding foundational concepts and the long-term evolution of the discipline.

On the other hand, the responses also reflect a notable presence of professionals with 5-10 years of experience (3 out of 13). These individuals are likely familiar with the modern advancements in HPC technologies, tools, and methodologies.

A smaller portion of respondents (3 out of 13) report having less than 5 years of experience. While this may indicate a relatively newer involvement in the field, it brings a fresh perspective to the survey, possibly emphasizing the emerging trends in HPC and how new professionals perceive and engage with the field's fundamental concepts.

HPC teaching experience

5. Do you have teaching experience in the field of HPC (at the university level or in professional training)?

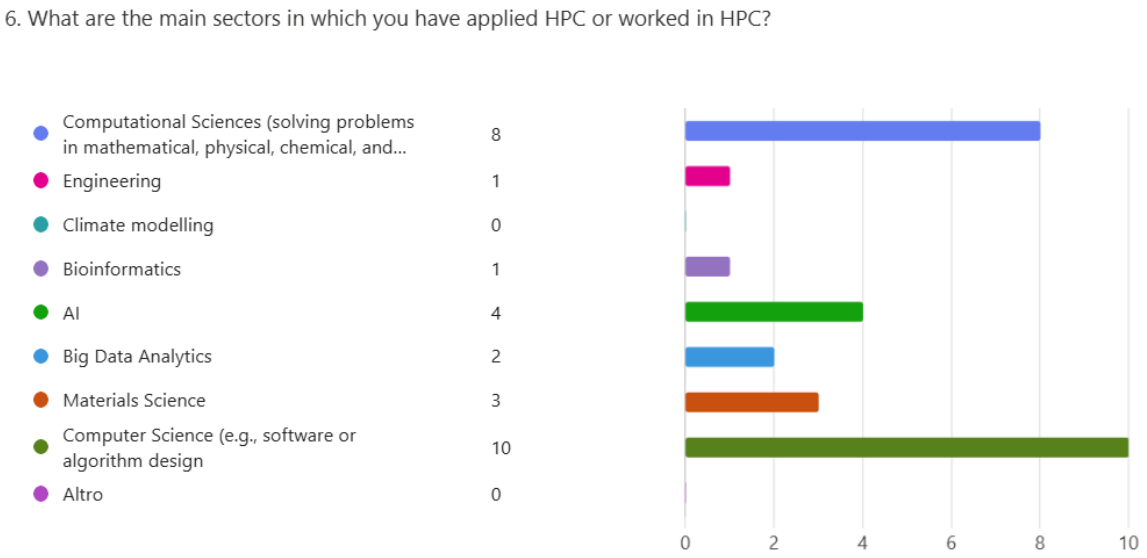
The responses suggest a highly engaged group of respondents with varying degrees of teaching involvement, providing a comprehensive range of perspectives on the educational implications of HPC. A majority of respondents (8 out of 13) indicate that they have continuous teaching experience. These respondents likely play an integral role in shaping the next generation of HPC professionals, both in academic settings and through professional development programs.

A smaller group of respondents (4 out of 13) report occasional teaching experience in HPC, which may indicate that teaching is not their primary role but that they contribute to education on a less frequent basis, potentially through guest lectures, workshops, or short courses. This group brings valuable insights as well, especially with their expertise in the field and more sporadic yet still meaningful involvement in education.

Interestingly, two respondents reported having no teaching experience in HPC, which could indicate that their work in HPC is purely focused on technical or research aspects. While they may not be directly involved in teaching, their responses may offer a unique perspective on the identification of "Big Ideas" from a more practical, hands-on standpoint.

HPC tools and applications

The analysis of the responses to the questions regarding the sectors in which participants have applied or worked with HPC and the technologies and tools they most frequently use in their daily work provides a complex and fascinating picture of the daily practices and application areas of high-performance computing. The sectors in which the respondents have applied or worked with HPC are numerous and varied, reflecting the importance of HPC across a wide range of academic and industrial fields.

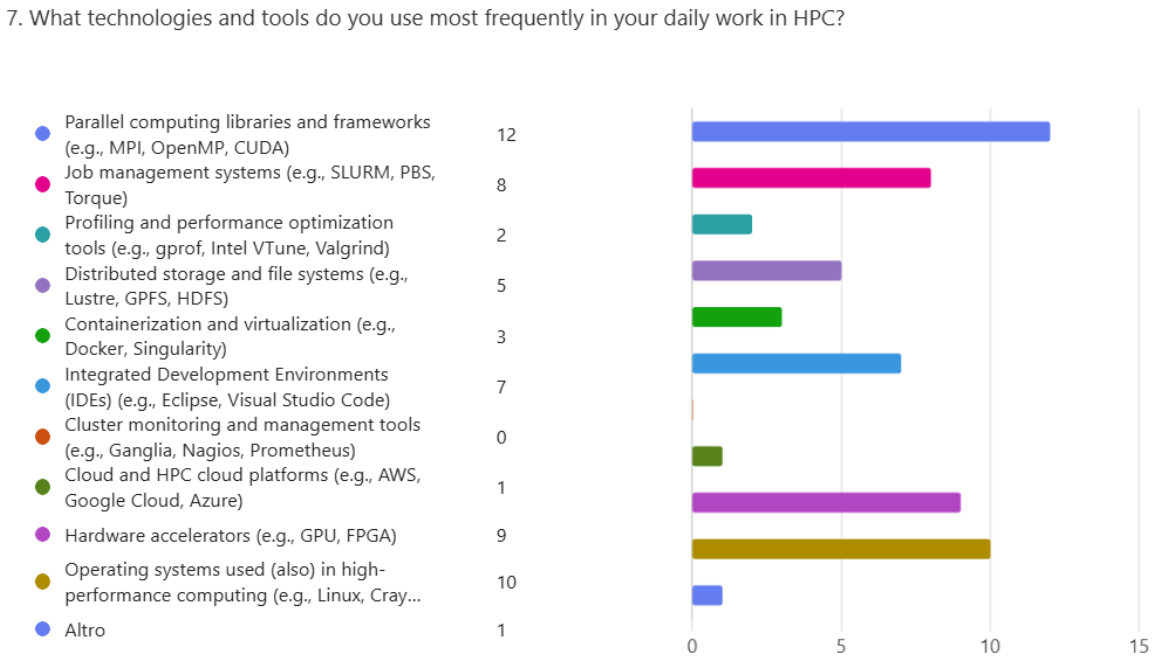


The majority of respondents indicated working in Computational Sciences, particularly in solving mathematical, physical, chemical, and natural science problems using HPC. This reflects a fundamental area where complex simulations and data processing require access to advanced computational resources. Computational Sciences represent a core part of the HPC field, and their impact extends far beyond academia, also influencing industrial research and technological advancement across multiple sectors.

Additionally, the significance of Computer Science is evident, with many respondents mentioning applications related to software and algorithm design. HPC is essential not only for developing new algorithms but also for optimizing existing ones and tackling complex computational problems, such as those arising in artificial intelligence and machine learning. In particular, the mention of AI among the sectors indicates the

growing integration of HPC in AI technologies, which require massive computational power for training complex models.

Other important sectors include Materials Science, Bioinformatics, and Big Data Analytics, which highlight how HPC is used in interdisciplinary fields to explore new frontiers in materials research, biomedical sciences, and the management of vast datasets.



Regarding the technologies and tools most used in their daily work, there is a strong emphasis on parallel computing libraries and frameworks such as MPI, OpenMP, and CUDA—fundamental tools for leveraging the power of modern multi-core systems and GPUs. These technologies are crucial for optimizing performance and ensuring that computations are carried out efficiently on high-performance hardware.

Job management systems (such as SLURM, PBS, Torque) and operating systems (particularly Linux and Cray OS) are also widely used, highlighting the need to manage and coordinate large-scale computing resources. Managing workloads and resources is a vital part of the HPC ecosystem, as it allows for the optimal use of distributed and parallel computing infrastructures.

The use of hardware accelerators such as GPUs and FPGAs is equally important, reflecting the growing role of these tools in accelerating complex computational tasks,

such as training AI models and processing large datasets. Containerization and virtualization technologies (like Docker and Singularity) are emerging as essential solutions to ensure the portability and scalability of HPC applications, especially in cloud or distributed environments.

Furthermore, tools for performance optimization and profiling (such as gprof, Intel VTune, Valgrind) are crucial for analyzing and improving the efficiency of HPC applications, ensuring that computational resources are used as effectively as possible.

Big Ideas identified

The open-ended questions on Big Ideas' identification were analysed following a methodology of thematic analysis (Braun & Clarke, 2006). This allowed us to identify recurring themes that the Big Ideas addressed. We report them below, with some examples from respondents' answers.

Theme	Example of respondents' Big Ideas referring to the theme
Parallelism (algorithms, data structures)	Parallelization is the foundation of HPC. This extends across different scales of magnitude — from microprocessors, to the nodes of a system, to multiple systems. Parallelization is fundamentally a problem of orchestrating multiple agents. (R8) At the core of HPC are parallel algorithms and data structures. (R12)
Problem decomposition	For 'Big Data' problems, divide-and-conquer is the most effective strategy. (R1) Many complex problems can be solved by breaking them down into simpler problems that can be solved independently. (R13)

Parallel design patterns	When writing HPC software, parallel design patterns can save you from having to 're-invent the wheel'. (R1)
Technological innovation and hardware/software advancements	The challenges in designing HPC systems lie in physical constraints: e.g. 1: The problem of designing chips that are both powerful and compact e.g. 2: Achieving the lowest possible energy consumption (R8)
Sophistication of parallel algorithms	Parallel algorithms are much more complex and powerful than their corresponding serial algorithms. (R6)
Increased performance of HPC with respect to traditional models	Distributed computing can be used to improve performance, or increase reliability, or both. (R1) Parallel computing is the computational equivalent of the old adage "many hands make light work." (R1)
Limitations and challenges of parallelism (hardware and software)	Communication overhead can reduce speedup. (R11) HPC is not a panacea. Adapting a problem to HPC-based solutions involves trade-offs, which can come as a surprise to new HPC users. For example, time-to-solution may depend strongly on your strategy for making the task parallel, and for some tasks, may not offer any benefit at all. (R2)
Heterogeneous environment and combination of different models	Various types of separate HPC methods can be combined in clever ways. An example would be multicore+GPU, or multicore+SIMD, or MPI+multicore+GPU. (R3)
HPC + AI	AI/ML is an application of HPC (R10)

	The integration of HPC and machine learning enables real-time simulations of complex systems, using surrogate models to reduce computational costs without compromising accuracy. (R7)
Scalability and distributed systems	In some cases it is possible to replace a powerful but expensive processor with a large number of less powerful but much more affordable processors. (R13)
Storage and data management	Disaggregating devices puts more stress on the network. High performance, secure. low latency data transfer is needed to support increasingly distributed workloads. (R4) A file can be split and its fragments stored across multiple servers, while still looking like a common single file to the user. (R5)
Practical applications of HPC	The integration of HPC and machine learning enables real-time simulations of complex systems. (R7)
Societal implications of HPC	There are computationally complex problems that require many calculations. Personal computers are insufficient to solve these problems. Society benefits from striving to push the boundaries in solving these problems as effectively as possible. The frontier is HPC (R8)

The themes identified can be clustered into 5 macro-themes as follows:

Cluster	Included Themes
Foundations of Parallel Computing	• Parallelism (algorithms, data structures) • Problem decomposition • Parallel design patterns • Sophistication of parallel algorithms • Limitations and challenges of parallelism (hardware and software)
Technological Infrastructure and Innovation	• Technological innovation and hardware/software advancements • Heterogeneous environment and combination of different models • Scalability and distributed systems • Storage and data management
Synergies with Artificial Intelligence	• HPC + AI
Applications and Societal Impact	• Practical applications of HPC • Societal implications of HPC
Performance and Strategic Advantage	• Increased performance of HPC with respect to traditional models

Educational needs emerged

To analyse the educational needs to train experts in academia and in the industry, we started from the themes highlighted for the Big Ideas' classification. Then, other themes were added to reach a comprehensive description of respondents' answers.

	Theme	Academia	Industry
From Big Ideas	Parallelism (algorithms, data structures)	✓	✓
New	Concurrency	✓	×
From Big Ideas	Problem decomposition	×	×
New	Data decomposition	✓	×
From Big Ideas	Parallel design patterns	✓	✓
New	Teaching on real-world datasets	✓	×
From Big Ideas	Technological innovation and hardware/software advancements	×	✓
From Big Ideas	Sophistication of parallel algorithms	×	×
From Big Ideas	Increased performance of HPC with respect to traditional models	✓	✓
From Big Ideas	Limitations and challenges of parallelism (hardware and software)	✓	✓
New	GPU	✓	✓

From Big Ideas	Heterogeneous environment and combination of different models	×	×
From Big Ideas	HPC + AI	✓	✓
From Big Ideas	Scalability and distributed systems	×	×
From Big Ideas	Storage and data management	×	×
From Big Ideas	Practical applications of HPC	×	×
From Big Ideas	Societal implications of HPC	✓	✓
New	Focus on educational research for HPC teachers/trainers	✓	×
New	Computer Science background for non CS majors	✓	×
New	Verification and reproducibility of results	✓	×
New	Open standards	×	✓
New	Knowledge transfer among employees	×	✓
New	Access to and use of public supercomputer	×	✓

The experts mention as fundamental themes for training both in academia and industry:

- Parallelism (algorithms, data structures)
- Parallel design patterns
- Increased performance of HPC with respect to traditional models
- Limitations and challenges of parallelism (hardware and software)
- GPU
- HPC + AI
- Societal implications of HPC

Other themes that are recognized important for training in academia are:

- Concurrency
- Data decomposition
- Teaching on real-world datasets
- Focus on educational research for HPC teachers/trainers
- Computer Science background for non CS majors
- Verification and reproducibility of results

Other themes that are recognized important for training in industry are:

- Technological innovation and hardware/software advancements
- Open standards
- Knowledge transfer among employees
- Access to and use of public supercomputer

Your vision is important

Contribute
and be part
of our research