Cost-Effective Clusters in Education: A Model for Practical HPC Learning

Tiago A O Demay[1], Lícia S. C. Lima[2] Michel S. Fornaciali [3], André Filipe M. Batista[4], Rafael C. Ferrão[5]
dept. of Computer Engineering.[1,2,4,5]
Emails: (liciascl[1], tiagoaodc[2], michelsf[3], andrefmb[4], rafael.corsi[5])@insper.edu.br
Insper, São Paulo, Brazil.

***Abstract*—This innovative practice paper details a teaching experiment utilizing a cluster that simulates a real HPC environment. Insper, a new higher education institution in the engineering field, aims to provide practical and interactive education. To this end, a course was implemented for 40 seventh-semester Computer Engineering students. The course explored the use of a low-cost cluster inspired by supercomputers from the TOP500 list, including Frontier in the United States, Fugaku in Japan, and Santos Dumont in Brazil.**

This research focuses on enhancing the HPC module within the Computer Engineering program by integrating practical HPC experiences using a cluster and assessing its educational impact through both quantitative and qualitative measures. Data were collected from system logs and a structured questionnaire administered to 40 participating students. Analysis of the logs provided insights into job submissions, resource utilization, and common errors encountered during cluster operations. The questionnaire assessed students' understanding of HPC concepts, their practical skills, and overall satisfaction with the course enhancements.

Results indicated a significant improvement in the students' practical understanding of HPC concepts, as evidenced by the successful completion of 177 jobs and enhanced student engagement with the practical components of the course. However, challenges such as resource allocation errors and variations in system performance highlighted the need for ongoing improvements in both educational strategies and cluster infrastructure.

The study concludes that integrating a hands-on HPC cluster within the curriculum significantly enhances the learning experience by making complex concepts such as parallelism and resource distribution more tangible. Future recommendations include focusing on reducing system cluster errors, improving resource estimation education, and standardizing the system to ensure consistent student experiences. This approach not only facilitates the application of theoretical knowledge but also prepares students for real-world computational challenges.

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

HPC is a crucial tool for conducting advanced research across interdisciplinary fields such as artificial intelligence, data science, atmospheric sciences, and biology. This technology enables the execution of complex calculations at high speeds, facilitating the use of sophisticated algorithms necessary for computational simulations and modeling large datasets. These capabilities are essential for predicting behaviors and trends [1]. Moreover, HPC is vital for running complex climate prediction models, playing a key role in enhancing responses to natural disasters and planning environmental actions [2]. In bioinformatics, HPC supports genome sequencing and the modeling of biological complexities, advancing personalized medicine and understanding of complex diseases [3]. Thus, HPC proves to be an indispensable tool for interdisciplinary research, not only speeding up intensive analyses but also fostering crucial scientific and technological innovations.

To ensure that graduates possess essential skills in parallel and distributed computing, the NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing [4] aims to provide comprehensive curricular guidelines to ensure that graduates in computer science and computer engineering possess fundamental skills in parallel and distributed computing. The latest version of the initiative, released in November 2020, features significant revisions with an enhanced focus on Big Data, energy efficiency, and distributed computing. The purpose of this session is to engage with the SIGCSE community (Special Interest Group on Computer Science Education) to gather feedback on the curriculum and discuss related activities from the Center for Parallel and Distributed Computing Curriculum Development and Educational Resources that support the implementation of this curriculum.

Additionally, the initiative has been promoting workshops, such as EduPar-20, to develop and evaluate educational innovations and curricular resources in HPC. These workshops bring together individuals from academia, industry, and various educational and research institutions to explore new ideas, challenges, and experiences related to HPC teaching and curricula [4]. The importance of integrating HPC competencies into computer science curricula is further emphasized by [4], who highlights the importance of integrating HPC competencies into computer science curricula to prepare students for modern computational demands. The essential HPC skills that should be developed include understanding parallel and distributed architectures, multithreaded programming, and distributed memory management. Additionally, it is crucial for students to develop skills in parallel and distributed algorithms, performance optimization, and scalable computing. These competencies enable graduates to improve computational efficiency and effectiveness while preparing them to innovate in data-intensive and complex calculation fields.

The integration of HPC into computing education presents significant challenges that demand more focused efforts and resources. [1] points out that a major hurdle is incorporating HPC into existing courses, a task made even more difficult by

the lack of HPC skills among educators and limited access to advanced computing facilities. This gap in educational practices and infrastructure significantly limits practical learning experiences, which are crucial for mastering the complex computing tasks that are vital in today's scientific and industrial fields.

Aligning with the ACM/IEEE curriculum recommendations, as discussed by [5], it is evident that integrating courses focused on Parallel Computation into undergraduate programs is increasingly important. These guidelines ensure that HPC courses offer students up-to-date and applicable education, preparing them effectively for careers in HPC. [5] also underscores the benefits of active learning through the use of multicore clusters, which provide students with hands-on experience. This practical engagement, allows students to apply theoretical knowledge to real-world challenges, thereby enhancing their technical skills and critical thinking abilities.

Several countries have recognized the strategic importance of HPC and have launched initiatives to bolster their leadership in this area. For instance, the United States has introduced the National Strategic Computing Initiative (NSCI) to advance American leadership in HPC. This initiative aims to develop exascale computing systems, enhance coherence among technology bases used for modeling and simulation, and establish a viable path for future HPC systems. Supported by public-private collaborations, this strategy focuses on maximizing the benefits of HPC for the United States, with a strong emphasis on workforce development and education to support these technologies [6].

Similarly, the European Union's HPC strategy emphasizes the development of next-generation HPC technologies, applications, and systems, with a clear trajectory towards exascale computing. The EU strategy also aims to facilitate access to top-tier supercomputing facilities and services for both industry and academia, while promoting excellence in the delivery and use of HPC applications through the establishment of Centers of Excellence. This approach is complemented by efforts to raise awareness and develop skills in HPC [7].

Despite the clear advantages of integrating HPC into education, significant challenges remain. According to [5], several major challenges were encountered in the implementation of a Cluster in the HPC course:

1) **Curriculum Adaptation**: Integrating parallel and distributed computing content into existing curricula proved challenging, particularly in balancing new parallelism modules with established course structures.
2) **Technical Complexity**: The advanced technical nature of topics such as the Message Passing Interface (MPI) and multicore cluster computing was difficult for students, who found these topics less appealing compared to more accessible topics like MapReduce.
3) **Hardware and Software Resources**: The need for adequate infrastructure, such as multicore clusters and access to cloud computing platforms like AWS, was a significant barrier. Limited access to advanced computing resources outside cloud environments impacted students' ability to directly experiment with HPC technologies.
4) **Student Engagement and Motivation**: Maintaining student engagement with technically demanding topics, like MPI computing, was difficult. Despite successful learning outcomes, students showed less interest in this topic, which could affect their depth of learning and practical application of the concepts.
5) **Software Development**: Using open-source software tools and platforms like Hadoop MapReduce, while cost-effective, presented challenges in setup, maintenance, and updates, along with a learning curve associated with using these less intuitive tools compared to commercial solutions.

These challenges reflect the typical difficulties encountered when integrating HPC technologies into educational settings. Addressing them requires significant resources and a carefully planned approach to curriculum development and student support.

II. Context

Insper is a private higher education institution located near the economic center of São Paulo-SP, Brazil. In the Computer Engineering program, the HPC module is introduced in the sixth semester out of 10. By this stage, students have already been trained in networks, hardware, and software, and have learned task management techniques. Additionally, they have developed a foundation in programming languages and operating systems. The structure of the Computer Engineering course is designed to be sequential, ensuring that prerequisites for more advanced courses are covered in earlier modules.

The course is structured into four meetings per week, with each session lasting two hours. The class is split into groups of 20 students per session, resulting in eight sessions per month for each group. Throughout the course, each student group attends a total of 40 sessions. The team overseeing the course consists of one instructor and two laboratory technicians. These technicians manage the cluster operations and provide hands-on support during the lab activities.

The course curriculum offered an introduction to HPC, beginning with its foundational and historical aspects, and exploring its current global relevance. Students were immersed in programming with C++, which served as a gateway to optimization, heuristics, and practical problem-solving techniques. Profiling methods were introduced to enhance code efficiency and performance. The curriculum extended into algorithmic strategies, focusing on heuristic algorithms for practical problem-solving, covering techniques such as greedy algorithms and metaheuristics. It included an exploration of randomization in algorithm design, showcasing randomized algorithms and Monte Carlo simulations.

As the course progressed, students learned to benchmark and assess algorithm performance, analyzing runtime, memory usage, and scalability. Parallel computing principles were introduced through modules on parallelism and GPU computing, where students learned to manage synchronization,

side effects, and load balancing in GPU programming. The curriculum integrated distributed computing concepts with a focus on MPI and OpenMP. Practical exercises in these areas were facilitated using Google Colab and local machines, allowing students to apply their learning in real-world computing environments.

During the course, it was observed that students faced challenges in grasping the abstract concepts of resource distribution and allocation within the clusters configured on their local machines. This issue was compounded by the varying performance experiences among students, which were dependent on the individual capabilities of their personal computers. Additionally, the use of local clusters and Google Colab presented limitations in terms of available resources, further impacting the uniformity and effectiveness of the learning experience. These disparities highlighted the need for a more standardized approach to ensure that all students have equal access to the necessary computational resources and a consistent learning environment.

To address these challenges, several approaches were tested. The first approach, as discussed [8], involved the application of a low-cost cluster, referred to as UpCluster [9] which comprised 24 Intel micro Upboards. The integration of the UpCluster into the course structure provided a practical platform for students to experiment with and optimize computational models and algorithms, thus enhancing their understanding and skills in HPC system design and management.

This solution enhanced the students' experience and reduced the abstraction of parallelism and distribution concepts, thanks to the visual feedback provided by the LED interface of the UpCluster. This interface effectively demonstrated where each job was allocated within the cluster machines. However, as the cluster was constructed with limited hardware capabilities, it was not possible to facilitate the resolution of computationally complex problems due to the cluster's limited computational power. This limitation necessitated a transition to the second phase of our approach.

The system architecture designed for the course equips each student with an Intel i5 NUC[1], serving as the primary hardware platform with 16GB of RAM and a 500GB hard drive for setting up individual virtual environments. VirtualBox is utilized to host virtual machines (VMs) on these NUCs, enabling the creation and management of multiple isolated VM environments on a single physical machine. One of these VMs functions as the System Management Server (SMS), which is responsible for managing the cluster, including task distribution and resource management. Additional VMs act as compute nodes where actual data processing and task executions occur, managed by the SMS to demonstrate principles of parallelism and task distribution.

To streamline the setup and provisioning of these VMs, Vagrant is used, ensuring consistency and ease of deployment across the NUCs utilized by students. The system has a Vagrant script to facilitate the installation of necessary operating systems and applications, which may include tools like ClusterShell, Nagios, Slurm, Ganglia, OpenHPC, and Node Health Check (NHC). These HPC tools are integrated to create a realistic computing environment, with ClusterShell for executing commands across the cluster, Nagios for system monitoring, Slurm for job scheduling, Ganglia for performance monitoring, OpenHPC for HPC environment management, and NHC for node health monitoring.

The diagram in Figure 1 illustrates how these components interact, forming a cohesive system for educational purposes to help students understand cluster management and distributed computing architectures.

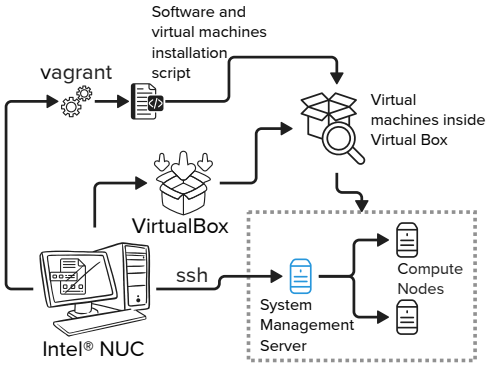

Fig. 1. Diagram of the virtual cluster architecture, created by the vagrant tool through a script, showing the configuration of login and compute nodes, the authentication flow and task submission by students, and network segregation to optimize practical learning in HPC.

Stimulated by the achievements made and recognizing an improvement in the student's learning experience, we reached the final step where we utilized five Intel i7 computers, each equipped with 64GB of RAM. One computer functioned as the login node, while the others served as compute nodes, as depicted in the setup diagram. This system was designed by modeling it on commercial clusters like Frontier or Santos Dumont, providing a realistic framework for the students to engage with advanced computing environments.

III. The HPC Cluster

This new approach was developed to address the issues mentioned in the previous section and was based on a practical HPC scenario, featuring essential components of an HPC architecture. In a real-world scenario, these include computing hardware, storage systems, and high-speed services, network interconnections such as InfiniBand or Ethernet. Additionally, the setup is equipped with software and middleware optimized for HPC, encompassing operating systems, communication libraries like MPI, parallel programming tools, and task schedulers.

To assemble a cluster for engineering students to study HPC concepts, several options are considered based on budget and specific needs:

[1]An Intel i5 NUC (Next Unit of Computing) is a small form factor personal computer developed by Intel.

Raspberry Pi clusters: These are economical but offer limited hardware resources, suitable for basic educational purposes.

NVIDIA Jetson-based clusters: Ideal for GPU-intensive applications such as machine learning and image processing. However, the support for NVIDIA GPU drivers in ARM environments, like Jetson Nano, may be limited compared to x86 platforms, which are more common in traditional HPC setups.

Cloud services: Amazon AWS, Microsoft Azure, and Google Cloud also offer scalable HPC infrastructures. While these platforms eliminate the need for physical hardware, they restrict direct access to hardware configurations and system-level optimizations. This limitation can hinder students' ability to learn about specific hardware configurations and optimizations. Additionally, managing and configuring cloud environments is complex and requires specialized knowledge, which could shift the focus away from the core HPC principles.

The new approach: This was the choice of the team: to develop a cluster using off-the-shelf machines, but with sufficient computational power to support classes of 40 students. Additionally, we ensured the capability to install and configure the necessary software to create an environment as close to the real world as possible.

In the development of the cluster architecture, a thorough analysis of software and hardware components was conducted to achieve a balance between cost and performance. A research study on the architecture of supercomputers from the TOP500 list [10] was conducted, including Frontier in the United States, Fugaku in Japan, and Santos Dumont in Brazil, particularly with regard to the software used and methods of accessing the cluster.

Slurm was chosen as the central manager of the cluster due to its robustness and flexibility. As noted by the TOP500 Supercomputer Sites [10], Slurm is run on the six largest HPC systems in the world and is used in 42% of the top 100 HPC systems globally. This supports an architecture that closely mirrors real-world scenarios. Its open-source platform helps us with scalability and fault tolerance, supporting adaptive growth and ongoing maintenance. This allows us to adapt to the demands of new students and evolving computational technologies.

The network configuration was established with Ethernet connections between commercial routers and switches. The system connects to five Dell machines, each equipped with:

- Two 512GB PCIe NVMe M.2 SSDs
- 13th Generation Intel® Core™ i7-13700T processors
- 64GB of DDR5 RAM at 6400MT/s

The storage system employs the Network File System (NFS), which is a distributed file system protocol allowing work nodes to access files on a network as if accessing local storage. All interactions with cluster resources are conducted via Secure Shell (SSH). Access to the cluster is secured using a pair of public and private keys, offering a more secure method than traditional passwords and allowing more precise access controls, in addition to the easier revocation of keys without needing to change passwords at multiple locations.

Figure 2 illustrates the interaction flow of students with the HPC cluster during the course. Initially, students must connect to institution's Wi-Fi network, ensuring a secure and exclusive environment for academic access. Subsequently, they are directed to the login node through specific routing rules configured by the technical team, ensuring that only authorized traffic reaches the cluster. Access to this node is restricted and requires authentication through public keys, previously configured in the login node files by a technical team. This procedure ensures that only authorized students can proceed. The creation of users is carried out manually by the team of technicians, requiring access to an updated list of students, user management skills in the cluster system, and profound knowledge of the installed software.

Although this method provides control, it also highlights an opportunity for optimization through automation. Once authenticated, students access the login node, where they submit and execute tasks, ranging from simulations to intensive data analyses.

This process is managed by control daemons, which use scheduling algorithms to distribute tasks across appropriate computing nodes. Students interact with the system primarily through commands on a Slurm command-line interface, allowing manipulation and control over their submissions. The database storing all this information is located on the same machine, which may not be ideal for security. Each student's user has standardized permissions that do not include prerogatives to install or modify the operating system of the login node, they do not have write access to directories outside their workspace.

Halfway through the semester, the need for more security and isolation of students' personal files was noted. Therefore, in the next version, we will implement the "jail chroot" technique to enhance security, and we will also modify user permissions. This approach isolates each user's workspace, restricting file access exclusively to the designated directory tree, thereby preventing any unauthorized access to files outside this environment. Additionally, we will replace the NFS file sharing system with the more advanced Lustre software, aiming to improve data handling efficiency. This update increases data throughput, facilitating direct interactions between metadata servers and storage components, ensuring superior cluster performance, and bringing it closer to a real-world scenario.

The internal network infrastructure of the cluster is established through software configured on the login node, which manages an internal network interface and effectively segregates the compute nodes from any external networks, including both institution and the Internet. This isolation is instrumental in enhancing the operational efficiency of HPC applications and in safeguarding the system's integrity and availability.

Within this infrastructure, computing resources are partitioned into specific nodes: four CPU nodes, each equipped

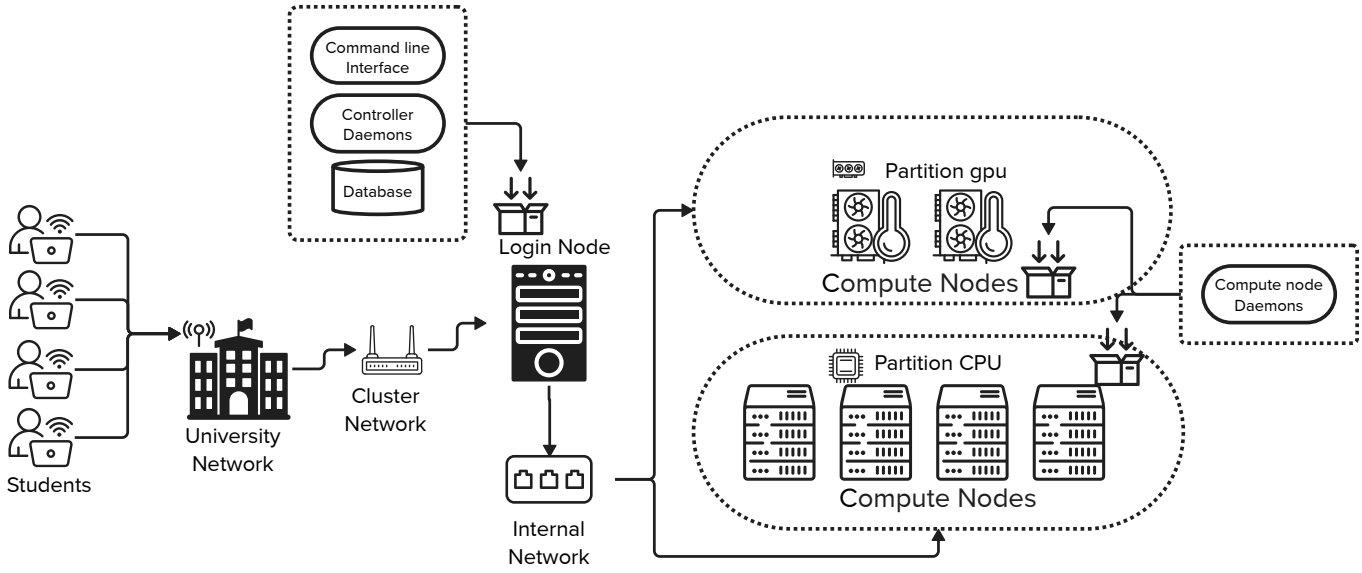

Fig. 2. Diagram of the network usage and architecture, showing the interaction flow of students with the cluster, task management through the login node, the distribution of computing resources across CPU and GPU nodes, and the allocation of tools and database on the login node, as well as daemons on the compute nodes.

with 24 threads and 64 GB of RAM, and two GPU nodes, each housing an NVIDIA 1080 Ti card, 16 GB of RAM, and 8 threads. This structure not only restricts the resources available per student request, thereby optimizing resource utilization, but also introduces dynamic elements into the educational activities, enhancing the students' acquisition of practical skills in HPC.

This setup not only facilitates students' understanding of real-time distributed systems but also enables the execution of a variety of educational projects and experiments.

IV. Methodology

The data collection process from the first semester of 2024 within the Computer Engineering course at Insper is outlined. The course design is sequential, building on prior coursework that equips students with knowledge in hardware and operating systems. The class under study comprised around 40 students in their seventh semester.

Data were collected from the *slurmctld.log* and *slurm_jobcomp.log* files of the cluster. The logs, totaling 4953 lines, were collected two weeks after the midterm exams, covering the period from the beginning of the semester up to the midterms.

The analysis covered various aspects of cluster usage within the course, such as the number of jobs submitted, peak utilization periods, individual student usage, resources requested by student jobs, and the outcomes of these jobs—categorized as successful, failed, or canceled by students.

Additionally, reasons for job failures and the status of the job queues were analyzed. This data collection allowed for a detailed analysis of the actual demand on the cluster, assessing whether a hardware upgrade was necessary to support class dynamics.

It also evaluated student engagement with the platform, identifying if there were spikes in access during critical times such as project submissions or exam periods. A significant focus was placed on identifying common difficulties leading to job failures, providing valuable insights into how the course could be improved.

To document student perceptions and conduct a qualitative analysis of the tool's implementation, a questionnaire was applied for the 40 participants. This survey was designed to evaluate various aspects of student learning and engagement with the HPC cluster.

It encompassed sections devoted to the theoretical foundations of the course, assessing students' comprehension of critical HPC concepts. Additionally, the questionnaire examined practical applications, testing how effectively students could apply their knowledge in real-world scenarios involving the cluster. Questions probed students' comfort levels with the HPC environment, focusing on their confidence in using specific cluster commands and system navigation. To enrich the analysis, the survey included two open-ended questions, allowing students to provide spontaneous feedback.

Additionally, we mapped the students' learning levels with two questions directly related to the cluster in the midterm exam, where their knowledge was assessed across three dimensions: the use of examples and references to answer questions, understanding of the concept, and the depth and detail of their responses. This approach provided a comprehensive evaluation of how well students grasped and applied the concepts associated with cluster usage.

After detailing the methodology employed in this study, including the design of the cluster used and data collection strategies through system logs and surveys, we now move to

the analysis of the results obtained. This section aims not only to present the performance metrics of the system and the effectiveness of the educational interventions but also to reflect on how the practical experiences influenced student learning. The information gathered is crucial for understanding the students' interaction with the HPC infrastructure and the implications of these interactions for their professional development. Therefore, the subsequent analysis seeks to directly correlate the methodological activities with the observed performances and perceptions, providing a holistic assessment of the educational impact of the HPC cluster.

V. Presentation of Results

The analysis of log data provides a comprehensive assessment of job performance and resource utilization among student submissions within the cluster. Detailed examination of the *JobState* metrics reveals a varied distribution of outcomes, 177 jobs successfully reached completion, indicating that a significant portion of the student submissions were able to utilize the cluster resources effectively and achieve their intended results.

However, 67 jobs did not complete successfully, indicating potential issues in job configuration. Additionally, 58 jobs were canceled by the users, likely due to errors discovered during execution. Moreover, 28 jobs exceeded the predefined execution time limits, highlighting the need for improved job time estimation and more stringent management of resource allocation to prevent excessive usage.

These findings suggest a mixed level of proficiency among students in managing and executing jobs on the cluster, with a clear indication that further training and support are required to enhance their competency in HPC environments.

Moreover, analysis of memory allocation for these jobs displayed substantial variation. A predominant number of jobs allocated 14 GB of memory, indicating a standard resource requirement for the majority. Conversely, several jobs utilized markedly lower amounts (1 GB or 4 GB), and one exceptional case required 28 GB, suggesting a job with particularly high computational demands.

The duration of the jobs provides additional insights into the performance and utilization patterns of the cluster. Analyzing a total of 325 jobs, it was found that the average execution time was approximately 1.02 minutes. However, the significant variability in job duration, with a standard deviation of 10.07 minutes, suggests a wide range of computational requirements and efficiencies among the submitted jobs. This variability likely indicates that the shorter jobs are tests and code prototypes, while the longer jobs are the proposed algorithms being effectively executed by the students.

The variation in job duration suggests that students are testing independent blocks of code to validate their functionality before running the complete code, as advised. By dividing the problem into smaller parts, students can more easily identify and correct errors. This approach not only aligns with the instructional guidance provided but also demonstrates an effective strategy for managing and troubleshooting complex computational tasks.

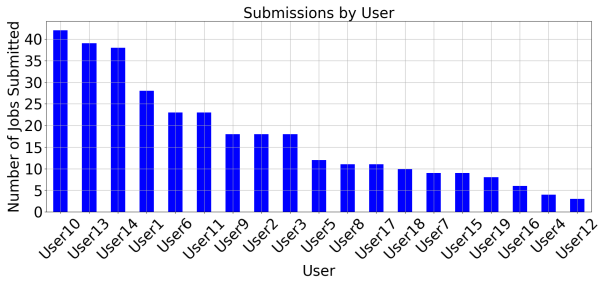

Fig. 3. This visualization underscores the variability in usage frequency among users, highlighting patterns that may correlate with user engagement and project demands.

Upon further analysis of student engagement with the cluster environment, we observed significant variations in usage frequency among participants. As demonstrated in Figure 3, the distribution of access frequency to the cluster reflects distinct patterns of engagement, which may be related to both the specific demands of the students' projects and their proactivity and comfort with HPC technology. This variability is critical for understanding how different students interact with the available resources and how this can influence the success of their academic projects.

The Figure 4 illustrates the distribution and prevalence of different types of errors that occurred in the educational cluster environment, helping to visualize which problems are most common among users. The analysis of error types reveals important insights into the challenges faced by students and can guide future improvements in instructional design.

The predominance of Resource Errors suggests that students may not be fully familiar with request resources in HPC environments or may be underestimating the demands of their tasks. This could be attributed to an insufficient understanding of HPC concepts at the initial stages of the course or a lack of practical experience before undertaking independent work on the cluster. Consequently, students may make inappropriate computational resource configurations, indicating a basic understanding of how to allocate resources.

Configuration Errors (Blue): Initially common, these errors, which represent about 11% of total errors (100 errors), suggest students' unfamiliarity with cluster configuration settings. Over time, the frequency of these errors decreased significantly, indicating improved user competence, likely due to effective initial orientation and training.

System Errors (Orange): Representing approximately 17% of total errors (150 errors), system errors occurred at a consistently low rate throughout the period. This consistency points to stable cluster operations and effective management of the cluster infrastructure, with prompt resolution of any issues.

Resource Errors (Green): The most prevalent, comprising about 61% of total errors (550 errors), resource errors showed

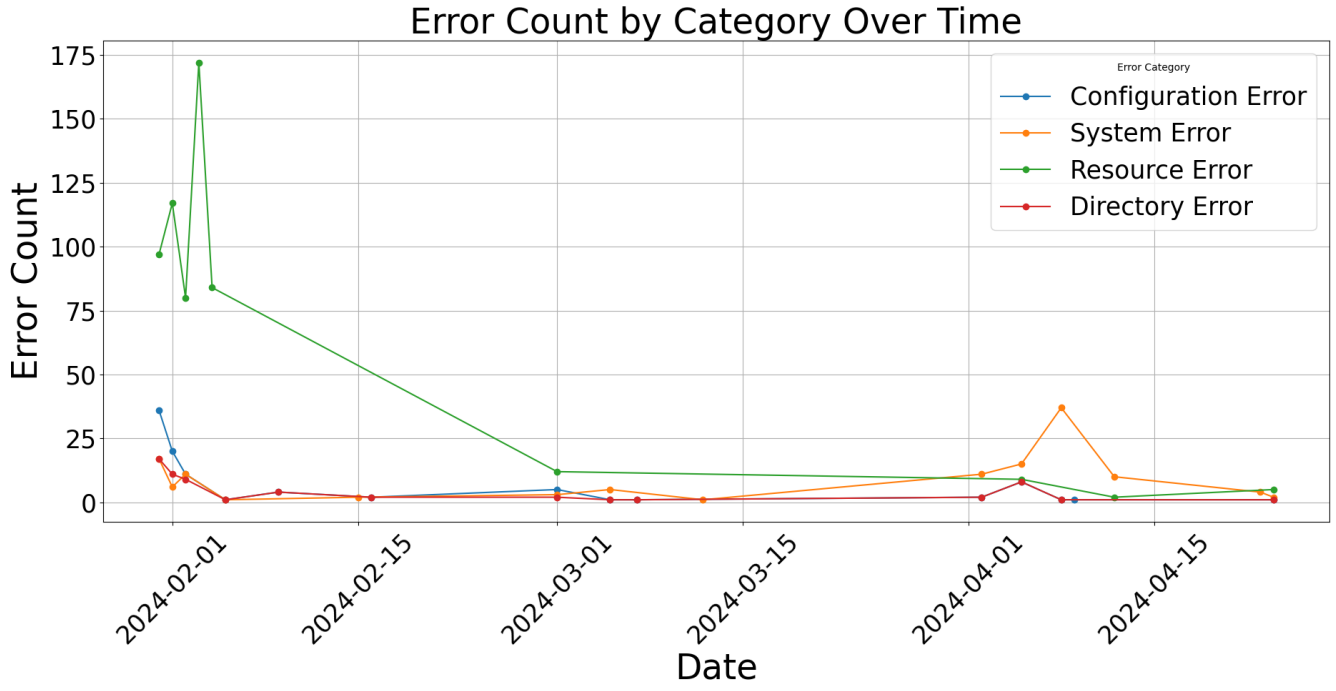

Fig. 4. Distribution of error types in the educational cluster over time. The colors represent different error categories: blue indicates configuration errors, orange represents system errors, green denotes resource errors, and red signifies directory errors.

a dramatic decrease over time. Initially high, these errors suggest a steep learning curve for students in managing resources or improvements in resource allocation protocols and user training. The significant initial error count indicates an insufficient preliminary understanding of resource management, which improved with experience and educational interventions.

Directory Errors (Red): Least frequent, at around 6% of total errors (50 errors), directory errors were infrequent but showed occasional spikes. These spikes might be due to specific incidents. Generally, these errors remained low, indicating a reasonable understanding of directory management among students.

Detailed Analysis of HPC Cluster Usage Survey

The survey data reflects a diverse range of familiarity and confidence levels among students regarding various aspects of HPC systems. The responses are structured to assess theoretical knowledge, practical skills, and overall satisfaction with the HPC lab experiences. Before we analyze the specific results, it is important to highlight that this section of the questionnaire was designed to understand how students have absorbed the theoretical HPC concepts presented during the course. The results discussed here directly reflect the success of the teaching strategies employed to convey these concepts.

Understanding of HPC

This section of the questionnaire aims to understand the students' level of knowledge about fundamental HPC concepts. The questions focus on assessing the depth of theoretical understanding that the students have acquired during the course.

The responses revealed that 35% of the students feel confident in their understanding of HPC systems, suggesting that the instructional content effectively covers complex aspects of HPC architecture for many students. However, 41.9% of the students possess only a basic understanding, indicating familiarity with general concepts but lacking depth. This may highlight a need for more detailed instructional content or additional practical exercises. Additionally, 12.9% of students struggle with fundamental HPC concepts, suggesting that remedial sessions or enhanced supplementary resources could help elevate their comprehension to the class average.

The next set of questions measures students' confidence in using SLURM commands, which are essential for task management in the HPC environment. The goal is to understand if the students feel prepared to use HPC environments to solve complex computational problems. The responses indicate that 35.5% of students demonstrate high confidence in using commands like 'srun' and 'sbatch', indicating their capability to efficiently manage computational tasks in HPC environments. Meanwhile, 41% of students have moderate confidence, understanding SLURM commands but remaining uncertain about when and how to use them effectively, highlighting a potential area for targeted training or workshops. Finally, 19.4% of students have low confidence in using SLURM commands, suggesting that basic training sessions focusing on these commands are essential for them to fully utilize HPC resources.

The final set of questions aims to measure the students'

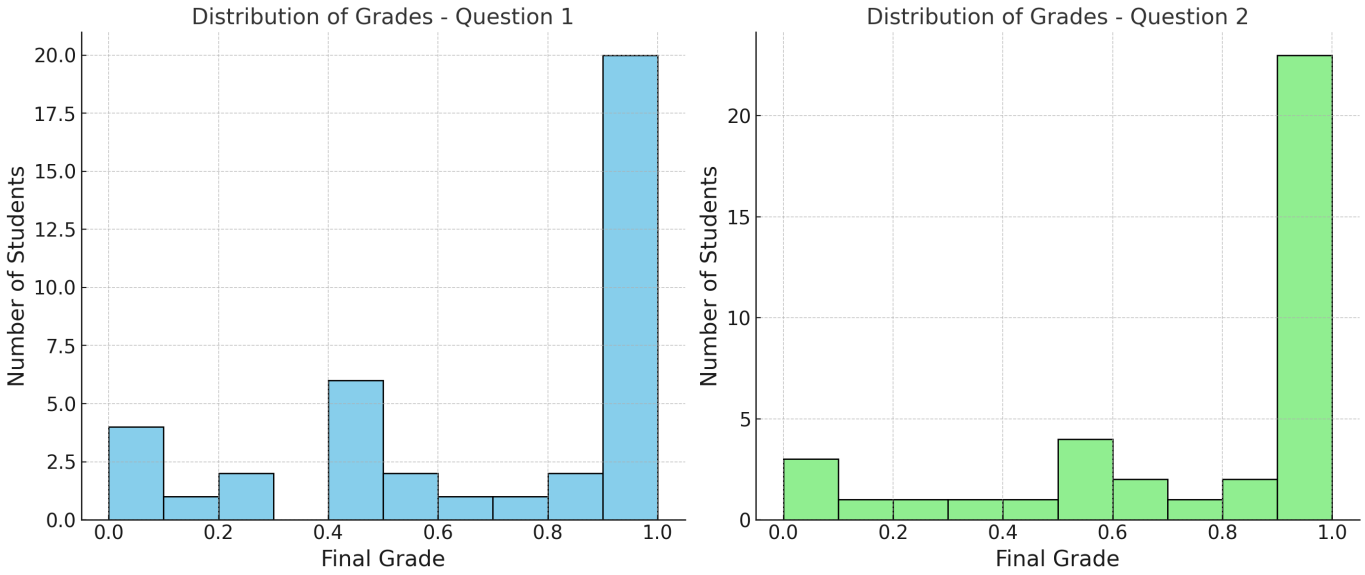

Fig. 5. Distribution of grades for 40 students across two distinct questions. Question 1: Importance of aligning software strategies with hardware resources for problem-solving, analyzing examples, concept comprehension, and depth. Question 2: Critical analysis of strategies for optimal job allocation in cluster administration, focusing on examples, concept comprehension, and depth.

satisfaction with their practical experiences in the HPC lab. The goal is to determine whether the students feel equipped to independently execute complex computational operations.

The responses show that 74.2% of students are very satisfied with their practical experiences in the HPC lab, reflecting positively on the lab's setup and the relevance of the exercises to their learning. However, 25.8% of students are somewhat satisfied, indicating room for improvement. Their feedback is crucial for refining the practical exercises and better integrating theoretical and practical content. Notably, no students reported being dissatisfied with their experience using the cluster.

The chart in Figure 5 provides a comparative view of student performance on two distinct questions during the midterm evaluation. Question 1, which addresses the importance of aligning software strategies with hardware resources for problem-solving, shows a broad distribution of grades, with a significant number of students achieving the maximum grade (1.0). This indicates a strong comprehension of the concept among many students, though some variation suggests room for further improvement.

Question 2, focusing on the critical analysis of strategies for optimal job allocation in cluster administration, similarly shows a majority of students scoring high, with many achieving the maximum grade. This suggests a solid understanding and ability to critically analyze the concepts taught.

However, the distribution in both questions indicates that while many students excel, there are still a number who do not reach the highest levels of comprehension and application. This points to the need for enhanced instructional strategies, particularly targeted support and additional resources, to help all students achieve a more uniform level of proficiency.

In conclusion, while the high performance of many students demonstrates the effectiveness of the current instructional approach, the variation in grades suggests that continuous improvement efforts are necessary. By providing additional support and refining teaching methods, educators can help ensure that all students develop a strong understanding of both theoretical and practical aspects of HPC.

VI. Conclusion

The research and analysis detailed in this article provide a experiential report of the educational use of a cluster in a Computer Engineering course. This experiential report demonstrates how using a cluster can demystify HPC concepts and bring practical activities to the forefront, enabling students to solidify their theoretical knowledge by solving problems in a real-world simulated environment. The key findings are summarized as follows:

Job performance and resource utilization data revealed that while a substantial proportion of jobs processed by the cluster were successfully completed, a number either failed or were canceled. Resource allocation errors were the most frequent, while configuration and system errors were less common. This indicates that lessons on job profiling and resource allocation need to be improved. Additionally, more activities should be developed on these topics to solidify students' understanding and proficiency in this area.

The analysis helps identify how the cluster resources are utilized and can guide improvements in resource management and scheduling strategies. Several enhancements could be made to optimize resource allocation and job scheduling, thereby reducing the number of failed or timed-out jobs and better supporting the educational objectives of the course.

Enhanced training on job configuration could provide more in-depth sessions focused on resource estimation, helping

students better understand how to allocate resources efficiently and reduce the number of failed jobs due to improper configuration. Automated resource allocation tools could assist students in accurately estimating the necessary resources for their jobs by analyzing past job data to provide recommendations on memory, CPU, and time requirements. Improved error feedback mechanisms could help students quickly identify and resolve issues by providing detailed error messages and potential solutions. Adaptive scheduling policies that prioritize jobs based on resource availability and job requirements could improve overall cluster efficiency.

Resource monitoring and alerts could help both students and instructors keep track of resource usage and identify potential bottlenecks or issues early on, preventing resource contention and ensuring smoother operation. Job profiling and optimization workshops could help students learn how to write more efficient code and optimize their algorithms for better performance on the cluster. Finally, standardized submission templates that include best practices for resource requests and job configuration could help students avoid common pitfalls and improve the success rate of their jobs.

By implementing these improvements, the cluster's resource management and scheduling can be optimized, leading to a more effective educational environment. This will not only enhance the students' learning experience but also ensure that they are better prepared for real-world computational challenges.

Survey results indicate a diverse range of understanding and confidence among students concerning HPC concepts and practical skills. Although many students are confident and highly satisfied with their HPC lab experience, a significant number still struggle with basic concepts and practical applications.

Furthermore, data from midterm exams suggest a high level of competency in the assessed topics, with most students achieving maximum scores. However, variations in performance across different questions suggest that additional support might be beneficial for some students.

Based on these findings, targeted training sessions focused on specific aspects of HPC can help bridge the knowledge gaps identified among students. These sessions should cover topics such as resource estimation, job submission procedures, and troubleshooting common errors. Additionally, creating and integrating tools for resource estimation and management can enhance students' ability to effectively utilize the cluster. These tools should provide intuitive interfaces and real-time feedback to assist students in making informed decisions about resource allocation.

The analysis also highlighted that the initial high incidence of resource errors underscores the need for improved training on resource management and allocation. The stability in system errors supports ongoing effective management but encourages continued monitoring and prompt issue resolution. The occasional peaks in directory errors suggest the need for clearer documentation to minimize confusion. The decrease in configuration errors indicates learning and adaptation but also highlights the importance of providing comprehensive initial training and user-friendly configuration tools.

This experiential report emphasizes the need for ongoing enhancements to instructional materials to improve student learning outcomes and operational efficiency. The use of the cluster as an educational tool has proven effective in reducing the abstraction of HPC concepts and providing hands-on experience that simulates real-world computational environments.

Recommendations include more targeted training sessions and the development of better tools for estimating and managing resources to support students' successful engagement with this new approach. Additionally, integrating real-world projects and case studies into the curriculum can bridge the gap between theory and practice.

By implementing these measures, students can develop a stronger grasp of HPC concepts and practical skills, leading to greater confidence and proficiency in their HPC lab experience. These improvements will significantly enhance the educational experience and technical proficiency of students in HPC, preparing them for real-world computational challenges.

REFERENCES

[1] R. K. Raj, C. J. Romanowski, J. Impagliazzo, S. G. Aly, B. A. Becker, J. Chen, S. Ghafoor, N. Giacaman, S. I. Gordon, C. Izu, S. Rahimi, M. P. Robson, and N. Thota, "High performance computing education: Current challenges and future directions," in *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, ser. ITiCSE-WGR '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 51–74. [Online]. Available: https://doi.org/10.1145/3437800.3439203

[2] H. Cornelius, *The Future of High-Performance Computing (HPC)*. IGI Global, 2018, pp. 4004–4017. [Online]. Available: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-5225-2255-3.ch347

[3] P. O. A. Navaux, A. F. Lorenzon, and M. d. S. Serpa, "Challenges in high-performance computing," *Journal of the Brazilian Computer Society*, vol. 29, no. 1, p. 51–62, Aug. 2023. [Online]. Available: https://sol.sbc.org.br/journals/index.php/jbcs/article/view/2219

[4] S. K. Prasad, T. Estrada, S. Ghafoor, A. Gupta, K. Kant, C. Stunkel, A. Sussman, R. Vaidyanathan, C. Weems, K. Agrawal, M. Barnas, D. W. Brown, R. Bryant, D. P. Bunde, C. Busch, D. Deb, E. Freudenthal, J. Jaja, M. Parashar, C. Phillips, B. Robey, A. Rosenberg, E. Saule, and C. Shen, "NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing - Core Topics for Undergraduates, Version II-beta," Online, p. 53, 2020, online: http://tcpp.cs.gsu.edu/curriculum/.

[5] A. Radenski, "Integrating data-intensive cloud computing with multicores and clusters in an hpc course," in *Annual Conference on Innovation and Technology in Computer Science Education*, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:17884185

[6] Networking and Information Technology Research and Development, "National strategic computing initiative," 2024, accessed: 2024-04-15. [Online]. Available: https://www.nitrd.gov/nsci/

[7] European Commission, "European declaration on high performance computing," 2024, accessed: 2024-04-15. [Online]. Available: https://digital-strategy.ec.europa.eu/en/news/european-declaration-high-performance-computing

[8] L. S. C. Lima, T. A. O. Demay, L. N. Rosa, A. F. M. Batista, and L. Silva, "Enhancing supercomputing education through a low-cost cluster: A case study at insper," *International Journal of Computer Architecture Education (IJCAE)*, vol. 12, no. 2, pp. 11–19, 2023.

[9]

[10] TOP500 Supercomputer Sites, "Top500 list," https://www.top500.org/, accessed: Mar. 3, 2024.