Filling the gap between education and industry: evidence-based methods for introducing undergraduate students to HPC

Fabio Banchelli
Computer Science
Barcelona Supercomputing Center
Barcelona, Spain
fabio.banchelli@bsc.es

Filippo Mantovani
Computer Science
Barcelona Supercomputing Center
Barcelona, Spain
filippo.mantovani@bsc.es

***Abstract*—Educational institutions provide in most cases basic theoretical background covering several computational science topics, however High Performance Computing (HPC) and Parallel and Distributed Computing (PDC) markets require specialized technical profiles. Even the most skilled students are often not prepared to face production HPC applications of thousands of lines nor complex computational frameworks from other disciplines nor heterogeneous multinode machines accessed by hundreds of users. In this paper, we offer an educational package for filling this gap. Leveraging the 4-years experience of the Student Cluster Competition, we present our educational journey together with the lessons learned and the outcomes of our methodology. We show how, in a time span of a semester and an affordable budget, a university can implement an educational package preparing pupils for starting competitive professional careers. Our findings also highlight that 78% of the students exposed to our methods remain within the HPC high-education, research or industry.**

***Index Terms*—Educational, HPC, Cluster Competition, Higher Education**

I. Introduction

Universities have the mission of training tomorrow's professionals. Due to the increasing importance of High Performance Computing (HPC) and Parallel and Distributed Computing (PDC) in several areas of science and society [1], [2], the universities have the role of prepare high-level profiles for this market.

The HPC and PDC professional roles require a wide range of technical skills, like installing and administrating complex systems, squeezing performance out of scientific applications or managing hundreds of virtual machines in the cloud [3]. The complexity of modern HPC systems is such that it requires a team of skilled professionals to operate it. Because of that, it is of paramount importance to have well-trained technicians with effective communication skills complementing a strong technical background.

As it is not clear how educational institutions offer curricula able to prepare these high-level professionals ready for the HPC market, we perform in our paper a background study on a small group of universities in Europe. The goal of this preliminary analysis is to survey what universities teach to students regarding HPC and PDC, and the result is that there is a non-negligible gap between what students touch in the universities and real life.

Leveraging four years of experience with the Student Cluster Competition (SCC), we provide here our methodology to form a team of students and train them such that they can face the job market with a more solid background in HPC and PDC. SCC, an initiative by the HPC Advisory Council [4], puts in fact students as users and administrators of their own mini-cluster. Participants of the SCC have to run real HPC applications and benchmarks while trying to get the most performance with a power cap of 3 kW.

The main contributions of this paper are: *i)* the presentation of a comprehensive educational package able to introduce students to a semi-professional HPC level; *ii)* the analysis and lessons learned from four years of participation at the SCC.

The document is structured as follows: in Section II we discuss previous work done in education for PDC and HPC. In Section III we analyze the Computer Science and Computer Engineering BSc curricula in HPC/PDC of four European universities. Section IV introduces the SCC and explains how it can have a positive impact on the students' career. Section V details the topics listed in Table I and how are covered by our method. We divide our methodology in Section V-A, where we disclose the technical background, and Section V-B which explains how we create and organize a team. Section VI quantifies the quality of our methodology based upon the results of four years of refining. Lastly, Section VII rounds up the document describing the lessons learned and the conclusions from our experience.

II. Related work

This paper is inspired by the work of a colleague advisor [5] who collaborated with one of the authors in [6]. It differentiates from it, as it presents a comprehensive set of guidelines to implement an educational package, rather than defining professional profiles needed in HPC and how SCC is a useful tool to train them.

Harrel et al. in [7] also presents the SCC as an educational tool, but their work focuses on how to form and manage a team for the competition. It also includes methods to get support from institutions and sponsorships.

The following works focus on effective methods to implement HPC and PDC into university curricula. The method proposed in this document differentiates from them in that it is an activity that happens outside the classroom, in addition to the university curriculum.

In [8] and [9], the authors propose an educational resource to be used as a teaching tool in HPC during a course: they both focus on the infrastructural part (clusters), while our educational package does not depend on a specific infrastructure.

Liu in [10] presents the coverage of a course in PDC. This document discusses the importance and benefits of such class and how to make parallel processing interesting to students. Cahill et al. [11] address the lack of knowledge of instructors in parallel computing. Their work describes some problems of current parallel computing courses and proposes a model to overcome them. Finally, in [12] the author details a parallel programming course in the Computer Science BSc program, including assignments. This work describes the contents of the course inside and outside the classroom.

The reader interested in fine-grained details of history and rules of relevant international cluster competitions should check out [13], a thorough view into the Asian student Supercomputing Challenge (ASC) insides.

III. About HPC in university curricula

The Partnership for Advanced Computing in Europe (PRACE) [14] is a non-for-profit European network of data centers, with 26 member countries represented by research institutions. Its goal is to enable high-impact scientific discovery and engineering research within Europe. PRACE performs education and training in HPC effort by organizing seasonal schools, workshops, and seminars hosted by member institutions.

The PRACE supercomputing infrastructure is provided by five member institutions. We selected four universities affiliated with PRACE members and analyzed their Computer Science and Computer Engineering BSc curricula: *Universitat Politècnica de Catalunya* (UPC) [15], related to BSC from Spain; *Università di Bologna* (UNIBO) [16], related to CINECA from Italy; *ETH Zürich* (ETH) [17], related to ETH Zurich/CSCS from Switzerland; and *Universität Stuttgart* (UST) [18], related to HLRS from Germany.

Our analysis of the curricula starts by looking if they include a major in HPC. Following the curriculum proposed in [19], which lists PDC and HPC critical topics and gives advice on how to teach them, we isolated a list of relevant topics. Also, we complement them with what is highlighted as essential by Sterling et al. in [20], which is a comprehensive overview of fundamental HPC concepts.

Our list of topics is divided into four sections. The *Crosscutting* topics cover PDC and HPC at a high level. *Computer Architecture* includes notions at the hardware level while *Parallel Programming* focuses on programming models. *Cluster management* is comprised of topics on how to use a cluster and do research from the user to the administrator point of view. The resulting list is shown in Table I. Each row represents a topic and each column one of the selected universities. A cell marked with an ✓ indicates that a certain topic is covered in that university's curriculum.

TABLE I
Coverage of HPC topics in four European universities

Topic	UPC	UNIBO	ETH	UST
1. Crosscutting				
1.1 Why and what is PDC/HPC?	✓	✓	✓	✓
1.2 Anatomy of a Supercomputer	✓			
1.3 History of Supercomputing				
1.4 Computer Performance	✓	✓	✓	✓
1.5 Fault Tolerance	✓		✓	
1.6 Security in Distributed System	✓		✓	
2. Computer Architecture				
2.1 Flynn's Taxonomy	✓	✓	✓	✓
2.2 Pipelines	✓	✓	✓	✓
2.3 Instruction Level Parallelism	✓	✓	✓	
2.4 Branch Prediction		✓		
2.5 Out-of-Order Execution		✓		
2.6 Multithreading	✓		✓	✓
2.7 Vector/SIMD	✓	✓	✓	
2.8 Memory Hierarchy	✓	✓	✓	✓
2.9 Shared Memory	✓	✓	✓	✓
2.10 Distributed Memory	✓	✓	✓	✓
2.11 Heterogeneous Architectures	✓	✓	✓	✓
3. Parallel Programming				
3.1 Vector/SIMD (SSE, NEON)	✓	✓	✓	
3.2 Shared Memory (OpenMP)	✓	✓	✓	✓
3.3 Distributed Memory (MPI)	✓	✓	✓	✓
3.4 GPGPU (CUDA)	✓	✓	✓	✓
3.5 Heterogeneous (FPGA)	✓			✓
3.6 Parallel Algorithms	✓	✓	✓	✓
4. Cluster management				
4.1 Basic usage (login, queues)	✓	✓	✓	✓
4.2 Applications/Benchmarks				
4.3 High Performance Libraries				
4.4 Profiling and Debugging				
4.5 Cluster/Node installation	✓			
4.6 System software installation				
Offers HPC/PDC specific course	✓	✓	✓	✓

All the universities have a course in HPC, but they seem only to cover parallel programming, often missing out in the history of HPC and its evolution. Universities also have substantial coverage of computer architecture with an apparent lack in some advanced topics like branch prediction and out-of-order execution, which are only briefly mentioned. We conclude that the available curricula fail to give a practical view of HPC to students. The theoretical background is strong, but it is missing a hands-on experience. Students only get a chance to login into a cluster and run micro-codes, but they do not get to know the whole software infrastructure that sits behind and that enables their research. Besides the highlighted lacks, it is important to note that neither in [19] nor

in the analyzed curricula is mentioned the power consumption of computational infrastructures. The development of energy awareness in the student minds is of paramount importance for creating next-generation HPC professionals [21], [22].

IV. SCC as educational challenge

To fill the educational gaps often left open by the university curricula, highlighted in section III, we present a package of activities result of the preparation of the student cluster competition. Therefore we dedicate this section to introduce the rules, the training and the participation activities related to the student cluster competition.

When referring to the SCC we mean the cluster competition organized by the HPC Advisory Council which is held every year in International Supercomputing Conference (ISC), the largest HPC conference in Europe. Two other renown competitions are *i)* Student Cluster Competition at the ACM/IEEE Supercomputing Conference (SC) in the USA and *ii)* ASC Student Supercomputer Challenge in Asia. Each one of the competitions has a different set of rules, but all of them share some common traits:

- Participants are undergraduate students;
- Teams have their cluster which they have to administrate;
- Students have to run a set of benchmarks and real scientific applications in their machines;
- The cluster has to run under a power consumption cap (usually 3 kW);
- External judges interview each team to evaluate their overall knowledge;
- Awards are given taking into account benchmark and application performance and the evaluation provided by the judges.

In the case of the SCC at ISC, the competition concludes assigning an award to the highest Linpack score, three prizes to the best overall teams and the *Fan Favorite award* for the most voted team through an online poll. The SCC is of high relevance for the HPC ecosystem beyond training undergraduate students. In Figure 1 we show the progression of the Linpack score (line/left axis) of the winning team throughout the last three years of cluster competitions. We include the power efficiency (points/right axis) of the team computed as the achieved performance divided by 3 kW which is the power cap. For comparison, we also plot the most efficient supercomputer in the Green500 [23] list. The reader will notice how the power efficiency of the systems closely follows the Green500, overperforming them on several occasions like in the SC16, ASC17, and SC17. This shows how close are the students participating in the SCC to the top HPC excellence. The students not also learn how to run HPC codes, but also to tune them to obtain the highest efficiency on their machines, taking into account both performance and energy consumption.

A. Team proposal and submission

The submissions for participation open around September. Teams must prepare a proposal document and submit it before

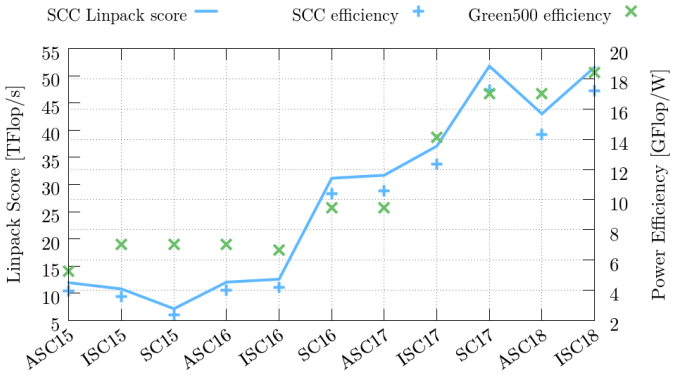

Fig. 1. Highest Linpack performance (line / left axis) and power efficiency (points / right axis) in previous cluster competitions compared with contemporary most efficient system in the Green500 list.

November. This document must describe the structure of the team, a motivation letter, a general overview of the hardware to be used during the competition and the commitment of the institution that supports the team to education in HPC.

Teams are formed by six undergraduate students or four master students. The team will have to be accompanied by an advisor that will guide the students during the preparation leading to the competition. Some teams may incorporate additional advisors who were participants in previous competitions. Students are also encouraged to name their team.

The list of the team accepted is announced during the SC conference in mid-November. Teams then have until April to submit the final architecture description of their cluster and other team details.

B. Preparations before the competition

Teams will be asked to run a series of benchmarks and applications during the competition, but these are not disclosed all at once. Instead, the HPC Advisory Council updates periodically their website uncovering the applications as the competition approaches. Other details are also addressed like application versions or additional rules.

The first update arrives around January which means that teams have two months between receiving the notification of acceptance and having the first code targeting the competition. This time window is essential to introduce theoretical and practical concepts to the team, such as how to access a cluster, how to run a parallel job, how to perform a scalability study, how to detect and classify bottlenecks. Figure 2 shows the timeline of announcements and deadlines of the SCC18.

Benchmarks are simple codes with little to no dependencies. A total of three benchmarks are included during the competition usually including Highly Parallel Linpack (HPL) and most recently, High Performance Conjugate Gradients (HPCG).

In contrast, the scientific applications are complex codes with multiple dependencies. Teams are required to install and run them but are also encouraged to profile executions and change the code to better fit their cluster, improve the performance or reduce power consumption. These applications

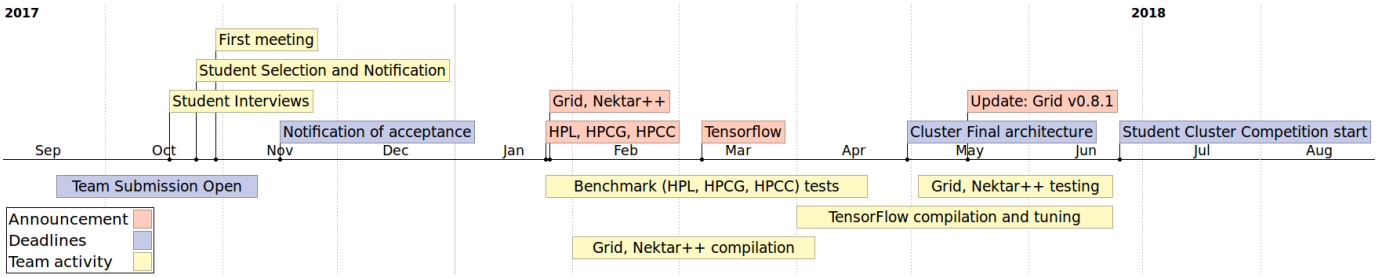

Fig. 2. Preparation timeline of the SCC18

usually need an input set to run. The input sets are disclosed during the days of the competition so the teams might not know which parts of the code the input set will target (e.g., solvers, preconditioners). The competition includes three applications. In recent years, one of the applications was Tensorflow, explicitly targeting artificial intelligence and machine learning. Table II shows the benchmarks and applications that the students had to run in the last four SCC.

An additional application or challenge is disclosed during the competition. These secret challenges can be a different input set of a previously know application or a code that does not require a large number of dependencies (e.g., mini-apps). Also, within the secret challenge, students could be judged not only on performance but also on other parameters, like for example minimizing power consumption peaks.

TABLE II
LIST OF BENCHMARKS AND APPLICATIONS REQUESTED IN THE LAST FOUR STUDENT CLUSTER COMPETITIONS

Year	Benchmarks	Applications	Secret App
2015	HPL, HPCC	LAMMPS, PyFR, Octopus	PyFR
2016	HPL, HPCC	Graph500, Splotch, WRF	Cloverleaf
2017	HPL, HPCC, HPCG	FEniCS, MiniDFT, Tensorflow	LAMMPS
2018	HPL, HPCC, HPCG	Grid, Nektar++, Tensorflow	NEK5000

C. During the competition

Once at the conference venue, students have to set up their booth where they will spend the following four days. This includes setup the cluster which may have been shipped disassembled. The cluster is connected to a PDU, provided by the organizers, that monitors the power consumption and displays the instantaneous power drain of the cluster. The PDU is also connected to a network through which the organizers can gather power information and show it in a graphical interface for all participant teams. If the power consumption of the cluster surpasses 3 kW during the competition the judges will ask the team to restart the run. Three peaks over the power cap will disqualify the team.

During the competition kickoff, teams present themselves and receive general instructions on how the competition will unravel. At this moment the students also receive their first assignments while at the end of the day they have to hand out the results to the judges. The second and third day follow the same structure: the students are instructed on which code and input set to run and the have to submit the results at the end of the day.

Throughout the competition, attendees of the conference can visit the SCC site, meet and vote for the teams. Votes are digitally collected and used for assigning the *Fan Favourite award*. Judges also interview the students asking about the work they did in preparation for the competition. These interviews are evaluated and weight towards the *Best overall award* and require a good understanding and speech ability from the students [24]. Teams are encouraged to add decorations to their booths such as posters and other material to aid them when explaining their work to judges and visitors.

V. THE EDUCATIONAL PACKAGE

This section lists the HPC knowledge that the students acquire during the SCC. The learning experience is based on the technical background discussed in Section I, but also includes soft skills like team communication and speech abilities as well as effective workflow organization.

A. HPC fundamentals

First and foremost, students need to know the machine they are working on. They have to be able to login to the cluster and be knowledgeable that multiple users may use the machine, so a queueing system will manage the workloads to be executed. Systems may vary in how many login and compute nodes they have. Users may also need to go through a gateway to access the machine. In our case, we introduce our cluster infrastructure during the first meetings: we noticed that the most effective communication requires a graphical representation of the network, the compute and login nodes. We use SLURM as a queueing system and NFS as distributed filesystem across compute nodes. Students quickly learn to interact with these tools as users. All of these requirements are within topic 4.1 of Table I. We established that all of the studied curricula cover this topic. However, students may need a refresh and the machine used at the SCC may be different than the one they are accustomed.

The competition will require the students to know how to compile, run and optimize complex applications. The team must be able to identify the code dependencies and link them accordingly. As an example, the library known as Grid, which was used in the SCC18, has up to 100K lines of C/C++ code and depends on libraries such as HDF5 and FFTW to work.

Even more significant is Nektar++, another application of the SCC18 which sums up to 260K lines of C/C++ code, 10K lines of Python and depends on 12 software packages to be able to implement all of its functionalities. Students have to know about these software tools and scientific libraries but, as seen in Table I, none of the curricula cover commonly used HPC libraries (topic 4.3).

In our four years of experience, the teams have always participated with Arm-based machines. In 2015, students used the Mont-Blanc project prototype [25], based on Armv7 SoCs. The following two years the team used nodes powered by Cavium ThunderX CPUs which are Armv8. In 2018, students had the opportunity to work with the very recent Cavium ThunderX2. The software ecosystem of Arm has been evolving in recent years, as described in [26], but it still lacks comprehensive support to scientific and numerical libraries. This means that students often found themselves with codes that have no support for their architecture or libraries not available as system packages. However, we find this a good opportunity to train the team, so they learn how to deploy a full software stack compiling all the libraries from source with most of them requiring some tinkering of the build scripts or even replacing small parts of the source code, e.g., architecture specific intrinsics.

Once the code is compiled the team faces two challenges:

1) To find a meaningful input set for the application to run. We encourage students to read the documentation and try to understand how to test their application. We also promote the interaction with the community or the developers of the applications.
2) To profile the execution and to identify key points of the code that could become a target of optimizations. The methodology for running, profiling, optimizing and testing a complex HPC code is labeled as topic 4.4, and it is not covered in any of the educational programs.

Our method leverages BSC performance analysis tools, Extrae and Paraver [27]. We teach the students how to collect performance traces using the Extrae instrumentation library. We then dedicate special sessions to study the traces using Paraver [28], a visual interface that helps to navigate the behavior of the code by plotting the data collected with Extrae in timelines or histograms.

As mentioned in Section III, understanding the power consumption of a cluster while running parallel application, is a crucial factor. We encourage students to measure the power consumption of their execution as well as proposing possible approaches to achieve better energy efficiency. To this end, we find instructive to gather power data and convert it to the Paraver format, as described in [29]. This method allows us to complement the studies using performance traces gathered with Extrae with power figures, using a single powerful tool, the Paraver visualizer. Combining both performance and power traces, the students can understand the fundamental relationships between power drain and hardware resources. The primary lesson is always that compute bound section of the codes are the one draining more power, while spots with slow power drain can identify memory bound regions. The topic of power consumption has not been included in Table I and is missing in most Computer Science BSc curricula.

Careful performance analysis drives the students to understand and attempt to overcome bottlenecks. Our method encourages changing the source code, maintaining its correctness, by experimenting with load balance techniques or improving data layout in memory. During SCC16, for example, students were asked to run Splotch [30], a ray tracer to visualize particle simulations. The students noticed a significant load imbalance while running with the hybrid MPI+OpenMP version of the application. Consequently, they have been exposed to the problem of load balancing, and they tested Dynamic Load Balancing (DLB) [31], [32], a library developed at BSC that dynamically reschedules workload at runtime to maximize resource utilization.

Our method also encourages the development of an HPC system administrator profile: the team needs in fact to administrate its own cluster. As the machine is usually delivered with minimal configuration, students will learn how to install an operating system and configure it for a cluster use (e.g. deploying a multi-user environment, a distributed filesystem and a queueing system). We also teach to think the cluster as a service accessed by users that will have specific software requirements. The administrators, therefore, have to provide a minimal set of "system software" for supporting at least the dependencies of the applications of the competition. One example of this experience is covered in Section VI: the team of students had to contact a component manufacturer to obtain a yet to release firmware for supporting the hardware of the cluster. This experience is, again, not covered in any of the curricula (topics 4.5 and 4.6).

B. Team organization and communication

There is little to no support material for advisors on how to create a team for the SCC, beside [7].

During our four year experience, we used two methods to find students. At first, we relied on recommendations from professors or previous participants. This made it easier to find candidates and the students had a strong technical background. However, we noticed that selecting the best students looking at their grades to compete in a team is not always the best method: these students are used to be on top of their respective classes and tend to be less collaborative and less proactive than other students. More important than having a strong technical background, students participating in SCC need the ability to work as a group during the competition. Each student has a different profile and behaves differently when asked to work with other people. For this reason, we decided to change our selection method to a more time consuming, but more effective. We published on the university portal an invitation for an info-session, and we dedicated a full hour to introduce the Student Cluster Competition, leaving extra time for questions. Immediately after this info-session, we performed personal interviews asking technical and non-technical issues. Of course, when following this selection method, the advisors

have extra work to evaluate the profile of each applicant, but we proved that this method yielded a highly motivated team, with a strong technical background. Beside these objective observations, this method has the psychological added value of creating the feeling of "being selected", which increases the sense of responsibility in the student's perception.

Once the team members have been selected, it is essential to establish a relation between all the members. Some teams in the SCC have the most experienced student as a captain. Our approach has always been to avoid a hierarchical structure: every individual is at the same level. The team members may not know each other beforehand, so the advisors must establish bonds between team members organizing meetings, friendly dinners and activities to enforce a good group dynamic.

Students will have to confront a significant workload throughout the competition. The advisors ought to help them at first, but the team must be able to identify tasks to be performed and prioritize them. Tasks are not only run benchmarks and applications but also reading literature on relevant subjects and promote the team image on social media. When discussing the more technical tasks like running, profiling and optimizing codes, we attempted two approaches: *i)* Assign tasks to individuals or pairs of students; and *ii)* Each student face the same assignment and compares results with the rest of the team. The first approach allows for more breadth in the tasks since the workload gets distributed throughout the team members.

On the other hand, the second approach accomplishes a better depth in each task since multiple points of view are shared for a single problem. We tried the latter method with one of the benchmarks of the SCC, HPCG. Students were asked to compile and run the code and write down their scores in a spreadsheet along with information of their build (e.g., compiler, compilation flags, input set). Students had fun competing to get more performance, and an excellent level of knowledge has been achieved, so we tend to prefer the second approach.

Besides technical skill, communication abilities are of great importance. Students must be able to explain the work they perform and how they obtain results. This is key for healthy internal communication, but it is also important in the preparation of the interviews that judges perform during the competition. To this end, we also organize "trial interview" in which each student is interviewed for a couple of minutes by the rest of the team. This activity has a tremendous effect on the self-consciousness of the group, highlighting in a simple and funny way the student more skilled for communications and the ones more introverted.

Reproducibility and time control of the jobs is another important aspect that we tend to stress. Students must always have the feeling of which are the vital parameters to obtain a given result and how long it takes to reach it. This is generally important, but above all, necessary for the competition, where time slots assigned to each application tends to be tight.

Another important aspect is time sharing and synchronization. Students have to coordinate their classes with the competition and the advisors need to fit their schedule with the one of the rest of the team. It is very important to establish efficient methods of communication between members of the team. When delivering a message to the team, we distinguish between time critical and non-blocking communications. For a message that needs quick response, with a maximum delay of 5 minutes, we use instant messaging platforms. For non-time-sensitive messages and for messages where it is useful to follow the discussion, an email is preferable. It is very important that students evaluate the importance of the message and use the appropriate channel of communication. Students are also prompted to share their results with the rest of the team. Some applications will yield a large output which is inconvenient to share via email. For this reason we also use a cloud-based filesharing system.

Periodic meetings are the best tool for synchronization. A meeting must always include an agenda preferably with a schedule to ensure that all the important topics are covered, and the pace is kept. We also experimented with assigning one or two organizers per meeting who had to propose the agenda and manage the meeting flow. To fix meeting appointments and to not miss any deadline we used a shared calendar for the team. During the meeting, tasks are usually created and assigned to one or more team members. We follow a SCRUM-like methodology to keep the team organized and efficient.

VI. Our method in numbers

This section gives a quantitative view of our methodology. We show the amount of work that the students have to face. We also show the impact of the SCC on past students. Lastly, we give some examples of the effect of the SCC beyond the competition.

A. Workload quantification

Meetings are the synchronization point of the team. These are usually around 2.5 hours long and happen periodically around every 15 days. Meetings start in January with the release of the first applications and get progressively more frequent when the competition approaches (June). In total, the students attend around 20 meetings throughout six months in preparation for the competition. Additionally, students are expected to work individually around 4 hours between meetings.

Last but not least, the competition requires the students to attend the ISC conference to run all the benchmarks and applications. This happens throughout 4 days where the team works 8 hours per day.

In total, the expected workload that the student has to go through sums up to:

$$20 \cdot (2.5 + 4) + 4 \cdot 8 = 162 \ hours$$

The European Credit Transfer and Accumulation System (ECTS) is a credit system designed to unify the recognition of educational qualifications within Europe. Credits represent an amount of workload that the student has achieved and have a direct correlation to hours of work. According to [33], one ECTS corresponds to 25 to 30 hours of work. It is also stated

that the workload of a full-time student year amounts to 60 ECTS and a one-term course corresponds to 6 ECTS.

Given our previous calculation, the workload that the students go through during the SCC ranges between:

$$162\ hours \cdot \frac{1\ ECTS}{30\ hours} = 5.40\ ECTS$$

and

$$162\ hours \cdot \frac{1\ ECTS}{25\ hours} = 6.48\ ECTS$$

B. Cluster usage

As our method originates from the Student Cluster Competition, the access to a cluster is critical. In our case, we took advantage of the Arm-based mini-clusters deployed at the Barcelona Supercomputing Center within the Mont-Blanc project. Even if we think a high-end cluster is not needed, gathering the team around some hardware tends to focus the work. Having the possibility of testing and measuring on a common platform, in fact, clarifies best practices and most effective work methods.

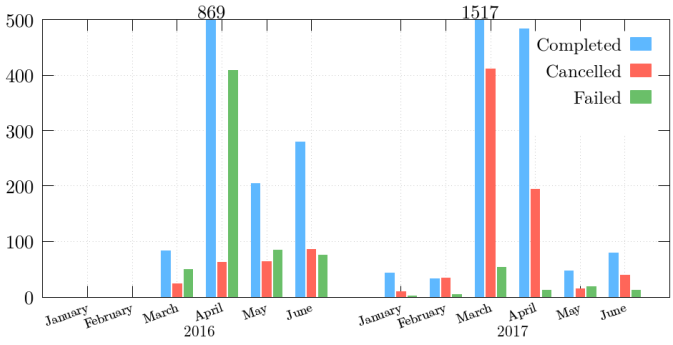

Fig. 3. Job dispatched by the SLURM job scheduler on the mini-cluster in preparation of the SCC in 2016 and 2017.

In Figure 3 we report a time distribution of the jobs executed for the SCC of 2016 and 2017. The total number of jobs executed for SCC 2017 was 3008, summing up to $\sim$ 530 hours of execution time. The utilization numbers show a light usage that we consider sustainable for any departmental shared cluster. However, a cheaper and more experimental solution can be found, like the one described in [34].

C. Budget

Implementing our SCC educational methodology has a cost. From a high level point of view, we can split the expenses relate to a complete SCC cycle (from team selection to the end of the competition) in three main classes: *i)* infrastructure cost, *ii)* work hours, *iii)* logistic cost.

We include in the *infrastructure cost* the expenses related to the cluster (renting/acquisition, transportation to/from the venue). Our experience tells that this is the most important direct cost that needs external support (sponsor): technological companies and system integrators are usually willing to help in this task in exchange of visibility within the competition. Also, it is easier for them to move/handle complex and bulky machines than for academic partners. Depending on the location and the cluster type, infrastructure cost can vary between 1 and 2 kEUR, for movement within Europe plus surcharges for renting. We do not have experience of intercontinental shipments.

The cost underneath the *work hours* is naturally absorbed by the educational institutions. Both students and advisors hours dedicated to the competition need to be "juggled" as extra-hours to be invested in meetings in the campus. A more quantitative analysis of this cost has been presented in Section VI-A.

Logistic costs include the registration to the event hosting the competition, generously offered by the organizers (HPC Advisory Council) in the case of ISC-SCC. Transportation, accommodation and living allowance are not covered by the organization, so they need some monetary support in advance (about two months before the competition in the case of the transportation). These expenses can be covered either by the educational institutions supporting the team or companies sponsoring the students. In our case, a balanced mix of both options allows us even out expenses. This cost can significantly vary depending on the lifestyle and the distance of the team to the venue. In our case, we quantify this chapter of expenses of $\sim$100 EUR per day per participant.

D. Where are they now

After four years of participation in the SCC, a total of 23 students were involved as participants: six new students for each iteration of the competition with one student who repeated as a team member for two years in a row. The experience was their first entry point to HPC and represented a turning point in their academic and professional careers.

After interviewing all of the students who participated in the SCC to know their current career status we classified them as follows: *i)* High Education (Students enrolled in MSc or PhD in HPC); *ii)* Industry (Professionals working in an HPC related company); *iii)* Research (Active researchers in the field of HPC); and *iv)* Employment unrelated to HPC (Currently employed in a company not related to HPC).

Figure 4 shows the spread of all the interviewed students. It is clear that the vast majority of students who participated in the SCC, 78.26%, are now part of the HPC community. A 34.78% of students were undergraduate by the time they participated in the competition and currently are completing their high education programs in HPC.

E. Impact on industry and research

The impact of the SCC is not limited to the students' career. Throughout our four year experience, we have had examples of the team's work impacting in industry and research. In this section, we will expose five examples of how the SCC team contributed to production systems and research.

Numpy bug in Armv7

The team at SCC15 had to face an issue with one of the dependencies of the applications, PyFR. The machine the students were competing with was the Mont-Blanc project

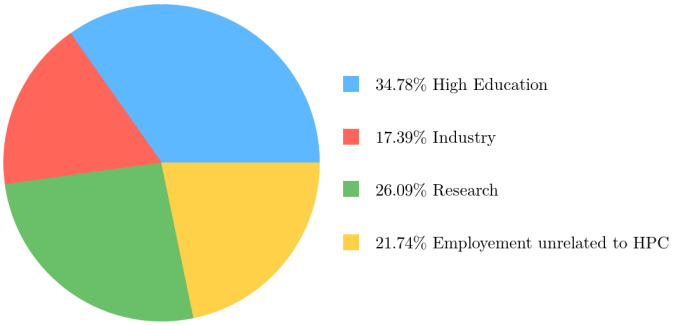

Fig. 4. Current career status of previous SCC participants

prototype [25], based on Armv7. Turns out that the Python library numpy had a bug which prevented the PyFR application to run as expected. The team contributed to the Python community by posting their issues in mailing lists [35] and interacting with the developers of the library.

Cavium ThunderX1 PAPI support

Academia also benefits from the effort of students during the SCC. The teams at SCC16 and SCC17 participated with a cluster based on Cavium ThunderX1. The ThunderX1 lacked Performance Application Programming Interface (PAPI) support, and it was not possible to du in-depth performance analysis of the platform. As a follow-up of the work done during the SCC, support for the PAPI library was implemented [36].

Poster in ACM Student Research Competition at SC17

Part of the work done in SCC17 was the stepping stone for a published poster in the SC17 conference [26] and is another example of the impact in academia of our methodology. In this work, the authors describe the experience of porting scientific applications to Arm-based machines in the context of the Student Cluster Competition.

Mellanox Infiniband drivers

Our team at SCC18 used a cluster based on Cavium ThunderX2 [37] which was announced to be generally available by the manufacturer just one month before the competition [38]. Thus, software support for the machine was sparse at the time being. Initially, the cluster nodes used Mellanox Infiniband FDR interconnect, but they were later upgraded to EDR. Students had trouble getting the Network Interconnect Controller because the manufacturer was not supplying the drivers needed to use this interconnect in the given platform. The team had to contact Mellanox to get the updated drivers which were made publicly available shortly after.

Tensorflow repository issue

SCC17 and SCC18 incorporated Tensorflow as one of the applications to be run during the competition. The students of the SCC18 had an active role within the community and were present in the official repositories giving feedback to users and developers [39].

VII. Lessons learned and conclusions

The ubiquity of parallel and heterogeneous computing requires well-trained technicians able to cope with varieties of complex tasks. The analysis provided in section III shows that universities not always offer a complete theoretical and practical coverage of the areas required for such specialized training.

For these reasons, we decided to leverage our experience in training multiple times a team for participating in the Student Cluster Competition for drafting an educational package that helps to complement the high education offered by the universities concerning HPC, PDC, cluster management, parallel computational science and performance analysis. The ultimate goal is to provide a tool for filling the gaps of knowledge left uncovered by the standard educational path.

We provide a quantitative approach to our educational package, detailing the amount of work that the students have to face before and during the competition, a budget overview, and the key subjects for creating next-generation HPC experts. We support our proposal analyzing the current occupation of our pupils who participated to SCC during the last four years and the impact of their work in industry and academia. The methodology presented in this document has complemented the educational curricula of ~ 30 undergraduate students from the University Politècnica de Catalunya, and the Barcelona Supercomputing Center, 78% of which are now actively part of the HPC community and have already contributed in research publications and production systems. A positive side effect that we discovered while deploying our educational package is that it allows students that do not know about research careers to get in contact with it.

The most important lessons learned are surprisingly non-technical. We discovered in fact that the natural introvert profile of several students in computer science is the most limiting factor when building a team for a long-term mission, as the SCC. A significant amount of time and effort needs to be invested therefore in developing robust communication methods, a common vocabulary to complement the rigorous scientific method deriving from the academic preparation. In person meeting and social events have been proven of great value for sharing technical information and building a more collaborative team.

Another key for the success is the adoption of simple and clear methods and tools for time management and responsibility assignment. Having problems well organized in tasks, with due dates in a SCRUM-like fashion makes the students part of a development team similar to a start-up company and boost enthusiasm and leave the brain of the students free for productivity.

Last non-technical lesson learned comes from the relationship among educational institutions, research centers, and industry. We noticed that having the economic and technical support of companies motivates the students. The role of industry is noticeable in the educational package: the industrial partners provide in fact “real-world” requirements that the educational institution can pre-digest and teach to the students. This dynamic makes the learning process focused and applied to concrete industrial needs. The advisors play a pivotal role in this process, as they are in charge of building and maintaining

the trust with industrial partners and to effectively transfer needs, problems and solution among all the players.

On the technical side, we noticed that using real systems (e.g., deployed production clusters) and practical problems (e.g., production applications) in the educational package slightly increases the complexity, but boosts the morale of pupils, as they see themselves close to the edge of research.

Acknowledgment

The authors of this paper and the participants in the SCC have been supported by the European Community's Seventh Framework Programme [FP7/2007-2013] and Horizon 2020 under the Mont-Blanc projects, grant agreements n. 288777, 610402 and 671697; the HPC Advisory Council; the Facultat d'Informàtica de Barcelona – Universitat Politècnica de Catalunya; Arm Ltd.; Cavium Inc.; E4 Computer Engineering. We warmly thank the pizzeria 7bello in Frankfurt for always having a table and a smile for us.

References

[1] B. Obama, "Executive order-creating a national strategic computing initiative," *The White House, US*, vol. 29, 2015.

[2] E. I. Bank, "Financing the future of supercomputing - how to increase investments in high performance computing in europe," 2018.

[3] C. Connor, A. Bonnie, G. Grider, and A. Jacobson, "Next generation hpc workforce development: the computer system, cluster, and networking summer institute," in *Proceedings of the Workshop on Education for High Performance Computing*. IEEE Press, 2016, pp. 32–39.

[4] Hpc advisory council. [Online]. Available: http://hpcadvisorycouncil.com/

[5] S. L. Harrell, H. A. Nam, V. G. V. Larrea, K. Keville, and D. Kamalic, "Student cluster competition: a multi-disciplinary undergraduate hpc educational tool," in *Proceedings of the Workshop on Education for High-Performance Computing*. ACM, 2015, p. 4.

[6] MIT/BSC Student Cluster Competition 2016, Arm HPC User Group at SC16. [Online]. Available: https://developer.arm.com/hpc/events/arm-hpc-user-group-sc16

[7] S. L. Harrell, P. M. Smith, D. Smith, T. Hoefler, A. A. Labutina, and T. Overmyer, "Methods of creating student cluster competition teams," in *Proceedings of the 2011 TeraGrid Conference: Extreme Digital Discovery*. ACM, 2011, p. 50.

[8] M. E. Baldwin, X. Zhu, P. M. Smith, S. L. Harrell, R. Skeel, and A. Maji, "Scholar: A campus hpc resource to enable computational literacy," in *Education for High-Performance Computing (EduHPC), 2016 Workshop on*. IEEE, 2016, pp. 25–31.

[9] T. Franczak, A. Nkansah, T. Marrinan, and M. E. Papka, "A path from serial execution to hybrid parallelization for learning hpc."

[10] J. Liu, "20 years of teaching parallel processing to computer science seniors," in *Proceedings of the Workshop on Education for High Performance Computing*. IEEE Press, 2016, pp. 7–13.

[11] K. Cahill, S. I. Gordon, and S. Lathrop, "A proposed model for teaching advanced parallel computing and related topics."

[12] E. Saule, "Experiences on teaching parallel and distributed computing for undergraduates," in *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, 2018.

[13] ASC Community, *The Student Supercomputer Challenge Guide: From Supercomputing Competition to the Next HPC Generation*. Springer Singapore, 2018.

[14] Partnership for Advanced Computing in Europe. [Online]. Available: http://www.prace-ri.eu/

[15] Universitat Politècnica de Catalunya, bachelor degree in informatics engineering. [Online]. Available: https://www.fib.upc.edu/en/studies/bachelors-degrees/bachelor-degree-informatics-engineering/curriculum

[16] Università di Bologna, study plan of computer science and engineering. [Online]. Available: https://www.unibo.it/en/teaching/degree-programmes/study-plan/2018/8615/

[17] ETH Zurich, computer science bachelor's program. [Online]. Available: https://www.inf.ethz.ch/studies/bachelor.html

[18] Universität Stuttgart, handbook of computer science for bachelor students. [Online]. Available: http://www.uni-stuttgart.de/bologna/modulhandbuecher/SoSe2018/MHB_82-079-2017-1.pdf

[19] S. K. Prasad, A. Y. Chtchelkanova, S. K. Das, F. Dehne, M. G. Gouda, A. Gupta, J. Jaja, K. Kant, A. La Salle, R. LeBlanc *et al.*, "Nsf/ieee-tcpp curriculum initiative on parallel and distributed computing: core topics for undergraduates," in *SIGCSE*, vol. 11, 2011, pp. 617–618.

[20] T. Sterling, M. Anderson, and M. Brodowicz, *High Performance Computing: Modern Systems and Practices*. Morgan Kaufmann, 2017.

[21] S. Ashby, P. Beckman, J. Chen, P. Colella, B. Collins, D. Crawford, J. Dongarra, D. Kothe, R. Lusk, P. Messina *et al.*, "The opportunities and challenges of exascale computing," *Summary Report of the Advanced Scientific Computing Advisory Committee (ASCAC) Subcommittee*, pp. 1–77, 2010.

[22] W. C. Feng and K. Cameron, "The Green500 list: Encouraging sustainable supercomputing," *Computer*, vol. 40, no. 12, 2007.

[23] The Green500 list. [Online]. Available: https://www.top500.org/green500

[24] Tsinghua powers through isc18 field. [Online]. Available: https://www.hpcwire.com/2018/07/10/tsinghua-powers-through-isc18-field/

[25] N. Rajovic, A. Rico, F. Mantovani, D. Ruiz, J. O. Vilarrubi, C. Gomez, L. Backes, D. Nieto, H. Servat, X. Martorell *et al.*, "The mont-blanc prototype: an alternative approach for hpc systems," in *High Performance Computing, Networking, Storage and Analysis, SC16: International Conference for*. IEEE, 2016, pp. 444–455.

[26] F. F. Banchelli Gracia, D. Ruiz Muñoz, Y. Hao Xu Lin, and F. Mantovani, "Is arm software ecosystem ready for hpc?" 2017, Student poster presented at SC17: International Conference for High Performance Computing, Networking, Storage and Analysis.

[27] BSC tools. [Online]. Available: https://tools.bsc.es/

[28] V. Pillet, J. Labarta, T. Cortes, and S. Girona, "Paraver: A tool to visualize and analyze parallel code," in *Proceedings of WoTUG-18: transputer and occam developments*, vol. 44, no. 1. IOS Press, 1995, pp. 17–31.

[29] F. Mantovani and E. Calore, "Performance and power analysis of hpc workloads on heterogeneous multi-node clusters," *Journal of Low Power Electronics and Applications*, vol. 8, no. 2, p. 13, 2018.

[30] Z. Jin, M. Krokos, M. Rivi, C. Gheller, K. Dolag, and M. Reinecke, "High-performance astrophysical visualization using splotch," *arXiv preprint arXiv:1004.1302*, 2010.

[31] M. Garcia, J. Labarta, and J. Corbalan, "Hints to improve automatic load balancing with LeWI for hybrid applications," *Journal of Parallel and Distributed Computing*, vol. 74, no. 9, pp. 2781–2794, 2014.

[32] M. Garcia-Gasulla, M. Josep-Fabrego, B. Eguzkitza, and F. Mantovani, "Computational fluid and particle dynamics simulations for respiratory system: Runtime optimization on an arm cluster," in *Proceedings of the 47th International Conference on Parallel Processing Companion*, ser. ICPP '18. New York, NY, USA: ACM, 2018, pp. 11:1–11:8. [Online]. Available: http://doi.acm.org/10.1145/3229710.3229736

[33] European Commission, *ECTS users' guide*. Luxembourg Publications Office of the European Union, 2015.

[34] P. Abrahamsson, S. Helmer, N. Phaphoom, L. Nicolodi, N. Preda, L. Miori, M. Angriman, J. Rikkila, X. Wang, K. Hamily *et al.*, "Affordable and energy-efficient cloud computing clusters: The bolzano raspberry pi cloud cluster experiment," in *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, vol. 2. IEEE, 2013, pp. 170–175.

[35] PyFR mailing list / cPickle has me in a pickle. [Online]. Available: https://groups.google.com/forum/#!topic/pyfrmailinglist/UYBMh6Uf-qI

[36] E. Calore, F. Mantovani, and D. Ruiz, "Advanced performance analysis of HPC workloads on Cavium ThunderX," in *International Conference on High Performance Computing & Simulation (HPCS 2018)*. IEEE, 2018, in press.

[37] E4 Computer Engineering and Cavium join forces to support Universitat Politècnica de Catalunya team at ISC 2018 student cluster competition. [Online]. Available: https://www.hpcwire.com/off-the-wire/e4-computer-engineering-and-cavium

[38] Cavium announces ThunderX2 general availability. [Online]. Available: https://www.cavium.com/news/cavium-announces-thunderx2-general-availability

[39] tf_cnn_benchmarks.py does not support –data_dir with my imagenet1k tfrecords. [Online]. Available: https://github.com/tensorflow/benchmarks/issues/197