Creating a Sustainable High-Performance Scientific Computing Course

E. R. Jessup[1] and H. M. Tufo[1]

[1] University of Colorado, Boulder, CO 80309, USA,
jessup@cs.colorado.edu,
WWW home page: http://www.cs.colorado.edu/~jessup
[2] University of Colorado, Boulder, CO 80309, USA,
tufo@cs.colorado.edu,
WWW home page: http://www.cs.colorado.edu/~tufo

Abstract. We describe our experiences with computational science and engineering (CS&E) education at the University of Colorado at Boulder in order to illustrate the difficulties associated with such a course of study. Our CS&E offerings began with a course on high-performance scientific computing (HPSC) developed under a CISE Educational Infrastructure grant from the National Science Foundation. The course was taught from 1991 through 1998 when various practical concerns ended its run. The course was revived in 2003 to meet the demands of several emerging programs in computational science. In this paper, we outline the rise, fall, and restoration of the HPSC course. We identify the technological developments that presently make such a course less labor intensive and more sustainable.

1 Introduction

The Department of Computer Science at the University of Colorado at Boulder (UCB) was an early player in the world of computational science and engineering (CS&E) education for undergraduates. Its course in high-performance scientific computing (HPSC) was introduced in 1991. The course provided an introduction to the use of high-performance computing systems in scientific and engineering applications. Its development was supported by the National Science Foundation under a CISE Educational Infrastructure grant awarded in 1990. The course enjoyed a successful seven year run before succumbing to the pressures of technological change and decline of student interest. In this paper, we describe our experiences with the HPSC course at the CUB. In section 2, we discuss the design and development of the original HPSC course. Analysis of its decline is provided in section 3. In section 4, we discuss the reasons for the rebirth of HPSC. Finally, in section 5 we discuss the changes in technology and local CS&E curricula that will now allow us to offer the course over the long term.

2 The Rise of HPSC

The original HPSC course was offered as a two semester sequence each academic year from 1991-2 through 1997-8. The course was based on the philosophy that students were best turned into effective users of supercomputers by introducing them to practical use of the machines. That is, students learned through real application problems and not just toy problems or simple numerical examples. The students learned which architectures were appropriate for which problems and how best to map the problems onto those machines. The students also learned how to assess the performance of their programs, including how to interpret what the compiler did and how to use the clock properly. Finally, the students learned to use the color, perspective, and animation capabilities of suitable visualization tools to display and interpret the large amounts of data typically generated by supercomputer programs.

Over the years, students in the course used DEC and SGI workstations and an Intel iPSC/2 hypercube multiprocessor located on the UCB campus as well as a variety of supercomputers at remote sites. The latter, which were accessed via the Internet, included a Cray Y-MP located at the National Center for Atmospheric Research (NCAR), an Intel Paragon at the National Oceanic and Atmospheric Administration (NOAA), and a Thinking Machines CM-2 at the National Center for Supercomputing Applications among others. Time on those machines was donated by the centers.

The first semester began with introductions to three different machines and to performance measurement. Students also studied the computational and scientific visualization tools Matlab [1] and an AVS product [2]. The concepts and tools learned in the first few weeks were then applied to solution of numerical problems in molecular dynamics and computerized tomography. A midterm project required evaluation of the various architectures for solving several specified numerical problems. A final project required the student to port a molecular dynamics code to a new architecture and to evaluate the results. The second semester covered one or more additional architectures and an application in advection. In that semester, students completed a final project on an advanced topic of their choosing. Both semesters were offered as three hour lecture courses supplemented by a three hour supervised lab.

Because no appropriate textbook was available, we wrote all of the course materials. Our series of single-topic tutorials and short reference guides eventually became a textbook [3]. A laboratory manual and manuals for the machines and computational tools are still available on the Web [4].

The only prerequisite for the course was one semester of numerical computation at the undergraduate level. During its tenure, the course, which was listed at both the graduate and undergraduate levels, attracted juniors, seniors, and first year graduate students majoring in computer science, physics, applied mathematics, and chemical, mechanical, or aerospace engineering.

The HPSC course was a success on several levels. The student evaluations were routinely positive. Several of our graduates went on to jobs at national laboratories (mainly NCAR and NOAA in Boulder) or into graduate programs relevant to computational science or engineering. They reported that the material they'd learned in the course was helping them in their jobs or studies. Other graduates became research assistants at UCB in the field of HPSC and ultimately took on employment in that area.

Our course was exported to other colleges and universities. As part of our proposal to NSF, we named several collaborating institutions. Professors from those institutions aided us in course development and received equipment purchased with the NSF funds to help support their own HPSC courses. The collaborators were chosen from institutions in our region with some emphasis on schools with significant minority enrollment and on schools that would not otherwise have the resources necessary for teaching HPSC. The ultimate goal of the collaboration was to put HPSC courses modeled on ours into those institutions, and that goal was reached at all of the schools.

For two years, we offered (separate) two-week summer workshops for students and for faculty from other institutions during which we covered the first semester's materials. The faculty members attending the workshops were subsequently able implement scientific computing courses like ours at their own institutions.

We were also aware of courses based fully on ours offered at other universities not associated with our program. Instructors at other institutions were using a sampling of our materials in related courses. Interest was demonstrated by the large number of downloads from our course materials Web site and continuing textbook sales.

The course received formal recognition in the form of a 1995 Undergraduate Computational Science Education Award from the Department of Energy.

3 The Decline of HPSC

Although the HPSC course was very well-received, it was not without its problems. While we were able to use a network of workstations via MPI [5] in the last two offerings of the course, MPI was not fully developed, and it was not yet considered standard. As a result, codes had to be ported explicitly to each new architecture, using constructs particular to that machine. Furthermore, supercomputers came and went at a rapid rate. Sometimes a computer we used in one offering of the course was not available for the next year's offering. As a result, the process of keeping the course materials up to date was difficult and time-consuming.

While the topics covered by the original materials served us well at UCB, they were not always perfect for courses at other institutions. Expanding the offerings represented another substantial investment of effort, and it was not obvious how to obtain funding for further course development. While many funding agencies have excellent programs for course initiation, they do not have programs for course continuation and maintenance.

While the UCB model of HPSC education was appropriate at many other schools, it did not work everywhere. In those days before ubiquitous Internet, smaller schools complained of the lack of fast Net access and of the dearth of funding to purchase and support workstations and visualization tools (especially the AVS program we used for 3D animation.) We considered the problem of downsizing the materials for use at those schools but again ran into lack of support for such efforts.

Perhaps the most serious problem confronting the HPSC course was a local one. The extremely tight curriculum in the Engineering College (which includes the Department of Computer Science) made it difficult for students to find time for the HPSC elective. Further, the number of Computer Science majors interested in HPSC was relatively small, and students in other departments were not always aware that the course existed. As a result, first semester class enrollments ranged from only 11 to 17 over the years (small numbers at our university.) The second semester class was typically even smaller. In some semesters, finding students at all required a significant amount of recruiting. Before the 1998 offering, the instructor opted not to recruit, no students enrolled, and the course was canceled. That semester marked the end of the original version of the HPSC course. Its run had lasted seven years.

The decline of HPSC at UCB mirrored a general lessening of interest in the topic in the educational community as a whole. The years 1992-1997 saw a large number of special conferences and conference sessions devoted to HPSC education. That time period also saw large-scale development of materials for computational science by other authors (e.g., DOE's Undergraduate Computational Engineering and Sciences Project [6].) Such activity decreased markedly in subsequent years, reviving only recently.

4 The Rebirth of HPSC

The beginning of the twenty first century has brought with it an increased demand for computational skills in a variety of disciplines. The source of that demand extends from academic researchers to employers in industry and the national laboratories. As a result, the seeds of CS&E education have spread to numerous engineering and applied science departments at UCB and have gradually developed into small pockets of CS&E training within those departments. In an effort to formalize this training, several departments have begun to construct undergraduate and graduate educational programs to teach their students the specific CS&E skills required by their disciplines. Because of the difficulty of creating new degree programs at UCB, all of these training programs have been offered as supplements to or tracks within current degree offerings.

The material and the practical skills once taught in our HPSC course are at the core of many of these new CS&E programs. One of the first approved programs came from the Department of Applied Mathematics in 2003. Instead of developing their own HPSC course or spreading the material over several core courses, the Applied Mathematics faculty asked us to revive HPSC. To address their discipline-specific content concerns we moved the open course project from the second semester to the first and allowed significant flexibility in project selection.

To address technological developments since its first offering, we embarked on a redesign of the course. First, several structural changes were effected. To further increase the potential pool of students the numerical analysis prerequisite was removed and now appears only on the recommended course list. Since the project had been moved to the first semester we chose to make HPSC a one semester course instead of two. However, as our intention was that it remain a hands-on project-based course, we kept the original four credit hour design (three hours of lecture and three hours of supervised lab per week.)

In addition to structural changes, rapid changes in technology and demands from employers in industry and the national laboratories needed to be reflected in the course content and tools. Parallel programming skills are currently in high demand. As MPI is now the de facto standard for writing message-passing-based parallel programs and can be either be easily installed on or purchased with a computing system, we concentrate on learning MPI programming in the first eight weeks of the sixteen-week course. Given that mixed model programming paradigms are being driven by the use of shared memory multiprocessing technology for machine building blocks, we then spend time learning the basics of OpenMP [7] and writing hybrid MPI/OpenMP programs. Because the recent success of the Japanese Earth Simulator program has sparked renewed interest in vector programming, we also provide a brief introduction to vector computing. The majority of the remaining time is spent examining parallel architectures and algorithm development in more detail, concentrating on mesh, tree, and hypercube architectures and developing architecture specific algorithms for commonly encountered problems (e.g., sorting, matrix and graph algorithms, FFT.)

5 The Sustainability of HPSC

The original HPSC course fell victim to a variety of problems: the rapid evolution of computer architectures, the difficulty of maintaining course materials, and the decline of local demand were the most serious.

That course relied on a combination of local and remote computing resources. The fact that many of the parallel systems were housed off-site and changed rapidly made keeping the course materials current and preparing for next year's offering difficult. The intervening five years (1998-2003) saw the advent of Beowulf cluster computing systems [8]. Because these systems are built using commodity off-the-shelf components and employ open source software, they are extremely cost effective, providing computing capacity at less then $300 per GigaFlop (e.g., [9]) and requiring no (expensive) service contracts. (To put this in perspective, in 1998 the world's fastest computer, ASCI Option Red, cost about $30,000 per GigaFlop [10].) Given the reduced cost and the fact that the open source software, in particular the Linux operating system, has become relatively mature and sufficiently stable, we purchased a 128-processor Beowulf cluster and the HPSC course relies solely on this platform for code development *and* production runs. Making long term predictions about the the future of high-performance computing is extremely difficult. However, we firmly believe that these cluster solutions will be around for next decade and remain the most cost effective solution for designing systems of less than a thousand processors.

The maturity and stability of the software and hardware environment have not only made life much easier for the instructor but also for the students. The fact that most students are already familiar with Linux and, hence, our computing environment, means that we were able to accelerate the rate at which we covered material. This accelerated rate allows us essentially to present all of the computational material previously taught in two semesters in one, thus removing any objections to the decision to make this a one semester course.

In the interest of providing a course appropriate to a variety of disciplines, the physics-based application problems are no longer covered. Scientific visualization also receives less coverage primarily due to time constraints, although freeware visualization (e.g., VTK [11]) is available on the cluster. The original intensive slate of programming assignments and laboratory exercises now provides an effective method for rapidly introducing and learning basic and intermediate MPI programming techniques.

The maintenance of course materials is no longer the problem it once was. While many of the original course materials are now dated and their emphasis on applications does not fit the current model of instruction, there is no longer a need to produce homemade materials to replace them. In the current course, we use two books on MPI [12, 13] and one on parallel algorithms and architectures [14]. A variety of other sources exist that could be used to supplement or replace these materials should the need arise.

Like its predecessor, the new version of the HPSC course is appropriate for export to other institutions. In particular, the availability of the original on-line materials (updated and possibly supplemented with on-line lectures) and of well-developed open source software (e.g., Linux [15], GNU [16], MPICH [17], OpenPBS [18]) and repositories (e.g., SourceForge [19]) make it possible to "package" the entire course in an affordable box. Furthermore, the availability of less expensive systems and free software make an HPSC course more accessible to smaller institutions. Even two networked workstations running MPI are sufficient to illustrate many important concepts in parallel computing.

Finally, we expect that the now broader interest in CS&E at UCB will help us to maintain healthier enrollments. We have requested that Applied Mathematics feed junior and senior level undergraduates and graduate students from the undergraduate and graduate numerical analysis courses and other appropriate courses into the new HPSC course. Although the partnership is still in its infancy, we anticipate that Applied Mathematics will supply a steady stream of 8-16 students per year. Additional demand is ultimately expected from emerging CS&E programs in Astrophysical and Planetary Sciences, Aerospace Engineering, Atmospheric and Oceanic Sciences, Physics, Civil and Environmental Engineering, and Computer Science. To increase the appeal to Computer Science graduate students, we have ensured that the course serves to satisfy the breadth requirement for the master's degree.

References

1. : The MathWorks: MATLAB and Simulink for Technical Computing (2003) http://www.mathworks.com/.
2. : AVS/Advanced Visual Systems: Data visualization software and solutions (2003) http://www.avs.com/.
3. Fosdick, L.D., Jessup, E.R., Schauble, C.J.C., Domik, G.: An Introduction to High-Performance Scientific Computing. MIT Press, Cambridge, MA (1996)
4. Fosdick, L.D., Jessup, E.R., Schauble, C.J.C., Domik, G.: Computer Science – Course Materials, CSCI 4576 (2003) http://www.cs.colorado.edu/ftp/pub/HPSC/README.html/.
5. : MPI The message Passing Interface Standard (2003) http://www.mcs.anl.gov/mpi/.
6. : The Undergraduate Computational Engineering and Sciences Project Homepage (2003) http://www.krellinst.org/UCES/.
7. : OpenMP: Simple, Portable, Scalable SMP Programming (2003) http://www.opemmp.org/.
8. : Beowulf.Org The Beowulf Clustering Site (2003) http://www.beowulf.org/.
9. : Terascale Cluster - Research Computing - computing.vt.edu (2003) http://computing.vt.edu/research_computing/terascale/.
10. : ASCI Red Web Site (2003) http://www.sandia.gov/ASCI/Red/.
11. : VTK Home Page (2003) http://www.vtk.org/.
12. Pacheco, P.: Parallel Programming with MPI. Morgan Kaufmann, San Francisco, CA (1997)
13. Gropp, W., Lusk, E., Skjellum, A.: Using MPI: Portable Parallel Programming with Message-Passing Interface. MIT Press, Cambridge, MA (1999)
14. Leighton, F.T.: Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes. Morgan Kaufmann, San Francisco, CA (1991)
15. : The Linux Home Page at Linux Online (2003) http://www.linux.org/.
16. : GNU's Not Unix! - the GNU Project and the Free Software Foundation (FSF) (2003) http://www.gnu.org/.
17. : MPICH - A Portable MPI Implementation (2003) http://www.mcs.anl.gov/mpi/mpich.
18. : OpenPBS (2003) http://www.openpbs.org/.
19. : SourceForge.net (2003) http://sourceforge.net/.