Enhancement of Computational Engineering within an Undergraduate Mechanical Engineering Curriculum

Robert Spall, Thomas Hauser
Utah State University

Abstract

The NSF supported Course, Curriculum and Laboratory Initiative (CCLI) project described herein addresses concerns regarding undergraduate education at research universities as highlighted in the 1998 Boyer Commission Report[1] by incorporating advances in information technology into the curriculum. This has been accomplished by developing an "emphasis" to the department's undergraduate mechanical engineering degree in the area of computational engineering. To complete the emphasis, students need to complete four upper division elective courses related to computational engineering. Three courses concentrate on applied modeling and simulation; the fourth (which was developed under the CCLI award) concentrates on implementing algorithms on parallel computing architectures.

To support the emphasis, the authors have designed and assembled a PC Beowulf teaching cluster. The cluster consists of a server node where students can log in and develop their programs, as well 10 dual Opteron compute nodes for running and testing parallel codes. AMD Opteron CPUs were selected for the cluster since they may be used to teach both shared and distributed memory programming techniques, and for their strong price/performance ratio. Using gigabit networking technology, the cluster was built at a cost of approximately $15k.

In addition to traditional coursework, a cluster computing workshop was developed and offered for the first time during the summer of 2004. Key issues covered were designing a Beowulf cluster, implementing and programming a cluster, and tuning/profiling of programs.

1. Introduction

In 1998, the Boyer Commission released a report on educating undergraduates in research universities titled "Reinventing Undergraduate Education: A Blueprint for America's Research Universities" [1]. The report points out that while only 3% of all higher education institutions are classified as Carnegie I and II research universities (since reclassified as Doctoral/Research University Extensive), these universities confer 32% of all undergraduate degrees. Furthermore, these universities also confer 56% of all baccalaureate degrees earned by recent recipients of doctorates in science and engineering. (Utah State University is classified as a Doctoral/Research University Extensive.)

However, the Commission also points out that the research universities have sometimes failed their undergraduate student populations. The Commission argued that what is needed is a new model of undergraduate education that makes the baccalaureate experience “an inseparable part of the integrated whole.” Toward that end, the Commission made a series of recommendations that would enable undergraduates to benefit from the unique opportunities and resources available at research universities. Many of the recommendations of the Boyer Commission turned out to be controversial, and others will likely occur only through actions of the university administration. However, many positive changes in undergraduate education must essentially take place through initiatives developed by groups of faculty members, typically at the department level. This work addresses one of what we consider to be the more important of the recommendations of the Boyer Commission in the context of our Mechanical and Aerospace Engineering Department – the use of information technology in education. This recommendation from the Boyer Commission is also consistent with recent emphasis from the National Science Foundation in the broad areas that encompass information technology research and education.

In particular, we address advances in science and engineering by introducing students to the field of computational engineering, a subset of information technology. With the emergence of low-cost, parallel supercomputers such as PC-based Beowulf clusters, a new era in computational engineering has begun. High performance computing, previously limited to government laboratories and research universities, is now available to the masses. This exponential increase in the number of potential users of high performance computing has the potential to dramatically impact the rate of increase in discovery over a wide range of engineering fields, including fluid and solid mechanics, engineering design, electromagnetics, and emerging technologies such as nano-scale engineered systems. To date however, significant focus on computational engineering within engineering curricula has been limited primarily to graduate level programs. The authors believe that there is no pedagogical reason for this limitation, and that we should be introducing this technology at the undergraduate level. This implementation would increase the number of graduates with backgrounds in this emerging area of information technology, students who may choose to enter industry directly or continue their education at the graduate level.

To summarize, the primary goal of this NSF sponsored Course, Curriculum, and Laboratory Initiative (CCLI) project is to develop an emphasis in computational engineering for our undergraduate mechanical engineering degree, and to develop the necessary computational infrastructure to support the emphasis. This provides an educational framework to better prepare students for careers in which they advance scientific discovery though high performance computing.

2. Project Plan

Requirements for Modeling and Simulation Emphasis

Currently, students graduating with a Bachelor of Science degree in mechanical engineering at Utah State University are required to complete 5 three-credit-hour technical electives, which are

typically taken during the senior year. Students choosing the computational engineering emphasis must complete the following four electives:
1) MAE 5020, Finite Element Methods in Solid Mechanics
2) MAE 5440, Computational Fluid Dynamics (CFD)
3) Math 5620, Numerical Solutions of Differential Equations
4) Engr 5500, High Performance Computing

The first two courses listed concentrate on applied modeling and simulation, the third provides a theoretical introduction to numerical solutions of ordinary and partial differential equations. The fourth course emphasizes both hardware and software requirements for high performance computing, and was developed specifically for the computational engineering emphasis. The above courses are in addition to the computer-oriented courses required of all mechanical engineering students: Engineering Graphics (MAE 1200), Numerical Methods I (MAE 2200, Fortran 95), and Numerical Methods II (MAE 2200, traditional topics in numerical methods).

We discuss below modifications to several of the courses in our curriculum that were made in order to accommodate the requirements of the computational engineering emphasis.

MAE 2200 Numerical Methods I

The Numerical Methods I course represents our primary programming course in which mechanical and aerospace engineering students are taught the fundamentals of structured programming using Fortran 95. This is a two credit hour course with an associated weekly programming lab session. Students following the "standard curriculum schedule" take this course in the fall term of their junior year. We have extended our treatment of Fortran 95 to include an introduction to High Performance Fortran (HPF), which is an extension to Fortran 95 that allows efficient code to be more easily written for a distributed memory parallel system. When creating parallel applications in HPF, coding is done using data distribution directives combined with FORALL and standard Fortran 95 array expressions, whereas directly coding message-passing calls is more like assembly-level programming. Consequently, HPF programs are much easier to maintain and understand than programs based on other models available for distributed memory parallel systems, and should be suitable for upper division undergraduate students. To make room for the new material, we no longer teach certain advanced features of Fortran 95 procedures and modules.

MAE 2210 Numerical Methods II

The Numerical Methods II course involves software development by students programming in Fortran 95. This approach has been quite successful in preparing students for more advanced courses where strong programming skills are a necessary prerequisite, such as CFD. Several of the required projects in this course will be modified for the spring 2005 semester so that students will utilize HPF extensions in developing their codes. Again, the projects are formulated so that only an introductory knowledge of HPF is needed to exploit the parallel environment. An example is the solution of Laplace's equation using Jacobi iteration. We note that approximately 75 students per year complete both the Numerical Methods I and II courses.

MAE 5440 Computational Fluid Dynamics

Recognizing that CFD has become sufficiently mature to act as a design tool in a wide range of industries, the MAE department has introduced a course in the subject at the undergraduate level[2, 3, 4]. We have achieved considerable success by teaching much of the class within a laboratory environment. In particular, we alternate between the traditional lecture format and a "hands-on" code development period where students, under the direction of the course instructor, develop working codes based on the lecture material. Under this format, individual student projects are assigned at approximately 3 week intervals that cover: 1) Finite volume solutions for diffusion-only phenomena (i.e., 2-dimensional heat conduction), 2) Finite volume solutions for convection only (i.e., 2-dimensional convection of a discontinuity in a scalar field), 3) Solution of a 2-dimensional convection/diffusion problem, 4) Solution of the 2-dimensional Navier-Stokes equations using a pressure-based finite-volume method. Required problems solved include the driven cavity and developing flow through a channel. This course will be modified during the spring 2005 semester to include instruction in the use of HPF to parallelize the four individual student projects listed above.

Engr 5500 High Performance Computing

This course, first taught during the fall 2004 semester, introduces students to high performance computing on modern microprocessor architectures and covers the following topics: Parallel Architectures, Designing parallel algorithms, Shared-Memory Programming using OpenMP, Performance Analysis and Tuning, Message-Passing Programming, Parallel Matrix-Vector multiplication, Parallel Monte Carlo Methods, Solving Linear Systems, and Finite Difference Methods. Similar to the Computational Fluid Dynamics class, students develop parallel applications in the labs, and are introduced to debugging and profiling of their parallel programs using the MAE department clusters. One difficulty encountered is that there is no textbook covering the material using Fortran 95. We use two textbooks[5, 6] that discuss programming in C; the instructor then converted all examples to Fortran 95 and explained the differences between C and Fortran 95 to the students.

Beowulf Cluster Design

We believe a primary factor limiting the integration of clusters into academia and industry is a general lack of knowledge of the operation and capabilities of these systems. Consequently, one of the goals of the computational engineering emphasis is to produce engineers with not only a theoretical background, but also with practical, hands-on computing capabilities that they may take with them to future places of employment, be that graduate school or industry. Toward that end, we offer our students the opportunity to design and maintain a working cluster. This is done during a workshop (described below) on the design, maintenance and programming of low-cost, high performance, Beowulf clusters.

The cluster developed with support from the CCLI project consists of one server node where students log in and develop their programs, as well as 10 dual processor compute nodes for running and testing parallel codes. We have used AMD Opteron CPUs for the cluster due to their exceptional price/performance ratio relative to Intel Pentium processors. The network uses Gigabit Ethernet since the motherboards have two integrated Gigabit network cards. The server node has 4 GB of main memory and 160 GB of disk space, of which 90 GB are used for home directories.

The compute nodes contain 2 GB of main memory and 20 GB disk space. The cluster uses Mandrake Linux as the operating system and is build and maintained using the Warewulf Cluster Toolkit (http://warewulf-cluster.org). Each compute node is managed from the server and it does not make a difference if a small or large cluster is maintained—the procedure is identical. Once one node is booting correctly all additional nodes will work in exactly the same manner. PGI compilers are installed to provide students with HPF, OpenMP and Fortran 95. Several versions of MPI, compiled with different compilers, are also available for more advanced users. These different versions, as well as additional commercial software such as FLUENT, are managed through env-switcher and modules (env-switcher.sourceforge.net).

In addition to being a valuable learning experience, we note that building a system from components is much more cost effective than purchasing a turn-key system from commercial vendors. The above system was constructed for approximately $15k.

Cluster Workshop

Students receive academic credit for the workshop which runs 8 hours per day for a one week period during the summer. Key issues involved in selecting or engineering a PC cluster supercomputer, along with a simple methodology for the design of a system optimized for specific application(s), are covered. To broaden the impact of this project, students from other areas of science, technology, engineering and mathematics (STEM) at USU are also invited to attend the workshop. As a result, for the summer 2004 workshop half of the participants were computer science students, the other half mechanical engineering students. For the hands on portion of the workshop, interdisciplinary teams were created, allowing the students to learn from each others experience. Some of the engineering students brought their own research code for the 4th and 5th day of the workshop which allowed the computer science majors to profile engineering research programs. An outline of the topics covered appears below.

Day 1. Overview of current Beowulf type cluster computers; capabilities, restrictions, requirements; Selecting components; microprocessors; motherboards; cases; networking components; current microprocessor architecture, and influence on cluster performance.
Day 2. Engineering a Beowulf cluster.
Day 3. Install your own cluster using Warewulf.
Day 4. Understanding your application. Profiling your application; tuning for the memory hierarchy; benchmarking a cluster.
Day 5. Case studies – tune your own code.

3. Initial Assessment Results

The initial implementations of parallel computing into our curriculum took place within the Numerical Methods I course during the fall 2004 semester. A survey consisting of 12 questions was administered to 74 students during the last week of the course. Results of the survey are tabulated below. Note that the scale for the responses ranges from 1 (strongly disagree with statement) to 6 (strongly agree with statement). The first 6 questions were aimed at determining the students perceptions regarding the importance of computing and parallel programming for mechanical engineers. The purpose of questions 7-12 was to determine the students level of confidence regarding programming in Fortran 95 and using HPF extensions.

Results from questions 1-6 indicate the students feel numerical simulation and programming are important tools for mechanical engineers. Although students felt learning Fortran 95 was valuable, results concerning parallel programming were mixed.

Results from questions 7-12 reveal that most students were comfortable using Fortran 95 and were able to understand basic programming concepts. However, mirroring the responses to questions 1-6, many students were less confident about their abilities to implement programs in parallel using HPF.

The majority of the students completing Numerical Methods I in the fall semester will enroll in Numerical Methods II for the spring semester. Given the results of the survey, it will be important to further develop the students programming skills using HPF. We intend to concentrate our applications using HPF toward the final third of the course, where iterative solutions to partial differential equations are studied.

Table 1: Numerical Methods I Survey Questions and Responses

Question	Strongly disagree 1	2	3	4	5	Strongly agree 6	Average
1. I now understand that simulation is an important part of mechanical engineering	0	1	8	11	31	23	4.9
2. Mechanical engineers do not need to know how to program	27	32	10	4	1	0	2.5
3. Programming is essential for my future as an engineer	2	4	14	21	16	17	4.3
4. Learning Fortran 95 is a waste of time	30	27	10	6	1	0	1.9
5. Parallel programming is only for computer scientists	12	23	14	18	5	2	2.8
6. I believe all engineers should learn about parallel programming	1	8	22	19	13	11	3.9
7. Fortran 95 is an easy language to learn	1	10	15	22	22	4	3.9
8. Fortran 95 arrays are easy to use	0	1	10	16	33	14	4.7
9. I had trouble understanding the programming concepts	9	33	13	9	7	3	2.7
10. Parallel programming using High Performance Fortran is easy to understand	7	20	26	14	7	0	2.9

Table 1: Numerical Methods I Survey Questions and Responses

Question	Strongly disagree 1	2	3	4	5	Strongly agree 6	Average
11. It was difficult to parallelize my program using HPF	1	26	18	10	12	6	3.3
12. Data distribution in HPF was easy to understand	4	18	24	17	8	3	3.2

4. Summary

With support form a National Science Foundation Course, Curriculum and Laboratory Initiative grant, the Mechanical and Aerospace Engineering Department at Utah State University has undertaken an initiative to increase the competence of its graduates in the area of computational engineering. Toward that end, an optional, formal emphasis in computational engineering is being added to the department's undergraduate mechanical engineering degree. Initial graduates from the program are expected in May 2006. To graduate with the emphasis, students must complete 4 elective courses pertaining to high performance computing. To support the degree emphasis, the department built and operates a computing cluster based on dual processor AMD Opteron technology. Initial implementation of the new material into the curriculum took place during the fall 2004 semester in our introductory programming class. Survey results showed that at this introductory level, we were successful in showing students the importance of numerical simulation and programming in the field of mechanical engineering. However, we found mixed results in terms of our students perceptions regarding their ability to understand and implement programming in a parallel environment. These same students enroll in Numerical Methods II for the spring 2005 semester, and it is clear to the authors that additional exercises which involve programming using HPF (in addition to standard Fortran90) will be necessary.

5. Acknowledgements

This project was funded through National Science Foundation Course, Curriculum and Laboratory Improvement A&I award 0310697.

6. References

1. Boyer Commission, "Reinventing Undergraduate Education: A Blueprint for America's Research Universities," (1998).

2. Hailey, C.E., Spall, R.E. and Snyder, D.O. "Computational Fluid Dynamics Presented in the Undergraduate Curriculum," *Computers in Education Journal*, Vol. 11, pp. 2-8, 2001.

3. Hailey, C.E. and Spall, R.E. "An Introduction of CFD into the Undergraduate Engineering Program," American Society for Engineering Educators (ASEE) Annual Conference, St. Louis, MO, June 18-21, 2000.

4. Spall, R.E., Hailey, C.E., and Snyder, D.O. "Development of a Windows-based CFD Code for Use in Undergraduate Education," 2000 American Society for Mechanical Engineers (ASME) Fluids Engineering Summer Conference, Forum in Advances in Fluids Engineering Education, Boston, MA, June 11-15, 2000.

5. Quinn, M.J., "Parallel Programming in C with MPI and OpenMP," McGraw-Hill, June 2003.

6. Foster, I., "Designing and Building Parallel Programs," Addison-Wesley, 1995.

Author Biographies

Robert Spall is a Professor of Mechanical and Aerospace Engineering. His research interests are in the area of applied computational fluid dynamics and heat transfer. Past work has involved swirling flows and vortex breakdown, buoyancy driven flows, aerodynamic flows, environmental flows, turbulence modeling, and algorithm development. He has also been active in developing computational fluid dynamics algorithms for use in engineering education.

Thomas Hauser is an Assistant Professor of Mechanical and Aerospace Engineering. His research interests are in the areas of computational fluid dynamics, parallel processing and large-eddy simulation. He recently received an Honorable Mention Gordon Bell Prize in the Price/Performance category.