An Undergraduate Computational Science Curriculum

Angela B. Shiflet and George W. Shiflet

Wofford College Spartanburg, South Carolina, USA
{shifletab, shifletgw}@wofford.edu,
http://www.wofford.edu/ecs/

Abstract. Wofford College instituted one of the first undergraduate programs in computational science, the Emphasis in Computational Science (ECS). Besides programming, data structures, and calculus, ECS students take two computational science courses (Modeling and Simulation for the Sciences, Data and Visualization) and complete a summer internship involving computation in the sciences. Materials written for the modeling and simulation course and developed with funding from National Science Foundation served as a basis the first textbook designed specifically for an introductory course in the computational science and engineering curriculum. The successful ECS has attracted a higher percentage of females than in most computer science curricula. The SIAM Working Group on Undergraduate Computational Science and Engineering Education summarized features of Wofford's ECS and other computational science programs. Besides its established curriculum, Wofford has incorporated computational science in other courses, such as in a sequence of three microbiology laboratories on modeling the spread of disease.

Keywords: computational science, education, modeling, simulation, undergraduate, internships, females.

1 Introduction

Much scientific investigation now involves computing as well as theory and experimentation. Computing can often stimulate the insight and understanding that theory and experiment alone cannot achieve. With computers, scientists can study problems that previously would have been too difficult, time consuming, or hazardous; and, virtually instantaneously, they can share their data and results with scientists around the world.

The increasing speed and memory of computers, the emergence of distributed processing, the explosion of information available through the World Wide Web, the maturing of the area of scientific visualization, and the availability of reasonably priced computational tools all contribute to the increasing importance of computation to scientists and of computational science in education.

With funding from the National Science Foundation (NSF Grant No. 0087979), Wofford College developed one of the first undergraduate programs in computational science, an Emphasis in Computational Science (ECS), which the college's faculty unanimously approved in 1998 [1]. One highly successful feature of the ECS is the requirement of a summer internship involving computation in the sciences. With its

M. Bubak et al. (Eds.): ICCS 2008, Part II, LNCS 5102, pp. 697–705, 2008.
© Springer-Verlag Berlin Heidelberg 2008

emphasis on applications, the program has attracted women to a much higher percentage than for the average computer science major.

1.1 Wofford College's Emphasis in Computational Science (ECS)

Wofford College is a selective residential undergraduate liberal arts institution of 1350 students, where the sciences are particularly strong [2]. The SAT range (mid 50 % of the class) for freshman entering in the fall of 2007 was 1140-1340. Approximately one-third of the students major in science, and about two-thirds of those majoring in science and mathematics attend postgraduate or professional schools.

For a year in 1997-1998, faculty members in biology, chemistry, mathematics, physics, psychology, and computer science met to discuss how better to prepare students to use computing in the sciences. The following general needs were identified:

- A balanced program for interested and qualified science majors through which they can expand their knowledge of and skills in computational science
- Increased opportunities for students to obtain internships, graduate work, and jobs in computational science
- Ready access for science and computer science students to modern computational software, such as scientific visualization tools and graphical computer algebra systems
- Familiarity for all computer science majors and many science majors with distributed processing and the UNIX environment because of their extensive use in the sciences

In response to these needs, with consultation from scientists at various laboratories, and with assistance from Dr. Bob Panoff and the Shodor Educational Foundation [3], the faculty developed the Emphasis in Computational Science (ECS), which has the following requirements:

- Complete a Bachelor of Science in lab science or Mathematics, Physics, or Psychology
- Complete five courses: Programming and Problem Solving, Data Structures, Calculus I, and two computational science courses: Modeling and Simulation for the Sciences (COSC/MATH 175) and Data and Visualization (COSC 370)
- Complete a summer internship involving computation in the sciences

1.2 Modeling and Simulation for the Sciences (COSC/MATH 175)

Prerequisites for Modeling and Simulation for the Sciences (COSC/MATH 175), which does not require computer programming experience, are minimal. The course uses the concept of rate of change, or derivative, from a first course in calculus throughout, but students do not need to know derivative formulas to understand the material or develop the models. With a brief introduction, all students who have taken COSC/MATH 175 without having had calculus have successfully completed the course with above average grades.

Modeling and Simulation for the Sciences prepares the student to understand and utilize fundamental concepts of computational science, the modeling process, computer simulations, and scientific applications. The course considers two major approaches to computational science problems: system dynamics models and cellular automaton simulations.

System dynamics models provide global views of major systems that change with time. For example, one such model considers changes over time in the numbers of predators and prey, such as hawks and squirrels. To develop such models, students employ a systems dynamics tool, such as *STELLA*®, *Vensim*®, or *Berkeley Madonna*®, to create pictorial representations of models, establish relationships, run simulations, and generate graphs and tables of the results. Typical applications include drug dosage, scuba diving and the ideal gas laws, enzyme kinetics, defibrillators and electrical circuits, cardiovascular system, global warming, carbohydrate metabolism, predator-prey, competition, radioactive chains, malaria, and other diseases.

In contrast to system dynamics, cellular automaton simulations provide local views of individuals affecting individuals. The world under consideration consists of a rectangular grid of cells, and each cell has a state that can change with time according to rules. For example, the state of one cell could represent a squirrel and the state of an adjacent cell could correspond to a hawk. One rule could be that, when adjacent, a hawk gets a squirrel with a probability of 25%. Thus, on the average at the next time step, a 25% chance exists that the particular squirrel will be no more. Students employ a computational tool, such as *Mathematica*®, *Maple*®, or *MATLAB*®, to complete simulations, such as Brownian motion, movement of ants, spread of fire, HIV in body, foraging behavior, spread of disease, fish schooling, pit vipers and heat diffusion, and snow-flake solidification.

1.3 Data and Visualization (COSC 370)

Because large Web-accessible databases are becoming prevalent for storing scientific information, Data and Visualization (COSC 370) covers the concepts and development of relational databases. With a prerequisite of the first programming course, currently in the Python programming language, students in the class learn Perl and HTML programming in the UNIX operating system environment. After learning to access and develop databases in MySQL, they create web pages with Perl CGI programs to interface between web pages and scientific databases. Additionally, they study a dynamic programming algorithm for alignment of genomic sequences. Interactive online modules, developed at Wofford with the help of its NSF grant, provide the textbook for this portion of the course [4].

The second half of the course covers scientific visualization. Effective visualization of data helps scientists extract information and communicate results. Thus, students learn fundamental concepts, tools, and algorithms of computer graphics and interactive scientific visualization animations using Steve Cunningham's text on *Computer Graphics: Programming, Problem Solving, and Visual Communication* [5], which has all scientific applications. For example, some of the animations are of DNA and other molecules, diffusion across a membrane, movement of ocean waves, heat diffusion, spread of disease, and Lorenz equations.

1.4 Computational Science Internships

Building on their classroom work, students obtaining the Emphasis in Computational Science have had exciting and meaningful summer internships involving computation in scientific research at such institutions as Los Alamos National Laboratory, the Jet Propulsion Laboratory, Oak Ridge National Laboratory, The Scripps Research Institute, Howard Hughes Medical Institute at the Wadsworth Center, The Shodor Education Foundation, the National Blood Data Resource Center, Greenwood Genetic Center, University of California at San Diego, Virginia Commonwealth University, Clemson University, University of South Carolina, and the Medical University of South Carolina. Examples of some of the projects are simulating the dynamics of the parasite that causes Chagas' disease, developing software for the science operations interface of Mars Rovers, optimizing a program to simulate aspects of heart behavior, developing programs to study the evolution of bacterial genomes, performing a microgravity scaling theory experiment, analyzing the relationship of diet to birth defects, creating an extensible framework for the mathematical manipulation of music, modeling of biochemical pathways involved with cardiovascular disease, performing computer image processing of the ribosome, modeling metabolic pathways of a bacterium for bioremediation, implementing a text mining approach to evaluate terms for ontology development, and analyzing traumatic brain injuries computationally.

After their internships, students have presented their results at Wofford and at conferences, such as The Society for Industrial and Applied Mathematics (SIAM) Annual Conference, the SIAM Computational Science and Engineering Conference, and the Consortium for Computing in Science in Colleges Southeastern Conference.

ECS graduates have attended medical school to become physicians; pursued such graduate degrees as genetics at the University of North Carolina, biotechnology and biomedical engineering at the University of South Carolina, computational physics at the North Carolina A & T University, physics at the University of Tennessee and Oklahoma State University, and computer graphics at Columbia University; and have obtained positions, such as medical researcher at GlaxoSmithKline; researcher at the National Institutes of Health, Oak Ridge National Laboratory, and Vanderbilt Medical School; and computational science educator at the Shodor Foundation.

1.5 Attracting Female, Minority, and Biology Students

A disturbing trend in recent years has been for a much smaller percentage of women to pursue undergraduate degrees in computer science. In 1984, women earned 37% of computer science bachelor's degrees, but they obtained only 28% of such degrees in 2000 [6]. Educational research indicates that on the average women prefer applications of computer science and teamwork to such areas as game development working individually [7] and [8].

Encouraging women to take more computer science was not one of the goals of Wofford's ECS, however, that certainly has been the effect. In the years 2002-2007, eighteen (18) students graduated with the ECS. Eight (8) of these, or 44%, were women. Perhaps emphases on applications to the sciences and working in teams have been two of the factors that have contributed to the higher percentage of interest by women in computational science at Wofford.

We also were pleased that minorities completed the ECS at a slightly higher percentage than their representation in Wofford's general population. Three (3) ECS graduates (17%) were minorities.

Another surprise is the number of biology majors who are attracted to computational science. Conventional "wisdom" is that biology majors do not like or do not excel in mathematics or other technical areas. That has not been our experience, and often biology majors are at the top of their computer science and mathematics classes. Since 2002, thirteen (13) of the eighteen ECS graduates (72%) have been biology majors.

2 Introductory Textbook

While designing the two computational courses, it became evident that there were no suitable, available textbooks written for undergraduates. Arising from this need, the authors of this paper developed such a textbook. One of the authors is a mathematician/computer scientist, and the other is a biologist. The interdisciplinary nature of this area inspired collaboration. Each author had sufficient science and mathematics background to make the partnership possible and successful. Thus, with a foundation of the materials developed through the NSF grant, the authors wrote the first textbook designed specifically for an introductory course in the computational science and engineering curriculum, *Introduction to Computational Science: Modeling and Simulation for the Sciences* [9].

2.1 Content

Introduction to Computational Science: Modeling and Simulation for the Sciences prepares the student to understand and utilize fundamental concepts of computational science, the modeling process, computer simulations, and scientific applications. The text considers two major approaches to computational science problems: system dynamics models and cellular automaton simulations.

One of the positive aspects and challenges of computational science is its interdisciplinary nature. This challenge is particularly acute with students who have not had extensive experience in computer science, mathematics, and all areas of the sciences. Thus, the text provides the background that is necessary for the student to understand the material and confidently succeed in the course. Each module involving a scientific application covers the prerequisite science without overwhelming the reader with excessive detail. The numerous application areas for examples, exercises, and projects include astronomy, biology, chemistry, economics, engineering, finance, earth science, medicine, physics, and psychology.

Most sections of a module end with Quick Review Questions that provide fast checks of the student's comprehension of the material. Answers, often with explanations, at the end of the module give immediate feedback and reinforcement to the student.

To further aid in understanding the material, most modules include a number of exercises that correlate directly to text examples and that the student usually is to

complete with pencil and paper. Answers to selected problems, whose exercise numbers are in color, appear in an appendix.

A subsequent "Projects" section provides numerous project assignments for students to develop individually or in teams. While a module, such as "Modeling Malaria," might develop one model for an application area, the projects section suggests many other refinements, approaches, and applications. The ability to work well with an interdisciplinary team is important for a computational scientist. Two chapters provide modules of additional, substantial projects from a variety of scientific areas that are particularly appropriate for teams of students.

2.2 Website

The text's website (linked from `http://www.wofford.edu/ecs/`) provides links to downloadable tutorials, models, pdf files, and datasets for various tool-dependent quick review questions and answers, examples, and projects. Moreover, an online *Instructor's Manual* includes solutions to all text exercises, tutorials, and selected projects [4]. To model dynamic systems, students using the text can employ any one of several tools, such as *STELLA*®, *Vensim*® Personal Learning Edition (PLE) (free for personal and educational use), *Berkeley Madonna*®, the Python programming language, or *Excel*®. The text also employs a generic approach for cellular automaton simulations and scientific visualizations of the results, so that students can employ any one of a variety of computational tools, such as *Maple*®, *Mathematica*®, *MATLAB*®, the Python programming language, or *Excel*®. Typically, an instructor picks one system dynamics tool and one computational tool for class use during the term.

3 SIAM Working Group Report

In 2006, a SIAM Working Group on Undergraduate Computational Science and Engineering Education issued a report [10]. The committee consisted of Peter Turner, Chair, Kirk Jordan, Linda Petzold, Angela Shiflet, and Ignatios Vakalis. To a large extent Wofford College's Emphasis in Computational Science follows the working group's recommendations.

3.1 The Report

The SIAM Working Group Report noted, "Some content elements appear to be common in the emerging undergraduate CSE curriculum: scientific programming, numerical methods/scientific computing, linear algebra, differential equations, mathematical modeling, and statistics are common mathematics components; advanced programming, parallel and high performance computing, and scientific visualization are commonly added where the program has its home closer to computer science; simulation, optimization, computational fluid dynamics, image and signal processing are among the offerings from some of the applications areas…. By the nature of CSE, the successful undergraduate CSE student will have skills in applied mathematics, computing including some parallel or high performance computing, and at least one application field".

The report continued, “It is absolutely essential that interdisciplinary collaboration be an integral part of the curriculum and the thesis research….Expressed in broad terms, the overall needs are a combination of disciplinary skills and cross-disciplinary skills, learning how to learn, ability to work in a team, adaptability, perseverance and an interest in solving problems that may be multi-faceted”.

Topics of the report include “Nature of CSE Undergraduate Education”, “Models for CSE Programs”, “A Few Examples”, “The Value of Internships”, “Needs that Undergraduate CSE Education Must Address”, “CSE Careers”, and “Conclusion and Recommendations”.

4 Modeling in the Biology Classroom

Computational science education not only refers to establish programs, but also can involve individual courses or projects in various science courses. For the past three years, George Shiflet has incorporated a three-laboratory sequence on the modeling of the spread of disease in Microbiology, a class with 30 to 40 students.

4.1 The Laboratories on Modeling

In the first week’s laboratory in the sequence on modeling, students are given an introduction to a systems dynamics modeling tool, in this case *STELLA*, with a model

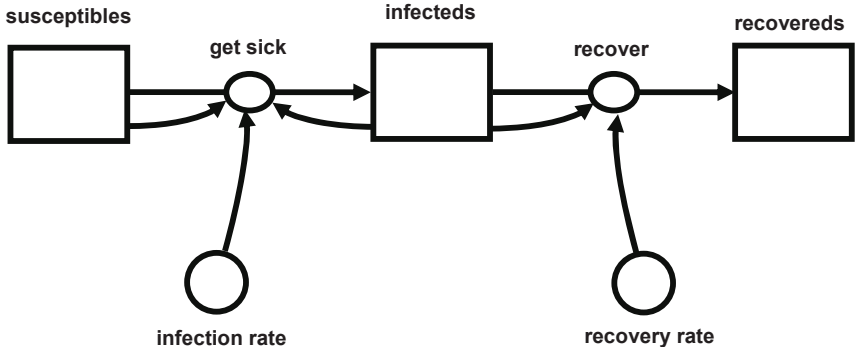

Fig. 1. SIR model diagram

```
susceptibles(t) = susceptibles(t - dt) + (-get_sick)*dt
infecteds(t) = infecteds(t - dt) +
               (get_sick - recover) * dt
recovereds(t) = recovereds(t - dt) + (recover) * dt
get_sick = transmission_constant*susceptibles*infecteds
recover = recovery_rate * infecteds
```

Fig. 2. Difference equations for SIR model generated by *STELLA*

of the interactions of predators and prey. Then, students progress through a tutorial on developing a simple SIR (susceptibles-infecteds-recovereds) model of the spread of disease. Fig. 1 displays an SIR model diagram typical of those that the students create with a systems dynamics tool. After establishing the relationships among the systems of *susceptibles* (*S*), *infecteds* (*I*), and *recovereds* (*R*), students double click each component and enter initial values, constants, and differential equations. For example, with *t* indicating time and with $dR/dt = cR$ for a positive constant *c* being the model for the rate of change of *recovereds* with respect to time, students enter the equation *recovery_rate * infecteds* into the *recover* flow from *infecteds* to *recovereds*. Similarly, the flow out of *susceptibles*, *get_sick*, is gets the equation *transmission_constant * susceptibles * infecteds*. *STELLA* translates the constants and equations into corresponding difference equations, such as in Fig. 2. Upon running simulation, *STELLA* can display tables and graphs.

After completing the tutorial, students select diseases, such as Typhus or Hepatitis C, at random. Each student is paired at with another student to investigate the disease before the next laboratory. The student pairs ascertain as much as possible about the nature of their assigned diseases, including data, such as rates of change.

In the next week's laboratory, each pair develops a model of the spread of their disease using the system dynamics modeling tool. The professor and student assistants, who are obtaining the Emphasis in Computational Science, mentor the teams.

In the final week of the lab sequence, each pair makes a presentation on their disease and model to the class. Individually, each student writes a report on his or her pair's model and what they learned from the experience. Computational science students that assist in the laboratories also help to evaluate the models.

4.2 Student Perceptions

Student comments about their projects reveal a deeper understanding of the spread of their diseases, the modeling process, and the utility of models. Comments from two students are insightful and typical. One student wrote, "I understand the relationship between these factors better now that I have worked with the model and adjusted the formulas that determine these relationships. Using (a system dynamics tool) for this modeling project was also interesting because it allowed researched facts to be projected into likely outcomes. It was fascinating to see the trends that developed as the model ran."

Another student in the class commented, "I thoroughly enjoyed creating a model to better understand a biological situation and to determine the rates at which an infection, recovery, or death can occur. Developing the mumps model allowed me to better understand how this disease can actually infect a population by producing real numbers. The graphs, in particular, helped me visualize the results of the model itself….I felt as if I wanted to add more and more things to make a more complicated but more *realistic* model. The ability to work with simulations like this will allow the scientist or researcher to be able to understand better the disease and relate to it in a way that could possibly allow for a better prevention of the disease".

4.3 Why Model in a Biology Lab?

Modeling in the microbiology lab has proved beneficial in several key areas:

- Understanding of fundamental concepts, such as rate of change
- Critical thinking skills, such as model construction, extension, and testing
- More effective problem-solving skills
- Communication skills
- Interactive learning experience.

5 Conclusion

Like others, we have found that computers have become fast and cheap enough; networks have become sophisticated enough; scientific visualization has become mature enough; and the Internet has become pervasive and friendly enough so that a meaningful undergraduate computational science program is not only desirable but also now possible. Students who successfully complete such a program enter a variety of scientific fields, where they will be able to collaborate more effectively with others and help to transform the way science is done.

References

1. Swanson, C.: Computational Science Education. The Krell Institute, http://www.krellinst.org/services/technology/CSE_survey/
2. Wofford College, http://www.wofford.edu
3. Shodor Educational Foundation, Inc., http://www.shodor.org/
4. Computational Science, http://www.wofford.edu/ecs/
5. Cunningham, S.: Computer Graphics: Programming, Problem Solving, and Visual Communication. Prentice Hall, New York (2007)
6. Spertus, E.: What We Can Learn from Computer Science's Differences from other Sciences. The Barnard Center for Research on Women, http://www.barnard.columbia.edu/bcrw/womenandwork/spertus.htm
7. Margolis, J., Fisher, A.: Unlocking the Clubhouse: Women in Computing. The MIT Press, Cambridge (2001)
8. Thom, M.: Balancing the Equation: Where Are Women and Girls in Science, Engineering and Technology? National Council for Research on Women, New York (2001)
9. Shiflet, A., Shiflet, G.: Introduction to Computational Science: Modeling and Simulation for the Sciences. Princeton University Press, Princeton (2006)
10. SIAM Working Group on CSE Undergraduate Education: Undergraduate Computational Science and Engineering Education. SIAM, http://www.siam.org/about/pdf/CSE_Report.pdf