CONDENSING THE CC-2001 CORE IN AN INTEGRATED CURRICULUM

Ingrid Russell, CS Department, University of Hartford, irussell@hartford.edu
Michael Georgiopoulos, EECS, University of Central Florida, michaelg@mail.ucf.edu
Jose Castro, EECS, University of Central Florida, jcastro@mail.ucf.edu
Todd Neller, CS Department, Gettysburg College, tneller@gettysburg.edu
Daniel McCracken, CS Department, City College of New York, ccnyddm@aol.com
Laurie King, CS Department, College of the Holy Cross, la@radius.holycross.edu
Dennis Bouvier, CS Department, Saint Louis University, bouvierd@slu.edu

PANEL DISCUSSION

1. INTRODUCTION

Volume I of the Computing Curricula 2001 (CC-2001) document contains a set of curricular recommendations for undergraduate computer science programs. The document presents a computer science body of knowledge and identifies a list of core units within each body of knowledge that are recommended as a required component of the computer science curriculum. While some of these core units span hours that warrant or are equivalent to a full course, the core units for other areas are less than a full course.

While some colleges have the privilege of being able to offer and in some cases require a full course in each of the body of knowledge areas presented, many colleges with smaller programs are not able to do so. This could be due to an already overcrowded curriculum or lack of faculty with expertise in these areas. The intent of this panel is to present ways to cover these core topic requirements in the curriculum in the case when a required full course in the knowledge area is not possible at a given institution. Areas covered in this panel are intelligent systems, human-computer interaction, social and professional issues, and graphics and visual computing. Faculty can then tailor these suggestions and experiences as they see fit in their curriculum to meet their curricular needs. In addition, it is also our goal to engage the audience with similar ideas that they have implemented.

2. POSITION STATEMENTS

2.1 Intelligent Systems

Ingrid Russell, University of Hartford
Michael Georgiopoulos and Jose Castro, University of Central Florida
Todd Neller, Gettysburg College

Computing Curricula 2001 (CC-2001) recommends ten core hours in the area of Intelligent Systems (IS). The core topics as stated in CC-2001 are (1) fundamental issues in intelligent systems, (2) search and constraint satisfaction, and (3) knowledge representation and reasoning.

While some colleges have the capability to offer and in some cases require a full course in Artificial Intelligence/Intelligent Systems that covers all these recommended IS units, many smaller programs in mostly teaching-oriented liberal arts colleges are not able to do so and may have to deal with a condensed curriculum model.

These panelists will discuss how these IS core units of CC-2001 can be integrated into the curriculum through the traditional core courses on discrete mathematics, data structures, and algorithms. For example, when breadth-first search, depth-first search, and Dijkstra's algorithms are covered in a Data Structures course, they are usually presented as algorithms on explicitly specified graphs or trees that are stored in memory. Such a discussion of searching small graphs in these courses can be extended to include the type of problems encountered in AI where these search algorithms apply to very large and in some cases infinite graphs or trees that are only implicitly defined. This will allow for the treatment of search algorithms in an AI setting.

Queues, stacks, and priority queues can be applied to node expansion within a general search algorithm, yielding breadth-first, depth-first, and uniform-cost search respectively. Breadth-first, depth-first, and iterative-deepening search provide excellent complexity analysis exercises and a compelling illustration of the art of trading off between time efficiency, space efficiency, and quality of solution. The introduction of a heuristic function leads to straightforward extensions of these algorithms to several informed search algorithms (e.g. greedy, A*, IDA* searches and weighted variants), making possible richer discussion of algorithm trade-offs concerning cost-versus-benefit of heuristic function computation.

Machine learning algorithms also provide good examples of heuristic approximation algorithms. We will extend this discussion to present examples of how machine learning concepts can be integrated into the curriculum through programming projects in the introductory computer science courses. We will report on a project in progress at the University of Central Florida that involves the integration of machine learning into the curriculum. The project is funded by an NSF CRCD grant[1].

2.2 Human Computer Interaction (HCI)[2]

Daniel D. McCracken, City College of New York

---

[1] This work was supported in part by National Science Foundation grant CRCD-0203446.

[2] This work was supported in part by National Science Foundation grant CCLI-DUE-0088184.

Human-Computer Interaction (HCI) can be the basis for a 10-course BS or MS program, one undergraduate course, 10% of a capstone senior project, or a week in software engineering. AT CCNY I teach a 3 semester-hour elective course called Web Site Design, which consists of an 8-week HCI core plus selected additional material on color, typography, accessibility, and globalization.

The core of HCI means bringing real users into every stage of development: at the beginning, at the prototype, and before release. The goal is to avoid the #1 complaint of Web users: "I can't find what I'm looking for!" There is a teachable body of knowledge on how to design Web sites that work.

HCI is *not* just making pretty Web pages. I know a university Web site that is graphically stunning—but I cannot find the address of the school, it takes work to discover the tuition, and the search engine has never heard of my friend who is past chair of philosophy. If the developers were forced to sit in a usability lab and watch a succession of intelligent, Web-literate people try to find the address of the school and fail, they would slink out a back door and go learn how to bring the user into the navigational design process.

Computing Curricula includes 8 core hours of HCI. Although it is desirable to cover the material in a full course, many schools may find it difficult to introduce a complete new course, even as an elective. I will present various ways of integrating HCI into the curriculum including a unit in software engineering and integrating HCI into a senior capstone course.

2.3 Social and Professional Issues[3]

Laurie King, College of the Holy Cross

Computing Curricula 2001 enumerates 17 core hours of social and professional issues that should be included in a computer science major. The two approaches for including this material are to devote an entire course to the material or to include modules in many different courses. As the notes in the Computing Curricula 2001 indicate, the module "strategy sometimes fails to have the desired effect" and therefore social, ethical and professional issues are relegated to last minute, low priority coverage. Although many faculty believe people who make technical decisions must be educated to make good ethical decisions, faculty lack ready access to materials that both highlight ethical issues and have technical merit. The NSF CCLI-DUE-9952841 Doing On/off-Line Computer Ethics (DOLCE) grant fills this need with web-based materials for teaching computer ethics to undergraduate computer science majors. The materials are integrated with the third edition of Computer Ethics, by Deborah Johnson. The materials, spanning all undergraduate levels, are designed for classes or modules that are online or face-to-face. The materials emphasize computer ethics theory and analytical skills, societal issues in computing and telecommunication, and professional ethics. These materials do not duplicate existing materials, but do, with permission, link to existing materials. This panelist will provide an overview of the

[3] This work was supported in part by National Science Foundation grant CCLI-DUE-9952841.

kinds of software tools, rubrics, assignments and classroom activities available from DOLCE, and discuss one or two specific examples in greater detail.

2.4 Graphics and Visualization

Dennis Bouvier, Saint Louis University

Computing Curricula 2001 recommends only 3 hours of core topics in the area of Graphics and Visual Computing (GV) for the computer science major. Three lecture hours of topics hardly justify a computer graphics course; however, the GV core topics, as well as several elective GV topics, are easily incorporated into a variety of lower-level courses beginning with CS1. The use of graphics topics can range from minor tangents in a course, such as using a 2D array to represent an image, to being a central part of the course, such as using the 3D graphics applications to demonstrate programming principles. Computer graphics topics are easily integrated into data structure and other courses.

An added benefit of the integrated approach to graphics is an increased level of interest for beginning programmers. This panelist will present a variety of ways to integrate GV topics into otherwise traditional undergraduate courses.

PRESENTERS' BACKGROUNDS

Ingrid Russell is a Professor of Computer Science at the University of Hartford. She is Immediate Past President of the Consortium for Computing Sciences in Colleges (CCSC). Her research interests are in artificial intelligence, web technologies and programming, and computer science education. She chaired the Intelligent Systems focus group of the IEEE-CS/ACM task force on Computing Curricula 2001 and served as a member of the focus group on the "Computing Core".

Michael Georgiopoulos is a Professor of Electrical Engineering at the University of Central Florida. His research interests are in neural networks and pattern recognition. He is a co-PI on the NSF CRCD grant, Combined Research and Curriculum Development in Machine Learning.

Jose Castro is as a Visiting Lecturer at the University of Central Florida where he is pursuing a Ph.D. in Computer Engineering. His research interests are in ART-like neural networks, parallel algorithms, and data mining.

Todd Neller is an Assistant Professor of Computer Science at Gettysburg College. His research interests include the extension of AI search algorithms to continuous and hybrid dynamical systems, reinforcement learning, and global optimization. He has a Ph.D. in Computer Science from Stanford University with a distinction in teaching. He received the Stanford George E. Forsythe Memorial Award in 1999 for excellence in teaching, and the Stanford Lieberman Fellowship in 1998 for excellent teaching potential. He was named a Cornell Merrill Presidential Scholar in 1993.

Dan McCracken is Professor of Computer Science at City College of New York, the author of 20+ textbooks, and a past president of ACM. He was the 1992 recipient of

the ACM SIGCSE Award for Outstanding Contributions to Computer Science Education. His latest book is *User-Centered Web Site Development: A Human-Computer Interaction Approach,* written with Rosalee Wolfe of DePaul University.

Laurie King is an Assistant Professor of Computer Science at the College of the Holy Cross in Worcester, MA. Her research interests include computer science education, programming languages, hardware/software co-design and ethics. She has a Ph.D. in computer science from the College of William and Mary in Virginia. She is a co-PI on the NSF CCLI-DUE-9952841 DOLCE grant.

Dennis Bouvier is an Assistant Professor of Computer Science at Saint Louis University. His research interests include computer graphics, visualization, novice programmers, and computer science education. Dennis earned a Ph.D. of Computer Engineering from the University of Louisiana at Lafayette and is a member of ACM, ACM SIGGRAPH, ACM SIGCSE, and ACM SIGCHI.