The Present and Future of Computational Thinking

Owen Astrachan
Computer Science
Duke University
Durham, NC 27708
(919) 660-6522
ola@cs.duke.edu

Susanne Hambrusch
Computer Science
Purdue University
West Lafayette, IN 47906
765-494-1831
seh@cs.purdue.edu

Joan Peckham
(Moderator)
NSF
Arlington, VA 22230
(703) 292-734
jpeckham@nsf.gov

Amber Settle
Computing
DePaul University
Chicago, IL 60604
(312) 362-5324
asettle@cdm.depaul.edu

Categories and Subject Descriptors
K3.2 [**Computing Milieux**]: Computers and Education - *Computer and Information Science Education*

General Terms
Design, Experimentation.

Keywords
Computational Thinking, Curriculum, Education.

1. Summary
Intellectual constructs and tools that are widely used to solve the problems of society have been woven into educational programs. For example, the three R's (reading, 'riting & 'rithmetic) are core to a strong fundamental education, and practitioners and researchers routinely apply these tools to their daily work. Computing has become an essential and pervasive problem-solving toolset. This development has fostered much discussion about the role of computing in a modern education, the broadening nature of computing majors and concentrations and their place in post-secondary institutions, for example, [6,7]. Computer science educators recognize the importance of improving information technology (IT) skills and fluency, and a number of studies have developed guidelines on how to do this [3,4]. However, computer science has analytical concepts and tools that offer educational benefits beyond simple IT fluency.

Computational thinking was introduced [9] as a continuation of earlier discussions on the nature of computing (e.g. [5]). Similar discussions and the same terminology were introduced independently in a series of workshops and reports [2]. This has helped the computing community to strengthen description and definition of the problem solving skills that computing brings to society, through education, outreach, and research.

Over the past few years, computational thinking concepts have served as a basis for several projects, workshops and efforts aimed at more precise, and at the same time, deeper and wider interpretation of computing. This includes attention to K-12 curricula, general education at colleges and universities, as well as interdisciplinary research and technology transfer.

This panel will outline a sampling of the activities and projects that have begun to define and address computational thinking. The moderator will start with an outline of national computational thinking activities and developments. The panelists will talk about their individual projects and activities, and outline their visions for future developments in the computing and broader educational communities around computational thinking. The session will then be opened for discussion; the audience will be encouraged to ask questions and contribute ideas for the development of computational thinking across its many dimensions.

SIGCSE'09, March 3–7, 2009, Chattanooga, Tennessee, USA.
ACM 978-1-60558-183-5/09/03.

2. Owen Astrachan
We are working in several areas to incorporate computational thinking into our core courses for majors, into new courses for non-majors, and as part of re-imagining introductory courses before college, e.g., at the high school level. Some of our work mirrors approaches seen at other institutions (for example, [8]) where introductory courses are changing to match the needs of students in other disciplines who are using computation and programming. We have developed interdisciplinary minors in Computational Biology and Computational Economics, but in some ways these represent low-hanging fruit since we have faculty active in these areas and developing the programs and courses naturally reflect their interests as well as those of the students enrolling in these programs. We are also changing the approach we use in our CS2 course so that programming is presented as a tool used to explore areas from computer science, other sciences, and society [1]. Programming is still of great importance in this course, but students are using the scientific method where appropriate and focusing on solving and understanding problem-domains in many areas rather than understanding inheritance. Although the early work exhorting the community to "think computationally" [2,9] specifically talks about skills and processes that go beyond and transcend programming, these articles do specifically ask that we develop approaches to explaining how to "think like a computer scientist". We have developed a course for majors, now being introduced to non-majors, that takes a different approach to computational thinking. The course is titled *Technical and Social Foundations of*

the Internet. One view is that we look at how computational thinking has transformed the world. But it is not computational thinking that we study, it is the products and methodologies of those who have done the thinking and built the internet, the world-wide web, Skype, and Youtube that we study. A somewhat related course is being developed at Harvard by Hal Abelson, Harry Lewis and others.

We will also discuss possible new directions for introductory courses, including Advanced Placement courses that directly incorporate computation in their development. These possible approaches reflect collaborations between educators at all levels, the College Board, and the National Science Foundation.

3. Amber Settle

Faculty in the School of Computing and the School of Cinema and Interactive Media in the College of Computing and Digital Media (CDM) began a project in August 2008 to integrate computational thinking in various general education courses at DePaul University. Over the past 5 years in partnership with many academic units, the CDM faculty have developed nearly 50 liberal arts courses in nearly every area of the Liberal Studies Program. While primarily developed to teach liberal arts concepts and skills, these and other courses developed by CDM faculty include elements of computational thinking.

In the first year of the project, the CDM faculty are working to make computational thinking explicit in a selection of these courses, to develop tools that evaluate the learning of computational thinking, and to create a framework that can be used by non-technical instructors to integrate computational thinking in their own courses. Computational thinking will be enhanced and expanded in 10 CDM courses in a variety of areas, including computer science, e-commerce technology, computer game development, animation, and digital cinema. Computational thinking activities, assessments, and learning goals from these courses will be highlighted, and time-permitting, the initial framework will be discussed.

Faculty from outside the College of Computing and Digital Media and from outside DePaul University will participate in the project during the second year. We would particularly like to see the integration of computational thinking in the humanities, arts, and social sciences.

4. Susanne Hambrusch

At Purdue, all science undergraduates must fulfill a computing requirement. CS Faculty, in collaboration with science departments, developed a new course on computational thinking. The course uses a problem-driven approach focused on scientific discovery through computational methods grounded in computer science principles. Course development was guided by five main principles: (1) Lay the groundwork for computational thinking, (2) Teach in a problem-driven way, (3) Present examples in a language familiar to the students, (4) Use a programming language that right away allows a focus on computational principles, and (5) Make effective use of visualization.

CS faculty met with faculty from science departments to understand computational needs and expectations. The results showed a wide range of goals. The decision was made to design a course that will serve all science disciplines and to keep the background for projects and examples at a sufficiently elementary level. Our main objective was to give science majors a firm foundation of basic programming and establish an understanding of the algorithmic thought process. We chose Python as the language as it quickly allows the writing of meaningful programs and is used in serious ways by many scientific communities. After programming basics were in place, we introduced MatPlotLib and VPython, a library that allows creating sophisticated 3D visuals. Visual computing is an engaging activity that is underutilized in many CS curricula. Some lectures were given by Physics and Bioinformatics faculty who conveyed an authentic applications perspective. These lectures included concepts such as Maxwell's Demon and used state-of-the-art software, such as NetworkX and CytoScape. The interaction with science faculty is a critical element in designing an effective course for science majors.

5. ACKNOWLEDGMENTS

Our thanks to Harriet Taylor of NSF for suggesting this panel as a mode for encouraging discussion in the computing community about computational thinking. Amber Settle would like to thank NSF for support of CPATH program No. 0829671. Owen Astrachan would like to thank NSF for support of CPATH program No. 072227, and BPC program No. 0848267. Susanne Hambrusch would like to thank NSF for support of CPATH program No. 0722210.

6. REFERENCES

[1] Astrachan, O. and Denning, P. Innovating our self image. SIGCSE Technical Symposium on Computer Science Education, SIGCSE Bulletin 40, 1 (March 2008), 178-179.

[2] Bundy, A. 2007. Computational Thinking is Pervasive. Journal of Scientific and Practical Computing. 1,2 67-69.

[3] Computer Science and Telecommunications Board, National Research Council. 1999. Being Fluent in Information Technology. National Academy Press.

[4] Computing Curricula 2001: Computer Science. IEEE Computer Society and the ACM.

[5] Denning, P. 2005. Is Computer Science Science? CACM. 48,4 (April 2005). 27-31.

[6] Guzdial, M. Paving the Way for Computational Thinking. CACM. 51,8 (August 2008), 25-27.

[7] Snir, M. 2008. Computing and Information Science and Engineering: Quo Vadimus. University of Illinois at Urbana Champaign, Technical Report 13-2008-1. http://www.cs.uiuc.edu/homes/snir/PDF/Computing%20and%20Information%20Science%20and%20Engineering%20TR.pdf

[8] Wilson, G. Alvarado C., Campbell J., Landau, R., and Sedgewick, R. 2008. CS-1 for scientists. Proceedings of the SIGCSE Technical Symposium on Computer Science Education, SIGCSE Bulletin 40, 1 (March 2008), 36-37.

[9] Wing, J. 2006. Computational Thinking. CACM. 15, 5 (March 2006), 33-35.