On the Philosophy of Computer Science within the Liberal Arts

By Charles Keleman, Swarthmore College
Henry M. Walker, Grinnell College

November 5, 2008

ACM published its first comprehensive curricular recommendations in 1968 [1] and updated those recommendations in 1978 [4], 1991 [3], and 2001 [2]. Although these curricula provides fine insight, they treated all institutions as being similar; the same recommendations were to apply to technical schools, research-oriented universities, and liberal arts colleges.

In response, a group of faculty from liberal arts colleges started meeting in 1985 to consider the special philosophy, opportunities, and constraints of computer science within the liberal arts. With partial funding from the Sloan Foundation, the group published its first "Model Curriculum for a Liberal Arts Degree in Computer Science" in 1986 [7], and revisions have followed in 1996 [9] and 2007 [8]. These model curricula build on the premise that computer science fits naturally with the liberal arts; and the liberal arts setting can reinforce the fundamental concepts, theory, and practice of an undergraduate program in computer science.

To clarify a philosophy of computer science within the liberal arts, this article begins with a clarification of what might be meant by "computer science". Over the years, various audiences have viewed the field of computing in many ways, and undergraduate programs have appeared in numerous related areas (e.g., information technology, information science). Section 1 gives several perspectives regarding the specific field of "computer science."

The concept of a liberal arts education has roots in medieval universities in Europe, emphasizing the Trivium (grammar, rhetoric, and logic) and the Quadrivium (geometry, arithmetic, music, and astronomy). A liberal arts curriculum promotes a breadth of study of multiple disciplines, develops reasoning and analysis, and invites multiple views of problem solving. Section 2 outlines several views regarding the nature of the liberal arts, and corresponding characteristics of undergraduate liberal arts programs.

Since a liberal arts environment brings several important strengths to the study of computer science, an undergraduate program in computer science can build from a solid foundation. However, the liberal arts philosophy of breadth and multiple perspectives also injects some practical limitations. Section 3 explores both opportunities and constraints of computer science within the liberal arts.

Since undergraduate liberal arts programs have been producing majors for years, it is natural to consider how well graduates succeed in highly technical areas, such as computer science. Section 4 reviews some measures that suggest a liberal arts background does indeed have a fine record of success.

Caution: Although this article focuses upon perspectives of liberal arts, the article does not seek to imply that all qualities listed here are the exclusive province of liberal arts institutions. Indeed, various qualities described here may be found in many types of schools. The point of this article is to clarify a unified view of the liberal arts; the authors leave it to others to write similar views of other types of institutions and to consider just which characteristics might overlap among several categories of schools.

1 The Realm of Computer Science

Over the years, the term "computer science" has come to be used in numerous ways by various social and cultural groups. Within an academic setting, a description of the discipline of computer science may proceed in at least three ways:

- clarifying the role of computer science in relationship to computer engineering and computer applications;
- distinguishing the discipline of computer science from the widespread use of computer technology on a college or university campus; and
- examining the central core of computer science within interdisciplinary projects and endeavors.

1.1 Computer Science, Computer Engineering, and Applications

The "2007 Model Curriculum" [8] of the Liberal Arts Computer Science Consortium clarifies computer science in relationship to related subjects as follows:

> Refining the definition given in 1986 [7], computer science is the study of algorithms and data structures: their creation, analysis, and realization. In particular, computer science is the study of algorithms and data structures with respect to their

1. formal properties,
2. linguistic realizations,
3. hardware realizations, and
4. applications.

> The curriculum for a program in *computer science* as described here follows from a specific ordering of emphasis among these four components. The formal properties (1) of algorithms and data structures are emphasized over specific languages (2), machines (3), and applications (4).
>
> This definition has held up well even though the discipline of computer science has evolved substantially. Further refinements might add problem solving to algorithms and data structures, a consideration of social and ethical implications, and the study of what is and what is not possible in the context of algorithmic problem solving. However, even with these refinements, computer science, particularly in a liberal arts setting, emphasizes principles and techniques over operational and syntactic details that change rapidly. [8]

Altogether, computer science focuses on fundamental principles and underlying problem-solving approaches in the use of computers to help people solve problems.

1.2 Computer Science, Computer Technology, and Computer Packages

Expanding beyond a specific academic department, faculty and administrators sometimes use the terms “computing” and “computer science” to refer to a wide range of topics. Again some clarification may be helpful.

1. Some people use “computing” to refer to the use of multiple paradigms to solve problems, drawing upon reasoning, logic, analysis, hypothesis testing, and formal problem-solving methodologies.
2. To some, “computing” refers to the computer hardware, software, printers, networking, etc. that comprise an organizations electronic infrastructure.
3. Some consider “computing” to mean the user directives and low-level skills involved with running specific software package (e.g., what key strokes should a user type to perform a desired spreadsheet computation).

Although these descriptions may not be completely disjoint, they illustrate different emphases and perspectives. To clarify responsibilities, a department of "computer science" focuses on the problem solving of the first perspective, a department of "information technology services" or "ITS" supports and maintains the electronic infrastructure, and a "help desk" or similar organization typically provides tutoring or non-credit workshops for running specific packages.

1.3 Computer Science in a Interdisciplinary Setting

As computer science has evolved as a discipline, researchers and developers have integrated insights and advances from many related fields. For example, Computing Curricula 1991 [3] argues that the discipline of computing integrates three fundamental processes:

- *theory:* from mathematics,
- *abstraction:* based upon the scientific method, and
- *design:* from engineering [3].

In recent years, computer science has connected with many interdisciplinary efforts (e.g., bioinformatics, neuroscience), and specific boundaries for computer science itself may be fuzzy. However, even in wide ranging research, a computer science perspective likely highlights underlying algorithms, data representations, and principles as part of an overall research team.

2 Characteristics of Liberal Arts Programs

Historically, the notion of liberal arts has evolved over centuries. The focus has been on education of the whole person, and medieval universities in Europe emphasized the Trivium (grammar, rhetoric, and logic) and the Quadrivium (geometry, arithmetic, music, and astronomy). Since this time, areas of study have expanded to include the arts, language, philosophy, history or social studies, mathematics, and science. The 2007 Model Curriculum [8] gives this description:

> Liberal arts programs in computer science generally emphasize multiple perspectives of problem solving (from computer science and other disciplines), theoretical results and their applications, breadth of study, and skills in communication. In addition to the material content of computer science, the algorithmic approach is a very general and powerful method of organizing, synthesizing, and analyzing information. Three general-purpose capabilities that are

> among those fundamental to a liberal arts education are the ability to organize and synthesize ideas, the ability to reason in a logical manner and solve problems, and the ability to communicate ideas to others. The design, expression, and analysis of algorithms and data structures utilizes and contributes significantly to the development of all three capabilities. [8]

To support breadth and exploration of multiple perspectives, a liberal arts program limits work in one area, and a typical computing curriculum might show the following balance:

- Computer science courses: about 30%
- Mathematics courses: about 10%
- Other science courses: 5 - 10 %
- Non-science (e.g., humanities, social science) courses: 50 - 55%

In the resulting degree, often labeled "Bachelor of Arts", computer science and mathematics typically make up only about 40% of the overall course work for a degree; 60% of an undergraduate program would be outside the major (outside both computer science and supporting mathematics). In contrast for degrees typically labeled "Bachelor of Science" or "Bachelor of Engineering", the percentages are reversed, with about 60% of the course work for these degrees being in the major and about 40% being outside.

Considering this breakdown in another way, in a school in which most courses carry four credits, graduation typically requires 31 or 32 courses overall, of which only 8-12 will be computer science and supporting mathematics for a Bachelor of Arts degree in computer science. For a Bachelor of Science or Engineering, a program might specify 20 or more courses specifically connected with the major.

On the other hand, a Bachelor of Arts degree typically has substantial expectations regarding breadth, and a well-constructed program for computer science can draw upon this breadth in several ways. Here are a few examples:

- Many courses outside a major will likely expect substantial work on communication skills (e.g., writing, oral skills), and students likely will have significant exposure to areas in the humanities and social sciences.
- Many liberal arts settings emphasize links between departments and programs. For example, computer science may work with a philosophy department on one or more courses that consider ethics within a technological society. Similarly, courses on digital art or electronic music may enrich students understanding of how computing might fit within other disciplines and within society.
- Many liberal arts degree programs promote and expect interdisciplinary work – including both technical and non-technical perspectives on common problems.

3 Opportunities and Constraints Shape Curricula and Courses

Since a liberal arts environment promotes breadth, general knowledge, and interdisciplinary connections, a program in computer science can only require 8-12 courses. Thus, a computer science program must focus on principles, fundamental ideas, underlying concepts, and key examples. When the number of courses is limited, offerings and selections must be particularly careful and strategic.

Rather than consider these limitations problematic, however, liberal arts programs organize their curricula and options to focus on what is important, with minimal distraction by relatively minor details. This seems particularly important in a field that changes as quickly as computer science. For example, William Wulf, former President of the National Academy of Engineering, reported that a 2000 workshop calculated the "half-life of engineering knowledge" as between 2.5 years and 7.5 years. Wulf concluded, "half of what we are teaching our students in some fields (computer science, by the way, was the field of 2.5 years) is obsolete by the time they [students] graduate." [10, p. 6]

Liberal arts programs also celebrate multiple views of problem solving, considering insights of many disciplines; expectations for breadth encourage students to take courses outside their areas of specialization as well as within. This has at least two practical advantages:

- Many computing professionals have observed that significant software development projects now take place in teams; computer scientists work with experts in various application areas. Since software systems focus on application domains, the common language for a project comes from the application, not from computer science; and computing folks on any development team must be comfortable with the perspectives of application.
- Much research arises from pushing current techniques and ideas further. However, break throughs in research often arise when a person connects different ideas in creative ways. This bringing together of multiple perspectives and disciplines underlies much of the liberal arts.

A common perspective for the liberal arts is that an undergraduate degree should provide a foundation. As Wulf observes, "Every other profession treats at least a Masters Degree as the first professional degree. Engineering is the *only* discipline that believes that the baccalaureate is a professional degree. I think the fact that we have not faced up to that causes all kinds of foolishness." [10, p. 6]

An undergraduate program has many opportunities to focus upon multiple viewpoints, fundamentals, and connections; and this perspective can have significant advantages in

many areas, such as large-scale software development and ground breaking research. With this liberal arts foundation, graduate work allows professional specialization.

4 Some Outcomes from Liberal Arts Programs

Since liberal arts programs have been producing science and computer science graduates for many years, it is natural to ask how well these graduates succeed. As already noted, a liberal arts graduate likely has taken somewhat less courses in the major than a graduate with a more technical degree, but the liberal arts graduate also has likely taken considerable breadth. Direct comparison of various types of graduates can be difficult, but some quantitative and anecdotal information provide some hints.

- In recent NSF study of "Baccalaureate Origins of S&E Doctorate Recipients", only 20 of top 50 schools (ranked by number of S&E doctorates per 100 graduates) came from research-oriented universities. Most of the remaining 30 schools are undergraduate, liberal arts colleges. [5]
- One of the authors of this article observed Members of Technical Staff years ago during a project at Bell Laboratories. In that environment, all technical people were competent. However, the folks in various lead positions were notable for their communication skills and ability to work with clients, customers, and other team members.
- In a series of about ten industry panels several years ago, representatives of the computing industry were asked to identify the most important skills for potential employees. Every industry panelist, without exception, listed "communication skills" and "the ability to work in groups" as the two most important qualifications for success. Although the ordering of these two areas varied by panelist, specific technical skills never made the top two qualifications on the list.

Altogether, the experience of breadth and interdisciplinary work of the liberal arts would seem to have a direct payoff in professional careers.

5 Coda

Overall, a liberal arts program emphasizes general knowledge, multiple perspectives, alternative ways of thinking, and connections among disciplines. This philosophy places constraints on specific requirements for a computer science program, but also encourages strategic choices and a focus on long-term and fundamental ideas. Within a computer science program, liberal arts graduates master core ideas, structures, algorithms,

and methodologies, but these graduates also have considerable experience with writing, oral communication, and ideas from other disciplines. Such a breadth of background provides a strong base for professional careers as well as explorations into new areas and interdisciplinary challenges.

6 Bibliography

1. ACM Curriculum Committee on Computer Science, *Curriculum '68: Recommendations for academic programs in computer science,* Communications of the ACM, March 1968.
2. ACM/IEEE-CS Task Force on the Curriculum, *Computing Curricula 2001*, ACM and the IEEE Press, 2002.
3. ACM-IEEE-CS Joint Curriculum Task Force, *Computing Curricula '91.* Association for Computing Machinery, 1991.
4. Richard Austing, Bruce Barnes, Della Bonnette, Gerald Engel, and Gordon Stokes, *Curriculum '78: Recommendations for the undergraduate program in computer science.* Communications of the ACM, March 1979.
5. Joan Burrelli, Alan Rapoport, and Rolf Lehming, *Baccalaureate Origins of S&E Doctorate Recipients*, NSF Report 08-311, July 2008.
6. Computing Research Association, *The Taulbee Report for 2005-2006*, www.cra.org/main/cra.info.html
7. Gibbs, N. and Tucker, A. A model curriculum for a liberal arts degree in computer science. *Communications of the ACM* 29, 3 (Mar. 1986), 202-210.
8. The Liberal Arts Computer Science Consortium, "A 2007 model curriculum for a liberal arts degree in computer science", *Journal on Educational Resources in Computing (JERIC)*, Volume 7, Issue 2, June 2007, article 2.
9. Henry M. Walker and G. Michael Schneider, "A Revised Model Curriculum for a Liberal Arts Degree in Computer Science", *Communications of the ACM*, December 1996, pp. 85-95.
10. William Wulf, *The Urgency of Engineering Education Reform*, Excerpts from the LITEE 2002 distinguished Lecture, Auburn University, AL March 22, 2002 in Journal of SMET Education, July-December 2002.