Connecting Across Campus

Mark D. LeBlanc, Tom Armstrong, Michael B. Gousie
Computer Science
Wheaton College
Norton, MA 02766
508.286.3970
{mleblanc, armstrong_tom, mgousie}@wheatoncollege.edu

ABSTRACT
Computer science holds a unique position to craft multidisciplinary curricula for the new generation of faculty and students across the academy who increasingly rely on computing for their scholarship. We propose that computer science programs cease curricula models that begin with a two-course sequence that emulates the natural sciences and mathematics. We report on an aggressive strategy to work with faculty from across the disciplines of arts, humanities, and the social and life sciences to help design and deliver sets of multidisciplinary, applied, and "connected" pairs of introductory courses. Preliminary results at our small liberal arts college include an increase in the percentage of women enrolling in our connected courses, more students taking an additional course in computing, a faculty energized with sharing their research early on, and new interdisciplinary research opportunities for computer science faculty and students.

Categories and Subject Descriptors
K.3.2 [**Computer and Information Science Education**] *Computer science education, Curriculum*

General Terms
Design, Experimentation

Keywords
Applied computer science, arts, bioinformatics, curricula, humanities, intelligent systems, interdisciplinary, multidisciplinary, pipeline, recruitment, retention, social sciences, web programming, women in computing

1. INTRODUCTION
Computer Science curricula at small colleges suffers from a unique dilemma: other than our own majors, few programs require their students to take a course in computer science. When compared to other science and mathematics programs, very few, if any, require their majors to enroll in a computer science course. On the contrary, biology and environmental science programs require their majors to take at least a year of chemistry, calculus, and possibly statistics. Pre-medical students are similar but have the additional requirement of a year of physics. Chemistry students often take a year of mathematics and biology. Physics majors require their students to take a strong complement of courses in mathematics.

SIGCSE'10, March 10–13, 2010, Milwaukee, Wisconsin, USA.

So why should this matter? In a word: recruitment. As an academic program, the absence of students in computer science (CS) courses who are exploring possible majors removes the opportunity for students, especially those early in their college years, to experience the joys of computational thinking. Missing from many computer science programs are opportunities for professors to look the undecided student in the eye and say, "You are good at this. You should consider taking another course." A counter case in point: our mathematics colleagues recruit many of their majors during the introductory calculus courses. Many of these students, taking calculus to satisfy a requirement in another potential major, find they enjoy problem solving and working in the symbolic world. Coupled with praise from a professor, calculus becomes the "math farm". Assuming that computer science has "no farm", we argue here for hosting our own farm, a CS "co-op" where students with interests in other disciplines are exposed to computing applied to their area of strength.

Historically, the CS1-CS2 sequence has attempted to emulate the introductory two-course sequence of the science and mathematics majors: Physics I and II, Chemistry I and II, Biology I and II, and Calculus I and II. The difference, of course, being that each of these programs relied on the others to offer enriched, focused content. Not true for computer science, despite the increasing reliance on and use of computational thinking in math and the sciences. Nationally and locally, we labored for and successfully established computer science as its own discipline, unique and independent of math and the sciences. Through these early decades, we relentlessly modified our content, language, and/or pedagogy in the initial two computer science courses.

On our own small liberal arts campus, we reached out to the science and math programs, offering to modify our introductory course syllabi to help prepare their students for computing within their respective disciplines. After a decade of lobbying, the results are barely minimal. Within our course catalog, only two places mention computer science courses other than in the computer science section: mathematics allows CS1 to count as an elective and physics recommends CS1. Once our computer science

program was built, we hoped "they would come". Yet beyond those few students who know they are interested in computing from the start, technologically savvy students from across campus do not find themselves electing or required to take computer science. This is especially true in the liberal arts setting where very few students intend to major in computer science from the start. In fact, like many institutions, a significant majority of our incoming first year students come to campus "undecided".

At this point, we restrain ourselves from a critique of the mistakes computer science may have made in its early history to dissuade programs from requiring our courses. Rather, we share a suite of introductory courses that taps into a larger multidisciplinary model to "enlarge the funnel" [10] that leads students to additional courses in computing. Clearly, today's undergraduates constitute a wired, communicating, computationally-dependent generation. Embedded and assumed, computers are no longer "cool" for their own sake; students' comments of their experience in computing courses concur [7]. Yet, we submit that student scholars from all disciplines, the arts, humanities, and life and social sciences, are eager to know how to apply computational thinking and tools to their areas of scholarship.

In the sections to follow, we introduce the idea of "course connections" and provide four examples of new introductory courses that not only reach out to faculty across campus but rely on their multidisciplinary collaboration for successful execution, some of which have spawned new research directions for participating faculty. We close with preliminary data that offers encouraging support and a proposal that computer science should cease attempts to emulate the I-II course sequence inherited from traditional natural science and mathematics by offering connected courses as an introductory but vital beginning to the major.

2. CONNECTIONS

In December of 2001, the Wheaton College faculty adopted an innovative new curriculum centered on "Connections," pairs of linked courses that connected significantly different disciplines. In short, the faculty voted to "teach together" in various combinations. This unique alternative to a list of requirements provides an exciting way to explore different areas of knowledge and different approaches to problems. Courses are linked across any two of six academic areas: creative arts, humanities, history, math and computer science, natural sciences, and social sciences. Pairs of connected courses may be taken in either order and do not need to be taken in consecutive semesters, although departmental planning across campus attempts to offer each course of a connection within a year of the other.

In response to this institutional curriculum change, computer science has established a strong suite of six connected courses, four of which are presented here: "Computing for Poets" connected with the humanities, "Web Programming, Graphics, and Design" with the arts, "DNA" with both the life sciences and philosophy, and "Intelligent Systems" with the social sciences. Unlike models that rely on "courses outside of computer science" [3], connected courses involve multidisciplinary faculty in the design and execution of the computer science courses.

Our "computing connections" offer a number of novel features, especially for programs on small liberal arts campuses. Creating a "funnel" of introductory courses is one recommendation for small institutions [10]. Similar models are in progress for IT [1] and CIS [8] programs, used at large universities [3], and employed for reaching multiple sub-disciplines in computing [2, 7]. Collectively, our connected courses have forged new relationships across campus, with both faculty and students from a range of majors. Because students are required to take two pairs of connections, students have an incentive to consider completing a connection while taking the first course in a pair. For example, humanities students, typically from the highly-enrolled English program who are taking either "Anglo-Saxon Literature" or "Tolkien" courses are encouraged to consider completing a connection by subsequently enrolling in "Computing for Poets". Professors from each side of the connection highlight the interdisciplinary connection as often as possible in each course. Participating faculty guest lecture in the other's course. Our experience supports the recent recommendation that faculty from other departments take an active role in course development and delivery [5].

In computer science, the faculty have additionally and strategically agreed that each of our connected courses satisfy the College's Quantitative Analysis ("math") requirement so that each connected course (i) satisfies one-half of a connection and (ii) and satisfies a core curriculum requirement. With limited faculty resources, a common constraint for most small programs in computer science, the decision to exceed a minimum level of "logical and algorithmic thinking" has led each course to share a common thread, often realized via an exposure to scripting in some language [2]. By doing so, we reject a "softer introduction to computer programming" [9] and promote our introductory suite of courses with a campus-wide standard for rigor.

In a concerted effort, we have designed these introductory, applied, interdisciplinary courses *around* the research agendas of the faculty involved. This is an important element of our plan that we see as positively impacting the longevity and robustness of the model. Rather than "taking turns" teaching a common course that covers a wide spectrum of topics in computing (*cf.* the many examples of breadth courses), we present four courses that stem from faculty research. Although outside the scope of this particular paper, our decision to support a teaching model that intersects with our individual research programs should not be missed. Not only do these introductory courses allow an opportunity for faculty to share their scholarship *while* teaching at the introductory level, the courses introduced below have the added advantage of acquainting students early in their college careers with faculty's research areas.

2.1 Connecting with the Arts

Students are naturally interested in building Web pages, and many courses exist that teach how to use various tools to do just that. While we do teach how to use tools such as Dreamweaver and Fireworks in the course entitled "Web Programming, Graphics, and Design" (COMP 161), we reach much further: we teach basic graphic design principles, delve into graphics issues, such as choosing between GIF and JPEG formats, and devote about half of the semester to cover Flash movie creation. The latter includes

significant programming in ActionScript, which enables students to incorporate non-trivial interactivity and custom computations to their Web site projects [4, 11].

The course is formally connected with "Graphic Design I" in the Arts, the idea being that students would first learn about design and then enroll in our course where they would implement their ideas. Students can be flexible in the order they take their courses, so we cannot assume they will take this path. Therefore, the first portion of the course focuses on design issues. Indeed, this portion was devised with the help of an Arts instructor, who also gives guest lectures in aspects of design. The course then morphs from an artistic and design focus to implementing Web pages with Flash components to, ultimately, programming as a major element of student projects. This offers non-majors a blend of creative and computational thinking that affords plenty of opportunity for rigor within a familiar domain. Another aspect of the course is that it is also open to computer science students as a non-major elective. The department gains breadth for majors, a perennial problem at small institutions, while students have used what they learned in this course in later courses. For example, one group of majors who took the course later applied their knowledge to further research involving an information visualization problem of a faculty member in Sociology.

2.2 Connecting with the Humanities

"Computing for Poets" (COMP 131) is connected with two courses in English: "Anglo-Saxon Literature" and "J.R.R Tolkien". For example, in the connection with Anglo-Saxon Literature, the relationship between Old English poems has been a vexed question for nearly 150 years. Most of the poetry is anonymous and exists only as tenth-century copies in manuscripts (some of it is assumed to be much earlier). We have only three named authors of poetry in the Anglo-Saxon period (Cædmon, Cynewulf and King Alfred), and there are various problems with linking these names (much less their biographies) with more than a very few specific poems. Although a few prose texts are by known authors (particularly the homilists Ælfric and Wulfstan), even the majority of the prose is anonymous. Thus for years scholars of Old English have struggled to divine relationships between texts based on vocabulary, meter, and style. These results have been at best contentious and at worst completely unsuccessful. Starting with a new completely digitized Old English corpus, the computer science course focuses on algorithmic thinking and experimental design while the Anglo-Saxon course provides an in-depth review of the poems and prose in the corpus. Because students may take the courses in either order, both English and Computer Science faculty participate in syllabus design and make several guest lectures in the other course to help reinforce the connection between the disciplines.

For students majoring in the humanities and taking the computer science course, an exposure to the application of computers to manage the storage and retrieval of written texts shows by example the many new opportunities for scholars of ancient and other written works. In particular, the "Poets" course teaches computer programming as a vehicle to personalize the exploration of vast collections of poems and corpora now available online, including stylometry and authorship attribution. Course work includes a rich exposure to pattern matching with regular expressions, parsing large collections of XML files (e.g., corpora marked up according to the Text Encoding Initiative (TEI) schema), building a concordance, examining the top-N words in your own writing, and elementary statistical methods for authorship attribution [11]. Final projects ask students to design and implement an experiment to search for relationships among texts across the entire Anglo-Saxon or Tolkien corpus.

2.3 Connecting with the Social Sciences

Intelligent Systems (COMP 198) is a new non-major course we plan to connect with a number of offerings from the Psychology department (e.g., Learning and Memory - PSY 211; Cognition - PSY 222; Comparative Animal Behavior - PSY 226). To assist in planning the connection, faculty from Computer Science and Psychology have met to discuss curricular overlap for the last year. Additionally, one Psychology faculty member is attending and participating in discussions during our current offering. Through these interactions we have identified existing content that links the two disciplines. For example, Valentino Braitenberg's book *Vehicles: Experiments in Synthetic Psychology* details excellent thought experiment examples (e.g., vehicles behaving cowardly or aggressively) and motivates the connection for perspective students.

The course provides an introduction to robotics and a survey of selected topics from artificial intelligence. Unlike the Artificial Intelligence-area courses we offer for majors, this course is available to students with no programming experience. Without an explicit computational thinking requirement in the College's curriculum, we are still able to include those concepts in this non-CS1 course to non-majors. We equip students with the technical know-how to program a modern robotics platform, iRobot's Create robot, using both Alice and Python. Programming exercise topics range from mimicking simple animal behaviors (e.g., wall following) to discovering the emergence of swarm behaviors. On the theoretical side, students critically analyze seminal articles in AI to ultimately address Alan Turing's question, "can machines think?" In addition to students from Psychology, the course has attracted majors from Linguistics to Economics. These students bring to bear perspectives and other literatures that further inform class discussions on sentience, consciousness, and self-awareness.

2.4 Connecting with Science and Philosophy

An amazing blend of science, computing, mathematics, and ethics emerges when considering the molecule "Deoxyribonucleic Acid" (DNA). DNA is the blueprint of life for all organisms on Earth and throughout evolutionary time and is at the center of future personalized medical profiles. Fully sequenced genomes including the human genome and hundreds of microbial genomes have become the starting point for attempts to answer a wide range of biological and quantitative questions.

In this connection, two courses in Philosophy "Ethics" (PHIL 111) and "Medical Ethics" (PHIL 242) are paired with the computer science – biology team-taught, cross-listed course called "DNA" (COMP/BIO 242). The "DNA" course teaches computer programming as an introductory means of data mining genomic data [6, 11]. Building off a "DNA as text" metaphor, students apply a heavy dose of regular expressions in assignments

for finding genes (e.g., olfactory seven trans-membrane proteins) and upstream regulatory motifs in microbial genomes. Emulating the instructors' ongoing research, final projects pair life science and computer science students in interdisciplinary teams to apply authorship attribution and genomic signature techniques for identifying and/or clustering like-genomes based on motif frequencies. Programming topics include processing large numbers of input files, professional documentation, the data structures of arrays and hash tables, and designing *in silico* experimental methods. While satisfying an elective in both the biology and computer science majors and attracting a rich diversity of students from the computational and life sciences, students in the DNA course are also exposed to the ethical aspects of living in a post-genomic world. In both the ethics and computing courses, particular attention is paid to the increasing use and challenges of sequenced genomes as applied to personalized medicine. A series of assignments and events with students in ethics classes include: a showing of the movie GATTACA and follow-up discussion, talks and discussions with both a genetic counselor and professor of bioethics, and student-produced, one-minute YouTube "commercials" of companies (e.g. 23andME, deCODEme) currently promoting and selling medical profiles based on individual genomes. The commercials are framed from one of two points of view: (i) from the point of view of the company or (ii) from a consumer advocacy point of view.

2.5 Teaching Leads to Research

From the outset, it was clear that teaching together in connected courses would be good for our students. What was not so clear was how good it would be for the faculty's professional development and research output. Years before the connections curriculum, computer science was actively engaged in interdisciplinary teaching which in turn led to new research agendas. An NSF-funded model for our campus centered around bioinformatics and the "linking" of biology's "Genetics" and computer science's "Algorithms" courses. We define "linked" courses as two independently run courses that share bioinformatics as a common thread in their respective syllabi and that share time in the form of guest lectures, some common lab sessions (e.g., 4 out of 12 labs over the semester), collaborative programming assignments outside of lab time, and/or final interdisciplinary team projects and presentations. This early work led to the previously mentioned CS-bio-ethics connection, an introductory scripting text for biologists [6], and an active faculty and student research group focusing on new techniques for detecting horizontal transfer in microbial genomes.

Intrigued and encouraged by the process of how teaching influenced research and how the research fueled new ideas for teaching, faculty applied bioinformatics techniques that distinguish microbial genomes to differentiate multiple authors in Old English manuscripts. This led to the previously mentioned connection between English and Computer Science. A successful experience with getting undergraduates to ask computational questions across an entire corpus led to another active research group and current funding from the National Endowment for the Humanities (NEH). New, exciting experimental techniques and results on the Anglo-Saxon corpus will be integrated into the next iteration of the connected courses. As a direct result of connecting our teaching, new research was spawned where faculty bring their research into the classroom so that undergraduates can ask novel questions across digitized corpora of their choice.

3. TAKING ADDITIONAL CS COURSES

Our current recruitment strategy includes an increase in the number of connected courses offered and early enrollment data is providing encouraging support. Table 1 shows that each of our new connected courses (Poets, Web Programming, Intelligent Systems, and DNA) encourage a larger percentage of students to take at least one more of these introductory courses than our traditional CS0. (Note that CS2 is not included in these data.)

Course	Percentage of students taking this course and then at least +1 more
FYS	31
CS0	8
CS1	23
Poets*	12
Web*	14
IntellSystems*	33
DNA*	12

Table 1. Percentage of students who initially took each course and then proceeded to take at least one more of the courses in this list. An asterik (*) denotes a new connected course.

In addition, we are pleased with the success of our recent First Year Seminar (FYS) entitled "Storytelling through Computer Animation". As a complement to our connected courses, participating in the seminar program continues to be a wise and worthwhile effort.

While we originally assumed that students would follow a path from a connected course to CS1, we are pleased to see that 23% of students who start in CS1 are subsequently enrolling in a connected course. We of course recognize that "connections" are part of the general requirements and that students' course preferences are so influenced. However, connecting widely across campus and consistently offering our connections is a strategic effort on our part. If students take connections, then let them take them with us.

This raises one additional and important rationale for connections beyond enrollment numbers and recruitment strategies for the computer science program and that is our commitment to providing relevant experiences in computing for students across the academy. The reality is that most students in our connected courses will only take but one computer science course. We are convinced that an increasing number of students in our connected courses will sample applications of computing in the arts, humanities, social or life sciences and for the first time experience the relevance and benefit of computing in their academic and professional lives.

4. ATTRACTING WOMEN TO CS

Our enrollment data reveals a strong increase in the number of female students in our connected courses relative to our traditional CS0, CS1, and CS2 offerings. As shown in Figure 1, since 2000 the percentage of women in our traditional CS0 continues to decline (note that CS0 was not taught in 2004 and 2006). On the other hand, the percentage of women in our set of connected courses outpaces the traditional CS0 and CS1 entry-level courses in all but one semester.

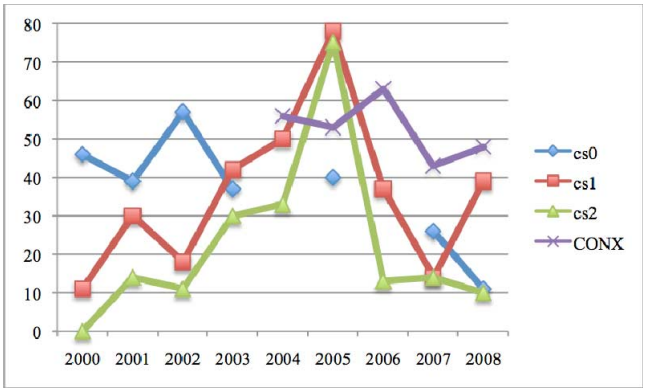

Figure 1. The percentages of female students in the traditional CS0, CS1, CS2, and new set of connected (CONX) courses. (Note that CS0 was not taught in 2004 and 2006).

5. THE END OF CS1 AS WE KNOW IT?

In our small liberal arts college setting, the model of offering non-majors courses in parallel with an introductory two-course sequence for majors is not attracting a healthy number of majors. While we are pleased with the rigor in our program (e.g., a full year of discrete math) and the success of our graduates to find jobs and get accepted in graduate programs, the tightening of academic budgets, decrease in computer science enrollments, and heightened administrative scrutiny of the number of students served is a call for creative change. We submit that computer science programs, especially those at small colleges, seriously consider (i) a more multidisciplinary approach to the "funnel model" of introductory courses, one that actively involves faculty from other programs in not only the design but also the delivery of the course(s) and (ii) eliminating the traditional CS1 course and replacing the starting curricula pipeline with a funnel from interdisciplinary, connected courses. The challenge is for each new "connected" course to equip students for further study, specifically to be ready for a course focused on data structures. We fully appreciate the scheduling constraints on small programs, and thus we assume that these new courses will replace the traditional CS1. In short, we reject the notion that by replacing CS1 with interdisciplinary courses that "something else in the curriculum must go." Rather, our proposal is an endorsement of and encouragement for the creativity of computer science faculty to craft curriculum that leverages the strengths of their local academy. No recommended two-course sequence should constrain our attempts to reach a new generation of scholars who desire to experience the relevance of computational thinking.

For decades, computer science has followed the I-II introductory model used in the natural sciences and mathematics, except these other curriculums send their students to the other programs. For various reasons, some beyond our control, computer science is not in this mix. What is increasingly clear to us and apparently to the new generation of faculty and students across campus who are confronted with and rely on computing in their scholarship, computer science has a unique opportunity to change our curriculum, flex our interdisciplinary strengths, and connect our courses to those across campus.

6. REFERENCES

[1] Besana, G.M., Dettori, L., and Steinbach, T.A. 2006. An Invitation to IT: Redesigning the First Year. *Journal of Computing Sciences in Colleges* 21, 6 (Jun. 2006), 130-139.

[2] Bills, D.P. and Canosa, R.L. 2007. Sharing Introductory Programming Curriculum across Disciplines. *SIGITE '07* (Oct. 2007), 99-106.

[3] Furst, M., Isbell, C., and Guzdial, M. 2007. Threads™: How to restructure a computer science curriculum for a flat world. Proceedings of 38th SIGCSE symposium on Computer science education. Covington, KT (Mar. 2007), 420-424.

[4] Gousie, M.B. 2006. A robust web programming and graphics course for non-majors. Proceedings of the 37th SIGCSE technical symposium on Computer science education. Houston, TX (Mar. 2006), 72-76.

[5] Guzdial, M. 2009. Teaching Computing to Everyone. *Communications of the ACM*, 52, 5 (May 2009), p31-33.

[6] LeBlanc, M.D. and Dyer, B.D. 2007. *Perl for Exploring DNA*. Oxford University Press.

[7] Margolis, J. and Fisher, A. 2001. *Unlocking the Clubhouse: Women in Computing*. MIT Press.

[8] Pearce, J. and Nakazawa, M. 2008. The Funnel that Grew Our CIS major in the CS Desert. Proceedings of the 39th SIGCSE technical symposium on Computer science education. Portland, OR (Mar. 2008), 503-507.

[9] Pokorny, K.L. 2009. Introduction to Computing: A Fresh Breadth of Disciplines. *Journal of Computing Sciences in Colleges* 24, 5 (May 2009), 166-172.

[10] The Joint Task Force for Computing Curricula 2005. The Computing Curricula 2005: The Overview Report. (Sept. 2005). Retrieved from http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf in Dec 2005.

[11] Computer science course URLs involved in connections:

with Art:

http://cs.wheatoncollege.edu/~mgousie/comp161.html

with Biology and Philosophy:

http://cs.wheatoncollege.edu/~mleblanc/dna/

with English:

http://cs.wheatoncollege.edu/~mleblanc/131/syllabus.pdf

with Psychology:

http://cs.wheatoncollege.edu/~tarmstro/IS-fall2009