CS + X: Approaches, Challenges, and Opportunities in Developing Interdisciplinary Computing Curricula

Valerie Barr
vbarr@bard.edu
Bard College
Annandale-on-Hudson, NY, USA

Carla E. Brodley
c.brodley@northeastern.edu
Northeastern University
Boston, MA, USA

Elsa L. Gunter
egunter@illinois.edu
UIUC
Champaign-Urbana, IL, USA

Mark Guzdial
mjguz@umich.edu
University of Michigan
Ann Arbor, MI, USA

Ran Libeskind-Hadas
rhadas@cmc.edu
Claremont McKenna College
Claremont, CA, USA

Bill Manaris
manarisb@cofc.edu
College of Charleston
Charleston, SC, USA

ABSTRACT

Interdisciplinary undergraduate computing curricula are of growing interest to students, institutions of higher learning, and employers, and range from single interdisciplinary courses to full majors. Interdisciplinary introductory CS+X courses and majors are often particularly attractive and compelling to students from groups that are traditionally underrepresented in STEM. Some CS+X majors, for example, have graduating classes in which the representation of women is approximately twice as high as for within-discipline computer science majors. CS+X programs provide students with a combination of complementary skills that allow them to engage with some of the most important problems to society and are also highly desirable to employers. In this paper, we survey the range of types of CS+X programs and give recommendations for engaging with the opportunities and challenges in developing such programs.

ACM Reference Format:
Valerie Barr, Carla E. Brodley, Elsa L. Gunter, Mark Guzdial, Ran Libeskind-Hadas, and Bill Manaris. 2023. CS + X: Approaches, Challenges, and Opportunities in Developing Interdisciplinary Computing Curricula. In *Proceedings of ACM CS202X*. ACM, New York, NY, USA, 8 pages. https://doi.org/

1 INTRODUCTION

There is growing interest in CS+X interdisciplinary undergraduate computing programs that interweave foundational computing concepts with those of specific disciplines in the natural sciences, social sciences, humanities, and the arts. These programs have been demonstrated to substantially promote diversity and inclusion in computing and address a rapidly growing need for a computationally-sophisticated workforce across many domains that are critical to society. We survey a number of types of existing CS+X programs, and offer some recommendations for departments and institutions that wish to develop their own CS+X programs.

ACM CS202X, March, 2023,

ACM ISBN . . . $0
https://doi.org/

This paper is written in the context of the joint effort by ACM, IEEE CS, and AAAI to update the Computing Curricula Guidelines for baccalaureate degrees in computing, currently known as CS202X.

There are several motivating factors that drive the imperative for CS+X programs. First, virtually every discipline has significant challenges and opportunities that require computational methods. While many researchers and practitioners in those fields are increasingly using computational methods, undergraduates in those fields often get little or no computational training beyond, perhaps, using existing software tools.

Second, there is ample evidence that interdisciplinary computing programs significantly broaden participation in computing. For example, interdisciplinary programs have been shown in many contexts to substantially improve gender diversity and, generally, engage diverse populations of students who are unlikely to pursue a computing degree on its own [1, 2, 9, 14, 24]. The gender diversity likely depends in part on the X in CS+X. For example, at some institutions with CS+X programs where X is related to the arts, the CS+X major has approximately equal numbers of women and men, which is more than twice the national statistic for CS programs (17-22% women). Indeed, Sax et al [39] report that over the last five years that women's representation among Black computing degree earners went up for interdisciplinary computing degrees. On the other hand, at some institutions with CS+X programs where X is a traditionally male-dominated STEM field, the CS+X major has a lower fraction of women than either CS or X.

Third, there is evidence that many students are interested in achieving a level of computing literacy that does not require a full bachelors degree in computer science, and that employers seek graduates with computing skills for jobs outside of the technology sector [10]. However, due to enrollment pressures and course caps in computer science departments, non-majors are often unable to access the computing courses that they seek. CS+X majors can help computing departments better manage enrollments and collaborate with computationally-sophisticated colleagues in other fields to mount programs that benefit all students [34].

Fourth, society in general, and employers in particular, benefit from having an ample computationally-literate workforce. Computer science programs alone cannot meet that demand but other academic programs and departments can help prepare students who have broad skill sets that include depth in computing [37].

Our CS students, of course, benefit as well by having improved employment opportunities across both disciplines.

In the next section we describe several types of CS+X programs, ranging from individual introductory courses to full majors. In Section 3, we offer recommendations for implementing CS+X programs.

2 APPROACHES TO INTERDISCIPLINARY COMPUTING

Interdisciplinary computing programs come in many shapes and sizes. In some cases, they comprise a single integrated introductory computing course or a short sequence of such courses. CS+X majors often comprise a set of existing foundational courses in both computer science and the paired discipline, coupled with one or more integrative (or "synthesis") courses and a capstone project. In other cases, computing and the paired discipline are deeply interwoven in specially-designed courses. In this section, we review some of these different approaches, pointing to existing exemplars of each.

2.1 Interdisciplinary Introductory Courses

Researchers and practitioners have suggested that by contextualizing computing as a medium and problem-solving tool applicable to many fields, students will be more inclined to see the computing discipline's value [5, 20]. Indeed, much of the research on interdisciplinary efforts in computing focuses on introductory courses [3, 4, 11, 13, 23, 31, 36, 43]. However, the extent to which students pursue longer-term interdisciplinary efforts after the interdisciplinary introductory courses is unknown [25].

Barr [3] observed that, at Union College, the introduction of interdisciplinary courses in the introductory sequence led to an increase in women's representation in the major from 10% to 37% over a decade, and, further, that the rate at which women and men drop the introductory courses is the same (a rate not observed at many universities [6]).

Guzdial led the development of an introductory course in computing for liberal arts, architecture/design, and business/management students at Georgia Tech. The course that it replaced had nearly a 50% withdrawal or failure rate. "Introduction to Media Computation" taught Python programming by having students create image filters, modify and combine sounds, and implement digital video effects [16, 19, 22]. Over the next 10 years, the course had a significantly higher retention rate than comparable classes (85% on average), and was adopted by other institutions [21].

Following Guzdial's lead, Manaris developed a course, "Computers, Music, and Art," which uses music making as the context to introduce computer programming [29, 30]. It also uses Python due to its syntactic elegance and simplicity [27]. This course borrows elements from algorithmic music composition, human-computer interaction, and artificial intelligence to introduce the creative side of computing in the context of music, sounds, images, and other digital artifacts. This course contributed to the development of the Computing in the Arts interdisciplinary major at the College of Charleston, presented in more detail below. Statistics over a six-year period indicate that this and related courses have a significant impact on broadening participation, similar to the impact observed by Guzdial.

Harvey Mudd College developed "CS 5 Green", a course that satisfies the college's core computer science requirement by teaching computer science principles and programming through applications in modern molecular biology [13, 14]. CS 5 Green has been shown to attract and retain women in both the computer science major and other computing-related majors such as mathematical and computational biology.

2.2 Interdisciplinary Minors

While interdisciplinary introductory courses are potentially impactful, they provide only modest exposure to computing and its context in a second discipline. An interdisciplinary minor that blends CS with another discipline is the next level of investment in CS+X curricula. At a minimum, such programs typically comprise approximately six courses: three from computing and three from the paired discipline. As discussed in the next section, there is significant benefit in offering at least one "synthesis" course, often a capstone, that integrates the concepts and content from the two disciplines.

2.3 CS+X Majors

Several colleges and universities have created one or more CS+X majors. Representative examples can be found at Colby College, College of Charleston, Georgia Tech, Harvey Mudd College, Lewis University, Northeastern University, Occidental College, the University of Illinois Urbana-Champaign, Union College, and the University of Kansas. In these majors, the required courses for the major are typically about half in CS and half in discipline X.

At Union College, for example, CS+X studies come under the umbrella of the interdepartmental (ID) major. Each department specifies the set of courses that comprise their half of an ID major. Students are free to combine any two of these, and the CS department adds the additional requirement that the student's senior project must combine both disciplines. Students have done ID majors combining CS with a wide range of fields including Studio Art, Classics, Music, Economics, Political Science, and Biology. Colby College takes a different approach with designated majors in Computational Biology, Environmental Computation, Computational Psychology, and CS+X majors involving Music, Theater and Dance.

The BS in Computational Media at Georgia Tech is a joint degree program between the Ivan Allen College of the Liberal Arts, the School of Music, and the College of Computing [18]. Students take approximately half of their required courses in computing, and the other half in their media specialty. Computational Media, like Computer Science at Georgia Tech, is a Threads-based major [17] so students can combine computing interests in AI or HCI with media specialties such as games, film and media studies, or music technologies. Computational Media is nearly gender balanced, while approximately 26% of students in the within-discipline CS major at Georgia Tech are women.

The BA in Computing in the Arts (CITA) at the College of Charleston is another such degree [28]. CITA offers an interdisciplinary experience in CS and the arts, with concentrations in art, dance, digital media, game development and interaction, music, and theatre. The CITA curriculum consists of 58 or more credit hours of required coursework in CS, art (music, art and art history,

Table 1: Example degree from the College of Charleston: Computer Science (CS) and Art (A). For each entry we specify the discipline and whether it is required (R), an elective (E), or integrative (I). Note that all electives are required, but in each case there is a set of courses from which to choose.

CS	R	Computer Programming 1
CS	R	Computer Programming 2
CS	R	Discrete Structures (taught by math)
CS	R	User Interface Development
CS	R	Seminar on Computing and Society
CS	E	Choose one of the following: 1. Data Struc. and Alg.; 2. Server-side Web Prog.; 3. Mobile-Application Dev.
CS	E	Intro to Architecture OR any 300+ level CS class
A	R	Drawing I
A	R	Select one of: 1. New Media in Contemporary Art; 2. The Art of Creativity
A	R	Junior Practicum
A	R	Senior Capstone
A	E	Select 4 electives from the the Arts
	I	Intro to Computing in the Arts
	I	Game Programming
	I	Computing in the Arts Practicum
	I	Choose one of: 1. Computing in the Arts Capstone; 2. CS Soft. Eng. Capstone

theatre, or digital media), math, and synthesis courses. The CITA program is structured around several synthesis experiences (offered as integrated CS+X courses) at the freshman, sophomore, junior, and senior levels. A six-year longitudinal study shows that a CITA degree attracts a different student population to computing, attracts and retains almost twice as many women students, and graduates 45.6% women compared to the 17-22% statistic in computer science across the nation [1, 2]. In Table 1 we show the CS and Art Concentration [12]. Note that for this degree there are specially designed integrative courses (designated as I in Table 1).

Computer Science at the University of Illinois Urbana-Champaign has a long history of blended programs, starting with the original Mathematics & Computer Science program established in 1968. The CS program came later, in 1972, and the Statistics & Computer Science program was established in 1988. The formal CS+X degree program framework was introduced in 2013. It comprised 8-9 courses in CS, 3 courses in mathematics, and 8-9 courses in X, with at least the possibility of an interdisciplinary course offered either in CS or in X. Currently, there are 14 "blended" CS programs across five colleges. As of the start of Fall 2022, there are 1296 students enrolled in the straight CS program, and 1098 students enrolled across the CS+X programs. In Table 2 we show the degree requirements for CS+Anthropology from UIUC [41]. Note that eight classes are required from the Anthropology courses and either nine or ten from Computer Science. This is an example of a degree program without special integrative courses.

Northeastern University offers 42 CS+X degrees (called Combined Majors) [9]. Students take 9-13 courses in each discipline, at least one integrative course, and a capstone course that further integrates the two disciplines. The first combined majors were created

Table 2: Example degree from UIUC: Computer Science (CS) and Anthropology (AN). For this degree, a minimum of eight Anthropology courses is required.

CS	R	Introduction to Computer Science 1
CS	R	Introduction to Computer Science 2
CS	R	Discrete Structures
CS	R	Data Structures
CS	R	Software Design Lab
CS	R	Introduction to Alg & Models of Computation
CS	R	Programming Languages and Compilers
CS	E	Choose one of the following: Statistical Analysis; Prob & Stats for CS; Biostatistics
CS	E	Select either <Computer Arch AND System Programming> OR <Intro. to Comp. Sys. AND 2 senior-level CS classes>
AN	R	Choose 4-6 from the following list of 7 options: 1. Introduction to Anthropology; 2. Humanizing Science; 3. Intro. to Archaeology OR World Archaeology; 4. Sociocul. Anthro OR Anthro in a Changing World; 5. Biological Anthro OR Human Origins and Culture; 6. Language in Culture OR Talking Culture; 7. Anthro of Science & Tech. OR Topics in Lang. & Cul.
AN	E	Choose 2-4 from the following list: 1. Human Biological Variation; 2. Images of the Other; 3. America in the World; 4. The Culture of Nature; 5. Special Topics; 6. Research Methods in Socio-Cultural Anthro; 7. Economic Anthro; 8. Landscape Anthro; 9. Topics in Anthro.

in 2000 with the goal of integrating CS with other disciplines and providing enough basic knowledge in each field to allow graduates to obtain a job in either field and/or pursue an MS or PhD in either CS or X. Brodley et al. report [9] that, as of Fall 2020, 44.6% of computing majors are pursuing a combined major at Northeastern; of these, 39% are women, which is significantly greater than the numbers reported in the IPEDS data for CIP-Code 11 (21.5% of CS graduates in the U.S. were women in 2019) [42]. Indeed, as of Fall 2020 at Northeastern only 22.5% of within-discipline computing majors are women. In Table 3 we show an example from Northeastern University's degree in Cybersecurity and Criminal Justice(CJ) [33]. This degree has nine required CS classes, nine required CJ classes, one integrative course, and a capstone course.

The University of Michigan College of Literature, Science, and the Arts (LSA) charged a task force to ask the question, "What do liberal arts and sciences students need to know about computing?" LSA at Michigan constitutes humanities, natural sciences, and social sciences. The task force identified three themes of how computing is used across these divisions, which are typically not addressed in CS programs [15]: Computing for *Discovery* (as in computational science), Computing for *Expression* (from social media to video games), and Computing for *Justice* (considering how computing systems exacerbate inequities, and might be designed to support democratic society). Michigan LSA has started a new program to create courses and majors to serve student needs in

Table 3: Example degree from Northeastern University: Criminal Justice and Cybersecurity.

CS	R	Discrete Structures
CS	R	Fundamentals of CS 1
CS	R	Fundamentals of CS 2
CS	R	Algorithms and Data
CS	R	Object Oriented Design
CS	R	Networks and Distributed Systems
CS	R	Foundations of Cybersecurity
CS	R	Systems Security
CS	R	Network Security
CJ	R	Intro to Criminal Justice
CJ	R	Criminal Due Process
CJ	R	Criminology
CJ	R	Criminal Justice Research Methods
CJ	R	Analyzing and Using Data on Crime and Justice
CJ	E2	Choose one of: 1. The Death Penalty; 2. Human Trafficking; 3. Corruption, Integrity and Accountability; 4. Crime, Media and Politics}
CJ	E3	Choose one of: 1. Courts, The Third Branch of Government; 2. Youth Crime and Justice; 3. Punishment in the Age of Mass Incarceration; 4. Criminal Violence; 5. Global Criminology; 6. Psychology of Crime; 7. Corporate and White-Collar Crime; 8. Addiction and Recovery
CJ	E4	Choose one of: 1. Gender, Crime and Justice; 2. Race, Crime and Justice
CJ	E5	Complete one additional upper-level CJ course
Capstone		Choose one course from a list of integrative Cyber, Law, Philosophy, and Political Science Courses
Capstone		Select one of: 1. Senior Cap. Seminar; 2. Cybersec. Cap.; 3. Research Projects on National Security

liberal arts computing. The new *Program in Computing for the Arts and Sciences* will not be offering a computer science or information degree; those programs already exist on-campus and they mean something different than liberal arts computing. Instead, LSA is creating something new to serve different needs.

CS blends relatively easily with other STEM majors since computing is pervasive across STEM. Perhaps the most common blend at universities today is a combination of CS + Statistics for a Data Science major. Many universities are implementing data science degrees that require a substantive set of courses from Statistics or other units outside of computing. Data science is among the fastest growing majors at many universities.

2.4 Deeply Integrated Interdisciplinary Majors

The most challenging approach to interdisciplinary computing majors is to set them up so that several or even *all* of the core requirements are integrated courses. Claremont McKenna College's nascent Kravis Department of Integrated Sciences is developing new majors that integrate the natural sciences and computing. The program is organized around three socio-scientific grand challenges (Health, Brain, and Planet) and computational approaches and methods are integrated into every course as vehicles for insight and discovery – beginning with an introductory course that integrates much of a CS 1 curriculum with molecular biology, and then providing increasingly deep computational approaches in the upper-division curriculum. Importantly, the computer science material is integrated into natural science courses so that the computing material is directly motivated by applications to the sciences. In contrast to more traditional CS+X programs, students in this program take no courses listed as "computer science" but rather have computer science topics interwoven throughout the entire science curriculum.

2.5 Major plus Minor

Another approach is to encourage CS majors to pursue minors in other fields. For example, at UIUC 20% of all students majoring in CS have taken at least one minor in the last five years. The most popular minors are Mathematics, Statistics and Business, but other frequent minors include Technology & Management, Informatics, Economics, Physics, Psychology, Linguistics, Electrical Engineering, and Philosophy. Conversely, UIUC's CS Department has endeavored to provide access to CS courses for students majoring in other fields so that they can minor in CS.

The major plus minor approach is relatively easy to implement, using a subset of existing courses in a major to create the minor. A disadvantage is that students have to do their own synthesis since there are often no integration or synthesis courses available.

Northeastern's approach to minors has an integrative option, called the "meaningful minor" in CS for non-CS-Majors. In a meaningful minor, the students can customize their minor in CS to their major field of study [32]. All students are required to take CS1 and CS2 and then, with the help of their advisor, select three additional courses most relevant to their major field of study and their own interests. Indeed, the website provides students with the name of the "liaison advisor" between CS and each of the other undergraduate colleges. Additionally, students can choose an elective from a list of over 100 courses in other fields at the university that have a computational component (e.g., Digital Imaging and Editing in the College of Arts Media and Design or Computational Social Science in the College of Social Sciences and Humanities).

2.6 Double Major, where one Major is Computing

Another long-standing and popular option at many institutions is the double major, where one major is in computing, and the other is in another discipline. The obvious advantage is that it requires no additional curricular effort by the institution as the major programs already exist. Outside of liberal arts colleges, this option often requires that students take on heavy loads and reduces the number of purely free electives that students can take. Often there are no synthesis or integration courses offered between the two fields. Some students, however, are able to pursue a senior thesis or capstone project that integrates their two areas of study. Sax et al. [38] in a multi-university study, found that women were more likely than men to pursue a double major with computing.

3 RECOMMENDATIONS

In this section we offer a number of recommendations for departments and institutions that seek to develop their own CS+X programs.

3.1 Synthesis and Integration

A CS+X program can often be relatively easy to implement in terms of academic programs and courses. The set of relevant CS courses will likely be different for each paired discipline and should be carefully chosen by CS faculty in conversation with their colleagues in the other field. For example, the upper-level CS courses selected for a CS+Design degree will differ from those selected for CS+Mathematics. Students are likely to gain even greater insights and appreciate the connections between the two disciplines when there are courses and co-curricular experiences that integrate the two fields. When creating a CS+X program, the department hosting X needs to have a clear understanding what a CS+X program would look like and, even more importantly, why it should exist.

In choosing the set of courses that make up the program, each participating department must be prepared to reduce their requirements. Programs in liberal arts and sciences tend to be more flexible with modest prerequisite chains. This greatly simplifies the process of creating a CS+X program. Programs in other disciplines tend to be less flexible with longer prerequisite chains. Engineering programs in particular are prone to this problem, making it very difficult to create a CS+X program that fits within the credit hours and leaves the students any freedom to explore courses of their choosing. Lighter weight options for incorporating CS into such heavyweight programs are still an option, but this option is strengthened if attention is paid to integration of the two subjects.

One approach to integration is to offer one or more capstone integrative courses (e.g., a senior-level computational biology course in a CS + Biology program). While such courses are important, early synthesis courses - ideally available to first-year students - can be particularly effective by motivating students and establishing the connections early, as is done in the first-year "CS 5 Green" course at Harvey Mudd [13, 14]. Student projects and research opportunities that span the paired disciplines are also important.

Integrative courses may be developed by one or more faculty from a single department (usually CS), who happen to be working in the particular CS+X intersection. In some cases, integrative courses may require that (at least) a dedicated pair of faculty members, one from each field, work together to develop and teach the course. If so, departments and institutions are well-served by supporting those relationships through stipends or teaching releases. This allows faculty teams to develop and pilot the courses, and there should be full teaching credit given to both faculty members for at least the first few offerings.

We note that CS+X programs rarely succeed without Dean-level or other higher-level support. Individual faculty can rarely pull off and sustain a CS+X degree program themselves. A notable example is the failed CS+X program at Stanford, which never received that level of support, so the individual departments had no incentive to bend and blend their majors. As a result, CS+X majors at Stanford were essentially double majors, and few students took them.

3.2 Developing Teaching Capacity

Computing courses can be taught by faculty, instructors, and postdocs in other fields. At Harvey Mudd College, for example, faculty members in Biology have co-taught the biologically-themed CS 1 course with colleagues in Computer Science. Similarly, Harvey Mudd has developed the Postdoctoral Program in Interdisciplinary Computation that trains recent PhDs in scientific fields to use computation in their teaching and research. In the first year, the postdocs audit several introductory computer science courses; in the second year, the postdocs co-teach two computational courses - one in the Computer Science Department and one with their postdoc advisor in their scientific area; and in the third year the postdocs teach computational courses on their own and may develop their own courses at the intersection of their area of science and computation. The program has contributed to both the breadth and capacity of computing courses available at the College and has strengthened the skill set of young scientists as they launch their careers. This example demonstrates one of several possible paths by which Computer Science departments can partner with other departments to build capacity in computing courses, provide rich course offerings that might not otherwise be available to students, and contribute to the intellectual and professional development of faculty and instructors.

As Brodley [8] points out in a recent op-ed, drawing in faculty from across the university to teach the computing intro sequence has the advantage of creating more computing in context courses, helping alleviate the burden of booming enrollments in computing, and can pave the way toward establishing CS+X at the institution.

3.3 Building Communities

Students in joint majors sometimes report that they do not have the same sense of community as do other students. The faculty in CS may be most attentive to CS majors and the faculty in field X may do the same for their own within-discipline students. When building joint programs, it is important to be deliberate in providing students with community through advising, social events, clubs, and other activities. Even just a single pair of faculty members, one from CS and one from X, can in many cases offer sufficient mentoring for the CS+X student cohort.

3.4 Promote a Growth Mindset about CS

Getting CS faculty to agree to integrated degree programs can be hard if they believe that it lessens the CS degree. At some schools with CS+X programs, there is a perception that the integrated degrees are "CS-lite" because there are fewer CS requirements or topics addressed in the program than in the within-discipline CS major.

There is often a fixed mindset in how CS faculty think about their discipline [26]. CS knowledge is perceived as linear in this mindset – there is more of it or less of it. There is a belief among some CS faculty that students are either born to do computing with a "geek gene" or they are unlikely to succeed in computing [35].

The reality is that CS knowledge is more nuanced. Scholars who use computing in other disciplines address issues that are rarely required in a typical CS major. For example, computational artists

deal with color encodings and sound representations, technical issues for sure, but rarely part of a CS curriculum.

For interdisciplinary majors to work, CS faculty need to accept the value of diverse, alternate perspectives on computing knowledge. They need to see the value of computing in jobs other than in the tech sector. Seminar series and other targeted efforts can help promote and encourage discussions and collaboration between CS and non-CS faculty and can broaden perspectives.

3.5 Combining with Accredited Degrees

Creating an interdisciplinary major within an academic unit that has an accredited degree has its own challenges. These are typically professional degrees such as architecture, engineering, nursing, and physical therapy (PT). For nursing, PT, and architecture, lack of flexibility in the required plan of study dictates that a student could complete a minor in computing, but an interdisciplinary major would not be feasible without adding significant time to degree completion. For engineering, ABET accreditation reduces the ability to adjust requirements to create a holistic combined major. It is particularly challenging when the computing degree itself is ABET accredited. Often the CS department must be willing to have a second non-ABET accredited degree in order to realize the full potential of interdisciplinary majors. Departments and institutions may need to decide whether to forego accreditation for certain degree programs. For example, at UIUC the within-discipline CS major in the Grainger College of Engineering is ABET-CAC accredited. However, none of the CS+X programs have ABET-CAC accreditation, even those combined with other engineering majors. As discussed above, such programs are difficult to create to begin with, owing to their heavy course demands and meeting the demands of both CAC and EAC accreditation can be difficult.

3.6 Advertising, Admissions and Internal Transfers

A key component to the success of interdisciplinary computing programs is clear communication with current and prospective students. A best practice is to have all students who express interest in discipline X be made aware of the joint programs and offerings between X and CS. It is critical that advisors be well-informed about these programs and degrees and there must be clear guidance on both the CS and the discipline X web pages. It is also important that the university provide an open pathway for students changing from a within-discipline major in X to a CS+X major. Universities with capped computing enrollments (often allowing only a handful of interdisciplinary transfers) are encouraged to rethink their processes for students to become CS+X majors [8].

3.7 Broadening Participation

As discussed above, several universities have demonstrated that creating interdisciplinary majors can dramatically impact gender diversity. Three factors can help or hinder results to broaden participation in computing. First, in selecting area "X" it is important to choose fields where the demographic distribution is different than that of CS (which is approximately 80% men at the national level) [24, 25]. For example, Northeastern University's first CS+X degrees were mathematics, physics, and cognitive psychology with demographics similar to their within-discipline CS degree. Only in 2014 did they begin to focus on majors where the representation of women was close to 50% or more, which dramatically increased the representation of women in CS+X [9].

Second, universities must handle the distribution of prior experience in computing in a way that encourages students with little to no coding experience to feel a sense of belonging in the computing intro sequence [7]. As Brodley notes: "Prior experience in CS is not uniformly distributed across all genders, races and ethnicities, and further CS is only offered in approximately half of U.S. high schools (with more of those high schools in regions of economic privilege). Thus the individuals experiencing the first course required for a computing major (CS1) in this way are more likely to be from less privileged geographies and from genders and races/ethnicities historically marginalized in tech. This is particularly true of students who applied to university to major in another discipline and now are interested in exploring CS+X. It is critical that their first experience in CS be one where they feel welcome."

Third, universities need to examine their processes for declaring a CS+X major. Booming enrollments in computing have caused many universities to cap CS majors [8, 40]. If there are substantial obstacles to entry into the major, such as a minimum GPA in CS1 and CS2, then students from populations historically marginalized in tech may be discouraged from considering a CS+X major.

Lastly, it needs to be kept in mind that achieving diversity is not generally going to be an easy side effect of creating a CS+X program. Some combinations are more likely to naturally bring diversity than others, but it is important to be watchful for ways in which the structure and implementation of the program biases toward populations traditionally participating in STEM and away from those historically marginalized in tech. Admission criteria and procedures, course selection and flexibility, student culture, and more all provide influences that can easily create a hostile environment, if not monitored.

4 CONCLUSIONS

In this paper, we provide arguments for creating CS+X programs and describe several possible implementation models, with concrete examples of each. Our goal is to emphasize the value of interdisciplinary computing curricula and degree options, and provide examples of how they can be created and supported. Of the four motivating factors for creating CS+X programs outlined in this paper, we believe the single most important one is increasing diversity among computing degree recipients, and perhaps, eventually, across the tech industry.

We are realistic about the challenges. CS+X programs can fail. They can be *less* diverse than the CS major, depending on the choice of X, how the major is advertised, and how students can declare CS+X as a major. As described in the previous section, CS+X programs usually require support from the upper-level administration, engaged faculty in CS and in discipline X, attention to details like advertising and appropriate admissions and internal transfer models.

We are convinced that successful and diverse CS+X programs are possible at any higher-education institution. We have seen CS+X programs assembled out of existing courses with a minimal amount

of new development, if the leadership and faculty are willing and engaged. We strongly believe that the benefits far outweigh the costs. We believe that it is the most promising direction for growing a diverse and capable workforce and for creating a just and sustainable technological society.

REFERENCES

[1] William Bares, Bill Manaris, and Renée McCauley. 2018. Gender equity in computer science through Computing in the Arts – A six-year longitudinal study. *Computer Science Education Journal* 28, 3 (September 2018), 191–210. https://doi.org/10.1080/08993408.2018.1519322

[2] William H. Bares, Bill Manaris, Renée McCauley, and Christine Moore. 2019. Achieving gender balance through creative expression. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 293–299. https://doi.org/10.1145/3287324.3287435

[3] Valerie Barr. 2016. Disciplinary thinking, computational doing: Promoting interdisciplinary computing while transforming computer science enrollments. *ACM Inroads* 7, 2 (May 2016), 48–57. https://doi.org/10.1145/2891414

[4] Jessica D. Bayliss and Sean Strout. 2006. Games as a "flavor" of CS1. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education* (Houston, Texas, USA) *(SIGCSE '06)*. Association for Computing Machinery, New York, NY, USA, 500–504. https://doi.org/10.1145/1121341.1121498

[5] Elizabeth S. Boese, Mark D. LeBlanc, and Beth A. Quinn. 2017. EngageCSEdu making interdisciplinary connections to engage students. *ACM Inroads* 8, 2 (May 2017), 33–36. https://doi.org/10.1145/3078321

[6] Carla Brodley, Catherine Gill, and Sally Wynn. 2021. Diagnosing why representation remains elusive at your university: Lessons learned from the Center for Inclusive Computing's site visits. In *The Sixth Annual Conference on Research in Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*. http://respect2021.stcbp.org/2021/05/24/respect-2021-papers/

[7] Carla E. Brodley. 2022. Expanding the pipeline: Addressing the distribution of prior experience in CS1. *Computing Research News* 34, 6 (2022).

[8] Carla E. Brodley. 2022. Why universities must resist GPA-based enrollment caps in the case of surging enrollments. *Commun. ACM* 65, 8 (2022), 20–22.

[9] Carla E. Brodley, Benjamin J. Hescott, Jessica Biron, Ali Ressing, Melissa Peiken, Sarah Maravetz, and Alan Mislove. 2022. Broadening participation in computing via ubiquitous combined majors (CS+X). In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1* (Providence, RI, USA) *(SIGCSE 2022)*. Association for Computing Machinery, New York, NY, USA, 544–550. https://doi.org/10.1145/3478431.3499352

[10] Burning Glass Technologies and Oracle Academy. 2018. Rebooting Jobs: How Computer Science Skills Spread in the Job Market. https://www.securitymagazine.com/articles/88627-employers-increasingly-demand-computer-science-skills-in-non-tech-jobs.

[11] Lori Carter. 2014. Interdisciplinary computing classes: Worth the effort. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. Association for Computing Machinery, New York, NY, USA, 445–450. https://doi.org/10.1145/2538862.2538882

[12] College of Charleston CS Faculty. 2022. Major in Computing in the Arts, College of Charleston. https://compsci.cofc.edu/undergraduate-programs/computing-in-the-arts.php.

[13] Zachary Dodds, Ran Libeskind-Hadas, and Eliot Bush. 2010. When CS 1 is Biology 1: Crossdisciplinary collaboration as CS context. In *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education* (Bilkent, Ankara, Turkey) *(ITiCSE '10)*. Association for Computing Machinery, New York, NY, USA, 219–223. https://doi.org/10.1145/1822090.1822152

[14] Zachary Dodds, Malia Morgan, Lindsay Popowski, Henry Coxe, Caroline Coxe, Kewei Zhou, Eliot Bush, and Ran Libeskind-Hadas. 2021. A Biology-based CS1: Results and reflections, ten years in. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education*. 796–801.

[15] August Evrard, Mark Guzdial, and Computing Education Task Force. 2022. Report of the University of Michigan Computing Education Task Force in the College of Literature, Science, and the Arts (LSA). https://sites.lsa.umich.edu/computingedu/2022/01/17/release-of-the-lsa-computing-education-task-force-final-report/.

[16] Andrea Forte and Mark Guzdial. 2004. Computers for communication, not calculation: Media as a motivation and context for learning. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4 - Volume 4 (HICSS '04)*. IEEE Computer Society, Washington, DC, USA, 40096.1–. http://dl.acm.org/citation.cfm?id=962752.962945

[17] Merrick Furst, Charles Isbell, and Mark Guzdial. 2007. Threads: How to restructure a computer science curriculum for a flat world. In *SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education* (Covington, Kentucky, USA). ACM, New York, NY, USA, 420–424. https://doi.org/10.1145/1227310.1227456

[18] Georgia Institute of Technology Faculty. 2022. Bachelor of Science in Computational Media. https://catalog.gatech.edu/programs/computational-media-bs/.

[19] Mark Guzdial. 2003. A media computation course for non-majors. In *ITiCSE '03: Proceedings of the 8th annual conference on Innovation and technology in computer science education* (Thessaloniki, Greece). ACM Press, New York, NY, USA, 104–108. https://doi.org/10.1145/961511.961542

[20] Mark Guzdial. 2010. Does contextualized computing education help? *ACM Inroads* 1, 4 (Dec. 2010), 4–6. https://doi.org/10.1145/1869746.1869747

[21] Mark Guzdial. 2013. Exploring hypotheses about media computation. In *Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research* (San Diego, San California, USA) *(ICER '13)*. ACM, New York, NY, USA, 19–26. https://doi.org/10.1145/2493394.2493397

[22] Mark Guzdial and Andrea Forte. 2005. Design process for a non-majors computing course. In *SIGCSE '05: Proceedings of the 36th SIGCSE technical symposium on Computer science education* (St. Louis, Missouri, USA). ACM Press, New York, NY, USA, 361–365. https://doi.org/10.1145/1047344.1047468

[23] Michael Haungs, Christopher Clark, John Clements, and David Janzen. 2012. Improving first-year success and retention through interest-based CS0 courses. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education* (Raleigh, North Carolina, USA) *(SIGCSE '12)*. Association for Computing Machinery, New York, NY, USA, 589–594. https://doi.org/10.1145/2157136.2157307

[24] Sami Khuri, Miri VanHoven, and Natalia Khuri. 2017. Increasing the capacity of STEM workforce: Minor in bioinformatics. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (Seattle, Washington, USA) *(SIGCSE '17)*. Association for Computing Machinery, New York, NY, USA, 315–320. https://doi.org/10.1145/3017680.3017721

[25] K.J. Lehman, K. Newhouse, M. Sendowski, and A. Wofford. 2019. Doing and defining interdisciplinarity in undergraduate computing. In *Annual meeting of the Association for the Study of Higher Education (Lecture Notes in Computer Science, Vol. 700)*. Springer-Verlag, Berlin, 253–264.

[26] Colleen Lewis, Paul Bruno, Jonathan Raygoza, and Julia Wang. 2019. Alignment of goals and perceptions of computing predicts students' sense of belonging in computing. In *Proceedings of the 2019 ACM Conference on International Computing Education Research*. 11–19.

[27] Bill Manaris. 2007. Dropping CS enrollments: Or the emperor's new clothes? *SIGCSE Bull.* 39, 4 (Dec 2007), 6–10. https://doi.org/10.1145/1345375.1345377

[28] Bill Manaris, Renée McCauley, Marian Mazzone, and William Bares. 2014. Computing in the Arts: A model curriculum. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. Association for Computing Machinery, New York, NY, USA, 451–456. https://doi.org/10.1145/2538862.2538942

[29] Bill Manaris and Blake Stevens. 2018. Connecting music and computer science: An interdisciplinary learning community for first-year university students. In *Performing Arts As High-Impact Practice*, Michelle Hayford and Susan Kattwinkel (Eds.). Palgrave Macmillan, 68–82.

[30] Bill Manaris, Blake Stevens, and Andrew R. Brown. 2016. JythonMusic: An environment for teaching algorithmic music composition, dynamic coding and musical performativity. *Journal of Music, Technology and Education* 9, 1 (2016), 33–56. https://doi.org/doi:10.1386/jmte.9.1.33_1

[31] Ananya Misra, Douglas Blank, and Deepak Kumar. 2009. A Music context for teaching introductory computing. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education* (Paris, France) *(ITiCSE '09)*. Association for Computing Machinery, New York, NY, USA, 248–252. https://doi.org/10.1145/1562877.1562955

[32] Northeastern University CS Faculty. 2022. Minor in Computer Science, Northeastern University. https://www.khoury.northeastern.edu/programs/minor-in-computer-science/.

[33] Northeastern University Faculty. 2022. Cybersecurity and Criminal Justice, Northeastern University. https://catalog.northeastern.edu/undergraduate/computer-information-science/computer-science/cybersecurity-criminal-justice-bs/.

[34] National Academies of Sciences Engineering and Medicine. 2018. *Assessing and Responding to the Growth of Computer Science Undergraduate Enrollments*. The National Academies Press, Washington, DC. https://doi.org/10.17226/24926

[35] Elizabeth Patitsas, Jesse Berlin, Michelle Craig, and Steve Easterbrook. 2019. Evidence that computer science grades are not bimodal. *Commun. ACM* 63, 1 (dec 2019), 91–98. https://doi.org/10.1145/3372161

[36] Janice Pearce and Mario Nakazawa. 2008. The funnel that grew our CIS major in the CS desert. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '08)*. Association for Computing Machinery, New York, NY, USA, 503–507. https://doi.org/10.1145/1352135.1352304

[37] Bianca Quilantan. 2018. Should colleges let ailing majors die or revamp them? *The Chronicle of Higher Education* (May 2018). https://www.chronicle.com/article/Should-Colleges-Let-Ailing/243447

[38] Linda J. Sax, Jennifer M. Blaney, Christina Zavala, and Kaitlin N. S. Newhouse. 2020. Who takes intro computing? Examining the degree plans of introductory computing students in light of booming enrollments. In *2020 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT)*,

Vol. 1. 1–7. https://doi.org/10.1109/RESPECT49803.2020.9272431
[39] Linda J. Sax, Kari George, Daniel Harris, and Fay Cobb Payton. 2020. Reframing the representation of Black students in undergraduate computing. *Journal of Women and Minorities in Science and Engineering* 26, 4 (2020), 325–356.
[40] Burçin Tamer. 2022. Responses of academic units in public and private institutions to increasing enrollments in computing. *Computing Research News* 34, 5 (2022).
[41] University of Illinois Urbana-Champaign CS Faculty. 2022. Computer science + Anthropology, UIUC. https://cs.illinois.edu/academics/undergraduate/degree-program-options/cs-x-degree-programs/computer-science-anthropology.
[42] National Center for Education Statistics U.S. Department of Education. 2021. The Integrated Postsecondary Education Data System. (Retrieved on August 1 2021). https://nces.ed.gov/ipeds/
[43] Zoë J. Wood, John Clements, Zachary Peterson, David Janzen, Hugh Smith, Michael Haungs, Julie Workman, John Bellardo, and Bruce DeBruhl. 2018. Mixed approaches to CS0: Exploring topic and pedagogy variance after six years of CS0. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education* (Baltimore, Maryland, USA) *(SIGCSE '18)*. Association for Computing Machinery, New York, NY, USA, 20–25. https://doi.org/10.1145/3159450.3159592