CUE.NEXT

Chicago – D.C. – Denver

Nov. 2019 Dec. 2019 Jan. 2020

Envisioning the Future of Computing in Undergraduate Education

Report on the CUE.NEXT Workshops

Larry Birnbaum, Northwestern University
Susanne Hambrusch, Purdue University
Clayton Lewis, University of Colorado

June 2020

Introduction

Computing and Computer Science have become relevant to undergraduate education in all disciplines. Academic institutions are challenged to meet the demand of the growing and increasingly diverse student body seeking to learn more about computing, computer science, and the role of computation in their own disciplines. Courses and curricula aimed at teaching the fundamentals of computer science to CS majors generally do not meet the needs of this wider, and growing, student audience.

Three NSF-funded workshops were held between November 2019 and January 2020 to initiate a national dialog on envisioning the future of computing in undergraduate education (CUE).[1] In these CUE.NEXT workshops, faculty and administrators from computing departments and their colleagues from a wide range of disciplines across the academy worked together to define the key challenges and to discuss possible approaches to achieving this broader vision of CUE. (See https://sites.northwestern.edu/cuenext/ for more information about the three workshops.)

This report summarizes perspectives, needs and challenges, potential strategies, existing efforts, and potential and needed partnerships that emerged during the workshop discussions. Participants and organizers agreed that achieving CUE involves addressing a complex set of issues. The workshops brought together people from different institutions, different kinds of institutions, different disciplines, and different roles (administrators, tenured faculty, untenured tenure track faculty, instructional faculty). The input from this broad constituency resists easy summarization or making straightforward recommendations. The report attempts to convey what we identify as key ideas and challenges expressed by the participants, even when claims for

[1] This work is supported by the National Science Foundation under grant number CNS-1944777. Any opinions, findings, and conclusions or recommendations expressed are those of the authors and do not necessarily reflect NSF's views.

these ideas are in conflict. For example, some participants like the idea of investing in sharable course modules, while others argue that such resources are rarely used, or don’t address what they see as the important challenges in CUE.

Organization of the Report

Each of the workshops included three breakout sessions followed by report backs from small groups and a subsequent discussion involving all participants. This structure was productive and led to significant insights into ongoing and planned institutional and departmental activities, challenges and needs, as well as concerns from individuals and groups of faculty.

The next sections summarize the main points made by the workshop participants followed by recommendations on three broad topics essential to achieving CUE. Section 1 focuses on curriculum design with 1.1 highlighting approaches to CUE curriculum design and 1.2 describing critical issues to be addressed. Section 2 addresses challenges related to diversity and inclusion and provides specific recommendations. Section 3 argues that CUE efforts should be driven by new thinking and have a need for greater innovation. Section 4 addresses possible impacts of the COVID pandemic on future CUE activities and Section 5 describes the workshop organization and participant backgrounds.

1. Curriculum Design

1.1. Approaches to CUE Curriculum Design

We start by providing a compendium of the most often heard participant comments related to achieving successful CUE curricula. We split comments into those from computing and those from non-computing participants. Our subsequent recommendations build on this feedback and discussions during the workshop.

Non-computing participants observe:

- Have a low-entry barrier approach to contextualized computing with a focus on the impact on humanity and society.
- Have shorter pathways and shorter prerequisite chains.
- Have ongoing communication between CS and other domains to help CS faculty understand the computational needs of non-computing disciplines.
- Realize that CUE-like efforts are already happening in some non-computing departments (media, arts, biology, etc).
- Recognize that computing is one family of problem-solving techniques among many, and that discipline-specific knowledge and skills are needed.
- Recognize that CS students in non-CS courses where computing is used in many cases fail to engage with the other discipline. Just as non-CS students can be turned off by coding, CS students can be turned off by writing, or artistic expression.
- Stop using CS1 as a weed-out course; stop making computing/programming look scary.
- The need for strong programming skills is contested in some disciplines. Most people in non-CS disciplines may not program much.

Computing participants observe:

- Students from other disciplines want to enroll in CS courses they don't have the prerequisites for. This particularly affects courses in machine learning and artificial intelligence. This can create tensions at various levels.
- The programming languages most appropriate for other disciplines change over time, and it is hard to adjust curriculum to track this.
- Most people in non-CS disciplines may not program much, but knowledge of programming in these disciplines is essential if quality computational tools are to be created.
- We need to establish confidence (e.g., through case studies) so courses that start from zero, with no prerequisites, can get students to a strong place technically.
- We need to somehow dispense with the language of "major" vs. "non-major" courses.
- Data-centric computing might offer an alternative path into high-level computing courses with the right mixture of topics.
- Student workload has to be reasonable. Many CS courses require much more time than other courses. Is that always necessary to achieve our learning objectives? Do we want to project a weed-out image?
- Other departments often don't think they need intellectual engagement with CS. They just want service courses for their students.

Two approaches to curriculum development emerged from the discussions. In the first, computing faculty work with representatives from a range of other disciplines to identify the most important computer science principles and practices for inclusion in courses intended for students in those other disciplines but offered by computing-centric departments. We will call this the **CS-centered** approach. In the second, faculty from a discipline outside of computing work with computing faculty to develop curricula and courses addressing computational knowledge and skills for that discipline. We will call this the **discipline-centered** approach. Both approaches are worthy of investment.

Participants shared successful examples of both approaches. A number of computing faculty had been involved in developing introductory CS courses for non-majors, or creating interdisciplinary programs (e.g., data science, or CS+X programs, combining CS courses with courses in another discipline). Participants from other disciplines described a number of worthwhile efforts, including:

- A resource for biology faculty interested in incorporating computing in their courses [Wright, A. M., Schwartz, R. S., Oaks, J. R., Newman, C. E., & Flanagan, S. P. (2019). The why, when, and how of computing in biology classrooms. *F1000Research*, *8*(1854)];
- A mechanical engineering class in which students without programming experience successfully implemented numerical methods for solving key equations; and
- A data science course, focusing on textual analysis, and offered within an English department, but also open to CS students -- addressing another potential benefit of expanding computing education, broadening the perspectives of students in computing disciplines themselves.

The NSF-sponsored QUBES project, which supports the integration of mathematics and statistics into biology curricula, provides a model for a discipline-centered approach.

A strength of the **CS-centered** approach is that it can readily draw on the knowledge and experience of computing faculty around computing pedagogy, as well as an understanding of important aspects of computer science that even highly knowledgeable users of computing technology may lack, such as software engineering, or issues of scalability. A CS-centered approach also offers the prospects of more easily sharing curricular elements and lessons learned among computing courses that may be aimed at somewhat different student audiences. The challenge of this approach is ensuring that the curriculum adequately connects with the needs of learners in other disciplines.

On the other hand, a **discipline-centered** approach ensures that the curriculum will meet the needs of learners in that discipline, and that computing concepts are presented in a context that will be meaningful to those learners. But work will be needed to make sure that key aspects of computing, like software engineering, are adequately addressed.

In both approaches funding and resource allocation are critical issues—computing instructors typically lack discipline-specific expertise and vice versa. A number of participants expressed that they lacked the institutional funding/support to approach curriculum design in a truly collaborative manner in order to overcome this challenge.

It’s worth focusing on software engineering, since it figures in the evaluation of both of these approaches. Software engineering consists of the practices needed to ensure that software produces accurate results, that it can be kept up to date over time (as bugs are fixed, as related hardware and software evolves, and requirements change), and that it can be scaled up to solve larger problems. Experience shows that software often seems to work satisfactorily at first, but later is found to have been inadequately tested, so errors appear. It can’t be kept up to date (and must eventually be abandoned), and it can’t cope with bigger problems, as more data become available, for example. Successful curricula must address this issue, by providing opportunities for learners either to develop software engineering skills, or at least to understand why and when those skills will be needed. This is a high-stakes issue: progress in many non-CS fields hinges on the development of high-quality computational tools, which can only be created with leadership from non-CS researchers, as mentioned earlier. We must therefore make it possible for these researchers to gain the necessary knowledge of software engineering or at least the understanding necessary to recognize when software engineering challenges arise.

Data science programs, in which students learn how to extract insights and knowledge from large data sets, can be either **CS-centered** or **discipline-centered**. Data Science courses introduce computation in the context of interesting and important applications, including many of clear social value. This kind of contextualization is helpful and appealing for many students, and can broaden participation. The level of CS and statistical knowledge required by or imparted in these courses can vary considerably. Creating Data Science programs can create relationships between CS and other domains that build relationships for further joint curriculum explorations. Creating pathways from such programs into the wider landscape of computation is thus a promising avenue for further development.

1.2 Critical Issues to be Addressed

We start by providing a list of the most often heard participant comments on critical issues a successful CUE curriculum should address. We split into comments from computing and non-computing participants. The key recommendations we make build on this feedback and discussions during the workshop.

Non-computing participants observe:

- Context matters! Abstract examples don't motivate.
- CS courses are technicalities-first, rather than problem-first. Change this in CUE courses.
- Identify core learning goals for computing. Develop a common core. Once articulated, faculty can adopt them and assessment can measure against them.
- Be aware of the challenges non-CS/STEM students face:
    - Fear (content, competence)
    - Anxiety (failure, matriculation time)
    - Preparation (lack of foundational vocabulary, comprehension, and skills)
    - Stereotypes ("not cool", who has access, white/Asian nerds, lack of role models)
- Faculty incentives and shared resources are the key curricular challenges limiting scalability and sustainability.

Computing participants observe:

- CS courses have high enrollment and designing courses for non-majors has low priority at the current time.
- Staffing courses for CS majors is currently a challenge. Unclear how departments could manage an additional load.
- Non-majors in CS courses may be perceived by CS instructors as weakening the rigor of the courses targeted at majors.
- Managing students with diverse backgrounds in a single class is hard and requires a redesign of the course.
- The CS department in a liberal arts college has responded to the need for problem-based learning in CS courses by gathering problems from other departments. Done in at least the first three CS courses.
- Students taking computing courses in non-CS departments are often not prepared for further course coursework in CS.
- Investing in CUE may have budget implications for CS departments in a Revenue Center Management model.
- Faculty incentives and shared resources are the key curricular challenges limiting scalability and sustainability.

These comments and discussions lead to the following recommendations for institutions adopting either a CS-centered or a discipline-centered approach to CUE curriculum design.

Develop incentives for qualified faculty to get involved and actively contribute to CUE.
Participants report that computing departments are struggling to meet the demand from their own majors, and that incentives for faculty to participate in wider-ranging initiatives are scant. Faculty in non-computing disciplines, too, have their own priorities. Hardly any institutions will find it possible to hire new faculty for these efforts from outside. Administrative support, including innovation in incentives for faculty, will be needed.

Define appropriate learning objectives.
Participants from other disciplines do not feel that current CS1 and CS2 courses are well aligned with the needs of their students. It is difficult for students to connect decontextualized ideas about data and procedures to the things they need to do in their disciplines, but learning objectives, when they are defined, are often of this character. Some CS faculty, as well, feel that a fresh look at learning objectives for these courses is needed.

Develop and disseminate sharable resources.
While some faculty want *content modules* they can *adopt*, others instead want *models of pedagogy* that they can *adapt* for their courses. Modules should of course target topics that faculty are interested in, but for which they feel the need for support. There are examples of facilities for collecting and making course modules available, e.g., the National Center for Case Study Teaching in Science, Socioenvironmental Synthesis Center (SESYNC) case study collection, and the Problem-Based Learning (PBL) Clearinghouse. In contrast, a model of pedagogy might outline how a course would be structured in terms of overall topic progression, design of kinds of activities the learners engage with, and examples of how particular topics could be taught, but would not directly provide specific content for that course.

Allow paths into more advanced computing topics.
Progress is likely to be enhanced if learners are able to increase their mastery of computational problem solving, while retaining their disciplinary identities. Early courses and experiences must provide enough in the way of computational knowledge and skills to enable students to successfully engage with more advanced topics in computing that matter for their discipline.

Make use of existing efforts, including disciplinary scholarly organizations.
For example, in English and Literature, the Modern Language Association (MLA); in Psychology, the American Psychological Association (APA); in History, the American Historical Association (AHA). We should understand how these organizations can help in terms of promulgating computing in the curricula of their respective disciplines, and how CUE efforts can influence this work.

Prepare students for ongoing changes, future developments and technologies.
All disciplines evolve, and the computational issues of interest in those disciplines are likely to evolve even faster. Learners should be prepared to follow and master new developments in their areas of interest, including specifically the computational aspects, after they graduate.

2. Diversity and Inclusion

In many settings, sadly, computing has developed a culture, or is perceived as having a culture, of superiority and exclusivity, in which the uninitiated are made to feel inferior and/or unwelcome. As participants in one breakout group said, “We lose potential students before class even starts because they don’t sign up for CS courses.” Participants understood many of the challenges BPC (Broadening Participation in Computing) efforts face.

Diversity and inclusion need to be addressed wherever computing is taught, as there is always the potential for students to engage in knowledge posturing or in other ways to display condescending attitudes, and likewise for other students to feel left behind. This issue is not limited to students: Some participants felt that faculty colleagues in computing see knowledge of computing as a kind of medicine that needs to be administered to those who don’t have it. Fixed mindset attitudes, as opposed to growth mindset attitudes, still exist on campus and in our society. When these attitudes are experienced in the classroom, learners may feel that they “don’t have what it takes”, or “don’t belong”.

Workshop participants from all disciplines, in breakouts and discussions, identified a number of key elements that CUE efforts should incorporate to ensure and improve diversity and inclusion. These aren’t all new ideas, and many have been proven effective. We again split into comments from computing and non-computing participants.

Non-computing participants observe:

- Provide welcoming and shared spaces for interaction and communication.
- Infuse more creativity into CS courses; don’t just talk about the creativity CS enables.
- We didn’t realize the diversity reality in CS.
- “Why has CS not done better?”

Computing participants observe:

- Provide welcoming and shared spaces for interaction and communication.
- We need to move beyond the ‘building stuff is cool’ ethos towards a problem-solving and impact-based dialog in our courses.
- Effective diversity and inclusion strategies are hard to implement, and require effort and commitment.
- Diversity and inclusion need institutional buy-in; CS departments don’t have full faculty buy-in. Some think it is only about gender.
- There still exist too many instructors who are not aware/do not use inclusive classroom practices. We need to teach all instructors about broadening participation, inclusive teaching, and why it matters.
- There are disadvantaged students in majority groups (rural, low income, first-generation, veterans).
- Support peer mentoring. Support team projects with social impact.
- There are many good diversity/broadening programs, many good efforts, and many ideas. The challenge lies in their implementation as well as broader and effective by-in.

The key recommendations we make build on participants' feedback and discussions during the workshop.

“Technology first, application second” computing courses can limit engagement.
Students who have limited experience with computing in school may especially be more interested in what computing can do than in how it works. Courses that begin with a focus on applications can have wider appeal.

Use a broad concept of audience(s).
Inclusion in CS should support women, students with disabilities, LGBTQ students, first-generation college students, and students from rural areas, as well URM students and students from diverse ethnic backgrounds. Assume that all CS courses include students with all of these backgrounds when designing them.

Projects should be designed deliberately, and be based on knowledge of the audience(s).
Investigators should understand the background, interests, and circumstances of students they hope to attract and retain, and curriculum elements should reflect that understanding. This includes project evaluation as well as recruiting.

Introduce bridge programs.
Create and run courses or workshops that strengthen students’ background, aimed at underrepresented groups, either pre-college, or students already in college. Such bridge programs should actively link computing to students’ goals and interests. Offering a computing course for a non-computing department having a diverse student body will not by itself broaden participation in computing: It may attract only white males. A potentially instructive parallel was provided by one participant, from Music: Music schools and programs are quite diverse; music theory, within those programs, isn’t.

Ensure a welcoming environment.
Care should be taken in creating a welcoming environment in classes from day one. Computing departments should recognize that many students feel they do not belong. Social structures and spaces to help and support students should be created. Providing opportunities for more senior students to share experiences can help.

Raise awareness that problem-solving is broader than CS.
Present computing as one problem-solving approach among many. Students (and faculty) in other disciplines are put off if the value of their disciplinary knowledge and skill is not recognized.

Introduce and manage peer working groups.
Peer learning is valuable, but groups need to be managed carefully to ensure that students who have less background, or might be assumed by peers to have less background, aren’t marginalized. A number of effective strategies exist. For example, multiple sections or courses

have been used to reduce differences in background among students; or mentoring groups where students can self-select groups in which they feel comfortable.

Use competency-based assessment.
Mastery- or competency-based assessments reward learning at all levels, and at the pace and path that individual students need, for example providing multiple opportunities to learn difficult material, and with less focus on what students already know. Similarly, more pertinent assessment models may reward cooperation rather than competition. CS courses are known for highly competitive, timed tests. Such assessment methods do not lead to a welcoming and learning-focused assessment environment, and create barriers for students who have less background, and possibly broader interests, than classmates. More generally, we heard specifically from CS faculty that an emphasis on de-contextualized problem-solving in pedagogical approach can itself lead to an overly competitive atmosphere.

3. The Need for Innovation

Participants from computing and non-computing disciplines alike emphasized that innovation is needed. They argued that we have to look beyond familiar models, and cultivate, and study, more radical ideas. The need for innovation is especially acute for institutions struggling with limited resources. Faculty at these institutions have heavy teaching loads, and often teach overloads, both to meet curricular needs, and to enable contingent faculty to increase their earnings. A few CUE.NEXT participants, especially from liberal arts colleges, reported success in developing educational programs that bring computing to a wider campus audience. But most reported that progress was difficult, and argued that progress under normal institutional processes and practices will be slow. New thinking is needed, including thinking and ideas from other disciplines interested in CUE.

Participants discussed numerous approaches with computing and non-computing participants learning from each other. Key ideas and needs discussed and viewed as having potential for innovation include:

Online learning and resources.
A wealth of tutorial material on virtually any topic in computer science and related to computation is freely available online. The current environment offers great opportunities for individuals who know what they are looking for and those who are self-driven. To make online resources an effective and valued resource for CUE programs, for both faculty and students, a more effective use of online material, its integration with traditional learning approaches, and effective scalability need to be better understood. Innovation on online instruction is needed before instructors as well as students can take full advantage of these resources. Participants argued that this is not a straightforward effort. While it is important to not reinvent the wheel, there was recognition that new ideas are needed. Some argued that we don’t need more software tools that were designed without input from scientists and educators who understand learning processes and appropriate human-computer interfaces.

Peer learning.
We should draw from the growing student population to support learning with fewer faculty resources. Peer mentoring is already successfully used within CS at larger institutions with well-established CS programs. Departments need to ensure sufficient numbers of qualified and motivated students, a budget, and administrative support over an extended period of time. These aren't new ideas, but their full potential has not been realized. Can we go beyond peer learning as a supplement to traditional pedagogy, to develop approaches that radically reduce demands on faculty? It would be especially exciting to create approaches that allow students in non-computing disciplines to learn about computing, without relying on faculty in those disciplines, or faculty in computing disciplines, to invest scarce time and effort. We must develop new approaches, and evaluate their effectiveness.

Project-based learning.
Many students find substantial projects highly motivating, and student motivation can enhance learning while reducing demands on faculty. As with peer learning, this isn't a new idea, but one whose full potential hasn't been realized. Projects, if chosen by students, can increase intrinsic motivation, so that students are measuring their progress against concrete goals rather than against grades on homeworks or exams. At the same time, students can evaluate online resources against project objectives, and their own knowledge and skills, rather than relying on faculty to lay out their learning paths in detail. Further, peer learning happens naturally when students are working together toward shared goals.

These benefits are only achievable when projects are grounded in students' real interests and ambitions, not when they are assigned to meet curricular goals. In the latter case, possible freeriding, where students leave it to team members to do the work, is always a threat: if I'm doing this project just to get a grade, the less effort I invest, the better. Thus careful faculty oversight is needed to produce quality outcomes. The challenge, and opportunity, for innovation is to create ways to integrate projects that are fully aligned with students' individual goals, into our curricula, in ways that maximize learning and minimize faculty investment.

New models for educational programs.
Diversity-aware hackathons can create substantial learning opportunities on a no-credit basis. Study abroad concepts and practices could be generalized to support a year in a different department on the same campus. "Virtual" departments could be created to facilitate faculty cooperation across discipline lines -- participants repeatedly stressed the challenges that traditional departmental structures create, when knowledge must be transported across boundaries. These suggestions illustrate the feeling of workshop participants that traditional curricular and organizational frameworks often stand in the way of progress, when intellectual and practical opportunities are changing rapidly.

Why did participants place so much emphasis on innovation?
Participants identified many factors that need to be overcome to achieve CUE. They also realized the limitations in implementing them: Faculty in computing departments are working at capacity to meet the needs of their own majors. Resources at most institutions are limited, making it difficult

to add faculty to respond to increased student demand. Senior faculty tend to be conservative. Senior mentors can perpetuate traditional/conservative ways of doing things, and faculty evaluations can reward conservatism. Innovation was seen as a crucial element for success in addressing these issues.

To illustrate the sense that participants had that really new ideas and innovation are needed, one group referred to the Thinkbelt, a 1960s proposal by English architect Cedric Price to create an educational institution using railbuses running on disused railway lines in the English Potteries district to connect distributed learning centers, while themselves serving as mobile learning environments. Participants imagined a rethinking of Price's ideas on various scales, with visible, mobile structures carrying people, projects, and displays from place to place. Such structures could connect universities to communities and industries, as well as to one another. On a smaller scale they could connect different departments on a single campus.

4. Addendum: Impact of the COVID pandemic

During the time we have been analyzing and reflecting on the results of the workshop, we have all experienced the abrupt transformation of higher education, forced by the COVID pandemic. While we certainly hope that much of this impact will prove temporary, we believe the experience opens up new possibilities for CUE that the workshops did not address, as well as creating a more favorable climate for some ideas that were discussed. Here are some thoughts.

Faculty workshops can be more affordable, in time and money.
CUE.NEXT participants repeatedly called for more faculty workshops. These would provide opportunities for non-CS faculty to learn about CS pedagogy, and how computational tools could be useful in their disciplines. CS faculty could learn about how computation is used in other disciplines, so as to better support non-CS students in their courses. CS and non-CS faculty could explore how software engineering knowledge can be cultivated outside CS, and how tools created in CS can be made more easily used, maintained, and adapted by others.

Pre-COVID, people would of course have thought of face-to-face workshops to address these needs. Post-COVID, everyone realizes that there are alternatives. Further, everyone also realizes that the alternatives can be far less costly, in time and money. At the same time, it's clear that online meetings have their limitations. For example, the literature shows that online interaction is much more effective among people who know one another, while face-to-face interactions can get strangers to interact productively.

Recognizing this, conference organizers everywhere are innovating rapidly. They are exploring ways for participants to spend time socially, as happens naturally at face-to-face events. They are taking advantage of the lack of logistical constraints to try out schedules that would be prohibitive for face-to-face meetings. For example, one workshop plans to have two technical sessions separated by six weeks, so that participants can discuss the issues raised in the first session, in small group interactions scheduled during the hiatus, and then reconvene to share the results of those conversations.

Further faculty workshops would be a good way to advance the CUE.NEXT agenda. People proposing workshops should be asked to innovate, so as to exploit the flexibility and economy of online interaction, while addressing its limitations.

A new era for online educational resources?
We've all learned how online education can work, combining it in new ways with face-to-face learning, because we've had to. Initiatives aimed at enabling students to take advantage of available online resources may now have increased appeal. Resources to support students in non-CS disciplines in understanding and using computing might be especially attractive, allowing these students to advance with less reliance on scarce faculty resources. As mentioned in an earlier section, initiatives like QUBES currently support faculty (in teaching math skills in the context of biology). Analogous initiatives aimed at students could be developed. In general, students and faculty alike may be more receptive to using shared online materials, and be more knowledgeable about how such materials can work, than suggested in our pre-COVID workshops.

Broadening participation programs that function well online are needed.
This is a major challenge. The forced move to off-campus and online classes produced inequities between students with good vs. poor connectivity, or crowded vs. uncrowded living conditions, besides differences in health vulnerabilities, that fell especially hard on students from under-represented groups. Innovations in policy and practice are badly needed to mitigate these impacts. Mentorship and community with peers can be very influential in learning and are a recognized part of a college experience. We need to better understand how to create and foster mentorship and community building experiences in an online learning environment. Further, we need to better understand how course policies, such as strict due dates, often found in computing and STEM courses, put working students, students caring for families, and students without equal access online (all of which are correlated with being an URM) at a disadvantage.

At the same time, possible opportunities created by the forced changes should be explored. For example, institutions that emphasized campus visits by middle class families must now recruit online. Can this move online help them reach a broader audience? There have been a few efforts to use social media to publicize computing among students from under-represented groups. Can these efforts be scaled up, as face-to-face outreach becomes impossible? As internships move online, can this open up opportunities for students in rural communities? Can online presentations by computing professionals with diverse backgrounds be incorporated into courses that now include a mix of online and face-to-face instruction, or that are all online? Overall, can increased use of online channels help broadening participation efforts scale up?

5. Workshop Organization and Participant Background

The three CUE.NEXT workshops were held in October 2019 in Chicago, in December in Washington, DC, and in January 2020 in Denver. Applications were made by teams consisting of 2 to 5 faculty, educators, or administrators. Teams consisting of more than one institution were encouraged but not required. Each team had to include at least one member from a computing-centric department (including CS, CE, ECE, or IS) and at least one member from a non-computing centric discipline. This interdisciplinary requirement was crucial to the discussions.

Applications were accepted on a rolling basis, with each team specifying location/date preferences. Teams whose composition was judged to lack a meaningful non-computing centric representation (e.g., CS and Statistics, CS and ECE) were asked to make changes to their team composition. In total, the workshops received applications from 63 teams, and 50 teams attended one of the workshops. Table 1 shows team and participant numbers for each workshop. A listing of represented non-computing departments appears at the end of this section.

	Chicago	DC	Denver	Total
Teams	15	17	18	50
Participants	50	62	89	201
Participants from non-computing departments	28	35	45	108
Institutions	12	21	34	66

Table 1: Team, participant and institution numbers

The 13 teams (of the 63 submitting applications) not attending any workshop either withdrew their participation or did not meet expectations on team composition. For Chicago, the majority of the institutions were within driving distance -- which was not surprising due the short notice. (We note that DePaul University was represented by 3 teams at the Chicago workshop.) No participant attended two workshops. A few institutions did send different teams to two different workshops. The majority of the participants came from Ph.D. granting institutions. The non Ph.D. and non MS granting institutions included ten 4-year liberal arts colleges, four HBCU's, one Community Colleges, and one Hispanic serving institution.

In addition to the participating teams, we had a number of observers attending the workshops. These included representatives from CRA, Google, NCWIT, Pivotal Adventures, and SageFox Consulting (as well as NSF representatives).

The majority of the workshop participants were tenure-track/tenured faculty. Each workshop had participants representing the administration. Table 3 shows the number of participants having an administrative position (head/chair, associate head/chair, dean, associate dean, position in provost office, Director). We show the breakdown by computing and non-computing centric participants (CS, non-CS) as well as PhD and non-PhD granting institutions.

	PhD CS	PhD non-CS	non-PhD CS	non-PhD non-CS
Chicago	3	3	3	2
DC	3	3	2	4
Denver	7	5	2	0
Total	13	11	7	6

Table 2: Number of participants holding an administrative appointment

As many PhD granting departments have in recent years hired teaching faculty to help manage the increased course enrollments, it is not surprising that the workshops had a significant number of participants holding a teaching faculty position. For the CS centric participants, 25% of them were teaching faculty.

	Teaching Faculty CS	Teaching Faculty non-CS
Chicago	8	3
DC	9	1
Denver	7	5
Total	23	9

Table 3: Number teaching faculty participants

The three workshops had a similar agenda structure and agendas are available here. Central to the workshops were the three breakout sessions with breakout groups consisting of 7-12 participants. The report back from breakout sessions was done in a moderated panel style.

The breakout sessions were:

- Breakout by domain (e.g., social sciences, STEM, humanities, etc.); participants were split up by domain (groups were determined by the organizers).
- Breakout by institution type (Chicago) and by initiative (DC, Denver); each team participated together as one group (groups were determined by organizers);
- Breakout by topics identified by participants; each participant selected a group.

The following lists the non-computing departments represented at each workshop:

Chicago

Accounting, Business Law and Finance
Art
Bio-Health Informatics
Biology
Biology and Neuroscience
Chemical and Biological Eng.
Cinematic Arts
Civil and Environmental Eng.
English
Environmental Science
Finance and Economics
Geography
Health Information Management
Healthcare Management
Learning Sciences
Marketing and Informatics
Mechanical and Civil Eng.
Mechanical Engineering
Physics
Psychology
Public Health
Radio/Television/Film
Radiology & Imaging Sciences
Social Sciences
Theology

Washington, DC

Academic Services
Accounting
Aerospace Engineering
Anthropology
Biological Sciences
Career and Technical Education
Chemistry
Communication Arts
Sociology
Economics
Education
English
Environmental & Health Sciences
Geography and GIS
History
Sociology
Statistics
Music
Psychology
Physics
Mathematics
Teacher Education
Textile Development & Marketing

Denver

Art and Design
Architecture
Art and Art History
Biology
Chemistry
Chemistry and Biochemistry
Civil, Environmental, and Sustainable Eng.
Communication Studies
Criminal Justice and Criminology
Economics
Eng. Fundamentals and Civil Engineering
Engineering, Design, & Society
English
Environmental Studies
Film and Media Arts
General Education
Geography and the Environment
History
History/Social Studies Education
Humanities
Learning Sciences
Libraries
Mathematics
Pharmacy Practice
Physics & Astronomy
Psychology
School of Education
Sociology
Teaching, Leadership & Professional Practice
Writing, Literature, and Film

Acknowledgements

We thank all workshop participants, scribes, and student volunteers for their enthusiasm and dedication towards CUE, and for the great ideas, challenges, and examples they provided. We also thank Jeff Stiers, Colleen Gallagher, Brianna White, and Gretchen Burnett from Northwestern for their reliable administrative support, and Katie Goggin for her excellent and professional event planning. We are grateful to Sara Sood (Northwestern) and Felix Muzny (UC Boulder) for providing valuable comments on a report draft.