Faculty Views on the Goals of an Undergraduate CS Education and the Academia-Industry Gap

Sander Valstar, Sophia Krause-Levy, Alexandra Macedo, William G. Griswold, and Leo Porter
University of California, San Diego
{avalstar,skrausel,almacedo,wgg,leporter}@eng.ucsd.edu

Abstract

Previous work has found that recent computer science graduates often experience difficulty transitioning into their new roles in industry due to a significant gap between their academic experiences and industry's expectations. Although multiple studies have identified the views of students and members of industry on the value of a CS degree as preparation for industry, the faculty perspective on this topic remains unclear. Understanding these views could shed light on why the academia-industry gap has persisted despite the attention. This study identified faculty views on the goals of an undergraduate education and a CS major, focusing on preparation for careers in industry. In order to identify a spectrum of faculty views, we interviewed 14 faculty from a variety of backgrounds across three institutions. A phenomenographic analysis of the transcripts reveals that many faculty believe that industry preparation is an important programmatic goal, yet they encounter significant resource obstacles to achieving that goal.

CCS Concepts

• **Social and professional topics → Computer science education**; **Software engineering education**.

Keywords

computer science education, academia-industry gap, faculty views, phenomenography

ACM Reference Format:
Sander Valstar, Sophia Krause-Levy, Alexandra Macedo, William G. Griswold, and Leo Porter. 2020. Faculty Views on the Goals of an Undergraduate CS Education and the Academia-Industry Gap. In *The 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11–14, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366834

1 Introduction

Students tend to take a relatively pragmatic approach in deciding to attend university. They often expect their degree to prepare them for the job market and increase their employability [9, 20]. This job-focused mentality is even more prevalent in first-generation students [14] and does not change significantly, among psychology students, as they progress through their degree [20]. The expected benefits are most heavily centered along extrinsic motivations, including obtaining a better career or fulfilling family expectations. The most important factor influencing students to study CS, in addition to general interest and prior experience in the field, are career prospects [4, 10].

In contrast to their expectations, many CS graduates feel unprepared for the challenges of their first job [5, 8, 12]. This disconnect between academia and industry in CS was identified at least two decades ago [15], yet studies conducted in the past decade show that not much progress has been made to close this gap [5, 8, 12].

This study aims to uncover some of the reasons why the academia-industry gap persists. Although it is generally well-understood that students desire more career-related training [8, 9], faculty views on the issue remain relatively unstudied in CS. Faculty are important to study, as they design and implement the curriculum (e.g., choosing to what extent to realize the ACM Curricula Recommendations [11]). Faculty views were recently studied focusing on the alignment between CS education and industry needs [7], assuming faculty see industry preparation as a primary goal. Our work explores faculty views more broadly, canvassing their views on the goals of an undergraduate education and a CS major to ultimately answer the question: "Are faculty views on undergraduate CS a potential cause for the persistence of the academia-industry gap?".

Previous work has found that enhancing employability is not always a major focus of curriculum design [9] and that learning outcomes of a degree are generally developed by faculty without referring to students' motivations for attending university [20]. Thus, we expected to find that a sizeable group of faculty oppose the idea that preparing students for industry is a main goal of a CS program. As such, our questions concern (a) whether faculty believe preparation for industry is a primary goal of a CS degree, (b) why they hold that view, and (c) whether there are other obstacles to closing the gap.

To identify the range of faculty views on the goals of a CS education, we conducted semi-structured interviews. We recruited a diverse group of 14 CS faculty across 3 institutions, including a large research-intensive university, a small liberal arts college, and a large primarily undergraduate institution. We employed Phenomenography to identify the full spectrum of experiences and beliefs of the participants.

We found, first, that faculty feel that, beyond the CS fundamentals, the goals of a CS degree should be defined by the student's objectives. Moreover, with the understanding that real-world impact is likely, students should be prepared for ethical leadership. Second, a range of opinions regarding industry preparation were expressed, including faculty either opposing or supporting industry preparation playing a significant role in CS education. Third,

SIGCSE '20, March 11–14, 2020, Portland, OR, USA

ACM ISBN 978-1-4503-6793-6/20/03…$15.00
https://doi.org/10.1145/3328778.3366834

faculty admitted struggling with properly preparing students for industry, citing increasing class sizes, lack of background in industry practices, and the difficulties of designing realistic projects without overtaxing both the students and instructors.

2 Background

2.1 Motivations for Attending University

Wilson et al. suggest that there should be a close alignment between student and instructor expectations to improve students' learning and performance. Hence, faculty designing curricula should understand student motives and program expectations [28]. Student motives among geography majors were also studied, finding that undergraduate and graduate students desire that their curricula focus more on career guidance and vocational training [9]. Similarly, in computer science, students value career-relevant training during their undergraduate degree [8] and, in some cases, may not always understand the relevance of many CS courses for their careers [22].

2.2 Academia and Industry Preparation Gap

Industry practitioners report that there is a gap between university graduates' abilities and industry expectations, including both technical and non-technical skills [23]. Recent computer science graduates have been found to undervalue software testing [23, 24], struggle with configuration management systems and other software tools [24], lack relevant project experience [25], and lack experience fixing bugs in, or adding features to, large or poorly documented legacy code bases [5]. For a literature review, see [24].

A number of studies have also shown that students lack social and "soft" skills relevant for success in industry. A study that analyzed the alignment of course syllabi and industry needs noticed a lack of mentioning of soft skills in the syllabi [18]. CS graduates have been found to be underprepared to effectively communicate with co-workers and customers [25] and they fail to reach out for help from colleagues or senior engineers when needed [5]. Moreover, they have a number of fundamental misconceptions about how software engineering work is performed in industry, including the importance of team interactions, although internship experience reduced the frequency of these misconceptions to some extent [27].

Craig et al. summarized prior work on this subject and conducted additional student interviews. They found that CS students were underprepared for work on customer-centered projects of vague and evolving scope, work on projects that would span multiple years, collaboration with larger teams to design complex systems, and use of professional tools and formal practices. Notably, half of CS students in a recent study reported that their CS program did not prepare them adequately for their professional experiences [12].

Caskurlu et al. conducted a study on faculty views on the alignment between CS programs and industry needs. They found many faculty believe CS programs prepare students well for industry jobs, citing the fact that the majority of their graduates have no problems finding high paying jobs. However, faculty also mentioned that soft skills are under-taught and that some topics such as software maintenance are hard, if not impossible, to teach [7].

2.3 Attitudes and Beliefs about CS

A study by Lewis surveyed 13 faculty and 160 undergraduates on their beliefs and attitudes about a variety of CS-related topics, and then compared how students and faculty differed in their views. Topics included things like the importance of group work and the role of aptitude for success. They found that although students and faculty disagree on many topics in general. In some cases, students later in their CS degree began to better align with faculty views than those earlier in their career, but in other cases the disconnect persisted [16]. One such statement rejected by faculty, but more accepted by the students, was "In the real world, computer scientists spend a lot of time working alone." This finding is relevant to our study, as it shows how faculty and students may disagree on basic assumptions about applications of CS, even after interacting with these faculty during their undergraduate education. These findings have been replicated and further explored, focusing on areas where faculty and students disagree [17, 21].

3 Research Questions

While the motivation for our study is primarily the gap between academia and industry, we framed our interviews more broadly. In particular, we included interview questions concerning the goals of an undergraduate CS education in general and the preparation of students for careers in academia. There are two reasons for this choice. The first is that we wanted our interviews to allow us to identify other important aspects of a CS education besides preparation for industry. The second is that we wanted to avoid potentially offending faculty by focusing solely on industry-specific interview questions that could cause them to take a defensive position, compromising our data collection process. Our research questions are:

- **RQ1** What are faculty views on the goals of an undergraduate CS education?
- **RQ2** What are faculty views on the role of an undergraduate CS education in preparing students for industry?
- **RQ3** What are faculty views on better preparing students for careers in industry?

Whereas RQ1 remains more broadly focused, RQs 2 and 3 focus solely on preparation for industry. For reasons of space we omit results related to preparation for careers in academia, only listing some of the most important findings in Sections 5.3 and 6.

4 Methods

In order to understand the range of faculty views regarding the goals of a CS degree, we found a qualitative interview method to be the best fit. A quantitative study using, say, a multiple choice questionnaire would have limited faculty responses to our predetermined questions and answers, likely biasing the results towards our imagination of what faculty views might be. Taking a qualitative approach allowed the participating faculty more freedom to directly and fully express their views in their own words.

Specifically, to both ensure that our research questions were addressed and allow faculty to express their full views, we decided to employ a semi-structured interview format, which prescribes a short list of questions that can be followed up by additional questions that are prompted by the participant's answers. For data analysis, we chose a phenomenographic approach, which seeks to identify the full spectrum of experiences and beliefs of participants, as opposed to finding a single objective truth [19].

4.1 Data Collection

The semi-structured interviews were audio-recorded and then transcribed for data analysis. We first devised an initial set of questions

School Type	Participants	Degrees awarded in 2016 [2]
Research	11	total: 6,477, CS: 470
Undergraduate	1	total: 7,058, CS: 101
Liberal Arts	2	total: 1,463, CS: 9

Table 1: Participant distribution and size of CS programs

followed by a pilot study on four senior PhD students. During this pilot we revised the interview questions until they rendered responses that adequately answered our research questions. Some examples of the resulting questions are: "What does it mean to you for someone to be a Computer Scientist?" and "..., what do you think is the role of an undergraduate CS education in preparing students for industry?". The full set of interview questions is publicly available online [1].

To achieve a high response rate we applied convenience sampling by sending personalized emails to personal connections of the last two authors. Unfortunately, recruitment from institutions other than our own proved challenging. We ultimately conducted interviews at one research-intensive university (ours), one large primarily undergraduate institution, and one small private liberal arts college. The distribution of participants across these schools and the size of their CS programs can be seen in Table 1. Author 1 conducted all the interviews and either author 2 or 3 was taking notes during the first 10 interviews.

We deliberately invited faculty from a wide variety of research areas. Resulting in participants with the following backgrounds: Theory of Computation, Algorithms, Real-Time Scheduling, Cryptography, Cloud Computing, PL, ML, Cyber-Physical Systems, Bioinformatics, Parallel Programming, CER, and Architecture. The seniority of the participants varied greatly, from faculty who were recently hired to faculty with several decades of experience. Of the 14 participants, 5 have some experience working in industry (only 2 as software engineer). Additionally, 1 participant owns a startup and 2 participants collaborated with startups.

4.2 Analysis

Phenomenography is a method used in qualitative research as an alternative to Grounded Theory. Where Grounded Theory allows the researcher to develop a theory grounded in the data, Phenomenography focuses on identifying the full range of understandings or beliefs exhibited by the subjects. An excellent explanation of the phenomenographic process can be found in Simon et al.'s work [26, §1.2]. A more in-depth explanation can be found in Åkerlind's work [3]. An overview of how Grounded Theory and Phenomenography are applied in computing education research can be found in Kinnunen and Simon [13].

In short, Phenomenography allows the researcher to identify all the understandings of and beliefs about a phenomenon and structure them in a hierarchical manner, where the lowest level in the hierarchy maps to the simplest understandings of the phenomenon and the highest level maps to the most complete understandings. The process we applied was derived from the process described in Carbone et al.'s work [6, §2.3] and follows the following steps. For brevity, authors are labeled A1-A5.

4.2.1 Step 1: Individual Labeling A1 applied labels to all interview transcriptions. A2 and A3 both labeled half of the transcriptions. Each author created their own label codes.

4.2.2 Step 2: Extracting Dimensions of Variation Next, a list of summarized beliefs per research question was extracted by collectively analyzing all labeled excerpts from step 1. A "summarized belief" refers to a collection of labeled excerpts that can be viewed as all exhibiting the same belief. Each summarized belief was then mapped to each faculty who exhibited it. From this collection of summarized beliefs, we determined the dimensions of variation in our data.

4.2.3 Step 3: Constructing Categories of Description The summarized beliefs and dimensions of variation from step 2 served as the basis for extracting the hierarchical categories of description. This was done collaboratively with the use of a whiteboard. This process also resulted in subcategories that make it more explicit which belief belongs where.

4.2.4 Step 4: Validation of Categories of Description We validated the output of step 3 by mapping each summarized belief from step 2 to its corresponding subcategory. Whenever there was a disagreement about where a belief should be placed, the names and descriptions of the (sub)categories were clarified. After creating this hierarchy, A2 and A3 labeled a single participant interview and refined the subcategories based on their disagreements. Step 4 resulted in Figures 1 and 2. The categories of description are depicted as the dotted boxes, with the blue boxes being their subcategories.

4.2.5 Step 5: Extracting Quotes The faculty quotes for this paper were extracted by back-tracing from the figures created in step 4, to the summarized beliefs from step 3, which gave us the faculty that exhibited these beliefs. The labels created in step 1 were then used to find the relevant sections in each interview.

5 Results

In this section we will discuss the outcomes of our phenomenographic analysis, as well as some other interesting findings that do not directly relate to our research questions. Wherever a participant is quoted, an anonymous identifier is provided (P1-P14).

5.1 RQ1: Goals of Undergraduate CS Education

The (sub)categories of description we identified during our analysis of faculty beliefs about the goals of an undergraduate CS education are displayed in Figure 1. The hierarchical relationship between the categories is that of dependency or prerequisite, i.e. it is not possible to be a competent problem solver without the required depth of knowledge and hard skills. However, while each subcategory depends on some subcategories in a lower category, it does not necessarily depend on all.

5.1.1 Fundamentals The simplest understanding of the goals of a CS education are the specific skills a student should attain. In this category, we differentiate between Hard Skills, Personal Soft Skills, and Collaborative Soft Skills.

Hard Skills. Examples include being a good programmer, having good software engineering skills, having a strong understanding of how a computer works, and being able to work with data.

Personal Soft Skills. Examples include having good time management skills or perseverance and dedication.

"... we need to get our students to do things they don't like ... to see through even when it's difficult ... perseverance and dedication." - P5

Collaborative Soft Skills. Examples include socializing and translating between human language, formal language and code.

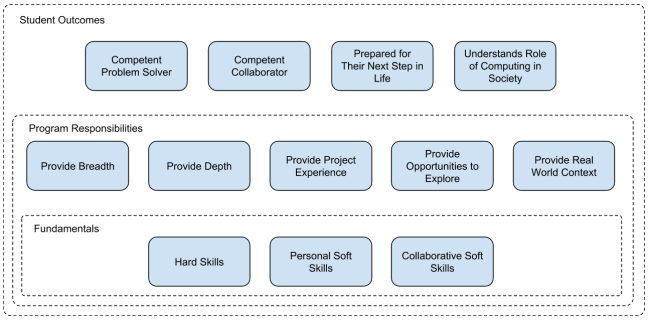

Figure 1: The Goals of Undergraduate CS Education: Categories of Description and Their Respective Subcategories.

"... be able to clearly communicate and translate between formal, precise language, and human oriented language." - P5

5.1.2 Program Responsibilities A more sophisticated understanding of the goals of CS are the responsibilities of the CS program.
Provide Breadth. Faculty believe the program should allow students to develop a broad knowledge base as this will help students approach problems in a more sophisticated way.
Provide Depth. Furthermore, the program should allow students to develop some area of specialization within CS.
Provide Project Experience. Faculty believe providing project experience is essential for students to learn to collaborate.
Provide Opportunities to Explore. Faculty mentioned how every student is a unique individual with their own strengths, weaknesses, and interests. The program should offer opportunities that will allow for each of these unique individuals to excel in their own way and find out what interests them.
"People discover their potential at various points in life ... you can give them multiple paths ... so taking advanced courses, graduate courses, doing a project, doing independent study." - P1
Provide Real World Context. It is important to study computing in the context of the world. The program should teach students what types of problems are good to approach computationally, how computing has changed the world in the past, and how computing itself has evolved over time.
"... what does it mean to be a computational problem? What kinds of characteristics do these problems have in common?" - P14

5.1.3 Student Outcomes We identified student outcomes as the most complete understanding of the goals of CS. This category applies when high level outcomes are mentioned, which will of course depend on the program, as well as the fundamentals.
Competent Problem Solver. Faculty mentioned that becoming a good problem solver was a major student outcome. Some even considered this to be a generic non-CS specific skill, that becoming a good problem solver is the goal of any undergraduate education.
Competent Collaborator. Often noted by faculty was the fact that outside of the classroom, people tend to solve problems in teams. So a major outcome of CS programs should be that students are able to contribute to such a collaborative environment.
Prepared for Their Next Step in Life. University education is seen by faculty as preparation for a person's next step in life. Hence, one major outcome of a CS education is that a student is reasonably sure what their next step will be, is well prepared for it, and able to gain the position they desire.
"... to prepare themselves for that next step, be it going into industry, starting their own company, going off to grad school ... the goal is to help students be successful in whatever that pursuit is." - P9
Understands Role of Computing in Society. Some faculty were adamant about the need for students to be able to relate their work to society. Specifically, students should understand the impact their work might have on society, as well as its ethical implications.

5.2 RQ2: Preparing Students for Industry

The categories of description we identified during our analysis of faculty beliefs about the role of an undergraduate CS education in preparing students for industry are displayed in Figure 2. The quotes in this section are all responses to questions that asked specifically about students headed for industry.

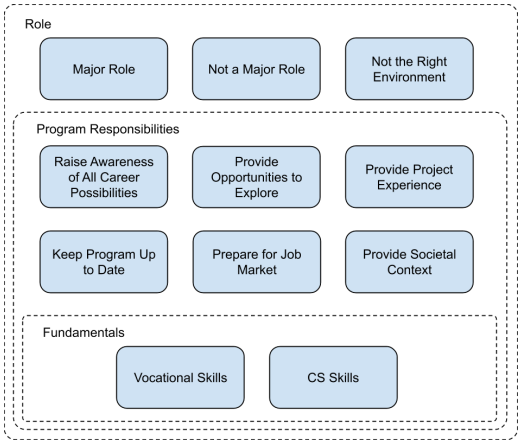

Figure 2: The Role of an Undergraduate CS Education in Preparing Students for Industry: Categories of Description.

5.2.1 Fundamentals Being successful in industry requires a minimal set of fundamental skills. In this category, faculty mentioned some vocational skills as well as fundamental CS skills.
Vocational Skills. Vocational skills mentioned by faculty include being fluent in software engineering principles, familiar with the tools, and able to collaborate and communicate well.
CS Skills. An example of a CS skill that faculty believe is important for industry is the ability to learn a new technology or algorithm quickly by leveraging a strong fundamental understanding of CS.

5.2.2 Program Responsibilities A CS program has specific responsibilities in preparing students for industry.
Raise Awareness of All Career Possibilities. Faculty believe that a CS program should raise awareness on industry career options and what knowledge is necessary for those careers.
Provide Opportunities to Explore. Along with raising awareness, there should be possibilities available to try things out. This can be an internship, co-op program, or working with a professor.
Provide Project Experience. There should be at least one significant project in the curriculum. Students should gain experience with large code bases that are being worked on by multiple people.

Keep Program Up to Date. Faculty say they have a responsibility to keep the program up to date and relevant. This includes listening to feedback from students returning from internships and alumni.
Prepare for Job Market. Faculty also state we should keep in mind that our students will have to enter the job market and interview with companies. Part of our program's responsibility is to ensure that our students will be able to pass interviews and find a job.
Provide Societal Context. CS is not an isolated field, it has a major role in society. Our programs should make students aware of the role they play in society and how their work can influence the lives of other people in both positive and negative ways.
"I think people should be aware when they take a job in industry, why that company is focusing on what they're focusing on, and that it's aligned to a problem that is a real problem, and that doesn't have sort of negative consequences for somebody, or if it does, that student goes in well aware of that. So I think helping students understand that it's not just about writing code ... but it's about evaluating the overall big picture as well." - P14

5.2.3 Role The highest level of understanding on the role of CS education in preparing students for industry is how that role should manifest itself.
Major Role. Some faculty believe industry preparation should be a major role of a CS program, perhaps even the most important.
Not a Major Role. Other faculty believe industry preparation is a responsibility of the students themselves or even of the companies they will work for in the future.
Not the Right Environment. Another group of faculty believe it is simply unrealistic to expect universities to provide industry preparation, because they are not well equipped for it.
"... faculty are not suited for preparing them for the workplace. They never worked, and very few worked outside the campus ..." - P1

5.3 RQ3: Better Preparing Students

We found that when faculty mentioned what they believed to be goals of a CS degree or the role of a CS degree in preparing students for academia or industry, those same things were usually brought up when asked what CS degrees could do better. Below we list our most notable findings on what faculty believe CS programs can do to improve in the preparation for academia and industry.
Reduce Class Sizes. One of the most widely mentioned problems was that class sizes have become too big. Issues resulting from large classes mentioned were: not being able to go as deep into the material because too many students wouldn't be able to keep up, grade inflation, and the quality of group projects suffering. To increase the quality of undergraduate CS programs, faculty recommended reducing class sizes or providing more resources.
"I would love to [better prepare for ind.], but I don't know how because it would require way more resources, and smaller classes" - P14
Increase the Number of Projects. Another widely mentioned recommendation was to increase the number of project courses and make them more authentic, e.g. working with a real customer or solving real-world problems. Interestingly, some faculty advocated for more vaguely defined problems, whereas other faculty claimed that having more project courses with well-defined problems would provide a more authentic preparation for industry.
"In industry projects there are often times where the requirements are not well defined ..." - P12
"Less open-ended projects. My experience with industry is that projects are usually pretty well-defined, especially for junior folks." - P7

Some faculty also mentioned that it is difficult to create an authentic experience, even in a project class.
"Maybe what they're most lacking is things like real experience on large project teams, and I don't think team projects in undergraduate classes really give you a sense of what it's like to work on a larger ongoing project. Maybe there's no way they could." - P3
The User Perspective. In industry, software is always shipped to a user. However, most classes don't teach how that works.
"There's not a sense across the curriculum that we're always talking how to ... put something out there for people to use ... [e.g.] a desktop app that runs on different operating systems or ... a web app." - P2
More Research Opportunities. There is a call from faculty to scale up research opportunities for undergraduates by creating more programs that make it easy for faculty to work with them.
Software Engineering. Another concern faculty have is that students may not be well prepared for the level of rigor required of software engineers in industry.
"They need to have more software engineering skills 'cause the industry is a lot more rigorous about that stuff than academia is." - P7

Furthermore, students should be given more opportunities to familiarize themselves with important industry standard tools.
"... more experience with the tools that you would use in industry that are kind of standard. Like version control software, or different build systems." - P12

While faculty may be willing to address these deficiencies, it is difficult to implement these changes without increasing the workload on the students or faculty or cutting back on course materials.
"And then in terms of like code quality ... I can't find the time in my classes to really teach it or to really get them to do it because I'm putting so much technical challenge on them, like, "You're going to solve this really hard problem, and then solve this other really hard problem." And then I don't have the time to check up on whether they've really thought about all the different edge cases, or whether their code looks good ... So I think we'd need to cut way, way back on the amount of content we're throwing at them, and have them focus on some of these other either more open-ended issues, or communication issues, or impact issues, or quality issues." - P14
Side Projects. Faculty mentioned the value of students having side projects in being better prepared for industry.
"That's often a distinction between strong and weak students, if they really love programming and like doing it in their spare time." - P3
Communication. Many faculty mentioned communication and other soft skills are not well addressed in the curriculum.
Inspire Students. Faculty are concerned we may focus too much on teaching content and forget to make students excited about CS.
Societal Context. Similarly, we often forget to put CS in the context of society and think critically about our work.
Curriculum is Already Adequate. While most faculty see many ways to improve our degrees, some faculty mostly mentioned things that were going well such as students securing good jobs.
"My feeling is that we do a great job of preparation and if they're feeling unprepared, then maybe they didn't do all their work as an undergrad or didn't achieve very well or something of that sort." - P9

Everything Needs to Be Done Differently. On the other end of the spectrum, a few faculty believed the way we teach our CS programs is fundamentally flawed.
"in many classes, attendance is below 50%. Which means, is that our undergraduates are paying for the product, but are not taking it, which is very unusual when you are a customer and paying ... Which means that our education is inferior and has serious, serious problems ... My response would be to change educational system. To ... ban traditional classroom lectures." - P10

6 Discussion and Take-Aways

This section summarizes our key findings.

Industry preparation is important. Contrary to our initial hypothesis that industry preparation might be a controversial programmatic goal, we found general agreement among the faculty in support of industry career preparation. Although Phenomenography does not offer quantitative conclusions, we note that the vast majority of our participants recognize that industry is the next step for most students and view preparing them for their industry careers as a key goal. Part of their motivation is simply to see their students succeed. However, as we discuss below, they also understand that CS plays a critical role in the making of the modern world and much of the impact students have on society comes from what they do after being hired by companies.

Two Curriculum Drivers: The Role of CS in the World and Student Choice. Beyond the belief that learning CS fundamentals is an essential goal of CS education, we've identified two additional organizing principles.

The first, as stated above, is that faculty understand that CS plays a critical role in the making of the modern world. This perceived role of computing in society drives several beliefs. Faculty believe that they should prepare their students to be leaders who can make decisions with a regard for the social and ethical implications of their work. Thus, faculty believe their courses should include ways for students to practice with real-world implications. In turn, faculty value project courses, authentic assignments, and devoting more time to code quality, testing, and industry standard tools. Additionally, they understand that real-world problems tend to be solved by teams of people. As such, they understand that "soft" skills are important, even while acknowledging that they are under-supported in their curriculum.

The second organizing principle that faculty expressed is that every student is unique. Hence, beyond having a common base of CS fundamentals, students should be allowed to create their own path through the curriculum by means of opportunities, including conducting research with a professor, side projects, internships, electives, graduate courses, etc. Furthermore, faculty advocate for creating more of these types of opportunities for students.

The Academia-Industry Gap May be a Resource Gap. The interviews reveal some possible explanations for why the academia-industry gap has persisted over the years in spite of the perceived importance of industry preparation and a significant body of research on the topic.

For one, faculty expressed concern over how the recent meteoric rise in enrollments has impacted the quality of education they are able to offer, specifically citing difficulties in achieving depth, grade inflation, and the difficulty of managing group projects. While previous enrollment booms have waned, the present one seems unlikely to subside. We would note that the gap has persisted even during periods of low enrollment, but the challenges related to enrollments may now be an additional barrier for change.

Second, a few faculty confided that they do not believe preparation for industry is a principal role of an undergraduate CS education. While this group is seemingly small, they no doubt have some influence on curriculum design.

Third, and perhaps most impactful, is the observation that, despite the fact that faculty cited interest in helping students prepare for careers in industry, many may not be sufficiently equipped to do so. Most of the faculty we interviewed never worked in industry and some told us to discount their views due to lack of experience: *"I've never worked in industry, so take that all with a grain of salt."* - P14. However, alumni boards and teaching assistants with industry experience were identified as possible resources for faculty to gain an industry perspective.

Finally, faculty cited struggling with finding ways to provide authentic experiences. One reason for this struggle is class size, but another is that it remains unclear for faculty how to make experiences such as projects and programming assignments more authentic without unacceptably increasing the workload of the students or the instructional team.

7 Limitations and Threats to Validity

This study is subject to the following threats and limitations. Participants were recruited through the personal networks of the last two authors; this may have biased some participants' answers due to their relationships. We believe this threat to be limited by the fact that only the first three authors were present during the interviews. On the other hand, we attempted to limit bias regarding our research questions by framing them about all goals, not just, say, industry preparation. Additionally, the participants were drawn from three institutions, however most participants were from the same research-intensive institution. Moreover, all three institutions are located in the same (US) metropolitan area. Both potentially limit the generalizability of the results and warrant replication to other institutions and regions. Finally, while Phenomenography identifies the range of understandings of a phenomenon, it, combined with the study's scale, did not clearly determine their prevalence. Since we identified conflicting beliefs—e.g., both opposition and support of industry as a major role of CS education—it will be valuable for future work to establish the relative support for such views.

8 Conclusion

We performed a phenomenographic analysis to identify the range of views faculty have on the role of undergraduate CS education, focused on student preparation for industry. We categorized these findings in this work while identifying multiple key themes relevant to remedying the gap between academia and industry: faculty generally want to prepare their students well for industry and want the program to address their career needs, however lack of instructional resources and, in some cases, industry experience are barriers to facilitating this preparation.

9 Acknowledgements

The authors thank the participating faculty for making time for us in their busy schedules. This work was supported in part by NSF award DUE-1712508.

References

[1] 2019. Interview Questions. (accessed Oct. 2019). https://paperdata.page.link/faculty-interview-questions

[2] 2019. University Data – Data USA. (accessed Oct. 2019). https://datausa.io

[3] Gerlese S. Åkerlind. 2005. Variation and commonality in phenomenographic research methods. *Higher Education Research & Development* 24, 4 (2005), 321–334. https://doi.org/10.1080/07294360500284672 arXiv:https://doi.org/10.1080/07294360500284672

[4] Amnah Alshahrani, Isla Ross, and Murray I. Wood. 2018. Using Social Cognitive Career Theory to Understand Why Students Choose to Study Computer Science. In *Proceedings of the 2018 ACM Conference on International Computing Education Research (ICER '18)*. ACM, New York, NY, USA, 205–214. https://doi.org/10.1145/3230977.3230994

[5] Andrew Begel and Beth Simon. 2008. Struggles of New College Graduates in Their First Software Development Job. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education (SIGCSE '08)*. ACM, New York, NY, USA, 226–230. https://doi.org/10.1145/1352135.1352218

[6] Angela Carbone, Linda Mannila, and Sue Fitzgerald. 2007. Computer science and IT teachers' conceptions of successful and unsuccessful teaching: A phenomenographic study. *Computer Science Education* 17, 4 (2007), 275–299. https://doi.org/10.1080/08993400701706586 arXiv:https://doi.org/10.1080/08993400701706586

[7] Secil Caskurlu, Iryna Ashby, and Marisa Exter. 2017. The Alignment Between Formal Education and Software Design Professionals' Needs in Industry: Faculty Perception. In *2017 ASEE Annual Conference & Exposition*. ASEE Conferences, Columbus, Ohio. https://peer.asee.org/28941.

[8] Michelle Craig, Phill Conrad, Dylan Lynch, Natasha Lee, and Laura Anthony. 2018. Listening to Early Career Software Developers. *J. Comput. Sci. Coll.* 33, 4 (April 2018), 138–149. http://dl.acm.org/citation.cfm?id=3199572.3199591

[9] Sharon Gedye, Elizabeth Fender, and Brian Chalkley. 2004. Students' Undergraduate Expectations and Post-graduation Experiences of the Value of a Degree. *Journal of Geography in Higher Education* 28, 3 (2004), 381–396. https://doi.org/10.1080/0309826042000286956

[10] C. Richard G. Helps, Robert B. Jackson, and Marshall B. Romney. 2005. Student Expectations of Computing Majors. In *Proceedings of the 6th Conference on Information Technology Education (SIGITE '05)*. ACM, New York, NY, USA, 101–106. https://doi.org/10.1145/1095714.1095739

[11] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM, New York, NY, USA. https://doi.org/10.1145/2534860

[12] Amanpreet Kapoor and Christina Gardner-McCune. 2019. Understanding CS Undergraduate Students' Professional Development Through the Lens of Internship Experiences. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 852–858. https://doi.org/10.1145/3287324.3287408

[13] Päivi Kinnunen and Beth Simon. 2012. Phenomenography and grounded theory as research methods in computing education research field. *Computer Science Education* 22, 2 (2012), 199–218. https://doi.org/10.1080/08993408.2012.692928 arXiv:https://doi.org/10.1080/08993408.2012.692928

[14] Wolfgang Lehmann. 2009. University as vocational education: working-class students' expectations for university. *British Journal of Sociology of Education* 30, 2 (2009), 137–149. https://doi.org/10.1080/01425690802700164

[15] Timothy C. Lethbridge. 1999. The Relevance of Software Education: A Survey and Some Recommendations. *Ann. Softw. Eng.* 6, 1-4 (April 1999), 91–110. https://doi.org/10.1023/A:1018917700997

[16] Clayton Lewis. 2007. Attitudes and Beliefs About Computer Science Among Students and Faculty. *SIGCSE Bull.* 39, 2 (June 2007), 37–41. https://doi.org/10.1145/1272848.1272880

[17] Clayton Lewis, Michele H. Jackson, and William M. Waite. 2010. Student and Faculty Attitudes and Beliefs About Computer Science. *Commun. ACM* 53, 5 (May 2010), 78–85. https://doi.org/10.1145/1735223.1735244

[18] Marcia A. Mardis, Jinxuan Ma, Faye R. Jones, Chandrahasa R. Ambavarapu, Heather M. Kelleher, Laura I. Spears, and Charles R. McClure. 2018. Assessing alignment between information technology educational opportunities, professional requirements, and industry demands. *Education and Information Technologies* 23, 4 (01 Jul 2018), 1547–1584. https://doi.org/10.1007/s10639-017-9678-y

[19] Ference Marton. 1981. Phenomenography — Describing conceptions of the world around us. *Instructional Science* 10, 2 (jul 1981), 177–200. https://doi.org/10.1007/bf00132516

[20] C. Norton and T. Martini. 2017. Perceived Benefits of an Undergraduate Degree. *The Canadian Journal for the Scholarship of Teaching and Learning* (2017).

[21] Jacob Perrenet. 2009. Differences in Beliefs and Attitudes About Computer Science Among Students and Faculty of the Bachelor Program. In *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '09)*. ACM, New York, NY, USA, 129–133. https://doi.org/10.1145/1562877.1562920

[22] Anne-Kathrin Peters, Anders Berglund, Anna Eckerdal, and Arnold Pears. 2015. Second Year Computer Science and IT Students' Experience of Participation in the Discipline. In *Proceedings of the 15th Koli Calling Conference on Computing Education Research (Koli Calling '15)*. ACM, New York, NY, USA, 68–76. https://doi.org/10.1145/2828959.2828962

[23] R. Pham, S. Kiesling, L. Singer, and K. Schneider. 2016. Onboarding inexperienced developers: struggles and perceptions regarding automated testing. *Software Quality Journal* 25, 4 (2016), 1239–1268. https://doi.org/10.1007/s11219-016-9333-7

[24] Alex Radermacher and Gursimran Walia. 2013. Gaps Between Industry Expectations and the Abilities of Graduates. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. ACM, New York, NY, USA, 525–530. https://doi.org/10.1145/2445196.2445351

[25] Alex Radermacher, Gursimran Walia, and Dean Knudson. 2014. Investigating the Skill Gap Between Graduating Students and Industry Expectations. In *Companion Proceedings of the 36th International Conference on Software Engineering (ICSE Companion 2014)*. ACM, New York, NY, USA, 291–300. https://doi.org/10.1145/2591062.2591159

[26] Simon, Michael de Raadt, Ken Sutton, and Anne Venables. 2006. The Distinctive Role of Lab Practical Classes in Computing Education. In *Proceedings of the 6th Baltic Sea Conference on Computing Education Research: Koli Calling 2006 (Baltic Sea '06)*. ACM, New York, NY, USA, 54–60. https://doi.org/10.1145/1315803.1315814

[27] Leigh Ann Sudol and Ciera Jaspan. 2010. Analyzing the Strength of Undergraduate Misconceptions About Software Engineering. In *Proceedings of the Sixth International Workshop on Computing Education Research (ICER '10)*. ACM, New York, NY, USA, 31–40. https://doi.org/10.1145/1839594.1839601

[28] Anna Wilson, Susan Howitt, Pam Roberts, Gerlese ÃĚkerlind, and Kate Wilson. 2013. Connecting expectations and experiences of students in a research-immersive degree. *Studies in Higher Education* 38, 10 (2013), 1562–1576. https://doi.org/10.1080/03075079.2011.633163 arXiv:https://doi.org/10.1080/03075079.2011.633163