Does Curricular Complexity in Computer Science Influence the Representation of Women CS Graduates?

Albert Lionelle
Khoury College of Computer Sciences
Northeastern University
Boston, MA, USA
a.lionelle@northeastern.edu

McKenna Quam
Khoury College of Computer Sciences
Northeastern University
Boston, MA, USA
quam.m@northeastern.edu

Carla Brodley
Center for Inclusive Computing
Northeastern University
Boston, MA, USA
c.brodley@northeastern.edu

Catherine Gill
Center for Inclusive Computing
Northeastern University
Boston, MA, USA
c.gill@northeastern.edu

ABSTRACT

Not all degree programs are created equal. Indeed, the structure, prerequisites and overall complexity of some programs create barriers that impede student success. Inspired by the methodology of previous papers investigating the inverse relationship between curricular complexity and program quality, in this paper we investigate the relationship between curricular complexity and the representation of women earning CS degrees. We created curricular maps of 60 computer science degrees and calculated measures such as program complexity, course blocking, delay factor, and total math/CS credits to understand complexity's correlation with the representation of women CS majors. Our results show that degree complexity, blocking factor, and delay factor are all inversely related to the representation of women. In addition, we present the courses that most commonly impede student progress and provide suggestions to enhance degree programs based on the insights gained.

CCS CONCEPTS

• **Social and professional topics** → **Computing education programs**; *Women*.

KEYWORDS

Computing education, curriculum design, retention, BPC

ACM Reference Format:
Albert Lionelle, McKenna Quam, Carla Brodley, and Catherine Gill. 2024. Does Curricular Complexity in Computer Science Influence the Representation of Women CS Graduates?. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024), March 20–23, 2024, Portland, OR, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3626252.3630835

SIGCSE 2024, March 20–23, 2024, Portland, OR, USA

ACM ISBN 979-8-4007-0423-9/24/03…$15.00
https://doi.org/10.1145/3626252.3630835

1 INTRODUCTION

As computing educators, our goal is to ensure that all undergraduate students can discover, thrive, and persist in the major to on-time graduation. For the past two decades, educators have focused on understanding the barriers to this goal, particularly for students from populations historically marginalized in tech.[1] Previous efforts to understand (and mitigate) gaps in representation include understanding the lack of community [11, 14, 16, 23, 26–28], social stigma [30, 40], and institutional barriers such as not addressing the distribution of prior coding experience in the introductory sequence [6–8, 13, 39, 43]. In this paper, we examine how curricular complexity can act as a structural barrier to students progressing through the requirements of a computing degree, and we find that degree complexity is inversely related to the representation of women computing majors.

Our analysis utilizes metrics developed by the growing field of *curricular analytics*, which facilitates the study of degree structure and how it impacts student success. Using open source tools, researchers can compare degrees across programs [18], and there is a growing body of research on how degree complexity impacts retention and student performance. [18–21, 33–37, 42].

We are interested in the role that curricular complexity plays in the gender gap of computing [40, 44]. To that end, we examined the Computer Science (CS) major degree plans at 60 universities. Using a plan-of-study obtained from a university's public-facing website, we are able translate a school's curriculum to a directed acyclic graph, from which we can then compute quantitative metrics for each course and the overall curriculum. To determine the relationship between gender diversity and degree complexity, we segmented the schools by their representation of women computing graduates into three groups: high (above 20%),[2] average (between 20% and 15%), and low (15% or below) and found that schools with lower curricular complexity have a higher representation of women graduates. We identified the similarities and differences among the 60 programs, looking at which courses have the highest blocking factor, the mean number of CS and math credits, and the

[1] Historically marginalized populations in computing include women, people with disabilities, and Black/African American, Hispanic/LatinX, American Indian/Alaska Native, Native Hawaiian/other Pacific Islander people.
[2] Women comprised 21.5% of U.S. computer science graduates in 2021 [38].

minimum/maximum delay to graduation. From the analysis, we form several concrete actionable recommendations to help schools reduce unnecessary curricular complexity in their degrees.

In the remainder of this paper, we first review existing research on curricular complexity in Section 2. In Section 3, we describe the process of curricular analytics and common measures of curricular complexity. In Section 4, we describe our experimental methodology, how we selected the 60 universities, and our assumptions in determining their degree plans. Section 5 presents our results including threats to their validity. In Section 6, we discuss the takeaways for computing departments and our conclusions in Section 7.

2 RELATED WORK

While the discipline of curricular analytics is new, a number of works have explored eliminating unneeded barriers and reducing time towards graduation. Johnson [22] proposed three policies to reduce time-to-degree: limiting credit creep, establishing clear degree maps with proven course sequences, and guaranteeing the transfer of courses within general education. While all are considered best practices, they are not universally adopted, which means that curriculum can still be a limiting factor for student achievement.

Klingbeil and Bourne [25] modified the prerequisite structure to make calculus an optional requirement for first year engineering. This had a positive affect on graduation rates and GPAs among students from historically marginalized populations.

Wigdhal et al. [42] proposed using a directed graph to compare degree programs based on Slim et al.'s [36] curricular network analysis. The set of authors for both of these papers together examined total credits, required courses, delay, bottle necks, and pass/fail rates. Using degree plan and course prerequisite data, Heileman et al. [20] built degree maps using directed graphs to compare delay factor, blocking factor, centrality, and complexity of programs. They argue that these four measures can be comparable metrics to guide program reform and simplification [18]. Slim et al. [34] compared course sequences and GPA. Students who had more complex semesters often had more delays due to failures and lower GPAs, whereas students who balanced out their course sequences had higher GPAs. They found an direct relationship between semester complexity and success and retention. In later work, Slim et al. [37] applied a Markov Decision Model to predict graduation rates demonstrating that more complex semesters have higher dropout rates. In follow up work, Slim et al. [33] applied a Bayesian Network model of curricular maps to predict student success, showing that curricular complexity has a direct impact on GPA. Heileman et al. [19–21] mapped complexity to program rankings. While acknowledging rankings are problematic [10, 15, 17], their results show that higher ranked EE and CS programs are less complex. They concluded that lower-ranked programs are relying on curricular complexity to justify/enforce the preparation of their students.

3 CURRICULAR ANALYTICS

Curricular analytics is the study of academic program design and layout. Figure 1 shows an example of a degree map generated by the Curricular Analytics website.[3] The degree map is built by listing the courses required for graduation, and then connecting the prerequisites for each course. From the resulting graph, we can calculate four measures of complexity.

A course's **Blocking factor** measures the extent to which the course blocks a student's ability to take other courses in the curriculum, and is defined by the number of courses reachable from the course node in the map. In Figure 1, the blocking factor for Data Structures (DS) is 11 because there are 11 courses requiring DS.

The **Delay factor** of a course measures the maximum number of courses that must be completed in sequential order. We can use delay to identify the magnitude of the challenge a student might face if they do not take a class early on in their time at university or have to retake said class. Because delay factor does not account for the availability of courses, including terms, enrollment caps, and other "real world" factors, it should not be thought of as the number of "semesters to graduate." For Figure 1, the delay factor for DS is six because the longest chain that requires DS is CS 0, CS 1, DS, Software Development, Software Engineering, and a CS Elective, meaning these six courses must be taken sequentially.

Centrality is the sum of the delay of all the paths containing that course. If a course has no pre or post-requisites, its centrality is zero. Courses with high centrality are usually those that departments see as essential to the major. In Figure 1, DS has a centrality measure of 39 which is the sum of the delay factors of the eight different paths in the curricular map (4, 4, 4, 5, 5, 5, 6, 6).

The **Structural Complexity** of a course is a measure of the overall complexity rating for a course, which is calculated as an unweighted linear combination of it's delay and blocking factors. In Figure 1, the complexity of DS is 17 (6+11). This measure does not evaluate course content, but rather how the course's placement in the program impacts the overall program complexity.

Overall program complexity is defined as the sum of the individual course complexities. For the plan in Figure 1 it is 175.

When comparing programs, it is important to look not only at overall program complexity, but also at the individual course measures, each of which provides insight into program design and how "late" into their post-secondary experience students can discover computer science and complete their degree within the standard four years. Imagine a scenario in which a student discovers computing in their second year. A highly complex degree would likely discourage them from pursuing computing and, in particular, a high delay factor could make it impossible for them to graduate on time. This hypothetical situation is a reality for many students from populations historically marginalized in CS who are more likely come to university without prior experience in computing and therefore discover CS later on in their time at university.

4 EXPERIMENTAL METHODOLOGY

Our methodology follows that of Heileman et al. [21], in which they randomly sampled 60 schools, partitioned them into three groups of 20 schools based on their rankings, and ran an ANOVA test to compare differences among group mean structural complexity. In our approach, we created groups based on the representation of women as measured by program graduation rates.

[3] https://www.curricularanalytics.org/

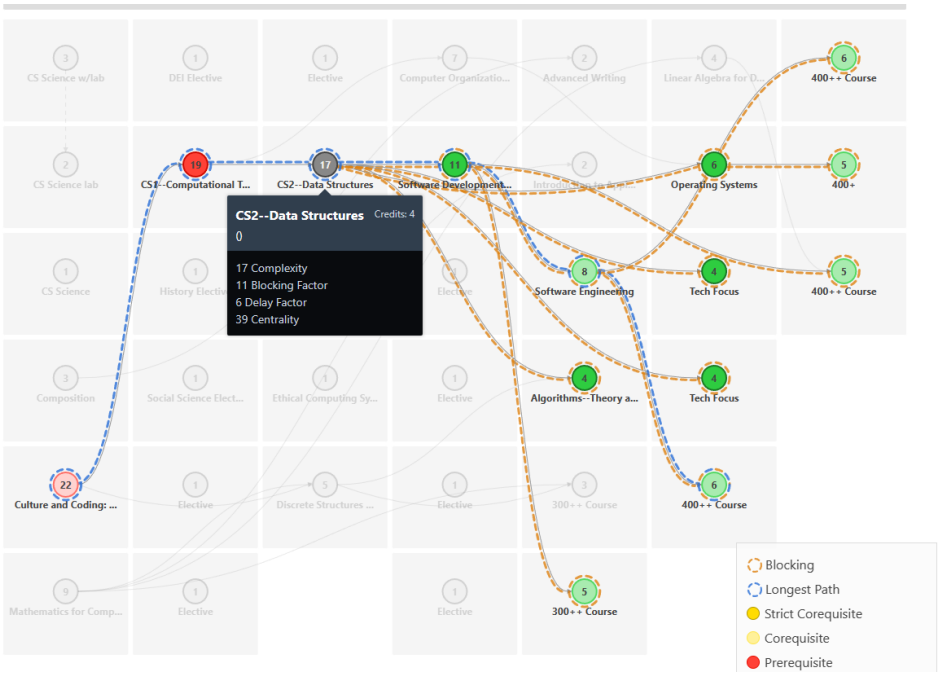

Figure 1: Example Degree Map generated using Curricular Analytics [1].

4.1 School Selection

We collected data from the National Center for Education Statistics' Integrated Postsecondary Education Data System (IPEDS) [38]. We identified all Title 4 degree-granting schools in the United States that have student bodies of 1000 or more students and graduated at least 75 students in computer science CIP-codes (11.0701 and 11.0101) in 2021, resulting in a list of 200 schools. This list was then sorted by percent of women computer science graduates in 2021.[4] However, percent women can be a misleading number when comparing programs, because some programs have more women relative to the university overall population of women. For example, consider two universities in which women represent 19% of CIP 11 graduates in a certain year. If the first has a representation of women across all majors of 25% and the second 70%, then we consider the problem of a lack of representation of women in CS in the second university to be greater. Given this, we normalized the representation of CIP 11 women graduates relative to the overall population at the university by scaling the numbers to the theoretical 50/50 split using the following formula:

$$\frac{Percent\ Women\ CIP\ 11\ Graduates}{Percent\ Women\ Graduates\ Overall} = \frac{x}{50\%}$$

Using the examples above, the first university would have a normalized representation of women of 38% and the second of 14%.

Using the normalized version, we randomly sampled 20 universities from each of three groups: schools with 20% of more women in CS (Group 1); schools with 15-20% women (Group 2); and schools with less than 15% women (Group 3).

[4]IPEDS only provides a binary representation, and the authors note the inherent bias.

4.2 Creating Degree Maps

To create the degree maps, we used publicly available plans of study accessed from each university's website, and information for each class on its pre- and/or co-requisites. Typically, a university's website presents one sample plan of study along with a list of requirements for the major. On average, building a degree map took two hours, but could take much longer due to poor website layouts. If the online material could not be accessed, we omitted that university from our study; this resulted in three universities being omitted from the original sample. Additionally, we removed two universities because they did not require general education credits, making it hard to compare their curriculum to other schools in the sample. These five universities were replaced with randomly selected programs from the original lists to ensure equal sample sizes among the three groups.

For each university, the degree map we created is an "optimized" four-year plan. The degree map includes all required courses for the CS major and general graduation requirements (distribution ("gen-ed") requirements and overall credit requirements). For each required upper-division CS elective, we selected a particular course with a bias toward classes without additional pre-requisites (e.g., we rarely chose computer graphics as often there are additional math prerequisites). Because the complexity of a degree varies based on which CS electives are selected, and our focus was to analyze the lowest complexity CS degree plan for that university, we chose the CS electives that led to the "quickest path to graduation." Most plans of study assume that a student is calculus ready, and thus we made this assumption for all universities to create fair comparisons. We further assumed that the student did not have any AP credits. We discuss possible threats to our results by making these assumptions in Section 5.4.

Table 1: Curricular complexity for three groupings of universities based on their representation of women graduating with a CS degree.

	% women	Mean Complexity	Median Complexity	Median Longest Delay Factor	ABET Accredited
Group 1	n >20%	162.7 ± 12.8	150	5	9
Group 2	20% >= n >15%	218.9 ± 13.1	214	6	11
Group 3	15% >= n	219.6 ± 14.4	214	6	13

4.3 Analysis

For each university we computed: the structural complexity, the blocking of four core classes, the longest delay to graduation, the total number of CS credits required, and the number of math, CS, and total credits required. We observed no consensus in how schools listed their credit-to-hour ratio (most universities counted one course as three to four credits but some schools used a one-to-one or one-to-eight ratio). As such, we normalized total credits for all CS and math courses in a plan to 120 credits. Normalizing allowed the total number of credits to have meaning when comparing programs, such that three credits is roughly three weekly lecture hours. We then performed an ANOVA test to check for significance differences among the groups in structural complexity, and normalized CS credits and math credits.

5 RESULTS

Our primary inquiry is whether there is a relationship between the structural complexity of CS curricula and representation of women (as measured by the percentage of women 2021 CIP 11 graduates per IPEDS). We measured whether or not curricular complexity and total number of required CS credits were significantly correlated with the representation of women in the program. We also explored the blocking factor for the most common courses.

5.1 Curricular Complexity

We observed that schools with higher representation of women have lower structural complexity. To determine if the difference is statistically significant we ran an ANOVA to compare groups [12]. The result was statistically significant (F-Statistic of 6.3203 and p value of 0.003); *higher curricular complexity is correlated with lower representation of women.* Figure 2 shows the degree complexity of the three groups as box plots, where the box shows the central 50% of the data, with the bottom of the box being the lower quartile (the lower 25% of the central 50%), the top being the upper quartile (the upper 25% of the central 50%) and the line in the "middle" being the median. The whiskers show the range of the data with the highest whisker being the upper quartile plus the interquartile range (the upper quartile minus the lower quartile) and the lowest whisker being the lower quartile minus the interquartile range. Anything else outside of that range is labeled as an outlier and displayed on the graph with a dot outside the box. The dot inside each box shows the mean value, which is also reported in Column 3 of Table 1.

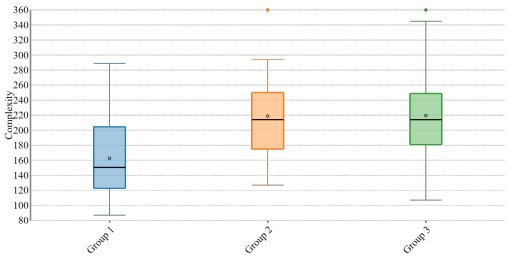

Figure 2: Degree complexity among groups.

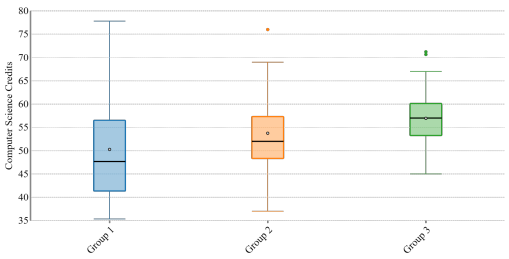

Figure 3: Required CS credits by group.

Table 1 shows the mean and median structural complexity and the median longest delay, which is the median length of the longest prerequisite chain for each group. The maximum delay was eight, which we observed in five of the 60 schools. Regardless of whether a school is on a semester or quarter schedule, a shorter delay is beneficial to on-time graduation, allowing students to discover CS later in college and still graduate on time.

5.2 Total CS Credits and Total Math Credits

Figures 3 and 4 show box plots for total number of required CS credits and math credits. For CS, the mean number of credits is 50, 53 and 57 for Groups 1, 2 and 3 respectively – these do not include the math requirements. The difference between groups was not statistically significant (F-Statistic of 2.225 and p value of 0.117). The mean required number of CS credits across all schools is 53.5 ± 10.22 credits, with a median of 53. CS programs with higher gender diversity (Group 1) had greater variability in CS credits including multiple options and pathways, which we discuss in Section 6.

The difference in the total required math credits was not statistically significant across groups, however, there was large variability within each group. The median required number of math credits across all schools was 14 and the mean was 13.31 ± 3.9. Most programs required the same set of core courses: {Calc 1, CS 1, CS 2, Discrete Math/Structures, Software Engineering, Algorithms, Data Structures}. Programs varied widely on whether or not Statistics and/or Linear Algebra were included, but 54 of the 60 required at

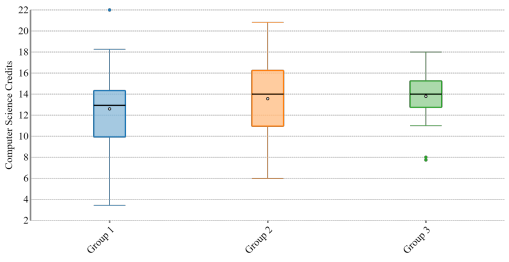

Figure 4: Required math credits by group.

Table 2: Blocking factor for introductory courses.

Course	Group	Mean	Median
CS 1	1	12.6 ± 3.0	13
	2	14.7 ± 3.4	14
	3	16.1 ± 3.2	15
CS 2	1	9.4 ± 3.9	10
	2	12.4 ± 3.6	12
	3	12.4 ± 3.6	12
Calc 1	1	10.2 ± 8.3	9
	2	14.2 ± 6.4	15
	3	13.7 ± 9.0	13
Discrete	1	7.6 ± 4.7	7
	2	8.3 ± 4.1	8
	3	7.9 ± 4.6	8

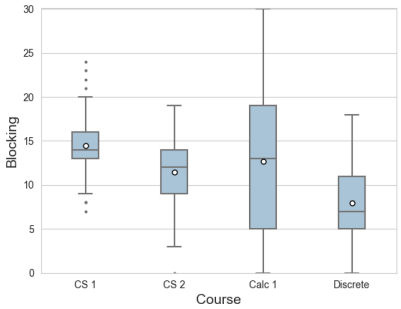

Figure 6: Blocking for intro courses across all 60 schools.

least one of the two. Every school included Calc 1 in their curriculum and 50 of the 60 schools required Calc 2. Shown in Table 1 there was not a significant difference between ABET and non-ABET schools. Thus we cannot conclude from this experiement that ABET requirements impact complexity.

5.3 Blocking Courses

Blocking courses are those that prevent progression in the degree. For every degree studied, the top *blocking courses* include a subset of {CS 1, CS 2, Calc 1, Discrete Structures}. There was a statistically significant difference among Groups 1, 2 and 3 in their blocking factor with $p < 0.05$ at $p = 0.0004$ and $p = 0.02$ for CS 1 and CS 2, respectively. Table 2 shares the mean, standard deviation, and median for each course calculated for each of the three groups. The blocking factor of CS 1 will necessarily be larger than CS 2, because CS 1 is required for CS 2. Figure 5 shows a box plot of the blocking factor for each group for CS1.

Figure 6 looks at the distribution of the blocking factor over all 60 schools for each of the four courses. We can see that for all groups CS 1 has the highest blocking factor of all courses (which is to be expected) and that Calc 1's blocking factor has the highest variability across schools. Calc 1 has a high blocking factor but a much larger standard deviation as some schools have it as a pre- or co-requistite to CS 1, and the majority have it as a pre- or co-requisite to discrete math: 28 schools had Calc 1 as a prerequisite for one of the two, and 14 had Calc 1 as a prerequisite for both.

5.4 Possible Threats to Validity

We address three potential concerns to the validity of our results. First, we note that higher-ranked schools have a greater percentage of women graduating; nine of the 20 schools in Group 1 are

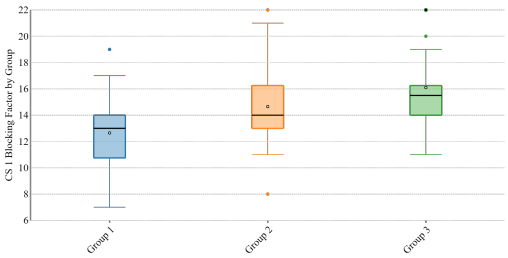

Figure 5: Blocking factor for CS 1 by group.

top twenty schools as measured by US News CS Rankings [31]. However, the remaining schools in Group 1 have a mean rank of 83.6 ± 48.7, and a median rank of 72. A correlation test across all 60 programs between school rank and percentage of women produced a -0.36 correlation coefficient indicating little correlation. While it is likely that schools with higher prestige attract and admit more women to their CS programs, the variation in rankings shows that it is possible for all schools to increase the representation of women if they reduce institutional barriers, such as curricular complexity.

Second, we constructed each degree plan with four assumptions: 1) students come to university in their first year; 2) they are calculus ready, 3) they chose electives with the fewest prerequisites and 4) they come with zero AP credits. The first three assumptions lead to a best case scenario with respect to time to graduation and complexity. The fourth is a non-best case scenario, which we chose because there is no consistency as to how universities treat AP credits and many US public high schools do not have the resources to offer AP classes. In the real world, the complexity of a degree will clearly vary from student to student. However, making these assumptions across all 60 schools allows us to compare across schools.

Third, when we grouped schools without normalization and ran an ANOVA on curricular complexity, we found the same inverse relationship between curricular complexity and representation of women ($p = 0.005$). However, this did not explicitly account for the differences among schools in the overall representation of women (some have 65+%). Context matters and yet the CS education research is not consistent in how to account for a population's representation in computing degrees relative to their representation in the university. The forthcoming paper by Barr et al attempts to create this standard [5].

6 RECOMMENDATIONS

Comparing the different degree maps, commonalities emerged that point to how programs can evaluate and make changes to their degree. We provide five recommendations to consider.

Minimize delay factor for transfer students. Recent studies have found that close to 50% of the student population at state universities are internal or external transfer students [32]. Very few end up graduating in 2+2 years. Indeed, students' number one concern when transferring is delay to graduation [24]. Examining the delay factor can help reduce course sequences acting as barriers, ensuring there are suitable opportunities for transfer students.

Eliminate choke points that prevent student progress. Curricular graphs make it possible to quickly identify "choke points" in a degree plan. A course with high delay, blocking, and centrality would end up preventing students from progressing through a program with a single failure. We recommend reducing choke points by offering pathways, such as those demonstrated by some programs who had algorithms, software engineering, and systems pathways throughout their degree. This means a student struggling in one pathway can continue to progress in another pathway, rather than feeling like their only option is to leave the major entirely.

Offer flexibility for when calculus must be completed. Calculus I has the greatest range of variability of blocking; depending on the school it could either have the largest blocking factor or block nothing at all. In other words, there is marked lack of consensus among universities with regard to the placement of calculus within the CS curriculum. However, if a degree places Calculus 1 before CS 1 in an effort to ensure students have "mathematical maturity", Calculus 1 then becomes a barrier to the entire CS curriculum. In their previous reports, the ACM curriculum committee does not recommend Calculus 1 as a prerequisite for CS 1 or for Discrete Structures [2–4, 41]. In the draft of the 2023 report,[5] the ACM recommendations emphasise the growing dependence on mathematics with topics like machine learning gaining more prominence, but does not recommend Calculus 1 as a prerequisite for CS 1 or Discrete Structures. Instead, they recommend that Pre-Calculus be the prerequisite for Discrete Structures and that there be no math prerequisite for CS 1 [29]. Our data supports the stance that Calculus 1 need not be a prerequisite for lower-level CS courses.

Require a small set of core requirements to increase flexibility. We observe that programs that made their entire set of lower division courses the prerequisite for any single upper division course have extremely high complexity and often low representation of women. In contrast, programs with higher representation of women had significantly greater flexibility. For Group 1, core CS courses[6] made up 26% of the total credits required whereas, for Group 3, they make up 33% of total credits. Group 1 also had more instances of permitting flexibility within the core by allowing one of multiple courses to fill a requirement. Group 1 is characterized with more CS upper division electives with minimal prerequisites (often only Data Structures). In these programs, if an additional requirement for a particular course was needed (e.g., Linear Algebra), that is where it appears in the prerequisite chain.

How course requirements and plans of study are communicated matters. When trained computer science advisors (e.g., one of the authors) struggle to build a degree plan from online resources, students will most likely give up before they have a sensible plan. Most plans of study are based on the assumptions that all students are calculus ready, and that students know where to find information relating to their field of study. In reality, this information is often in multiple locations and presents conflicting statements. Poor website design creates burdens for advisors and students, and having only one sample plan of study can imply that only students who fit that one path should be in the major. A best practice is to have multiple, easy-to-find degree plans, each with a different starting point. A program will be more welcoming if it shows that students can start with different backgrounds and still be successful in completing the degree.

[5]As reported on August 14, 2023.
[6]Any CS class that is explicitly required was counted as core.

7 CONCLUSIONS AND FUTURE WORK

In this paper we explored the relationship of representation of women and curricular complexity for 60 CS degree programs. We found that there is a relationship between the two, with less complex programs having higher representation of women graduates. Additionally, programs with higher representation of women have lower blocking factors and delays. We found that the number of math credits was similar among programs, but how those math credits impacted blocking and delay differed.

In this paper, we did not address curricular complexity against intersectional identity. How to accurately sample and compare schools when diverse populations are influenced by regional demographics, resource distribution, and cultural nuances requires greater research. Promising avenues for future exploration include the development of robust school sampling methodologies addressing the intricacies of often limited intersectional representation.

There are a handful of universities that offer interdisciplinary computing degrees (often referred to as CS+X). For example, Northeastern University offers 43 interdisciplinary computing majors and University of Illinois Urbana-Champaign (UIUC) offers 14 interdisciplinary computing degrees [9]. Because we used the graduation rates for IPEDS CIP-codes 11.01 and 11.07, our results are limited to within-discipline CS majors. At both Northeastern and UIUC, some interdisciplinary computing majors have greater representation of women than within-discipline majors and future research would need to analyze the curricular complexity of such degrees and the respective representation levels of women.

A measure missing from the curricular analytics tool is a measure of flexibility and options. Using a single degree map, it is difficult to measure student options. A promising area of future research could be to expand these tools to give an idea of the strength of flexibility and options in degree programs, and how attractive options are for students from different backgrounds and intersectional identities.

Overall, the results and recommendations presented here are meant to go in conjunction with previous work focusing on best practices for increasing diversity. Curricular analytics is an important method to help departments reduce unneeded barriers for all students. Just as the discipline of computer science is evolving, so too should the field's analysis of curricular design.

ACKNOWLEDGMENTS

This project was partially funded by the CIC at Northeastern University. Any opinions, findings and conclusions or recommendations expressed are those of the authors and do not necessarily reflect the views of the CIC or partner institutions. We would like to thank Neel Patel, Felix Muzny, and Heather Lionelle for their contributions toward editing and reviewing.

REFERENCES

[1] [n. d.]. Curricular Analytics. https://www.curricularanalytics.org/

[2] 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science.* ACM, Inc. https://doi.org/10.1145/2534860

[3] William F. Atchison, Samuel D. Conte, John W. Hamblen, Thomas E. Hull, Thomas A. Keenan, William B. Kehl, Edward J. McCluskey, Silvio O. Navarro, Werner C. Rheinboldt, Earl J. Schweppe, et al. 1968. Curriculum 68: Recommendations for academic programs in computer science: A report of the ACM curriculum committee on computer science. *Commun. ACM* 11, 3 (1968), 151–197.

[4] Richard H. Austing, Bruce H. Barnes, Della T. Bonnette, Gerald L. Engel, and Gordon Stokes. 1979. Curriculum'78: Recommendations for the undergraduate program in computer science — A report of the ACM curriculum committee on computer science. *Commun. ACM* 22, 3 (1979), 147–166.

[5] Valerie Barr, Carla E. Brodley, and Manuel Pérez-Quiñones. To appear in 2024. Visualizing Progress in Broadening Participation in Computing: The Value of Context. In *Communications of the ACM.*

[6] Carla E. Brodley. 2022. Expanding the Pipeline: Addressing the distribution of prior experience in CS1 - CRN. https://cra.org/crn/2022/06/expanding-the-pipeline-addressing-the-distribution-of-prior-experience-in-cs1/

[7] Carla E. Brodley. 2022. Why universities must resist GPA-based enrollment caps in the face of surging enrollments. *Commun. ACM* 65, 8 (Aug 2022), 20–22. https://doi.org/10.1145/3544547

[8] Carla E. 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 *2021 Res. Equity Sustain. Particip. Eng. Comput. Technol. RESPECT 2021 - Conf. Proc.* https://doi.org/10.1109/RESPECT51740.2021.9620552

[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 *SIGCSE 2022 - Proc. 53rd ACM Tech. Symp. Comput. Sci. Educ.*, Vol. 1. ACM, 544–550. https://doi.org/10.1145/3478431.3499352

[10] John Byrne. 2018. How much attention should you pay to U.S. News' College Rankings? *Forbes* (Sep 2018). https://www.forbes.com/sites/poetsandquants/2018/09/10/how-much-attention-should-you-pay-to-u-s-news-college-rankings/?sh=320d4c667daf

[11] Tanya L. Crenshaw, Heather Metcalf, Erin Wolf Chambers, and Umesh Thakkar. 2008. A case study of retention practices at the University of Illinois at Urbana-Champaign. *SIGCSE'08 - Proc. 39th ACM Tech. Symp. Comput. Sci. Educ.* (2008), 412–416. https://doi.org/10.1145/1352135.1352276

[12] Antonio Cuevas, Manuel Febrero, and Ricardo Fraiman. 2004. An ANOVA test for functional data. *Computational Statistics & Data Analysis* 47 (8 2004), 111–122. Issue 1. https://doi.org/10.1016/j.csda.2003.10.021

[13] Wendy DuBow, Joanna Weidler-Lewis, and Alexis Kaminsky. 2016. Multiple factors converge to influence women's persistence in computing: A qualitative analysis of persisters and nonpersisters. In *2016 Res. Equity Sustain. Particip. Eng. Comput. Technol.* 1–7. https://doi.org/10.1109/RESPECT.2016.7836161

[14] Wendy M. DuBow, Beth A. Quinn, Rosario Robinson, Gloria Childress Townsend, and Valerie Barr. 2016. Efforts to make computer science more inclusive of women. *ACM Inroads* 7, 4 (Nov 2016), 74–80. https://doi.org/10.1145/2998500

[15] Robert Farrington. 2022. How much should you trust college rankings? *Forbes* (Sep 2022). https://www.forbes.com/sites/robertfarrington/2022/09/29/how-much-should-you-trust-college-rankings/?sh=c6cd8917f01a

[16] Grant E. Gardner, Jennifer H. Forrester, Penny Shumaker Jeffrey, Miriam Ferzli, and Damian Shea. 2015. Authentic science research opportunities: How do undergraduate students begin integration into a science community of practice? *J. Coll. Sci. Teach.* 44, 4 (2015), 61–65. http://www.jstor.org/stable/43631866

[17] Malcom Gladwell. 2011. The Order of Things What college rankings really tell us. *New Yorker* (Feb 2011). https://www.newyorker.com/magazine/2011/02/14/the-order-of-things

[18] Gregory L. Heileman, Chaouki T. Abdallah, Ahmad Slim, and Michael Hickman. 2018. Curricular analytics: A framework for quantifying the impact of curricular reforms and pedagogical innovations. (Nov 2018). arXiv:1811.09676 https://arxiv.org/abs/1811.09676v1

[19] Gregory L. Heileman, Hayden W. Free, Johnny Flynn, Camden Mackowiak, Jerzy W. Jaromczyk, and Chaouki T. Abdallah. 2020. Curricular complexity versus quality of computer science programs. (Jun 2020). arXiv:2006.06761 https://arxiv.org/abs/2006.06761v1

[20] Gregory L. Heileman, Ahmad Slim, Michael Hickman, and Chaouki T. Abdallah. 2017. Characterizing the complexity of curricular patterns in engineering programs. In *ASEE Annu. Conf. Expo. Conf. Proc.*, Vol. 2017-June. American Society for Engineering Education. https://doi.org/10.18260/1-2--28029

[21] Gregory L. Heileman, William G. Thompson-Arjona, Orhan Abar, and Hayden W. Free. 2019. Does curricular complexity imply program quality? *ASEE Annu. Conf. Expo. Conf. Proc.* (Jun 2019). https://doi.org/10.18260/1-2--32677

[22] Nate Johnson. 2011. Three policies to reduce time to degree. *Complet. Coll. Am.* (2011).

[23] Philip M. Johnson, Carleton Moore, Peter Leong, and Seungoh Paek. 2022. Improving engagement, diversity, and retention in computer science with RadGrad: Results of a case study. *ACM Trans. Comput. Educ. 1, 1, Artic.* 1, 1 (2022), 27. https://doi.org/10.1145/3418301

[24] Amir Karimi, Randall D. Manteufel, and Lynn L. Peterson. 2015. What delays student graduation? *ASEE Annu. Conf. Expo. Conf. Proc.* 122nd ASEE, 122nd ASEE Annual Conference and Exposition: Making Value for Society (2015). https://doi.org/10.18260/p.25055

[25] Nathan Klingbeil and Anthony Bourne. 2015. The Wright State model for engineering mathematics education: Longitudinal impact on initially underprepared students. (Jul 2015), 26.1580.1–26.1580.11. https://doi.org/10.18260/P.24917

[26] Nancy Kober. 2015. Reaching students: What research says about effective instruction in undergraduate science and engineering. *Reach. Students What Res. Says About Eff. Instr. Undergrad. Sci. Eng.* (Feb 2015), 1–240. https://doi.org/10.17226/18687

[27] George D. Kuh, Jillian Kinzie, John H. Schuh, and Elizabeth J. Whitt. 2011. *Student success in college: Creating conditions that matter.* John Wiley & Sons.

[28] Linda K. Lau. 2003. Institutional factors affecting student retention. *Education* 124, 1 (2003).

[29] The Joint Task Force on Computing Curricula Association for Computing Machinery, IEEE-Computer Society, and Association for Advancement of Artificial Intelligence. 2023. CS2023 version beta. https://csed.acm.org/cs2023-beta/

[30] Oluwafemi Osho, Bart P. Knijnenburg, Eileen Kraemer, Cazembe Kennedy, Gloria Washington, Stacey Sexton, John Porter, and Kinnis Gosha. 2023. Societal factors that impact retention and graduation of underrepresented computer science undergraduates. *SIGCSE 2023 - Proc. 54th ACM Tech. Symp. Comput. Sci. Educ.* 2 (Mar 2023), 1399. https://doi.org/10.1145/3545947.3576343

[31] US News A World Report. [n. d.]. Best Undergraduate Computer Science Programs Rankings. https://www.usnews.com/best-colleges/rankings/computer-science-overall

[32] Isabel Sam. 2021. *University Transfer Students' Challenges and Graduation Rates - What Educational Instutitions Need to Know.* Ph. D. Dissertation. California State Polytechnic University.

[33] Ahmad Slim, Gregory L. Heileman, Chaouki T. Abdallah, Ameer Slim, and Najem Sirhan. 2021. *Restructuring curricular patterns using Bayesian networks.* Technical Report. Online.

[34] Ahmad Slim, Gregory L. Heileman, Wisam Al-Doroubi, and Chaouki T. Abdallah. 2016. The impact of course enrollment sequences on student success. *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA* 2016-May (may 2016), 59–65. https://doi.org/10.1109/AINA.2016.140

[35] Ahmad Slim, Jarred Kozlick, Gregory L. Heileman, and Chaouki T. Abdallah. 2014. The complexity of university curricula according to course cruciality. *Proc. - 2014 8th Int. Conf. Complex, Intell. Softw. Intensive Syst. CISIS 2014* (Oct 2014), 242–248. https://doi.org/10.1109/CISIS.2014.34

[36] Ahmad Slim, Jarred Kozlick, Gregory L. Heileman, Jeff Wigdahl, and Chaouki T. Abdallah. 2014. Network analysis of university courses. *WWW 2014 Companion - Proc. 23rd Int. Conf. World Wide Web* (Apr 2014), 713–718. https://doi.org/10.1145/2567948.2579360

[37] Ahmad Slim, Husain Al Yusuf, Nadine Abbas, Chaouki T Abdallah, Gregory L Heileman, and Ameer Slim. 2021. A markov decision processes modeling for curricular analytics. In *2021 20th IEEE Int. Conf. Mach. Learn. Appl.* 415–421. https://doi.org/10.1109/ICMLA52953.2021.00071

[38] National Center for Education Statistics. 2021. IPEDS : Integrated Postsecondary Education Data System.

[39] Chris Stephenson, Alison Derbenwick Miller, Christine Alvarado, Lecia Barker, Valerie Barr, Tracy Camp, Colleen Lewis, Erin Cannon, Mindell Lee Limbird, Debra Richardson, Mehran Sahami, Elsa Villa, Henry Walker, and Stuart Zweben. 2018. Retention in Computer Science Undergraduate Programs in the U.S. Data Challenges and Promising Interventions The Association for Computing Machinery The ACM Education Board Retention Committee. (2018).

[40] Jane Stout and Tracy Camp. 2014. Now what? *ACM SIGCAS Comput. Soc.* 44, 4 (Dec 2014), 5–8. https://doi.org/10.1145/2695577.2695578

[41] Allen B. Tucker. 1991. Computing curricula 1991. *Commun. ACM* 34, 6 (1991), 68–84.

[42] Jeffrey Wigdahl, Gregory L. Heileman, Ahmad Slim, and Chaouki T. Abdallah. 2014. Curricular efficiency: What role does it play in student success? *ASEE Annu. Conf. Expo. Conf. Proc.* (2014). https://doi.org/10.18260/1-2--20235

[43] Chris Wilcox and Albert Lionelle. 2018. Quantifying the benefits of prior programming experience in an introductory computer science course. In *Proc. 49th ACM Tech. Symp. Comput. Sci. Educ.*, Vol. 2018-January. ACM, New York, NY, USA, 80–85. https://doi.org/10.1145/3159450.3159480

[44] Stuart Zweben and Betsy Bizot. 2014. 2013 Taulbee Survey Second Consecutive Year of Record Doctoral Degree Production; Continued Strong Undergraduate CS Enrollment. 26, 5 (2014). http://cra.org/resources/crn-online/